ABSTRACT

Title of Dissertation: DATA-DRIVEN ANALYTICAL MODELS FOR IDENTIFICATION AND PREDICTION OF OPPORTUNITIES AND THREATS

Saurabh Mishra, Doctor of Philosophy

Dissertation directed by: Professor Bilal M. Ayyub, Department of Civil and Environmental Engineering

During the lifecycle of mega engineering projects such as: energy facilities, infrastructure projects, or data centers, executives in charge should consider the potential opportunities and threats that could affect the execution of such projects. These opportunities and threats can arise from different domains; including for example: geopolitical, economic or financial, and can have an impact on different entities, such as, countries, cities or companies. The goal of this research is to provide a new approach to identify and predict opportunities and threats using large and diverse data sets, and ensemble Long-Short Term Memory (LSTM) neural network models to inform domain specific foresights. In addition to predicting the opportunities and threats, this research proposes new techniques to help decision-makers for deduction and reasoning purposes. The proposed models and results provide structured output to inform the executive decision-making process concerning large engineering projects (LEPs). This research proposes new techniques that not only provide reliable time-series predictions but uncertainty quantification to help make more informed decisions. The proposed ensemble framework consists of the following components: first, processed domain knowledge is used to extract a set of entity-domain features; second, structured learning based on Dynamic Time Warping (DTW), to learn similarity between sequences and Hierarchical Clustering Analysis (HCA), is used to determine which features are relevant for a given prediction problem; and finally, an automated decision based on the input and structured learning from the DTW-HCA is used to build a training data-set which is fed into a deep LSTM neural network for time-series predictions. A set of deeper ensemble programs are proposed such as Monte Carlo Simulations and Time Label Assignment to offer a controlled setting for assessing the impact of external shocks and a temporal alert system, respectively. The developed model can be used to inform decision makers about the set of opportunities and threats that their entities and assets face as a result of being engaged in an LEP accounting for epistemic uncertainty.

DATA-DRIVEN ANALYTICAL MODELS FOR IDENTIFICATION AND PREDICTION OF OPPORTUNITIES AND THREATS

by

Saurabh Mishra

Dissertation submitted to the Faculty of the Graduate School of the
University of Maryland, College Park, in partial fulfillment
of the requirements for the degree of
Doctor of Philosophy
2018

Advisory Committee:
Professor Bilal Ayyub, Chair
Professor Roland Rust, Dean's Representative
Professor Aris Christou
Professor Enrique Droguett
Professor Peter Sandborn

Dedication

This work is dedicated to my wife, Delphine and to my parents Subhash and Renuka.

“*Education is the kindling of a flame, not the filling of a vessel.*”

— Socrates

Acknowledgements

I would like to extend my deep appreciation to my Ph.D. advisor and committee for their time, support, and assessment of this research:

Professor Bilal Ayyub, Committee Chair and Advisor.
Professor Roland Rust, Dean's Representative.
Professor Aris Christou.
Professor Enrique Droguett.
Professor Peter Sandborn.

I will remain eternally grateful to Professor Bilal Ayyub for believing in me and guiding me over the last four years.

While working on this dissertation, my research was nourished by my former career as an Economist and the research I conducted then. I would like to thank my former colleagues at the International Monetary Fund, the World Bank and the International Finance Corporation. This work is reflected in Chapter 1.2 and Chapter 4.1 of this dissertation. Many components of these chapters were submitted to refereed journals.

The inspiration for this dissertation came from my longtime interest in the advancement of technology to enhance decision-making in the business and policy communities.

I would like to give special thanks to all the decision-makers from diverse fields and countries I talked to, for sharing their perspectives and experiences with me in order to better understand the ground realities of project failures and their impact on communities. I hope this work will be used to enrich our monitoring, planning and understanding of the future.

Table of Contents

List of Tables

TABLE PAGE

Appendices

List of Figures

FIGURE PAGE

Appendices

List of Abbreviations

Acronym	Description
ACT	Activation function
ANN	Artificial Neural Network
ARIMA	Autoregressive integrated moving-average
ARMA	Autoregressive moving-average
BLSTM	Bi-directional LSTM
BNN	Bayesian Neural Network
BP	Backpropagation
BPTT	Backpropagation through time
CNN	Convolution Neural Networks
CV	Cross Validation
DTW	Dynamic Time Warping
EC	Economic Complexity Algorithm
GRU	Gated Recurrent Unit
HCA	Hierarchal Clustering Analysis
HMM	Hidden Markov Model
KF	Kalman Filter
KNN	K Nearest Neighbor
KAL	Knowledge Abstraction Layer
LASSO	Least absolute shrinkage and selection operator
LSTM	Long-Short Term Memory
MCMC	Markov Chain Monte Carlo
MCS	Monte Carlo Simulation
MLP	Multi-Layer Perceptron
MRM	Model Reliability Matrix
NM	Number of Mentions variable from GDELT
PCA	Principal Component Analysis
PDA-Z	Z-Threshold Algorithm Peak Detection
RBF	Radial Basis Function
RBM	Restricted Boltzmann Machine
RNN	Recurrent Neural Network
RVM	Relevance Vector Machine
SLG	Sequence Learning Gate
SGD	Stochastic Gradient Descent
S(P)	Shannon Entropy
SVM	Support Vector Machines
t-SNE	t-distributed stochastic neighbor embedding
TLA	Time Label Assignment
UQ	Uncertainty Quantification

Chapter 1: Problem Statement

1.1. Monitoring Opportunities and Threats

Large engineering projects or **LEPs,** also known as **mega-projects,** are "endeavors characterized by large investment commitment, vast complexity, especially in organizational terms, and long-lasting impact on the economy, the environment, and society" (Brooks and Lokatelli 2015). Generally, projects requiring over $1 billion in financing are considered mega-projects. The main objective of LEPs is to create long-term economic benefits for the societies and countries, in general. Nevertheless, as the number, complexity, and scope of LEPs increases worldwide, their vast stakes may endanger the survival of corporations and threaten the stability of countries that approach these projects unprepared (Millar et al. 2001).

The challenges facing LEPs stem from a number of sources; firstly, since typically several private and public entities are involved in LEPs, the failure of any these entities could lead to the failure of the project as a whole. For example, the failure of a big bank that finances LEPs or poor financial performance of counter-parties vested in the project can make the project exposed (Hassan et al. 2013). Secondly, poor economic conditions, such as, high sovereign debt or pessimistic growth outlook, can disrupt public-private partnership (PPP) projects, especially in the oil and gas and power-grid sectors (Rao et al. 2014, Platon et al. 2014, Xing and Guan 2017). Similarly, political events, such as, terror attacks, government instability, coups, policy uncertainty, and rigid business regulations, can destabilize LEPs in the construction sector (Kosnik 2005, Deng et al. 2014, Larsen 2017, Coats 2017). Corruption is more likely to endanger LEPs that are unique and complex in nature

(Locatelli et al. 2017).

In order to face these challenges effectively, executives need to implement a risk management plan. **Risk**, in itself, is defined as the potential for events whose impacts and dynamic interactions result in outcomes that are different than the ones anticipated, noting that ISO 31000 defines risk as the effect of uncertainty on objectives. Risk is often described in probabilistic terms, whereas **uncertainty** applies to situations in which potential outcomes and causal forces are not fully understood (Ayyub 2010, Lessard and Miller 2001). Although **threats**, which are part of the risks, are out of our control, their consequences can be mitigated through implementing a risk management plan. **Risk management** is defined as the identification, assessment, and prioritization of risks followed by action to minimize, monitor, and control their impacts to maximize the realization of **opportunities** (Antunes and Gonzalez 2015, Hubbard 2009). However, since LEPs can have an impact on many entities; such as: governments, corporations, national or global populations; therefore, in order to implement a comprehensive risk management plan for LEPs, the different stakeholders and their risk perspectives should be accommodated.

In order to be able to understand the risks associated with LEPs, the sources of the threats and opportunities facing them need to be defined. The threats and opportunities associated with LEPs can be both internal, i.e., organization specific, or external, i.e., not directly related to the organization. Regarding the former, it is estimated that nine out of ten mega-projects go over budget (Flyvbjerg 2014); while an example of the latter is the metro systems in Salvador and Brazil which took a dozen more years to be functional (Kennedy et al. 2014, UN Habitat 2013). On the other hand, there are also growing opportunities for LEPs. The world needs to spend about $57 trillion on infrastructure alone by 2030 to enable the anticipated levels of GDP growth globally (McKinsey 2015); and two-thirds of this investment will be required in developing economies, where there are rising middle classes, population growth, urbanization, and increased economic growth (Garemo et al. 2015).

At the same time, LEPs face a number of uncertainties during their lifecycle; the four primary sources of these uncertainties are: the mission of the project, political and social conditions, economic and financial conditions, and technical conditions related to untested technologies (Bertolini and Salet 2007). For instance, opting to use uncertain technologies will effectively negate the benefits of the solutions for cost overruns in mega-projects such as reference class forecasting (Flyvbjerg 2006, Flyvbjerg 2013) simply because there are too many unknowns (Sommer and Loch 2004). For example, the Dead Sea mega-project was encumbered by ample uncertainties. In the early coverage of the project, uncertainties were dominated by economic feasibility of the project and raised primarily by politicians; while more contemporarily, they were dominated by ecological uncertainties voiced by environmental non-governmental organizations. The strategies most often used to address uncertainties is still 'uncertainty reduction' and to a lesser degree 'project cancellation' (Fischhendler et al. 2013). Uncertainty management is key to the successful execution of mega-projects and are discussed in detail in Sections 1.3 and 2.8.

All the above uncertainties amplify the magnitude of the impacts of LEPs which can be felt across governments, non-governmental agencies, private corporations, and citizens or consumers; or a combination of all of these entities. Even for a country as large as China, analysts had warned that the economic ramifications of an individual mega-project, such as the Three Gorges Dam "could likely hinder the economic viability of the country as a whole" (Salazar 2000); while the nuclear tragedy at Fukushima in Japan impacted the national economy negatively, the cost of decommissioning the plant would rise from the US $690 million per year in the first five years to several billion US dollars per year in preceding years costing the Japanese taxpayers in excess of $100 billion, socio-political disruptions, indirect cost of increased imports of fossil fuels that resulted in deteriorating trade balance (METI 2016, ASME 2016, Nanto et al. 2011, NEA 2017, Ferris and Solís 2013, Harding 2016, McCurry 2017). LEPs can also be economically transformative. Consider the Panama Canal which accounts for a significant share of the country's GDP; and Dubai's international airport which accounts for 21 percent of Dubai's employment and 27 percent of its GDP. Hong Kong would grind to a halt without its clean and speedy subway system, named the MTR, which has enabled the densely packed city to build beyond the downtown districts (McKinsey 2016). Furthermore, LEPs can have direct social impact by influencing the pattern of city growth and the lifestyle of the population. For example, projects such as Atlantic Yards in Brooklyn, New York City or the larger Thames Gateway in London; and Amsterdam South (Zuidas) will have significant impact on the evolution of these cities.

The last risk component for LEP's is accommodating different stakeholders and their risk perspectives. The stakeholders could be a CEO of a company, Department Director, Manager at multi-national corporations (MNC's), executives at international banks, officials at government agencies, sub-contractors, or investment fund advisors. Of interest are the decisions facing executives such as whether to invest in a location (or country) or resource allocation decisions concerning protection of tangible (or intangible) assets from potential external threats. The different risk perspectives entail that the same shock to a mega-project will be viewed differently by different stakeholders. Decision-maker from lending institutions are concerned with impact on their banks' financial performance, the government stakeholders are concerned with geopolitical threats and impact on economic and development outcomes, and executives at companies in proximity to the LEP care about the economic impact on their business.

As a result of the different risk components associated with LEPs, there are various **domains** of opportunities and threats – such as geopolitical, economic, financial, reputation, or climate, hereafter simply referred to as risks that influence the success or lead to the failure of mega-projects. These domain-specific risks impact **entities** that may include companies, countries, or cities. Departments at specific entities are collectively referred to as organizations. More specifically, these domain-specific risks, also called **external risks,** are of interest rather than organization specific internal risks. Such potential external risks can be transferred between entities or mega-projects and play a critical role in assessing risks for mega-projects (Ayyub 2008, Modarres 2007).

Decision-makers are interested to learn how to use monitoring technologies for risk management purposes, as executive-decision makers are confused about how to operate in an ever-more complex and uncertain global environment (WEF 2017, Diehl 2017). Moreover, the availability of large-scale data is adding more noise and complexity that may trigger unintended societal events, indecisiveness for executives, and potential failure to adopt to long-term risks (Strauß 2015, Kenny 2015). The complexities in domain knowledge, computational limits, knowledge fusion, and interdependencies among domains pose a challenge to design a *flexible* risk management framework. As a result, recent technological advances in machine learning (ML) and artificial intelligence (AI) are rapidly transforming business operations and how executives take decisions. Although, many executives are concerned how to best adopt these technologies to aid decision-making. Organizational culture, transparency, and trust are major concerns of the executives regarding these technologies; these new technologies also provide new opportunities to improve risk management and project-related executive decision-making. For example, it is anticipated that executives will have a broad view of new information that did not exist before and economic theory suggests that AI will substantially raise the value of human judgment (Agarwal et al. 2017). **Judgment** is the process of how we work out the benefits and costs of alternative decisions in different situations (Agarwal et al. 2017). It is expected that human judgment will increasingly specialize in weighing the costs and benefits of different decisions, and then that judgment will be combined with machine-generated predictions to make decisions. Hence, cheaper predictions will generate more demand for decision-making to exercise human judgment related to mass movements in finance, economies, political, or technological changes (Agarwal et al. 2017, Ransbotham et al. 2016). These mass movements denote large shifts in domain-specific trends, i.e., short or long-term directional change in market, population, or price movements. Consequently, CEOs and their boards, as examples, need to monitor these aspects very closely and promote use cases for practical applications and validation to enhance the decision-making process (Schrage 2017).

1.2. Identification of Opportunities and Threats

In the risk management framework, risk identification is the first step that organizations take. Traditionally, the process of risk identification requires the participation of the project team, risk management team, subject matter experts from other parts of the organization, customers, end users, stakeholders, and/or outside experts (Ayyub 2012). Conventionally, expert opinion is used for risk identification purposes that can lead to subjective and inaccurate results. In addition, there are a number of other shortcomings associated with this methodology including: the availability of experts from inter-disciplinary fields, the high costs of acquiring experts, and the timeliness associated with such methods to dynamically assess and mitigate the complexities of the global risk landscape. Therefore, this section will provide examples for domain-specific opportunities and threats that shape the risk profile of organizations and mega-projects. A data-driven approach is proposed to automate real-time signals for domain-specific opportunities and threats. The goal is to use multiple information sources to identify and predict opportunities and threats impacting organizations that will provide important feedback to assess the viability of mega-projects.

1.2.1. Opportunities

There are several sources of domain-specific opportunities that can be recognized by the executives in charge of LEPs that could later help in their risk identification process. The advancement in information technology provides opportunities to inform decision-makers on mass-movements. Information technology is leading to a more rapid coalescence of herd behavior i.e. people making decisions "on the basis of their observations of other people's actions" and "people imitating each other's behavior" (Ellis and Fender 2011). The knowledge of such mass movements is embodied in data about prices, political events, and macro-economic shifts that provide important prior information for project related decision-making. On one hand, for public sector decision-makers, these movements can signal allocating resources to high-risk natural disasters areas or counties endemic to poverty and depressed economic development outcomes. On the other hand, for private sector decision-makers, opportunities can involve the sale or lease of any product, service, equipment, business growth etc. based on location intelligence. The knowledge of threats can also be an important signal for opportunities. For example, foresight regarding the onset of a financial crisis would provide opportunities to hedge against risks; large engineering companies can receive government contracts in the after-math of natural or man-made disasters.

Second, are geo-political opportunities that include new defense negotiations, bi-lateral trade agreements, cross-border business deals, and new partnerships between multi-national corporations (MNC's) and governments. Such geopolitical opportunities inform business and policy executives on the appropriate strategy for their respective countries as many new coalitions and partnerships can arise as a response to geopolitical tensions. Moreover, prior knowledge of geo-political threats provides an opportunity to protect assets and possible gains from the foresight. For example, China is seizing the geopolitical opportunities of the melting Arctic for resource-extraction; many investors are shifting assets to emerging markets (EM's) in anticipation of a geo-political crisis in the USA. These signals provide important insights for sectoral projects' feasibility and can help in identifying features, such as: countries, cities, or companies, for successful mega-projects.

The third source of domain-specific opportunities is the economic conditions, which is perhaps the most important source when assessing opportunities for mega-projects. A country that is predicted to grow faster than expected will influence the expansion strategy i.e. which country to invest in. Similarly, knowledge of districts, counties and cities projected to grow faster will help inform within country location strategy. Many other economic features such as: fast growth in sectoral value-added, exports, real-estate prices, or new business growth in a city or country provides key signals for economic opportunities. For instance, one of the most important questions that a decision-maker faces during the planning phase include for example which district or city can they build a new project? Therefore, economic signals on predictions for city specific value-added growth in computer, and information services or employment growth in network engineers, for example, will help inform them about that decision related to a data-center project.

Another important aspect of the economic opportunities is to accommodate the mass movements in economic production, which is referred to in economics as structural transformation. Such economic shifts inform executives about their country's strategy in respect of which services, or goods, are performing better, or expected to; especially in emerging-markets (EM's). Domain-specific knowledge to accommodate structural shifts in services, referred to as the globalization of services. Algorithms can be used to extract new features, such as complex products (or services) and macroeconomic competitiveness of countries will help inform executives about their projects' strategy. For example, the business components that large-scale engineering projects require for their production plants are no longer solely manufacturing "hardware", but also require a host of other "services" to materialize. Nowadays, many countries and companies are switching to sourcing their value creation from manufacturing "hardware" to a host of services. In order to capture such large economic shifts, domain-specific features such as diversity, sophistication, complexity-fitness etc. are necessary to view the set of new opportunities for countries and companies (Loungani et al. 2017, Mishra et al. 2017).

Finally, opportunities in the financial domain are perhaps the most important for model testing and real-world applications. Actions to inform trading strategies are simple binary, i.e. signal to either buy or sell. Financial market signals provide an opportunity to (a) assess a country's risk, (b) inform profit-maximizing trading strategies, and (c) hedge against mass financial movements that can impact the stability of an organization and, in return, LEPs.

1.2.2. Threats

LEPs face a heterogeneous set of external threats. For instance, although oil platforms are technically difficult and are socially desired because of the high revenues they bring to communities and countries, they typically face a number of institutional risks and production concentration risks. Hydroelectric-power plants projects tend to be moderately difficult in engineering terms, but very difficult in terms of social acceptability; nuclear-power plants pose high technical risks but still higher social and institutional risks. In a study conducted by Miller and Lessard (2001) and investigating 60 LEP's across the world, managers were asked to identify and rank the risks they faced in the early front-end period of each project. The results showed that market related risks were the dominant type of risks with 41.7 percent of the mangers ranked them highest, followed by technical risks with 37.8 percent, and institutional/sovereign risks with 20.5 percent. The following sub-sections will discuss the various domains that act as sources for the threats facing LEPs.

First, as discussed earlier, political threats can determine the success or failure of large international projects. The increasingly global nature of markets, competition, and events make projects undertaken, investment portfolios constructed, and strategies enacted susceptible to political and country risks. The resulting complexity and interdependencies are intractable which necessitates the deployment of judgement and opinions in an analytical framework. The commonly used methods can be characterized by an index-based framework including Erb at al. (1996), Bank of America World Information Services, Control Risk Information Services (CRIS), Euromoney, Political Risk Services, International Country Risk Guide

(ICRG)/Coplin-O'Leary Rating System, and Moody's Investor Services. For example, the ICRG provides an underlying framework that is based on the methodology developed by Coplin and O'Leary (1994). It provides a series of risk factors' ratings to summarize the forecasts for each country based on the 17 risk components, broken down into 12 sub-components in the 18-month forecast and five sub-components in the 5-year forecast.

Like political threats, economic threats can have a disruptive impact on public-private partnership (PPP) projects and LEPs in general. Economic factors; such as: economic concentration, local infrastructure, and public debt are important signals to predict the success or failure of a mega project (Rao et al. 2014, Platon et al. 2014, Xing and Guan 2017). Another contributing factor to economic threats is the instability from commodity prices, in general, and energy prices, in particular. The hikes in oil prices in 1973-1974 affected an abundant array of commodity prices and posed serious challenges to countries and corporations. Nevertheless, the impact of oil price shock has declined due to better conduct of monetary policy with transport sector and energy subsidies explaining cross-country variations during 2010-15 (Choi et al. 2017).

Not only do political and economic threats impact an individual country, but such threats can also be transferred between countries. Catastrophic events, regardless of their source, can have massive impacts on regional economies of afflicted areas, and lead to adverse and sometimes favorable impacts on other adjacent or far economies through economic substitutions. Extreme losses can cause ripples through the interdependencies and globally integrated economy. Moreover, understanding the process of economic recovery that is affected by the resilience of the infrastructure and economy is important in managing recovery efforts.

Such catastrophic events impact economies as well as destabilize financial markets and trigger economic crises in a manner that is not well understood due to complexity associated with individual, corporate, and government behavior and actions. For example, the credit crunch of 2008 has caused the biggest economic crises for at least 80 years. Although it started in the United States, the entire global economy was impacted and lost $15 trillion and 7 million jobs as a result (Treasury 2012, Anderson 2012, Ball et al. 2013). Although approaches, such as behavioral economics and complexity science, suggest that business cycles, stock market volatility, and catastrophic collapse are inherent properties of the economy as a complex system (Ayyub 2012), most economists and macroeconomic models failed to predict that threat. This renewed the efforts to enhance modeling and crisis management in order to help financial risk managers and macroeconomic planners to assess the likelihood and severity of future downturns.

At the same time, the financial domain also acts as a source of threats to LEPs. Financial market volatility and predictability are important to assess countries' risks (Hassan et al. 2013) as financial movements provide high frequency proxy for financial volatility (Fornarei et al. 2009, Zhang et al. 2010). And since infrastructure financing instruments and incentives are key for financing international engineering projects, diverse asset classes such as commodity futures, foreign exchange rates, and fixed income securities provide a strong signal for the sectoral developments. Moreover, private sector infrastructure finance hinges on sensible transfer of risks and returns between various counter-parties (OECD 2015a, OECD 2015b). Furthermore, financial market signals can also inform new business opportunities for public companies; like trading strategies to outperform the market and the availability of large government contracts in the aftermath of natural disasters (Mutter 2015).

In addition, within the financial domain, several factors can impact the performance of LEPs. Interest rate is one of these factors. Interest rates can have a significant impact on the costs of financing for a project that, in return, impacts the corporate's cash flows and asset values. For example, interest rates in the United States shot up in 1979 and peaked in 1981, followed by a gradual decline with some fluctuations until the credit crush of 2008 which led to a persistently shrinking economy with unemployment levels at their highest for several years afterwards (Ayyub 2012). The contemporary economic threat arises from the long-run effects of short-run developments and the inability of monetary policy to accomplish much more when interest rates have already reached their lower bound (Summers 2014).

Exchange rates is a similarly an important factor that affect LEPs. An example of exchange rate instability is when the value of the British pound plummeted against the Euro as news related to Brexit was announced. The Brexit has a direct impact on countries far away, such as Bangladesh, as the UK represents the third largest export destination for Bangladesh after the United States and Germany. The depreciation of the pound as well as the euro risks eroding the competitiveness of Bangladeshi exports to these markets (World Bank 2017). An important source for uncertainty in mega-projects is from their cost projections that generally involve multiple foreign investors. Long-term debt is the primary source of financing LEPs with inherent risk of exchange rate fluctuations. For example, energy projects typically generate revenues in local currency, while there financing and fuel costs are denominated in U.S. dollars or other hard currencies. Such exchange rate fluctuations can skew revenue and cost projections for LEPs (Sovacool and Cooper 2013).

Furthermore, many projects depend on the availability of venture capital and the stock performance of public companies, thereby introducing another risk source related to stock market volatility. The stock market collapse in 2001 impacted many corporations' cash flows and led to a failure of a number of LEPs. Finally, other sources of financial threats include the credit risk. These risks are associated with potential defaults on notes or bonds by corporations, including subcontractors; also, credit risks can be associated with market perceptions regarding the likelihood of a company defaulting. All the above could affect a corporation's bonds' rating and ability to raise money and maintain their projects and operations. Consequently, to assess the importance of this domain of threats, Erb et al. (1996) used regression and time series analysis to determine how financial risk measures do contain the most information about future equity returns and country risk measures and are highly correlated with country equity valuation measures.

In conclusion, the wide array of opportunities and threats that face LEPs and the high magnitude of their impacts, provide a major problem for decision makers when they try to identify the risks facing their projects and entities and decide whether to engage in that particular project or not. Therefore, with the current unreliable methodologies to identify these factors, the need for a new data-driven prediction model is of paramount importance for executives, decision makers, and other stakeholders involved with LEPs.

1.3. Uncertainty Quantification and Decision-Making

The mis quantification and mis management of the uncertainties associated with mega-projects is the main reason why these projects fail. As a result, current management philosophy dictates that successful executives are moving away from a world where uncertainties are viewed from a static perspective. In other words, from, uncertainties' assessments during the early project's phases or its execution, to operating in a dynamic environment, where simultaneous management of uncertainties are identified as the successful trait of project managers (Laufer 1997). Moreover, successful project managers build buffers to incorporate redundancy, and isolate and reduce uncertainty in the project's execution (Goldratt 1997). They are the ones who are aware of (a) what is happening in the location of their project; (b) what is happening in the country of their project; and (c) what is happening to the assets related to their project; and are constantly monitoring trends and uncertainties about the external environment which constitutes one-third of the reasons why mega-projects fail (Laufer 1997). The management of these uncertainties is of interest since the goal of this research is to control or reduce the "out of our hand" uncertainty factors to be able to better manage the mega-projects.

In addition, uncertainty is closely tied to intelligence. Intelligence is broadly defined as the ability to perceive information, understand knowledge, analyze uncertainty, and make decisions under conditions of uncertainty and in dynamic environments. The definition of intelligence is applicable to all living systems, and this ability of a living system or machine to make appropriate decisions can be taken as a measure of intelligence (Ayyub 2010). Nevertheless, this decision-making ability requires the processing of data and information, construction of knowledge, and assessment of associated uncertainties and risks. Uncertainty of forecast is of interest to this research, not just the uncertainty surrounding a particular event.

The Knightian distinction between risk and uncertainties is that risk applies to situations that can be measured but we do not know the outcome. In this view uncertainty applies to situations that we cannot know all the information we need in order to set accurate judgements in the first place (Knight 1917). A known risk is "easily converted into an effective certainty", while "true uncertainty", as Knight called it, is "not susceptible to measurement" (Knight 1917). For example, the known risk may be the prediction of economic performance indicator such as GDP growth which in a country, like Egypt, may be projected to be lower than the mean of the previous five years. Uncertainty are the various unaccounted-for factors that we are unaware of; such as: sovereign debt or fiscal policy management, geopolitical unrest, or financial performance of Egypt's largest companies such as Alexandria National Iron and Steel or Orascom Telecom. All these factors (either positive or negative) may lead to bias in informing projected realities in Egypt.

However, the focus of this dissertation is on assessing and forecasting elements of opportunities or threats, and comprehensively assess uncertainty. Unlike, uncertainty, risk (opportunities or threats) is an outcome whose likelihood can be quantitatively estimated. Hence, this research focuses on assessing and predicting the domain-specific knowledge at a future state accompanied with uncertainties quantification associated with that prediction. If there was a way to blend the two approaches, e.g. by exploring the uncertainty surrounding the variables that might provide a substantive basis to assess future opportunities and threats surrounding a given entity-specific outcomes (or predictions).

In project management literature, the Project Management Body of Knowledge or PMBOK Guide Sixth edition (2017), defines Risk "an uncertain event or condition that, if it occurs, has a positive or negative effect on one or more project objectives; such as: scope, schedule, cost, and quality." Essentially, an unexpected event is a risk and it can directly or indirectly impact the execution of the mega-project. For example, a positive risk or an opportunity such as better country's economic performance and better financial performance of the contractor may affect the project execution in a positive manner. On the other hand, negative risk or a threat such as pessimistic financial performance of the contractor and country economic performance may provide a negative signal for the project as a whole that could delay or halt the mega-project. The uncertainty lies in the fact that any of the aforementioned outcomes is completely unknown for tomorrow or the next year as the information concerning the events is not traditionally measured or guessed.

In this wave of thinking, it is important to rely on system science to define the focused upon uncertainty's type of interest. This dissertation provides systems construction for various purposes; such as: predicting and diagnosing the external environment. In every system there exists a relationship among different variables; for example, in order to predict the variable GDP growth for Egypt in the macroeconomic system, there are important relations with the stock index, human capital, economic factors, institutions, etc. that are used to better inform the prediction system. The relation among these various variables can be used to determine the unknown state of a given variable by knowing the state of other variables. When the unknown states are determined uniquely, the systems are called deterministic; otherwise, they are called nondeterministic. However, it is a common observation that nondeterministic systems are far more prevalent than deterministic systems in complex dynamic environments of interest to this research. Each nondeterministic system inevitably involves some uncertainty which is associated with the purpose for which the system has been constructed. In each nondeterministic system, the relevant uncertainty must be properly incorporated into the formal description of the system (Ayyub 2010).

In engineering and scientific design, uncertainty is commonly defined as knowledge incompleteness due to inherent deficiencies in acquired knowledge (Ayyub 2010). It can also be used to characterize the state of a system as being settled or in doubt, such as uncertainty of the outcome. Uncertainty is an important dimension in the analysis of risks. In this case, uncertainty can be present in the definition of the hazard threats and threat scenarios, the asset vulnerabilities, and their magnitudes, failure consequence types and magnitudes, prediction models, underlying assumptions, effectiveness of countermeasures and consequence mitigation strategies, decision metrics, and appropriateness of the decision criteria (Ayyub 2010). Traditionally, uncertainty in risk analysis processes is classified as follows:

1- Aleatory uncertainty (or inherent randomness): Some events and modeling variables are perceived to be inherently random and are treated to be nondeterministic in nature. The uncertainty in this case is attributed to the physical world because it cannot be reduced or eliminated by enhancing the underlying knowledge base; for example, the public perception of the S&P500 market index. In the past, it was hard to capture such uncertainty; nonetheless, nowadays, it is possible to reduce this uncertainty by leveraging the characteristics of the social and public sentiments on the web for the given asset class.

2- Epistemic uncertainty (or subjective uncertainty): In many situations uncertainty is also present as a result of lack of complete knowledge. In this case, the uncertainty magnitude could be reduced as a result of enhancing the state of knowledge by expending resources. However, sometimes, this uncertainty cannot be reduced due to resource limitations, technological infeasibility or sociopolitical constraints. This type of uncertainty is the most dominant type in risk analysis. For example, the prediction of a time series data like a stock price will be the point estimate of that variable; however, this value can be treated as a random variable bounded using probability intervals or percentile ranges. By enhancing our knowledge about this potential political, economic, or financial trend, these ranges can be updated.

In classical statistics, probability bounds can be viewed as a mix of probability theory and interval analysis (Ferson et al. 1999). They have similar bases as interval probabilities and concepts in probabilistic analysis using limited or incomplete information. Probabilities in this case are uncertain, and hence represented by probability bounds. Of interest to this research are the prediction uncertainties; however, a key challenge has been uncertainty quantification in machine learning tasks. For example, lack of domain knowledge which impacts the selection of model structure, regularization, methods, and definition of the “best” model can be a leading cause of epistemic uncertainty in machine learning applications (Starcuzzi 2017). Nevertheless, it still remains very difficult to assess the model form uncertainty in machine learning models (Ling and Templeton 2015).

In order to enhance machine-aided decision-making processes, bringing awareness to epistemic uncertainty as well elements of aleatory uncertainty surrounding accurate and reliable predictions both are crucial. As seen from above discussion on uncertainties, our main focus will be on epistemic uncertainty. Nevertheless, the nature of this research in terms of its use of diverse data sets, and related unsupervised learning models also addresses some attributes of aleatory uncertainty that can inform critical factors in the external environment concerning a LEP.

Chapter 2: State of the Art Models

This chapter provides a comprehensive review of the state-of-the-art models which can be useful to identify and predict opportunities and threats across many domains. The chapter begins by discussing the models' attributes related to feature extraction and the associated algorithms for visualizing the time-dependent relationships between domains and entities including clustering. Second, details of machine learning and other statistical methods to convert stochastic random variables into label-based prediction problems are discussed. Third, the different architectures of neural networks are presented with specific attention to deep neural networks' architectures; such as: RNN's and a variant of RNN's called Long-Shorter Term Memory (LSTM) neural networks. Lastly, a detailed assessment of state of the art methods for uncertainty quantification in deep neural networks is also presented.

Before examining all the available methods, it is important to define their two-most important features i.e. complexity and classification. Regarding the former, the degree of computational complexity is a function of the time or space (memory) required to solve a problem by a given algorithm (Ayyub 2010). These requirements are expressed as a function *f*, of a single parameter, *n*, that represents the size of the problem. This function is called a time (or space) complexity function (Ayyub 2010). The main distinction between algorithms whose complexity function can be expressed in terms of a polynomial is as follows

$$f(n) = a_k n^k + a_{k-1} n^{k-1} + \cdots + a_1 n + a_0 \qquad (2.1)$$

for some positive integer *k*, and algorithms for which *f*(*n*) is expressed by an exponential form, e.g. 2^n, 10^n, etc. where *f*(*n*) is an exponential function of *n*. Polynomial time algorithms are considered efficient (or tractable) while exponential time algorithms are considered inefficient (or intractable) (Garey and Johnson 1979).

Concerning the latter, Machine Learning can be supervised or unsupervised. **Supervised Machine Learning** methods include: regressions, classifications are used to find a rule, an 'equation' that can be used to predict a variable. For instance, a decision-maker may want to look for a momentum (trend following) signal that will have the best ability to predict future market performance by running advanced regression models, **Unsupervised Machine Learning** include: clustering, factor analyses uncovering the structure of data. Figure (2-1) shows a graphic summary and classification of different machine learning and artificial intelligence models. Given the many vastness, variants, complexity, and ever-evolving field of study, and the nature of these networks to be combined together in stack or layers to give better results than one network alone makes consolidating and classifying these shall and deep learning models difficult. Table (2-1) provides a summary of various models grouped by their uses and applications. Similarly, Figure (2-1) provides a graphic illustration of many machine learning and artificial intelligence models in one chart.

However, despite the fact that there are many applications of "shallow learning" methods in the domains of interest, only a handful of the systematic framework for using "deep learning" in domains of economics, finance, and geopolitics exist. These usually take the form of a neural network with few layers, however, when they are stacked in multiple hidden layers or complex architecture, they can be called deep learning. Although all these models help in providing predictions for dynamic problems, the specific case of how these external domain opportunities and threats can be used to reduce the uncertainty related to the mega-project's management and decision-making is a new field that is not well studied in the literature. Hence, there is an imminent need for a prediction model that is capable of incorporating these opportunities and threats to provide a better prediction for the mega-projects and reduce the uncertainties associated with them.

Moreover, in spite of all the advancement and capabilities of the current state-of-the-art models and techniques, it is necessary to ensemble a variety of these models to study and understand complex investment and policy decision-making related to the impact of external opportunities and threats on mega-projects. For instance, the dynamic time warping (DTW) algorithms can be used to learn the similarity of a threat; such as: recessions but cannot be used for sequence prediction of contracting economic growth, like the LSTM. Similarly, algorithms like z-threshold peak detection might help to classify a peak or trough point of unemployment rate in a country, but hierarchical clustering will be able to group different countries together based on similar unemployment trends. Consequently, the use of these techniques in ensemble will help to tackle some pressing real-world challenges. In this vein, a critical component of the decision-making process is not just the knowledge of a point estimate forecast for a point in the future, but also the uncertainty bounds and confidence intervals concerning the prediction estimate.

It is important to note that uncertainty quantification (UQ) in itself should be considered modular sub-component of analysis for decision-making. The system control architecture should be modular to accommodate various data structures, models and visualizations can aid the best possible decision, since it is not one model or model reliability (MR) metrics including UQ, but together that can best aid policy or private investment decision-making. These issues raise other important elements such as human factors, computer graphics, information system monitoring, and control architecture that are referred to in Appendix B. In order to aid decision-making concerning assets related to a mega-project, this work is focused on statistical model building attributes related to new data applications, multi-variate time series neural networks, MR and UQ to track the external environment including forecasts and related analysis concerning assets related to a mega-project. Figure (2-2) provides an abstraction for consolidating multiple data sources, models, and visualization toolkits.

Model	Description	Model Use					Applications	References
		C[1]	T[2]	L[3]	P[4]	S[5]		
K-Means	k-means clustering aims to partition "n" observations into "k" clusters in which each observation belongs to the cluster with the nearest mean, serving as a prototype of the cluster.	✓				✓	Vector quantization, Cluster analysis, feature learning.	MacQueen, 1967; MacKay, 2003; Cai et al 2016; Oyelade et al 2010.
Dynamic Time Warping (DTW)	DTW algorithms are used for measuring similarity between two temporal sequences.	✓					temporal sequences of video, audio, and graphics data, automatic speech recognition, speaker recognition, online signature recognition, shape matching application.	Ratanamahatana and Keogh, 2008; Hayashi, Mizuhara, and Suematsu, 2005.
t-distributed stochastic neighbor embedding (t-SNE)	t-SNE is a machine learning algorithm for dimensionality reduction. It is well-suited for embedding high-dimensional data into a space of two or three dimensions for visualization purposes.	✓					computer security research, music analysis, facial expression recognition, cancer research, visualize high-level representations learned by an artificial neural network.	van der Maaten and Hinton, 2008 ; van der Maaten, 2009 ; van der Maaten and Hinton, 2012 ; van der Maaten, 2014 ; Watternberg, 2016.
Multi-dimensional Scaling (MDS)	MDS is a means of visualizing the similarity of individual data based on the distance matrix of the datasets. It is a form of non-linear dimensionality reduction. It is well-suited for embedding low-dimensional data into a higher dimensional space.	✓					Mix-marketing models, Psychometrics, Cognitive Psychology, Ecological analysis, physics, political science', biology.	Takane, 2006 ; Cha, 2009 ; Borg and Groenen, 2005 ; Shoben, 1983 ; Young, 1984.
Hierarchical Clustering Analysis (HCA)	HCA is a method of cluster analysis which seeks to build a hierarchy of clusters.	✓					Image recognition, Natural Language Programming (NLP), Robotics, Computer	Jianbo Shi and Jitendra Malik, 2012; Cai et al, 2014; Chipman, 2005; Balcan and Gupta, 2010.

					Graphics, data compression, pattern recognition.	
Autoencoders (AE)	AE is an artificial neural network used for unsupervised learning in feature learning. AE learns a representation (encoding) for a set of data, typically for the purpose of dimensionality reduction. AE is also used in learning generative models of data.	✓		✓	Natural Language, Programming (NLP), Dimensionality Reduction, pre-training deep Network, one-class classification, de-noising financial data.	Le, 2015 ; Bao et al, 2017 ; Wang et al, 2012; Lopez-Martin et al, 2017.
Z-threshold peak detection algorithm	The algorithm takes 3 inputs: lag = the lag of the moving window, threshold = the z-score at which the algorithm signals and influence = the influence (between 0 and 1) of new signals on the mean and standard deviation		✓		Online change-point estimation, trajectory generation, peak and trough signal detection	Open Source
Bayesian Change-point Anomaly detection algorithm	Bayesian change-point detection algorithm is used for online inference. It is a simple and exact method for calculating the posterior probability of the current run length.		✓		EEG analysis, DNA segmentation, econometrics, and disease demographics, nuclear magnetic response drilling of a well, stock prices, coal mine disaster	Adams and MacKay, 2014
Naïve Bayes Classifier (NBC)	NBC are used for constructing classifiers: models that assign class labels to problem instances, represented as vectors of feature values, where the class labels are drawn from some finite		✓		review of a movie, book, or product on the web expresses the author's sentiment toward the product, while an	Bayes (1763); Mosteller and Wallace (1964), Russel and Norvig, 1995.

	set. NBV is a family of algorithms based on a common principle.					editorial or political text expresses sentiment toward a candidate or political action.	
Linear Regression	Linear approach for modeling the relationship between a scalar dependent variable y and one or more explanatory variables (or independent variables) denoted x. The case of one explanatory variable is called simple linear regression. For more than one explanatory variable, the process is called multivariate linear regression.			✓		Trend line, epidemiology, economics, finance, environmental science	Ubiquitous
Logistics Regression (Logit)	Logit model is a regression model where the dependent variable (DV) is categorical or just a binary output of 0 or 1.	✓		✓		Ubiquitous	Cox, 1948; Ubiquitous
Support Vector Machines (SVM)	SVM's are supervised learning models that can be used for classification or regression analysis. In addition to performing linear classification, SVMs can efficiently perform a non-linear classification using what is called the kernel trick, implicitly mapping their inputs into high-dimensional feature spaces.		✓	✓		text and hypertext categorization, image segmentation, Gene Expression	Cortes, 1995 ; Ben Hur, et al, 2001 ; Ubiquitous
Ridge Regression and least absolute shrinkage and selection	Ridge regressions are a technique for analyzing multiple regression data that suffer from multicollinearity. performs			✓	✓	Ridge regression are applied in various regression problems across	Chan-Lau 2017; Hoerl 1962; Tibshirani 1996; Brieman 1995; Marquardt 1975.

operator (LASSO)	both variable selection and regularization in order to enhance the prediction accuracy and interpretability of the statistical model it produces				discipline such as medial, finance, and computer LASSO is used for forecasting models when the number of potential covariates is large.	
Vector Autoregression (VAR)	VAR is a stochastic process model used to capture the linear interdependencies among multiple time series.		✓	✓	Macroeconomics, econometrics, time-series forecasting, control theory.	Hamilton 1994; Zelner 1962; Hacker and Hatemi 2008, 2009; Canova and Ciccarelli 2015.
Kalman Filters (KF)	KF uses a series of measurements observed over time, containing statistical noise and other inaccuracies, and produces estimates of unknown variables that tend to be extremely accurate.	✓		✓	guidance, navigation, and control of vehicles, particularly aircraft and spacecraft, trajectory optimization	Kalman, 1960 ; Zarkan, 2000 ; Walpern, 2007 ; Andreasen, 2008.
Particle Filter (PF)	PF also known as Sequential Monte Carlo (SMC) methods are a set of genetic, Monte Carlo algorithms used to solve filtering problems arising in signal processing and Bayesian statistical inference.		✓	✓	signal and image processing, Bayesian inference, machine learning, risk analysis and rare event sampling, engineering and robotics, artificial intelligence, bioinformatics, phylogenetics, computational science, molecular chemistry, computational physics, pharmacokinetic	Del Moral 1996; Lliu and Chen 1998;

Monte Carlo Simulation (MCS)	MCS is a computational algorithm that rely on repeated random sampling to obtain numerical results. Their essential idea is using randomness to solve problems that might be deterministic in principle.			✓	optimization, numerical integration, and generating draws from a probability distribution.	Kroese et al 2014; Hubbar et al 2009; Hastings 1970;
Convolutional Neural Networks (CNN)	CNN is a class of deep, feed-forward ANN that has successfully been applied to analyzing visual imagery. A CNN consists of an input and an output layer, as well as multiple hidden layers. The hidden layers are either convolutional, pooling or fully connected.	✓		✓	object recognition for ImageNet, image and video recognition, recommender systems and natural language processing.	Zhang et al, 2017 ; Matusugu et al, 2013 ; LeCun, 2013 ; Zhang, 1988 ; van den Oord et al, 2013 ; Collobert et al, 2008
Recurrent Neural Networks (RNN)	RNN is a class of ANN where connections between units form a directed cycle. This allows RNN to exhibit dynamic temporal behavior. Unlike feedforward neural networks, RNNs can use their internal memory to process arbitrary sequences of inputs.	✓	✓		unsegmented, connected handwriting recognition, speech recognition, stock prices, keyword spotting	Fernandes et al, 2007 ; Graves et al, 2009 ; Gal et al, 2015 ; Lipton, 2015 Felix and Schmidhuber, 2000;
Deep Belief Networks (DBN)	DBN is a generative graphical model, or alternatively a class of deep neural network, composed of multiple layers of "hidden units" with connections between the layers but not between units within each layer.	✓	✓		natural language understanding, generating and recognizing images, video sequences, motion-capture data, non-linear dimensionality reduction	Sarekas, Hinton, and Deoras, 2014 ; Hinton, Osindero & Teh 2006, Ranzato et. al. 2007, Bengio et.al.. (2007). Sutskever and Hinton. (2007). Taylor et. al. 2007 Hinton and Salakhutdinov,2006; Salakhutdinov and Hinton,2007

Long-Short Term Memory (LSTM) RNN	LSTM is well-suited to classify, process and predict time series given time lags of unknown size and duration between important events.	✓	✓	speech recognition, text-to-speech synthesis, Google Android, Siri, Google Voice Search, machine translation, multilingual language processing	Sak, Senior and Beaufays, 2014 ; Li and Wu, 2014 ; Fan et al, 2015 ; Schmidhuber, 2015 ; Gilick et al, 2015 ; Felix, 2001 ;
Gradient Boosting	Tree boosting is a highly effective and widely used machine learning method. XGBoost is an open-source software library which provides the gradient boosting framework for Python		✓	boosting algorithms are used in many areas of machine learning and statistics beyond regression and classification.	github.com/dmlc/xgboost ; Chen et al 2016; Brieman 1997
Gated Recurrent Units (GRU)	GRU's are a gating mechanism in RNN. They have fewer parameters than LSTM, as they lack an output gate.		✓	polyphonic music modeling, speech signal modeling, emotion classification in noisy speech, any sequence prediction	Dey and Salem, 2017 ; Rana et al, 2016 ; Heck and Salem, 2017
Bi-Directional LSTM	Bidirectional LSTM is a variant of a Hopfield network store associative data as a vector. The bi-directionality comes from passing information through a matrix and its transpose.		✓	Keyword spotting, time-series analysis, speech recognition, handwritten recognition, protein structure prediction	Schuster and Parival, 1997 ; Graves and Schmidhuber, 2005 ; Fernandes et al, 2007 ; Liwicki et al, 2007 ; Baldi, 1999
Reinforcement Learning (RL)	RL is concerned with how software agents ought to take actions in an environment so as to maximize some notion of cumulative reward.	✓	✓	the theory of optimal control, economics, game-theory, dynamic programming technique, operations	van Otterlo and Wiering, 2012; Tokic and Palm 2011; Watkins 1988; Ng and Russel 2000.

					research, information theory, simulation-based optimization, multi-agent systems, swarm intelligence, statistics and genetic algorithms	
Bayesian Neural Networks (BNNs)	A Bayesian neural network is a neural network with a prior distribution on its weights with posterior inference. Here standard NN training via optimization from a probabilistic perspective equivalent to maximum likelihood estimation (MLE) for the weights.	✓	✓	✓	Gene Regulatory Networks, Medicine, Information Retrieval, Image Processing, Spam Filtering, System Biology	Neal 2012, Gal 2015, Courville 2006, Bate et al. 1998, Ghahramani 2016.
Generative Adversarial Networks (GANs)	GANs are a class of artificial intelligence algorithms used in unsupervised machine learning, implemented by a system of two neural networks contesting with each other in a zero-sum game framework.	✓		✓	visualizing industrial design, shoes, bags and clothing items or items for computer games' scenes, Facebook Image Recognition	Goodfellow et al 2014; Karpathy et al 2016; Schawinski 2017.

Table 2-1. Summary of model, description, model uses, applications, and references

1: C – Clustering
2: T – Transfer Random Variable to Classification Problem
3: L – Classification Problem
4: P – Time-Series Prediction Problem
5: S – Inference Problem

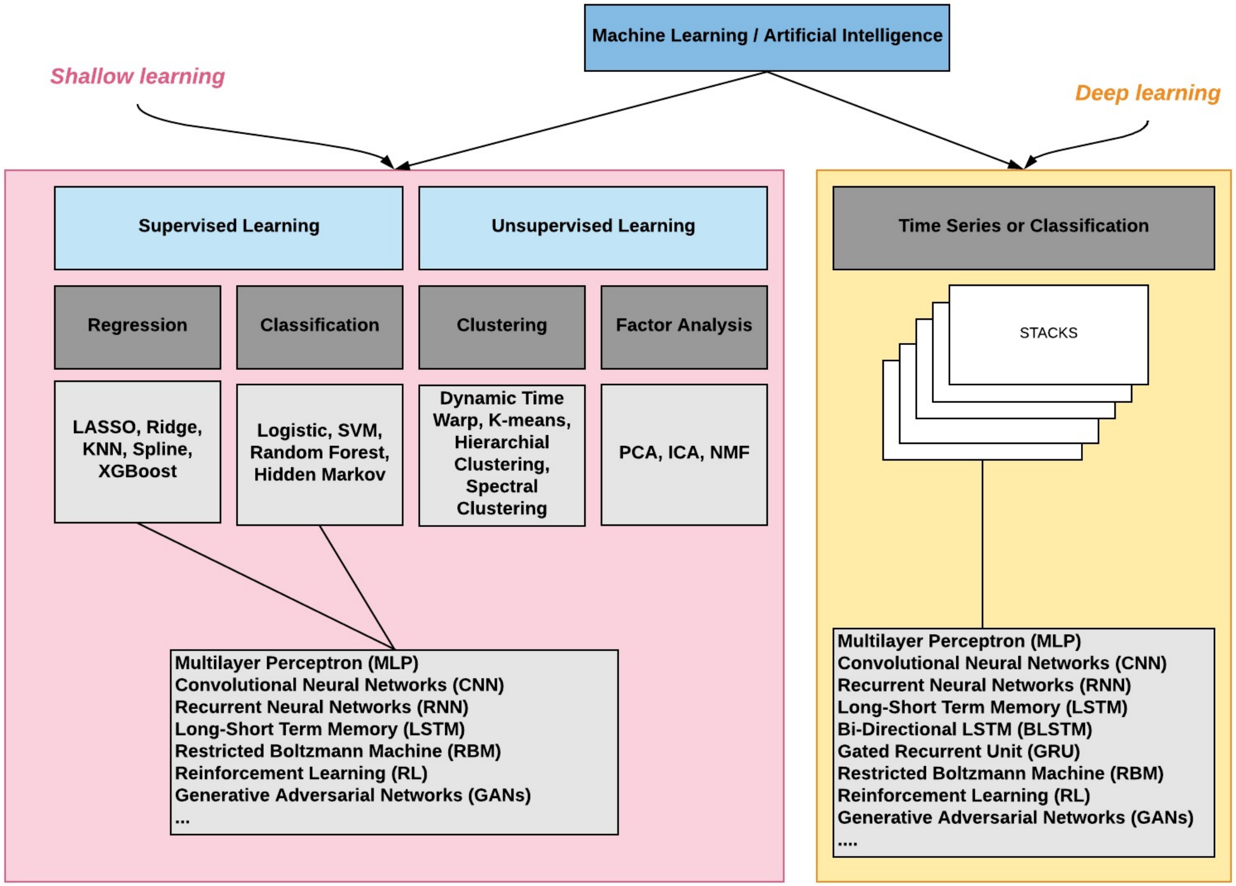

Machine Learning / Artificial Intelligence
Shallow learning
Deep learning
Supervised Learning
Unsupervised Learning
Time Series or Classification
Regression
Classification
Clustering
Factor Analysis
STACKS
LASSO, Ridge, KNN, Spline, XGBoost
Logistic, SVM, Random Forest, Hidden Markov
Dynamic Time Warp, K-means, Hierarchial Clustering, Spectral Clustering
PCA, ICA, NMF
Multilayer Perceptron (MLP)
Convolutional Neural Networks (CNN)
Recurrent Neural Networks (RNN)
Long-Short Term Memory (LSTM)
Restricted Boltzmann Machine (RBM)
Reinforcement Learning (RL)
Generative Adversarial Networks (GANs)
...
Multilayer Perceptron (MLP)
Convolutional Neural Networks (CNN)
Recurrent Neural Networks (RNN)
Long-Short Term Memory (LSTM)
Bi-Directional LSTM (BLSTM)
Gated Recurrent Unit (GRU)
Restricted Boltzmann Machine (RBM)
Reinforcement Learning (RL)
Generative Adversarial Networks (GANs)
....

Figure 2-1. Different Machine Learning and Artificial Intelligence Models

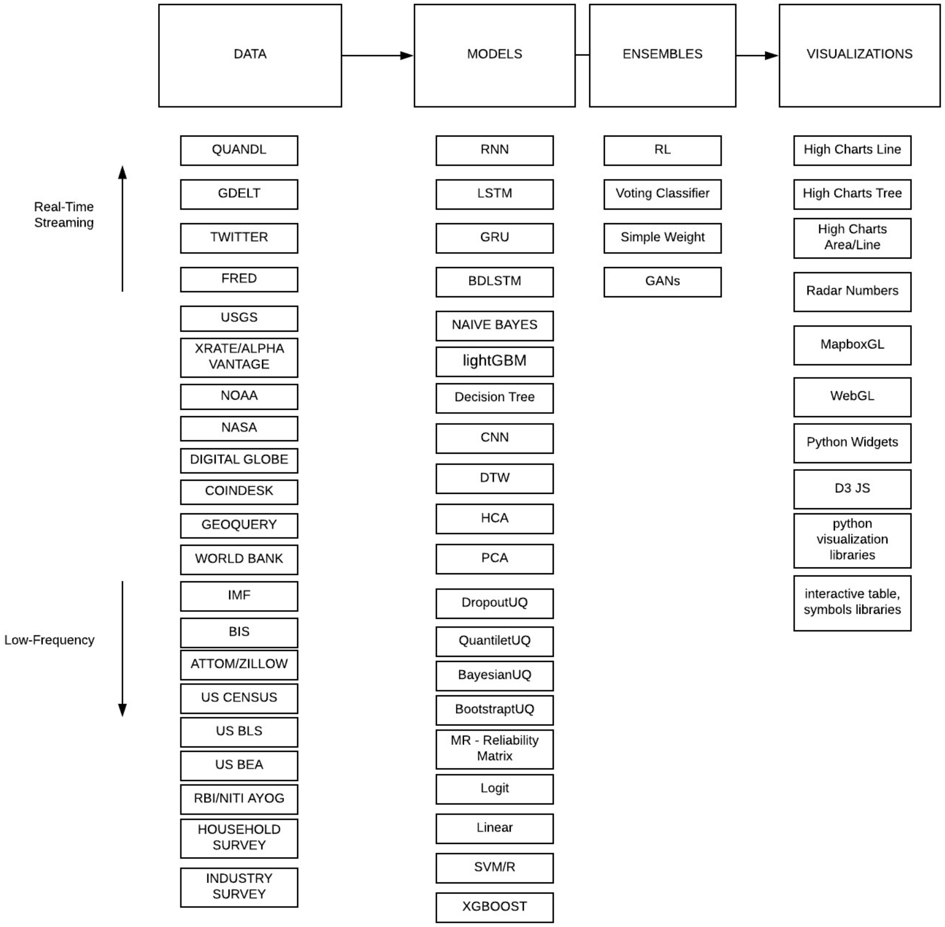


Figure 2-2. A simple schematic of standardization of data, models, and visualizations for various data sources concerning the external environment and useful models

2.1. Dimensionality Reduction and Clustering

The growing availability of large-scale hi-frequency data has added complexity to analytical model building. This increasing computational demands led to what is termed as trans-computational problems capped by the Bremermann's limit (Bremermann 1962). Large-scale data is also plagued with the curse of dimensionality: a phenomenon whereby an increase in the dimensionality of a data set results in exponentially more data being required to produce a representative sample of that data set.

However, there are many ways to process and analyze data for meaningful output and deal with these challenges, especially in high-dimensional space for the domains of interest. One of these ways is called feature extraction which transforms the original data set into a data set with fewer dimensions. As a subset of feature extraction, other processes; such as: dimensionality reduction, help in reducing the number of random variables under consideration via obtaining a set of principal variables.

Similarly, other approaches; such as: Clustering, help in grouping a set of objects in such a way that objects in the same group, called a cluster, are more similar to each other than to those in other clusters. There are a number of other unsupervised machine learning algorithms and methods used for similarity matching and clustering. **The t-Distributed Stochastic Neighbor Embedding (t-SNE)** algorithm is one such algorithm used for visualizing high-dimensional data in lower dimensional space (van der Maaten and Hinton 2008, van der Maaten and Hinton. (2012). van der Maaten 2014). **Multi-dimensional scaling (MDS)** is also used for visualizing low-dimensional data into a high-dimensional space (Syrquin 1978, Machado and Mata 2015). **Autoencoders (AE),** which are artificial neural network that learns a representation/encoding for a set of data, can be used for the purpose of dimensionality reduction (Gamboa 2017).

The main focus of this research is on time-series data. Amongst the many different dimensionality reduction and clustering techniques, only the ones that will be used in subsequent chapters are discussed more in details.

2.1.1. Dynamic Time Warping (DTW)

One of the main techniques used for time-series data is called **Dynamic Time Warping (DTW). DTW** is a valuable algorithm used for: learning of similarity based on distance between two sequences that may vary in speed and quantifying the time dependent similarity between any two pairs. In general, DTW is a machine learning algorithm that calculates an optimal match between two given sequences with certain restrictions. The sequences are “warped” non-linearly in the time dimension to determine a measure of their similarity independent of certain non-linear variations in the time dimension (Ratanamahatana and Koegh 2005). Figure (2-2) below shows the difference between calculating a Euclidian distance versus dynamic time warped distance. The Euclidean distance uses the distance between each pair of the time series and compares them using Euclidean distance. On the other hand, the DTW looks for the best alignment between the two-time series. Moreover, the graph shows that each point is used to compare the point with, not just its pair, but also with other points to create the best alignment between the two-time series based on a distance matrix.

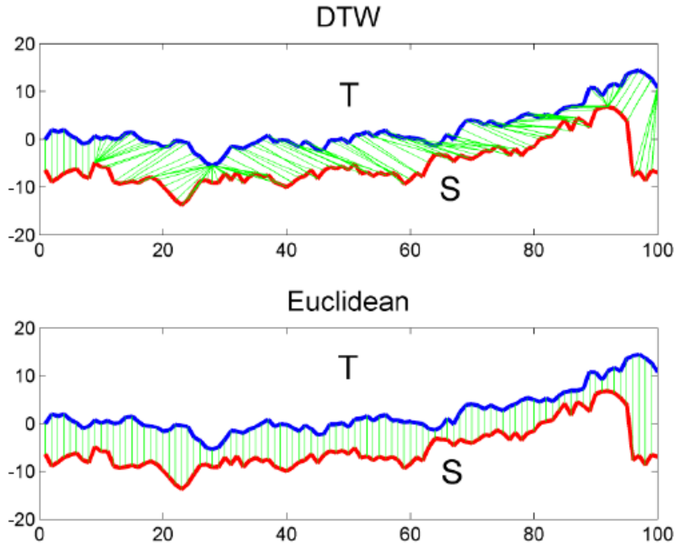


Figure 2-2. Difference between Dynamic Time Warping (DTW) distance and Euclidean distance (green lines represent mapping between points of time series T and S). The former allows many-to-one point comparisons, while Euclidean point-to-point distance (or one-to-one).
Source: Cassisi et al. 2012.

Given its effectiveness, the DTW technique has been used in many applications including: speech recognition (Cassidy 2002), pattern recognition in equity markets (Coelho 2011), and as a similarity measure in finance (Tsinaslanidis et al. 2014). Furthermore, the DTW algorithm is used to measure similarity between two temporal sequences which may vary in speed. For instance, similarities in walking could be detected using DTW, even if one person was walking faster than the other, or if there were accelerations and decelerations during the course of an observation (Ratanamahatana and Keogh 2008). In addition, DTW has been applied to temporal sequences of video, audio, and graphics data …etc since it has the capability of finding the "optimal match between two sequences of feature vectors which allows for stretched and compressed sections of the sequence" (Sakoe and Chiba 2009).

This sequence alignment method is often used in time series classification. For example, consider two sequences

$$\mathcal{A} = a_1, a_2, \dots, a_i, \dots, a_n \tag{2.2}$$

$$\mathcal{B} = b_1, b_2, \dots, b_j, \dots, b_m \tag{2.3}$$

The objective of DTW is to compare two (time-dependent) sequences $\mathcal{A}$ of length $N \in N$ and $\mathcal{B}$ of length $M \in N$. These sequences may be discrete signals (time-series) or, more generally, feature sequences sampled at equidistant points in time. A feature space denoted by F; then $\mathcal{A}_{\mathrm{n}}, \mathcal{B}_{\mathrm{m}} \in F$ for n ∈ [1 : N] and m ∈ [1 : M]. To compare the two different features $\mathcal{A}, \mathcal{B} \in$ F, one needs a local cost measure, sometimes also referred to as local distance measure, which is defined by the following function:

$$c : \mathcal{F} \times \mathcal{F} \rightarrow \mathbb{R}_{\geq 0} \tag{2.4}$$

Typically, c($\mathcal{A}, \mathcal{B}$) is small (low cost) if $\mathcal{A}$ and $\mathcal{B}$ are similar to each other; otherwise c($\mathcal{A}, \mathcal{B}$) is large (high cost).

The two sequences can, also, be arranged on the sides of a grid, with one on the top and the other up the left-hand side. Both sequences start on the bottom left of the grid.

The distance between the two pairs is defined as:

$$\mathcal{D}(\mathcal{A}, \mathcal{B}) = \left[\frac{\sum_{s=1}^{k} d(p_s) \cdot w}{\sum_{s=1}^{k} w_s}\right] \tag{2.5}$$

where $d(p_s)$ is the distance between i_s and j_s, and $w_s > 0$ is the weighting coefficient.

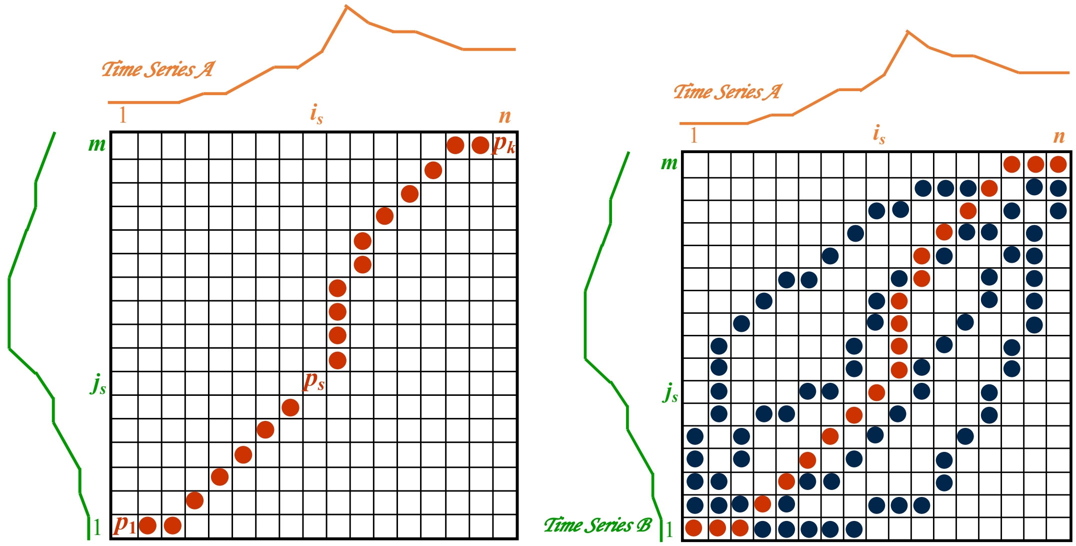


Figure 2-3. Example of Two Time Sequence Dynamic Time Warp (DTW) Grid
Source: Tsiporkova, E. Dynamic Time Warping Algorithm for Gene Expression Time Series.

To align these two sequences using DTW, first an n-by-m matrix is constructed where the (i-th, j-th) element of the matrix corresponds to the squared distance, $d(a_i\ b_j) = \left(a_i - b_j\right)^2$ which is the alignment between points a_i and b_j. To find the best match between these two sequences, the path through the matrix that minimizes the total cumulative distance between them is retrieved. In particular, the optimal path is the path that minimizes the warping cost. The best alignment path between $\mathcal{A}$ and $\mathcal{B}$ can be written as:

$$P_0 = \underset{P}{\operatorname{argmin}}\, \mathcal{D}(\mathcal{A}, \mathcal{B}\,) \tag{2.6}$$

The number of possible warping paths through the grid is exponentially explosive. This warping path can be found using dynamic programming to evaluate the following recurrence.

$$\gamma(i,j) \;=\; d(a_i\, b_j) + \,\min^{\{\gamma(i-1,j-1),\gamma(i-1,j),\gamma(i,j-1)\}} \qquad (2.7)$$

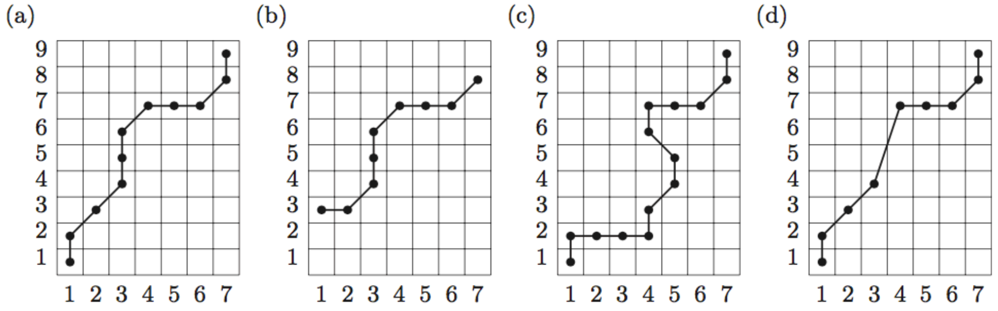
where $d(i,j)$ is the distance found in the current cell, and $\gamma(i,j)$ is the cumulative distance of $d(i,j)$ and the minimum cumulative distances from the three adjacent cells.

An (N, M)-warping path (or simply referred to as warping path if N and M are clear from the context) is a sequence p = ($p1$,...,pL) with pl = (nl,ml) ∈ [1 : N]×[1 : M] for l ∈ [1 : L] satisfying the following three conditions (Ratanamahatana and Keogh 2008):

1- **Boundary Conditions:** p_1 = (1, 1) and p_L = (N, M)

2- **Continuity condition:** no element in $\mathcal{A}$ and $\mathcal{B}$ can be omitted and there are no replications in the alignment (in the sense that all index pairs contained in a warping path p are pairwise distinct)

3- **Monotonic condition:** $n_1 \leq n_2 \leq \ldots \leq n_L$ and $m_1 \leq m_2 \leq \ldots \leq m_L$



.

Figure 2-4. DTW Conditions.
Illustration of paths of index pairs some sequence X of length N = 9 and some sequence Y of length M = 7. (a) Admissible warping path satisfying the conditions, (b) Boundary condition is violated (c) Monotonicity condition is violated, and (d) Step size condition is violated.
Source: Ratanamahatana and Keogh 2008.

2.1.2. Hierarchical Clustering Analysis (HCA)

Although the distances measured in a manner like Euclidean or DTW provide a measure of similarity; nonetheless, in large scale data, it is important to make sense of the data and use a form of hierarchy to map similar information by groups, or by aggregation. Hence, **Hierarchical Clustering Analysis (HCA),** which is used to build a binary tree of the data merged in similar groups of points (Burgard et al. 2010, Ahlquist and Breunig 2009), can solve this problem. However, HCA is computationally expensive, even for medium-size datasets, due to the time complexity of $O(n^3)$ and the $O(n^2)$ memory requirement.

There are different types of HCA; however, the prominent type of Hierarchical clustering is called agglomerative clustering (Manning et al. 2008). The agglomerative clustering algorithm can be simply illustrated in the following form (Blei 2008):

1. Place each data point into its own singleton group
2. Repeat: iteratively merge the two closest groups
3. Until: all the data are merged into a single cluster

Furthermore, agglomerative clustering is monotonic; therefore, the similarity between merged clusters is monotone and decreases with the level of the merge. In addition, each level of the resulting tree is a segmentation of the data and the algorithm results in a sequence of groupings; hence, it is up to the user to choose a "natural" clustering from this sequence. Given the distance measure between points, the user has many choices for how to define intergroup similarity The properties of intergroup similarity include: Single linkage which can produce "chaining" where a sequence of close observations in different groups cause early merges of those groups; complete linkage which has the opposite problem in which it might not merge close groups because of outlier members that are far apart; and finally, group average which represents a natural compromise, but depends on the scale of the similarities as applying a monotone transformation to the similarities can change the results. The different clustering techniques are presented in Table (2-2).

Moreover, dendrogram plot merge at the (negative) similarity between the two merged groups and provides an interpretable visualization of the algorithm and data. This is a useful summarization tool and one of the reasons why hierarchical clustering is popular.

At the same time, there are caveats associated with Hierarchical clustering that should be treated with caution. For instance, different decisions about group similarities can lead to vastly different dendrograms. The algorithm imposes a hierarchical structure on the data, even data for which such structure is not appropriate (Rokach and Oded 2005; Blei 2008). Table (2-3) shows the various distance measures that can be used for building hierarchical clusters.

Readers should not that the research shown in this research used the Dynamic Time Warping (DTW) distance measures.

Table 2-2. Hierarchical Clustering Techniques

Name	Formula
Maximum or complete-linkage clustering	$\max\{d(a,b): a \in A, b \in B\}$
Minimum or single-linkage clustering	$\min\{d(a,b): a \in A, b \in B\}$
Mean or average linkage clustering, or UPGMA	$\frac{1}{\lvert A\rvert\lvert B\rvert}\sum_{a \in A}\sum_{b \in B} d(a,b)$

Table 2-3. Distance Measures for Clustering

Name	Equation	Interpretation
Euclidean distance	$d(p,q) = d(q,p) = \sqrt{(q_1 - p_1)^2 + (q_2 - p_2)^2 + \cdots + (q_n - p_n)^2}$ $= \sqrt{\sum_{i=1}^{n} (q_i - p_i)^2}$ $\lVert p \rVert = \sqrt{p_1^2 + p_2^2 + \cdots + p_n^2} = \sqrt{p \cdot p}$	Euclidean distance or Euclidean metric is the "ordinary" straight-line distance between two points in Euclidean space. A generalized term for the Euclidean norm is the L2 norm or L2 distance.
Manhattan distance	$d_1(p,q) = \lVert p_i - q_i \rVert_1 = \sum_{i=1}^{n} \lvert p_i - q_i \rvert$	The taxicab metric is also known as L1 distance or L1 norm, Manhattan distance. The latter name alludes to the grid layout of most streets on the island of Manhattan, which causes the shortest path a car could take between two intersections in the borough to have length equal to the intersections' distance. Hamming distance can be seen as Manhattan distance between bit vectors.
Kullback–Leibler divergence (KLD)	$D_{KL}(p,q) = \sum_{j} p(i) \log \frac{p(i)}{q(i)}$	KLD measures how one probability distribution diverges from a second expected probability distribution. KLD is the expectation of the logarithmic difference between the probabilities P and Q, where the expectation is taken using the probabilities P. The KLD is defined only if Q(i)=0 implies P(i)=0, for all *i* (absolute continuity). Whenever P(i) is zero the contribution of the i-th term is interpreted as zero because
Chebyshev distance	$D_{Chebyshev}(P \Vert Q) = \max_{i}(\lvert p_i - q_i \rvert)$ $= \lim_{n \to \infty} \left(\sum_{i=1}^{n} \lvert p_i - q_i \rvert^2 \right)^{n}$	Chebyshev distance or L∞ metric is a metric defined on a vector space where the distance between two vectors is the greatest of their differences along any coordinate dimension
Minkowski distance	$d(p,q) = \left(\sum_{i=1}^{n} \lvert p_i - q_i \rvert \right)^{1/X}$	The Minkowski distance is a generalization of both the Euclidean distance and the Manhattan distance. For $X \geq 1$, the Minkowski distance is a metric as a result of the Minkowski inequality. When $X < 1$, the distance between (0,0) and (1,1) is $2^{1/X}$, but the point (0,1) is at a distance 1 from both of these points. Since this violates the triangle inequality, for X<1 it is not a metric.

2.2. Classification Problems

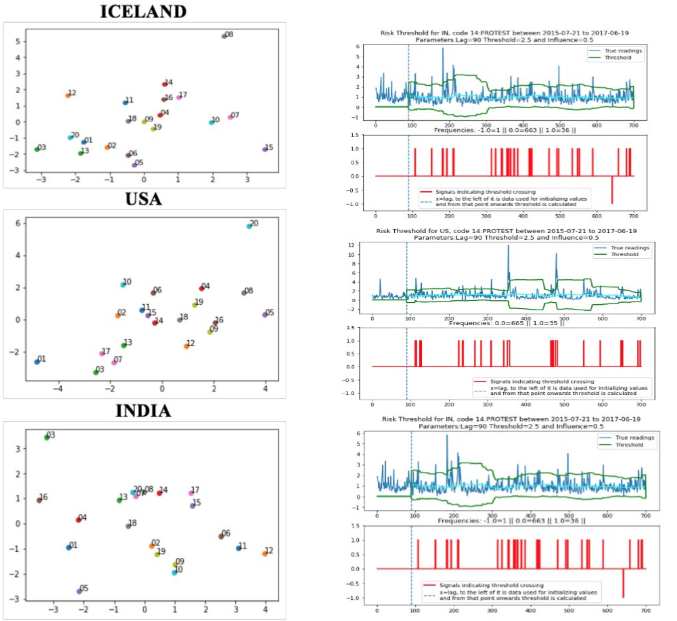
In machine learning, many of the time series random variables are traditionally treated as regression problems; however, they can be also be converted into class label prediction problems. For example, in image recognition, machines might learn to identify images that contain cats by analyzing example images that have been manually labeled as "cat" or "no cat" and using the analytic results to identify cats in other images. Nevertheless, it is not clear how time-series variables, like GDP growth, can be converted to a label problem. One approach would be to classify any GDP growth event below 0 as label "contraction" which enables the treatment of the regression prediction problem as a classification problem.

For such classification problems, a set of algorithms can be employed to process the stochastic random variable and convert them into labels. The first and preferred method is called anomaly detection with **Z-Score Peak-detection Algorithms**. This method is a simple yet powerful algorithm that works very well for time series data to find periods of threshold crossing pre-defined by the analyst (Scholkmann 2012, Palshikar 2009, Ijima and Ohsumi 2010). The z-score method is based on the **principle of dispersion**, i.e. if a new value in a data stream is a given *x* number of standard deviations away from some moving mean, the algorithm signals (also called z-score) (Ijima and Ohsumi 2010).. The second method is the **Bayesian change-point anomaly detection** algorithm. The probability distribution of the length of the current "run," or time since the last change-point is computed using a simple message-passing algorithm (Adam and Mackay 2016). This algorithm is highly modular and designed for a variety of real-time applications. Finally, the third method is used extensively in **sentiment analysis** and is called **Naïve Bayes classifier (NBC).** NBC is a simple probabilistic classifier based on applying Bayes' theorem with strong (naive) independence assumptions between the features (Stuart and Norvig 2010, Hand and Yu 2001, Rennie et al. 2003). NBC has played an important for role in marketing, measuring public sentiment in politics and financial market predictions (Jurafsky and Martin 2017). The coverage of these methods, their descriptions, uses, applications and references are presented in Table (2-1).

Dimensionality Reduction Clustering Similarity based on t-SNE

Z-Threshold Peak Detection Algorithm for Protests (number of mentions)

Code	Event
01	Make Public Statement
02	Appeal
03	Explore Intent to
04	Consult
05	Engage in Diplomatic
06	Engage in Material
07	Provide Aid
08	Yield
09	Investigate
10	Demand
11	Disapprove
12	Reject
13	Threaten
14	Protest
15	Exhibit Force Posture
16	Reduce Relation
17	Coerce
18	Assault
19	Fight
20	Unconventional Mass



Note: above results are based on TSNE dimensionality reduction algorithm. The data is used for past 500 days for GDELT major cameo codes. Parameters are following:perplexity = 20,output dimensions =2, iterations=2000. Time-step of 90 days are taken, where for each time-step difference features are calculated. These features are used as input to TSNE algorithm. The results are based on number of mentions with following features extracted (1) mean, (2) peak, (3) kurtosis, (4) skewness, (5) standard deviation, (6) root mean square (RMS), (7) clearance Factor, (8) minimum, (9) maximum.

Notes: Peak signal detection in real-time timeseries data. The algorithm works well for these types of datasets. It is based on the principle of dispersion: if a new datapoint is a givenxnumber of standard deviations away from some moving mean, the algorithm signals (also called z-score). The algorithm is very robust because it constructs a separate moving mean and deviation, such that signals do not corrupt the threshold. Future signals are therefore identified with approximately the same accuracy, regardless of the amount of previous signals. The algorithm takes 3 inputs: lag = the lag of the moving window, threshold = the z-score at which the algorithm signals and influence = the influence (between 0 and 1) of new signals on the mean and standard deviation. For example, a lag of 5 will use the last 5 observations to smooth the data. A threshold of 3.5 will signal if a datapoint is 3.5 standard deviations away from the moving mean. And an influence of 0.5 gives signals half of the influence that normal datapoints have. Likewise, an influence of 0 ignores signals completely for recalculating the new threshold: an influence of 0 is therefore the most robust option; 1 is the least.

Figure 2-5. Dimensionality reduction (t-SNE) and Z-threshold peak detection algorithm example for GDELT data

Notes: Panel A shows the different event codes from GDELT based on a 500-day sample based on t-SNE. Similar events are closer together for Iceland, USA, and India, respectively. Panel B shows the Z-threshold peak detection algorithm for threshold crossing probability of event “Protests” for the same countries.

2.3. Model Accuracy

The most common metrics used to measure the accuracy of models are the **Mean Absolute Error (MAE)** and the **Root mean squared error (RMSE)** which will be used throughout the rest of this research. The MAE is the average magnitude of errors in the test sample i.e. the absolute difference between the prediction and actual observation, where individual observations have equal weight. The RMSE is a quadratic scoring rule that additionally accommodates the average magnitude of the error (Holmes 1999).

$$MAE = \frac{1}{n}\sum_{j=1}^{n}\left|y_j - \hat{y}_j\right| \tag{2.8}$$

$$RMSE = \sqrt{\frac{1}{n}\sum_{j=1}^{n}\left(y_j - \hat{y}_j\right)^2} \tag{2.9}$$

where y_j represents the actual value of the continuous random variable and $\hat{y}_j$ predicted by any of the unsupervised or supervised machine learning models presented. A RMSE or MAE closer to 0 indicates that the forecast model fits the actual data much better. A related measure is the **Mean Squared Error (MSE)** which is essentially the RMSE without the squared root.

Similarly, **correlation,** measured as a correlation coefficient, indicates the strength and direction of a linear relationship between two variables; for example, model output and observed values. The Pearson correlation coefficient is obtained by dividing the covariance of the two variables by the product of their standard deviations. A correlation coefficient of +1 indicates perfect increasing linear relationship, while -1 indicates decreasing linear relationship; and a correlation coefficient of 0 means that there is no linear relationship between the variables. The square of the Pearson correlation coefficient (r^2) describes how much of the variance between the two variables is described by the linear fit (NCME 2014). Take one dataset $\{x_1,...,x_n\}$ containing n values and another dataset $\{y_1,...,y_n\}$ containing n values, where $\bar{x}$ and $\bar{y}$ represent the sample mean, then the formula for r is:

$$r = \frac{\sum_{i=1}^{n}(x_i - \bar{x}) \cdot (y_i - \bar{y})}{\sqrt{\sum_{i=1}^{n}(x_i - \bar{x})^2 \cdot \sum_{i=1}^{n}(y_i - \bar{y})^2}} \tag{2.10}$$

2.4. Regression Problems

In addition to the above techniques, machine learning methods are used to analyze large and complex datasets providing significant developments in the field of pattern recognition and function approximation (uncovering relationship between variables) (Samuel 1959, Kohavi 1998). These analytical methods, called regression, are part of the broader disciplines of Statistics and Computer Science. First, computers are given an input (set of variables and datasets) and an output that is a consequence of the input variables. Then, the machine finds or "learns" a rule that links the input and output. There are a number of regression techniques used which will be detailed in the following sections.

2.4.1. Linear Regression

Linear regression is the classic regression technique and is considered the 'work horse" of statistics (supervised) machine learning (Seal 1967, Murphy 2012).

The simple form where the response is a linear function of the inputs is written as follows

$$y(x) = w^T x + \epsilon = \sum_{j=1}^{D} w_j x_j + \epsilon \qquad (2.11)$$

where w^T represents the inner or **scalar product** between the input vector *x* and the model's **weight vector w** (or in economics and statistics referred to as β), and ϵ is the **residual error** between our linear predictions and the true response (Seal 1967, Freedman 2009, Yan 2009, Rencher et al. 2012).

It is often assumed that ϵ has a Gaussian or normal distribution which is denoted by $\epsilon \sim \mathcal{N}\ (\mu\ , \sigma^2)$ where μ is the mean and σ is the variance. Plotting this distribution, we get the **bell curve**.

To make the connection between linear regression and Gaussian more explicit, the model can be rewritten in the following form

$$p(y|x, \theta) = \mathcal{N}\big(y\big|\mu(x), \sigma^2(x)\big) \qquad (2.12)$$

This makes it clear that the model is a conditional probability density. In the simplest case, it is assumed that μ is a linear function of x, so $\mu = w^T x$, and that the noise is fixed, $\sigma^2(x) = \sigma^2$. Then, $\theta = (w, \sigma^2)$are the parameters of the model.

This method, when augmented with kernel or other forms of basis function expansion, can model also non-linear relationships (Murphy 2012). by replacing the *x* with some non-linear function of input, $\phi(x)$ and can be re-written as

$$p(y|x, \theta) = \mathcal{N}(y|w^T\ \phi(x), \sigma^2) \qquad (2.13)$$

This is called the **basis function expansion**. Suppose that basis function has the following form where $\phi(x) = [\ 1, x, x^2, x^3, \dots, x^d]$; this is known as **polynomial regression**. In fact, many popular machine learning methods; such as: support vector machines, neural networks, classification, regression trees etc. can be seen as different ways of estimating the basis function (Murphy 2012, Rencher et al. 2012).

2.4.2. Logistic Regression

One can generalize linear regression to the (binary) classification setting, known as **logistic regression**, by making two changes (Cox 1958, Walker and Duncan 1967, Freedman 2009). First, replace the Gaussian distribution for *y* with **Bernoulli distribution**, which is more appropriate for the case when response is binary, $y \epsilon$ {0, 1}, the following expression can be used

$$p(y|x, w) = Ber\left(y\middle|\mu(x)\right) \tag{2.14}$$

where $\mu(x) = \mathbb{E}\,[y\,|\boldsymbol{x}] = p(\,y = 1\,|x)$

Second, compute a linear combination of the inputs, as before, but when we pass this through a function that ensures $0 \ \le \ \mu(x) \le 1$ by defining

$$\mu(x) = \ sigm\,(w^T\ x\ + b) \tag{2.15}$$

where $w\ \in\ \mathbb{R}^{N_x}, b\ \in\ \mathbb{R}$ and $sigm(\eta)$ refers to the **sigmoid** function, also known as the logistic or logit function defined as

$$sigm\,(\eta) \triangleq \frac{1}{1 + exp\,(-\eta)} = \frac{e^{\eta}}{e^{\eta} + 1} \tag{2.16}$$

The term "sigmoid" means S-shaped and is also known as a **squashing function,** since it maps the whole real line to [0, 1] which is necessary for the output to be interpreted as a probability (Walker and Duncan 1967). The graphical representation of the sigmoid function is shown in figure (2-6).

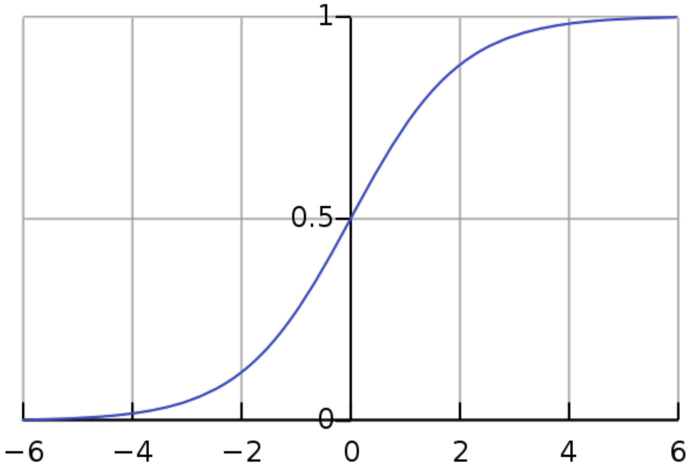


Figure 2-6. Sigmoid function

Putting above two steps together yields

$$p(y|x,w) = Ber\ (y|\ sigm\ (w^T\ x)\ ) \qquad (2.17)$$

This is the **logistic regression** due to its similarity to linear regression, although it is a form of classification and not a regression (Murphy 2012).

A simple example of logistic regression can be presented as follow

$$p(y_{it} = 1\ |x_{it},\ w)\ = sigm\ (w_0 +\ \ w_1\ x_i) \qquad (2.18)$$

assume here that x_{it} is the Public Debt (% of GDP) of country *i* at time *t* and y_{it} is whether they passed a certain threshold like 100% of GDP to define “Risk - 1” or “No Risk - 0”. If we threshold the output probability at 100, we can induce a decision rule of the form

$$\hat{y}\ (x) = 1\ \ \Leftrightarrow p(y = 1\ |x)\ > 100 \qquad (2.19)$$

If *x* is given to us, we want to predict $\hat{y}$. If we assume that $(w^T\ x\ + b) = z$, then $\hat{y} =\ sigm\ (z),$ we know that $sigm\ (z),\ = \frac{e^z}{e^z+1}$.

If z is too large a number then $sigm\ (z) = 1$. If z is very small or a very large negative number then $sigm\ (z) \approx 0$. The idea is to learn parameters w and b so that $\hat{y}$ becomes a good estimate of y.

2.5. Artificial Neural Networks (ANNs)

Inspired by biological neurons, neural networks were first proposed in 1944 by Warren McCullough and Walter Pitts (McCullough and Pitts 1944). In order to apply the brain's biological processes to artificial intelligence led to finite automate (Kleene 1956), computer calculators (Farley and Clark 1956), perceptions (Rosenblatt 1958). Neural networks were a major area of research in both neuroscience and computer science until 1969 when the computation limits of large neural networks were identified (Minskey and Papert 1969). **Artificial Neural networks (ANN's)** systems learn (with progressively improving performance) to do tasks by considering examples, generally without task-specific programming.

A neural network is composed of a large number of highly interconnected processing elements (**neurons**) working in parallel to solve a specific problem (Anthony 2001, Maan et al. 2016, Hinton et al. 2015). An artificial neuron is a device with many inputs and one output. The neuron has two modes of operation: the training mode and the using mode. The neurons fire based on a set of flexible rules. Neural networks have similar structure as linear models. In supervised learning, back-propagation requires a known, desired output for each input value, but it is also used in unsupervised networks such as auto-encoders as described earlier. **Back-propagation** algorithms are used calculate the error contribution of each neuron after a batch of data is processed (Werbose 1975, Rumelhart and McClelland 1986, Schmidhuber 1992). This is used by an enveloping optimization algorithm to adjust the weight of each neuron, completing the learning process for that case. Technically it calculates the gradient of the loss function. It is commonly used as the **stochastic gradient descent (SGD)** optimization algorithm, which is also called backward propagation of errors, because the error is calculated at the output and distributed back through the network layers (Ferguson 1982, LeCunn et al. 2012, Rumelhart et al. 1986). Neural networks always end up being an **optimization** problem. Over the last decade, the best-performing artificial-intelligence systems; such as: the speech recognizers on smartphones or Google's latest automatic translator, have resulted from a technique called "deep learning." Essentially deep learning is an iteration on traditional neural networks with more hidden layers. The different architectures of neural networks are discussed later in this section.

There are different types of ANN; **Feed-forward ANNs** allow signals to travel one way only, from input to output without feedback (loops). In this type, time has no role, there are no cycles or loops in the network. Based on the earlier introduced the logistic regression, feed-forward ANN's have a similar structure as linear model. Here we have input x_n linearly combined with weights summed up to a signal and then the signal passes through a threshold labeled $\boldsymbol{\theta}$. The model implements a general probability measure which has a probability error. In which we maximize the ouput (y_n) based on the inputs x_n. The goal being that ***h(x)*** is the identified function mapping relation of x and y. The bottom is a bounded regression problem. Similar to logistic regression, the only exception is (a) we do it multiple times, and (b) involve backward calculation. A simple example of a two-layer neural network is presented in Figure (2-7).

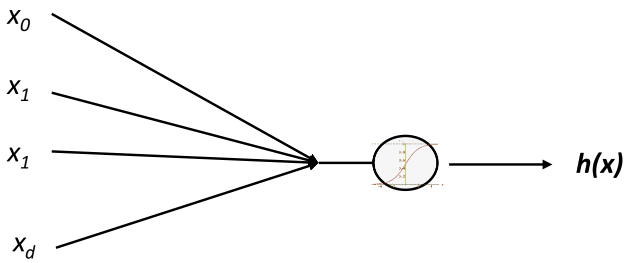


Figure 2-7. A Simple Neural Network

The Perceptron

Another parameter of the ANNs that can be used sub-categorize them is perceptron. The perceptron is a mathematical model of a biological neuron (Rosenblatt 1957, Freund and Schapire 1999). While in actual neurons the dendrite receives electrical signals from the axons of other neurons, in the perceptron these electrical signals are represented as numerical values. At the synapses between the dendrite and axons, electrical signals are modulated in various amounts (Hormuzdi et al. 2003). This is also modeled in the perceptron by multiplying each input value by a value called the weight. An actual neuron fires an output signal only when the total strength of the input signals exceeds a certain threshold. We model this phenomenon in a perceptron by calculating the weighted sum of the inputs to represent the total strength of the input signals and applying a step function on the sum to determine its output. As in biological neural networks, this output is fed to other perceptron.

The perceptron can be considered as the first important implementation of artificial neural networks. It was created in the fifties by Rosenblatt (1968, 1969). As its name indicates, the perceptron was designed to mimic or to model perceptual activities. The main goal was to associate binary configurations, i.e., patterns of [0,1] values, presented as inputs on an artificial retina with specific binary outputs.

Essentially, the perceptron is made of two layers: the input layer (i.e., the “retina"), and the output layer. The activation of the cells of the input layer is transmitted to the cells of the output layer through connections (“synapses”) which can change the intensity of the signal (by multiplying the incoming signal by a “weight" or “synaptic efficacy"). Figure (2-10) provides a simple visual explanation. The response of the perceptron is a function of both the stimulus applied to the cells of the input layer and of the weights of the connections. The cells of the output layer compute their state of activation as a function of the stimulation they receive through the synapses, and then give a binary response as a function of their state of activation. More formally, the response of the output cells depends upon their level of activation which is computed as the sum of the weights coming from active input cells.

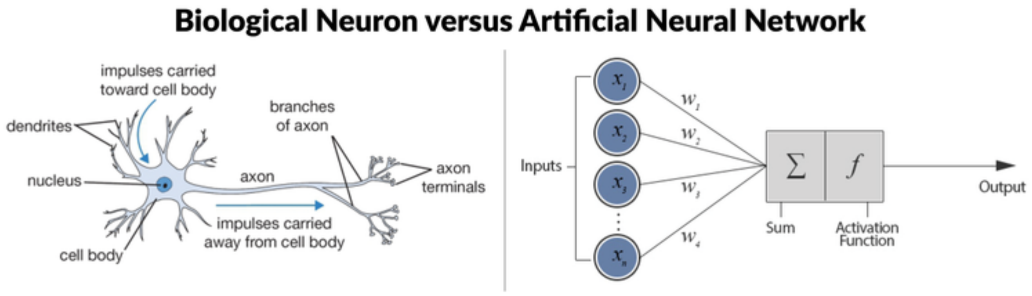


Figure 2-8. Biological neural and an artificial neuron (perceptron)
Source: Willems 2017.

Next, the response is then obtained by thresholding the activation which means that the cell will be in the active state only if its activation level is larger than a given threshold.

Specifically, the activation of the *j*-th output cell is computed as

$$a_j = \sum_{i}^{I} x_i \, w_{i,j} \tag{2.20}$$

with:

a_j is the activation of the *j*-th output cell, x_i is the state of the *i*-th cell of the retina (0 or 1), and $w_{i;j}$ is the value of the weight connecting the *i*-th cell of the retina to the *j*-th output cell (Abdi 1994).

The output cells will then take either the active state (i.e., give the response 1) or the inactive state (i.e., give the response 0) if their level of activation is greater or less than their threshold noted ϑ_j (quite often ϑ_j is set to 0). Precisely, the response of the *j*-th output cell is given as:

$$o_j = \begin{cases} 0\, for\ a_j \leq \vartheta_j \\ 1\ for\ a_j > \vartheta_j \end{cases} \tag{2.21}$$

Moreover, the threshold value ϑ_j can also be modified by learning the weights. The threshold can be implemented by setting an input cell always active (the 0-th input cell) so that the $w_{o,j}$ = - ϑ_j. Essentially ϑ_j can be computed as $w_{o,j}$ is "clamped" to the value -1. The following term is often called the **response bias** and a synonym for threshold. This can be now represented as:

$$o_j = \begin{cases} 0\, for\ a_j + w_{0,j} \ \leq\ \vartheta_j \\ 1\ \ for\ a_j +\ w_{0,j}\ >\ \vartheta_j \end{cases} \tag{2.22}$$

The inputs are noted x_i, the synaptic weights are noted $w_{i;j}$, the total activation of the cell is noted a_j , its response or output, noted o_j , is equal to 0 if $a_j \leq 0$ and 1 if $a_j > 0$.

Accordingly, the **SLP (Single Layer perceptron)** has a single hidden layer and is most commonly used as a Gaussian function. Every neuron has a center and a radius/spread; and the number of neurons in the hidden layer and weights are determined while training. On the other hand, **MLP (Multi-Layer Perceptron)** has multiple hidden layers designed in such a way that the input signal passes through each node of the network only once (Rosenblatt 1961, Rumelhart et al. 1986, Cybenko 1989). **MLP** takes a decision by approximating hyperplanes where the number of hidden layers and neurons in the hidden layers is set by the programmer.

With ANNs we have an input, x_n, linearly combined with weights summed up to a signal and then the signal passes through a threshold labeled $\boldsymbol{\theta}$. The model implements a general probability measure which has a probability error to maximize the output (y_n) based on the inputs x_n. MLP is similar to logistic regression with the only exceptions been: (a) we do it multiple times; and (b) involves backward calculation. A simple example of a 2-layer neural network is presented in Figure (2-7). The likelihood measure for logistic models is defined as

$$\prod_{n=1}^{N} p(y_n|x_n) = \prod_{n=1}^{N} \theta\ (y_n w^T x_n) \tag{2.23}$$

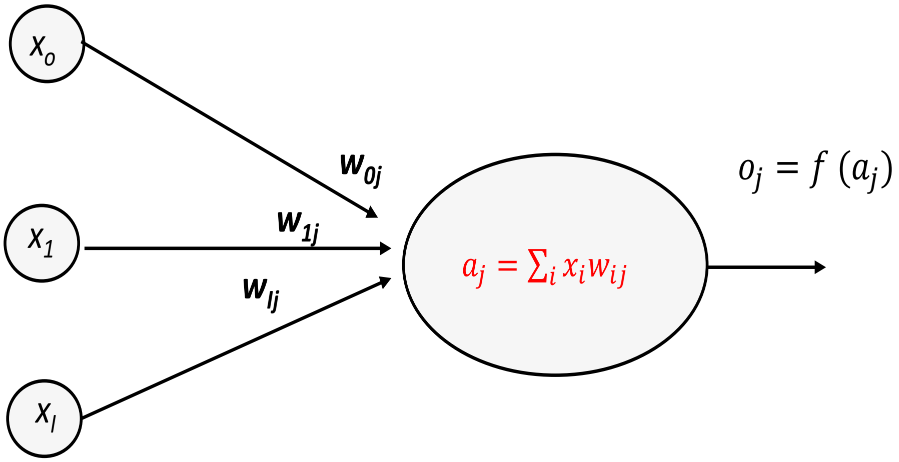


Figure 2-9. The output cell of a perceptron.

Combining Multiple Perceptrons

Moreover, the **activation function** defines the output of a node in the ANN given an input or set of inputs. A standard computer chip circuit can be seen as a digital network of activation functions that can be "ON" (1) or "OFF" (0), depending on input. In its simplest form, the activation function is usually an abstraction representing the rate of action potential firing in the cell, as described earlier in the case of a perceptron. However, only nonlinear activation functions allow such networks to compute nontrivial problems using only a small number of nodes. In artificial neural networks this function is also called the **transfer function**. There are different activation functions with different functions: the sigmoid activation function is used for binary classification for the output layer; the tanh activation function squishes the output between -1 and 1; while the Rectified Linear Unit (ReLu) is increasingly the default function for activation function.

On the other hand, one major issue with neural networks and machine learning is overfitting. The **Variance-Bias Tradeoff** states that an out of sample forecast will deteriorate because of three factors: in-sample forecast error i.e. model instability, model bias, and error due to our inability to forecast completely random events. **Overfitting** will likely result in much larger errors on out of sample batch of test data with increasing model complexity leading to higher instability. The 'art' is selecting the model that finds optimal balance between 'model bias', and 'model variance'. There are some techniques to prevent overfitting; for example, **dropout** can be used to randomly drop units (along with their connections) from the neural network during training to prevent units from co-adapting too much (Srivastava et al. 2014). Furthermore, **Cross-validation** can split the training set into two: a set of examples to train with, and a validation set. Prediction on the validation set is used to determine which model to use (Pool and Mackworth 2017, Russel and Norvig 2012). More details on aspects of empirical risk minimization and techniques to avoid overfitting are discussed in Appendix E.

The building block of multiple perceptron's are simple Boolean logic gates AND or OR gates. The following two examples show (a) system returns 1 if and only if both are 1's, (b) implement an OR gate if only one of them is 1.

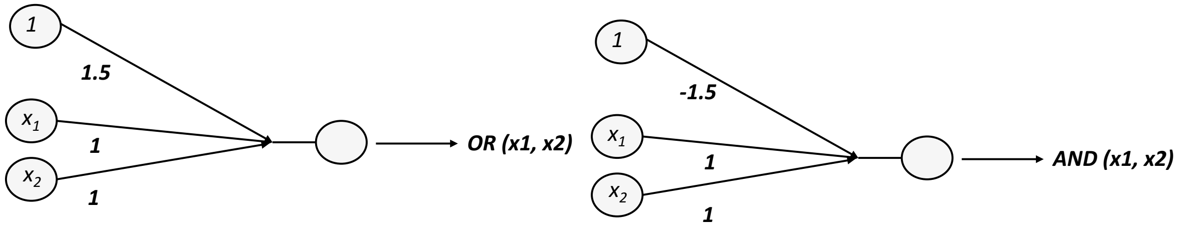
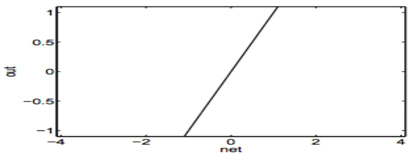
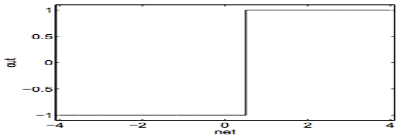
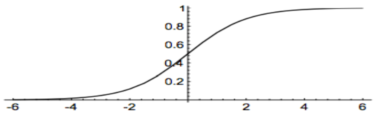
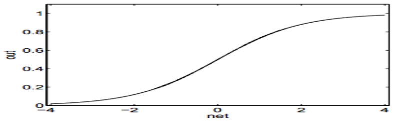
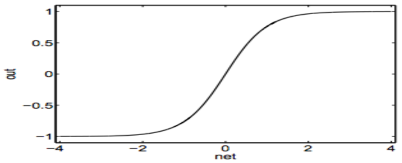
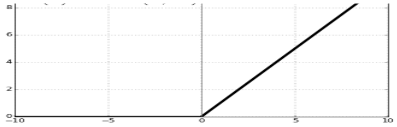


Figure 2-10. AND-OR GATES in Perceptrons

Table 2-4. Activation Functions

Name	Plot	Equation	Range	Derivative Monotonic
Linear or Identity		$f(x) = x$	(-∞, ∞)	Yes
Threshold or Binary Step		$f(x) = \begin{cases} 0 & for\ x < 0 \\ 1 & for\ x\ \geq 0 \end{cases}$	{0,1}	No
Logistic (or soft step function)		$f(x) = \frac{1}{1+\ e^{-x}}$	(0,1)	No
Sigmoid		$f\ (x) \triangleq\ = \frac{e^{x}}{e^{x}+1}$	(0,1)	
TanH		$f(x) = \tanh(x) = \frac{2}{1+\ e^{-2x}} - 1$	(-1,1)	No
Rectified Linear Unit (ReLU)		$f(x) = \begin{cases} 0 & for\ x < 0 \\ x & for\ x\ \geq 0 \end{cases}$	[0,∞)	Yes

Name	Plot	Equation	Range	Derivative Monotonic
Softmax		$f_i(\vec{x}) = \frac{e^{x_i}}{\sum_{j=1}^{J} e^{x_j}}\ for\ i = 1, \ldots, J$	(0,1)	
Maxout		$f_i(\vec{x}) = \max_i x_i$	(-∞, ∞)	

Notes: The last two rows of the table lists activation functions that are not functions of a single fold *x* from the previous layer or layers:

Deep Neural Networks

Deep learning is a new name for an old approach to artificial neural networks. **Deep learning (DL)** is similar to neural networks but has many hidden layers; more specifically DL is a Machine Learning method that analyzes data in multiple layers of learning (hence 'deep'). DL starts doing so by learning about simpler concepts and combining these simpler concepts to learn about more complex concepts and abstract notions. A simple illustration of an economic case of a neural network is presented in Figure (2-11). The input can be a sequence of variables such as current account balance, GDP growth, unemployment etc. over the history of sequence. There can be many hidden layers. The output would be a prediction of the consumer price inflation sequence in the testing set. The forecast errors can be measure deviations from the actual. Moreover, an illustration of a shallow - layer neural network and a deep neural network are provided in Figure (2-12). Modern neural networks have many additional layer types to deal with dropout, convolutional, pooling, and recurrent layers to deal with complex representations or automatic feature engineering for the layers (Hinton et al. 2006).

Furthermore, there are many different architectures of deep neural networks; nevertheless, in most of the time-sequence prediction problems, the many-to-one deep neural network architecture is used and will be discussed briefly in this section. **Convolutional neural networks (CNN**) makes spatial consistency and are deployed regularly in vision and image recognition tasks. CNNs are often used for classifying images as they extract data features by passing multiple filters over overlapping segments of the data (this is related to mathematical operation of convolution) (Zhang 1988, Zhang 1990, LeCunn 2013). The focus of the following section will be on **Recurrent Neural Networks (RNN)** and its variants that make temporal consistencies for parameters (Hopfield 1982, Schmidhuber 1993, Miljanovic 2012).

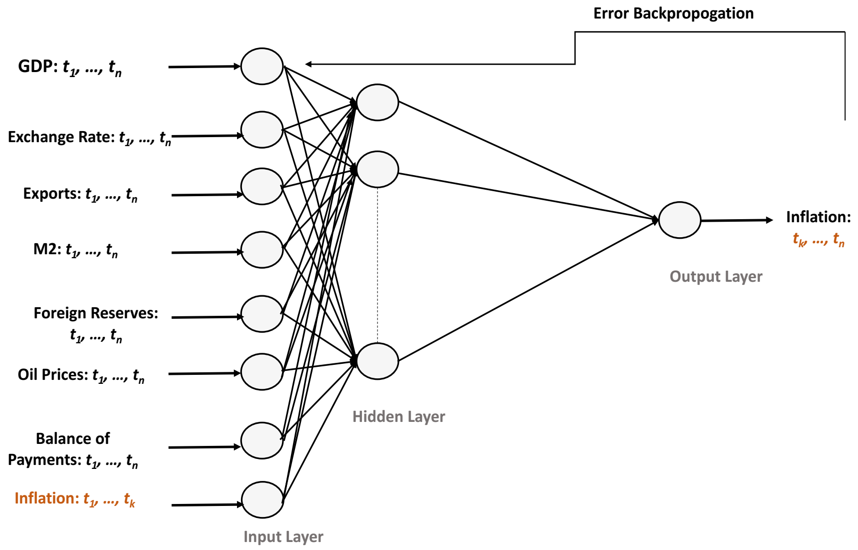

Error Backpropogation
GDP: t_1, …, t_n
Exchange Rate: t_1, …, t_n
Exports: t_1, …, t_n
M2: t_1, …, t_n
Foreign Reserves: t_1, …, t_n
Oil Prices: t_1, …, t_n
Balance of Payments: t_1, …, t_n
Inflation: t_1, …, t_k
Input Layer
Hidden Layer
Output Layer
Inflation: t_k, …, t_n

Figure 2-11. A simple Neural Network architecture for economic cases of prediction and time series forecasting

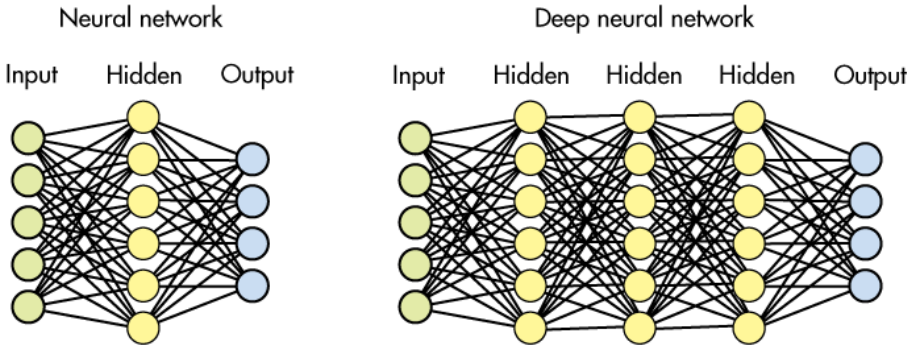


Figure 2-12. Shallow Neural Network (left) and Deep Neural Network with Many "hidden" layers (right) *Source*: Electronic Design 2017.

Neural Network Model	Description	Applications	References
Convolutional Neural Networks (CNN)	CNN is a class of deep, feed-forward ANN that has successfully been applied to analyzing visual imagery. A CNN consists of an input and an output layer, as well as multiple hidden layers. The hidden layers are either convolutional, pooling or fully connected.	object recognition for ImageNet, image and video recognition, recommender systems and natural language processing.	Zhang et al, 2017 ; Matusugu et al, 2013 ; LeCun, 2013 ; Zhang, 1988 ; van den Oord et al, 2013 ; Collobert et al, 2008
Recurrent Neural Networks (RNN)	RNN is a class of ANN where connections between units form a directed cycle. This allows RNN to exhibit dynamic temporal behavior. Unlike feedforward neural networks, RNNs can use their internal memory to process arbitrary sequences of inputs.	unsegmented, connected handwriting recognition, speech recognition, stock prices, keyword spotting	Fernandes et al, 2007 ; Graves et al, 2009 ; Gal et al, 2015 ; Lipton, 2015 Felix and Schmidhuber, 2000;
Deep Belief Networks (DBN)	DBN is a generative graphical model, or alternatively a class of deep neural network, composed of multiple layers of "hidden units" with connections between the layers but not between units within each layer.	natural language understanding, generating and recognizing images, video sequences, motion-capture data, non-linear dimensionality reduction	Sarekas, Hinton, and Deoras, 2014 ; Hinton, Osindero & Teh 2006, Ranzato et. al. 2007, Bengio et.al.. (2007). Sutskever and Hinton. (2007). Taylor et. al. 2007 Hinton & Salakhutdinov,2006; Salakhutdinov and Hinton,2007
Long-Short Term Memory (LSTM) RNN	LSTM is well-suited to classify, process and predict time series given time lags of unknown size and duration between important events.	speech recognition, text-to-speech synthesis, Google Android, Siri, Google Voice Search, machine translation, multilingual language processing	Sak, Senior and Beaufays, 2014 ; Li and Wu, 2014 ; Fan et al, 2015 ; Schmidhuber, 2015 ; Gilick et al, 2015 ; Felix, 2001 ;
Gated Recurrent Units (GRU)	GRU's are a gating mechanism in RNN. They have fewer parameters than LSTM, as they lack an output gate.	polyphonic music modeling, speech signal modeling, emotion classification in noisy speech, any sequence prediction	Dey and Salem, 2017 ; Rana et al, 2016 ; Heck and Salem, 2017
Bi-Directional LSTM	Bidirectional LSTM is a variant of a Hopfield network store associative data as a vector. The bi-directionality comes from passing information through a matrix and its transpose.	Keyword spotting, time-series analysis, speech recognition, handwritten recognition, protein structure prediction	Schuster and Parival, 1997 ; Graves and Schmidhuber, 2005 ; Fernandes et al, 2007 ; Liwicki et al, 2007 ; Baldi, 1999

Table 2-5. Neural Network Architectures: CNN, RNN, DBN, LSTM, GRU, BDLSTM

2.6. Recurrent Neural Networks (RNN)

A recurrent neural network (RNN) is an extension of a conventional feedforward neural network, which is able to handle a variable-length sequence input. The two big limitations of vanilla neural networks are that (a) they accept fixed-sized vector input and output, and (b) traditionally models use a fixed number of computational steps (Karapathey 2015). The RNN handles the variable-length sequence by having a recurrent hidden state whose activation at each time is dependent on that of the previous time. However, in contrast to simple neural networks which map one input to one output, RNNs can have any of the one-to-many or many-to-many configurations as information can flow in both directions. Moreover, RNNs can use their internal memory to process arbitrary sequences of inputs and can incorporate delays based on certain trained parameters. Many variants of the RNN, especially LSTM and GRU are described briefly in the table (2-5). RNNs take different configurations as shown in figure (2-13).

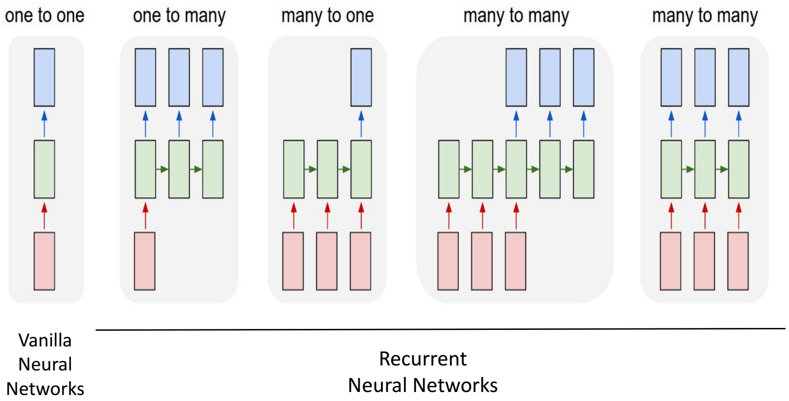


Figure 2-13. Vanilla Neural Networks and Forms of RNN's
Source: Karpathy 2015.

Notes: Each rectangle is a vector and arrows represent functions (e.g. matrix multiply). Input vectors are in red, output vectors are in blue and green vectors hold the RNN's state (more on this soon). From left to right: (1) Vanilla mode of processing without RNN, from fixed-sized input to fixed-sized output (e.g. image classification). (2) Sequence output (e.g. image captioning takes an image and outputs a sentence of words). (3) Sequence input (e.g. sentiment analysis where a given sentence is classified as expressing positive or negative sentiment). (4) Sequence input and sequence output (e.g. Machine Translation: an RNN reads a sentence in English and then outputs a sentence in French). (5) Synced sequence input and output (e.g. video classification where we wish to label each frame of the video). Notice that in every case are no pre-specified constraints on the lengths sequences because the recurrent transformation (green) is fixed and can be applied as many times as we like.

A large number of papers are published in the literature on RNN's. The fundamentals of RNN and methods of training are discussed next, followed by discussion on variety of RNN architectures.

Similar to artificial neurons with recurrent cycles of feedback loops, RNN's require training a set of input-output target pairs with the objective minimize the loss (difference between output and target pairs by optimizing the network's weights (Hayken 1994, Mikolov 2010). Figure (2-14) shows a simple three-layer neural network with *N* inputs and a recurrent hidden layer. The importance of initialization and momentum of hidden units using small non-zero elements can help improve performance (Hinton et al. 2003, Salehinejad et al. 2018). The hidden layer defines the state space or "memory" of the system as (Salehinejad et al. 2018)

$$h_t = \zeta_H(o_t) \tag{2.24}$$

where

$$o_t = W_{IH}x_t + W_{HH}h_{t-1} + b_h \tag{2.25}$$

where $\zeta_H\,(.)$ is the hidden layer activation function, b_h is the bias vector of hidden unit, the connections of the RNN's hidden layer are represented by matrix W_{IH} which has *M* hidden units

$h_t = (h_1, h_2, .., h_m\,)$ that are connected through time with recurrent connections to each other, and finally the hidden units are connected to the output layer with a weighted connection W_{HO}. The output layer has P units $y_t = (y_1, y_2, .., y_P)$ computed as

$$y_t = \zeta_O(W_{HO}h_t + b_o) \tag{2.26}$$

where ζ_O is the activation function and b_o is the bias vector of the output layer. The above three steps are repeated consequently over time in a consistent non-linear state iteration through time (Salehinejad et al. 2018). The information over past states of the network over many timesteps and integrated information defines the nature of the network yielding accurate predictions in the output layer (Sutskever et al. 2011).

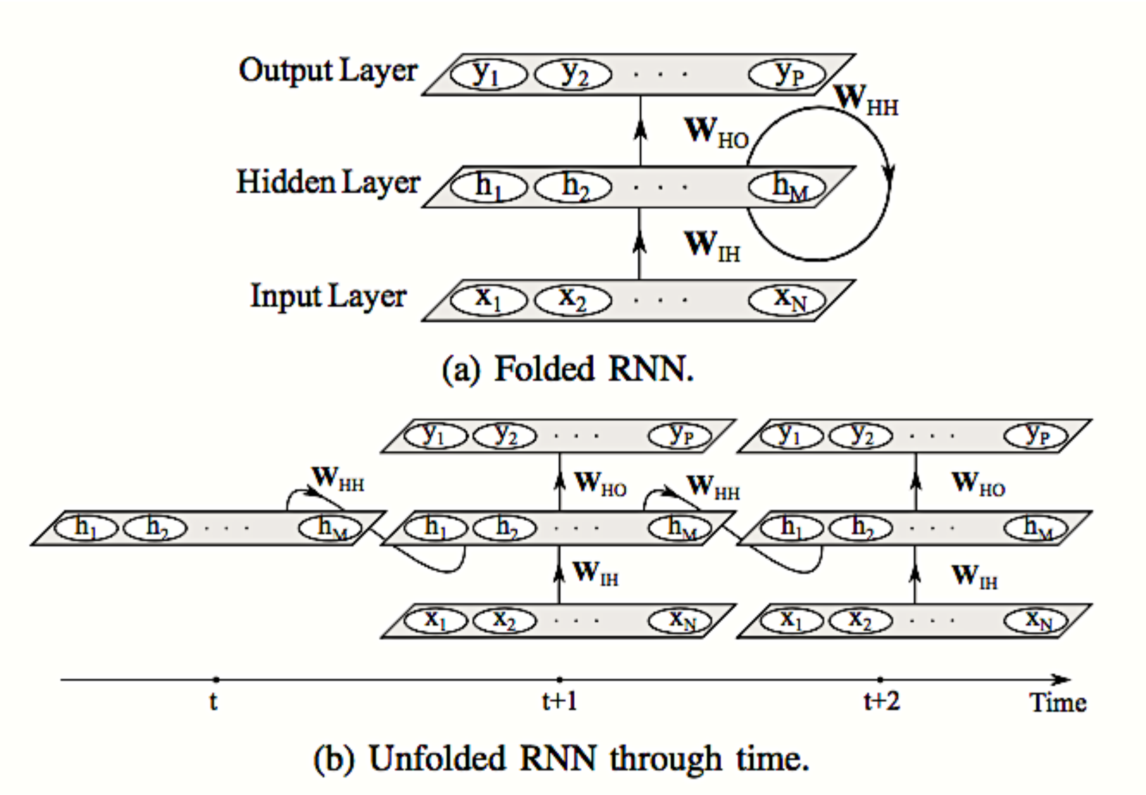


Figure 2-14. A simple recurrent neural network (RNN) and its unfolded structure through time t. Each arrow shows a full connection of units between the layers. To keep the figure simple, biases are not shown.
Source: Salehinejad et al. 2018.

Nonlinear activation functions are more powerful than linear activation functions since they offer non-linear boundaries and non-linear connections between hidden layers of the RNN to learn input-target relationships (Bengio 2007). The activation functions can be chosen based on the problem of interest, readers can refer back to Table (2-4) for relevant activation functions. The ReLU activation function which is computationally cheaper leads to sparser gradients and greatly accelerates the convergence of stochastic gradient descent (SGD) compared to the “sigmoid” or “tanh” activation functions (Krizhevsky 2012).

The loss function evaluates the performance of the network by comparing the output y_t with corresponding target z_t (Salehinejad et al. 2018) which can be defined as

$$\mathcal{L}(y, z) = \sum_{t=1}^{T} (y_t, z_t) \qquad (2.27)$$

which is essentially the overall summation of losses in each timestep (Goodfellow 2016). Now there are many options for hyper-parameters for loss function which depends on the problem of interest, loss functions include the ones referred to in Table (2-3) such as Euclidian distance, Hamming distance for real-values and using cross-entropy for probability distributions and classification problems (Williams and Zipser 1995).

Training RNN’s comes with its own quirks. Properly initializing the network, optimizing the parameters, hyper-parameter tuning, instability caused through hidden layer dependencies have all been identified as points to note (Bengio 1994, Salehinejad et al. 2018). There has been growing literature on various techniques to train RNN's better including extended Kalman Filter (EKF) (Puskorius and Feldkamp 1994), hessian-free expectation maximization (EM) (Ma and Ji 1998), approximated Levenberg-Marquardt (Chan and Szeto 1999), and others (Ruder 2016).

For initialization of RNNs, gaussian draw with standard deviation between 0.001 and 0.01 is considered a reasonable choice (Bengio 2007, Pascanu et al. 2013, Salehinejad et al. 2018). The bias is usually set to zero where the dimensionality of the input data is a highly dependent task for initialization (Sutskever et al. 2013) and setting initialization problem based on problem of interest has to be based on prior knowledge (Bengio 2007).

Now optimization of the RNN can leverage the popular and simple method called **Gradient Descent (GD)**. The essence of GD is that based on each member of weight matrices in the model, finding the error function derivatives with respect to the weights itself (Bengio 2007, Salehinejad et al. 2018). In order to minimize the loss, each weight is changed proportional to the derivative of the error with respect to the weight (given that the non-linear activation functions are differentiable) (Salehinejad et al. 2018). The instance of computing the gradient for the whole dataset in each optimization iteration to perform a single update is called **batch GD**. This update can be written as

$$\theta_{t+1} = \theta_t - \frac{\lambda}{U} \sum_{k=1}^{U} \frac{\partial \mathcal{L}_k}{\partial \theta} \tag{2.28}$$

here U represents the size of training set, λ is the learning rate, and θ is set of parameters. This is a basic approach that is computationally expensive for extremely large datasets and not equipped for online training i.e. where the data is streaming and model updates in real-time (Salehinejad et al. 2018).

In order to accommodate temporal behaviors and memory for long or short during trends, the GD has to be extended through time to train the network, this is called **backpropagation through time (BPTT)** (Werbos 1990). There are many complexities involved with computing error-derivaties through time (Le et al. 2015), capturing dependencies and duration of dependencies (Bengio 2007). Long term temporal dependencies cannot be learnt in RNN's (Bengio 2007), if the derivatives of the loss function with respect to the weights only consider the distance between the current output and the corresponding target, without using the history information for weights updating (P´erez-Ortiz et al. 2003, Salehinejad et al. 2018). Some of these challenges such as the **vanishing gradient problem** occurs due to the exponential decay of gradient is back-propagated through time. In another instance, the back-propagated gradient can exponentially blow-up increasing the variance of the gradients leading to unstable learning situation, this is called the **exploding gradient problem** (Sutskever et al. 2011).

2.7. Long-Short Term Memory (LSTM) RNN

One of the main problems that RNNs suffer from is the vanishing gradient descent problem (Hochreiter 1991, Hochreiter et al. 2001). This problem occurs when the output is between 0 and 1, and since neurons generally have a *sign* or *tanh* function, and if, it keeps getting multiplied by a real small number, which ultimately reduces the learning rate of the network rendering the the network failing to learn long-term sequential dependencies in data. In order to overcome this problem, a variant of the RNN called **Long-Short Term Memory (LSTM)** are the most popular and efficient neural network architecture that includes feedback loops between elements and as the name suggests maintains long-term and short-term memory (Hochreiter and J. Schmidhuber 1999). LSTMs are a special kind of RNN, capable of learning long-term dependencies, where the standard neuron output is feedback into the input and can simulate memory by passing the previous signals through the same nodes (Le et al. 2015). Furthermore, the hidden units now change from “sigmoid” or “tanh” to memory cells, where their inputs and outputs are controlled by gates that control the flow of information to hidden neurons and preserve extracted features from previous timesteps (Hochreiter and J. Schmidhuber 1999, Le et al. 2015, Zhu et al. 2015). LSTM neural networks are suitable for time series analysis because they can more effectively recognize patterns and regimes across different time scales.

LSTMs were introduced at the turn of the century (Hochreiter and Schmidhuber 1997) and later improved (Gers et al. 2000). They are used in various domains; such as: natural language question and answering for novels (Iyyer et al. 2017), handwriting recognition (Graves et al. 2009), and commercial products such as: Google Translate (Wu et al. 2016, Metz 2016), Apple’s Siri (Smith 2016), and Amazon’s Alexa (Vogels 2016). All RNNs have an architecture in the form of a chain of repeating modules of neural network. Nevertheless, in standard RNNs, this repeating module will have a very simple structure; such as: a single *tanh* layer; while in LSTMs

the repeating module has four layers interacting in a very special way instead of a single neural network layer (Hochreiter and Schmidhuber 1999, Zhu et al. 2015)
LSTM can be extremely complex to understand since a typical LSTM has about four times more parameters than a simple RNN (Mikolov et al. 2014). Thereby in order to enhance learning, improve learning long-range dependencies, optimal scheme, many further variants of LSTM's have also emerged that will be discussed briefly later in this subsequent chapter.

LSTM tries to solve the vanishing gradient problem by not imposing any bias towards recent observations, but it keeps constant error flowing back through time. It works essentially in the same way as the Elman Recurrent Neural Network (ERNN) architecture, with the difference that it implements a more elaborated internal processing unit called *cell.* To demonstrate this behavior, the first panel of Figure (2-17) shows a gate of the LSTM neural network where the LSTM neuron is made of four LSTM neural network modules and the four functions are inside one gate. Traditionally one activation function uses the input and the output is spit out; however, in LSTM's these neurons and pathways act as gates. The main idea is that the long line conveyer belt updates the module's state which allows to keep long or short-term memory. These gates essentially take an input x_t to decide how much of that input should be used to update the module state. The three *sign* gates and the *tanh* gate show how the input state determines the update, where a number "1" updates the whole module state and the numbers smaller than 1 updates the module state by respective increments. As explained, LSTMs are explicitly designed to avoid the long-term dependency problem as remembering information for long periods of time is their default behavior, not something they struggle to learn. It is worth noting that in figure (2-17), each line carries an entire vector, from the output of one node to the inputs of others. The pink circles represent pointwise operations, like vector addition, while the yellow boxes are learned neural network layers. Lines merging denote concatenation, while a line forking denotes its content being copied and the copies going to different locations.

LSTM has attracted a great attention for time series data recently and yielded significant improvement over RNNs. Since LSTMs take inputs in a sequence, like time series data, the temporal dynamics that connects the data is more important than the spatial content of each individual frame. Moreover, since the input and output are both variable, i.e. it can input a sequence of numbers, and the output is a sequence (many to many) similar to machine learning, the M2M does not have to be the same sequence.

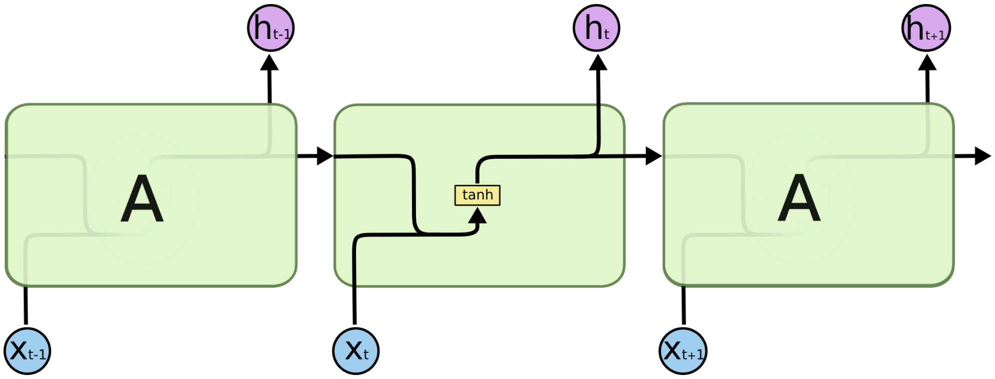


Figure 2-17. The repeating module in a standard RNN contains a single layer
Source: Olah 2015.

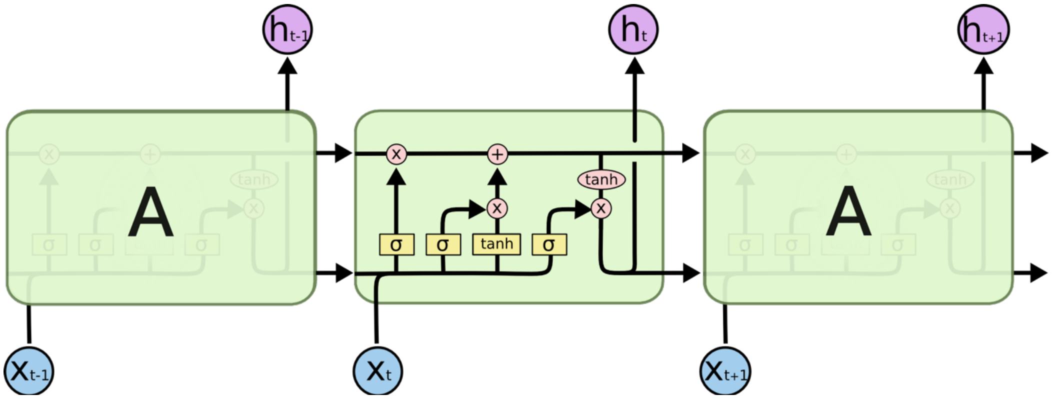


Figure 2-18. The repeating module in an LSTM contains four interacting layers.

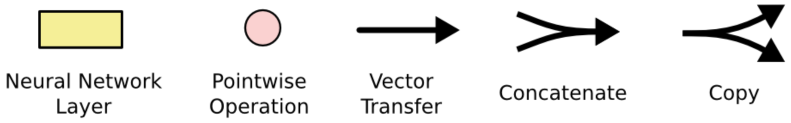


Figure 2-19. Sign Denotation in LSTM RNN
Source: Olah 2015.

LSTMs also have this chain like structure, but the repeating module has a different structure. Instead of having a single neural network layer, there are four, interacting in a very special manner. In the above diagram, each line carries an entire vector, from the output of one node to the inputs of others. The pink circles represent pointwise operations, like vector addition, while the yellow boxes are learned neural network layers. Lines merging denote concatenation, while a line forking denotes its content being copied and the copies going to different locations.

The Conveyer Belt Model Data Flow

The cell state is kind of like a conveyor belt. It runs straight down the entire chain, with only some minor linear interactions. It's very easy for information to just flow along it unchanged. The LSTM does have the ability to remove or add information to the cell state, carefully regulated by structures called gates. Gates are a way to optionally let information through. They are composed out of a sigmoid neural net layer and a pointwise multiplication operation.

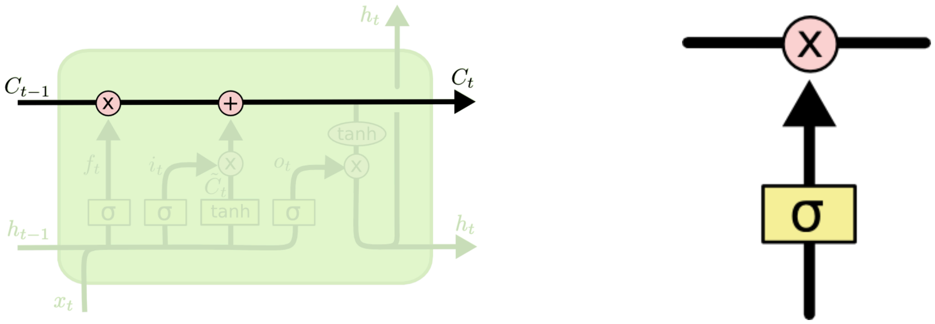


Figure 2-20. Conveyer Belt and Gates
Source: Olah 2015.

The *sigm* gates indicate how much of that input should be in module state, if it is 0 then no update, 1 then complete module update. Essentially update the module state and allows the module to keep a long-term memory of what is going on.

The memory cells for the hidden states are introduced formally The cell state is like a conveyor belt (Olah 2015); it runs straight down the entire chain, with only some minor linear interactions. It's very easy for information to just flow along it unaltered as the LSTM has the ability to remove or add information to the cell state which are carefully regulated by structures called gates. Gates are a way to optionally let information through and composed out of a sigmoid neural net layer and a pointwise multiplication operation (Graves 2014, Bianchila et al. 2017).

The first step in this process is to split the training and test windows. In finance and economic time series the **training window** can be either 120, 200, 500 days for daily data like Stock Prices predictions, or 5, 8, 12 for annual data. Similarly, the **test window** can be 10, 20, 30 for daily data, or 2, 3, 5 for annual data. The training sample is split into *n* samples which iterates over consecutive test window samples in the training window.

(LSTM) cell architecture shown in Figure (2-21) uses purpose-built memory cells to store information which is better at finding and exploiting long range dependencies in the data. It is composed of input, forget, and ouput and cell activation components. For the version of LSTM used in earlier papers (Sutskever et al. 2016, Azouni and Poujell 2017, Bao et al. 2017, Cohan 2015, Bernal et al. 2015, Dindi et al. 2015, Salehinejad et al. 2018), the hidden layer function $\mathcal{H}$ is implemented by the following composite functions:

The input gate g_t^i is defined as

$$g_t^i = \sigma\left(W_{Ig^i}x_t + W_{Hg^i}h_{t-1} + W_{g^c g^i}{}_{g_{t-1}^c}c_{t-1} + b_{g^i}\right), \qquad (2.29)$$

where σ is the logistic sigmoid function, W_{Ig^i} is the weight matrix from the input layer to the input gate, W_{Hg^i} is the weight matrix from the hidden state to the input gate, $W_{g^c g^i}$ is the weight matrix from the cell activation to the input gate, b_{g^i} is the bias of the input gate (Salehinejad et al. 2018).

The forget gate g_t^f is defined as

$$g_t^f = \sigma\left(W_{Ig^f}x_t + W_{Hg^f}h_{t-1} + W_{g^c g^f}{}_{g_{t-1}^c} + b_{g^f}\right), \qquad (2.30)$$

where σ is the logistic sigmoid function, W_{Ig^f} is the weight matrix from the input layer to the forget gate, W_{Hg^f} is the weight matrix from the hidden state to the forget gate, $W_{g^c g^f}$ is the weight matrix from the cell activation to the forget gate, b_{g^f} is the bias of the forget gate (Salehinejad et al. 2018).

The cell gate g_t^c is defined as

$$g_t^c = g_t^i \tanh\left(W_{Ig^c} x_t + W_{Hg^c} h_{t-1} + b_{g^c}\right) + g_t^f g_{t-1}^c , \qquad (2.31)$$

where tanh is the tanh activation function, W_{Ig^c} is the weight matrix from the input layer to the cell gate, W_{Hg^c} is the weight matrix from the hidden state to the cell gate, b_{g^c} is the bias of the cell gate (Graves 2014, Bianchila et al. 2017).

The output gate g_t^o is defined as

$$g_t^o = \sigma\left(W_{Ig^o} x_t + W_{Hg^o} h_{t-1} + W_{g^c g^o} g_t^c + b_{g^o}\right), \qquad (2.34)$$

where W_{Ig^o} is the weight matrix from the input layer to the output gate, W_{Hg^o} is the weight matrix from the hidden state to the output gate, $W_{g^c g^o}$ is the the weight matrix from the cell activation to the output gate, b_{g^c} is the bias of the output gate (Graves 2014, Bianchila et al. 2017, Bao, et al. 2017, Salehinejad et al. 2018).

All the above are the same as the hidden vector h or the hidden state h_t computed as

$$h_t = g_t^o \tanh(g_t^c) . \qquad (2.32)$$

First there is a sigmoid function that decides what to forget and what to ignore by taking in the new input x_t and the state of the previous output and deciding if it wants to create a memory and transfers to next cell. In the hidden state, there are both slopes and valleys in the stock price movement or political risk movements, non-convex optimization; moreover, it is important to account for long term dependencies, for example the 2008-09 financial crisis period so the LSTM or GRU knows how the on-set sequence of the "bad" performance period.

Furthermore, as the previous state is running through the *cell*, three sigmoid layers: for 1, in order for the info to go through, and 0 when you don’t want the information to go through. First sigmoid function decides to forget ignore or remember; second sigmoid layer, is a gate for state to update and decide what values to insert into the new update; third, sigmoid layer performs the actual forgetting and retaining memory. Finally, an output from the *cell* is produced whose state is the activation multiplied by the weights.

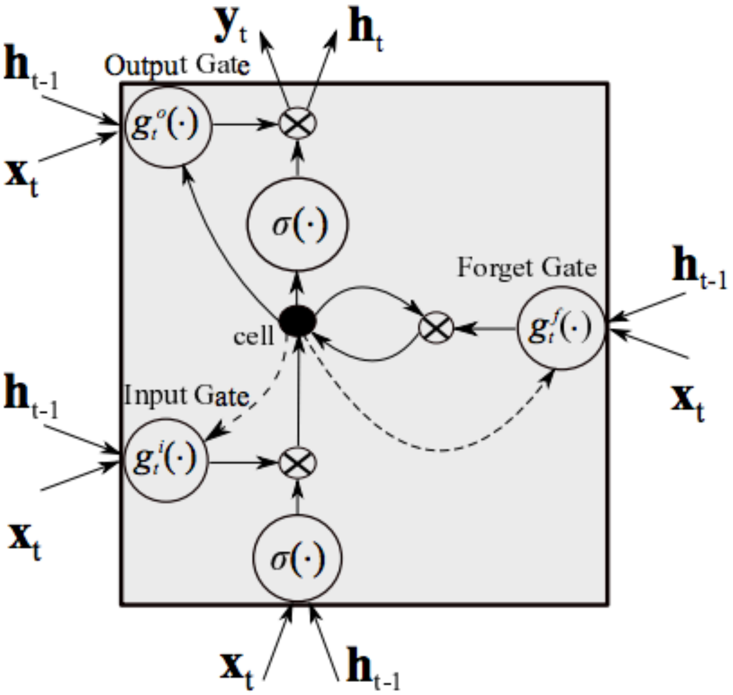


Figure 2-21. The LSTM memory block with one cell. The dashed line represents time lag.
Source: Salehinejad et al. 2018.

Stacked LSTM

The capacity of LSTM RNN can be increased by increasing the depth or stacking different hidden layers with LSTM cells (Graves et al. 2013, Kalchbrenner et al. 2015). In a stack of *L*, a hidden layer *l* in this stack is defined as

$$h_t^l = f_H\left(W_{IH}h_t^{l-1} + W_{HH}h_{t-1}^l + b_h^l\right), \tag{2.33}$$

where h_t^l is the hidden vector that is computed over time $t = (1, \dots, T)$ for $l = (1, \dots, L)$, and the initial sequence of hidden vector is defined on the input sequence $h^0 = (x_1, \dots, x_T)$ (Graves et al. 2013).

In order for LSTM to maintain constant space and time embeddings of the stack content and in a controlled contracture, a stack pointer controller can push to and pop from the top of the stack to maintain temporal order (Yao et al. 2015, Ballesteros et al. 2015). The output of the stacked LSTM network is

$$y_t = f_O(W_{HO} h_t^L + b_0). \qquad (2.34)$$

Similar to machine learning, there are variants of LSTM and RNN such as **Gated Recurrent Units (GRU)** and **Bi-directional (BRNN)** which are very similar to **LSTMs** and are described briefly in Table (2-2) and subsequent sub-sections. The variations of RNN's and LSTM's are an active area of research in computer science and statistics. The GRU has performed as well as LSTMs and a comparative study is documented for short-term load forecasting (Bianchi et al. 2017). Moreover, original GRU RNN model have reduced the computational costs (Dey and Salem 2017, Lipton et al. 2015), by classifying emotions through speech recognition using smartphones (Rana et al. 2016) using different classes of state-of-the-art RNN's.

Gated Recurrent Unit (GRU)

The gated recurrent units (GRU's) capture the dependencies over different time scales by gating the units to modulate the flow of information inside the unit, without having memory cells like the LSTM (Chung et al. 2014). Unlike the LSTM, GRU exposes the whole state at each time step to compute a linear sum between the existing state and the newly computed state (Cho et al. 2014). A simple block diagram representation of the GRU is presented in Figure (2-22). The linearly modeled activation in a GRU can be

modeled as

$$h_t = (1 - z_t)h_{t-1} + z_t\tilde{h}_t, \tag{2.35}$$

where z_t is the update gate controls the value of the activation which then defined as

$$z_t = \sigma(W_z\, x_t + U_z\, h_{t-1})\,, \tag{2.36}$$

where W and U are the weight matrices to be learnt (Cho et al. 2014, Chung et al. 2014). The activation function is defined as

$$\tilde{h}_t = tanh(W_h\, x_t + U_h(r_t \odot \quad h_{t-1}))\,, \tag{2.37}$$

here r_t is a set of rest gates defined as

$$r_t = \sigma(W_r\, x_t + U_r\, h_{t-1}) \tag{2.38}$$

This allows the units in the network to forget to forget the first state by reading the first symbol of an input sequence. There are few studies with thorough comparison between LSTM and GRU's (Chung et al. 2014). Nevertheless, all such models must be used in a systematic manner to learn from them and use the gained information in a useful manner to directly aid decision-making.

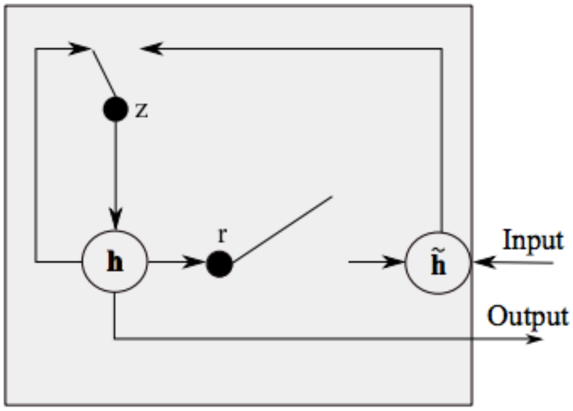


Figure 2-22. A gated recurrent unit (GRU). The update gate ***z*** decides if the hidden state is to be updated with a new hidden state h˜. The reset gate ***r*** controls if the previous hidden state needs to be ignored.
Source: Salehinejad et al. 2018.

Bidirectional LSTM (BLSTM)

BLSTM neural networks were invented to increase the amount of input information when context is necessary (Schuster and Paliwa 1997). BRNN’s have been applied to protein structure prediction (Pollastari et al. 2005), translations (Sundermeyer 2014), and generating news headlines (Lopyrev 2015). The reasoning behind BLSTM is that traditional RNN’s only incorporate historic information and not the future context (Graves et al. 2013). BLSTM considers all the past and future state to estimate the output vector (Schuster and Paliwal 1997). In this instance, there are different RNN’s at play, where one RNN goes forward in time from beginning to end, another RNN goes backwards in negative-time direction i.e. from end to beginning, with no interaction between the outputs of the two networks (Schuster and Paliwal 1997). If the forward and backward hidden sequences were denoted as $\vec{h}_t$ and $\overleftarrow{h}_t$ respectively, Figure (2-23) shows a depiction of unfolding through time of BLSTM.

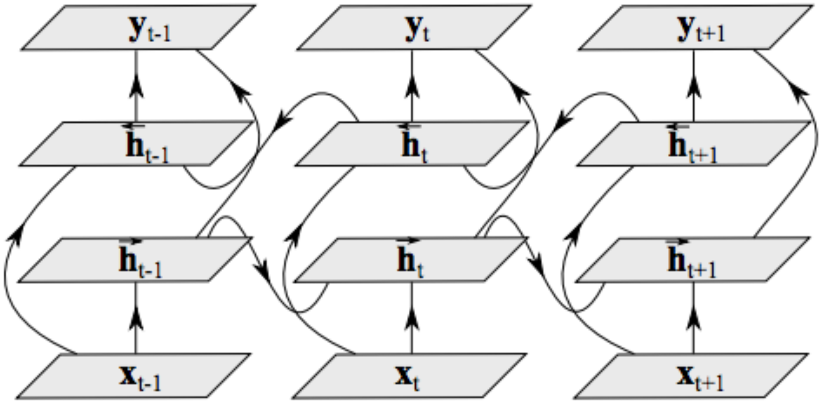

$\mathbf{y}_{t-1}$
$\mathbf{y}_{t}$
$\mathbf{y}_{t+1}$
$\overleftarrow{\mathbf{h}}_{t-1}$
$\overleftarrow{\mathbf{h}}_{t}$
$\overleftarrow{\mathbf{h}}_{t+1}$
$\overrightarrow{\mathbf{h}}_{t-1}$
$\overrightarrow{\mathbf{h}}_{t}$
$\overrightarrow{\mathbf{h}}_{t+1}$
$\mathbf{x}_{t-1}$
$\mathbf{x}_{t}$
$\mathbf{x}_{t+1}$

Figure 2-23. Unfolded through time bi-directional recurrent neural network (BRNN). Source: Salehinejad et al. 2018.

Now there are many variants of LSTM models. Even though LSTM's have proved versatile and efficient to deal with long-term sequential dependencies, it has been argued that the gating mechanism has no clear-cut method to discriminate between salient and non-salient information sequence (Veeriah et al. 2015). For example, vanilla LSTM's do not capture spatio-temporal dynamics in robotics or action recognition (Veeriah et al. 2015). **Differential recurrent neural networks (dRNNs)** refer to detecting and capturing of important spatio-temporal sequences to learn dynamics of actions in input (Veeriah et al. 2015). **Multi-dimensional LSTM (MDLSTM)** uses interconnection from previous state of cell to extend the memory of LSTM along every *N* dimensions as compared to one-dimensional LSTM (Graves et al. 2007, Graves et al. 2009). Similarly, there are many other variants of the LSTM and RNN neural network family that are designed more specifically for a given domain.

Another promising variant to LSTMs is the applications of **Reinforcement learning (RL)** where the goal is to choose a course of successive actions in order to maximize the final (or cumulative) reward. RL has only recently been applied to complex multi-agent problems involving stochastic games, deep machine learning solution to the portfolio management problem, and to inform real-time financial trading strategies (Jiang et al. 2017, Bertoluzzo and Corazza 2012, Hughes 2014). Another equally promising variant is the proliferation of **Generative Adversarial Networks (GAN's)** in which two neural networks contest with each other in a zero-sum game framework. GAN's were introduced by Ian Goodfellow et al. (2014) and have also been applied to discover disentanglement in order to separate market behavior from specific stock's movements (Hadad et al. 2017).

L1 and L2 Regularizer

Regularization refers to the process of controlling neural network capacity to help avoid overfitting issues. During training of RNN's, a portion of data is considered as validation set which is used to watch the training procedure and prevent the network from underfitting or overfitting (Bishop 2006). Regularizors are used to successfully train RNNs (Srivastava 2013).

The L_1 and L_2 regularization method adds a term to the loss function that penalizes certain parameter configurations to prevent the network from overfitting. The updated loss function with the regularization term is written as

$$\mathcal{L}(y,z) = \mathcal{L}(y,z) + \eta\|\theta\|_p^p \,, \qquad (2.39)$$

where η is the set of the neural network parameters (weights) and η controls the relative importance of the regularization parameter.

The network parameters can be abstracted as

$$\|\theta\|_p = \left(\sum_{j=0}^{|\theta|} |\theta_j|^p\right)^{1/p} \qquad (2.40)$$

where *p = 1* corresponds to the regularizer L_1 and *p = 2* corresponds to the regularizer L_2. The L_1 is the sum of the weights, whereas the L_2 is the sum of square of the weights. Readers can refer to Appendix E for more details.

Dropout

Dropout, as the name suggests omits a fraction of the connections between any two layers of the network during training. Dropout is used in many models in deep learning as a way to avoid over-fitting (Srivastava et al. 2014). Dropout for the hidden layer outputs in eq. (2-24) will yield

$$h_t = k \odot h_t, \qquad (2.41)$$

where **k** is the binary vector mask that can also apply withdrawal that follows a statistical pattern and $\odot$ is the element wise product (Pham et al. 2014). Dropouts for RNN were introduced as a single dropout mask at the beginning of each training sequence is called RNNDrop (Moon et al. 2015). This may be adjusted over the sequence allowing the network connections to remain constant over time. Other research has suggested that the RNNDrop can be applied by dropping previous hidden states of the network, thereby masking samples at every input sequence per step (Semeniuta et al. 2016).

2.8. LSTM Example

A very simple example of recurring pattern is financial time-series data; specifically speaking, the most volatile are cryptocurrencies like Bitcoin, is used to show the above behavior. A traditional neural network will struggle to capture such dynamic stochastic movement, since it does not have the history to map the whole function. For the LSTM model, first, a split into training and test windows based on a window size of seven days can be set for this model. This means that the model will take every seven-days data iteratively to spit the *t+1* i.e. the 8th day forecast, sliding with one each time. The last two panels in Figure (2-24) show this is what gets fed into the model, the network zooms into the window size to extrapolate next time sequence output given a set of windows. The training period mostly consists of periods when bitcoins were relatively cheaper. As such, the training data may not be representative of the test data, undermining the model's ability to generalize for the unseen data. The LSTM network is fed a total of 12 features that are depicted in the table (2-6) below the chart; these features include: closing price of bitcoin, total volume of bitcoin, total bitcoins, miners' revenue, bitcoin exchange rate vs. USD, bitcoin average block size, bitcoin difficulty, bitcoin median transaction confirmation time, bitcoin hash rate, and number of transaction per block. Furthermore, the model's data is arranged in order of earliest to latest and the LSTM model will use previous data to predict the next day's closing price of bitcoin.

Data frames consisting of seven consecutive days of data, called windows, has been built, so the first window will consist of the 0-6th rows of the training set (Python is zero-indexed), the second will be the rows 1-7…etc. By picking a small window size means (more windows can be fed into the model) there may be a downside that the model may not have sufficient information to detect complex long-term behaviors. Next, the built model's functions construct an empty model, unimaginatively, called model (model = Sequential), to which an LSTM layer is added. That layer has been shaped to fit the inputs (*n* x *m* tables, where *n* and *m* represent the number of time points/rows and columns, respectively). In addition, the function includes more generic neural network features. To see how well the model performs, the performance on the training set is examined first (data before June 2017) through the model's mean absolute error (MAE) on the training set after the 80th training iteration (or epoch), which is represented by the number below the code.

Another robust capability of the model is that it can access the source of its error and adjust itself accordingly; in fact, it is not hard to attain almost zero training errors. Furthermore, the Dropout is included in the built model's function to mitigate this risk for a relatively small model. From the results, LSTM model seems to have performed well on the unseen test set as the predicted values regularly mirror the previous values (e.g. October). Finally, with the addition of more features can help reduce the mean absolute error and this model will provide better prediction for the downward trend in the bitcoin.

The key attributes of cases in our domains of interest are non-linearities in time, non-homogenous sample, context dependencies of time-trends, memory of the past, long-term and short-term dependencies, highly dynamic and interconnected markets. For example, the memory of time-trend from the 2008 financial crisis may be important to remember, or the geopolitical or financial market trends from last week could have significant impact in the following days. In order to address various financial, economic, and geopolitical data and decision-making environments, a flexible framework for high-frequency, daily, weekly, or even annual data, LSTM and variants of RNN architectures are the appropriate choices for modeling time-series predictions with time-varying dependencies; whereas DTW-HCA ensemble offer a meaningful way to cluster entities and domains based on extracting time-dependent similarity.

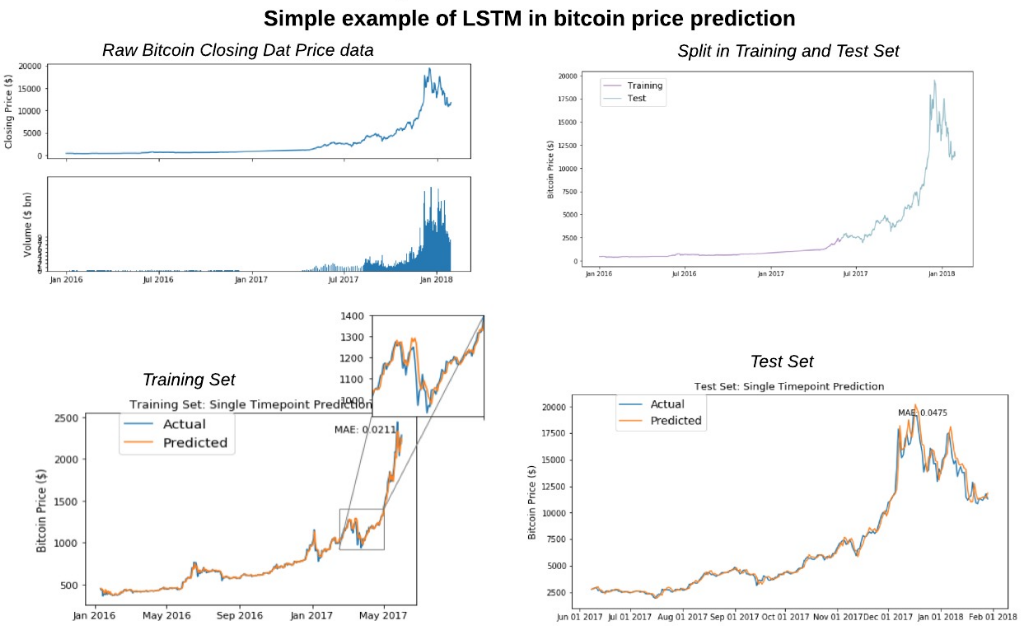


Figure 2-24. LSTM Module architecture, and simple example of LSTM to predict Bitcoin prices

Table 2-6. Features used for training LSTM model to predict Bitcoin prices

Feature	Description
Closing Price	Price at which a bitcoin is traded on a given trading day.
Volume	Sum of all the transactions.
Total Bitcoins	The total number of bitcoins which have been mined.
Miners Revenue	((number of bitcoins mined per day + transaction fees) * market price)
Volume (BTC)	
Volume (USD)	
Weighted Price	
Average Block Size	The Average block size in MB
Difficulty	Difficulty is a measure of how difficult it is to find a hash below a given target.
Median Transaction Confirmation Time	The Daily Median time take for transactions to be accepted into a block.
Hash Rate	The estimated number of giga hashes per second (billions of hashes per second) the bitcoin network is performing.
Number of Transaction per Block	The average number of transactions per block.

2.9. Uncertainty Quantification in Neural Networks

Traditional neural networks are fraught with a big missing element: quantification of prediction uncertainty. The different types of uncertainty are of great importance in many fields; such as: AI safety (Amodei et al. 2016), and autonomous decision making, where the model's epistemic uncertainty can be used to avoid making uninformed decisions with potentially life-threatening across domains, for example, diagnostic and prognostics in the healthcare industry (Krizhevsky et al. 2012, Long et al. 2014); autonomous driving vehicles to identify and detecting high level cognitive tasks such as sensory signal input (Bojarski et al. 2016) as well as low-level cognitive task analysis such as distinguish the white side of a turning trailer from a bright sky, which led to the first fatality of an assisted driving system (NHTSA 2017); post office sorting letters according to their zip code (LeCun and Cortes 1998, LeCun et al. 1998); or nuclear power plant with a system responsible for critical infrastructure (Linda et al. 2009).

In economic and financial decision-making; such as: high frequency trading, computers could destabilize national or global financial market crisis (Gal 2015), the risks of not accounting for epistemic uncertainty could be catastrophic. In our domains of interest, human error can creep into a financial trading decision leading to an ill-judged one (Gal 2014). As systems get more automated, especially machine learning based systems, the variable degrees of automation (DOA) can override human expert decisions (Li 2017, Leaver and Reader 2014). Such automated systems can lead to sub-optimal real-world results when machines interact with humans; for instance, even if the prediction is accurate on some days, it may be inaccurate on other days. Hence, the work on uncertainty and its applications should have a prime importance since decision makers need to know the uncertainty around the predictions to help make the best decisions. Nevertheless, most deep learning tools operate in a very different setting to the probabilistic models which possess this invaluable uncertainty information (Gal 2015). Moreover, uncertainty quantification in deep learning has focused primarily on the classification problem (Bertozzi 2017, Gal 2018, Lakshminarayanan 2016, Weidman 2016). For example, how sure is the model whether the image is that of a cat or a dog; however, when the model is given a new animal; such as: a polar bear, the uncertainty for the model's prediction is not well studied. Furthermore, limited work has been conducted on time-series deep learning with uncertainty quantification; especially, in the domains of interest. Consequently, this section will discuss how to bring transparency to the decision-making process concerning how and which algorithms can lead to a decision-maker's awareness concerning forecast accuracy by incorporating uncertainty quantification.

Incorporating prediction interval (PI) estimation in domains of finance, economics, and geopolitics, using deep LSTM or RNN-variant algorithms would greatly benefit the decision makers by providing the reliability of the prediction. In relation to neural networks, the literature has been focused on image related problems pertaining CNN's (Gal 2012, Gal 2013, Zhu et al. 2018, Tirpathy et al. 2018). RNN or LSTM models have also been used for uncertainty quantification but primarily in the context of text and sentiment analysis (Gal 2014, Gal 2015). Thus, opening up deep LSTM for such uncertainty quantification is a highly dynamic and new area of research; nevertheless, reliable and accurate uncertainty quantification for time-series predictions remains critical.

The nascent yet growing body of literature has put forward many different approaches for confidence interval estimation in the confines of LSTM's. There are different schools of thought including: Classical, Bayesian, Bootstrap methods, Delta method, Mean-Variance Estimation (MVE), and Quantile methods. A thorough assessment of various methods indicates that the Bootstrap method had the smallest width of confidence interval and are best in terms of low computation costs; however, Bayesian methods are best in terms of quality and repeatability (Khosravi et al. 2010; Saleum 2014) albeit being computationally expensive. Other methods such as the quantile recurrent forecasters (Wen et al. 2017) have tremendous versatility that does not have a comparison in the literature. In addition, the emerging and nascent literature examining the combination of Bayesian modeling and neural networks is also promising. In classical statistical modeling, probabilistic conjugation has been used in the past; however, only recently, attention has been garnered towards novel end-to-end Bayesian deep learning with LSTM (Zhu and Laptev 2017, Khosravi et al. 2015, Gal 2015). Although deep neural networks, such as: LSTM, are being used extensively, related fields of representing model uncertainty is still at an early stage of development and has crucial importance for the future of the field (Krzywinski and Altman. (2013). Ghahramani, 2015). Bayesian uncertainty is rapidly shifting this field of deep learning (Herzog and Ostwald 2013, Trafimow and Marks 2015, Nuzzo 2014). Standard deep learning for regression problems do not capture model uncertainty. As a result, the following section briefly reviews the key uncertainty quantification measures and proposes a simpler, yet equally efficient, implementation of uncertainty quantification for deep neural networks.

2.9.1. Classical Uncertainty Prediction

It is often assumed that targets can be modeled by the following equation:

$$t_i = y_i + \epsilon_i \tag{2.42}$$

where t_i is the i-th measured target, and ϵ_i is the noise, also called error, with a zero expectation. The error term moves the target away from its true regression mean, y_i , toward the measured value, t_i (Khosravi et al. 2010).

In all PI construction methods discussed in this research, it is assumed that errors are independently and identically distributed (Khosravi et al. 2010). In practice, an estimate of the true regression mean is obtained using a model, $\hat{y}_i$ leading to the following equation:

$$t_i - \hat{y}_i = [y_i - \hat{y}_i] + \epsilon_i \tag{2.43}$$

Confidence Intervals (CIs) deal with the variance of the first term in the right-hand side of (2.45). They quantify the uncertainty between the prediction, $\hat{y}_i$, and the true regression, y_i. CIs are based on the estimation of characteristics of the probability distribution $P(y_i|\hat{y}_i)$. In contrast, PIs try to quantify the uncertainty associated with the difference between the measured values, t_i, and the predicted values, $\hat{y}_i$ (Khosravi et al. 2010). This relates to the probability distribution $\mathrm{P}(t_i|\hat{y}_i\ )$. Accordingly, PIs will be wider than CIs and will enclose them. If the two terms in (2) are statistically independent, the total variance associated to the model outcome will become,

$$\sigma_i^2 = \sigma_{\hat{y}i}^2 + \sigma_{\hat{\epsilon}i}^2 \tag{2.44}$$

The term $\sigma_{\hat{y}i}^2$ originates from the model's misspecification and parameter estimation errors, $\sigma_{\hat{y}i}^2$ is the measure of noise variance. Upon proper estimation of these values, PIs can be constructed for the outcomes of ANN models.

A comprehensive analysis of all the methods based on the quality of constructed PIs, repeatability of results, computational requirements, and the variability of PIs against the data's uncertainty was conducted by Khosravi et al. (2010).

Furthermore, multi-step, long-horizon forecasts are usually needed in conjunction with precise prediction intervals in order to be able to quantify forecast uncertainties and the risks in the decision-making process (Wen et al. 2017). Therefore, many decision-making scenarios require richer information provided by a probabilistic forecast model which returns the full conditional distribution $p(y_{t+k}|y_{:t})$, rather than just a point forecast model that predicts the conditional mean $\mathbb{E}(y_{t+k}|y_{:t})$. And for real-valued time series, this is traditionally achieved by assuming an error distribution, usually Gaussian, on the residual series $\epsilon_t = y_t - \hat{y}_t$ (Wen et al. 2017).

Based on the above, and by taking a trained neural network $f^{\widehat{W}}(.)$ where $\widehat{W}$ are fitted parameters and new sample data x^{*}, the goal is then defined as the uncertainty of the model prediction

$$\hat{y}^{*} = f^{\widehat{W}}(x^{*}) \tag{2.45}$$

The approximate α-level prediction interval can be constructed like this

$$[\hat{y}^{*} - z_{\propto}/2\eta,\ \hat{y}^{*} + z_{\propto}/2\eta,] \tag{2.46}$$

with the quantification of η which is the prediction standard error (Zhu and Laptev 2017).

2.9.2. Mean-Variance Estimation (MVE)

In the MVE method, the PI's can be constructed if the mean and variance parameters are known and the errors are distributed around the target mean *y*(*x*) (Nix and Wiegend 1994). Figure (2-25) shows a schematic of the MVE method.

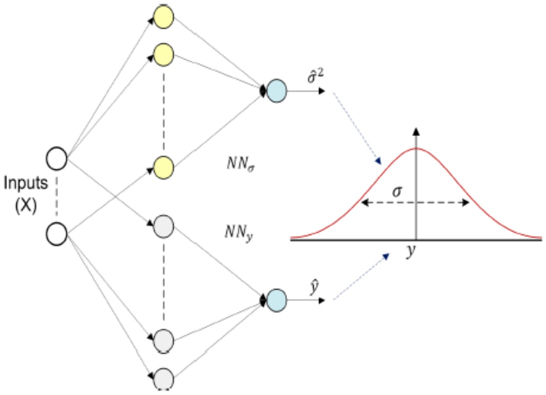


Figure 2-25. Abstraction of Mean Variance Estimation (MVE) for Uncertainty Quantification (UQ). *Source*: Khosravi et al. 2010)

By implementing an exponential activation function, a strictly positive variance estimates is guaranteed (Khosravi et al. 2010). Therefore, assuming that the NN provides accurate and reliable forecast point estimates for *y*(*x*), the approximation of PI with (1 - α)% confidence level can be constructed with w_y and w_σ as the NN parameters to estimate $\hat{y}$ and $\hat{\sigma}^2$ respectively, and σ_i is apriori

$$\hat{y}\left(x, w_y\right) \pm z_{1-\frac{\alpha}{2}} \sqrt{\hat{\sigma}^2(x, w_\sigma)} \qquad (2.47)$$

Now assume a gaussian normal distribution around y_i, the conditional distribution is then defined as:

$$P\left(t_i \middle| x_i, NN_y, NN_\sigma\right) = \frac{1}{\sqrt{2\,\pi\hat{\sigma}_i^2}}\, e^{-\frac{(t_i - \hat{y}_i)^2}{2\hat{\sigma}_i^2}} \qquad (2.48)$$

Consequently, the cost function is then defined by taking the natural logs of the above distribution while ignoring the constant terms to minimize all the samples

$$C_{MVE} = \frac{1}{2}\sum_{i=1}^{n}\left[\ln(\sigma_i^2) + \frac{(t_i - \hat{y}_i)^2}{\hat{\sigma}_i^2}\right] \tag{2.49}$$

2.9.3. Bootstrap

Bootstrap is one of the most common methods deployed for CI and PI using ANN's. It has been proposed that an ensemble of NN's will predict the target with lower bias than a single model (Giordano et al. 2007). The Bootstrap method captures the model misspecification, $\sigma_{\hat{y}}^2$, through M different NN models as shown in figure (2-26) below.

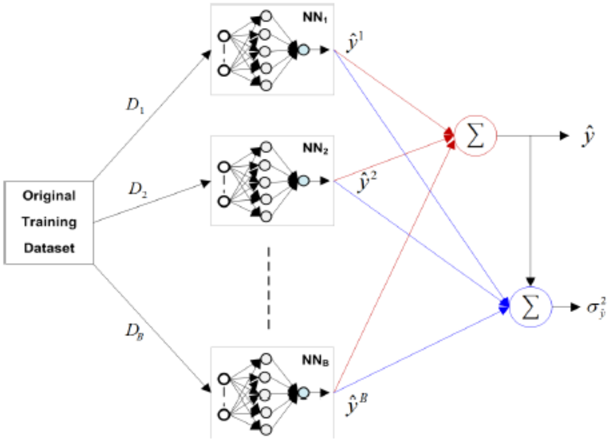


Figure 2-26. A schematic of *M* NN models used by Bootstrap method
Source: Khosravi et al. 2010.

In the above figure, the true regression mean is estimated by averaging the point forecasts of *M* models, with $\hat{y}_i^m$ is the prediction of i-th sample generated by m-th bootstrap model which can be defined as:

$$\hat{y}_i = \frac{1}{M}\sum_{m=1}^{M} \hat{y}_i^m \tag{2.50}$$

The model misspecification variance $\sigma_{\hat{y}_i}^2$ is then estimated using the variance of *M* model outcomes in the following manner:

$$\sigma_{\hat{y}_i}^2 = \frac{1}{M-1}\sum_{m=1}^{M} (\hat{y}_i^m - \hat{y}_i) \tag{2.51}$$

As a result, the CI can be constructed based on (Dybowski and Roberts 2000) to get an approximation of $\sigma_{\hat{y}_i}^2$ and the PI's can be estimated using the variance of the errors $\sigma_{\hat{\epsilon} i}^2$

$$\sigma_{\hat{\epsilon} i}^2 \simeq E\,[(t - \hat{y}_i)^2] - \sigma_{\hat{y}}^2 \tag{2.52}$$

Furthermore, based on (Tibshirani 1996, Hagan and Menhai 1994, Dybowski and Roberts 2000, Khosravi et al. 2010, Khosravi et al. 2010), the variance squared residual r_i^2 are estimated, and these residuals are linked to form a new dataset D_{r^2} where a new NN model can be used to estimate the unknown values of $\sigma_{\hat{\epsilon} i}^2$ and maximize the probability of observing those sample in the new dataset D_{r^2}. This can be represented as:

$$r_i^2 = \left((t_i - \hat{y})^2 - \sigma_{\hat{y}_i}^2 ,\ 0\right) \tag{2.53}$$

$$D_{r^2} = \{x_i, r_i^2\}_{i=1}^{n} \tag{2.54}$$

Finally, the training cost function is defined as C_{BS} where the minimization can be conducted in various ways including traditional gradient descent methods

$$C_{BS} = \frac{1}{2}\sum_{i=1}^{n}\left[\ln\left(\sigma_{\hat{\epsilon}i}^2\right) + \frac{r_i^2}{\sigma_{\hat{\epsilon}i}^2}\right] \tag{2.55}$$

Once the model has been trained offline, the online computation load for PI construction is only limited to 1 + *M* forecasts (Rivals and Personnaz 2000) which is less complex than the MVE or Bayesian methods which need to calculate matrices and derivates.

2.9.4. Bayesian Method

Bayesian Neural Networks (BNN's) were introduced to measure uncertainty in deep learning models from a Bayesian perspective. Network parameters *W* are given a prior with the goal of finding the posterior distribution of *W* instead of point estimate (Zhu and Laptev 2017). However, there are still limitation to this method stemming from (a) highly non-linear behavior, (b) non-conjugacy, and (c) traditional Bayesian inference does not scale to large set of parameters. Most recent studies have focused on lower-bound estimation using variational inference by utilizing different techniques; such as: probabilistic back-propagation or variational Bayes (Paisley et al. 2012; Kingma and Welling 2014; Hernandez-Lobato and Adams 2015; Blundell et al. 2015; Fortunato 2017). Moreover, new algorithms have extended the approximation framework to α-divergence optimization (Gal and Ghahramani 2016a; Gal and Ghahramani 2016b; Gal 2016; Li and Gal 2017). In addition, more recent studies, inspired by (Gal and Ghahramani 2016a; Gal 2016), using MC dropout framework that require no change to existing models, estimates the model's uncertainty using sample variance of the model prediction and stochastic dropouts after each hidden layer form a random sample for the posterior distribution (Zhu and Laptev 2017; Gal 2016).

As denoted earlier, neural networks as $f^{\widehat{W}}(.)$, where $\widehat{W}$are the model parameters and f is the network architecture. A gaussian prior is usually assumed (Zhu and Laptev 2017) for the weight parameters

$$W \sim N(0, I) \qquad (2.56)$$

In regression-based models, the data-generating distribution $p(y|f^{W}(x)\ )$ is assumed as follows

$$y \mid W \sim f^{W}(x), \sigma^2 \qquad (2.57)$$

Based on a pair of N observations of $X = \{x_1, \dots, x_N\}$, and $Y = \{y_1, \dots, y_N\}$ in

Bayesian inference, the optimal posterior distribution over model parameters $p(W|X,Y)$ is found with new estimates of x^*; and the prediction distribution is obtained by marginalizing the posterior distribution (Zhu and Laptev 2017)

$$p(y^*|x^*) = \int_W p\left(y^*\middle|f^W(x^*)\right) p(W|X,Y)\, dW \tag{2.58}$$

The prediction uncertainty is quantified based on the variance of the prediction distribution that can be decomposed further using the law of total variance, yielding the following form (Zhu and Laptev 2017)

$$Var(y^*|x^*) = Var[\mathbb{E}(y^*|W,x^*)] + \mathbb{E}\,[Var(y^*|W,x^*)] \tag{2.59}$$

$$= Var\left(f^W(x^*)\right) + \sigma^2 \tag{2.60}$$

where $Var(y^*|x^*)$ represents the ignorance of model parameters W i.e. the model uncertainty, σ^2 is the noise level during the data generation process i.e. inherent noise. Especially in the case of time-series data where the pattern over test data may not resemble the pattern during the training data, a complete measure of prediction uncertainty should account for model uncertainty, model misspecification, and inherent noise (Zhu and Laptev 2017). Eq. (2.62) can be decomposed in two terms, the model uncertainty and the inherent noise respectively.

Bayesian Neural Networks (BNNs) introduce uncertainty to deep learning models from a Bayesian perspective i.e. by assigning priors to the network parameters W, the network now finds the posterior distribution of W instead of the point estimate (Zhu and Laptev 2017, Gal 2015). The earliest references to BNNs signal that the only statistical interpretation of neural network Euclidean loss is as maximum likelihood w.r.t. a Gaussian likelihood over the network outputs and that inference could theoretically be performed through the invocation of Bayes' theorem (Tishby et al. 1989). As a way of model regularisation, modern approaches rely on a fully factorised approximation—the approximating distribution assumes independence of each weight scalar in each layer from all other weights (Hinton and Van Camp 1993). This is generally referred to as **posterior inference**, since exact inference is rarely possible, they are approximators.

Many new approximate inference methods have been put forward for BNNs that are based on the variational inference to optimize the lower bound (Zhu and Laptev 2017, Khosravi et al. 2015). For example, stochastic search (van Hinsbergen et al. 2009), variational Bayes (Khosravi et al. 2010), probabilistic backpropagation (Hwang and Ding 1997), Bayes by BackProp (Veaux et al. 1998) and its extension (MacKay 1992). Several algorithms further extend the approximation framework to α-divergence optimization (Nix and Weigend 1994, Efron 1979). Readers can refer to earlier studies for a more detailed review of modern BNNs (Heskes 1997, Neal 2012, Mullachery et al. 2018).

A critical point to note concerning all the BNN methods is that they require a completely new set of training methods, adjusting the loss function to the optimization problems, and training new batches, all this and more building a new BNN non-trivial. BNN's are also slow and expensive to apply to large datasets. Many such inference approaches introduce many new hyper-parameters, and the management of a massive number of parameters.

2.9.5. MC Dropout

Recent studies have shown that dropout applied before every weight layer is mathematically equivalent to an approximation of the probabilistic deep Gaussian process (Damianou and Lawrence, 2013; Gal and Ghahramani 2016). This problem is very critical because while LSTM and other neural networks can be fine-tuned for specific time-series prediction problem, there are instances when it gets it right and ones that the model will not get it right. Hence, it is essential to quantify the uncertainty of the prediction point estimate; nonetheless, previous models have not uncertainty bounding for neural network models (Gal 2015). For example, an adverse exogenous factor may impact the GDP growth of countries or stock prices and it would be important to constantly monitor the uncertainty surrounding the prediction over time.

Achieving the same objective as a BNN without the computational complexity, recent work has shown that the neural network architecture can be directly applied to trained models. Monte Carlo dropout (MC) framework can be used in neural network architecture to provide uncertainty quantification for almost free (Gal 2015, Gal and Ghahramani 2015b, Gal and Ghahramani 2015c). Dropping some hidden connection will lead to a sampling procedure in neural networks by giving weights from the Bayesian network with the same affect. More specifically, stochastic dropouts are applied after each hidden layer, and the model output can be approximately viewed as a random sample generated from the posterior predictive distribution where the model uncertainty is estimated by sample variance over iterations (Gal 2015).

In Bayesian inference, model uncertainty is estimated by obtaining the posterior distribution $p(W|X,Y)$. The MC dropout solves for the critical challenges of non-conjugacy and non-linearities in neural networks (Gal 2015, Gal and Gadhot 2015a, Gal and Ghahramani 2015b). The sample variance can approximate the model uncertainty as

$$\widehat{Var}\left(f^{\widehat{W}}(x^*)\right) = \frac{1}{B}\sum_{b=1}^{B}\left(\hat{y}^*_{(l)} - \overline{\hat{y}^*}\right)^2 \qquad (2.61)$$

where x^* is a new input, the neural network output with stochastic dropouts at each layer with probability p, this feedback is repeated B times to obtain $\left\{\hat{y}^*_{(1)}, \dots, \hat{y}^*_{(L)}\right\}$, and $\overline{\hat{y}^*} = \frac{1}{B}\sum_{b=1}^{B}\hat{y}^*_{(l)}$ (Gal, Gron, Hron, and Kendall 2017; Zhu and Laptev 2017).

Assume we have a neural net model $y^* = f^w(x^*)$ where $y*$ is the prediction provided W as parameters and $x*$ as the input. To perform UQ on this model we apply the algorithm 1 mentioned below. The factors that we are going to consider for the uncertainty or standard prediction error are model uncertainty and inherent noise. MC Dropout requires an optimal probability value $\boldsymbol{p}$ usually considered to begin with 0.3 to 0.5. this measure gives us how much fluctuation is present in the model (Gal 2015).

```
Input: prediction model h(·), dropout probability p, number of iterations B, test dataset $x*$, validation set $x'$
Output: uncertainty η
for b = 1 to B do
    $\hat{y}^*_{(b)}$← Dropout (h(x*), p)
end for
// model uncertainty
$y^* = \sum_{b=1}^{B} \hat{y}_{(b)}$
$\eta_1^2 \leftarrow \frac{1}{B}\sum_{b=1}^{B} \left(\hat{y}^*_{(b)} - y^*\right)^2$
// inherent noise
for each $\boldsymbol{x'}$ in validation set $\{\boldsymbol{x'_1}, \boldsymbol{x'_2}, \ldots, \boldsymbol{x'_v}\}$ do
    $y'_{(v)} \leftarrow h(x')$
end for
$\eta_2^2 \leftarrow \frac{1}{V}\sum_{v=1}^{V} \left(\hat{y}'_{(v)} - y'_{(v)}\right)$
// The total uncertainty
$\eta = \sqrt{\eta_1^2 + \eta_2^2}$
return η
```

Algorithm 1: UQ for a regression task

A neural network with some input x^* the end point is the expected model output given the input i.e. $\mathbb{E}(y^*)$ - the predictive mean, and how the model is confident in its prediction i.e. $Var(y^*)$ - the prediction variance (Gal 2015, Gal and Ghahramani 2015a, Gal and Ghahramani 2015b). The dropout network is essentially a Gaussian process approximation (Gal and Ghahramani 2015b, Gal and Ghahramani 2015c). This implementation is essentially saying that over a RNN or LSTM, we are essentially putting a gaussian distribution on the weights, and mathematical gymnastic shows that doing gaussian weights same as multiple gaussian sampling over the dropout. Essentially going back into BNN to capture posterior for uncertainty, the recent work proves that sampling over LSTM with a dropout layer functionally the same as add tao parameter (τ). It is not just adjusting the dropout value, sample on predictions leveraging expectation value and variance of prediction without changing the model or other parameters optimizing the hyperparameters of LSTM (Zhu and Laptev 2017, Wen et al. 2017).

In this approach, firs the prior length scale $\boldsymbol{l}$ captures the belief over a function, for example a high-frequency stock ticker indicator would correspond to a short length scale $\boldsymbol{l}$ and a low-frequency annual macroeconomic indicator would correspond to long-length scale. Mathematically the Gaussian process precision factor τ can be defined as:

$$\tau^2 = \frac{\boldsymbol{l}^2 p}{2N\lambda} \tag{2.62}$$

Here the length scale $\boldsymbol{l}$ has been squared and is divided by the weight decay, *p* is the probability of the units not being dropped (Gal 2015).

A network output is simulated with input x^* treating dropout during training time in a classic fashion i.e. do random dropout on the test set (Gal and Ghahramani 2015b, Gal and Ghahramani 2017) This is repeated over *T* iterations where different units are dropped every time and the results for $\{\hat{y}_t^*(x^*)\}$ are collected. This process resembles an empirical sampling procedure from the approximate posterior predictions (Srivastava et al. 2015, Gal 2015). The empirical estimator for predictive mean and predictive variance (or uncertainty) can be obtained from the sampling in the following manner

$$\mathbb{E}(y^*) \approx \frac{1}{T}\sum_{t=1}^{T} \hat{y}_t^*(x^*)) \tag{2.63}$$

$$Var(y^{*}) \approx \tau^{-1}\, \mathrm{I}_D + \frac{1}{T}\sum_{t=1}^{T} \hat{y}_t^{*}(x^{*})^T \hat{y}_t^{*}(x^{*}) - \mathbb{E}(y^{*})^T \mathbb{E}(y^{*}) \qquad (2.64)$$

Eq. (2.66) was introduced for averaging model to scale the weights at test time without dropout gives reasonable approximation (Srivastava et al. 2015). The second equation is the sample variance over *T* forward passes through the network plus the inverse model precision.

The MC dropout in neural network acts as an approximator to variational inference in gaussian distributions. The dropout neural network has been argued to be identical to variational inference in Gaussian process (Gal 2015, Gal and Ghahramani 2015b, Gal and Ghahramani 2015c). Stochastic nature of dropout captures the variance in the long term direct correlation to uncertainty, through experimentation and mathematical rigour (Gal 2015, Gal and Ghahramani 2015b, Gal and Ghahramani 2015c), the *tao* interpretation offers precision utilizing the wider distribution of sample from a gaussian process.

In addition to model uncertainty discussed, there are many elements to comprehensively capturing epistemic uncertainty including model misspecification and inherent noise. In some more complex architecture, encoder-decoder framework where the encoder extracts the representative features from a time-series and constructs the learned embedding, whereas the decoder reconstructs the time-series from the embedded space, and variational encoder is applied to the LSTM layer in the encoder and regular dropout to the prediction network (Wen et al. 2017, Zhu and Laptev 2017). The encoder-decoder architecture can help capture the uncertainty when predicting unseen samples with very different patterns from the training data set i.e. **model misspecification**. These architectures can easily get extremely complex and perhaps not scalable across domains. Computing the noise level via the residual sum of squares, evaluated on an independent held-out validation set captured the **inherent noise**. This factor is always present in any form of mathematical model and it could only be minimized but not eliminated from a trained and tuned model.

2.9.6. Quantile Methods

Uncertainty in Deep Learning is still a nascent area of study, there are growing areas of research and many ways to obtain similar objectives. Here two recent yet novel applications of quantile methods in deep learning are discussed.

2.9.6.1. Quantile Recurrent Forecaster (MQ-RNN)

Quantiles of the forecast distribution are useful in making optimal decisions, both to quantify risks and minimize losses (e.g. risk management, power grid capacity optimization), leading to the use of Quantile Regression (QR) (Koenker and Gilbert, 1978). QR predicts the conditional quantiles of the target distribution. Different multi-step strategies with neural networks, including the Direct Multi-Horizon strategy that avoids error accumulation, yet retains efficiency by parameters sharing for multiple horizons (Taieb and Atiya 2016). In QR, the conditional quantiles are predicted i.e. $y_{t+k}^{(q)}|y_{:t}$ for the target distribution $\mathbb{P}\left(y_{t+k} \leq y_{t+k}^{(q)}|y_{:t}\right) = q$. QR is robust to parametric distributional assumptions, accurate forecasts, and serve as a post-processor for prediction calibration (Tayler 2000). The MQ-RNN was proposed to solve for multi-horizon quantile forecasts to solve large scale time-series regression problem (Wen et al. 2017). It can be presented as

$$p\left(y_{t+k,i}, \dots, y_{t+1,i}\middle| y_{:t,i}, x_{:t,i}^{(h)}, x_{t:,i}^{(f)}, x_{i}^{(s)}\right) \tag{2.65}$$

where $y_{.,i}$ is the *i*-th time series to forecast, $x_{:t,i}^{(h)}$ are the temporal covariates available in history, $x_{t:,i}^{(f)}$ is the knowledge about the future, and $x_{i}^{(s)}$ are the static, time-invariant features (Wen et al. 2017). In order to enable cross-series learning for items with limited history, each time-series is accounted for as a single sample which is then fed into a single RNN (Wen et al. 2017).

The QR models are trained to minimize the total Quantile Loss (QL) represented as

$$\mathcal{L}_q(y, \hat{y}) = q(y - \hat{y})_+ + (1 - q)\,(\hat{y} - y)_+ \tag{2.66}$$

where $(.)_+ = \max(0, .)$ and q = 0.5, QL is the Mean Absolute Error (MAE), and the minimizer is the median of the prediction distribution. Now if we take *K* to be the number of horizons of forecast, *Q* be the number of quantiles of interest, then the *K* × *Q* matrix $\hat{Y} = \left[\hat{y}_{t+k}^{(q)}\right]_{k,q}$ is the output of a parametric model $g(y_{:t}, x, \theta)$ which happens to be a RNN or LSTM in this case. The model parameters are trained to minimize the total loss, $\sum_t \sum_q \sum_k L_q \left(y_{t+k} \middle| y_{t+k}^{(q)} \right)$ where *t* iterates through all forecast creation times (FCTs) (Wen et al. 2017).

This is in similar view as work on Seq2Seq, where LSTM to encode all history into hidden states and adopts two MLP branches where encoder and decoder exchange information. (Cho et al, 2014). The encoder LSTM output plus all future inputs into two contexts: a series of horizon-specific contexts for each future point of time capturing network structural awareness of temporal distance between forecasts, and a horizon-agnostic context capturing common information (Wen et al. 2017). This extension is computationally expensive.

2.9.6.2. Tilt (Pinball) Loss Function

In addition to MC dropout, one of the other key UQ's used in this work will be the tilted loss function approach. Now this method has been used only for MLP's so far, here one of the first use for UQ in RNN's and LSTM's is presented.

In this method we are aiming to predict the range of uncertainty quantile range for a specific time-series problem such as closing price prediction for tomorrow, GDP growth for next quarter, or the unemployment rate next year. One of the key advantages of the Quantile Regression Tilted Loss function is that it is computationally least expensive, especially compared to Bayesian RNNs and other algorithms. Furthermore, from a purely epistemic uncertainty perspective in decision making, most people actually require quantiles as opposed to true uncertainty in an estimate (Abeywardana and Ramos 2015). On the non-parametric inference of quantiles also draws a variational Bayes approximation in a expectation maximization algorithm (Abeywardana and Ramos 2015).

The tile (pinball) loss function was introduced to penalize predictive quantiles at wrong locations (Koenker and Bassett Jr.). The function is defined as

$$\mathcal{L}(\xi_i|\alpha) = \begin{cases} \alpha\xi_i & if\ \xi_i \geq 0 \\ (\alpha - 1)\xi_i & if\ \xi_i < 0 \end{cases} \tag{2.66}$$

The loss function is defined over the error ξ_i for a specified quantile $\alpha \in (0,1)$, and ξ_i represents the error between the observation and the y_i inferred quantile f_i i.e. $\xi_i = y_i - f(x_i)$, $f(x_i)$ is the predicted (quantile) model and y is the observed value for the corresponding input x. The final overall loss function for a quantile model of the whole dataset is defined as

$$\mathcal{L}(y, \mathbb{f}|\alpha) = \frac{1}{N}\sum_{i=1}^{N} \mathcal{L}(y_i - f(x_i)|\alpha) \tag{2.67}$$

The negative exponential of the loss is presented below in Figure (2-27) that shows that if we were to find the area under the graph to the left of zero it would be alpha, the required quantile (Abeywardana and Ramos 2015).

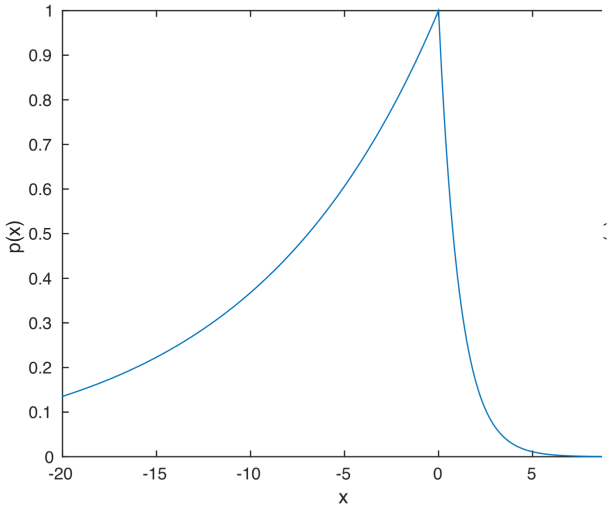
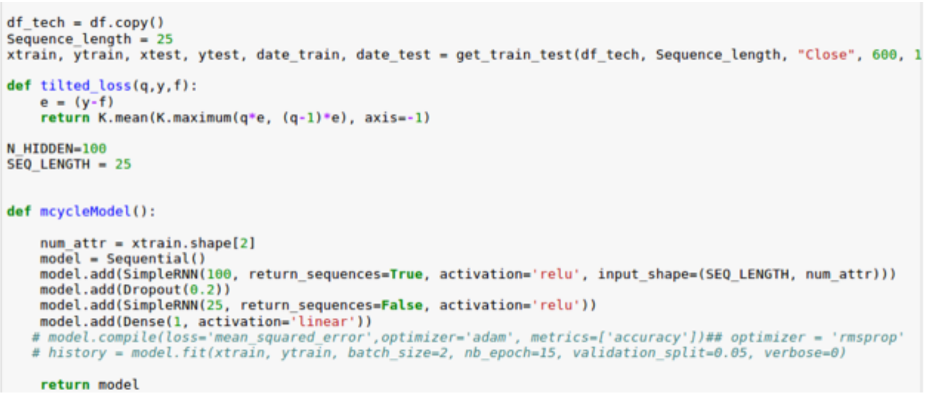


Figure 2-27. Probability distribution function (PDF) of an Asymmetric Laplace distribution.
Source: Abeywardana 2018.

In terms of implementation, the parameters that require input are the quantiles, values that can be tuned i.e. the hidden layer, model network such as MLP, RNN, LSTM, or GRU, nodes, quantile range, sequence length, activation functions. Now traditionally this approach has only been applied to MLP and shallow neural networks. This is the first application of the tilted loss to LSTMs and related RNN architectures. Figure (2-28) provides a simple class abstraction of the implementation where the model is trained using a Vanilla RNN with 100 and 25 nodes in hidden layer 1 and 2 respectively, the loss function is switched to Quantile Loss Function, optimizer is Adam (with learning rate of 0.02) and quantiles = 0.25 and 0.75, and finally values that can be tuned are the hidden layer.



```
df_tech = df.copy()
Sequence_length = 25
xtrain, ytrain, xtest, ytest, date_train, date_test = get_train_test(df_tech, Sequence_length, "Close", 600, 1

def tilted_loss(q,y,f):
    e = (y-f)
    return K.mean(K.maximum(q*e, (q-1)*e), axis=-1)

N_HIDDEN=100
SEQ_LENGTH = 25


def mcycleModel():

    num_attr = xtrain.shape[2]
    model = Sequential()
    model.add(SimpleRNN(100, return_sequences=True, activation='relu', input_shape=(SEQ_LENGTH, num_attr)))
    model.add(Dropout(0.2))
    model.add(SimpleRNN(25, return_sequences=False, activation='relu'))
    model.add(Dense(1, activation='linear'))
   # model.compile(loss='mean_squared_error',optimizer='adam', metrics=['accuracy'])## optimizer = 'rmsprop'
   # history = model.fit(xtrain, ytrain, batch_size=2, nb_epoch=15, validation_split=0.05, verbose=0)

    return model
```

Figure 2-28. Code Abstraction for a Vanilla RNN with Quantile Loss Function for Confidence Intervals

Now to summarize the uncertainty quantification for our decision-making cases, this research will use the MC dropout method for results shown across different research domains. The verdict on the optimal UQ method is still out; however, the literature has recently provided relevant comparisons. For example, when Bootstrap is compared to Classical and Bayesian methods, it had the smallest width of confidence interval which indicates that this method was more thorough and recommended (Solimum 2014). In addition, comprehensive review of delta, Bayesian, mean-variance estimation, and bootstrap techniques showed that an ensemble of the UQ methods is the most reliable (Khosravi et al. 2015). Variational Inference Networks, MC dropout method, and the quantile loss method have become widely adopted due its efficiency and reliability, where extensions of this framework have also leveraged new neural network frameworks such as encoder-decoders at Amazon and Uber (Gal 2015, Gal and Galambhri 2015b, Abeywardana and Ramos 2015, Wen et al. 2017, Zhu and Laptev 2017). The current state of the schools of thought for UQ in Deep Learning are summarized in Table (2-7).

Table 2-7. Schools of Thoughts References for Epistemic Uncertainty Quantification in Deep Learning

Papers	Methods	Summary

Gal, Y. and Z. Ghahramani, 2015;
Gal, Y. and Z. Ghahramani, 2016a;
Gal, Y. and Z. Ghahramani, 2016b;
Gal, Y, 2016
	Combining modern deep learning and Bayesian techniques for uncertainty quantification, variational inference, MC Dropout as a Bayesian approximator	Bayesian method is computationally expensive, but MC dropout framework can estimated for almost free to obtain prediction intervals

Wen, Torkkola, Narayanaswamy, 2017	Multi-horizon Quantile Recurrent Forecaster (MQ-RNN), large-scale time-series forecasting at Amazon	Nonparametric nature of Quantile Regression and the efficiency of Direct Multi-Horizon Forecasting is scalable but computationally expensive in network framework
Khosravi et al. 2014	Delta, Classical, MVE, Bootstrap, Bayesian methods comparisons, Genetic Algorithm for ensembling various UQ methods	Comprehensive review and comparison, ensemble method is promising and important for system architecture UQ modularity
Solimum 2014	Empirical comparison of Classical, Bayesian and Bootstrap methods for UQ	Bootstrap method has the smallest width confidence interval, thereby more precise than Bayesian and Classical methods
Zhu, L., and N. Laptev, 2017	Deep and Confident Prediction for Time Series at Uber, real-time anomaly detection across millions of metrics, Bayesian deep LSTM for time-series prediction and uncertainty	Computationally expensive and useful for anomaly detection, method is essentially the same as Gal 2015, Gal and Ghahramani 2016, however, a computationally intensive encoder-decoder neural network framework
Abeywardana and Ramos 2015	Quantile Regression Loss Function application MLP, our application applies to RNN and LSTMs	Non-parametric method of inferring quantiles and derivation a novel Variational Bayesian (VB) approximation to the marginal likelihood, similar to Gal 2015, Gal and Ghahramani 2016, however using a tilt (pinball) loss function, easier implementation

Chapter 3: Gaps and Research Objectives

3.1. Gaps

Despite the significant contributions of the existing research studies and practices, there are many opportunities to improve the state-of-the-art risk management practices as current studies: (1) are only based on one domain (ICRG 2017, Moody's 2017, IMF WEO 2017); (2) do not deploy high-dimensional real-time time-series information or leverage deep neural network (WEF 2017, Bloomberg 2017); (3) rely solely on expert opinion data with no information related to comparison of models (WEF 2017, World Bank 2013); and (4) do not provide transparency for decision-makers to go beyond the black box (Sarlin 2015, Bao et al. 2017, KPMG 2016, McKinsey 2015).

From a decision-maker's perspective, there are many practical needs that are missing in the current practices. Decision-makers want to have a more complete picture of the opportunities and threats in the world's poorest countries including low and middle-income countries. Moreover, some policymakers are concerned about the state of fragile and conflict-affected states (FCS), as well as, other domains such as: the impact of climate change. Most importantly, decision-makers care about granular and time foresights facing the world's biggest cities.

At the same time, from a model-building perspective, there are many neglected attributes for domain-specific learning and predictions. First, it is not clear whether a structured logic approach may yield a higher accuracy at lower computational costs in the domains of interest. Second, it has not been established whether and how the state-of-the-art techniques can be applied in the domains of interest. Third, there is no straightforward methodology on how to make black box models more transparent so decision-makers can get in-depth reasoning behind the forecasts. Fourth, it is not well understood how these signals can inform about project related risks. For simplicity, Table (3-1) establishes relationships between the existing research gaps, the research objectives, models and case studies.

Table 3-1. Research Gaps, Objectives, Models, and Case Studies

Research Gaps	**Objectives**	**Models**	**Case Studies**
Domain Knowledge	Pre-process domain knowledge using algorithms	Entropy, Sharpe Ratio, Vortex, Growth rate, Economic complexity etc.	Economic, Finance, and Geo-political
Large Scale Training or Structured Logic?	Show structured logic to training data yield higher accuracy at lower computational cost and complexity	Knowledge Abstraction Layer (KAL), Sequence Learning Gate (SLG), Decision Gate	Economic, Finance, and Geo-political
Do not deploy real-time information or leverage deep neural network	To generalized ensemble model architecture and framework	Proposed ensemble models	Theory
	Show a scalable framework, code ready to deploy	Ensemble Long-Short Term Memory (LSTM) Recurrent Neural Network	Economic, Finance, and Geo-political
Model Comparisons	Establish as a reliable model for domains	Compare ensemble LSTM with Logit, Linear, SVM, different NN architecture	Economic, Finance, and Geo-political
Don’t go beyond Black	Go beyond black box to provide structured output.	Controlled Experiment: Use MC simulation for stress-testing. LSTM RNN, GRU, Bi-directional LSTM	Economic and Finance
		Selected Memory: Preserve memory of past crisis to predict current time label and accommodate anomaly time LSTM, DTW, Logic	Economic and Finance

3.1.1. Research Gap 1 – Domain-Knowledge and Large-Scale Training

The first research gap arises from the lack of validation of the importance of large-scale training. Traditionally, engineers use a "kitchen sink" approach if they do not have domain expertise as in the case of geopolitics, economics, and finance (Rahimi and Recht 2008, Falat et al. 2015, Herbrich et al. 1999). In these instances, the engineer used to build as big a training data-set as they can attain and then tunes the hyper-parameters to get the lowest error in the prediction model. However, there are significant computational costs associated with this approach, especially in new domains of applications where large training data is not readily available. Such large-scale training could also lead to spurious relations, overfitting, and higher information uncertainty. Consequently, there is a trade-off between model complexity and overfitting. This trade-off is not defined in current models; and it is not clear if an alternative approach to abstracting structured domain logic will help reduce the computational costs by optimizing the training data size as well as preserve this knowledge for deeper analysis.

Therefore, in this view of accommodating domain-specific knowledge, the abstraction and generalization of time-series forecasting problem is needed. Many researchers have worked on domain centric problems for specific aspects such as: political or economic failure (Wittman 1995, Bookstaber 2017). However, these studies do not provide a schematic for model predictions, including knowledge abstraction, to inform multi-variate time series sequence predictions. More specifically a structured approach to abstract reasoning about domain knowledge may provide better results than the "kitchen-sink" approach where non-domain experts dump all the data they have. Furthermore, a set up for a structured logic has not been explored for the particular domains of interest. The structured logic approach could preserve abstracted knowledge and help inform decision-makers beyond black box predictions.

3.1.2. Research Gap 2 – Real Time Information and Deep Neural Networks

The second challenge is how to incorporate daily and real-time data and deploy deep neural networks for domains of interest. Decision-makers need information for real-time decision related to financial, economic, and political mass movements. It is not clear how the explosion in data and methods, such as deep neural networks, will impact human judgment and the decision-making process concerning LEPs. Hence, structured logic framework for time-series related problems is an important element that can inform the decision-making process. For example, to inform decision-makers how certain external domain risks may or may not impact their projects or assets, it will be important to extract the features of the project like: location, sector, assets, partners, size, etc. These project features will then have to be mapped with the entity and domain specific risk features. However, current available architecture and framework do not leverage large-scale open source data, or abstract relevant knowledge from multiple sources and models; thus, the use of ensemble deep neural networks, especially in our domains of interest will help address this gap.

3.1.3. Research Gap 3 – Model Comparisons

Currently, there is a lack of proof, or disproof, that proposed models indeed perform better than traditional models and industry standards. In the literature and various communities of practice there is an on-going debate whether deep learning practices perform better than tried and tested models (Heaton et al. 2016, Kosner 2016, D'Acunto 2016, Smith 2017). More specifically, can a more structured logic approach be used to ensemble-models in a transparent manner and yield higher prediction accuracy?

The unsurpassed performance of LSTM's has been well-documented in sequence prediction problems such as videos, digit rejection; nonetheless, and has been applied in limited capacity for our domains of interest. Hence, there is a research gap in the lack of understanding whether new ML/AI models perform more reliably in informing location-based opportunities and threats monitoring. The data flows and model architecture attributes in the domains of interest are also not well understood.

3.1.4. Research Gap 4 – Do Not Go Beyond Black Box

Decision-makers are wary of black box models. The black box can approximate any function but studying its structure will not give any insights on the structure of the function being approximated. A big concern decision-makers face is how to go beyond just predictions and have deduction and reasoning abilities to gain deeper insights into the problem (Knight 2017, Wolchover 2017, Shwartz-Ziv and Tishby 2017). For instance, the policy and investment community can learn from interdisciplinary fields such as vision and pattern recognition; similarly, in the computer science community, the treatment of videos as slices of images at each point of time. i.e. as a time-series problem, has delivered many breakthroughs. However, the challenge is how to convert these tasks in our domains of interest into a compatible model framework without compromising the simple insights that decision-makers need. Decision-makers are interested in stress-testing to understand the impact of an external shock on their forecasts; they may want to infer that the current time-pattern is similar to an anomalous, prior crisis, or normal pattern. Such deeper insights have not been represented in the current literature and state-of-the-art models. Furthermore, uncertainty quantification for our domains of interest leveraging deep neural networks is a nascent, yet growing body of literature (Gal 2015, Wen et al. 2017).

3.2. Research Objectives

The main goal of this research study is to develop models that address the above gaps by providing structured output to inform the executive's decision-making process concerning large engineering projects (LEPs). To achieve this goal, the following research objectives are identified. The first objective is to provide a methodology to convert geo-political, economic, and financial signals with pre-processed domain knowledge. Second, is to develop a generalized architecture for ensemble-models in sequence forecasting problems and compare the results of the ensemble-models with the traditional techniques to validate the model performance, accuracy and scalability. Third, a collection of ensemble-model architecture is proposed in order to go beyond the black box. Through achieving these objectives, a number of significant research questions will be answered.

3.2.1. Structure Domain Logic for Training Data

This research objective is to optimize deep neural networks training based on domain knowledge. For example, in the domain of finance, predicting the stock price for Goldman Sachs versus Amazon would require different set of training features. In the domain of geo-politics predicting a protest event in Syria versus the United States, or in the economics domain predicting the GDP growth of Singapore versus Brazil will result in the same problem of appropriate training data. In such cases, domain knowledge must be extracted. Therefore, a decision-gate could help in selecting the relevant domain-specific training examples. This decision-gate will select similar features given the prediction problem and grouped clusters; like technical indicators' cluster, entities' cluster, domains' and classes' cluster.

In addition, this research objective will show that increasing the sample of training data to include unstructured features is not the right approach in the domains of interest. A structured logic approach preserves the entity-domain relations and achieves state-of-the-art accuracy.

By attaining this objective, the persistent research question of: which models can help in the identification and prediction of opportunities and threats, can be thoroughly answered.

3.2.2. Establish Proposed Ensemble Model

The second research objective is to establish the Long-Short Term Memory (LSTM) neural network as the state-of-the-art in sequence prediction problems in our domains of interest. The evidence should make the case that the proposed ensemble LSTM model performs better than industry standards and other current models used in the domains. This ensemble model should have prior stored knowledge to make selective decisions regarding domain-specific predictions. Moreover, evidence will be provided to prove that the structured logic approach for domain-knowledge representation is important for executive decision-making. For example, an event in Syria may not impact the economic performance of the USA, but China's export growth pattern may be significant for predicting US economic performance. Consequently, this research objective will provide an answer to an imminent research question, which is how to best inform the model from knowledge of other different models to get best accuracy and inform strategy?

3.2.3. Go Beyond Black Box to Provide Structured Output

The third objective of this research is to provide insights beyond a black box model without compromising the predictability power of the model. Decision-makers look for a structured output to inform them about their decisions. This structured output should spit out the prediction output accompanied with all relevant details. For example, a prediction of a country's indicator like GDP growth should be accompanied with associated knowledge such as: (a) most similar countries, (b) most similar indicators, (c) a signal whether the current trend follow an anomaly or crisis pattern, (d) dynamic view of relevant moving factors, or (e) locations like cities or states with high or low growth.

A related area of research called Structured inference requires structured prediction to solve domain specific problems. Structured prediction models include Conditional Random Fields (CRF) or Max-margin Markov Networks (M^3N). Recent studies have shown that without understanding combinations of inference and learning that are appropriately compatible, learning performance under approximate inference cannot be guaranteed (Kulesza and Pereira 2007). Similarly, deep Markov models (DMM) for structured inference problem has also been proposed (Krishnan et al. 2016). Other researchers have also used Hidden Markov Models (HMM) for predicting social unrest in East Asian countries (Qiao et al. 2017). Recent developments to capture epistemic uncertainty (model, model misspecification, and inherent noise) in both time-series and classification problems has opened new doors to account for uncertainty quantification in our domains of interest (Gal 2015, Wen et al. 2017, Khosravi et al. 2014).

The recent developments provide some inspirations for going beyond the black box; however, an important element that is still missing is the incorporation of the interdependencies between and within domains and entities and having a more transparent view of prediction beyond the black box.

The realization of this research objective will help in answering the following research questions: (1) Does "black box" model architecture yield higher predictability power than standard models; (2) Is it possible to go beyond the "black box" models; and (3) How can these models provide deduction, reasoning, and uncertainty quantification to aid project-related decision-making.

Chapter 4: Proposed Methodology

In order to achieve the previously-mentioned research objectives, the following methodology is proposed. The proposed methodology is based on a methodology architecture with three main components. The first component consists of algorithms for processing domain knowledge for technical feature clustering and training. Second, is a structured ensemble LSTM model for predicting opportunities and threats; and finally, the third consists of ensemble models to accommodate the dynamics of the system that aid inter-and-intra sectoral reallocation decision problems.

The model system architecture is illustrated in Figure (4-1). There are two meta-components to the overarching system including the Knowledge Base layer and the Intelligence Layer that are pipelined together. In the Knowledge Base layer, the various sources of information are fused together in a structured manner such that spatial-temporal tags for entities, risk domains, classes, indicators…etc. available in a single database repository. The pre-processing domain knowledge module uses domain specific algorithms and measurements to classify the data in a post-processed format that can be used for analytical model building. The intelligence layer is composed of two modules: the ensemble sequencing learning program and deeper ensemble programs. The ensemble sequence learning program is the proposed generalized ensemble framework that extract what features are relevant for a given prediction problem. It, also, consists of a collection of methods that provide a controlled environment for testing hypothesis, deduction, and reasoning purposes by accommodating uncertainty quantification. This ensemble generates the time-series forecast output in period *t+1* which is explained in section (4.2).

The next section discusses the attributes of pre-processing domain knowledge in geo-politics, economics, and finance; such as: entropy, economic complexity algorithm, and Sharpe ratio.

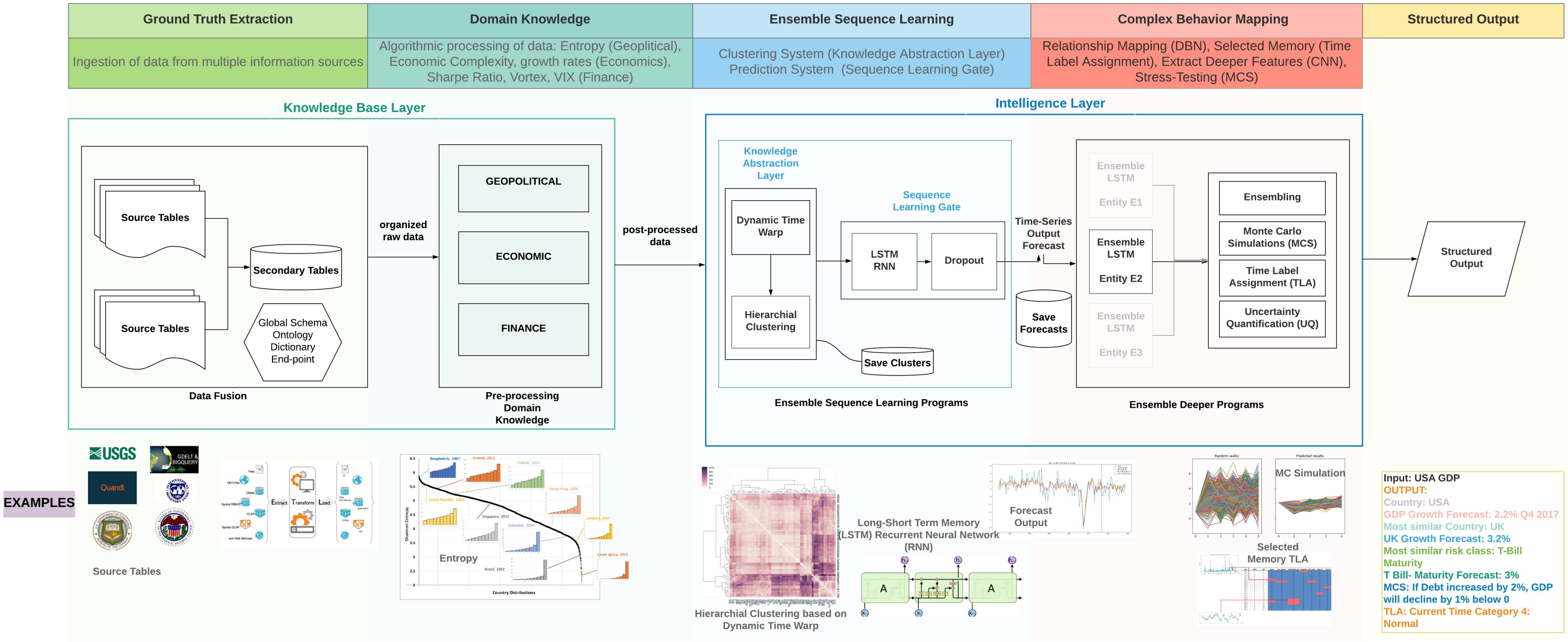

Ground Truth Extraction
Ingestion of data from multiple information sources
Domain Knowledge
Algorithmic processing of data: Entropy (Geoplitical), Economic Complexity, growth rates (Economics), Sharpe Ratio, Vortex, VIX (Finance)
Ensemble Sequence Learning
Clustering System (Knowledge Abstraction Layer) Prediction System (Sequence Learning Gate)
Complex Behavior Mapping
Relationship Mapping (DBN), Selected Memory (Time Label Assignment), Extract Deeper Features (CNN), Stress-Testing (MCS)
Structured Output
Knowledge Base Layer
Source Tables
Secondary Tables
Source Tables
Global Schema Ontology Dictionary End-point
Data Fusion
organized raw data
GEOPOLITICAL
ECONOMIC
FINANCE
Pre-processing Domain Knowledge
post-processed data
Intelligence Layer
Knowledge Abstraction Layer
Dynamic Time Warp
Hierarchial Clustering
Sequence Learning Gate
LSTM RNN
Dropout
Save Clusters
Ensemble Sequence Learning Programs
Time-Series Output Forecast
Save Forecasts
Ensemble LSTM Entity E1
Ensemble LSTM Entity E2
Ensemble LSTM Entity E3
Ensembling
Monte Carlo Simulations (MCS)
Time Label Assignment (TLA)
Uncertainty Quantification (UQ)
Ensemble Deeper Programs
Structured Output
EXAMPLES
USGS
GDELT & BIGQUERY
Quandl
Extract
Transform
Load
Source Tables
Entropy
Country Distributions
Hierarchial Clustering based on Dynamic Time Warp
Long-Short Term Memory (LSTM) Recurrent Neural Network (RNN)
A
A
Forecast Output
Random walks
Predicted results
MC Simulation
Selected Memory TLA
Input: USA GDP
OUTPUT:
Country: USA
GDP Growth Forecast: 2.2% Q4 2017
Most similar Country: UK
UK Growth Forecast: 3.2%
Most similar risk class: T-Bill Maturity
T Bill- Maturity Forecast: 3%
MCS: If Debt increased by 2%, GDP will decline by 1% below 0
TLA: Current Time Category 4: Normal

Figure 4-1. Proposed Model Architecture for Identification and Prediction of Opportunities and Threats

4.1. Pre-processing Domain Knowledge

The first analytical step in the proposed model is to process the data with state-of-the-art-models used in assembling knowledge in domain specific circumstances. In each domain, there is an extensive list of algorithms; nonetheless, to simplify matters, only key measures for constructing these features will be discussed here, which are backed by both domain and measurement science literature. This processed domain-knowledge data will be useful to construct features for training data and model building.

Pre-processing domain knowledge can also include generating technical features related to the system of interest. Examples of broad measures used in Economics, Finance, and Geo-political are described. First, summary tables for broad sub-domain measures are presented. For example, in order to measures economic disparity, many measures such as Gini coefficient, Atkinsons measure, or Shannon Entropy can be used, and within economic disparity, a main method like Shannon Entropy will be elaborated more in detail. Similarly, a sub-domain of economic domain is economic diversification, and a specific method for quantifying economic diversification called economic fitness-complexity is discussed. Reader should note this is only a non-exhaustive source of features, since many other features are incorporated.

4.1.1. Economics

There are two family of important training features based on measurement science for economics are discussed in this sub-section. These measures can be applied to extract meaning information about the system in question. Measurements to incorporate attributes of (a) economic disparity, and (b) economic competitiveness and diversification are discussed below. The main measures are first presented in tables, with specific attention given to one of the important measures from the host of measures presented.

Economic Disparity: Entropy

Economic disparities can become a systemic risk and a root cause of economic and societal failure. Hence it becomes fundamental to assess inequality with its deficiencies. It is important to appropriately treat information (or the lack thereof) and conflict in the distribution as significant deficiencies, i.e., uncertainties or risks. The measurement science related to economic disparity has remained silent on the role of information deficiency, degeneracy, and uncertainties in the development of models and metrics suitable for investment and policy decision making. Economists consider three sources to measure economic disparity: wealth inequality, income inequality, and consumption-based inequality. Economic disparity is usually measured by metric such as the Gini Index (Gini 1912), Thiel Index (Thiel 1967, 1972), or Atkinsons measure (Atkinson 1970) (see Table 4-1). However, from an information science perspective, measurement science for economic disparity ignores the information deficiency associated with income distributions explicitly. The uncertainties measured as conflict in the probability density function (pdf) regarding the information and state of economic disparity might offer insights. Any gleaned knowledge on the relative order or for that matter disorder, i.e., the information or lack thereof encoded in the economic distribution, remain important prior knowledge for important resource allocation problems. Below an information theory-based approach to quantify uncertainty and risks related to the state of economic disparity is discussed.

Table 4-1. Comparative Analysis of Models for Economic Disparity

Model	Equation	Range	Feature	Reference
Shannon Entropy	$H = -\sum_{i=1}^{n} p_i log p_i$ *P*P_i is the probability of the i^{th} decile/percentile of income range. *n* is the number of population groups.	$[0, \infty)$	As the value of Shannon Entropy increases, incomes or wealth becomes more equal. Maximum if all the outcomes are equally likely; "0" if only one outcome in certain.	Shannon (1948)
Gini index	$G = \frac{\sum_{i=1}^{N}\sum_{j=1}^{N}\lvert x_i - x_j \rvert}{2N\sum_{i=1}^{N} x_i}$ x_i is the wealth or income of person *i*. *N* is the number of persons.	[0,1]	As the value of Gini index increases, incomes or wealth becomes more unequal. "1" expresses maximal disparity among incomes; "0" expresses perfect equality, where all incomes are the same. The value may be greater than "1" when some persons have negative incomes or wealth.	Gini (1912); Pizetti and Salvemini (1955)
Theil index	$T = \frac{1}{N}\sum_{i=1}^{N} \frac{x_i}{\mu} \ln\,(\frac{x_i}{\mu})$ $\mu = \frac{1}{N}\sum_{i=1}^{N} x_i$ x_i is the wealth or income of person *i*. *N* is the number of persons. μ is the mean income.	[0,ln*N*]	As the value of Theil index increases, incomes or wealth becomes more unequal. Maximum if one person has all the income; "0" if everyone has the same income.	Theil (1967)
Atkinson index	$A = \begin{cases} 1 - \frac{1}{\mu}\left(\frac{1}{N}\sum_{i=1}^{N} x_i^{1-\varepsilon}\right)^{\frac{1}{1-\varepsilon}} & 0 \le \varepsilon \ne 1 \\ 1 - \frac{1}{\mu}\left(\prod_{i=1}^{N} x_i\right)^{\frac{1}{N}} & \varepsilon = 1 \end{cases}$ x_i is the wealth or income of person *i*. *N* is the number of persons. μ is the mean income.	[0,1]	For $\varepsilon = 0$, (no aversion to inequality) it is assumed that no social utility is gained by the complete redistribution, and the Atkinson index is "0". For $\varepsilon = \infty$ (infinite aversion to inequality), it is assumed that infinite social utility is gained by the complete redistribution, and the Atkinson index is "1".	Atkinson (1970)

Grouped data such as income distribution, CO2 emissions or export values of countries by themselves might not provide useful features for training data. Entropy can be used to classify various information sources to process features about domain knowledge in areas such as disparity in income, climate emission disparity, or trade diversification. Rather than using individual choice probabilities, entropy can be used for feature-training. For example, recent evidence provides applications of entropy to quantify risk and uncertainty in economic disparity for countries and cities (Mishra and Ayyub 2017a, Mishra and Ayyub 2017b).

Entropy is a notion associated with system or information disorder with implications for diverse range of problems involving information, uncertainty, and risk. The most well-known Entropy measure was first introduced in the late 1940's by an electrical engineer at Bell Labs called Claude Shannon. Shannon Entropy (1948) was further developed into a relative measure of entropy by Kullback and Leibler (1951). Shannon entropy has played a central role in the development of fields as diverse and far ranging as electrical engineering, statistics, mathematics, statistical physics, thermodynamics, etc.

The Shannon entropy measure *S*(*p*), for a probability distribution of a random variable with discrete values defined over $x \in X$, where *X* is a random variable with '*n*' possible outcomes and probability p_i for the *i*-th outcome, $1 \leq i \leq n$. The entropy is thus defined as:

$$S(p) = -\sum_{i=1}^{n} p_i \log_2(p_i) \tag{4.1}$$

This entropy measure takes on values larger than 0. Its value is zero if $(p_i) = 1$ for an $i \in \{1,2,\dots,n\}$, and is maximum for equally likely outcomes of $p_i = 1/n$ for all i. The Shannon entropy expression can be rewritten as:

$$S(p) = -\frac{1}{\ln(2)} \sum_{i=1}^{n} p_i \ln(p_i) \qquad (4.2)$$

The logarithm in Eq. 4-2 may be taken to any base and it is customary to use base 2 in electrical engineering and computer science because of the binary character of digital logic circuitry. The choice of the functional form in the definition of entropy is that it can be mathematically proved to be the unique function that satisfies intuitively sensible properties that a reasonable measure of information ought to satisfy. For example, given the definition of entropy, it can be shown that the information content of two independent random variables must be equal to the sum of their individual information contents. And as mentioned earlier, S(*p*) is maximized when all possible outcomes are equally likely, which accords well with intuition. If *X* is completely predictable, then a specific outcome will happen with probability one and in that case, *S*(*p*) = 0. This too makes intuitive sense because, an event with probability one is deterministic and the uncertainty content (entropy) of a deterministic variable is zero because nothing learned by observing it since the outcome is pre-determined.

Consider a simple experiment with ten sample groups. Several income distributions are possible. For example, all ten people in a given sample can have the same level of income, or one person can capture 90 percent of the total group income etc. These hypothetical group distribution cases are illustrated as cumulative distribution functions (*cdf*) with their relative Shannon entropy values in Figure. 1. As the *cdf* approaches the line of equality i.e. income parity, the maximum entropy point associated with highest conflict and uncertainty is reached. The case of extreme low entropy denotes the case where 90 percent of total group income is captured by one person. The extreme low entropy case signals low information uncertainty but high-risk of social disorder. In other words, Shannon entropy becomes largest for elements that are equally likely i.e. there is highest conflict in the case for uniform distribution of resources. The information obtained from the distribution gives a signal on the value of information embedded in a given distribution.

Some cases characterizing high entropy include (a) period of relatively low accumulation of wealth and equal distirbution of income, or (b) rich socieites in Nordic countries (like Sweden, Norway, Iceland) that have low income inequality. The communist China in the 1970's or socialist East European economies prior to 1990 (such as Slovak Republic, Uzbekistan, Hungary, Romania, Ukraine, Czech Republic) are instances of low wealth accumulation-and-equal distribution of resources. In the communist or socialist model, optimality is maintained closer to societal preference of equal distribution of resources. Similarly, results indicate that happy countries tend to have higher entropy, for example in Nordic countries of Iceland, Norway, or Netherlands. This is an instance of high entropy and high wealth accumulation and is most desirable from a utilitarian point of view. High entropy instances indicate high conflict from given probability distribution, corresponding to greater *equality* and *parity* with higher information *uncertainty.*

On the other end of the spectrum, low entropy cases are Brazil in the 1990's or the United States and South Africa in modern times. Low entropy corresponds to higher *inequality*, greater *disparity,* high risk of social disorder with low *uncertainty* and *conflict*.

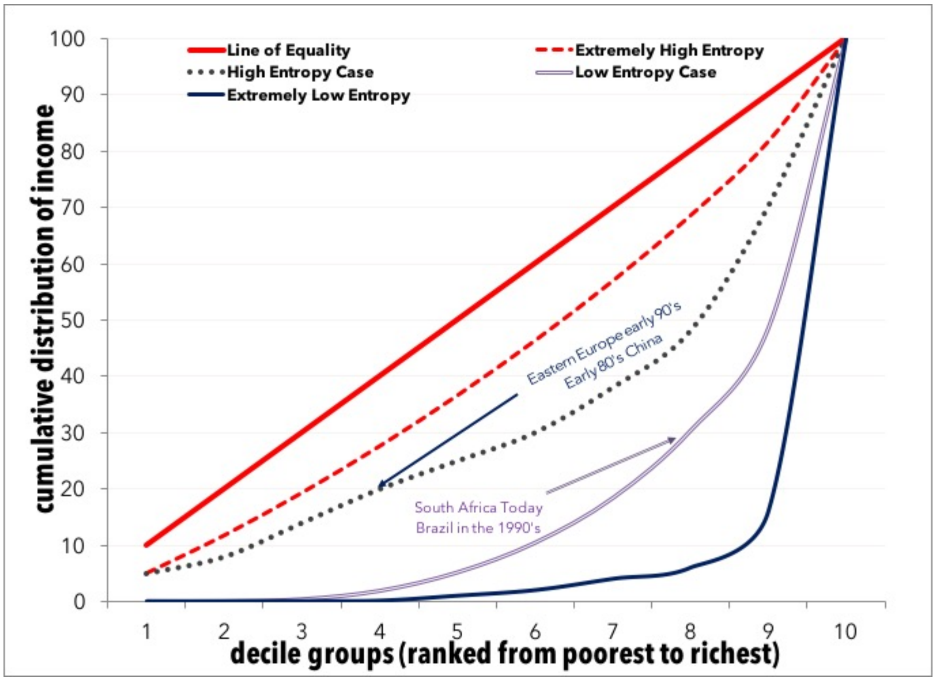


Figure 4-2. Shannon Entropy Examples with Income distribution.

Notes: The line of equality represents equal income distribution that is the 45-degree line. Such a uniform probability distribution function is one in which every person has the same income (or wealth). The lines for High Entropy Case (Eastern Europe in early 1990's or China in early 1980's) and Low Entropy Case (South Africa in 2015 or Brazil in the 1990's) are extreme distributions in the global sample based on the fraction of total income held by deciles. The other countries lie in the area between the two cases.

Entropy has also been interpreted as a measure of diversity (Reardon and Firebaugh 2002, White 1986). If all individuals in a population are associated with the same group (e.g. racial classification or income level) there is no diversity in the population. On the other hand, if individuals are evenly distributed among two or more mutually exclusive groups, there is maximum diversity (and maximum entropy or uncertainty) in the population.

Entropy has been widely applied in regional and urban science. In geographical studies, entropy has provided diverse insights on spatial decomposition, including: (1) entropy as expected information used to verify hypotheses about the spatial distribution of phenomena, (2) entropy as a measure of dispersion of random phenomena, and (3) entropy-maximizing models for the identification of the most probable spatial distribution and allocation of phenomena in a system.

In spatial analysis the notion of entropy has been applied since as early as the 1960s. Berry and Schwind (1969) justify the use of entropy and information in spatial analysis in the following words: "*Increasingly persuasive arguments are being advanced that many spatial regularities result from purely random processes, that is, that they represent most probable steady-states. These arguments continue that deviations from such regularities should therefore be worthy of more attention than the regularities themselves, because it is the deviations that reveal underlying organization and order. Information theory provides the means for formalizing these ideas. The statistically most probable state is equated with maximum entropy*".

The next figure provides a real-world interpretation for Shannon entropy and income distribution. Figure (4-3) provides an illustration of Shannon Entropy for US cities and hypothetical distributions sorted in ascending order. The vertical axis plots the value of Shannon entropy and horizontal axis various cases from hypothetical examples and city-level data for Washington, DC, San Francisco, Baltimore, and Detroit. The underlying income distributions are scaled to 9 groups. For city data, the income groups are following: (1) 0, (2) $1-$9,999, (3) $10,000-$14,999, (4) $15,000-$24,999, (5) $25,000-$34,999, (6) $35,000-$49,999, (7) $50,000-$64,999, (8) $65,000-$74,999, and (9) $75,000 and over. Higher information uncertainty is observed in San Francisco which is closest to a uniform distribution. Relatively lower entropy values are for Detroit, representing higher risk of social disorder.

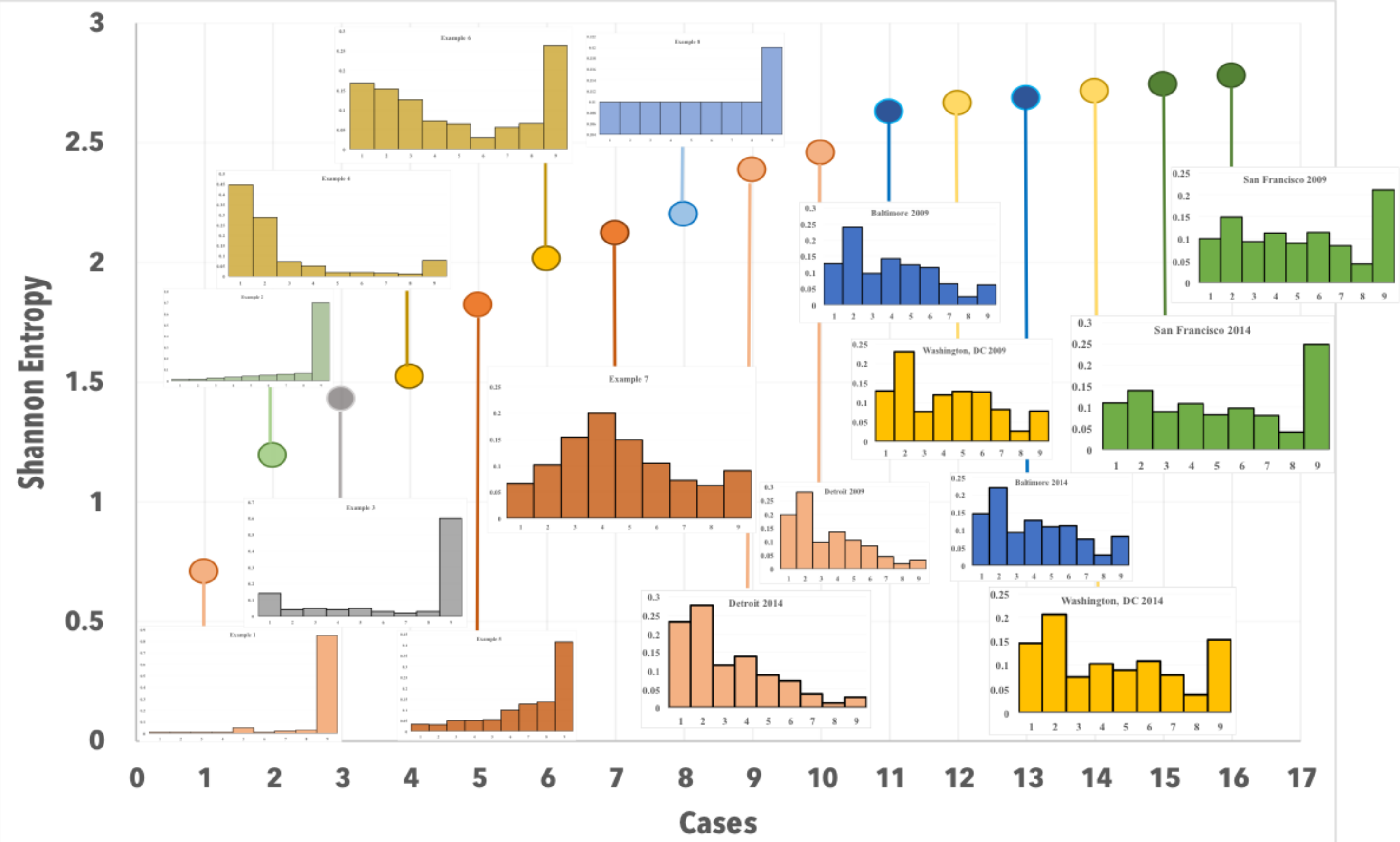


Figure 4-3. Shannon Entropy for US Cities and Hypothetical Distributions (sorted in ascending order).

The chart plots Shannon Entropy S(p) on the vertical cases with many different distribution cases on the horizontal axes that are sorted from low to high entropy values. Entropy values are for hypothetical scenarios and city data from US Census Bureau. The individual bar charts show vertical axis as fraction of household incomes and the horizontal axis as income range bins. The cases references Example 1-7 are hypothetical cases. Cases with relatively lower entropy signals low level of uncertainty and conflict based on the information provided. This corresponds to higher *inequality*, greater *disparity,* and high risk of social and economic disorder. The cases with relatively higher entropy corresponds to greater *equality* and *parity* as well as lower risk of social disorder.

Competitiveness and Diversification: Fitness-Complexity algorithm

Similar to pre-processing domain knowledge using entropy for economic disparity economic opportunities and threats at a macro or micro level can provide valuable processed information using algorithms for quantifying economic diversification. A growing body of economic literature has shown that export diversification is the appropriate strategy. Diversification, both geographically and product-wise, is found to expand revenues and enhance growth (Hummels and Klenow 2005, Pham and Martin 2007, Brenton and Newfarmer 2007). More specifically, the accounting for economic diversification or concentration can be conducted with micro-economic data for cities, but here are elaborated for export earnings to assess external macroeconomic competitiveness and diversification. Countries and regions that are dependent on a narrow export basket often suffer from export instability arising from unstable global demand (Anand et al. 2015). Diversification of exports and destinations helps in stabilizing export earnings in the longer run, with benefits analogous to the portfolio effect in finance (Ghosh and Ostry 1994).

Furthermore, the only sources of detailed raw data is available export-import values of a country. These export earnings figure capture “hidden” capabilities of competitiveness that are encoded in the specialization pattern of countries.

The variety of methods for measuring export diversification are presented in Table 4-2. For many of these measures the results are based on a newly created database that combine detailed data on goods as well as service exports to construct a universal matrix of world trade (Mishra et al 2018a, Mishra et al 2018b).

These various elements of country and item features can be incorporated for macroeconomic predictions. Of particular interest is the introduction in recent papers that have laid the foundation of non-monetary metrics used to assess the competitiveness of countries by measuring intangible assets or the strength of an economic system, measure of economic complexity (Hausman et al. 2016, Tachella et al. 2016, Pietranero et al. 2014).

Economic Complexity approach are data-driven and interdisciplinary, borrowing from diverse fields such as dynamical systems, complex networks, and big data analysis methods such as machine learning algorithms. The spirit is similar to the Google PageRank approach, in the sense that we are interested to optimize both large data analysis and algorithms to enhance our capability to extract information and maximize the signal to noise ratio. The idea is the following. Each country has some capabilities, which represent its social, cultural, and technological structure (Dosi et al., 2000). These capabilities permit to produce and export products, so products and their complexities are linked to the fitness of each country; in particular, the complexity of a product increases with the number and the quality of the capabilities needed in order to produce it, and the fitness is a measure of the complexity and the number of the exported products. This algorithm is discussed more in detail in the next sub-section.

Table 4-2. Measures of export diversification (concentration)

Method	Definition and Use Cases
Herfindahl Index (HI)	**Herfindahl Index (HI)** is the most commonly used measure of export diversification. It lies between 0 and 1 where being close to 0 indicates well diversified exports. There is a non-monotonic relationship between sectoral diversification and development, as economic activity tends to re-concentrate at later stages of development, after diversifying during much of that process (IMF 2014, Imbs and Wacziarg 2003).
Revealed Comparative Advantage (RCA)	**Revealed Comparative Advantage (RCA)** calculates the relative advantage or disadvantage of a country in a certain class of goods or services as evidenced by trade flows (Balassa 1963).
Export Quality (EQ)	**Export Quality (EQ)** measures export quality by using unit value adjusted for differences in production costs and for the selection bias stemming from relative distance (Spatafora et al 2014).
Export Sophistication (EXPY)	**Export Sophistication (EXPY)** measures the similarity of a country's exports to the structure of advanced economy exports. The concept is based on the notion that what matters for growth is not how much you export but what you export (Hausmann, Hwang and Rodrik. (2007). herein referred to as (HHR)). Goods and services exports with high productivity and sophistication contribute more to overall economic growth (Hausmann et al 2007, Mishra et al 2011).
Economic Complexity Index (ECI)	The **Economic Complexity Index (ECI)** is an extension of the sophistication measure, and suggests that the large income gaps between rich and poor nations are an expression of the vast differences in productive knowledge amassed by different nations. The ECI, developed by Hausmann et al (2011), approximates the productive knowledge in a country and helps explain differences in the level of income of countries.
Fitness-Complexity (FC)	Similar in spirit to the ECI approach, recent methods of the **Fitness-Complexity (FC)** algorithm use a non-monetary metrics for country competitiveness (fitness) allows for quantifying the hidden growth potential of countries by the means of the comparison of this measure for intangible assets with monetary figures, such as GDP per capita (Tacehlla et al 2013, Pietranero et al 2014).
Product Space (PS)	**Product Space (PS)** approach presents a variety of statistical measures that facilitate an understanding why a country that exports a certain set of products was able to diversify in another set of new exports. The changes in the RCA are governed by the pattern of relatedness of products at the global level (Hidalgo et al, 2007). As countries change their export mix, there is a strong tendency to move towards goods that are more closely related to ones already being produced rather than to goods that are less closely related.

Fitness-Complexity

The algorithm uses detailed novel data on world trade in goods and services to extract information on diversity of a country and ubiquity of a given activity. In the next steps, the matrixes of products are notated as (*P*), services as (*S*) and the universal matrix that incorporate both goods and services as (*U*).

First, export specialization is defined. The Revealed Comparative Advantage (*RCA*) is a fundamental concept introduced by Balassa (1965). *RCA* informs whether a country's share of an item in world market, is larger or smaller than the items' share of the entire world market. *RCA* is a measure of relative specialization of a country in a given item.

Mathematically, the *RCA* of a country (another feature variable) is measured by the relative weight of a percent of total exports of a given item from the *U* matrix proportional to the percent of world export in that given item. Formally, if $X_{c,\,u}$ represents the export values from universe *u* of exports by country *c*. The *RCA* that a country *c* has in item *u* from the universal trade matrix is defined as:

$$RCA_{\ c,u} = \frac{X_{c,u}/\sum_c x_{c,u}}{\sum_s X_{c,u}/\sum_c \sum_s x_{c,\ \ u}} \tag{4.3}$$

The information from RCA is used to construct a country-item binary universal matrix for various years. This is a 3-dimensional matrix of countries, items and years. It is important to note that the final results are highly sensitive to the set *RCA* threshold value. The entries of the country-product-services provide a binary matrix *M* that describes country specialization pattern over time based on the universal matrix of world trade. The universal matrix *M* is related to the corresponding *RCA* index. For consistency, the $M_{c,u}$ entry of the adjacency matrix is described as:

$$M_{c,u} = \begin{Bmatrix} 1, & RCA_{\,c,u} \geq 1 \\ 0, & otherwise \end{Bmatrix} \tag{4.4}$$

A country is considered to be specialized in an item if it has an $M_{c,u} = 1$ for the country-item pair. Therefore, diversity *d* for country *c* is defined as the number of activities that a country is specialized in. More formally, d_c (country diversity) is equal to the sum of items the country is specialized in i.e. $\sum_{u=1}^{N_u} M_{cu}$ for all goods and services *N*. Similarly, an item from the *U* matrix is considered to be ubiquitous if many countries are specialized in the export of that activity. Therefore, ubiquity *k* for activity *u* from the *U* matrix is based on the number of countries that are specialized in that activity. More formally, k_u is the sum of countries specialized in a given good or service from the *u*- matrix i.e. $\sum_{c\;=1}^{c} M_{cu}$.

A nonlinear iterative algorithm is applied to construct these new features, such as competitiveness of a country by a single number called *Fitness*, and the level of technology and advancement required to export a product (or service) by its associated *Complexity*. Fitness of a country F_c is proportional to the sum of the products or services exported by that country, weighted by their complexity, $\sum_u M_{cu} Q_u$, where Q_u is the complexity of an item in the universal matrix (goods or services). On the other hand, the complexity itself is inversely proportional to the number of countries which export that product, and directly proportional to how fit those countries are, or as this algorithm says, inversely proportional to the inverse of the fitness. All these are mathematically summarized in the iteration of Equations 4-5.

$$\begin{cases} \widetilde{F}_c^{(n)} = \sum M_{cu} Q_u^{(n-1)} \\ \tilde{Q}_u^{(n)} = \dfrac{1}{\sum M_{cu} \dfrac{1}{F_c^{(n-1)}}} \end{cases} \rightarrow \begin{cases} F_c^{(n)} = \dfrac{\tilde{F}_c^{(n)}}{\langle \tilde{F}_c^{(n)} \rangle_c} \\ Q_u^{(n)} = \dfrac{\tilde{Q}_u^{(n)}}{\langle \tilde{Q}_u^{(n)} \rangle_u} \end{cases} \qquad (4.5)$$

This iterative method is composed of two steps at each iteration: first compute the intermediate variables $\widetilde{F}_c^{(n)}$ and $\tilde{Q}_e^{(n)}$ using the fitness and complexity resulted from the previous iteration of the algorithm (or the assumed initial values for the very first iteration) and then normalize them such that the average of the fitness for all countries and the average of complexity for all products have a value of 1. The initial conditions are $\tilde{Q}_u^{(0)} = 1\ \forall u$ and $\tilde{F}_c^{(0)} = 1\ \forall c$ (i.e. we started out by assuming that all countries are equally fit and all products are equally complex). However, it is noted that the algorithm converges to the "fixed points" of the map, meaning that no matter what the initial guess is it always converges (or supposed to converge) to the same values (Tachella et al. 2012, Cristelli et al. 2015).

4.1.2. Finance

There are many technical features that are used specific to asset classes to inform trading strategies. This section provides specific features for equity prices as a sub-domain of finance.

In finance, technical features, also referred to as technical analysis is the about forecasting the price direction and patterns based on historic information, such as price and volume to inform active risk management in trading practices (Kirkpatrick and Dahlquist 2006). The focus here is on equity markets for stocks. Generally, prices related to Open, High, Low, Close (OHLC), and Volume are available for either stock tickers (like GOOG for Google, or AMZN for Amazon), from composite indices such as the S&P500 for different time frequencies. The technical features can be extracted for either individual tickers or composite market indices. For daily focused financial time series prediction, a list of non-exhaustive technical features is presented in Table 4-3. These technical features are extremely powerful and provide a foundation to obtain meaning features of the historic time series patterns. Given the variety and different degrees of complexity for stock price technical features, the following discussion is focused on the fundamental Sharpe Ratio.

Table 4-3. Technical Features for Equity Prices, some examples

Model	Equation
Sharpe Ratio	$S = \frac{\mathbb{E}(r_s - r_b)}{\sqrt{Var(r_s - r_b)}}$ r_s- asset return r_b – risk free-rate
Rolling Sharpe Ratio	$S_t = \sqrt{k}\frac{\mathbb{E}\left(r_{s,t-k} - r_{b,t-k}\right)}{\sqrt{Var\left(r_{s,t-k} - r_{b,t-k}\right)}}$ $r_{s,t-k}$ - the previous k truncated strategy returns
Annualized Return (AR)	$AR = (1 + cumulative\ return)^{(\frac{365}{daysheld})} - 1$
Maximum Drawdown (MDD)	$MD\ (T) = \frac{(P - L)}{P}$ P- Peak value before largest drop L – Lowest value before new high established
Normalized Mean Square Error (NMSE)	$NMSE = \frac{1}{\sigma^2 n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2$ $\sigma^2 = \frac{1}{n-1}\sum_{i=1}^{n}(y_i - \bar{y}_i)^2$ $\bar{y} = \sum_{i=1}^{n} y_i$
Annualized Volatility (AV)	$AV = \frac{\sigma}{\sqrt{252}}$ Where $\sigma = \sqrt{\frac{\sum_{i=1}^{n} y_i^2}{n} - \left(\frac{\sum_{i=1}^{n} y_i}{n}\right)^2}$
Bollinger Band	Bollinger Bands denoted (20,2) means the Period and Standard Deviation are set to 20 and 2, respectively. The indicator is calculated using the following formula. First calculate the Middle Band, then calculate the Upper and Lower Bands.Middle Band = 20-day simple moving average (SMA). Upper Band = 20-day SMA + (20-day standard deviation of price x 2). Lower Band = 20-day SMA – (20-day standard deviation of price x 2), where

	SMA = the sum of closing prices over n periods is divided by n.
Vortex Indicator (VI)	Vortex indicators require a three-step process. VI is essentially an oscillation between uptrend (VI+) and downtrend (VI-). It is composed of three steps: Uptrend and downtrend movement, second calculating the true range, and third, normalize uptrend and downtrend. (Botes and Siepman 2010).
Relative strength index (RSI)	The RSI is a momentum oscillator between zero and 100 that measures the speed and change of price movements (Wilder, 1978). Traditionally the RSI is considered overbought when above 70 and oversold when below 30.
Accumulation/Distribution ROC	1. $Money\ Flow\ Multiplier = \frac{\{(C-L)-(H-C)\}}{(H-L)}$ C- Close L- Low H- High 2. Money Flow Volume = Money Flow Multiplier x Volume for the Period 3. ADL = Previous ADL + Current Period's Money Flow Volume
Moving average convergence divergence (MACD), MACD Signal and MACD difference	MACD is a momentum indicator that shows the relationship between two moving averages of prices. The MACD is calculated by subtracting the 26-day exponential moving average (EMA) from the 12-day EMA.
Average Directional Indicator	The Average Directional Index (ADX) is a three-line indicator that includes the ADX line, Minus Directional Indicator (-DI) and Plus Directional Indicator (+DI).
On Balance Volume (OBV)	Cumulative measure of positive and negative volume flow. OBV adds a period's total volume when the close is up and subtracts it when the close is down.

Sharpe Ratio

In finance, Sharpe Ratio is a measure for calculating risk-adjusted return. It was developed by Nobel laureate William F. Sharpe. The Sharpe ratio is the average return earned in excess of the risk-free rate per unit of volatility or total risk. Subtracting the risk-free rate from the mean return, the performance associated with risk-taking activities can be isolated. One intuition of this calculation is that a portfolio engaging in “zero risk” investment, such as the purchase of U.S. Treasury bills (for which the expected return is the risk-free rate), has a Sharpe ratio of exactly zero. Generally, the greater the value of the Sharpe ratio, the more attractive the risk-adjusted return.

The Sharp ratio also tends to fail when analyzing portfolios with significant non-linear risks, such as options or warrants. Alternative risk-adjusted return methodologies have emerged over the years, including the Sortino Ratio, Return Over Maximum Drawdown (RoMaD), and the Treynor Ratio. These features are used for training data in addition to Sharpe Ratio.

The Sharpe Ratio is formally defined as follows

$$S = \frac{\bar{r}_p - r_f}{\sigma_p} \tag{4.6}$$

S Sharpe ratio

$\bar{r}$ Mean portfolio return

r_f Risk-free rate

σ_p Standard deviation of portfolio return

The Sharpe ratio is often used to compare the change in a portfolio's overall risk-return characteristics when a new asset or asset class is added to it. If the addition of the new investment lowered the Sharpe ratio, it should not be added to the portfolio. The Sharpe ratio can also help explain whether a portfolio's excess returns are due to smart investment decisions or a result of too much risk. The greater a portfolio's Sharpe ratio, the better its risk-adjusted performance has been. A negative Sharpe ratio indicates that a risk-less asset would perform better than the security being analyzed.

4.1.3. Geopolitics

There are two type of technical features that are obtained from geopolitical information from GDELT database (discussed later in Chapter 5.3):

- Geolocation data from GDELT GEO API. This contains two most important features i.e. the label of cameo code and latitude and longitude. This data is mainly used for visualization of real-time events.
- Events related data are also obtained from GDELT that provides a dataframe structure with 62 features. Out of them, the main ones of interest are the following: NumMentions, QuadClass, GoldsteinScale, AvgTone for prediction. Their definitions are the following:

  NumMentions (NM): This is the total number of mentions of any event across all source documents.

  EventRootCode: This is collection of CAMEO codes to classify events in categories.

QuadClass: The entire CAMEO event taxonomy is ultimately organized under four primary classifications. The numeric codes in this field map to the Quad Classes as follows: 1=Verbal Cooperation, 2=Material Cooperation, 3=Verbal Conflict, 4=Material Conflict.

GoldsteinScale: Each CAMEO event code is assigned a numeric score from -10 to +10, capturing the theoretical potential impact that type of event will have on the stability of a country.

AvgTone: This is the average "tone" of all documents containing one or more mentions of any particular event. The score ranges from -100 (extremely negative) to +100 (extremely positive). Common values range between -10 and +10, with 0 indicating neutral.

This sub-section documented the key algorithms and methods for pre-processing domain knowledge. Entropy was proposed for various data with prior probability available, economic complexity for global trade or technology fitness attributes, and indicators such as Sharpe ratio and others technical indicators in finance domain. The next sub-sections delve deeper into the model architecture after the obtaining the post-processed domain knowledge data. The two main methods are discussed are (a) ensemble sequence learning programs, and (b) deeper ensemble learning programs for complex behavior mapping.

4.2. Structured Ensemble Learning Models for Sequence Prediction

The general sequence ensemble layer of the proposed model is based on two modules. The first module can be called the Knowledge Abstraction Layer (KAL) which processes the global input data. This input data is split into domain-specific clusters such as: technical indicators clusters, country clusters, or domain clusters. The KAL module first runs the Dynamic Time Warping (DTW) algorithm, which is based on Euclidian distance for individual domain-clusters; then visualization techniques, like Hierarchical Clustering Analysis (HCA), group the global dataset based on the clusters they belong to. The resulting output from the KAL module saves the different clusters formed through the HCA and offers conditional computation where training sets are active based on the input prediction problem. In theory, this proposed approach may dramatically increase the model's predictability at lower computational costs than traditional methods because it performs training on the relevant data. Users should take note that the DTW-HCA can be a computationally long process; hence, it is recommended to conduct the KAL at certain time-intervals; such as: weekly or monthly.

Next, based on the input parameter and the output from the KAL, the second module, which includes a logic-based decision-gate, determines the training examples to use. The selected training dataset is then fed into the Sequence Learning Gate (SLG). The SLG is a deep LSTM neural network that has a dropout operation (threshold 0.2) to reduce the losses on the training data. Other similar sequence learning models can be adopted. The final output of this component is a time-series output forecast as shown in figure (4-4). To illustrate how this component works, assume the following hypothetical example. The input data is number of mentions for protests for the USA from GDELT database. The KAL module stored the relevant variables in three groups: technical cluster (Goldstein score, tone), country clusters (UK, Brazil), and sub-domain cluster (mass violence and fight). The decision is made since these variables are in the same group as the input variable i.e. number of mentions for “USA” and domain “protests”. Separate LSTM models are trained for different cluster groups. The output is a weighted ensemble of the different models. The final output is the forecast for US Protests in t+1 based on domain-knowledge of similar features. In a similar manner, the ensemble learning model can be looped for different entities and domains of interest to monitor them on a daily or real-time basis.

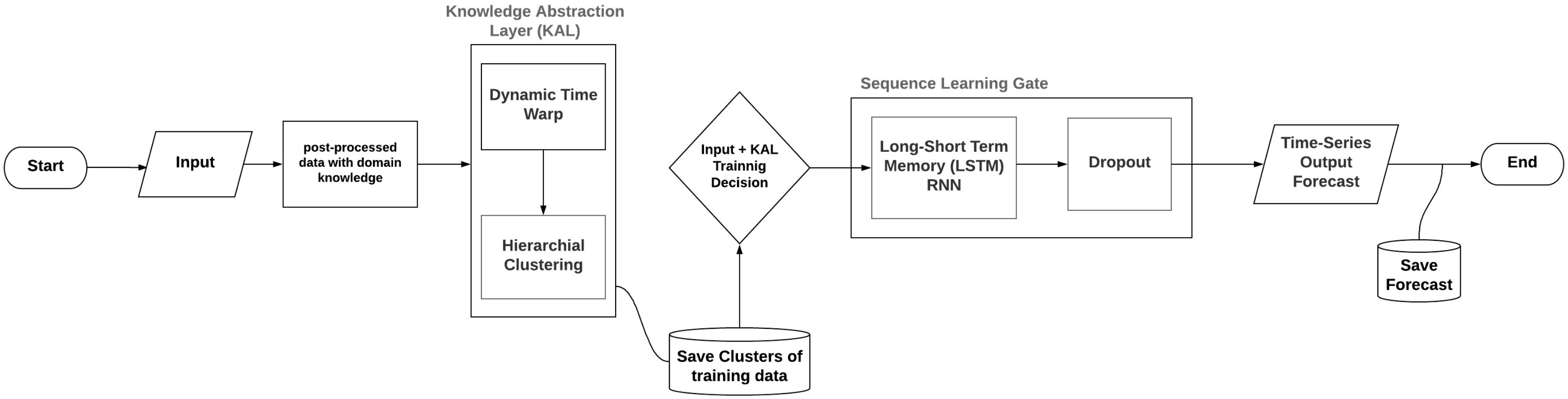

Knowledge Abstraction Layer (KAL)
Start
Input
post-processed data with domain knowledge
Dynamic Time Warp
Hierarchial Clustering
Save Clusters of training data
Input + KAL Trainnig Decision
Sequence Learning Gate
Long-Short Term Memory (LSTM) RNN
Dropout
Time-Series Output Forecast
End
Save Forecast

Figure 4-4. Generalized Architecture for Sequence Prediction Problems

4.3. Ensemble Models for Structured Output

Building on top of the sequence prediction architecture which was achieved through the previous model's component, there are potentially four approaches to build deeper ensemble programs for structured inference are illustrated in Figure (4-5). The forecast output from the SLG is fed into three separate models whose results are combined together to provide more details and a structured output based on the predictions.

The first approach in ensembling models can help get obtain high correlation and low error final results. The second approach is the Monte Carlo Simulations (MCS) that can be used to build a more controlled environment for stress-testing the prediction results. The third approach is the class label problem or, as formally defined here, the Time Label Assignment (TLA). In the TLA, selected memory is added to the time series model, where the output provides a daily prediction of similarity in most recent pattern with saved memory. Last but not least, the uncertainty quantification (UQ) module can leverage state of the art UQ techniques given the ensemble model.

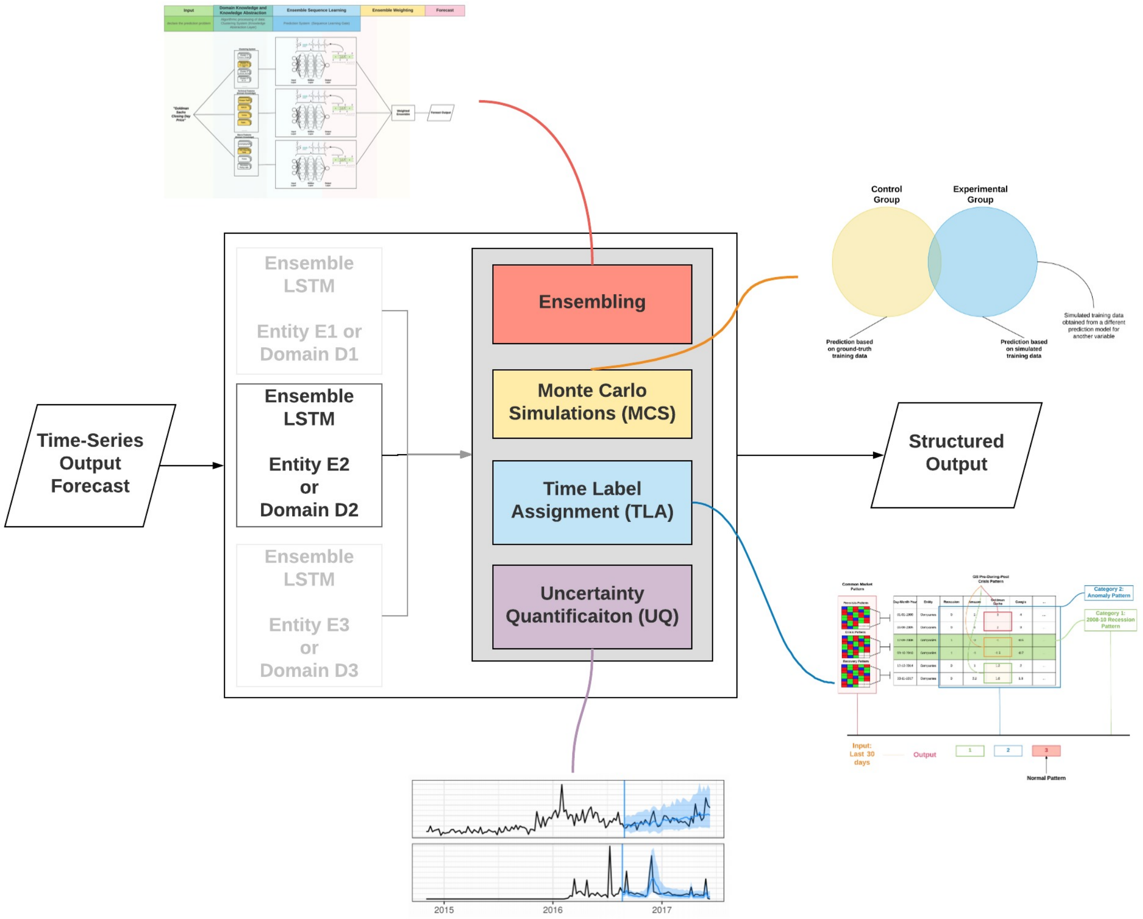

Input
Domain Knowledge and Knowledge Abstraction
Ensemble Sequence Learning
Ensemble Weighting
Forecast
Ensembling
Monte Carlo Simulations (MCS)
Time Label Assignment (TLA)
Uncertainty Quantificaiton (UQ)
Control Group
Experimental Group
Prediction based on ground-truth training data
Prediction based on simulated training data
Simulated training data obtained from a different prediction model for another variable
Time-Series Output Forecast
Ensemble LSTM
Entity E1 or Domain D1
Ensemble LSTM
Entity E2 or Domain D2
Ensemble LSTM
Entity E3 or Domain D3
Structured Output
Common Market Pattern
Precrisis Pattern
Crisis Pattern
Recovery Pattern
GS Pre-During-Post Crisis Pattern
Day-Month-Year
Entity
Recession
Amazon
Goldman Sachs
Google
Category 2: Anomaly Pattern
Category 1: 2008-10 Recession Pattern
Input: Last 30 days
Output
1
2
3
Normal Pattern
2015
2016
2017

Figure 4-5. Generalized Module Architecture for Post-Sequence Prediction and Going Beyond Black Box

4.3.1. Monte Carlo Simulations (MCS) Approach

Perhaps the most useful and simple approach to untangle the deep network is using MC simulation to do stress-testing within the confines of the generalized ensemble models. In the controlled experiment setting, there are two model outputs, as illustrated in figure (4-6): (a) the ensemble prediction based on the ground-truth training data-set i.e. the control group; and (b) a model simulated training dataset to have the same forecast i.e. the experimental group. The experimental group with simulated training data can help uncover the sensitivity of forecast outputs to specific training data. This approach will help go beyond black box by treating the model in a controlled experiment setting. In a controlled experiment, an independent variable is the only factor that is allowed to be adjusted, with the dependent variable as the factor that the independent variable will affect. This, in return, will provide a deeper engagement with decision-makers to address the impact of movement in one feature impacts the predicted output.

There are two powerful applications of this approach. First is the case of predicting quarterly GDP growth. The control group yields the forecasts from ground-truth data, while the experimental group will simulate two new data points for the latest period to update training data for GDP growth prediction. Furthermore, as an example the experimental group will inform that an *x*-percentage point change in public debt yields the lowest drop in GDP growth forecast compared to other features in the training data.

The second application is a more sophisticated prediction and inference approach. Take that the decision-maker is interested in a separate variable, for example the government bond yield, and more specifically, the 10-year maturity US T-bill yield. Now assume that the T-Bill yield is found to be most sensitive to fluctuations in GDP growth. The control group in this setting is predictions based on the ground-truth for T-Bill yield; whereas the experimental group in the second model takes the new information of GDP growth forecast from the control group of the first model. Consequently, adding the forecast from the first model in the second model's experimental group i.e. training the data-set using updated GDP growth forecast will yield an updated forecast of T-Bill yield, based on new information.

Neural networks are highly non-linear where the different layers and nodes capture these interdependencies between entities, domains, and risk-classes. However, decoupling these non-linear dynamics has not been transparent so far and remained a black box. In this regard, the MC simulation approach provides a simple yet powerful way to untangle the black box for deeper insights to inform decision-making in domains of interest to us.

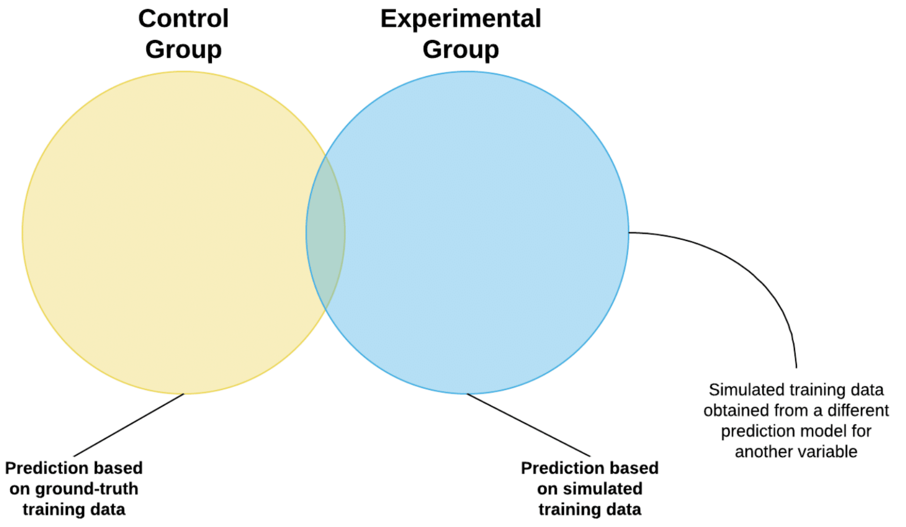

Control Group
Experimental Group
Simulated training data obtained from a different prediction model for another variable
Prediction based on ground-truth training data
Prediction based on simulated training data

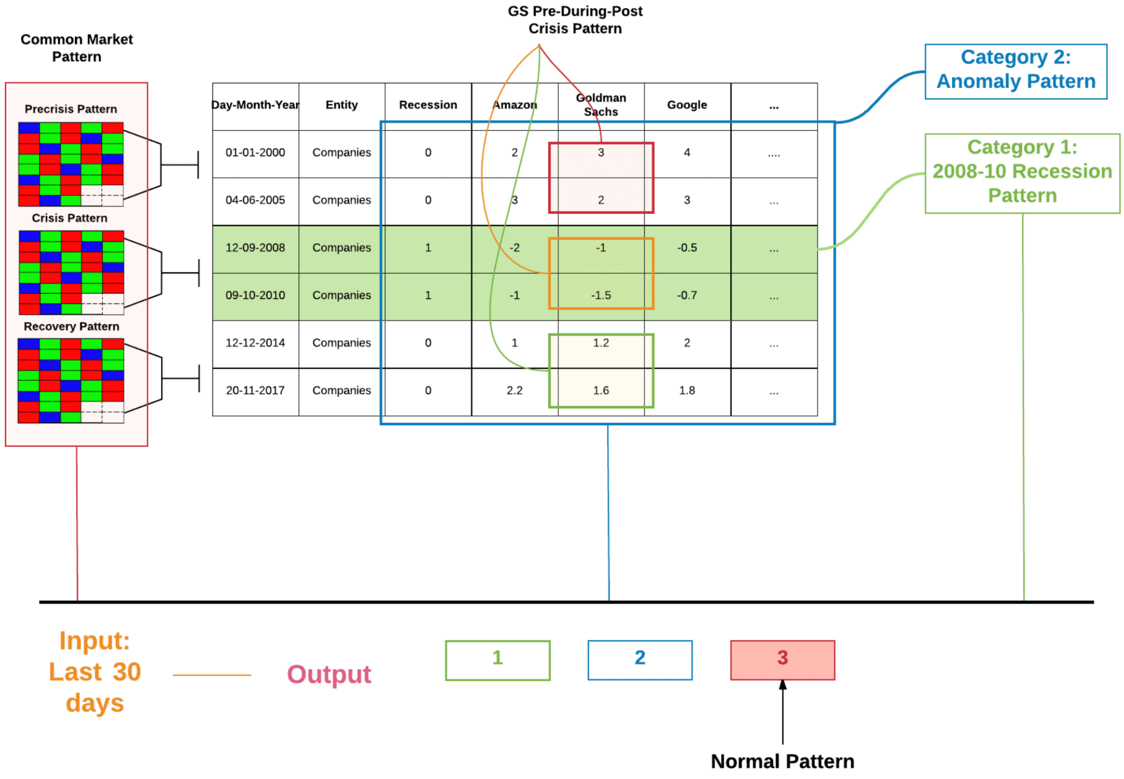
Figure 4-6. Controlled Experiment Setting for Stress-Testing

4.3.2. Time Label Assignment (TLA) Approach

Using class labels**,** the TLA approach can convert the problem into label predictions. The idea is similar to the one used in several previous studies in image recognitions that aim to identify entities and dynamics of entities in a given image, using various ways to have labels including based on joint image/label embedding, label co-occurrence dependency, and pixel-level labeling (Wang et al. 2016, Pinheiro and Collobert 2015). A big drawback in time-series random variables of interest to us is that labels are largely missing. There are two new innovations proposed here: the labels for specific time-periods will be treated in a special manner; and, use algorithmic approaches to convert random time-series variables into meaningful labels.

To illustrate this special treatment, the case of daily sample data of S&P 500 stock prices is used (Figure 4-7) where the selected memory variable is "Recession". Therefore, assuming the period between 12-09-2008 and 09-10-2010 is labeled as category 1, the daily temporal features of each company as well as the common pattern between companies can be extracted for the pre, during and post Category 1 crisis period. Moreover, category Label 2 can represent common pattern between companies to define an anomaly pattern i.e. a pattern dissimilar to historic series. The output will be a switch that is updated every-day to give an output of either 0, 1 or 2; where "0" represents dissimilar pattern to earlier crisis period, "1" represents common stock market pattern as period prior to 2008-10, and "2" represents an anomalous common pattern that is dissimilar to any point of time in history.

In addition, this model approach adds selected memory to treat certain time-periods in a special label; for example, the 2001 crisis can be a new category 4. This method can also be scalable in domains of missing labels. The data from various sources can be filtered using the z-threshold peak detection algorithm or Bayesian anomaly change point algorithm to create labels; then a special treatment of time labels based on the added selected memory. For example, specific days of big mass violence or economic crisis like asset bubbles can be added to the selected memory; and the resulting output will be binary or categorical labels to match the high-frequency forecast with historic patterns obtained from selected memory.



Day-Month-Year	Entity	Recession	Amazon	Goldman Sachs	Google	...
01-01-2000	Companies	0	2	3	4	....
04-06-2005	Companies	0	3	2	3	...
12-09-2008	Companies	1	-2	-1	-0.5	...
09-10-2010	Companies	1	-1	-1.5	-0.7	...
12-12-2014	Companies	0	1	1.2	2	...
20-11-2017	Companies	0	2.2	1.6	1.8	...

Figure 4-7. Example of Selected Memory for Time Label Assignment Prediction

Chapter 5: Case Studies

This chapter provides case studies for the proposed methodology. The cases presented covers a number of domains; including: (a) Economic Domain: the case will demonstrate a state-of-the-art prediction of economic growth and real-effective exchange rate (REER) that yields substantially higher accuracy than current industry standard IMF GDP growth forecasts. Further, a test-example of MC simulations shows that there is enough sensitivity of GDP growth output forecast to changes in the training data, (b) Financial Domain: This case will explore the structured logic approach in depth for the ensemble-model prediction for closing day stock price; and (c) Geo-political Domain: These cases will present results for the geopolitical environment.

International organizations have to monitor multiple assets around the world. For example, the Department of Defense (DoD) or Department of Homeland Security (DHS) manages a global real property portfolio of more than 562,000 facilities (Department of Defense 2017, Homeland Security 2017). Corporations like Bechtel or Lockheed Martin (LM) have finished over 23000 engineering projects worldwide (Lockheed Martin 2017, Bechtel 2017). Similarly, many other multinational corporations (MNCs) across industrial sectors have multiple assets (or projects) in a given country. Executives at MNCs and international governments receive frequent updates on the state of global risks to make decisions that help them make judgments to protect their assets (Federal Reserve Board of Governors 2017, International Monetary Fund 2017, World Economic Forum 2017, United Nations 2017, Accenture 2008).

Many of these organizations use quarterly or annual updates that are considered a static form of risk communication. At the same time, in the instance where organizations use short-time scale dynamic monitoring methods, they use proprietary data and models.

The case studies present opportunities to standardize data, models, and visualizations that can aid-executive decision-making. In the following sub-sections, economic, financial, and geo-political domains are discussed. The case studies provide validation of the proposed model for domain-specific sequence learning predictions. The case studies also provide a framework to track disruptive mass movements such as protests to inform policy-making concerning asset protection. The other case studies provide how emerging technologies can help track economic recessions and opportunities for economic development. First, the dataset used for the specific domain is described. Second, the ensemble sequence model framework is presented. The following chapter presents the data and model framework.

5.1. Economic Opportunities and Threats

GDP growth and government bond yield are two key macro-economic signals that decision-makers care for in countries. Economic growth is the increase in the inflation-adjusted market value of the goods and services produced by an economy over time. It is conventionally measured as the percent rate of increase in real gross domestic product, or real GDP. Government bond yields is the interest that the government pays to borrow money for different lengths of time.

Beyond macro-economic signals, micro-economic signals are equally important for identifying opportunities and threats in mega-projects. For example, take the case of a large-data center project. The decision-maker would find it valuable to know the value-added growth in computer services, talent pool i.e. job growth in higher in computer network engineer professions, aggregate economic-shifts across cities to inform their business strategy. The next section provides details on the economic dataset.

5.1.1. Dataset

Here the organization and retrieval of economic data from Bank of International Settlements (BIS), International Monetary Fund (IMF), and World Bank Global Economic Monitor (WGEM) is briefly discussed.

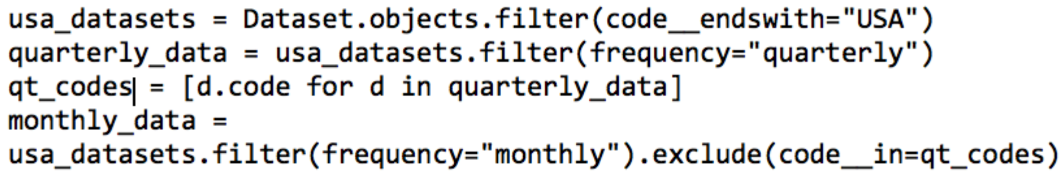
The basic idea, architecture and software used are similar to as described in the control architecture section and knowledge base layer in Annex 2. Essentially two models are available "Databases" and "Datasets". The "Database" model stores the data of the QUANDL database that are being accessed. The Database contains all the different indicators and time-series that are necessary to be acquired. The "Dataset" model contains the actual time-series data for a specific indicator. These models are defined using Django ORM and the data can be easily accessed using the same. Some details about the fields used in the above-mentioned model are as given below.

The database has following attributes: name, code (Unique code provided by Quandl), API url (for metadata about the database e.g. description), codes url (for all the possible dataset codes under that database). The dataset has the following attributes: Name, code, database (foreign key), start date, end date, frequency, data (the actual time-series), description. Data is also collected from World Bank Global

Economic Monitor. “NY.GDP.MKTP.PP.KD” is one of the indicators used. The particular variable for example is “GDP, PPP (constant 2011 international $)”. The broad list of macro-indicators can also be made available.

There are some preprocessing data cleanup attributes**.** For example, countries only with a valid ISO 1366_3 code are considered. Countries with all NaNs are dropped. Prediction level abstraction and inclusion of WGEM, BIS, or IMF database into Postgresql.

Now with just a few lines of code any frequency, indicator and country from the WGEM data can be accessed. All of the preprocessing and sorting is handled internally. An example code snippet is given below.



```
usa_datasets = Dataset.objects.filter(code__endswith="USA")
quarterly_data = usa_datasets.filter(frequency="quarterly")
qt_codes = [d.code for d in quarterly_data]
monthly_data =
usa_datasets.filter(frequency="monthly").exclude(code__in=qt_codes)
```

5.1.2. Economic Model Architecture

The macroeconomic system is designed in the following manner. The various data sources are integrated accompanying domain pre-processing, such as calculating natural log difference for GDP growth and change variables. The integrated data is extracted using Euclidian distance measure to produce a MDS image and (or) Hierarchical Clustering image. This step is conducted very infrequently, for example once a quarter for the case of macro features. These clusters are saved and then fed in the LSTM RNN model. These results provide a time series output forecast. An alternative sample can be obtained by switching the training data set with alternative numbers using MC simulation to provide an adjusted output after shock variable. Figure (5-1) illustrates the model architecture.

An alternative presentation for class labeling problems can also be generated using the z-threshold or Bayesian change-point algorithm described in earlier sections.

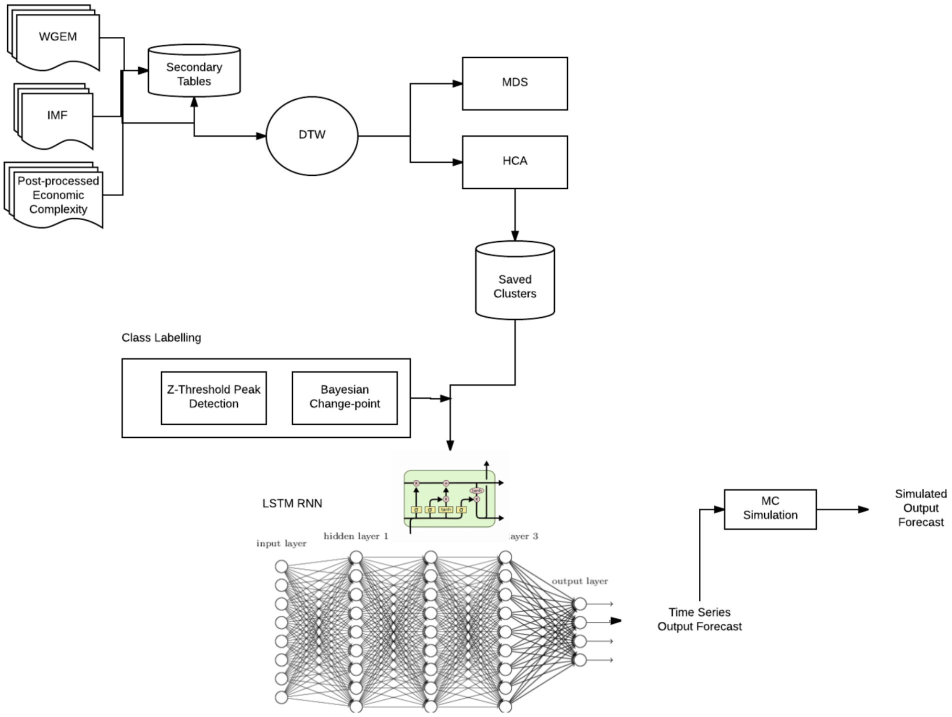


Figure 5-1. General Architecture for Macroeconomic System Model

5.2. Financial Opportunities and Threats

Large engineering companies like Lockheed Martin, Bechtel, as well as governments and international organizations have a treasury department. In larger firms, Treasury may include trading in bonds, currencies, financial derivatives and the associated financial risk management. These organizations manage global assets with a diversified portfolio in international capital markets. The Treasury management helps to manage the organizations liquidity and mitigate operational, financial and reputational risks. In addition to Treasuries, hedge-funds, investment banking industry closely monitor various asset classes, such as a Fixed Income or Money Market – buy and sell interest bearing securities, Foreign exchange or "FX" - buy and sell currencies, Capital Markets or Equities desk – buy and sell shares listed on the stock market. There are a host of derivate such as ETF and leveraged indexes that can be high yield but volatile. The model and results presented here provide valuable insights for trading strategies with a focus on equity market strategy.

Many banks such as JP Morgan or Goldman Sachs are central lending institutions for LEP's. The performance specific-asset classes provide important signals for other investors. The mass-movement of the stock market can also inform important attributes of country of the project.

Mega-projects are capital-intensive and require complex financing arrangements due to the large-scale nature of highway, building, transportation systems. Project financing requires consortium of investors, lenders and other participants to undertake infrastructure projects that would be too large for individual investors to underwrite. the Trans-Alaskan pipeline and exploration and exploitation of the North Sea oil fields provide interesting examples of financing structures. The identification of instruments, sectors, and market prediction can help inform overall market risk that can inform project risk.

High frequency financial data can be insightful for many important country and project decisions. For example, after a disaster, companies vested in housing, transport, or infrastructure from that location would have a lot at stake. Many public companies can use the disaster signal to gain large government contracts and bump up their prices. Such financial opportunities and threats also be important economic shifts. For example, a banking crisis or asset bubble burst could lead to catastrophic financial melt-down. The assets of an enterprise in the given location could be frozen or lose value. Broadly

Such signals have also been used in domains beyond finance including national security. Projects such as MARKINT and others studies use complex credit default swaps (CDS) and other complex derivate behavior to inform on national security threats. Large-scale events that take place in short run frequency such as tomorrow or the following weeks can be predicted using hi-frequency financial data.

5.2.1. Dataset

The source of all data is Quandl. Crunchbase (CB) is used for mapping companies to their headquarter location and extract spatial features of a given company. CB data integration will also be useful for standardization visualizations to spatial-temporal coordinate space. There are broadly two divisions of indicators from the database technical indicator cluster and company clusters.

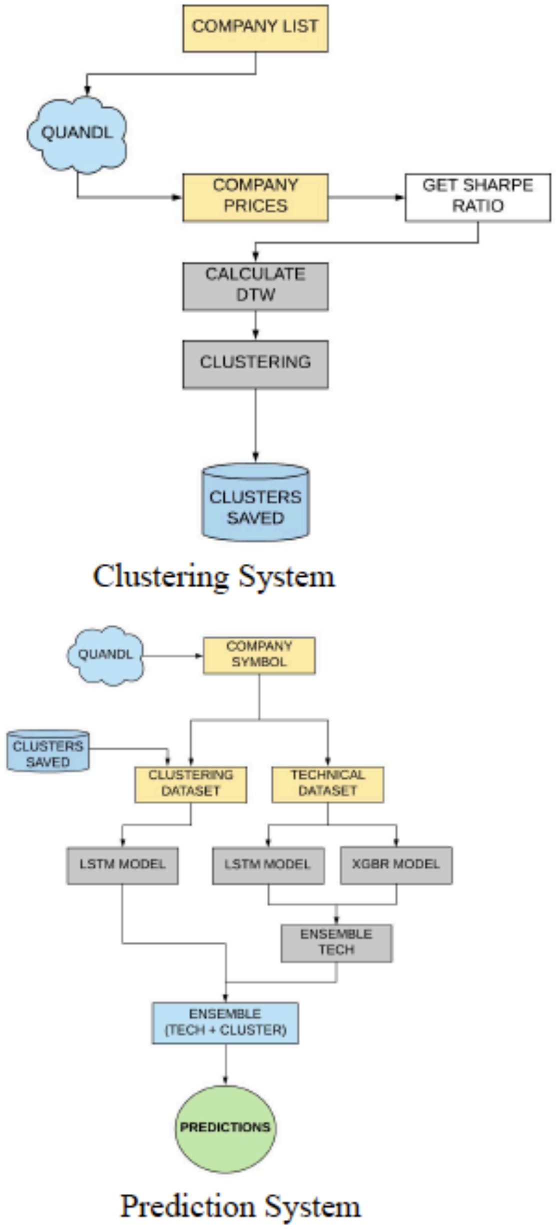
The **technical indicators cluster** contains only Ticker related data. The sole purpose is to see what predictions are given when only Ticker related features are used. For example, for GS (Goldman Sachs), Indicators such as Open, Low, High, Close along with Momentum, Vortex, Sharpe Ratio, etc. will be used to make predictions.

The **company clusters** has a set of indicators to capture moment of market for particular type of company. For example, from SP500 companies a cluster in extracted which contains set of companies. These clusters are based on DTW-Hierarchical Clustering for companies in same cluster that have shown similar time-sequence trend. The company clusters can give a broader output which are based on movement of different stocks in market.

5.2.2. Financial Model Architecture

The financial system architecture is presented succinctly below. The system consists of two main sub-systems. First, the clustering System, and second, the prediction system as shown in Figure (5-2).

The illustration of the ensemble prediction component i.e. the third component is presented for finance domain is presented in Figure (5-3). This illustration shows the specific case of Goldman Sachs prediction system where the input is Goldman Sachs closing day’s price. The pre-processed domain knowledge can have multiple clusters; however, for simplicity, technical indicators (Sharpe ratio, Vortex indicator), company cluster (Charles Schwab, Berkshire Hathway, JP Morgan), and macro clusters (T-bill maturity yield) are the processed domain knowledge. Based on the decision-gate, dissimilar features to the prediction problem are dropped. Separate ensemble LSTM model based on each cluster training provide predictions based on each training set. A weighted sum of ensemble models provide the final forecast.



Clustering System

Prediction System

Figure 5-2. Clustering and Prediction System for Stock Price data

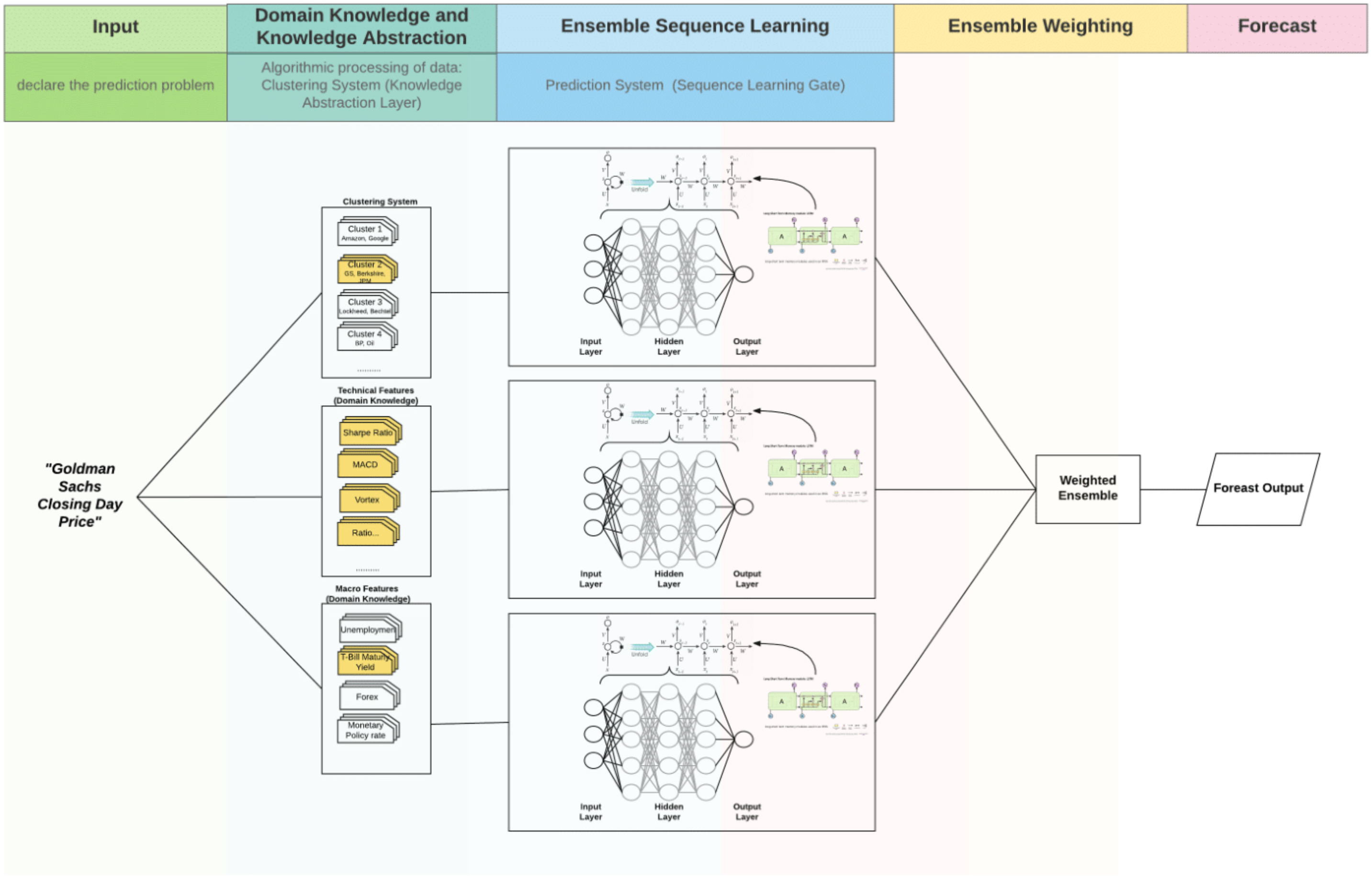

Input
Domain Knowledge and Knowledge Abstraction
Ensemble Sequence Learning
Ensemble Weighting
Forecast
declare the prediction problem
Algorithmic processing of data: Clustering System (Knowledge Abstraction Layer)
Prediction System (Sequence Learning Gate)
"Goldman Sachs Closing Day Price"
Clustering System
Cluster 1
Amazon, Google
Cluster 2
GS, Berkshire, JPM
Cluster 3
Lockheed, Bechtel
Cluster 4
BP, Oil
Technical Features (Domain Knowledge)
Sharpe Ratio
MACD
Vortex
Ratio...
Macro Features (Domain Knowledge)
Unemployment
T-Bill Maturity Yield
Forex
Monetary Policy rate
Input Layer
Hidden Layer
Output Layer
A
Weighted Ensemble
Foreast Output

Figure 5-3. Prediction System Architecture, Example Goldman Sachs Stock Price

5.3. Geopolitical Opportunities and Threats

The main source of data for geopolitical movements is from the Global Database of Events, Language, and Tone (GDELT) which are aggregated through millions of real time news article, translations, search queries, herd-traffic behavior on the web. However, it is important to be cognizant of the availability of heuristics with such data. The classic example is that of newspaper reporting a plane crash versus a car crash. In reality, there is a higher probability of occurrence for car crash, and it happens all the time but we do not hear about it because reports are published, mainly, on extreme events, such as plane crashes, or hijacks, when there is much greater risk of many people being killed. Nonetheless, due to availability heuristics, people may gain a false impression that car crashes are not a major event. Moreover, there are uncertainties associated with such estimates. There is a growing body of literature that has shown that many such communication technologies like Twitter are used by elites in the political sphere; hence, creating a bias to collect national sentiment. (Stier 2016). Similarly, media can be biased towards certain topics. In some instances, this information may not capture the gravity of the situation or miss some data. For example, the hurricanes in Puerto Rico in 2017 was not well covered by newspapers; and hence, would not show a big movement in the GDELT database. Furthermore, in 2016 there were total 47 records of port strikes – with 47 percent (22 records) with correct location, 9 percent (4 records) port strikes with incorrect location (NIKNEJAD et al. 2017).

Going beyond these statistical uncertainties, GDELT provides valuable real-time geopolitical activity that are classified in details, and the “number of mentions” variable provides a good proxy of how the global public sphere reacts to geopolitical movements; such as: big policy shifts, national leaders, military action, and social movements.

5.3.1. Dataset

The GDELT data “monitors the world's broadcast, print, and web news from nearly every corner of every country in over 100 languages and identifies the people, locations, organizations, counts, themes, sources, emotions, counts, quotes and events” (Leetaru and Schrodt 2013). The events data records over 300 categories of manmade events from protests and violence to diplomatic exchanges. The Global Knowledge Graph (GKG) contains data on mentions of organizations, themes, locations, people, tone of language, dates, counts, etc. in the news.

Consequently, there are many defense, intelligence, and private sector risk mapping examples that have used GDELT database. For example, GDELT has been used for mapping the action of geopolitical actors, monitoring big events; such as: the Arab Spring or protests or mass movements like the refugee crisis. The data is now available and is updated every minute with the results, in subsequent, are focused on daily data. GDELT uses Google bigquery to organize, process and provide real-time data, with one-minute updates by latest revisions. Visualizations of the GDETL events database and the global knowledge graph are presented in Figure (5-4).

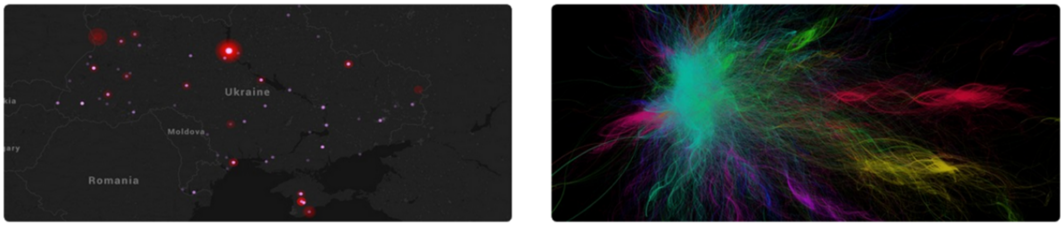


Figure 5-4. Visualizing GDELT Events Database and Global Knowledge Graph
Source: GDELT 2016.

5.3.2. Geopolitical Model Architecture

Domain knowledge for GDELT is processed by extracting several technical features based on the help of listed features, that include the average of Number of Mentions, AvgTone and GoldsteinScale (described in Chapter 4.1.3) in the lag day range. The percentage contribution of Number of Mentions (herein referred to as NM) in overall event on same day is also taken as a feature. Other mathematical operations; such as: standard deviation, skewness, kurtosis for NM as well as composite measure using all event counts to measure entropy are accommodated as features. Ensembling all the features proves to be a powerful set of input for the algorithms that will be discussed in the latter sections.

The clusters are extracted by using the following two general methods: First, DTW distances between number of mentions for the event code of interest for varied data-groups are computed, and then clustering; such as: K-means, Hierarchical, are applied to build optimal training data-sets that is important for the dependent variable prediction.

As an example, raw data from GDELT is presented is presented in figure (5-5). The figure shows the NM variable for the United States daily frequency for risk indicator protests. The special dates of known protests are highlighted between December-2013 - September, 2018. These similar variables will be used for prediction problem in geopolitical environment.

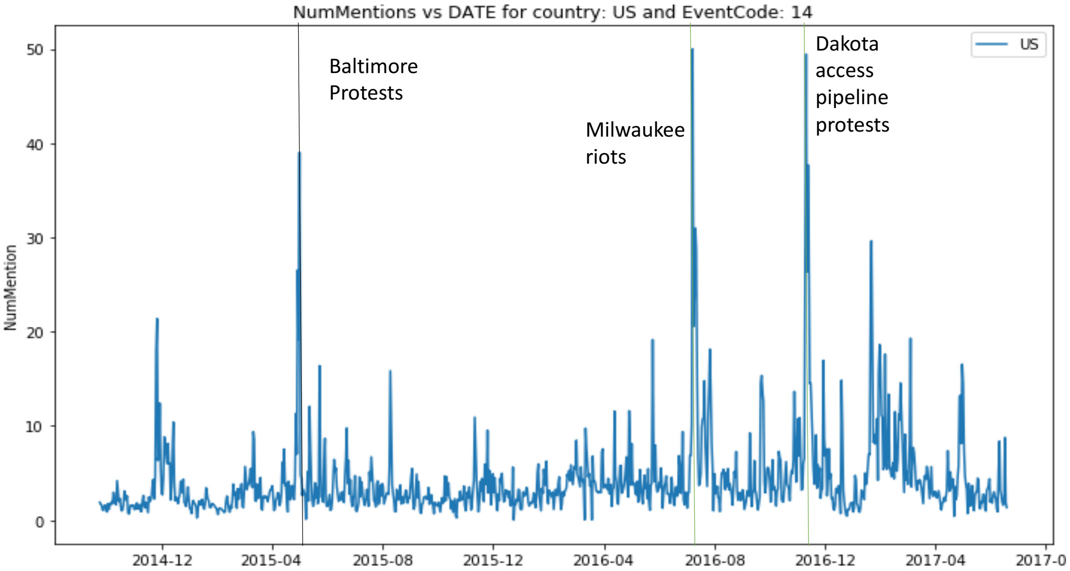


Figure 5-5. Distribution of Number of Mention of Event (14 for Protest) for country US.

5.4. Training Across Domains

Now using stacks of features meaninglessly will clearly lead to overfitting. It is to be noted that adding a large number of training sample (*n*) will always not lead to overfitting. However, a large number of features (*p*) can cause overfitting issues in neural networks. Thereby, in order to address selection and contextualization of training data in our domains of interest such as economics, finance, or geopolitics, it is important to abstract groups of features for training data for a given prediction problem. The goal is to abstract out, have flexibility and modularity to use different methods, however in the following fashion.

The generic abstraction layer for data training groups for a given entity-domain for different domains is presented in figure (5-6). The DTW-HCA tiered approach to similarity and clustering assessment for static and longitudinal data is valuable to incorporate non-linearities in time and a time-dependent framework for trend-matching across multiple sources. Nevertheless, DTW struggles with longitudinal data that is of highly disparate ranges. Some standard tricks that have been used are the following. First, bin data by length of operations - < 6 months, 1-3 years, 3 – 10 years, 10 years or more and then perform DTW on each one to assess their similarity. Second, ignore data that has less than 1 year of data in the similarity assessment. Third, leverage pattern of life identification and extract features from the data, without having to time warp is an extended option. This provides trend, seasonality, and other cyclical features that can be compared and assessed for similarity as opposed to direct comparison with DTW.

In addition, this framework can be extended once the set of features for the longitudinal data has been extracted, this data can be combined with other features (sector, location, etc.) and then a cosine similarity assessment can be done. This is straightforward and looks a lot like the Euclidean and Ward distance measures without having to scale/normalize features, and it can deal with categorical data. Most importantly, using cosine distance as a metric for Hierarchical Agglomerative Clustering (HAC), we know naturally the cutoff point for the number of clusters. As clusters cannot be more similar or dissimilar than 1, we can determine the number of clusters automatically. With other techniques, the cutoff point for clusters is ill-defined.

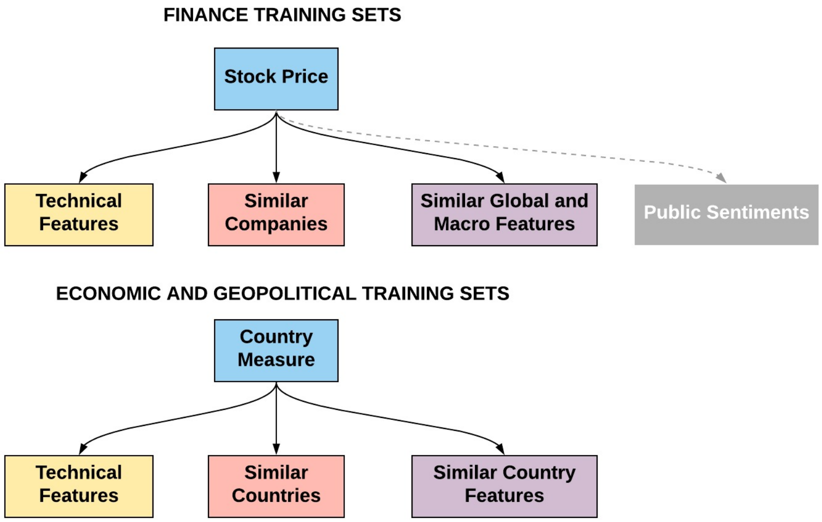


Figure 5-6. Different Groups of Training Data Set, Example for Domains

Chapter 6: Results

This chapter presented the results for experiments across economics, finance, and geopolitics domains. To the extent possible, a standard structure is followed across domains, nevertheless, given complexities and uniqueness of data sets, availability of training features, there is heterogeneity in model building and parameter tuning across domains. To this affect, the research project aims to build standardized data harvesting, model, and visualization class objects to have a pipeline that is replicable, scalable, and flexible.

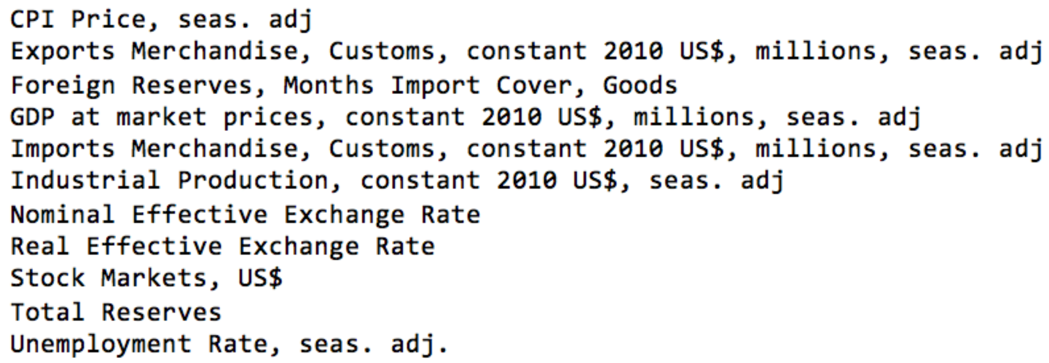
There are commonalties across the pipelines from different domains. For example, technical features generation could be domain specific to generate useful training features; DTW-HCA clustering could be used to define training groups; RNN variant models used for time-series forecasting; MC dropout method for uncertainty quantification, and so forth. However, there are many idiosyncrasies associated iwth managing and analyzing each asset-specific prediction problem. Consider the dependencies between the different training features raising important questions concerning dependencies, causality, and endogeneity; however, the focus up till now has been to use some of the knowledge on these attributes in a structured manner with the goal of improving prediction models by accommodating domain knowledge.

There are few different options considered in this work to ensemble models. Ensembling models are considered “meta-algorithms” to decrease variance (bagging), bias (boosting), or improve predictions (stacking) (Smolyakov 2017). In context of bagging, the same approaches as *k*-means or hierarchical clustering can be conducted to classify the different models’ accuracy. A weighting mechanism based on the analysts’ understanding of the model results is the simplest option that is discussed for specific experiments. The point is that there will be overfitting and underfitting models for the same prediction using different training data or different models. This information can be used to build a stronger model that fits the ground-truth data just right. At the same time, there are other options as well, including using parametric models to learn the optimal parameters. Many of these attributes are mechanical but also about the art of doing machine learning.

The results shown in this chapter will use different methods to ensemble for different domains. For example, bagging is used for the exchange rate predictions, weighted average ensembling are used for stock prices and geopolitical data on protests, and voting classifier for macroeconomic experiments for country level GDP growth predictions.

6.1. Results for Economic Domain

Based on the GDP and relevant features from WDI and WGEM data, the predictions for US GDP growth are presented here. The following is a list of selected indicators from which country training features are selected:



```
CPI Price, seas. adj
Exports Merchandise, Customs, constant 2010 US$, millions, seas. adj
Foreign Reserves, Months Import Cover, Goods
GDP at market prices, constant 2010 US$, millions, seas. adj
Imports Merchandise, Customs, constant 2010 US$, millions, seas. adj
Industrial Production, constant 2010 US$, seas. adj
Nominal Effective Exchange Rate
Real Effective Exchange Rate
Stock Markets, US$
Total Reserves
Unemployment Rate, seas. adj.
```

The prediction models are built both for quarterly and annual series, however, only annual forecasts are shown here. After clustering and all following indicators were selected. An ensemble of GRU and LSTM models are presented.

First, the results from the DTW-Hierarchical Clustering are presented. Many advanced economies are clustered together and developing countries that are resource rich, versus in the middle-income stage are clustered together. This informs an important attribute of which countries are closer to each other and useful information for future generalized training samples for unique prediction cases. The world cluster map is presented in Figure (6-1) identifying similar clusters by their color.

Similarly, clustering can also be conducted for testing similarity between indicators for a given country. Figure (6-2) shows a hierarchical cluster of key macro-economic variables for the United States based on the DTW distance. The full spectrum of data and localized abstraction are presented. This shows that reserves, unemployment, and liquidity variables are in the same cluster, whereas financial indicators like stock prices and export patterns in a different cluster together. A non-exhaustive list of training features for country-time macroeconomic feature are presented in table (6-1).

Finally, Figure (6-3) shows the summarized prediction results for US GDP growth using various deep neural network models including LSTM, GRU, as well as gradient boosting algorithms. This simple chart shows that while LSTM and GRU are good fit for catching the trough points, gradient boosting methods are better for capturing the trend in recent period. Each model captures non-linear dependencies in its own way with benefits and deficiencies. The prediction results provide interesting insights. The model results with UQ and comparisons with IMF estimates are reported next.

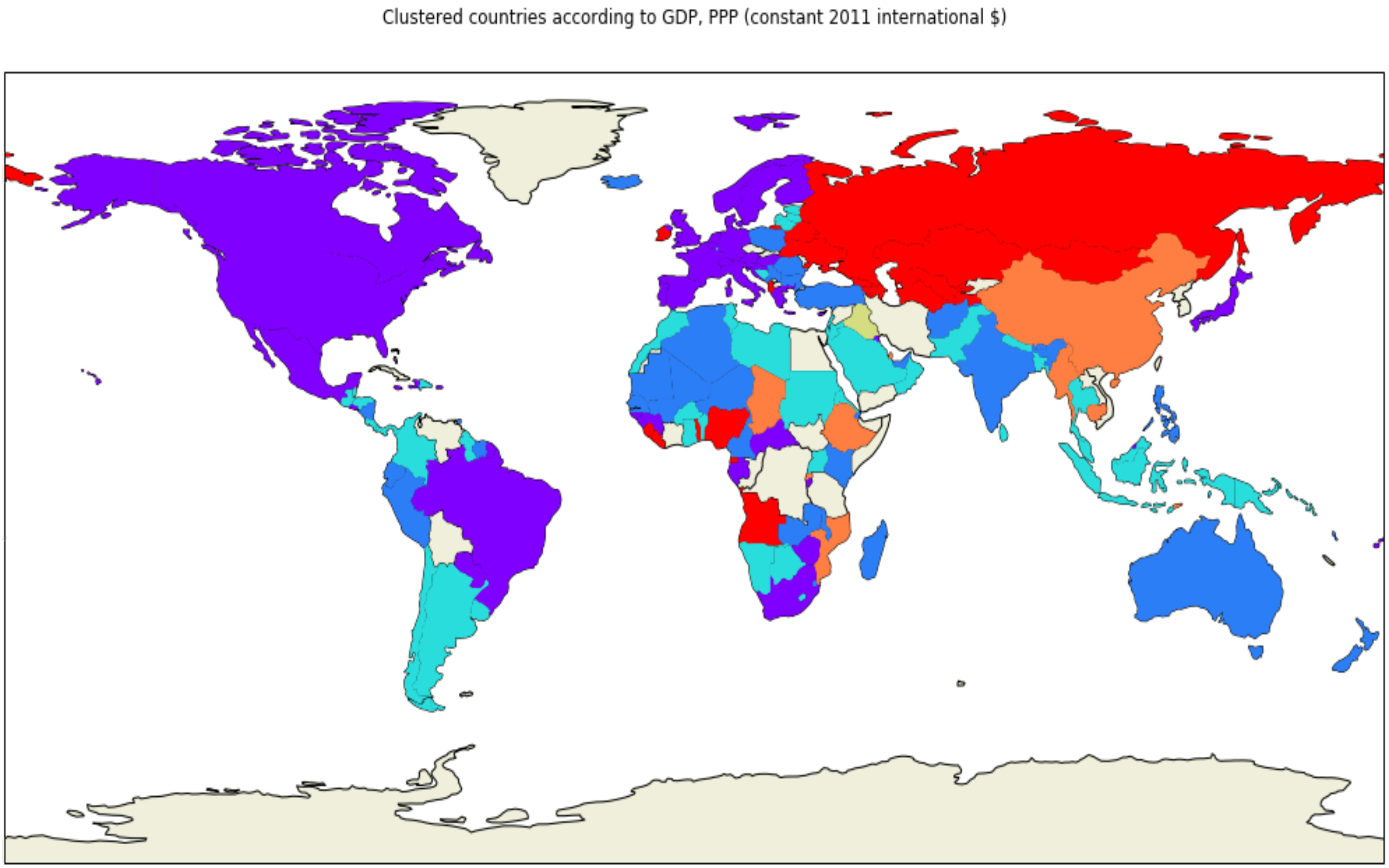

Clustered countries according to GDP, PPP (constant 2011 international $)

Figure 6-1. Hierarchical Clustering Map of the World based on GDP time-trend

Table 6-1. Selected List of Country Features for country GDP growth prediction

Macro Structural Features	Macro Financial Features	External Accounts	Social and Public Finance
Agriculture, value added (% of GDP)	Central government debt, total (% of GDP)	Current account balance (% of GDP)	Final consumption expenditure, etc. (% of GDP)
Coal rents (% of GDP)	Claims on other sectors of the domestic economy (% of GDP)	Exports of goods and services (% of GDP)	Revenue, excluding grants (% of GDP)
Discrepancy in expenditure estimate of GDP (current LCU)	Claims on central government, etc. (% GDP)	External balance on goods and services (% of GDP)	Research and development expenditure (% of GDP)
Forest rents (% of GDP)	Domestic credit provided by financial sector (% of GDP)	Merchandise trade (% of GDP)	Tax revenue (% of GDP)
GDP (constant 2010 US$)	Domestic credit to private sector by banks (% of GDP)	Trade (% of GDP)	
GDP growth (annual %)	Foreign direct investment, net inflows (% of GDP)	Imports of goods and services (% of GDP)	
Industry, value added (% of GDP)	Foreign direct investment, net outflows (% of GDP)		
Manufacturing, value added (% of GDP)	Net acquisition of financial assets (% of GDP)		
Military expenditure (% of GDP)	Net incurrence of liabilities, total (% of GDP)		
Mineral rents (% of GDP)	Net investment in nonfinancial assets (% of GDP)		
Natural gas rents (% of GDP)	Net lending (+) / net borrowing (-) (% of GDP)		
Services, etc., value added (% of GDP)	Stocks traded, total value (% of GDP)		
Total natural resources rents (% of GDP)	Market capitalization of listed domestic companies (% of GDP)		

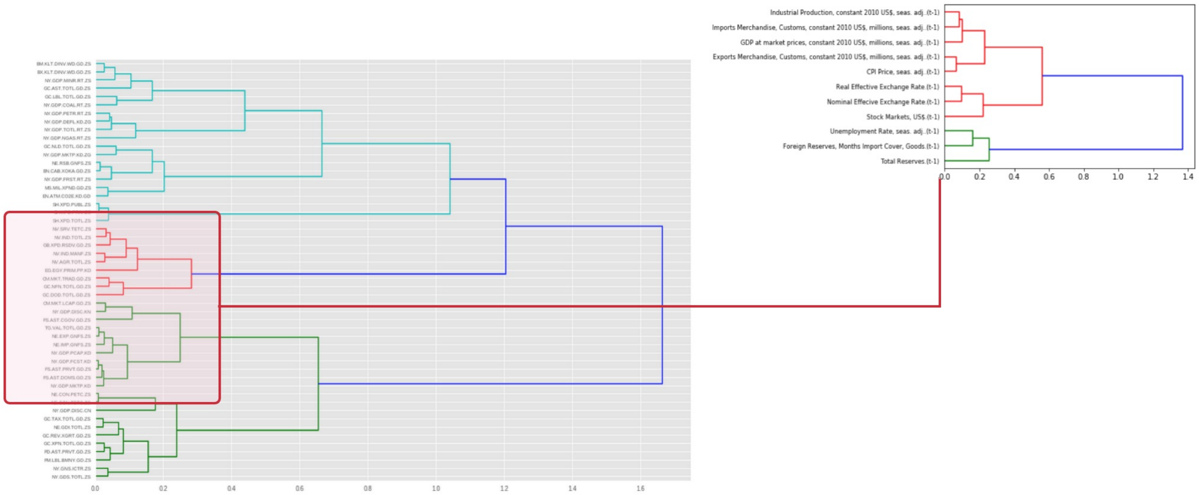


Figure 6-2. Clustering macro-economic indicators to GDP

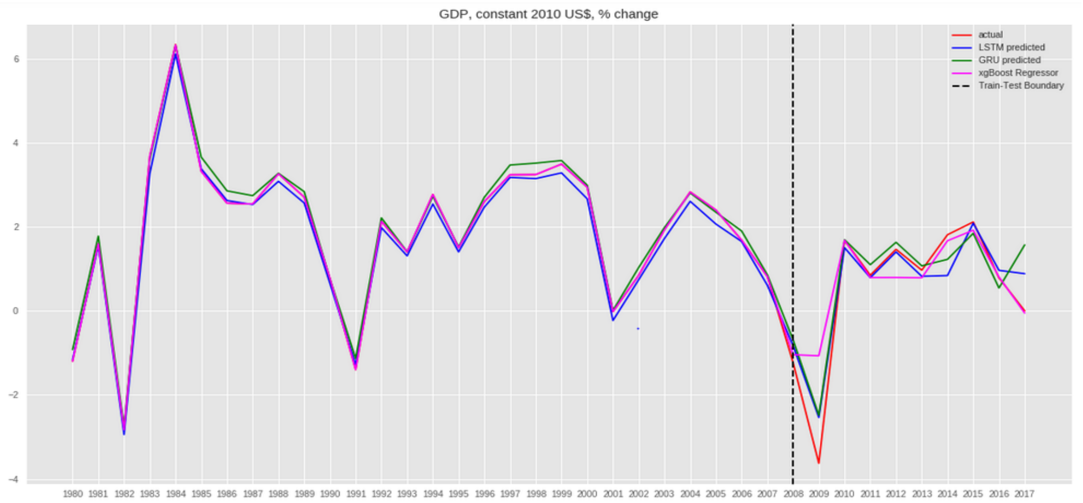


Figure 6-3. LSTM, GRU and Gradient Boosting Model Results for US GDP Growth prediction, 1980-2017

Figure (6-4) compares the ensemble model results with the industry standard IMF GDP growth forecasts. The chart shows that the results yields substantial improvement in forecasting macro-economic indicators like growth than current methods. Further, the deep ensemble architecture has been generalized so many other indicators can be predicted at scale. Figure (6-5) presents the model reliability matrix (MRM) providing a simple visual signal on assessing model performance. In the instance of GDP growth, voting classifier for LSTM, GRU, XGboost is used to ensemble overall improve the overall model accuracy.

Both the base LSTM and GRU starts converging around 300 and 150 epochs respectively with of rmse of 0.65 and 0.51 respectively. The train-test split on LSTM, GRU, XGBoost and Actual Data for USA is 80-20. The LSTM and GRU perform better in short run, but XGBoost out-performs them in long run. The predictions for 2017 are not finalized yet by countries or international institutions, so the latest actual data-point is up to 2016, and our predictions are up to 2017. For details concerning the base model parameters, readers can refer to table (6-2). Briefly, the reason for parameter choices include 128 neurons as depth for each neuron for time series with lots of feature vectors/parameter. Due to irregularity, dropout is 0.1 and activation given yearly time-series for consistency is ‘linear’. A lot of trial and error went into choosing the epochs and batchsize based on the choice of data and algorithms.

The model reliability matrix in figure (6-5) confirm that the ensemble model has smaller rmse and higher r2 and explained variance (EV) score, collectively implying closer to accurate prediction. The Uncertainty Quantification (UQ) chart is finally presented in figure (6-6) to not only obtain point estimates but range of the prediction estimates. The quantified uncertainty error is 0.4544.

This base format for US GDP growth prediction showed how to make a simple pipeline for multiple countries clustered by economic development and well-being. The clusters for GPD growth rate, both between countries, and between features within a country provided powerful unsupervised ensemble methods to abstract knowledge such as similarity to form optimal training group. The analysis shows details on model parameters that can be tuned for specific cases, such as advanced economies that display smaller deviation from GDP percent change need less tarining time i.e less number of epochs and less batch size (to be more precise). Emerging markets (EM's) and low-income countries (LIC's_ need more training time as their GDP change varies at a much higher rate and more batch_size (for generalised convergence).

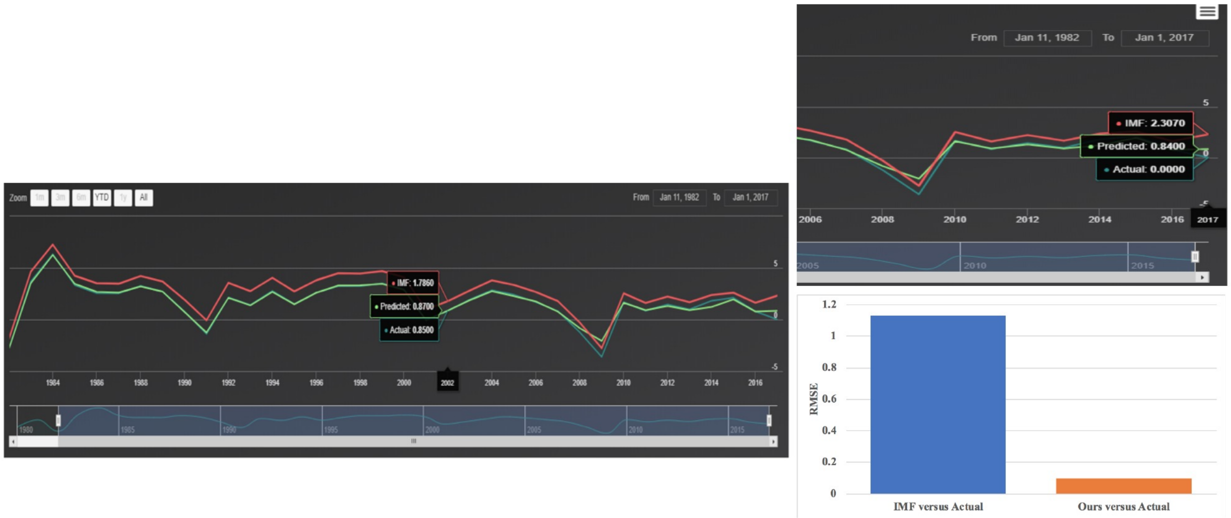


Figure 6-4. Ensemble Model based on Voting Classifier comparison with IMF GDP growth forecasts

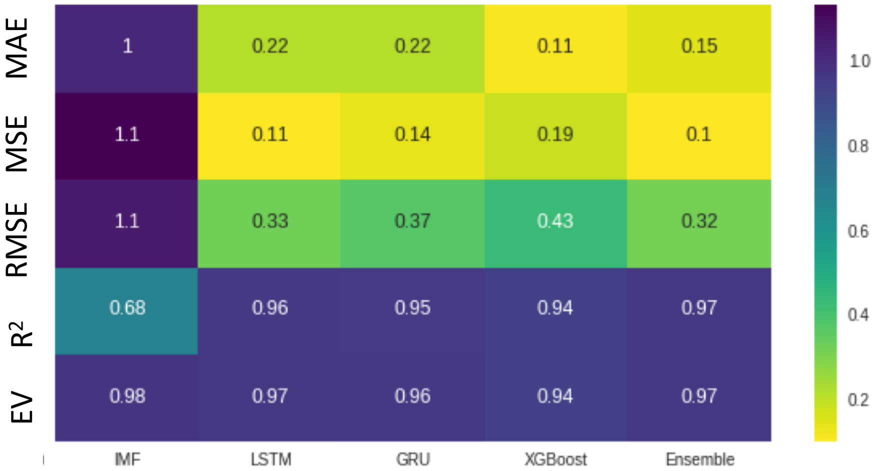


Figure 6-5. Model Reliability matrix for comparison across performance metrics

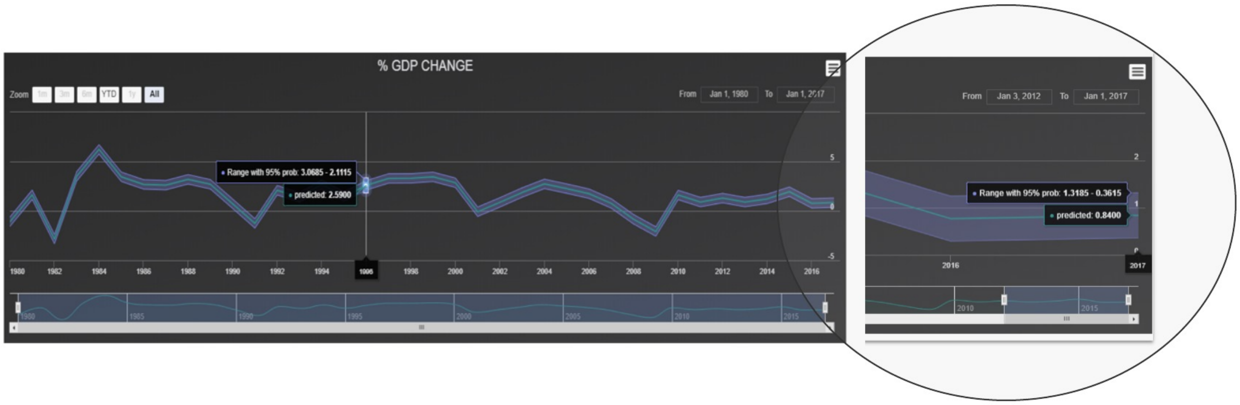


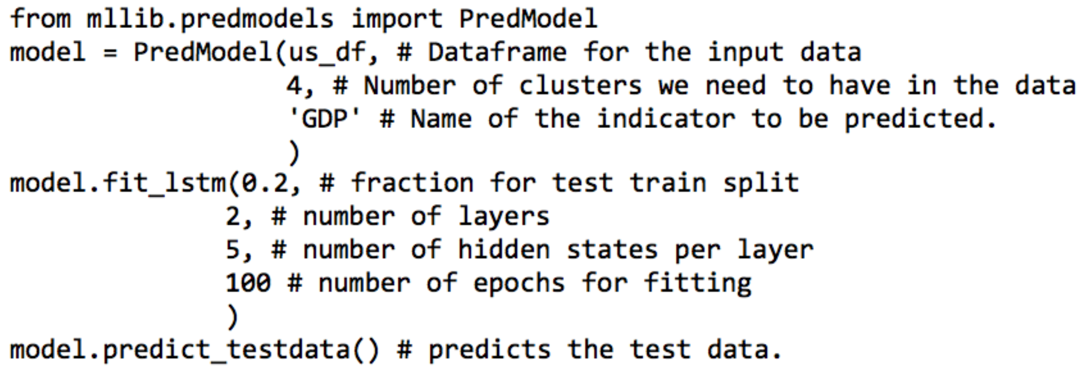
Figure 6-6. US GDP growth forecast with uncertainty quantification (UQ) using MC dropout

Table 6-2. Base Model Parameters, US GDP Growth, annual

Model	Neurons	Dropouts	Activation	Epochs	BatchSize	Metrics	Optimiser	Max-depth	Estimators
LSTM	128,4x1	0.1, 3x1	Linear	600	64	mean squared error	RMSProp		
GRU	128,4x1	0.1, 3x1	Linear	64	400	mean squared error	RMSProp		
XGBOOST (using grid search)								[**5**,10,15]	[50,100,**150**]

6.1.1. Abstraction of Prediction layer

The design of the prediction layer API is inspired from sklearn and keras API, thus being very familiar and easy to understand. An example snippet to access the prediction mechanism is given below.



```
from mllib.predmodels import PredModel
model = PredModel(us_df, # Dataframe for the input data
                  4, # Number of clusters we need to have in the data
                  'GDP' # Name of the indicator to be predicted.
                  )
model.fit_lstm(0.2, # fraction for test train split
              2, # number of layers
              5, # number of hidden states per layer
              100 # number of epochs for fitting
              )
model.predict_testdata() # predicts the test data.
```

6.1.2. Exchange Rates

Effective exchange rates (EER) are useful for gauging whether a currency has appreciated overall relative to trading partners; and, also, used to assess the relative competitiveness of countries. For example, if the local currency of a country is expected to appreciate in the coming five years from a substantial margin, this would provide a strong signal to invest in that country. In addition, EER can provide more valuable information; for example, in 2015 the Chinese RMB depreciated about 8 percent against the US dollar; however, the US dollar appreciated against other Asian and European currencies and with China's growing trade with Asia and Europe the net effect was that once weighted by trade shares the value of the Chinese currency actually appreciated approximately 10 percent relative to its trading partners (Smitka and Ruggles 2015).

Such exchange rate fluctuations can affect project costs if the funding is through global capital markets and international investors. A case in point is the Australian projects, where appreciation of the local currency against the US dollar has been a contributing factor to project cost escalations in the oil and gas sector (Ernst and Young 2014).

The Bank of International Statistics (BIS) provide effective exchange rate (EER) indices that cover 61 economies. The most recent weights are based on trade in the 2011-13 period, with 2010 as the indices' base year. Previous studies have concluded that a Vector Autoregression (VAR) which is a standard in macro-econometrics, generates the most accurate forecasts during a 1-month horizon, while the ARIMA is the more suitable model during a 3-month horizon of EER (Varenius 2017). Here, a simple test case for quarterly US nominal exchange rate is presented to showcase the versatility of the model performance across different time ranges and domains. In this light, other aspects of interest rate parity, and cross-currency basis may also provide important signals for the countries' financial and economic stability.

The EER data is integrated with various other sources of BIS data, including the BIS Central Bank Policy Rates, Credit gap, BIS long series on total credit, BIS Debt service ratio BIS Effective Exchange, Exchange rates series, Locational Banking Statistics, BIS derivatives statistics, global liquidity indicators…etc. These other features may be important for training the data-set and help in making the model scalable. Figure (6-7) shows the nominal EER trend of the USA from 1964Q1 to 2017Q4. The data is split into training and test sets, with training data size including the first 192 quarters, and the test data set is the next 24 quarters.

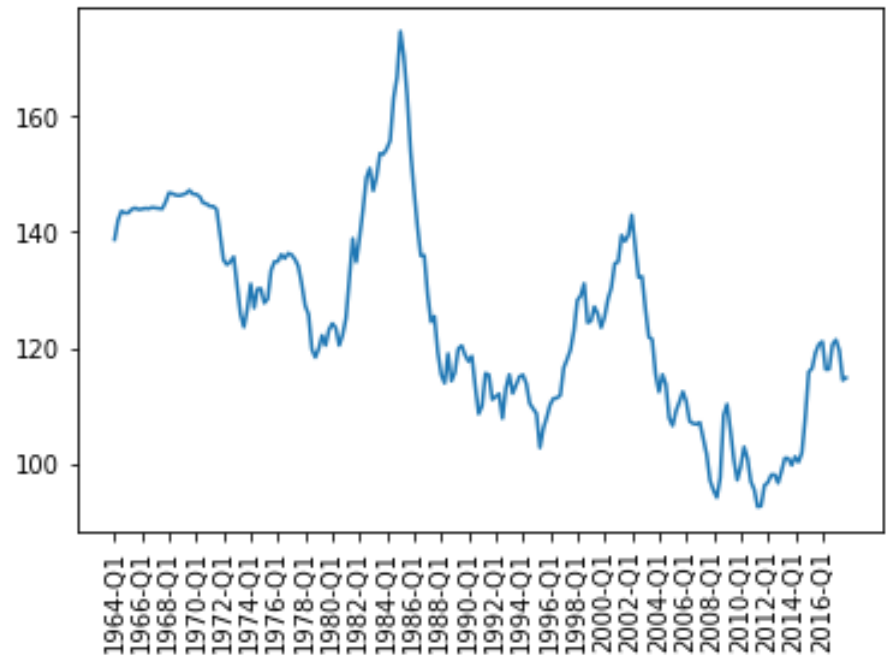
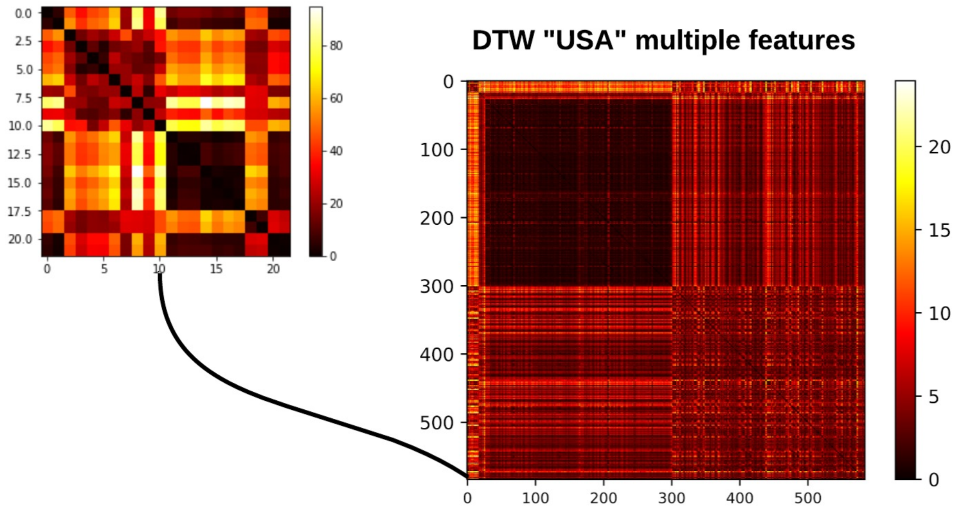
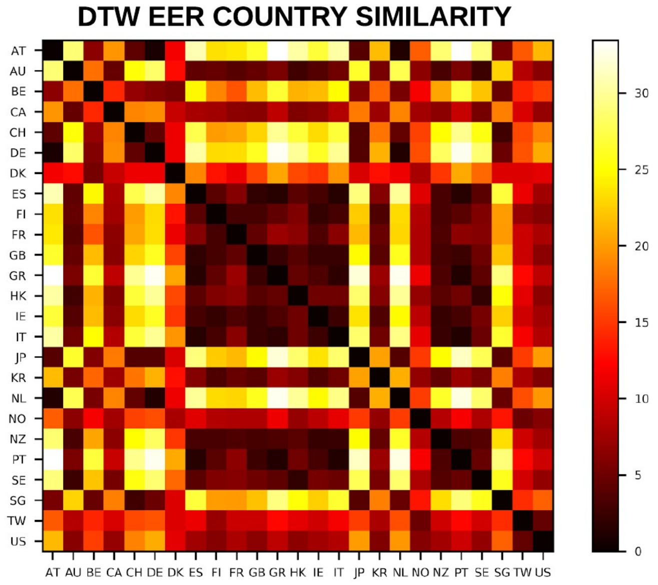


Figure 6-7. Nominal Effective Exchange Rate for the USA, based on Broad measurement from 61 countries, in quarters.
Source: Bank of International Statistics, 2018.

Figure (6-8) shows the DTW heat-maps for (a) all features available in BIS, and (b) country similarity based on EER. As an alternative to hierarchical clustering, MDS is used to cluster and group the different pair-wise distance measures. A sample graph for country similarity grouped in three clusters is provided in Figure (6-9).





AUT	Austria
AUS	Australia
BEL	Belgium
CAN	Canada
CHE	Switzerland
DEU	Germany
DNK	Denmark
ESP	Spain
FIN	Finland
FRA	France
GBR	United Kingdom
GRC	Greece
HKG	Hong Kong
IRL	Ireland
ITA	Italy
JPN	Japan
KOR	Korea, Republic of
NLD	Netherlands
NOR	Norway
NZL	New Zealand
PRT	Portugal
SWE	Sweden
SGP	Singapore
TWN	Taiwan, Province of China
USA	United States

Figure 6-8. Dynamic Time Warp (DTW) for all country features from USA based on BIS data, and DTW for country similarity based on EER

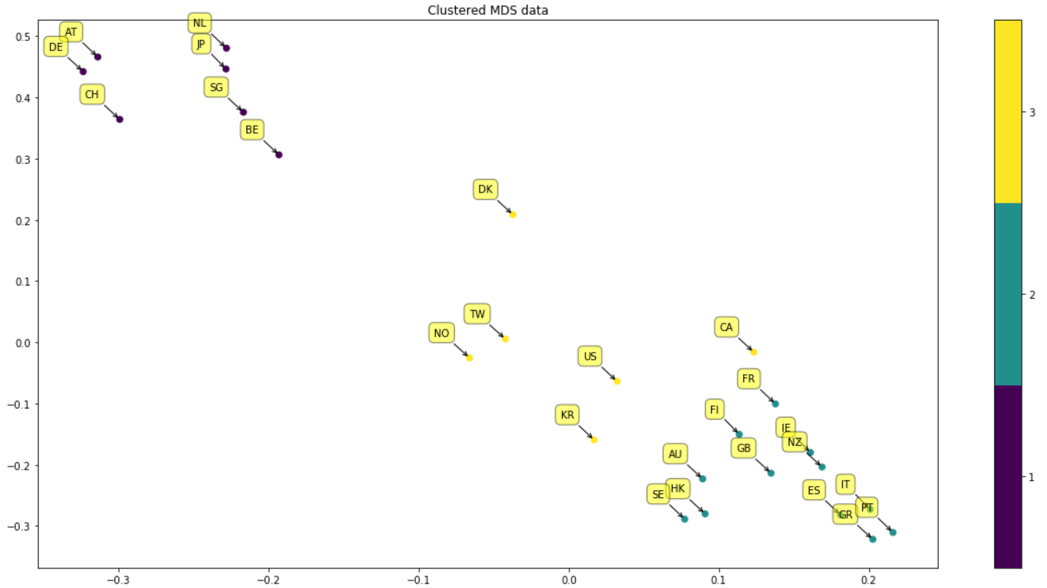


Figure 6-9. Three Clusters of countries based on Multi-dimensional Scaling (MDS) for Effective Exchange Rate

The data slice used for the experiment is the USD/EUR daily exchange rate from BIS database, using Quandl API. The start date used is 2010-01-01 and the split date used is 2014-01-01. The test set for the experiment span till the latest available entry in the dataset. Three models are used for this experiment including LSTM, deep LSTM, and bagging model for ensembling. The following pages contains a summary of the models.

In the figure (6-10) below, the black line represents the actual value of the US dollar and EUR daily nominal exchange rate. Classical LSTM model with a single layer of 128 LSTM cells no dropouts or recurrent dropouts. The model is trained for 10 epochs using various regression metrics. The model overfits due to lack of training features and lack of dropouts.

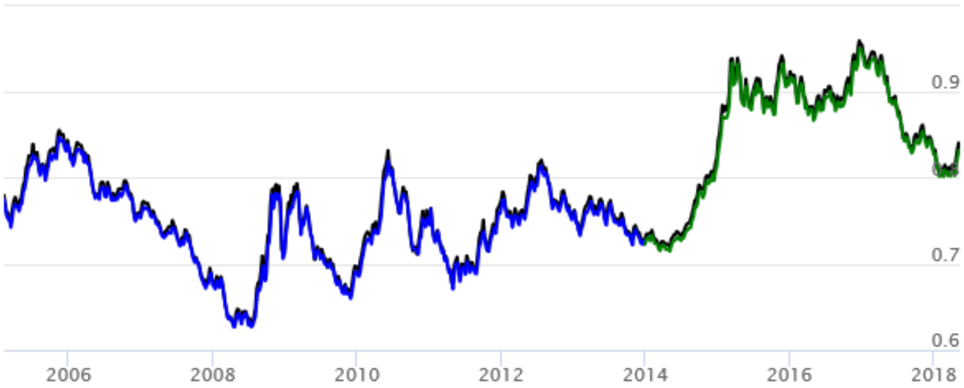


Figure 6-10. Vanilla LSTM model - USD/EUR daily exchange rate movement, black line is the actual, blue is the training set, and green is the testing set.

Second, a deep LSTM model is deployed and the result is shown figure (6-11). This is a deeper LSTM in a manner of speaking, because this model has more hidden layers in proportion with 6 hidden layers of 128 neurons each. This model was trained on various regression metrics for 20 epochs. This is slightly better than the vanilla LSTM model due to dropouts and recurrent dropout. This fares better in terms of the metrics but still overfits.

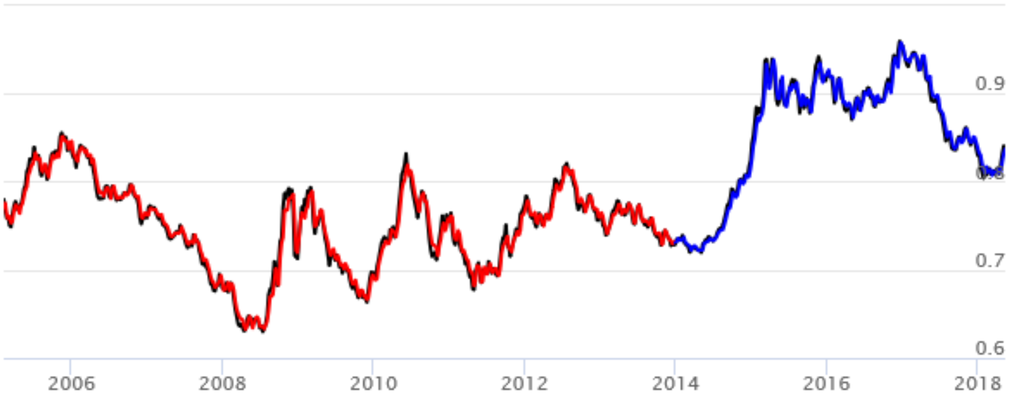


Figure 6-11. Deep LSTM model - USD/EUR daily exchange rate movement, black line is the actual, blue is the training set, and green is the testing set.

Finally, an ensemble model using bagging is presented in figure (6-12). This is an Ensemble of 7 LSTMs with three hidden layers each. This model is trained on various splits of train data. Each LSTM is trained on each split of train data and the results from each model are combined using equal weight average method (the result of each model is averaged to acquire the result). The model is trained on various regression metrics for 10 epochs on each model. These metrics are better than the vanilla LSTM model, easier to train but take a long time to train and predict on the data. Even still this model is not as complex as deep LSTM, some amount of overfitting is avoided in this model.

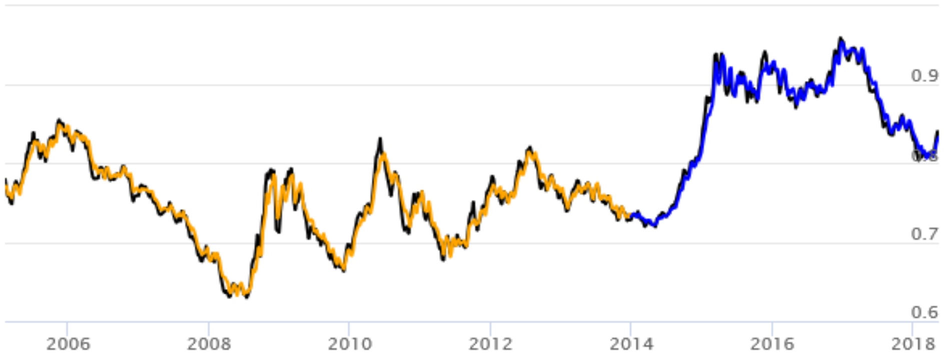


Figure 6-12. Ensemble Deep LSTM model - USD/EUR daily exchange rate movement, black line is the actual, blue is the training set, and green is the testing set.

Finally, the result for the ensemble UQ model is presented in figure (6-13). The UQ method used for this experiment is the Monte Carlo Dropout (MC dropout). Currently the implementation is only able to quantify the model misspecification and the inherent noise. This model contains a confidence interval function for estimating the prediction bounds. The blue region is for the train data prediction of the bagging model and the grey region is for the test data prediction of the bagging model on a confidence value of 90% the graph is as above and the range of the values on average is 0.6.

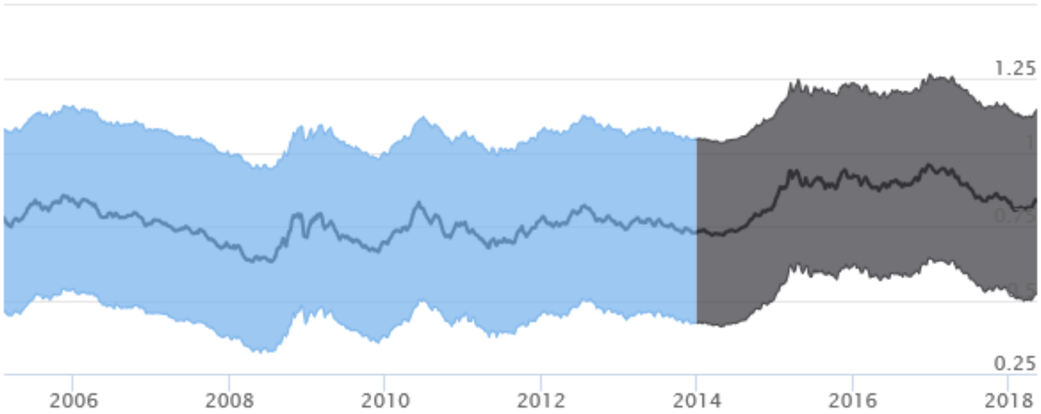
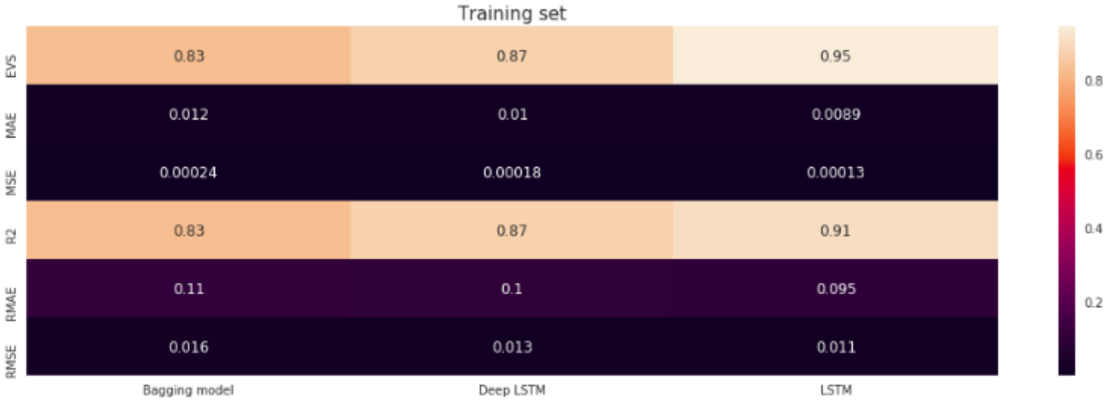


Figure 6-13. Ensemble Deep LSTM model with MC dropout UQ - USD/EUR daily exchange rate movement, black line is the actual, blue is the training set, and green is the testing set.

In order to assess model reliability, different performance metrics are used and presented in figure (6-14). R^2 score for the train data is smaller in the case of the ensemble model whereas the deep LSTM in terms of testing data performs better than the ensemble model. Ensembling the model yields marginally better result. The variation in the metrics is least in the ensemble model with the results presented so far. The model parameters details are presented in table (6-3).



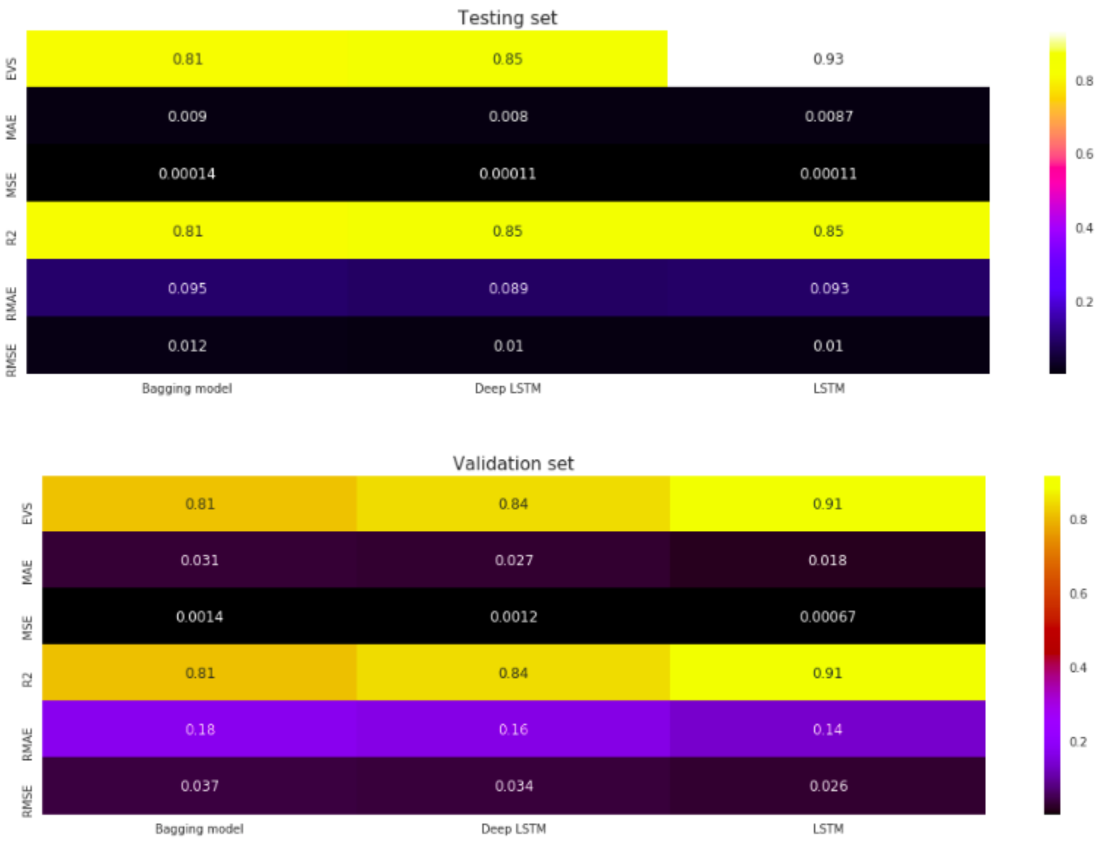
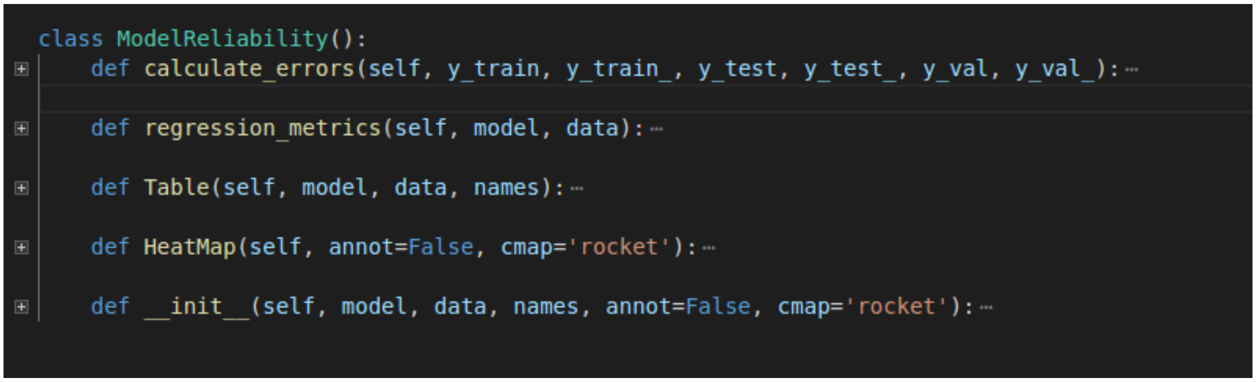


Figure 6-14. Model Reliability Matrix Visualizations for USD/EUR exchange rates

Table 6-3. Models parameters for different USD/EUR exchange rates

Model	**Input shape**	**LSTM cells**	**Dropouts**	**Recurrent Dropouts**	**Epochs**	**BatchSize**	**Metrics**	**Optimiser**	**# rounds of random samples**
Vanilla LSTM	(28, 1)	[128]	[0]	[0]	10	100	mean squared error	RMSProp	
Deep LSTM	(28, 1)	[128, 128, 128, 128, 128]	[0, 0.2, 0.3, 0.4, 0.5]	[0.3, 0.3, 0.3, 0.3, 0.3]	20	100	mean squared error	RMSProp	
Ensemble LSTM (bagging) 7	(28, 1)	[128, 128, 128]	[0, 0.2, 0.5]	[0.3, 0.3, 0.3]	10	100	mean squared error	RMSProp	
MC drop UQ									**100**

Throughout this research, the effort has been made to build standards of data, models, and visualization for the time series data by building an abstraction for the various source code. From an application standpoint, the Abstraction internally converts the data into json formats for easy manipulation by the JavaScript engine. Similarly, there are abstractions, for example calculating the model metrics for a regression problem and the functions are as follows



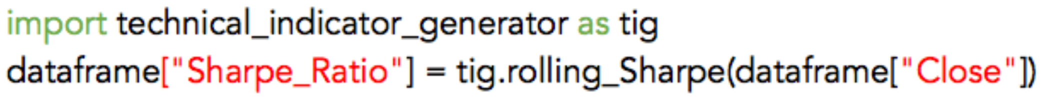
```python
class ModelReliability():
    def calculate_errors(self, y_train, y_train_, y_test, y_test_, y_val, y_val_):⋯

    def regression_metrics(self, model, data):⋯

    def Table(self, model, data, names):⋯

    def HeatMap(self, annot=False, cmap='rocket'):⋯

    def __init__(self, model, data, names, annot=False, cmap='rocket'):⋯
```

6.2. Results for Finance Domain

In this section, the prediction of stock market ticker are formulating using a backtesting mechanism. Here forecasting models are created using python and deployed using Django. Many different methods have been used to increase the efficiency of the forecast using ensemble learning. There is also emphasis on representing the predictions and prediction related indicators in more suitable visualization which uses Mapbox and other libraries available in python. Custom reusable libraries are also made that can be used to derive data that is not available but are essential in predictions.

Pre-processing Domain Knowledge – Financial Technical Indicator Extraction

Initial stock data were extracted using Quandl API. These contained Open, Low, High, Close, and Volume of each Ticker. These data were furthermore used to extract following technical indicators such as Rolling Sharpe Ratio, Relative Strength Index(RSI), Accumulation/Distribution ROC, Accumulation/Distribution Index, MACD, MACD Signal and MACD difference, Average directional movement index (Pos,Neg), Vortex Indicator (Pos,Neg). Selectively technical indicators can be chosen to train the model by using custom library, an example of technical_indicator_generator.py



```
import technical_indicator_generator as tig
dataframe["Sharpe_Ratio"] = tig.rolling_Sharpe(dataframe["Close"])
```

Dynamic Time Warping based Hierarchical Clustering

The cluster extraction module extracts the relevant features. The cluster extraction techniques used are the following. First, Dynamic Time Warp (DTW) is used to measure the distance between different asset prices. In this particular case, the focus will be on closing day stock prices form the S&P 500. Second, clustering techniques are used such as K-means and Hierarchical clustering. The DTW finds an optimal match between two sequences of feature vectors which allows for stretched and compressed sections of the sequence. The research paper on dynamic programming algorithm optimization for spoken word recognition can be very useful (Sakoe and Chiba 1978)

The DTW algorithm was used to find distance between stocks. Initially tests were conducted based on Normalized Closing Prices, but Sharpe Ratio are also used. This gives us a distance matrix of stock tickers. By using this distance matrix in Hierarchical Clustering clusters were created for all stocks in Figure (6-15). The chart of hierarchical clustered DTW heat map for a specific group of companies is zoomed in. In the sub-sample heat-map below, there are 63 companies from SP500 clustered with significant number of trading period under 5 clusters. The clustering here is based on the time trend of Sharpe Ratio over 500 days.

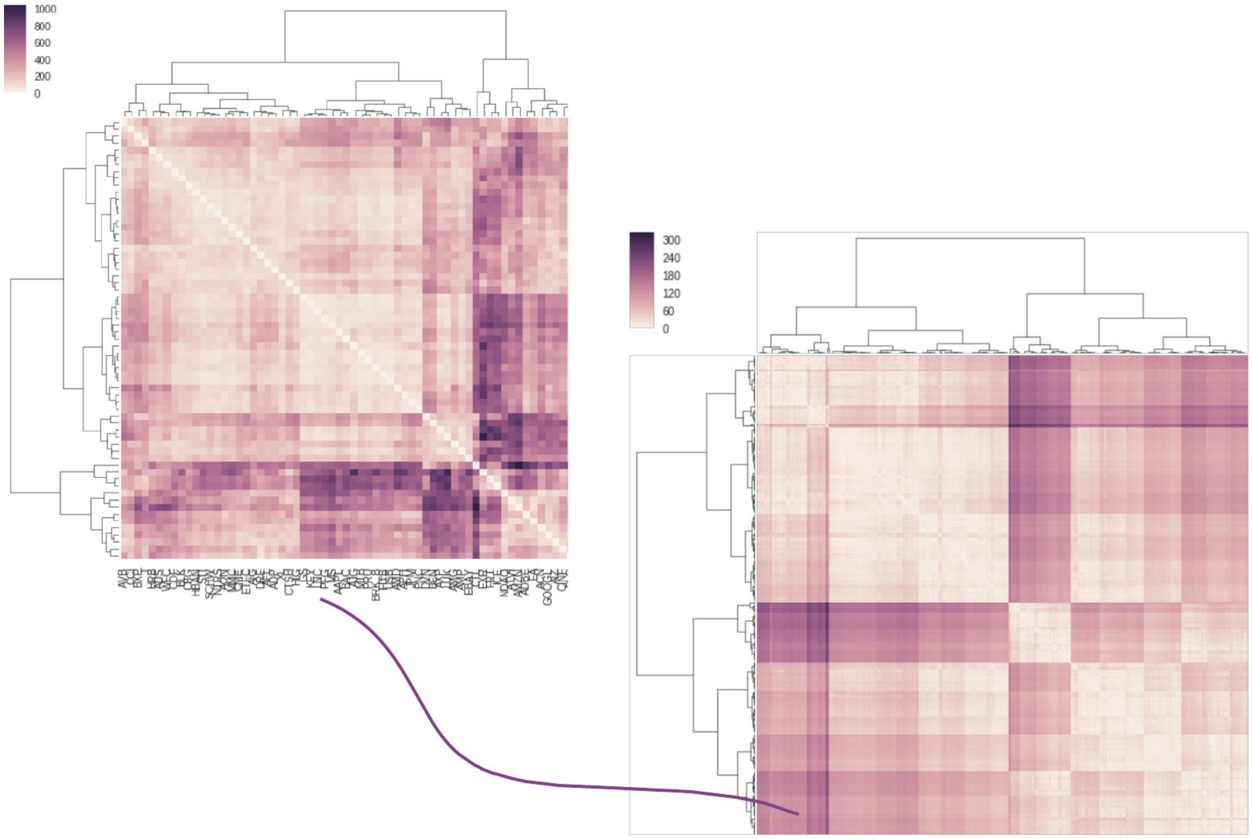
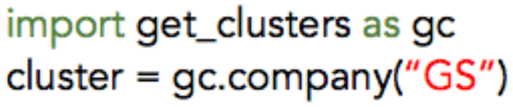


Figure 6-15. DTW distance after applying Hierarchical Clustering.

In this view, of dependencies of a given measure to other domains will be highly valuable. For example recent studies showed that there was "high correlation between the value of the stock market and the unemployment rate in U.S. data since 1929" (Farmer 2012). The crash in the stock market has been labelled as one key reason behind the great financial crisis of 2008. The results presented on prediction system can prove valuable monitoring capabilities for decision-makers, even in other domains and assessing project prospects. For example, if a construction company in a developing country is the contractor to a mega-project. This company is a publicly listed company, and the low-error predictions indicate that the price of the company will crash. This would be an important indication for risks associated with a given project.

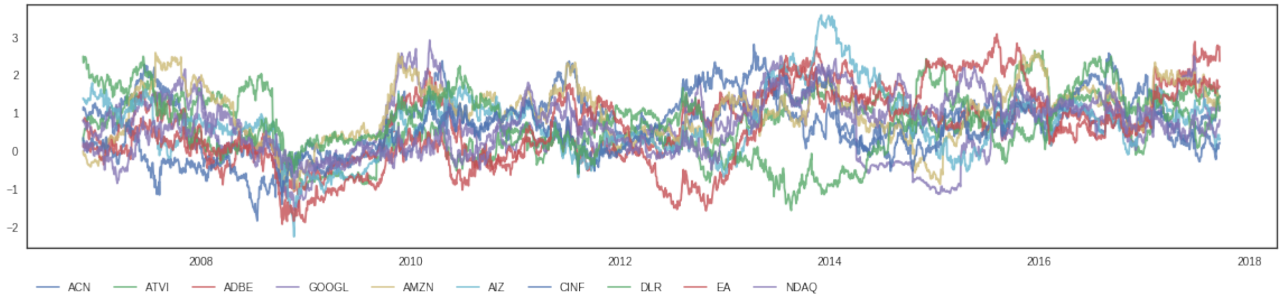
After DTW, the clusters are saved to extract them when required. Because it can be very time consuming sometimes to calculate DTW distance between the stocks sometimes, especially on a daily basis. These clusters can be accessed by using the following custom function.



```
import get_clusters as gc
cluster = gc.company("GS")
```

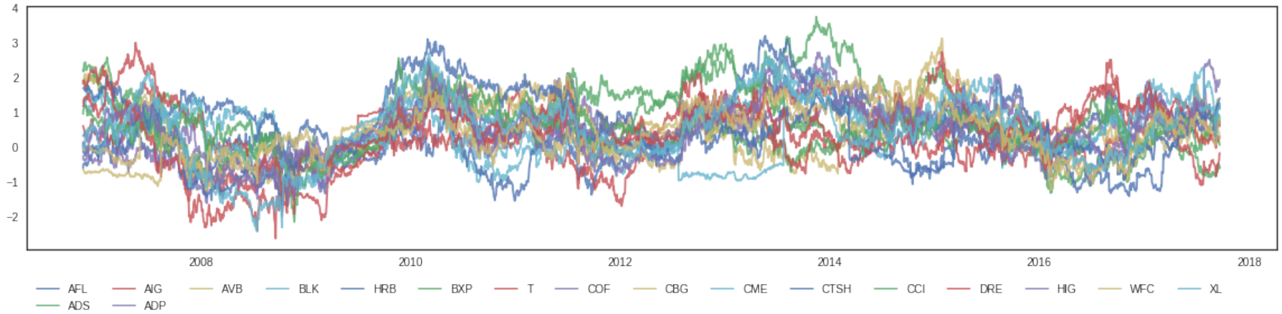
Specific clusters are visualized in Figure (6-16). This shows Cluster 2, 4, and 5, where it is evident that companies in similar sectors are in the same cluster. For example, financial companies like Goldman Sachs (GS), Berkshire Hathway (BRK_B), Charles Schwab (SCHW), JP Morgan (JPM), USB (USB) are all in Cluster 5. The whole network can be visualized using hierarchical clustering to map the similarity of companies in Figure (6-16). The chart is rather extensive with 500 companies but provides the reader a perspective of different computations. Similarly, other models for similarity of technical features are also conducted for stock tickers. For composite market indices such as NASDAQ, Dow Jones, London Stock Exchange, or Bombay Stock Exchange a similar set of macroeconomic features are also extracted. These training features are particularly valuable, however the results delved into here will be for composite S&P500 index. For composite indices, the group of similar countries may not be as meaningful as similar other composite indices. In this instance, the major composite indices are included.

Cluster 2:



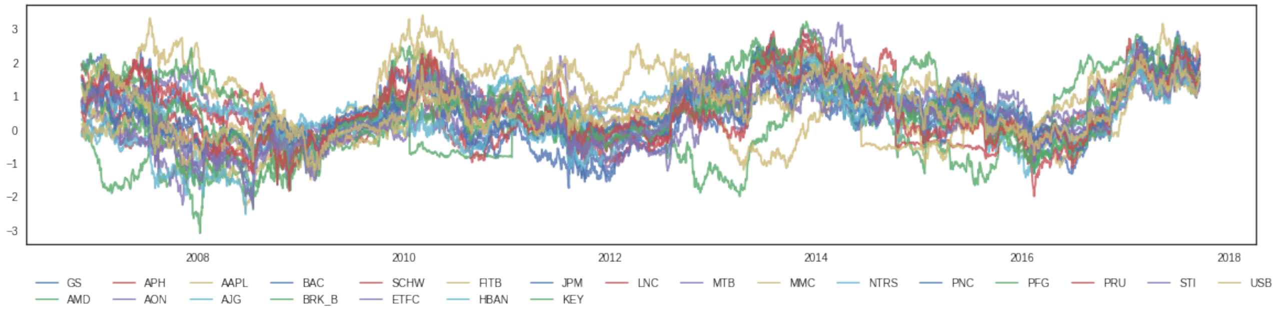
['ACN', 'ATVI', 'ADBE', 'GOOGL', 'AMZN', 'AIZ', 'CINF', 'DLR', 'EA', 'NDAQ']

Cluster 4:



['AFL', 'ADS', 'AIG', 'ADP', 'AVB', 'BLK', 'HRB', 'BXP', 'T', 'COF', 'CBG', 'CME', 'CTSH', 'CCI', 'DRE', 'HIG', 'WFC', 'XL']

Cluster 5:



['GS', 'AMD', 'APH', 'AON', 'AAPL', 'AJG', 'BAC', 'BRK_B', 'SCHW', 'ETFC', 'FITB', 'HBAN', 'JPM', 'KEY', 'LNC', 'MTB', 'MMC', 'NTRS', 'PNC', 'PFG', 'PRU', 'STI', 'USB']

Figure 6-16. Cluster 2, 4, and 5 for selected companies, stock tickers are identified Insightful information can be processed by observing the similarity between companies belonging to same clusters.

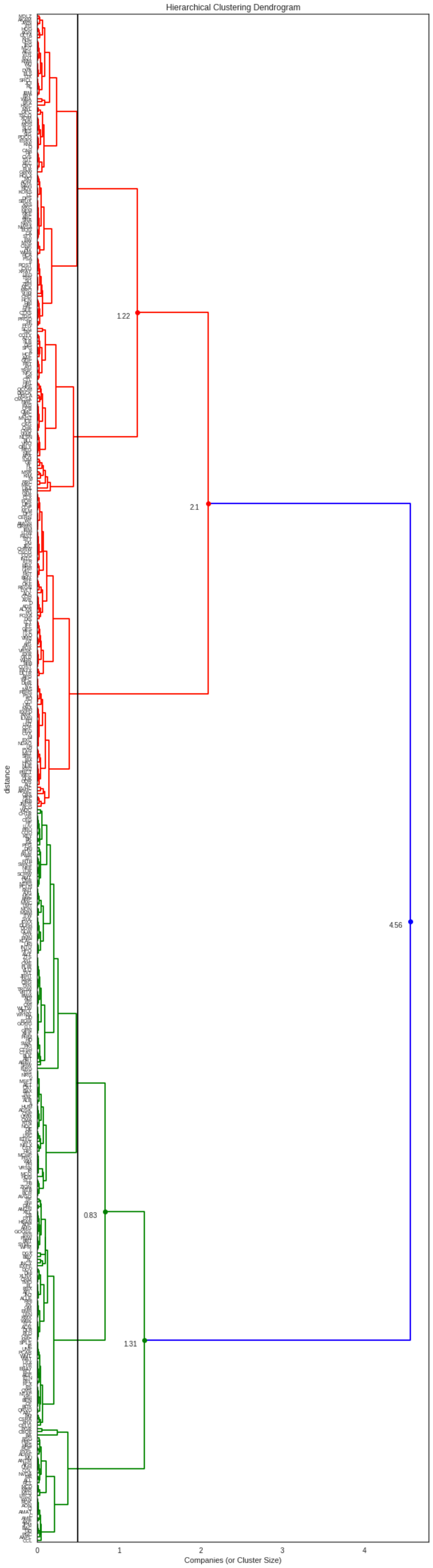


Figure 6-17. Hierarchical Clustering of S&P 500 Stock Tickers

Next the results for ensembling models in the finance domain are presented. Several categories of deep learning models with different complexity have been deployed. In this instance, the full list of models presented in table (6-4). First, the results for a specific ticker, as Goldman Sachs from the S&P 500 is presented, and second, the composite S&P500 daily index price results are presented.

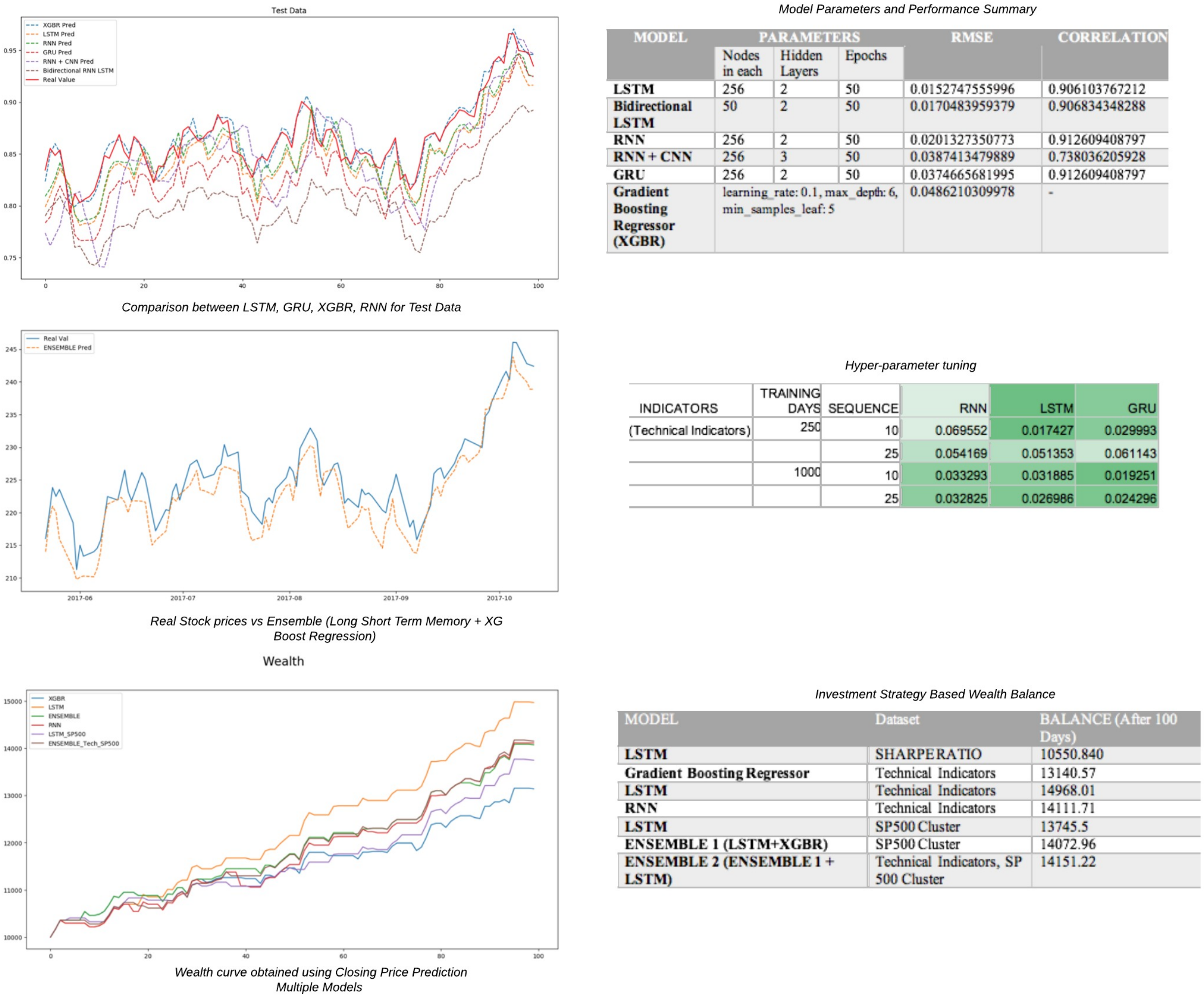
Table 6-4. Different deep learning models used for stock price prediction

Model	Description
LSTM	Long-Short Term Memory
Bi directional LSTM	Bi-directional LSTM
RNN	Recurrent Neural Networks
GRU	Gated Recurrent Unit
GRADIENT BOOSING REGRESSOR	Gradient boosting regressors are a type of inductively generated tree ensemble model. At each step, a new tree is trained against the negative gradient of the loss function, which is analogous to (or identical to, in the case of least-squares error) the residual error.
ENSEMBLE	Ensemble learning is the process by which multiple models, such as classifiers or experts, are strategically generated and combined to solve a particular computational intelligence problem.

The summary results for Goldman Sachs are presented in figure (6-18). The first graph shows comparison of different deep learning architecture model resluts. The dataset is based on technical indicators, the sequence length is 25 days, and the training days is 1000, and testing days is 100. The second graph shows the stock price of GS (Goldman Sachs) being predicted over 100 testing days. As it can be seen the performance of XGBOOSTING REGRESSOR and LSTM and RNN are comparable and are good approximators. Even other models have been able to retain the trend but with a lag. XGBR and LSTM/RNN for making our ensemble model seem the right ensemble choices. The automated hyper-parameter tuning programs are utilized in this instance (Panel B table). This step is essential to find a trade-off between time and increasing the accuracy of the model. Since, most of the algorithms are using more than 1 Hidden layer and have larger number of indicators, it takes a very long time to train a model. So it is better to first get the perfect tuned parameters and then deploy similar models with same parameters. It can be inferred that a training set of ~1000 Days and a sequence length of 25 days will give the most optimum solution for all the models.

Here, there are two proposed ensemble methods: (a) between subdomains of technical/stock indicators, and (b) between models based on technical indicators and S&P 500 indicators. LSTM models can have the overfitting problem with such time-series information. A weightage scheme for example - .33 .33 .33 for assembling different models is chosen as a starting points. The literature giving higher weights to the poorer models helps avoid overfitting. A dropout of 0.20 i.e. 20 percent of neurons in each layers are inactive. Here, ensemble models between LSTM and XGBR are used to train using technical indicators. The results show that ensemble deep learning helps in overcoming overfitting issues. In this instance, the weight distribution between XGBR and LSTM is 0.8 : 0.2. One rule of thumb applied is to give more weight to the weaker model. But with fine tuning the weights a more robust ensemble model can be achieved.

The ensemble model performance and the comparison between different models based on the wealth generated are also present. The agent starts with $10,000. The lines represent how much money the agent would make based on following the strategy provided by different models. The model outperforms the annual returns of the S&P 500 which is approximately 67 percent for this time-period. The balance of wealth after 100 days for different models and datasets are succinctly presented in the last table.



Comparison between LSTM, GRU, XGBR, RNN for Test Data

Model Parameters and Performance Summary

MODEL	PARAMETERS			RMSE	CORRELATION
	Nodes in each	Hidden Layers	Epochs		
LSTM	256	2	50	0.0152747555996	0.906103767212
Bidirectional LSTM	50	2	50	0.0170483959379	0.906834348288
RNN	256	2	50	0.0201327350773	0.912609408797
RNN + CNN	256	3	50	0.0387413479889	0.738036205928
GRU	256	2	50	0.0374665681995	0.912609408797
Gradient Boosting Regressor (XGBR)	learning_rate: 0.1, max_depth: 6, min_samples_leaf: 5			0.0486210309978	-

Real Stock prices vs Ensemble (Long Short Term Memory + XG Boost Regression)

Hyper-parameter tuning

INDICATORS	TRAINING DAYS	SEQUENCE	RNN	LSTM	GRU
(Technical Indicators)	250	10	0.069552	0.017427	0.029993
		25	0.054169	0.051353	0.061143
	1000	10	0.033293	0.031885	0.019251
		25	0.032825	0.026986	0.024296

Wealth curve obtained using Closing Price Prediction Multiple Models

Investment Strategy Based Wealth Balance

MODEL	Dataset	BALANCE (After 100 Days)
LSTM	SHARPERATIO	10550.840
Gradient Boosting Regressor	Technical Indicators	13140.57
LSTM	Technical Indicators	14968.01
RNN	Technical Indicators	14111.71
LSTM	SP500 Cluster	13745.5
ENSEMBLE 1 (LSTM+XGBR)	SP500 Cluster	14072.96
ENSEMBLE 2 (ENSEMBLE 1 + LSTM)	Technical Indicators, SP 500 Cluster	14151.22

Figure 6-18. Goldman Sachs Prediction Summary.

The results for S&P 500 stock price index using the different deep neural network models are presented next in Figure (6-19). The charts compare RNNs, LSTMs, and variants such as GRU here. Many other models were also tested but are not shown since the accuracy is substantially higher for these deep learning models. The results show that both RNN, and

LSTM's have lower error as shown in the model reliability matrix in figure (6-20). This also raises the prospects of how information from these different models can be processed in a final prediction that provides the best fit. Finally, the MC dropout based UQ for ensemble model for S&P500 are presented in figure (6-21).

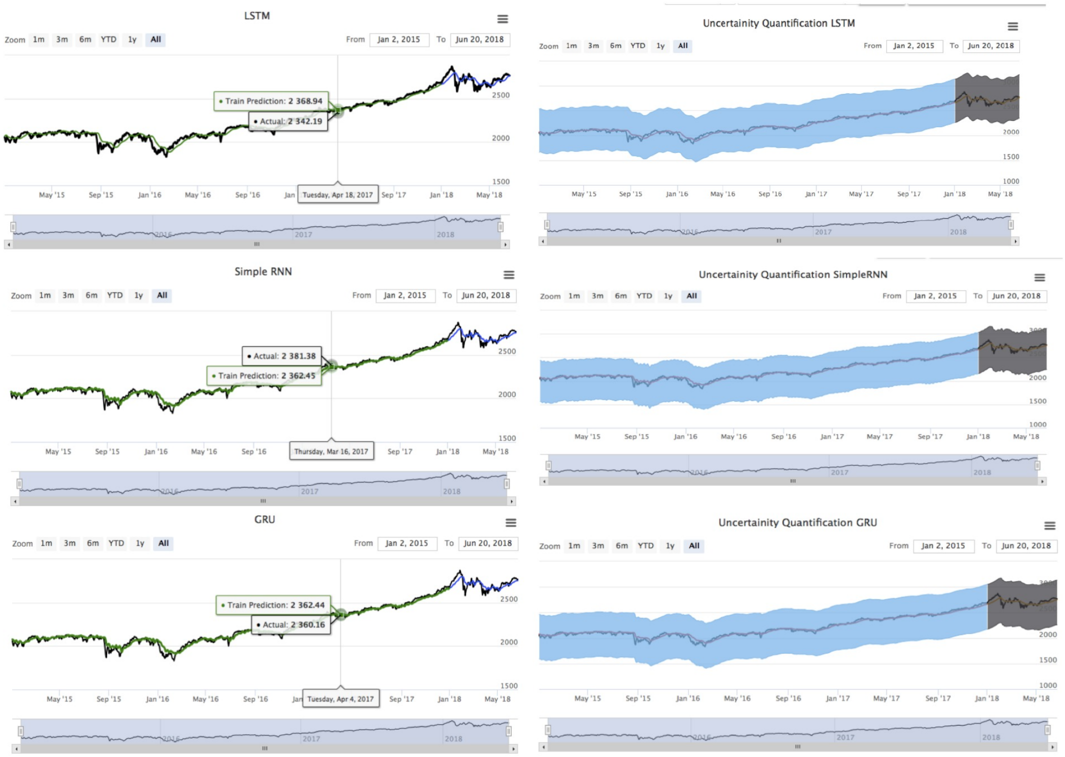
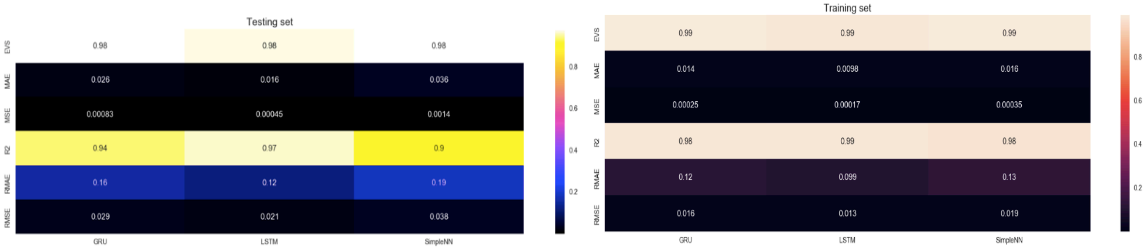


Figure 6-19. Different Deep Learning Models for S&P500 Index Price Prediction



Testing set

	GRU	LSTM	SimpleNN
EVS	0.98	0.98	0.98
MAE	0.026	0.016	0.036
MSE	0.00083	0.00045	0.0014
R2	0.94	0.97	0.9
RMAE	0.16	0.12	0.19
RMSE	0.029	0.021	0.038

Training set

	GRU	LSTM	SimpleNN
EVS	0.99	0.99	0.99
MAE	0.014	0.0098	0.016
MSE	0.00025	0.00017	0.00035
R2	0.98	0.99	0.98
RMAE	0.12	0.099	0.13
RMSE	0.016	0.013	0.019

Figure 6-20. Model Reliability matrix for S&P500 stock prices

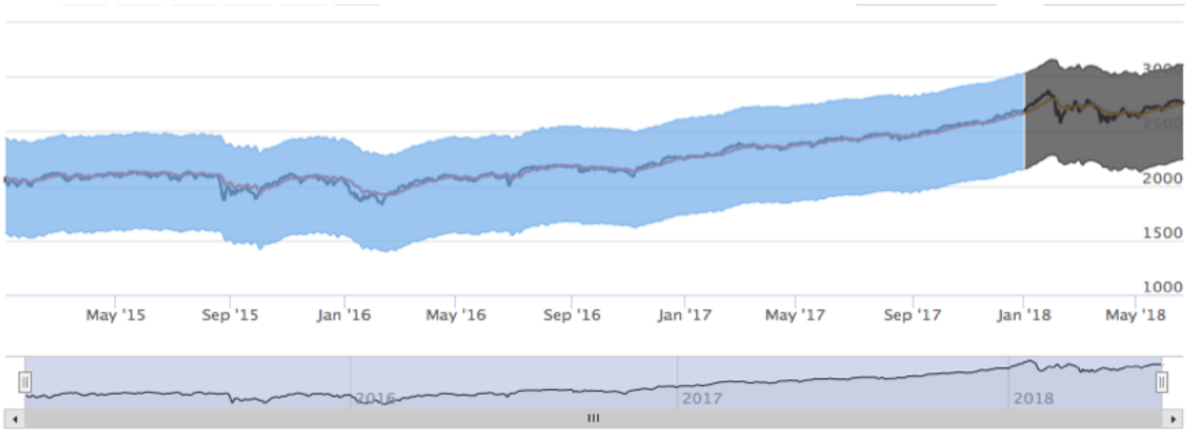


Figure 6-21. MC dropout based uncertainty quantification for ensemble LSTM model, S&P 500 composite index, daily, closing price

The ensemble model is quite scalable and shows promising results for international equity prices as well as composite stock indices. The results for specific tickers from S&P 500 such as Goldman Sachs, as well as, composite index S&P500 show promising results. Furthermore, model-based wealth generation back-tested results are provided in Appendix K for international stock indices are presented in figure. The back-tested ensemble deep learning models are used with a simple logic function. If the stock price is predicted to rise by a sharpe-ratio of greater than 2 for the next day, the decision is buy the stock; however, when the sharpe ratio is predicted to go below 2 for the next day, the decision is to sell the stock; otherwise, the agent holds the stock. Readers can find results for diverse companies such as Advanced Micro Devices Inc. (AMD). AMD has been considered not as a good stock during this time period. The returns vary but in long run it has been able to generate more profit compared to normal buy-hold or buy-sell strategy. Similarly, the results for Amazon (AMZN) stock prices are also shown. The results show considerable amount of profits. Moreover, other composite indices like the Bombay Stock Exchange and Nikkei are also presented in Appendix K.

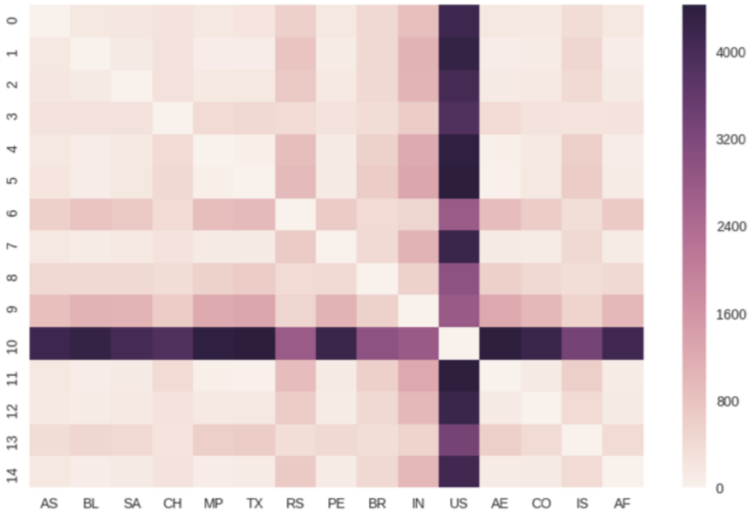
6.3. Results for Geopolitical Domain

Consider the GDELT geopolitical information available in time-series format and ready for analysis. The forecasting models are created using the machine learning and deep learning algorithms used earlier. There are different approaches to increase the efficiency of forecast by ensembling different models. Similarly, visualization of events based on cameo codes is done on map-based visualization which can help monitor real-time events over the globe based on geolocation data provided by GDELT (results now shown).

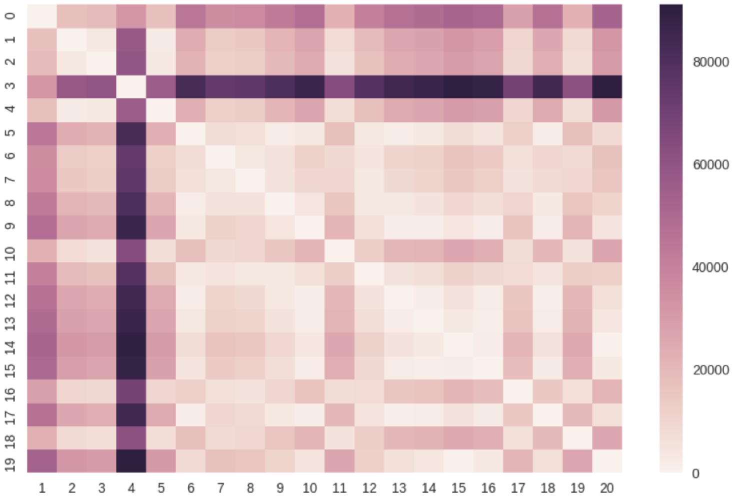
Here a specific instance of predicting protests in the United States is presented. The reader should note that the dependent variable is NM, which is considered a proxy for actual protests, and a direct measure of how much public perception and media will mention protests and matters related to protests. There are broadly three groups of training data that is assembled using machine learning techniques to make this prediction. First, there may be a relationship between various other events in the country; thus, comparing the dependent variable in context of all other events in the country is necessary. Second, geopolitical networks are crucial to global production networks with diverse processes that range from labor struggles to inter-state competition and even war have a direct impact (Glassman 2011). For example, protests in one country may be linked to other countries through information networks such as the case of the Arab Spring (Dupont and Passy 2011). Similarly, geographical proximity may feed similar cultures while it is possible to be different, there are indeed countries more similar to each other and others that are not (Walton 2007). Hence, accounting for similar protest trends in other countries could also be valuable. Third, it is important to incorporate domain features, and statistical features related to the key dependent variable.

The DTW algorithm is used to find an optimal match between two sequences of feature vectors which allows for stretched and compressed sections of the sequence. Figure (6-22) shows the DTW heat-maps for NM feature based on the distance matrix. The distance based on the DTW matrix is used to show the similarity between time-series trend in protests in USA with all other countries in Panel A, could be done for any other particular event, and between all events in the USA in Panel B, could be done for any other particular country. This matrix can be used as input for clustering similar countries and similar events. Panel A shows that the trend of protests in USA is quite distinct as represented by a high DTW with all the pairs, and panel B shows that the trend of protests in the USA is also similar to other geopolitical movements in the 500-day sample.

Afterwards, the DTW distance measure can be used to cluster these different events. For example, for clustering countries and events, both k-means and hierarchical clustering approaches were tried; and the hierarchical clustering gave more significant results. The results clustered all 20 cameo codes into 5 clusters. Based on the DTW matrix from NM feature of all cameo codes, it is found that cameo codes 1 to 16 in a cluster where 14 is also present are in the same cluster. In a similar fashion as the last charts, the results for DTW based clustering are presented in Figure (6-23) Panel A and Panel B.



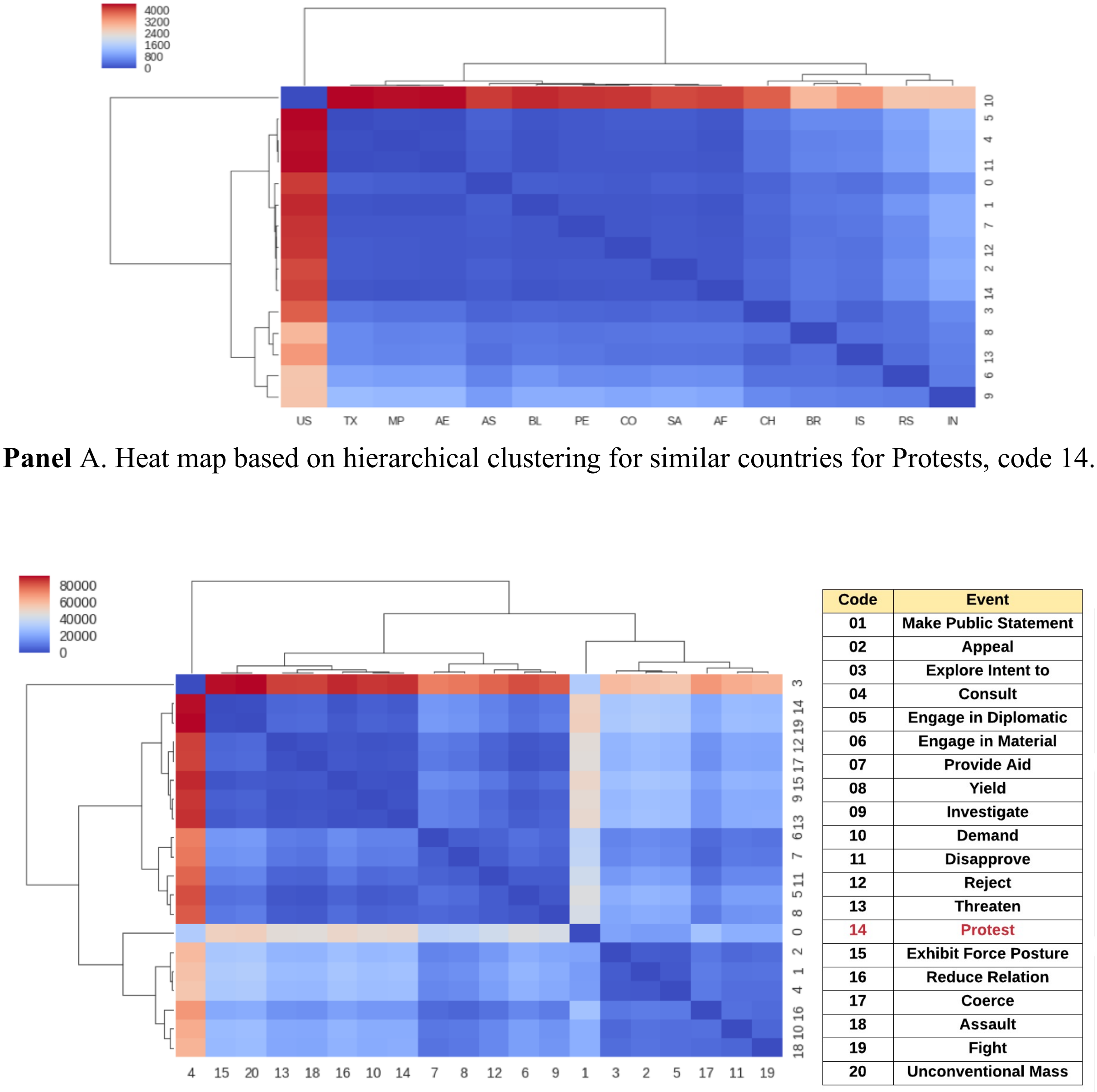
Panel B. Heat map for similar countries for "Protests", code '14', only 15 countries are taken for clear visualization.



Code	Event
01	Make Public Statement
02	Appeal
03	Explore Intent to
04	Consult
05	Engage in Diplomatic
06	Engage in Material
07	Provide Aid
08	Yield
09	Investigate
10	Demand
11	Disapprove
12	Reject
13	Threaten
14	Protest
15	Exhibit Force Posture
16	Reduce Relation
17	Coerce
18	Assault
19	Fight
20	Unconventional Mass

Panel B. Heat map for similar events in the USA

Figure 6-22. Dynamic Time Warp Similarity between for Protests in the USA, 500-daily sample



Panel A. Heat map based on hierarchical clustering for similar countries for Protests, code 14.

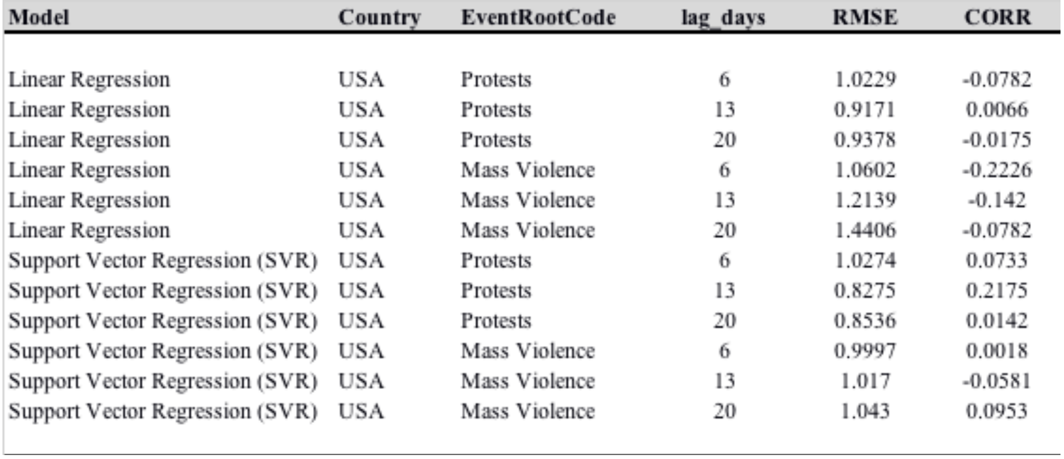
Code	Event
01	Make Public Statement
02	Appeal
03	Explore Intent to
04	Consult
05	Engage in Diplomatic
06	Engage in Material
07	Provide Aid
08	Yield
09	Investigate
10	Demand
11	Disapprove
12	Reject
13	Threaten
14	Protest
15	Exhibit Force Posture
16	Reduce Relation
17	Coerce
18	Assault
19	Fight
20	Unconventional Mass

Panel B. Heat map based on hierarchical clustering for similar events for USA

Figure 6-23. Dynamic Time Warp distance based Hierarchical Clustering for Protests in the USA, 500-daily sample

To compare more simple models to predicting highly-noisy stochastic random variables such as the Number of mentions (NM) variable, the results of linear regression and support vector regressions are presented in Table (6-5). The linear regression model with training on technical feature (NM, GoldsteinScale, AvgTone) for lag days (6, 13, 20) with target variable as NM of event on 7th, 14th, 21st day is presented below. The Support Vector Regression(SVR) model with technical feature (NM, GoldsteinScale, AvgTone) for lag days (6, 13, 20) and target variable as NM of event on 7th, 14th, 21st day is also presented. The prediction sequence is presented in Figure (6-24), where green is actual data and red is predicted data. The results highlight that traditional models fail when trying to predict based on linear regressions and support vector regressions. These models are not good fit to identify the rise and fall patterns in highly dynamic environments such as global geopolitical internet and news movements.

Table 6-5. Linear Regression and Support Vector Regression for Protests in the USA, daily data



Model	Country	EventRootCode	lag_days	RMSE	CORR
Linear Regression	USA	Protests	6	1.0229	-0.0782
Linear Regression	USA	Protests	13	0.9171	0.0066
Linear Regression	USA	Protests	20	0.9378	-0.0175
Linear Regression	USA	Mass Violence	6	1.0602	-0.2226
Linear Regression	USA	Mass Violence	13	1.2139	-0.142
Linear Regression	USA	Mass Violence	20	1.4406	-0.0782
Support Vector Regression (SVR)	USA	Protests	6	1.0274	0.0733
Support Vector Regression (SVR)	USA	Protests	13	0.8275	0.2175
Support Vector Regression (SVR)	USA	Protests	20	0.8536	0.0142
Support Vector Regression (SVR)	USA	Mass Violence	6	0.9997	0.0018
Support Vector Regression (SVR)	USA	Mass Violence	13	1.017	-0.0581
Support Vector Regression (SVR)	USA	Mass Violence	20	1.043	0.0953

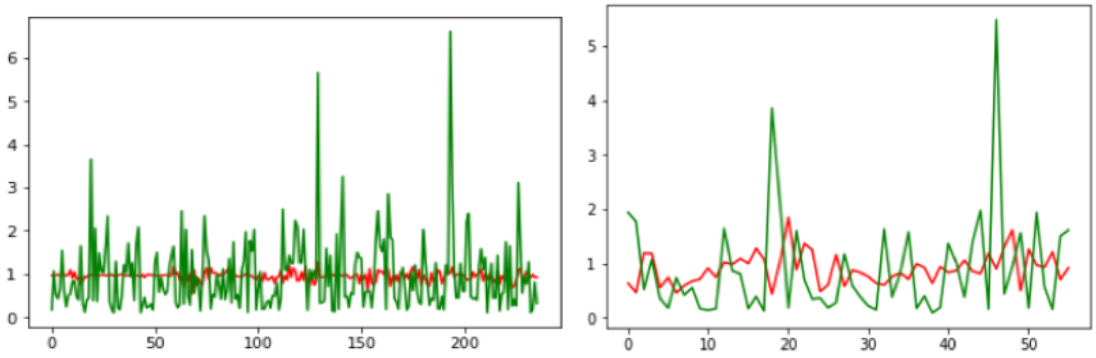


Figure 6-24. Linear Regression and Support Vector Regression for Protests in the USA. Note: Green is the actual trend, and red is the predicted.

Now in order to predict highly stochastic and dynamic random variable such as number of mentions, it is clear that traditional models do not cut it. Now deep RNN and LSTM models are used for training and testing the Protest trends in the USA from GDELT. The plot on test data ('US', 14 = Protests) based on LSTM is shown in Figure (6-25). Results on test data are reasonable, and at least are able to capture a nice pattern as expected by the noisy data. The models can recognize rise and fall patterns very well on training as well as test data. The data slice used for the experiment is the average and total number of mentions of cameo code 14, US from GDELT database, using GDELT API. The start date used is 2015-02-18 and the split date used is 2017-06-01. The test set for the experiment span till the latest entry in the dataset.

For this experiment, the dependent variable chosen are (a) total number of mentions, and (b) average number of mentions. The difference between the interpretation of these two variable makes a difference for decision-making. The total number of mentions in our interpretation is a good proxy for real-world protest event, however, more specifically the total number of popularity of mentions across news articles, web media, and so forth. The average number of mentions, can be interpreted as the average popularity of protests being mentioned on article on a specific day. Thereby in order to incorporate the different risk perspectives associated with a mega-project, the total number of mentions may be more relevant for policy makers to get a sense of aggregate protest movement, whereas public relations and news media agencies may be more interested in average number of mentions since it measures the average popularity in the new media for a given topic.

The training model accommodates three groups of features; including (a) similar events in the USA, (b) similar protest trends in other countries, and (c) technical features of the protest trend in the USA. The similar events time-trend as protests in USA are fed to the LSTM model.

The noise in other events across USA, this model does not perform very well. Based on training on similar countries, the model performs better but still do not cut it (results not shown). The uniqueness of USA in the time-trend of protests, this grouping may be more relevant for other countries with similar protests; nevertheless, these results are informative for ensembling the models later in this chapter. Lastly, the LSTM model's results based on training with the technical features are presented yield the most important training features. The input technical features for this data structre from GDELT include how many other mentions of this article (ToNumMentions), average number of mentions (avgNumMentions), the average tone around the article (avgAvgTone), standard deviation (stand_dev), and other mathematical operations such as kurtosis, skewness, etc. of the dependent variable. The output is the total number of mentions or the average number of mentions as per the prediction problem

The results show that vanilla LSTM model with a three layer of 128 LSTM cells with dropouts [0, 0.3, 0.5] and recurrent dropouts [0.3,0.3,0.3]. The model is trained for 50 epochs using various regression metrics. Similarly, the vanilla LSTM for the average number of mentions also has three layer of 128 LSTM cells with dropouts [0, 0.3, 0.5] and recurrent dropouts [0.3,0.3,0.3]. The model is trained for 50 epochs using various regression metrics. Figure (6-26) summarizes the different model performance metrics succinctly in the model reliability matrix. The parameters of the models for the experiment performed are as follows in table (6-6).

The uncertainty quantification method used for this experiment is the Monte Carlo Dropout. Currently the implementation is only able to quantify the model misspecification and the inherent noise. This also contains a confidence interval function for estimating the prediction bounds based on a confidence interval. The UQ implementation is present in the appendices section for reference the following scores are the result of applying the models for different domains.

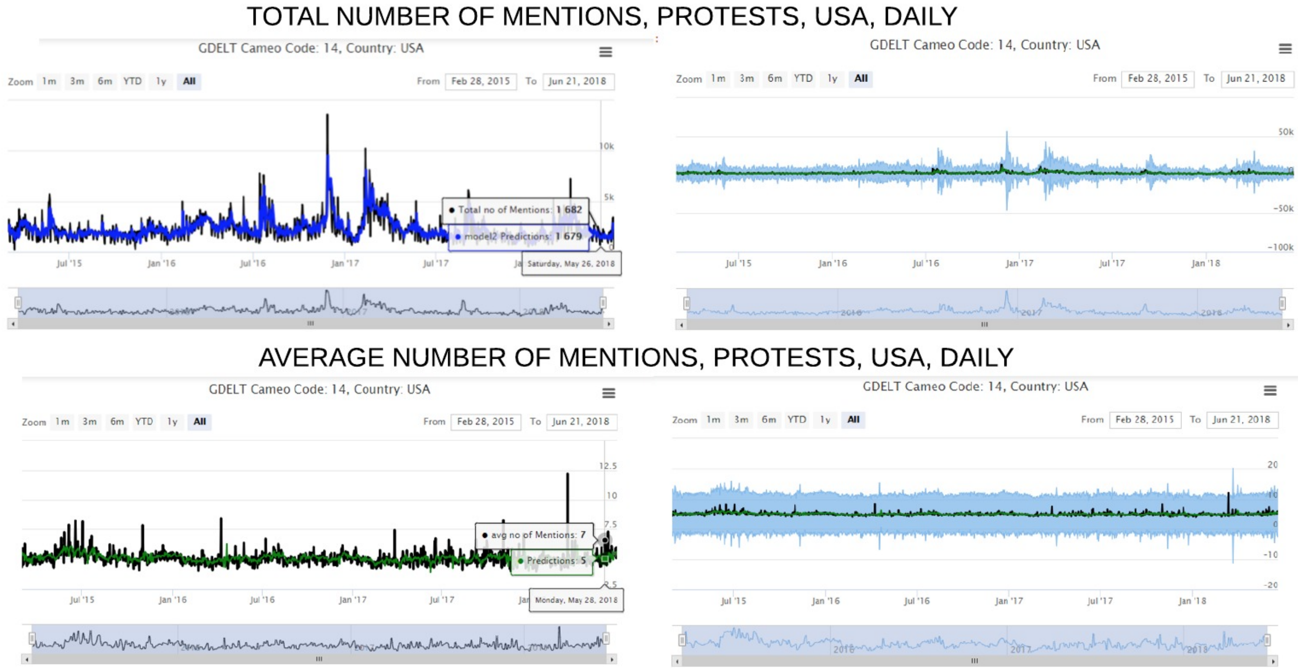


Figure 6-25. LSTM models for Protests (number of mentions) in the USA

Table 6-6. Model Parameters, Protests, USA, daily

Model	**LSTM (total number of mentions)**	**LSTM (average number of mentions)**
Input shape	(10, 6)	(10, 6)
LSTM cells	[128, 128, 128]	[128, 128, 128]
Dropouts	[0, 0.3, 0.5]	[0, 0.3, 0.5]
Recurrent Dropouts	[0.3, 0.3, 0,3]	[0.3, 0.3, 0,3]
Epochs	50	50
BatchSize	100	100
Metrics	mean squared error	mean squared error

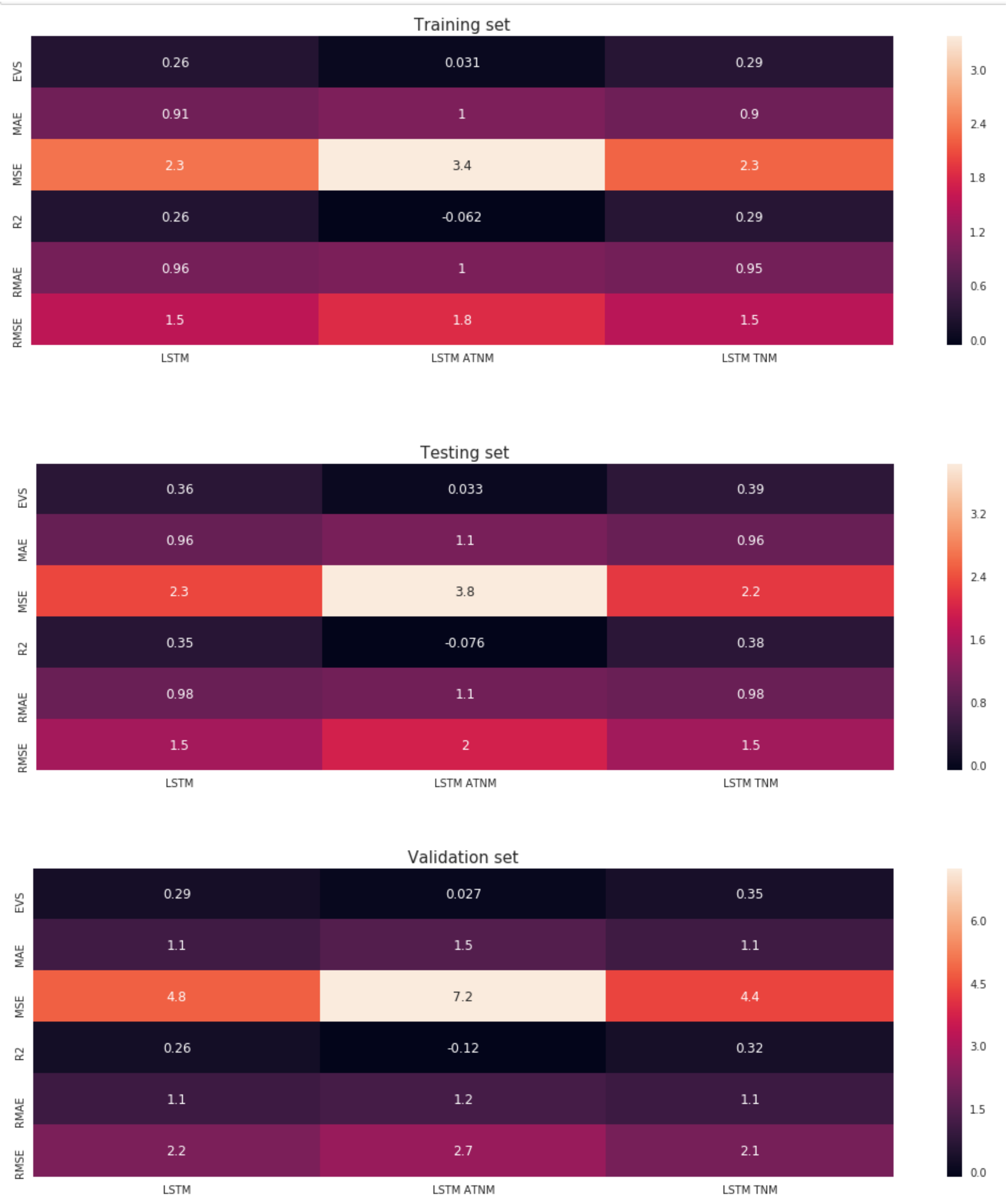


Figure 6-26. Model Reliability Matrix for US Protests, daily trends. ATNM – Average number of mentions, and TNM – Total number of mentions

Finally, the ensemble model using a simple weighted average for total number of mentions in USA protests is presented. A weighted sum of all the different prediction model is done to generate some randomness in prediction and control for overfitting. The weightage assignment chosen is the following: (a) similar events – 10 percent, (b) similar countries – 10 percent, (c) technical features – 75 percent. The final model results for protests in the USA in the testing set are presented below in figure (6-27).

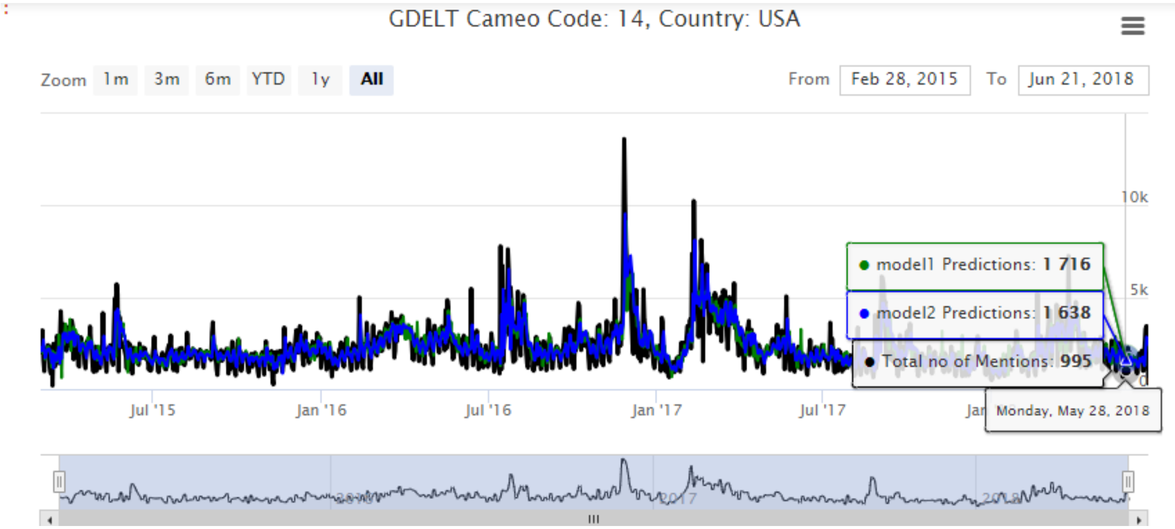


Figure 6-27. Ensemble Weighted deep LSTM for Protests in the USA

6.4. Monte Carlo (MC) Simulation in LSTM Networks

The last section validated the ensemble LSTM neural networks architecture as powerful and reliable tools for economic prediction problems. This sub-section provides a basis to go beyond the black box models to illustrate treatment of stress-testing problems in the confines of deep learning architecture. Decision-makers are keen on issues such as **endogeneity** and **causality** that help them reason and make important deductions about a given situation. They seek explanations, uncertainty bounds, and causal dynamics of the forecasts. The simulations approach may help decoupling the interdependence amongst training features.

Based on the results presented earlier for GDP growth for USA, a Monte Carlo simulation was conducted. The simulation was done by randomly choosing walks from a normal distribution to check the response of the model. At the testing stage, replaced the actual test data of the '*value*' variable with the random walks. Following results were obtained. Panel A shows the updated simulations of new training data set, and Panel B shows the GDP growth predictions by shocking the various training data sequences. The results show the sensitivity of GDP growth prediction based on the experimental group. More specific cases will be shown. For example, identifying the which feature shock in Panel A yields fastest growth and (or) contract. Similarly, the ground truth forecast in presented in figure (6-28) for GDP growth will be used to update the training data for the second Control Experiment Model for another prediction problem say T-Bill yield prediction.

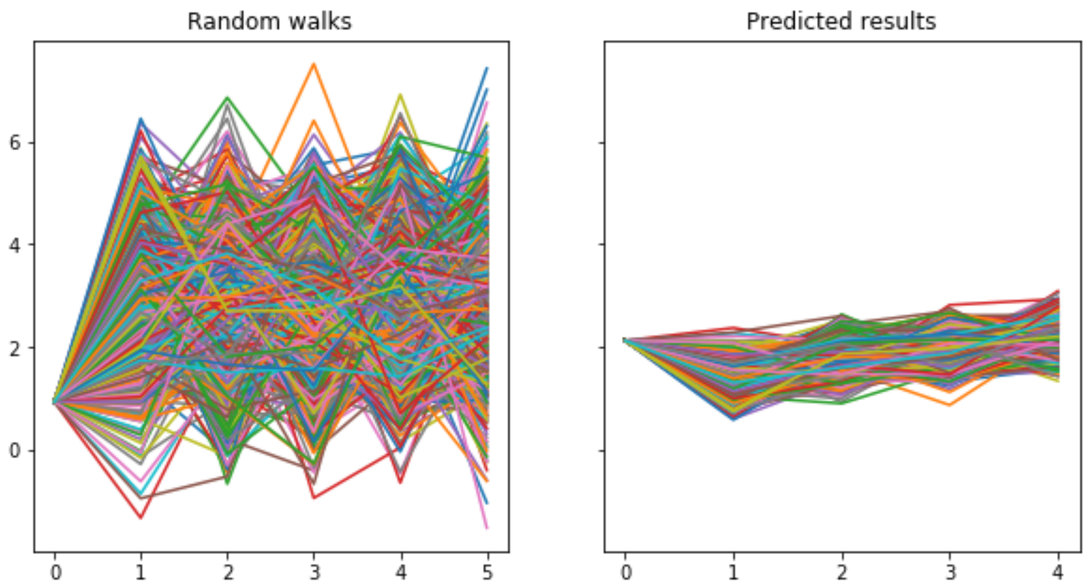


Figure 6-28. Example of shocking features and predicted result. Panel A. Random Walks in the Testing Dataset with MC Simulation, Panel B. Predicted change in output (GDP growth) for ensemble-LSTM model with simulated training dataset

The MC simulation cases can also be valuable in time-series information for “What-IF” analysis, and question and answering. For example, “which feature is most important for growth to decline in coming 2 years in the USA?”, or “which feature shall we focus on that will yield highest per capita income growth?”. This may not get to the heart of causality but provide inferences with their errors by decoupling interdependence.

One approach is to do simulations inside the confines of LSTM networks, that uses the capabilities of deep neural networks to gain more insights about the workings of the hidden layer and go beyond the black box. A concrete example is provided for per capita GDP growth in USA from 2015-17.

The model is set up in the following manner. Annual data for the USA is used with various features between 1960-2012. The testing set is for 2013-16 and the output is prediction for 2017. The actual value for 2017 are available. In order to make an insightful simulation original forecast are shown to have very low error with per capita GDP growth prediction in figure (6-29). Simulation of features in the test set for 2013-16 shock only the last 2 years i.e. 2015 and 2016 data with equivalent increments and decrements in each feature. The results are shown in table (6-7) and (6-8) for a 1 percent increment and decrement, respectively.

The results show that capital formation, domestic credit to the private sector, and exports are a key ingredient that could have raised per capita GDP growth in 2017. Similarly, a negative shock to reserves, exports, or domestic credit would hurt per capita GDP growth. These results provide a basis to untangle the black box of deep neural networks for deeper causal reasoning and evidence based reliable simulations that can directly aid decision-making. For example, an executive might be aware that there will be a protest tomorrow and knows the relationship between the protest and a given stock price ticker. This information can be used to simulate the impact on concerned prices based on reliable forecast estimate from another domain.

Deciding the optimal shock in itself is a very important question and within the confines of the model environment. There are various aspects related to the size of training features, batch size, window size, etc. that need to be consideration. For example, experiments were done increasing the size of annual training features by a massive quantity. The results showed that sensitivity increased substantially faster for the dependent variable as the training size increased. This was contrary to the pre-existing hypothesis that there would be more sensitivity to small data since the error was higher and degrees of freedom lower. Nevertheless, the attribute of simulation for understanding shocks in complex dynamic systems could be useful inference tool for decision-makers.

Table 6-7. Simulation 1 with change 1% increment

Changed Feature	Change in GDP per capita growth
Inflation, consumer prices (annual %)	0.62%
Merchandise imports (current US$)	2.72%
Merchandise exports (current US$)	3.96%
Total reserves (includes gold, current US$)	4.30%
Foreign direct investment, net (BoP, current US$)	4.50%
Population growth (annual %)	4.57%
Gross capital formation (% of GDP)	4.72%
Exports of goods and services (% of GDP)	4.73%
Domestic credit to private sector (% of GDP)	4.71%
CO2 emissions (metric tons per capita)	4.67%
Population density (people per sq. km of land area)	4.65%

Table 6-8. Simulation 2 with change 1% decrement

Changed Feature	Change in GDP per capita growth
Inflation, consumer prices (annual %)	-0.22%
Merchandise imports (current US$)	-0.78%
Merchandise exports (current US$)	-0.95%
Total reserves (includes gold, current US$)	-1.00%
Foreign direct investment, net (BoP, current US$)	-0.88%
Population growth (annual %)	-0.78%
Gross capital formation (% of GDP)	-0.92%
Exports of goods and services (% of GDP)	-1.05%
Domestic credit to private sector (% of GDP)	-1.15%
CO2 emissions (metric tons per capita)	-1.08%
Population density (people per sq. km of land area)	-1.05%

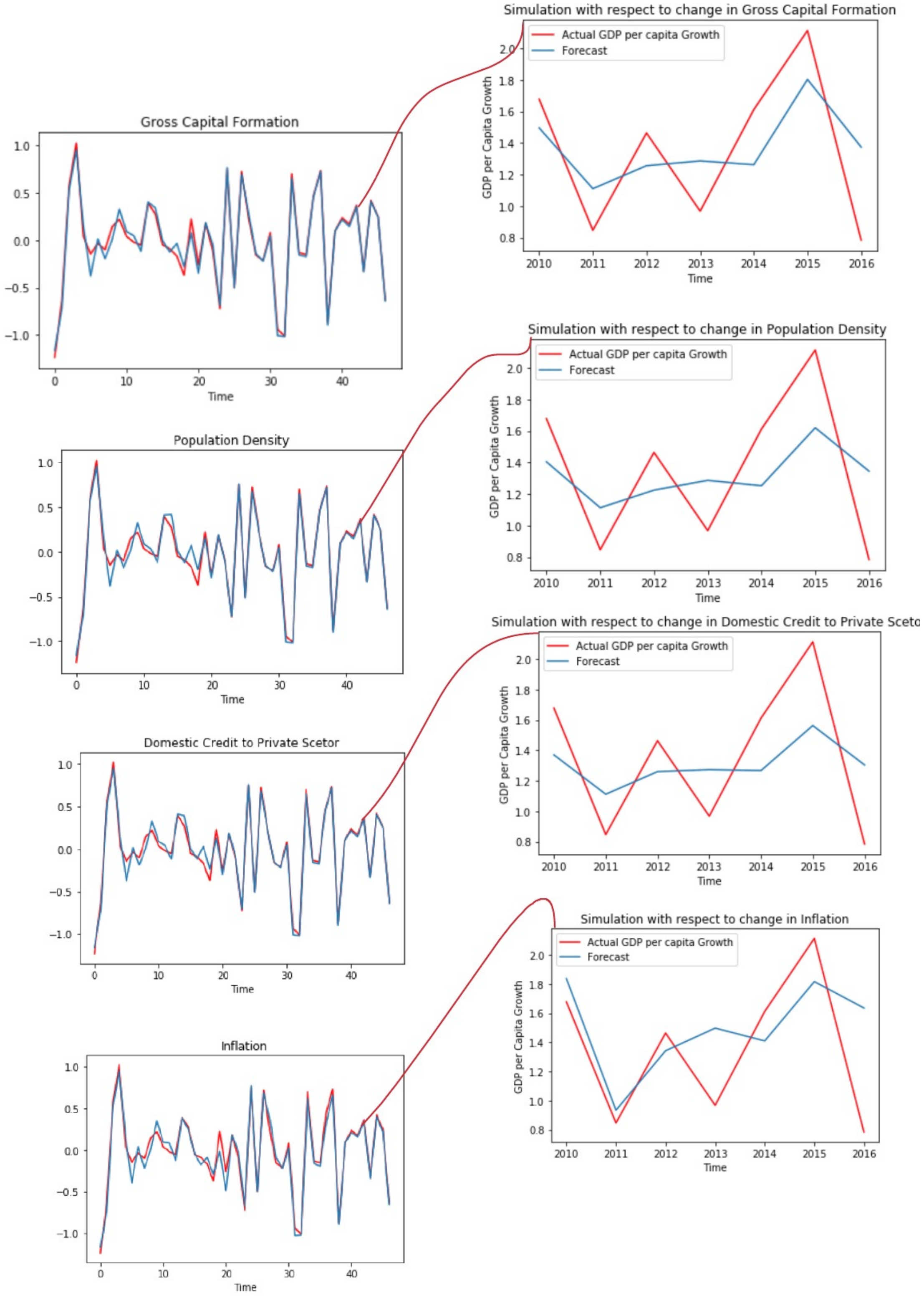


Figure 6-29. Visualization of 1 p.p. increment to Given Variables and impact on USA GDP per capita growth

6.5. Time Label Assignment (TLA) – Alert System

This section is shows a toy example of an Alert System that can be developed for dynamic monitoring of time-series information. As an example of this selected memory, simple labels for NBER Recession months are taken with movements in S&P500.

The US recession is a binary label that represent the official recession months for the US economy between 1st of January 1971 to 1st of February 2017. For better accuracy of the system, a larger dataset of training features can be selected but to show the test case, only S&P 500 stock indices are selected.

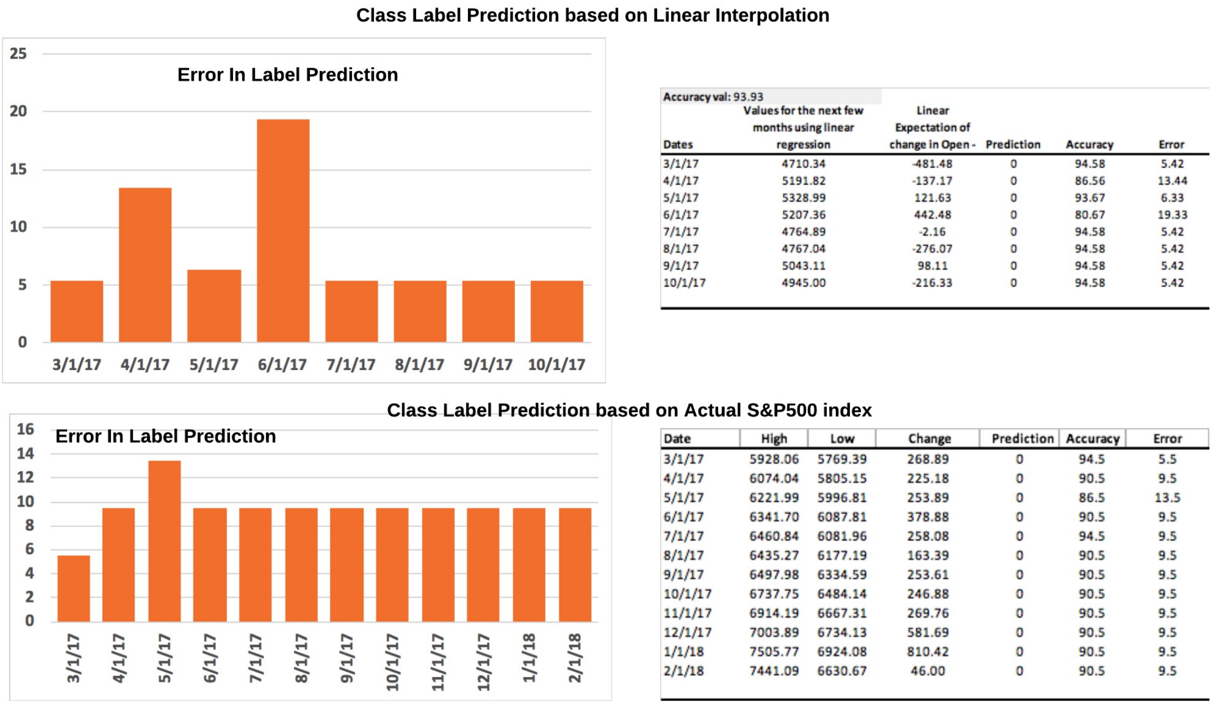
The input values are defined as “x”, i.e. the input variables passed to the RNN, and “y” represents the values across which the RNN will be trained. Since only on feature is used, a single LSTM is used to keep account of the previous set of values. For a higher accuracy, the number of features from various other datasets can be increased and complexity of the LSTM required shape and size can be made.

The output “y” values is the categorical variable i.e. the prediction 0 if there is no recession and 1 if there is a recession. The model is trained after verifying the data is correctly structured. For validation i.e. to check the accuracy of the model, values from the test data is used. Here the rolling window is 10 months. This is a sequential RNN model that will allow to add layers on top of the model. The shape of the data is defined by using embedding’s to provide the shape. Dropout is added it to the model which allows for faster and more accurate training of the model. The LSTM model is used to recognize patterns in the data where softmax is the LSTM's activation with 2 hidden layers. The loss function is used to increase the accuracy where a neural network is rewarded the output is right. If the network is wrong in the output it is provided with no rewards. Categorical cross entropy is used for the loss function. Now the training values in the model that were used training earlier are used to define the batch size (Number of inputs that a neural network such as RNN takes at one time and trains on it) and set up the epochs (Number of times a network is trained on the same set of values received from the batch size). To find the score and accuracy of the network the validation set is given. The system will pass some inputs to score the output by scoring which outputs it got correct and which it got wrong.

This approach can function as an alert system. The LSTM network is used to predict recession giving output as '0' or a '1' (0: represents no recession in that year. 1: represents recession in that year). A liner regression is used to predict the required set of values for sample forecasts. According to the predicted values the RNN predicts the incoming recessions. The network is trained for a time period of 1st of January 1971 to 1st of February 2017 of the American market. Features are the values which represent the data along which the prediction will be made. Many technical features such and macroeconomic features are accommodated. This is used to create the training and testing data use cross validation process that validate and test the system for its accuracy. A linear regression as one sample of a simulation and another with actual data. The results for the inaccuracy of the model are presented in month in figure (6-30). The alarm could go off at a certain threshold where the decision-maker set, for example, anything month that provides an inaccuracy of greater than 80 percent could trigger the alarm.

There are plenty of opportunities to accommodate many new features, and the test alarm system on other class label data. In many real-world cases, one will note that the cases of failure are less recorded and there are only few training samples for such tipping-points in the instances.

This section has provided a simple basis to accommodate using time as a label and predicting the current or simulated time-period in similar labels.



Accuracy val: 93.93

Dates	Values for the next few months using linear regression	Linear Expectation of change in Open -	Prediction	Accuracy	Error
3/1/17	4710.34	-481.48	0	94.58	5.42
4/1/17	5191.82	-137.17	0	86.56	13.44
5/1/17	5328.99	121.63	0	93.67	6.33
6/1/17	5207.36	442.48	0	80.67	19.33
7/1/17	4764.89	-2.16	0	94.58	5.42
8/1/17	4767.04	-276.07	0	94.58	5.42
9/1/17	5043.11	98.11	0	94.58	5.42
10/1/17	4945.00	-216.33	0	94.58	5.42

Date	High	Low	Change	Prediction	Accuracy	Error
3/1/17	5928.06	5769.39	268.89	0	94.5	5.5
4/1/17	6074.04	5805.15	225.18	0	90.5	9.5
5/1/17	6221.99	5996.81	253.89	0	86.5	13.5
6/1/17	6341.70	6087.81	378.88	0	90.5	9.5
7/1/17	6460.84	6081.96	258.08	0	94.5	9.5
8/1/17	6435.27	6177.19	163.39	0	90.5	9.5
9/1/17	6497.98	6334.59	253.61	0	90.5	9.5
10/1/17	6737.75	6484.14	246.88	0	90.5	9.5
11/1/17	6914.19	6667.31	269.76	0	90.5	9.5
12/1/17	7003.89	6734.13	581.69	0	90.5	9.5
1/1/18	7505.77	6924.08	810.42	0	90.5	9.5
2/1/18	7441.09	6630.67	46.00	0	90.5	9.5

Figure 6-32. Alert System Example: Inaccuracy of “Recession” Label Prediction based on S&P500, over months

Chapter 7: Executive Decision-Making in Mega-Project Management

Accurate and reliable forecasts can aid the executive decision-making process related to managing engineering mega-projects by bringing awareness to uncertainties associated with potential external opportunities and threats. The previous chapter presented the results of using ensemble deep learning-model approach and the capabilities of simulation-based reasoning tasks. These forecasting capabilities and collective forecasts provide a basis for monitoring technologies that can be deployed for managers to monitor the state of key indicators and prognostics for the system of their interest. According to Georgoff and Murdick (1986), the holy grail for all professions, from financial trading activities to management science, is the constant monitoring of forecasting for the external environment has been identified as key to successful executive decision-making. Therefore, applying the resulted predictions, with transparency to their error provided together for multiple domains, will help make these problems of uncertainty-management more manageable. For example, in an energy mega-project that is being planned for across the nation, it will be significantly beneficial if the executives engaged, across multiple organizations, in the project had access to a common-pool of knowledge that provided them with forecasts for different time-horizons and for various pertinent domains. These forecasts could include financial performance of contractors, governance, economic growth, jobs related to the energy sector, trade, social and economic development related to the districts and the nation. Furthermore, the application of the prediction models will be of greater value to executive-decision making when integrating different domain-forecasts related to the country and geolocations of the given mega-project and assets related to the project. In this manner, cross-domain forecast feedbacks can be monitored for different time-frequencies simultaneously. This has opened a new area of systems research that encompasses what can be called “high-dimensional multi-variate time-series ensemble deep learning" or HMT ensemble deep learning. Layers of complexity are added in the information architecture as preference matching and mapping HMT ensemble deep learning to the mega-project.

There are few success cases for such monitoring technologies, a good example is applications in global climate policy. Organizations such as National Oceanic and Atmospheric Administration (NOAA), United States Geological Survey (USGS), or National Aeronautics and Space Administration (NASA), as well as defense laboratories such as Sandia Labs or Applied Physics Lab (APL) are leading innovative new work to directly aid various catastrophic risk management techniques using monitoring technologies for climate policy-related decision-making. Moreover, monitoring technologies are also used in engineering systems; such as: off-shore wind-farms, large-power plants, industrial robotics, and for maintenance and prognostics of varied large-scale complex engineering systems. However, the application of such monitoring technologies for complex and dynamic decision-making concerning international business and policy strategy is at a nascent stage of study, especially in context of high-level executive decision-making (Schrage 2017, Aggarwal et al. 2017). Therefore, in this chapter the applications of the monitoring methods that can aid executives in managing LEP's will be presented.

In the modern-day business environment, executives have to perform important and dynamic functions for mega-project management including fostering new partnerships and orchestrating large-teams and various stakeholders. They also need to be cognizant of potential tipping points, risks and opportunities, from the perspective of multiple stakeholder. Increasingly, executives have to operate in an even-more dynamic and uncertain global environment; hence, the best executives are made of certain traits and qualities, that include incorporating monitoring feedback systems to inform uncertainty-adjusted decision-making throughout the life-cycle of the project (Laufer 1997).

This chapter will start with discussing the application of incorporating foresight monitoring technologies in project management, to better plan and be informed about the potential external threats and opportunities. Then, an assessment of mega-project success and failure from an empirical lens will be conducted; and, finally, the applications of foresight monitoring will be discussed in the last section. Nevertheless, given the complexity of the massive data, model computation, forecast integration across domains, deployment strategies, and the varied choices for visualizations, this study provides a basis towards a systematic framework that will be valuable for decision-makers in the future.

7.1. Incorporating Foresight Monitoring Technologies in Project Management

There is a truism that “Forecasting would be a subset of prediction.”, and “all forecasts are predictions, but not all predictions are forecasts.” (Snowberg et al. 2012). In similar vein, there is a psychology construct called “foresights” that embodies forecasts and is very useful concept for planning and strategy development (EPAS 2017, UNDP 2011, Wiklinson and Koopers 2013). Therefore, the awareness of in-coming opportunities or threats can provide valuable foresights to aid uncertainty-adjusted decision-making.

Foresights include forecasts but are not dependent on forecasts alone. Foresight includes both cross-disciplinary forecasting, and the process of getting the forecast into the decision-making process. Lot of attention has been given to the former since the 1980’s but little attention has been paid to the latter (Grant 1988). Hence, an important element to foresight is to anticipate the different risk perspectives of the multiple stakeholders engaged in the mega-project. For example, a country manager needs to be aware of broad macroeconomic trends like economic growth, inflation, or real-effective exchange rate prospects; an investment bankers or trader may be interested more in the stock performance of companies on new deal assignment, interest-rate parity, or cross-currency basis; governments' interest may be more on the potential political threats, uncertainty concerning geopolitical tensions, central bank policy moves, and strategies to diversify their sources of income by economic diversification; and, finally, local population i.e. the beneficiaries of the mega-project, may have concerns about the impact of these projects on jobs, social, and economic development in the city. Consequently, visualizing, anticipating, foreseeing and perceiving these different factors from different risk perspectives becomes crucial for successful managers.

In addition, foresights can aid important decisions-making throughout the lifecycle of the project. Foresights should be generated particularly at the inception of the project cycle, but also during and after the project. As demonstrated by figure (7-1), the impact of the executives' decisions on the project's cost performance declines dramatically as the project progresses and the maximum potential for influencing the project's cost occurs in the conceptual and definition phase (HBR 2016, Laufer 2012). Moreover, it is important to note that more than one third of the reasons why projects fail is out of the manager's hand (Laufer 1997). There are the external threats and opportunities that are unaccounted for; thus, this research can help in reducing this "out of hand" uncertainty.

A simple example of how foresights can aid decision-making on mega-projects can be the following case. Consider, that a reliable forecast for variables that could include household-income, growth in software-engineers and network scientists' occupations, living costs…etc. of different districts and cities in a given country, will aid the decision-maker during the planning stage as to where to set up the mega-project like a data-center. The executive could understand the cases by running simulations of the potential opportunities and threats in that city or country based on foresights of various measures of interest. Similarly, such foresight monitoring technologies should be used during the life-cycle of the project and assets related to the project, as new data is received to improve the short, medium, and long-run foresight capabilities.

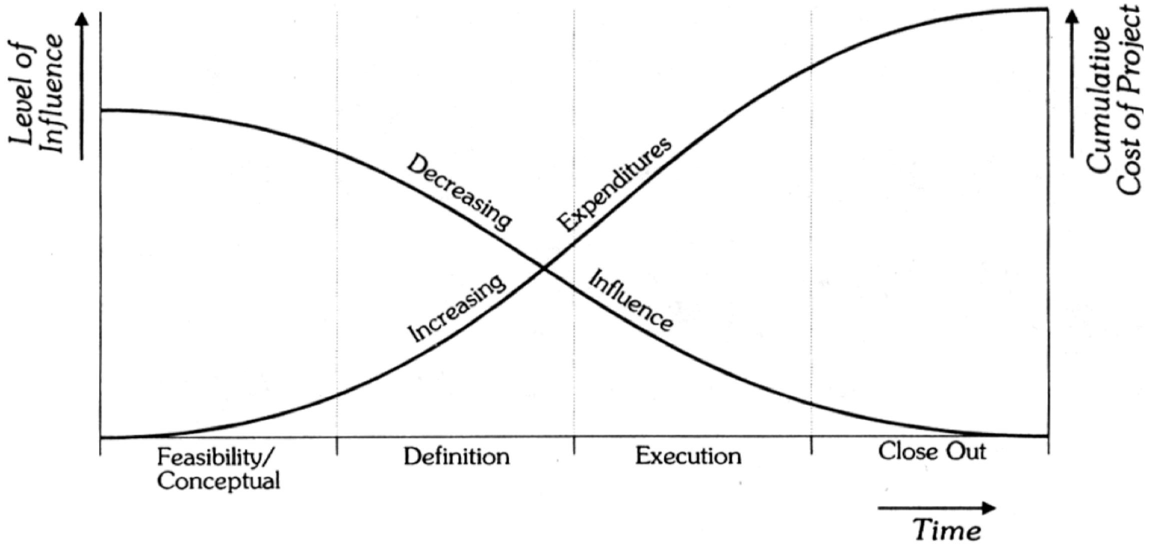


Figure 7-1. Ability to Influence Project Costs
Source: Alexander Laufer 1997.

7.2. Assessment of Mega-Project Success and Failure

There is a paucity of detailed data concerning mega-projects. Most companies and governments keep this information as a proprietary. The only source of reliable, global and cross-sector mega-projects information are publicly available from the World Bank Group Projects database. This database contains detailed information concerning over 17,000 large global projects executed between 1947 – 2017; ranging in size between $1 million with largest individual projects approaching $4 billion in over 173 countries. These projects range across sectors such as: agriculture, forestry fishing, education, energy and extractives, finance, information and communication technologies, transportation, infrastructure, and water and sanitation. Furthermore, the database contains a list of active and pipeline projects for the coming decade. Consequently, the World Bank database will be utilized to extract data about mega-projects which will then be used to demonstrate the application of the developed models and assess their accuracy. In order to demonstrate the application of the developed models, this section will, first, give a brief description about the World Bank Group's projects' structure; second, the specifics of the World Bank projects database, and the key trends are highlighted. This will be followed by leveraging machine learning classification models to assess the accuracy of predicting mega-project that were closed (success) or dropped (failure).

There are two types of World Bank projects: one, that have succeeded in the prosperity of a nation, and another where mega-projects leave many negative externalities as well as associated with many social and economic failures. Given the very complex nature of the global portfolio, on the ground realities in developing countries, multiple stakeholders, paucity of data, assessing the core response of these projects in a more comprehensive manner to address sustainability and positive spillovers such as long-term social and economic impacts is very difficult. Furthermore, isolating the causality, dependency, or endogeneity factor is harder. Ideally, this would require a massive simulation of various of these geo-locations, cities, districts, states, and countries, across tens of thousands of measures related to entities associated with these locations of the mega-project. In this view, there are simple lessons of how integrating mega-project analysis with foresights concerning uncertainties across various domains. This can play an instrumental role to enhance the executive decision-making process in a systematic manner.

In addition to the projects' classification, it is important to understand the project lifecycle for the World Bank which consists of six stages:

1. Identification: The World Bank, in coordination with other organizational units; such as: the International Finance Corporation (IFC) and Multi-lateral investment Guarentee Agency (MIGA) produce a strategy, called Country Partnership Framework (CPF), to identify the country's highest priorities for economic development.

2. Preparation: This stage is used to assess the capacity of the borrower to follow through on the project, they include conformity with World Bank's Safeguard Policies, Environment Assessment Report, Environment Action Plan, and other external threats reported such as Indigenous Peoples Plan.

3. Appraisal: This stage provides the stakeholders an opportunity to review the details of the project's design and resolve any outstanding concerns.

4. Negotiation/Approval: In this stage, the project appraisal document is prepared along with other financial and legal documents.

5. Implementation/Support: In this phase, the World Bank's role is to ensure adequate fiduciary controls, and adjust them in case of any restructuring of finance is necessary given local and external environmental conditions.

6. Completion/Evaluation: An implementation and completion report is compiled from the implementing government agency, and the various stakeholders. The Independent Evaluation Group (IEG) assesses the performance of the project on objectives such as sustainability and economic development, which is filed in the Impact Evaluation Report.

Next, the data for the projects' database is discussed briefly. The data is used from open source library World Bank projects database (http://projects.worldbank.org/). This data contains more than 17,000 projects with different categories like "Commitment Amount", "Country", "Status" and "Approval Date". Different types of data for the largest number of financed projects can be extracted from this database, but, first, to be able to interpret this data the regional codes used by the World Bank are shown in Table (7-1):

Table 7-1. 3-digit Regional Codes

Region name	**Region code**
Africa	AFR
East Asia and Pacific	EAP
South Asia	SA
Europe and Central Asia	ECA
Latin America and Caribbean	LAC
Middle East and North Africa	MENA
Other	Other

Figure (7-2) gives a sample of the data that can be extracted from the database. The bar graphs in figure (7-2) Panel A and B show the top 25 countries' total number of projects and total project amounts for different regions, respectively. Panel B shows the success rate i.e. the proportion of closed project divided by total number of projects in a given region, the error bars are graphical representations of the variability of data and used on graphs to indicate the error or uncertainty in a reported measurement. Panel C displays the different project status for the whole database, while Panel D highlights the shares of each status variable. This indicates that approximately 78 percent of projects are closed followed by “Active”, “Dropped” and “Pipelined” projects.

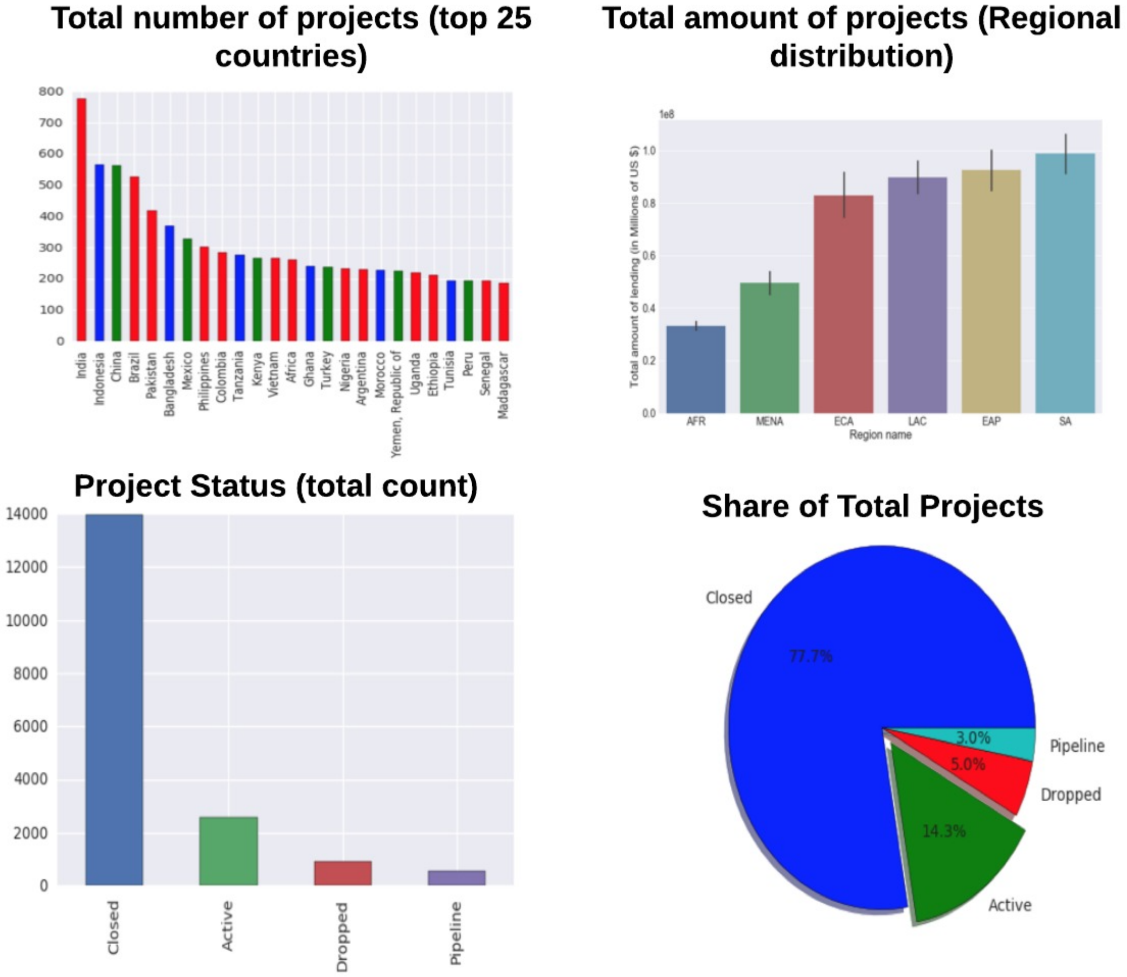


Figure 7-2. Summary of World Bank Projects Database

Nonetheless, the World Bank has no formal definition of the mega-projects and this definition is dependent on the preference of the analyst or the decision-maker. Thus, for conformity, this study defines the projects according to the following main project categories:

"Small", "Big", "Large", and "Mega". These project categories are defined based on the lending amount; hence, different mega-project distinctions are provided in Tables (7-2) & (7-3) below.

Table 7-2**.** Defining International Bank for Reconstruction and Development (IBRD) mega-project by project amount

Project code	Project category	Budget Range ($)
0	Big	0.1 Millions - 10 Millions
1	Large	10 Millions - 1 Billions
2	Mega	1 Billions - 100 Billions
3	Small	0 - 0.1 Millions

Table 7-3. Defining International Development Assistance (IDA) mega-project by project amount

Project code	Project category	Budget Range ($)
0	Big	0.1 Million - 10 Millions
1	Large	10 Million - 200 Millions
2	Mega	200 million - 100 Billions
3	Small	0 - 0.1 Millions

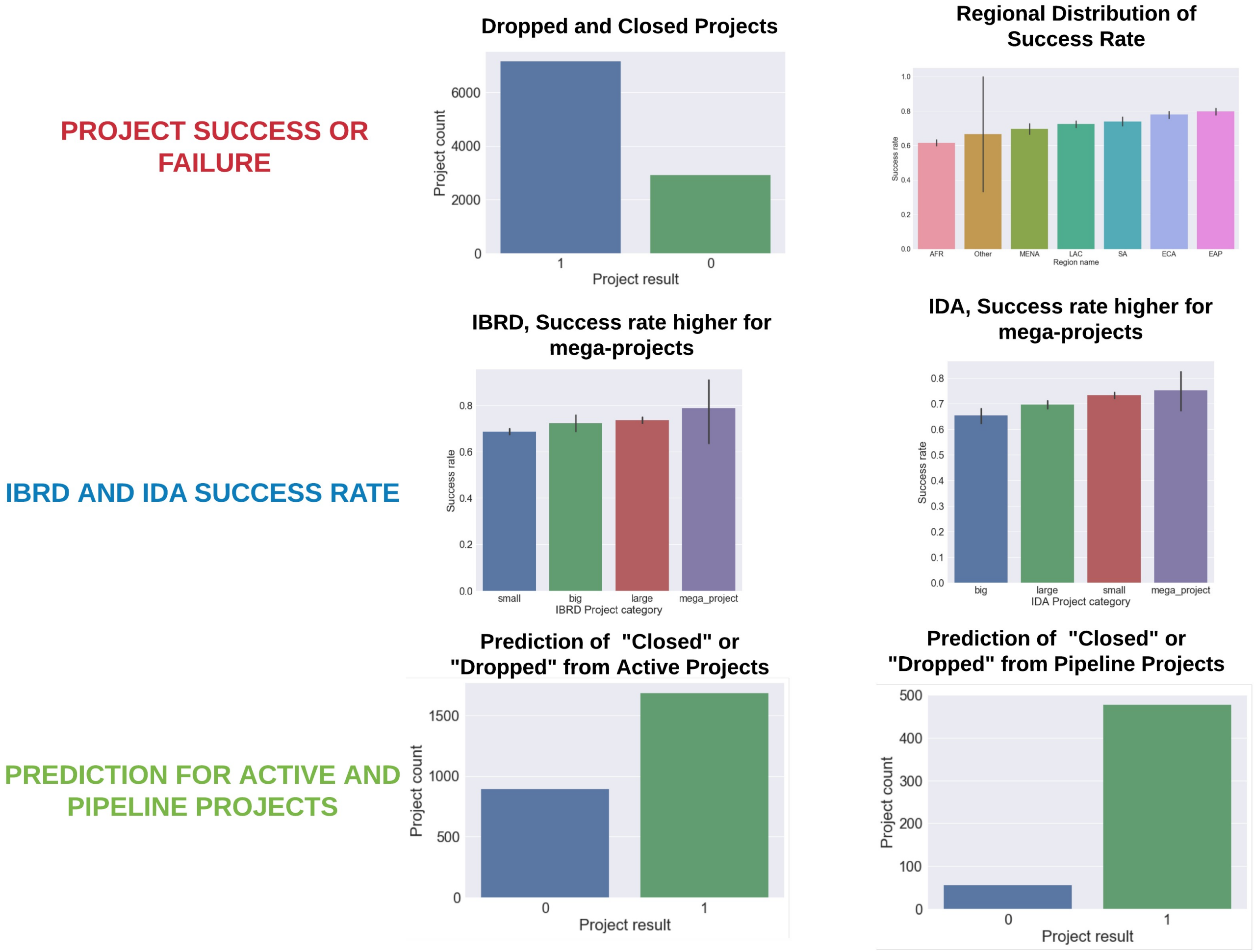
To further classify these projects into success or failure, the Independent Evaluation Group (IEG), which is a department at the Bank that does post-project evaluations created the dependent binary variable and the binary classification of project success or failure is presented in Table (7-4) below:

Table 7-4. Binarization of IEG Project Performance Categories

IEG Project Performance Categories	Binary Class
Satisfactory	1
Moderately Satisfactory	1
Highly Satisfactory	1
Unsatisfactory	0
Moderately Unsatisfactory	0
Highly Unsatisfactory	0
Not Rated	0
Not Available	0
Not Applicable	0

Finally, the next charts show an empirical view of project success and failure. Figure (7-3) Panel A shows the count of “Dropped” and “Closed” projects, whereas Panel B shows the regional distribution of project success rate. Panels C and D show the IBRD and IDA project success rates respectively, showing that smaller projects are generally less successful than mega-projects.

After providing a brief overview about the World Bank’s database, the simple classification model’s results are presented here to predict the performance of the projects as either satisfactory or non-satisfactory, which will then be compared to the predictions resulting from the current World Bank practices. In order to perform this task, the first step is to divide the data into training and test data, 20 percent test data and 80 percent training data. Second, for the test data, the columns representing the status of the projects are removed and the data is pre-processed where encoding is done, and unnecessary features are dropped. Furthermore, the World Bank IEG data that rates the ex-post performance of “Closed” projects are merged based on the project id. Finally, the data is trained on Status: “Closed” projects based on the IEG binarized performance criteria; and the results indicate that given certain project features in Table (7-4), the likelihood of either a “Satisfactory” or “Non-Satisfactory” project evaluation rating differs. Panels E and F show the prediction of whether “Active” and “Pipeline” projects will be satisfactory or non-satisfactory.



PROJECT SUCCESS OR FAILURE

IBRD AND IDA SUCCESS RATE

PREDICTION FOR ACTIVE AND PIPELINE PROJECTS

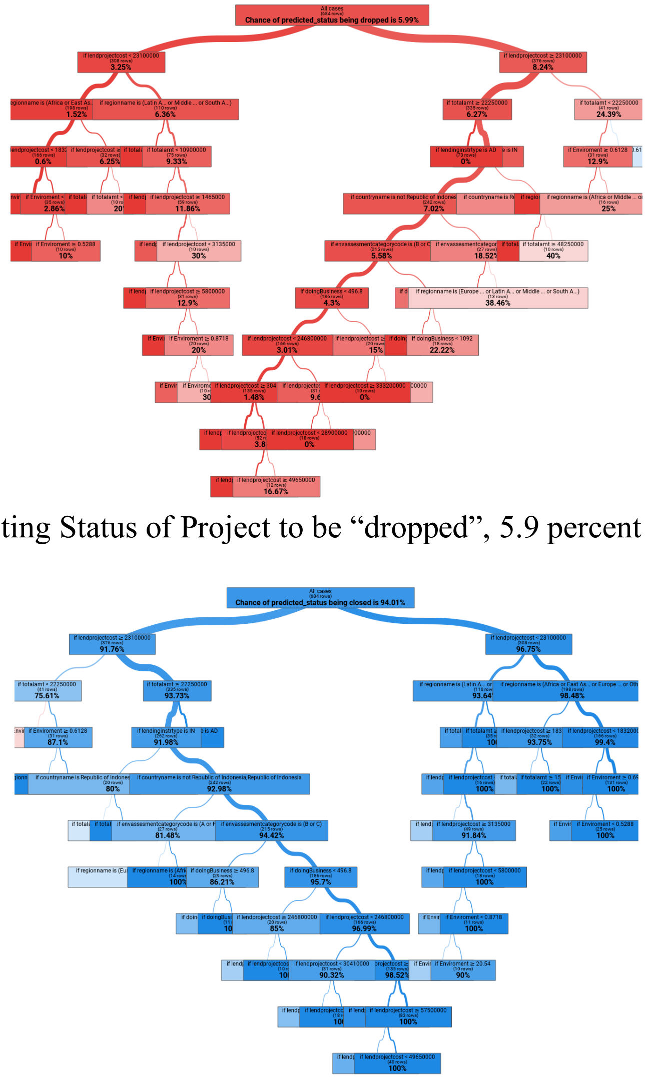
Figure 7-3. Aggregating Project Success/Failure rate

Similarly, classification models can also be developed to predict whether a project will fail (i.e. be dropped) or succeed (i.e. closed) based on the project and country features. The different classification model results accuracy rate is presented in the following table (7-5). Many other methods were also tested, but decision tree classifier seemed to be most accurate with small data. A visualization of the decision-tree for the test set is provided in figure (7-4). It is important to note that these models' results are useful priori information to assess the success or failure of the project; however, the domain and location specific intelligence, and many external features are not yet incorporated in these instances. This result provides the decision-maker a project-by-project basis to get a crude and base level classification given the project features; such as: the total lending amount, sector, the environment for doing business, macroeconomic, environmental and political features of the country. This is incomplete, since there are many local features in proximity to the project location, such as district or city specific monitoring for potential opportunities and threats are not incorporated here.

In spite of its robust accuracy when compared to the current practices, this stand-alone information can be improved by merging data from different domains related to the location and country of the project and foresight monitoring. However, integrating forecasts from various domains is a highly complex task. Nevertheless, this research has opened the door for a new array of research related to system design and reliability to accommodate complex computation in dynamic environments. Information monitoring and forecasting have to be built into the decision to provide continuous testing, against actual events, of the expectations that underlie the decisions.

Table 7-5. Results of the Classification Models for dropping project

Classification models	**Accuracy of classification model**
lightGBM	0.94
Naïve Bayes	0.93
Decision Tree	0.99


All cases
(684 rows)
Chance of predicted_status being dropped is 5.99%
if lendprojectcost < 23100000
(308 rows)
3.25%
if lendprojectcost ≥ 23100000
(376 rows)
8.24%
if regionname is (Africa or East As...
(198 rows)
1.52%
if regionname is (Latin A... or Middle ... or South A...)
(110 rows)
6.36%
if totalamt ≥ 22250000
(335 rows)
6.27%
if totalamt < 22250000
(41 rows)
24.39%
if lendprojectcost < 1832...
(166 rows)
0.6%
if lendprojectcost ≥ ...
(32 rows)
6.25%
if totalamt < 10900000
(75 rows)
9.33%
if lendinginstrtype is AD
(73 rows)
0%
if Enviroment ≥ 0.6128
(31 rows)
12.9%
if Enviroment < ...
(35 rows)
2.86%
if lendprojectcost ≥ 1465000
(59 rows)
11.86%
if countryname is not Republic of Indones...
(242 rows)
7.02%
if regionname is (Africa or Middle ... o...
(16 rows)
25%
if Enviroment ≥ 0.5288
(10 rows)
10%
if lendprojectcost < 3135000
(10 rows)
30%
if envassesmentcategorycode is (B or C...
(215 rows)
5.58%
if envassesmentcategor...
(27 rows)
18.52%
if totalamt ≥ 48250000
(10 rows)
40%
if lendprojectcost ≥ 49650000
(12 rows)
16.67%
All cases
(684 rows)
Chance of predicted_status being closed is 94.01%
if lendprojectcost ≥ 23100000
(376 rows)
91.76%
if lendprojectcost < 23100000
(308 rows)
96.75%
if totalamt < 22250000
(41 rows)
75.61%
if totalamt ≥ 22250000
(335 rows)
93.73%
if regionname is (Africa or East As... or Europe ... or Oth...
(198 rows)
98.48%
if Enviroment ≥ 0.6128
(31 rows)
87.1%
if lendinginstrtype is IN
(262 rows)
91.98%
if lendprojectcost ≥ 183...
(32 rows)
93.75%
if lendprojectcost < 183200...
(166 rows)
99.4%
if countryname is Republic of Indones...
(20 rows)
80%
if countryname is not Republic of Indonesia;Republic of Indonesia
(242 rows)
92.98%
if envassesmentcategorycode is (A or F...
(27 rows)
81.48%
if envassesmentcategorycode is (B or C)
(215 rows)
94.42%
if lendprojectcost ≥ 3135000
(49 rows)
91.84%
if Enviroment < 0.5288
(25 rows)
100%
if lendprojectcost < 49650000
(40 rows)
100%

Panel A. Predicting Status of Project to be "dropped", 5.9 percent in global test set

Panel B. Predicting Status of Project to be "closed", 5.9 percent in global test set

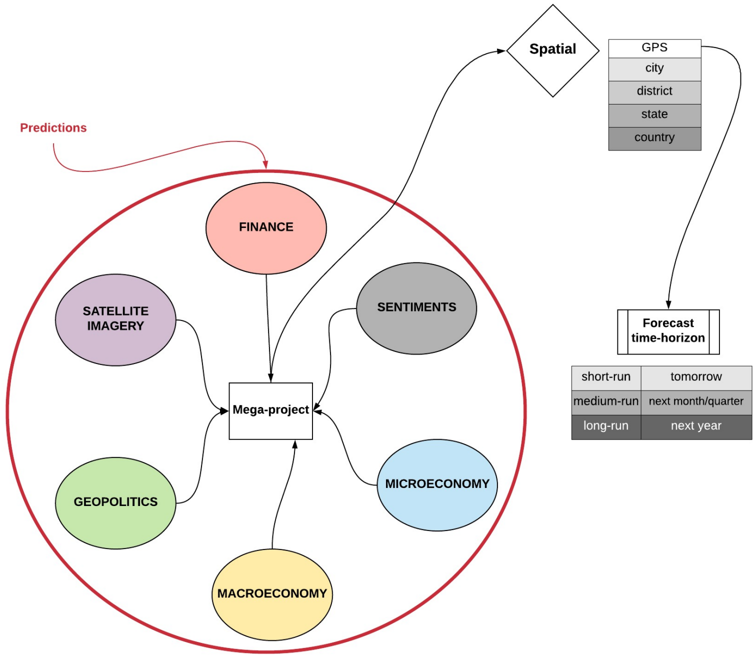
Figure 7-4. Decision Tree Visualization of Probability of dropping a project

Therefore, to help the executive decision-makers of LEPs, this research will contribute in bringing awareness to external opportunities and threats to inform timely uncertainty adjusted decision-making rather than assessing the failure or the success of the project beforehand. The importance of the above models and results highlighted in earlier chapters is to train the decision-maker to practice value of planning and monitoring constantly.

Now the cross-domain features of importance given the project features can be forecasted and provided to the decision-maker to monitor these external opportunities and threats. The decision-maker must make his, or her, own judgement related to the forecast features, errors associated with the forecasts to plan to get either new information, communicate the findings with the stakeholders, and take action. It is important to take into account the external opportunities and threats and bring visibility to these foresights for greater awareness of the decision-maker. In some circumstances, predictions for country features; such as: a contracting economy, rising unemployment, or higher than expected rise in central government troubles as witnessed through increase in public debt, and financial constraints to do business, would help inform tactics to plan and prepare to successfully deliver the project. In other instances, the country features may not be that insightful; however, city and district-specific local features predictions for housing, income, labor markets, or environment, can help prepare the executives to deal better with the in-coming threats and opportunities. In few cases, purely the technical design of the project, engineering constraints, and mismatch between the engineering details could be the root cause of the failure. However, the latter case of such internal engineering project failures is not the focus of this research and not pertinent to the applications.

Nevertheless, the information-system attributes of this problem entail many complexities. The information system would require the following attributes: (a) a systematic framework to integrate massive volumes of data, both at high-frequency as well as historic that is already organized to various spatial and temporal coordinates as discussed in Annex 2 on system architecture; (b) a system that would have computational capacity to make varied forecasts across various domains and different time horizons constantly, this requires massive computational capacity and new methods to address the computation of LSTM models for hundreds of thousands if not tens of millions of records on a frequent basis; (c) accommodate micro geospatial information from each government sources such as the Census, Survey of Industries, unit level household surveys…etc. to get estimates of districts and cities in close proximity to the project which are currently, not readily available across countries; and (d) a matching-system that maps the parameters of decision-making given project features to the country, state, and district/city level most that is most relevant to the external opportunities and threats features specific to that project. This would allow decision-makers to monitor or run simulations as provided in section 6.5 for monitoring and experimentation of the decision-making, that could especially be valuable for various scenario planning, testing at early stages of the project cycle, as well as, live on-going monitoring of “active” or “pipeline” projects. Some of these elements highlight the complexity of scaling this approaches that can directly aid executive decision-making. Consequently, as demonstrated above, this research has put forward a foundation for this new and highly specialized area of research, which is highly interdisciplinary and requires knowledge of different social sciences fields; such as: economics, finance, sociology, political science; in addition to engineering fields; such as: reliability engineering, systems engineering, computer science, scalable artificial intelligence and big data sciences; as well as understanding the perspectives from business and policy fields related to resource allocation and decision-making fields.

Moreover, this research has laid the foundation for many deeper work across these disciplines. Figure (7-5) provides a simple graphic illustration of integrating the data, model, and visualization capacity from forecast of features from various domains. The red circle represents prediction system for various features from a given domain, or data source, such as: finance, which would contain stock prices, foreign exchange rates, cross-currency basis; sentiments, such as: twitter or google search trends, and related sentiment classification of positive, negative, or neutral sentiments; microeconomy: 3-5 year forecasts of state, district or city features such as household income, labor markets, environment; macroeconomy: economic growth, central government finances, fiscal balances of central and state government; geopolitics: GDELT forecasts for protests, mass violence, or refugee events, and political stability rankings; satellite imagery to obtain more high frequency features of the project’s location and extract features related to the environment, as well as monthly monitoring of the project’s location through updated images. These forecasts for various countries, states, districts, and cities should be automated and occurring at scale. Given the parameters of the mega project like the location, amount, sector…etc., the available spatial data for these domains are identified. Similarly, if entities such as companies or organizations are identified in the project preference, the geolocation of those entities as well as the locations related to those entities; such as: the headquarters' location of the entity, can be used to compute these domain specific features. Lastly, forecasts for different time horizons; such as: short, medium, and long-term, are computed and can be displayed for monitoring purposes. The time-horizon of forecasts can also be automated based on: (a) taking high frequency data; for example, stock prices which can be available daily, and aggregated to monthly or annual frequency, and (b) based on the data, for example if GDP growth data is only available at best quarterly, only quarterly predictions will be available for that given feature.



short-run	tomorrow
medium-run	next month/quarter
long-run	next year

Figure 7-5. Graphic Illustration of Integrating multiple-streams of information from different domains related to mega-project location co-ordinates and multiple-time horizons

7.3. Mega-Project Application using Foresight Monitoring

Based on the above ideas of foresight monitoring for mega-project management, the findings of this research will be discussed in light of a specific case: The Luhri hydro electric dam project. The Luhri project was initiated by the World Bank in 2010 and dropped in 2015. The project was estimated to be $1.150 billion of which the World Bank committed $650 million. The project involved multiple stakeholders, including the Asian Development Bank (ADB), and SJVN Ltd was the main contractor for this project, the USAID commissioned review the environment and social impacts. Furthermore, from the government's perspective the project included stakeholders such as central government, prime minister's office, as well as various ministries and public-private entities; such as: the Power Corporation of India, and State government entities including the Chief Minister's Office, and state level regulatory bodies for Power ministry.

The hydro dam project was around the Sutlej river which is a glacial river originating from the Kailash Mansoravar in Tibet traversing through the Himalayas and Punjab plains in India, meeting the Indus river in Pakistan. The project was initially planned for a hydro dam project that involved the construction of an 86 m high concrete gravity dam that generated 775MW of power which involved building 38-kilometer long twin tunnels (9 m in diameter) that was estimated to be longest tunnel in the world, in the Indian state of Himachal Pradesh.

There are many factors that led to the failure of this project including: the lack of environmental and social impact planning, lack of foresight, and lack of accommodating potential external threats. There was an under-assessment of land requirements, for example, forest land area was underestimated by hundred hectares, and most importantly the impact on the perception of local population. The project would have impacted residents of at least 78 villages of Kullu, Mandi and Shimla district (Environmental Justice Atlas 2017) and its 38 kilometer long tunnel would have impacted the livelihoods of these villages. Protests from local villagers were growing, that led local protestors meet with investors and officials and made the World Bank withdraw from the project.

Environmentalists were worried that building the project will lead to major environmental crises like drying of river and loss of vegetation, this was supported by local communities due to the fear of loss of their “lands” and “homes”. In hindsight, the cumulative impacts would have disseminated the only stretch of free-flowing river Sutlej, flooding disaster with the vulnerability of the hilly region being exposed due to hydropower development, environmental impacts, local people’s claims. Other issues included air pollution, soil erosion, surface water pollution, groundwater pollution or depletion, large-scale disturbance of hydro and geological systems, reduced ecological / hydrological connectivity, food insecurity (crop damage), loss of landscape/aesthetic degradation, so on and so-forth.

The project was dropped but is currently still undergoing changes in design and capacity. In particular, the tunnel component that would have affected three districts of Kullu, Mandi and Shimla was dropped later (HIM DHARA 2011). Similarly, other technical changes; such as: single tunnel was made and its capacity reduced from 7775MW to around 600MW.

Nevertheless, executives on the project could have made better decisions and anticipated many of these threats based on foresight monitoring methods. Figure (7-6) shows a spatial aggregation of the project i.e. country, state, district of the project. Associated table of relevant predicted features for this time period of these different spatial aggregations are presented alongside the maps. In a similar fashion, executives can be presented daily or high-frequency monitors for factors associated with the project's location, and locations associated with assets related to the project.

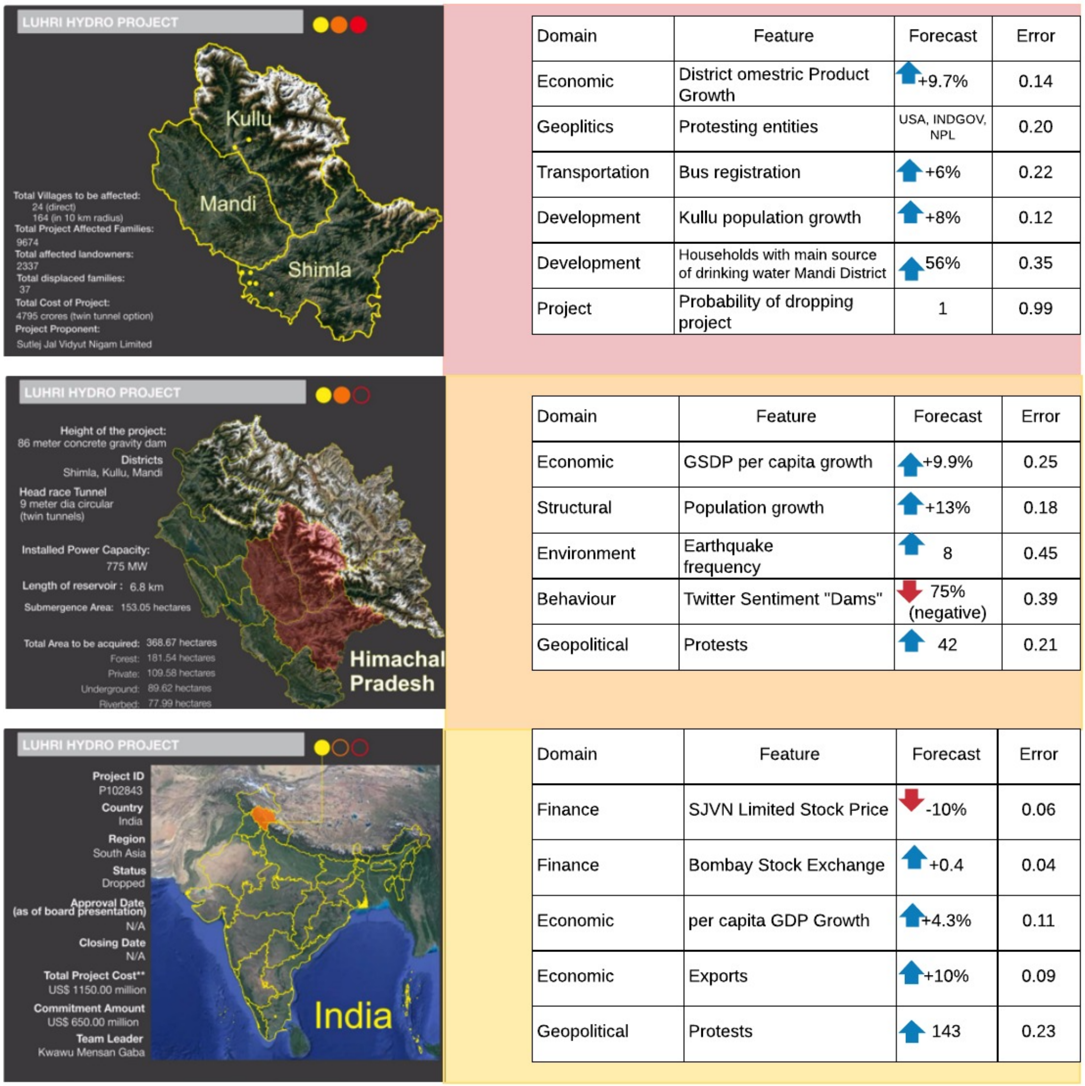
The project's model predicted that the probability of the project being "dropped" was 1. This result with high accuracy implies that there are was some missing elements in the project and country features that may have led this project to be dropped.

On the external opportunities and threats concerning domains such as economics, finance and geopolitics, the tables presented next to the map provide back tested forecasts. Now reader should note that acquiring country, state, and district level reliable time-series information for the various domains is possible but quite extensive. The complexity of each domain and entity specific model would require fine-tuning based on the proposed approaches. Hence, for illustration purposes the graph attempts to show a practical approach to monitor different time-horizon foresights.

At the country level, in the aftermath of the global financial crisis, the models showed that India would continue to have volatile economic growth in the year 2011, below 6 percent versus its expectation of 8 percent. Similarly, forecasts for public debt and exchange rate would have shown unstable macroeconomic conditions.

Another important factor that should have been monitored is the state of other stakeholders, for example the contractor. The back tested forecasts for SJVN stock prices would have shown, either in daily or monthly terms, that the financial performance of the SJVN would be dull for the coming years of 2011-13 which should raise alert flags in the executive decision-maker.

Similarly, state and district features can be incorporated to add information and reduce uncertainty. State level forecasts for state gross domestic product predicted that growth would be much slower in Himachal Pradesh than the previous years. GSDP growth in mid-2000's in Himachal Pradesh was over 12 percent and expected to decline below 12 percent; similarly district specific features, including the population projections, were pointing towards a push factor of greater density and worsening economic climate. Last, but not least, using the geopolitics database from GDELT, it would have shown that the likelihood of protests in Himachal Pradesh and specifically those districts would have been much higher than expected. In conclusion, all these back tested forecasts collectively would have provided the executives engaged with the project a good sense of vulnerabilities in the external domains of economics, finance, and geopolitics to better pre-empt threats related to the Luhri dam project and plan better to address the root causes of failure.



LUHRI HYDRO PROJECT ●●●

Total Villages to be affected:
24 (direct)
164 (in 10 km radius)
Total Project Affected Families:
9674
Total affected landowners:
2337
Total displaced families:
37
Total Cost of Project:
4795 crores (twin tunnel option)
Project Proponent:
Sutlej Jal Vidyut Nigam Limited

Domain	Feature	Forecast	Error
Economic	District omestric Product Growth	↑+9.7%	0.14
Geoplitics	Protesting entities	USA, INDGOV, NPL	0.20
Transportation	Bus registration	↑+6%	0.22
Development	Kullu population growth	↑+8%	0.12
Development	Households with main source of drinking water Mandi District	↑56%	0.35
Project	Probability of dropping project	1	0.99

LUHRI HYDRO PROJECT ●●○

Height of the project:
86 meter concrete gravity dam
Districts
Shimla, Kullu, Mandi
Head race Tunnel
9 meter dia circular
(twin tunnels)
Installed Power Capacity:
775 MW
Length of reservoir : 6.8 km
Submergence Area: 153.05 hectares
Total Area to be acquired: 368.67 hectares
Forest: 181.54 hectares
Private: 109.58 hectares
Underground: 89.62 hectares
Riverbed: 77.99 hectares

Domain	Feature	Forecast	Error
Economic	GSDP per capita growth	↑+9.9%	0.25
Structural	Population growth	↑+13%	0.18
Environment	Earthquake frequency	↑ 8	0.45
Behaviour	Twitter Sentiment "Dams"	↓ 75% (negative)	0.39
Geopolitical	Protests	↑ 42	0.21

LUHRI HYDRO PROJECT ●○○

Project ID
P102843
Country
India
Region
South Asia
Status
Dropped
Approval Date
(as of board presentation)
N/A
Closing Date
N/A
Total Project Cost**
US$ 1150.00 million
Commitment Amount
US$ 650.00 million
Team Leader
Kwawu Mensan Gaba

Domain	Feature	Forecast	Error
Finance	SJVN Limited Stock Price	↓-10%	0.06
Finance	Bombay Stock Exchange	↑+0.4	0.04
Economic	per capita GDP Growth	↑+4.3%	0.11
Economic	Exports	↑+10%	0.09
Geopolitical	Protests	↑ 143	0.23

Figure 7-6. Luhri hydro electric mega-project, foresight monitoring example for timely uncertainty adjusted decision-making in dynamic environments

Notes: The reader should note that integrating and acquiring the various spatial-temporal domain specific information is highly complex. The chart above is an illustration with approximations from the publicly available raw data.

Chapter 8: Conclusion

Megaprojects' executive managers are busy orchestrating and managing many project-related tasks. They are too busy to pay attention to external opportunities and threats that may impact their project or how their operations may impact the local, national, or global economy. Given their lack of attention, many state-of-the-art methods can be automated for accurate and reliable analysis of economics, finance, and geopolitical domains that can be used for quickly monitoring updates, errors, vulnerabilities in the spatio-temporal space as it relates to the mega-project or assets related to the mega-project. However, there is an important element of how relevant features concerning a decision-makers' domain, entity, or features of interest can be incorporated in their decision-making process. Therefore, this research presented this element and showed how the different features can be incorporated. This research provided an ensemble deep learning framework for analyzing external opportunities and threats. Furthermore, the research shows how integrating forecasts or foresights related to a specific asset or a large investment plan can bring value by bringing greater awareness to external environment for uncertainty-adjusted decision-making. The predictions in themselves do not say much, but systematic monitoring of the error, uncertainty, disorder, and variability in prediction should be given more importance for executives who wants to be successful at managing large projects by monitoring economic, financial, or geo-political mass-movements.

This study presented a novel deep learning framework where Dynamic Time Warping (DTW), Hierarchical Clustering Analysis (HCA) and long-short term memory (LSTM) are combined for application in new domains; such as: economics, finance, and geopolitics. The proposed ensemble model framework consists of a number of steps: first, processed domain knowledge is used to extract a set of technical features including dependent variable and domain-specific's measurement science to accommodate the training data; second, structured learning based on Dynamic Time Warping (DTW), to learn similarity between sequences and Hierarchical Clustering Analysis (HCA), is used to determine which features are relevant for a given prediction problem. Third, this step is used to automate a decision based on the input and structured learning from the DTW-HCA training data-set which is fed into a deep LSTM neural network for time-series predictions. The developed model can be used to inform decision makers on *t+1* (daily, monthly, quarterly, annual) predictions with great reliability. The model outperforms other similar models in its predictive accuracy. Furthermore, the study accounts for potential model-based uncertainties which provides a more robust and reliable framework for quantifying and managing uncertainty for time-series predictions.

Moreover, the model is scalable, modular and yields higher accuracy at lower computational costs compared to other methods. The elements of processing domain knowledge and structuring domain logic using machine learning techniques can be relatively computationally expensive but in the developed model they work well to understand knowledge. The time-warped distance provides valuable insights for aggregating and clustering. This approach of using machine learning to build an appropriate training data-set provides a scalable framework. For example, in order to predict the GDP growth for the USA, a totally different set of features will be required, versus predicting the GDP growth of India. Similarly, as people, we do not know whether there is co-relation or similarity between some random events, like the “sale of bubble-gum” and the price of a large financial institution. While it will remain impossible to predict, small, unseen, rare events; however, provided the data, the proposed approach helps to cluster and use machine learning methods to build appropriate training data-set to be fed into a model designed to cross-chaotic, dynamic feed-back loops retaining long and short-term memory such as the LSTM. Indeed, there are numerous methods that can be used to achieve similar results, and this framework falls in line with other frameworks that are being used.

In addition to this domain-knowledge based prediction framework, there are four other methodological innovations that were presented in this research. First, aspects related to dependencies, causality, and question of separability of different features, are dealt with MC simulations which provides the ability to learn the sensitivity factor of training features and study potential impact of shocking features. Second, ensembling methods were proposed to accommodate the information and knowledge and/or ignorance from different models. Third, methods similar to LSTM networks and ensemble models were presented to build a time-based alert system that monitors if the time-sequence forecast is crossing a threshold defined by the decision-maker. Fourth, the application of uncertainty quantification methods like MC dropout and quantile loss function provides novel approach to better handle epistemic uncertainty.

The results from different domains showed great success. The economic domain showed results for predicting GDP growth of the USA, as well as effective exchange rates with a low error and high accuracy. Similarly, detailed results from stock price indices and individual tickers were shown for day ahead predictions, and predictions for protests in the USA from the geopolitical domain were presented. The results showed a new approach to use the information from different models to take advantage of over-fitting and under-fitting models to create a crude, yet useful, approach to build good fitting models for highly noisy and stochastically random variables. The MC simulation showed promising results, there is small but enough variation in results, and examples for factors like capital formation and exports importance for US economic growth provided simple, yet practical cases. Similarly, a toy algorithm based on LSTM model to use a single time-series information to build a classifier to predict the label of time-period was developed. In this instance, the monthly labels for the US recession years showed that such monitoring and alert systems can have practical implications for decision-makers.

The application of the models is particularly valuable when different forecasts can be integrated together to be displayed on a single page, which are also referred to as foresights. There are a variety of applications for such foresight monitoring technologies. The conducted research showed a specific case of integration of foresights for the Luhri hydro electric dam project. There were many facets of the economic climate in the country, state and district level signals on environmental factors such as the deforestation, or economic predictions for income, and political threats such as protests could have provided the decision-maker awareness of opportunities and threats related to the project. Furthermore, the computational capacity to deliver large-scale foresights for planning related to an asset, which only requires less than a minute to view, can greatly improve the awareness of project managers to better understand the different risk perspectives of the counter-parties engaged in the mega-project. By incorporating multiple-time horizons and multiple time-series foresights, decision makers can be better equipped to plan and constantly monitor mega-projects. The applications have also raised important questions concerning how to display large volumes of complex, yet valuable information, and computational capacity to run multiple automated deep learning algorithms for different time periods, locations, features, and entities.

8.1. Research Contributions

The conducted research contributed to the state of knowledge within and across multiple domains, including reliability engineering, systems engineering, computer science, economics, finance, geopolitics, project management, and decision-making. The general contribution of this research has been to provide a new ensemble deep learning model framework that can be used as monitoring technologies. The specific contribution was the applications of the ensemble deep learning framework across different domains of economics, finance, and geo-political forecasting to aid the executive decision-making process concerning risk management of mega-projects. The multi-disciplinary framework provided the following new contributions to the current state of knowledge:

1- This research showed promising new application of deep learning to new domains such as economics, finance, and geopolitics. The research has the stage for operationalizing the findings for the investment and policy related decision-making community. The methods and results presented a reliable data-driven approach for monitoring and forecasting at scale for numerous components in a complex dynamic system; and had the ability to eliminate signal from the noise in big data by accommodating domain knowledge.

2- The structured logic approach is close to the optimal choice for training data and yields higher accuracy at relatively low computational costs and reduced complexity. The results show superior performance of abstracting domain-knowledge and prediction accuracy when compared to traditional statistical practices and industry standards, when available.

3- The model architectures have significance in variety of other domains that deal with real-time signal about complex systems; for example, reliability engineering and other engineering domains using SCADA. The modularity and flexibility of the model's approaches offer many variant possibilities, especially with ensembling different models using an ensemble weighting scheme as a simple powerful method to use knowledge gained from different models.

4- The novel controlled experiment scenario for neural networks, provided a framework to go beyond just black box. For example, MC simulation cases provided an experimental setting to have more controlled setting for deep LSTM networks. Similarly, the TLA approach showed simple cases for building alert systems for time-series information based on decision-makers' threshold preference. The insightful foresights and approaches to integrate accurate forecasts will re-enforce the use on such decision-making methods for investment and policy related resource-allocation problems.

5- Approaches to standardizations of data and models for geopolitical, economic, and financial domain-knowledge extraction were provided. The data-model system has applications in many other domains where time-series information about various components of a system are available.

6- Novel application of uncertainty quantification (UQ) and model reliability (MR) matrices directly aid decision-making capabilities across communities of practice including financial trading, economic policy, financial management, risk management, and resource-allocation related to executive decision-making in LEP's. The modular nature of the system control architecture should provide ability use state of the art UQ and MR methods in a scalable manner.

7- Application of what it called foresight monitoring technologies i.e. integrating of predictions from different domains and time-horizons were examined which showed how planning and real-time foresight monitoring of mega-projects can help bring executive decision-makers greater awareness of external opportunities and threats.

8.2. Limitations and Suggestions for Future Research

In spite of the major contributions of the conducted research, there are still some limitations and boundary conditions to this research that provide opportunities for future research. First, the future forecasts deduced from this research are dependent on past information; hence, if there are any inaccuracies in the latter data, the forecasts' quality will be affected. Second, this research discussed briefly the attributes related to separability, dependencies, causality, in the training data-set through the MC simulations case; thus, examining these attributes in-depth might impact the quality of the forecasts and to disentangle causal factors in a more controlled setting in neural networks. Third, measurement errors of the raw data are out of the scope of this research. The focus of this research has been on reliable data that may have uncertainties associated. Fourth, this research showed a generally applicable and reliable *t+1* (that could be either daily, monthly, quarterly or annual) forecasts while longer time-sequence predictions would require other ensembling options. Fifth, the probability of unseen small probability events that are not captured in the data are not in the scope of this research focus.

At the same time, this research has opened the door for many new areas of research. First, there are many deeper applications within the domains discussed in the research. In particular, a richer framework can be developed on evaluating the domain-specific validity of different ensembling methods. These applications could include parametric models, reinforcement learning, voting classifier etc. Second, more insightful examples on appropriate and reliable thresholds can be incorporated in the MC simulation to provide meaningful results related to the most important feature that pushes or pulls the dependent variable and studies on ranking feature sensitivity factor could be promising. A deeper study is required for an experimental study simulation in the confines of neural networks. Third, questions related to data reliability, such as do we trust some measures from countries like Afghanistan, or other measures from a given country are open questions. In the discipline of big data architecture, indeed the research has raised questions on standards for a level of trust, validity of “good” data versus “inaccurate” data. Fourth, scaling the database through a study on spatio-temporal knowledge base based on micro and macro statistical knowledge with open-source architecture could have a big public impact. Fifth, there is scope to improve the standardization of the model’s architecture and data. In this light, aspects of computer graphic visualizations incorporating the cognitive capabilities of human beings to best inform their decision making concerning large investments, assets, and projects are high-value research projects to pursue. For example, how could vast amounts of complex data and predictions from spatio-temporal be automated for easy access on one page is a big challenge. In this thought, attributes of Natural Language Processing (NLP) can also be used for converting such information to questions and answers systems. Most importantly this research has also raised deeper questions on system architecture to run such computation on-line that integrating multiple forecasts in real-time, from multiple geolocations, entities, domains, and time periods. The deployment strategy for such a system architecture requires deeper study.

In addition, from the perspective of multiple stakeholders, it would be good to understand how these monitoring technologies can directly aid their decision-making process. This could be explored in the future research with an experimental design setting to incorporate feedback from executives' expert opinion to design an interface for human-machine interaction. Moreover, future research can explore how Convolutional Neural Networks (CNNs) approach can be used in exploring the idea of whether entities; like countries, can be monitored based on building a standard daily image, or heat-map, and then extract features of this daily-moving country image. In this approach, CNN's can be used in the same manner as used for scene mapping in video and image recognition tasks. Another area of future research is going beyond *t+1* predictions to longer time sequence predictions. Many new models can be explored for ensembling or stand-alone, such as GAN's and Reinforcement learning for important feature extraction, to improve prospects for long term sequence predictions.

Attributes of aleatory uncertainty also requires a special focus in these domains of interest. For example, private sector executives and policymakers associated with a mega-project are interested in potential local or national economic or political events uncertainties that could impact the execution of the project. The DTW-HCA ensemble allows a mechanical case-specific basis to cluster similarity of trends and provide potential recommendations of related asset forecasts. However, a structured approach to combine aleatory uncertainty and epistemic uncertainty in our domains and decision-making cases of interest are areas of future research.

There are also many important issues concerning causality that require deeper analysis. For example, the interaction between classes in different clusters, can be separated to provide a better fit to ground-truth data. In traditional machine learning, such domain knowledge is neither processed in a structured manner nor provided unsupervised methods in machine learning are used to build an appropriately “similar” training data-set for the prediction or classification problem based on the variable of interest. However, the provided methods for structuring knowledge could be useful for building causal reasoning models and this knowledge could also be used to provide recommendations for given policy and investment challenges but requires deeper study. Nevertheless, this research has set the stage for standardization of data, models, and visualization libraries for economic, finance, and geopolitical domains of interest. Furthermore, it has presented the findings that can make a direct impact on decision-making concerning mega-project management, as well as policy, and investment decision.

The study also provides interesting new research areas to avoid human error in financial trading and economic policy related decision-making. In this context, there is a gap in the human-computer interaction (HCI) and human reliability analysis (HRA) literature as to how human errors can manifest with the proliferation of accurate and reliable artificial intelligence (AI) based predictions. Forecasts clearly have an explicit psychological impact on the observer (or operator), thereby, well-founded psychology theories including inductive reasoning, anchor heuristics, reinforcement expectancy theory, and cognitive dissonance could have significant impact to enhance prognostic decision-making. Many of the semi-automated and inductive reasoning tasks are transferred from human cognitive load to the machine conducting pattern matching, similarity, clustering, reliability, uncertainty etc. After thorough testing in a laboratory setting with decision-makers, standards and guidelines could be developed to better track human-error with real-time computer systems and optimize resource allocation decisions.

Finally, in the wave of thinking from the concepts such as Black Swan and Chaos Theory, resilient systems thrive greater disorder and idiosyncrasies, fragile systems break under stress. From a mathematical perspective, it is impossible to come up with probability of rare (or unseen) small event, especially in the fat tail domain (known as the problem of induction). In Arabic, predictions are called “prophecies”, and indeed, forecasting is like prophesying which has many psychological aspects. Any economic number from oil prices to GDP growth forecasts from most forecasters have been incorrect; however, people still pay attention to it. The methods, results, and applications offered in this research could either end up adding or reducing redundancy to the system, where technology itself should be a domain of risk. There are heuristics and simple solutions that have implications for human judgement that require a system for orchestration of information and knowledge between man and machine, and vice-versa, “externalities” that this research has identified.

Appendices

Appendix A. Definitions

Information can be defined as sensed objects, things, places, processes, and information and knowledge communicated by language and multimedia. Information can be viewed as preprocessed input to our intellect system of cognition, and knowledge acquisition and creation. Information can lead to knowledge through investigation, study and reflection.

Knowledge is defined in the context of humankind, evolution, language, and communication methods, as well as social and economic dialectical processes; knowledge cannot be removed from them. As a result, knowledge always reflects the imperfect and evolutionary nature of humans, which can be attributed to their reliance on their sense for information acquisition; their dialectical processes; and their mind for extrapolation, creativity, reflection, and imagination, with associated biases as a result of preconceived notions due to time asymmetry, specialization, and other factors. An important dimension in defining the state of knowledge and truth about a system is nonknowledge or knowledge deficiency or ignorance due to many factors and reasons including information deficiency, that is, uncertainty.

Risk is a measure of the potential loss occurred due to natural or human activities. Potential losses are the adverse consequences of such activities in form of loss of human life, adverse health effects, loss of property, and damage to the natural environment (Ayyub 2008, Modarres 2017). Risk is defined as a consequence of action taken in spite of uncertainty. In 2009, the ISO provided a broadly applicable definition of risk in its standard (ISO 2009a) as the "effect of uncertainty on objectives" in order to cover following considerations as noted in the standard:

- An effect is a deviation from the expected that can be positive and/or negative effect
- Objectives can be different aspects such as financial, health and safety, and environmental goals, and can apply at different levels, such as strategic, organization-wide, project, product, and process.
- Risk is often expressed in terms of a combination of consequences of an event, including changes in circumstances, and the associated likelihood of occurrence as provided in the commonly used definition.

Risk analysis is the process of characterizing, managing and informing others about existence, nature, magnitude, prevalence, contributing factors, and uncertainties of the potential losses. In engineering systems, the loss may be external to the system, caused by the system to one or more recipients (e.g., human, organization, economic assets, and environment). The loss may be internal to the system i.e. only damaging the system itself. In a nuclear power plant the loss can be damage to the plant due to partial melting of the reactor core, or it can be release of radioactivity into the environment by the power plant. The former case is a risk internal to the system, and the latter case represents a loss caused by the system (the nuclear plant) to the environment.

Risk communication is the activity of transferring, exchanging or sharing data, information and knowledge about risk, risk assessment results and risk management approach between the decision makers, analysts and the rest of stakeholders. The information can relate to the existence, form, likelihood, frequency, severity, acceptability, controllability, or other aspects of risk. Communicating the nature of the risk and the nature of the benefits is important in risk communication.

Uncertainty in engineering and design is commonly defined as knowledge incompleteness due to inherence deficiencies in acquired knowledge. IT can also be used to characterize the state of a system as being unsettled or in doubt, such as the uncertainty of the outcome. Uncertainty is an important dimension in the analysis of risks. In this case, uncertainty can be present in the definition of the hazard threats and threat scenarios, the asset vulnerabilities, and their magnitudes, failure consequence types and magnitudes, prediction models, underlying assumptions, effectiveness of counter measures and consequence mitigation strategies, decision metrics, and appropriateness of the decision criteria. Uncertainty is defined as potential, unpredictable, and uncontrollable outcome.

Risk Management involves the coordinated activities to direct and control an organization with regard to risk. Risk management is a process by which system operators, managers, and owners make safety decisions and regulatory changes, and choose different system configurations based on the data generated in the risk assessment. Risk management involves using information from the risk assessment stage to make educated decisions about system safety. Risk treatment and control to risk assessment defines risk management. Risk treatment and control include risk prevention, avoidance, transfer, countermeasures, consequence mitigation, and so on. The objective of risk management is to assure uncertainty does not deflect the endeavor from the business goals (Antunes and Gonzalez 2015).

Resilience is “the ability to prepare for and adapt to changing conditions and withstand and recover rapidly from disruptions. Resilience includes the ability to withstand and recover from deliberate attacks, accidents, or naturally occurring threats or incidents” (PPD-21 2013, Ayyub 2014).

Cognition can be defined as the mental processes of receiving and processing information for knowledge creation and behavioral actions. Cognitive science is the interdisciplinary study of mind and intelligence (Stillings 1995). Cognitive science deals with many disciplines including philosophy, psychology, artificial intelligence, neuroscience, linguistics, and anthropology. The intellectual origins of cognitive science started in the mid-1950’s when researchers in several fields began to develop theories on how the mind works based on complex representations and computational procedures. Cognitive science is based on a central hypothesis that thinking can best be understood in terms of representational structures in the mind and computational procedures that operate on those structures (Johnson-Laird 1988).

Awareness is the ability to directly know and perceive, to feel, or to be cognizant of events. More broadly, it is the state of being conscious of something.

Consciousness is the state or quality of awareness, or, of being aware of an external object or something within oneself (Webster Dictionary, van Gulick 2004). It has been defined variously in terms of sentience, awareness, qualia, subjectivity, the ability to experience or to feel, wakefulness, having a sense of selfhood or soul, the fact that there is something "that it is like" to "have" or "be" it, and the executive control system of the mind (Farthing 1992).

Threat is the potential intent to cause harm or damage on, with, or through a system by exploiting its vulnerabilities. Threats can be associated with intentional human actions as provided in table below, that lists examples under several threat types including chemical, biological, cyber, etc… Threat is the potential intent to cause harm or damage on, with, or through a system by exploiting its vulnerabilities (Ayyub 2002).

Opportunities has been defined as "*a process to identify business and community development opportunities that could be implemented to sustain or improve the local economy*" (Opportunity Management Facilitators Guide 2012). Opportunity management is a collaborative approach for economic and business development. The process focuses on tangible outcomes (Hilson and Murray-Webster 2004).

System is defined as "*a regularly interacting or interdependent group of items forming a unified whole*", such as solar system, school system, financial system or system of highways. It follows from this definition that the term system stands in general for a *set of things* and a *relation among the things*.

Metasystems are used for the purpose of describing changes within a given support set. The metasystem consists of a set of systems defined at some lower knowledge level and some support-independent relation. Referred to as a replacement procedure, this relation defines the changes in the lower-level systems.

Ontology is a formal naming and definition of the types, properties, and interrelationships of the entities that really or fundamentally exist for a particular domain of discourse. It is thus a practical application of philosophical ontology, with a taxonomy. A **knowledge base** is an object model (often called an ontology in artificial intelligence literature) with classes, subclasses, and instances. An ontology compartmentalizes the variables needed for some set of computations and establishes the relationships between them.

Complex systems is a system composed of many components which may interact with each other. In many cases it is useful to represent such a system as a network where the nodes represent the components and the links their interactions. Examples of complex systems are Earth's global climate, organisms, the human brain, social and economic organizations (like cities), an ecosystem, a living cell, and ultimately the entire universe.

Dynamic complex systems have distinct properties that arise from these relationships, such as nonlinearity, emergence, spontaneous order, adaptation, and feedback loops, among others. Dynamical systems theory is an area of mathematics used to describe the behavior of the complex dynamical systems, usually by employing differential equations or difference equations. When differential equations are employed, the theory is called *continuous dynamical systems*.

Project management is the planning, organizing and controlling of a firm's resources to achieve reasonably short-term goals that have been established to complete specific targets and objectives (Field and Keller 1998). It is usually management driven and focuses on setting targets, problem solving and obtaining results. The purpose of project management is to act as a change agent, delivering a change to the status quo of a project, and achieving this in a controlled and managed way (Hillson 2004).

Operational risks are associated with several sources including out-of-control operations risks that could occur when a corporate branch undertakes significant risk exposure that is not accounted for by corporate headquarters, leading potentially to its collapse, for example, the British Barings Bank, which collapsed in 1995 primarily as a result of its failure to control the market exposure create within a small overseas branch of the bank. Another risk source in this category is **liquidity risk**, in which a corporation requires more funding than it can arrange. Also, such risks could include, money transfer risks and agreement breaches. Operational risks include model risks, which are associated with the models and underlying assumptions used to value financial instruments and cash flows incorrectly.

Reputation Risks are the loss of business attributable to a decline in a corporation's reputation can pose another risk source. This risk source can affect a company's credit rating, ability to maintain clients, workforce, and so on. This risk source usually occurs at a slow attrition rate. It can be an outcome of poor management decisions, business practices, and high-profile failures or accidents.

System Functional Component Definitions

Entities are the individual units or modules how risk can be transferred. The risks facing a decision-maker can be influenced by various entities. For the purpose of this proposal, the *entities* are defined as - *countries, companies, megacities, or actors*. Figure A2 shows a simple entity map – how they influence each other in a spatial context. The losses facing an entity can be transferred between entities and (or) domains. For example, a "too big to fail" entity like a company such as "Lehman Brothers" could trigger a country-wide or global catastrophe in time-stamp "*September, 2008*". An aspect of the complexity is the multi-dimensional nature of risks faced by different entities. There are both systemic and idiosyncratic rare events that constitute the risk profile of a country, company, or a project. Figure (A-1) illustrates the relation between entities and different spatial scales of aggregation.

Risk domains are the aggregate data source where the signal of risk for a given entity comes from. The risks facing entities are multi-dimensional in nature. These risks can be organized in a hierarchical cluster, labeled by experts or in more complex forms. For simplicity, the architecture defines *risk domains* as aggregate sectors such as economic or political as depicted in Figure (A-3). More specifically, in the object model (or ontology) with risk domains refer to the aggregate classes.

Risk classes are dis-aggregated risk domains that are aggregated by a collection of risk indicators. Risk class are dis-aggregated clusters of topics such as asset price bubble under the risk domain economic and market risks. Figure (A-4) depicts the relationship between risk domains, classes, and indicators. Risk classes are the subclasses of risk domains.

Risk indicators Within each risk class, there are *indicators* such as GDP growth in macroeconomic (risk class) under the macro-economic and market (risk domain). For example, at a country level the loss can be macroeconomic or geopolitical, for companies the loss could be stock prices or brand/reputation equity, and for cities it could be a bankruptcy crisis or high-income inequality. Many aspects of ontology and global schema discuss details of risk domains. Risk indicator provide instances at a given point of time for a given risk class.

Ensemble Models are predictive modeling and other types of data analytics techniques that provide a single model based on one data sample. These models can have biases, high variability or inaccuracies that affect the reliability of analytical findings. In statistics and machine learning, ensemble methods use multiple learning algorithms to obtain better predictive performance than could be obtained from any of the constituent learning algorithms alone (Opitz and Maclin 1999, Polikar 2006, Rokach 2010). One common example of ensemble modeling is a random forest model. This approach to data mining leverages multiple decision trees, a type of analytical model that's designed to predict outcomes based on different variables and rules. *Ensemble forecasting* is a specific method used in numerical weather prediction. Instead of making a single forecast of the most likely weather, a set (or ensemble) of forecasts are produced. This set of forecasts aims to give an indication of the range of possible future states of the atmosphere. Ensemble forecasting is a form of Monte Carlo analysis. Ensemble models comprise a finite set of diverse predictive models whose combined output is expected to yield an improved predictive performance as compared to an individual model.

System of Ensemble Models is the process by which multiple models, such as classifiers or experts, are strategically generated and combined to solve a particular computational intelligence problem. Ensemble learning is primarily used to improve the (classification, prediction, function approximation, etc.) performance of a model, or reduce the likelihood of an unfortunate selection of a poor one. There are new method for learning ensembles of process-based modeling paradigm employing domain-specific knowledge to automatically learn models of dynamic systems from time-series observational data (Simidjievski et al. 2015). For example, the Global Ensemble Forecast System (GEFS), previously known as the GFS Global Ensemble (GENS), is a weather forecast model made up of 21 separate forecasts, or ensemble members. The National Centers for Environmental Prediction (NCEP) started the

GEFS to address the nature of uncertainty in weather observations, which is used to initialize weather forecast models.

Planning Horizon is the time horizon of prediction (or forecast) that a decision maker is interested in. The *planning horizon* provides a system of forecasts for individual entities-risk indicators based on the planning horizon of the decision-maker: short, medium, or long term. For example, stock prices for Goldman Sachs can be predicted for a closing day price strategy every day or every month, or long-term international investor would be interested in the GDP growth forecast for a country in the next 5 years and factors influencing a given outcome. Organization are interested to monitor both short term and long-term fluctuations. Many organizations monitor global risks on a daily basis – while others are interested in 3-5 year or longer planning horizon. Figure (A-3) shows an abstraction of different time-horizons.

Mega-projects occur in various sectors such as infrastructure, energy, aerospace, disaster clean up, space etc. Mega-projects can include data management and storage of large-scale websites such as Google or Amazon. Mega-projects can also be scientific projects such as software and big data components of the CERN Large Hadron Collider (LHC) or the International Space Station (ISS).

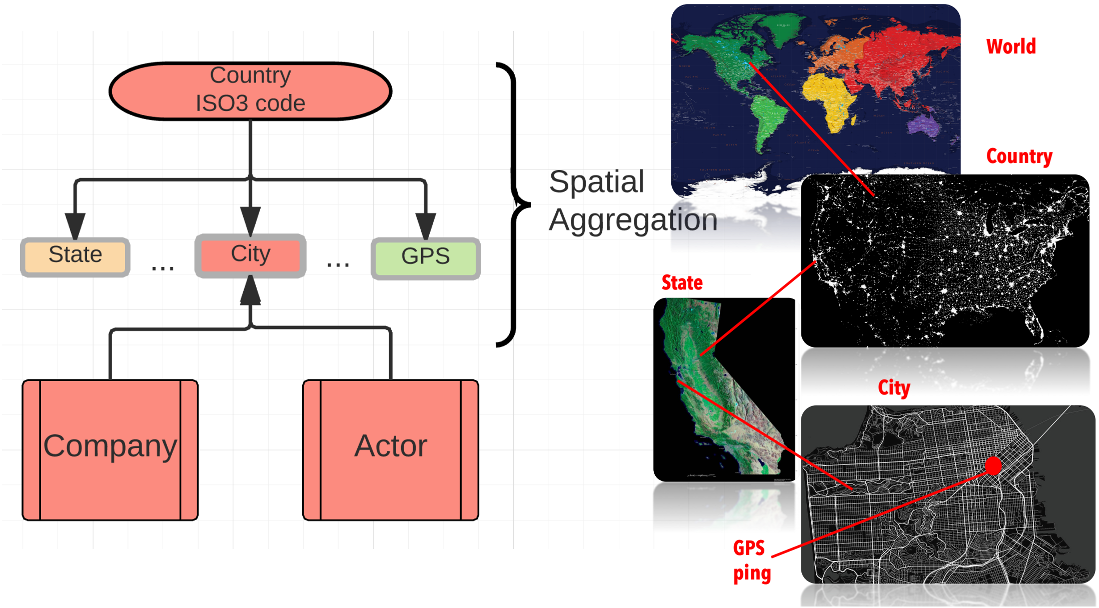


Figure A-1. Spatial Aggregation and Entities (Countries, Cities, Companies, Actors)

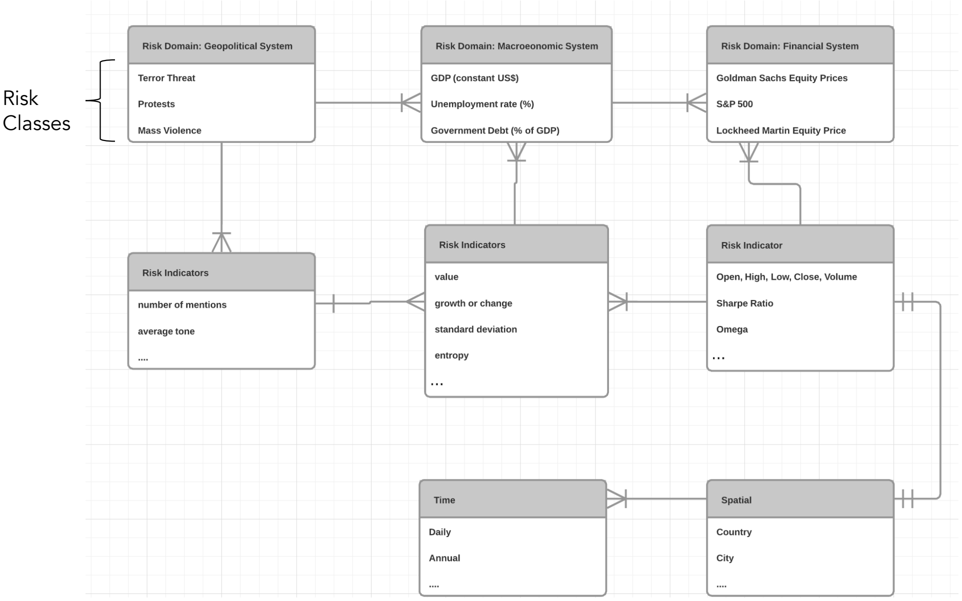


Figure A-2. Risk Domains, Risk Classes (Sub-Classes), and Risk Indicators (Instance)

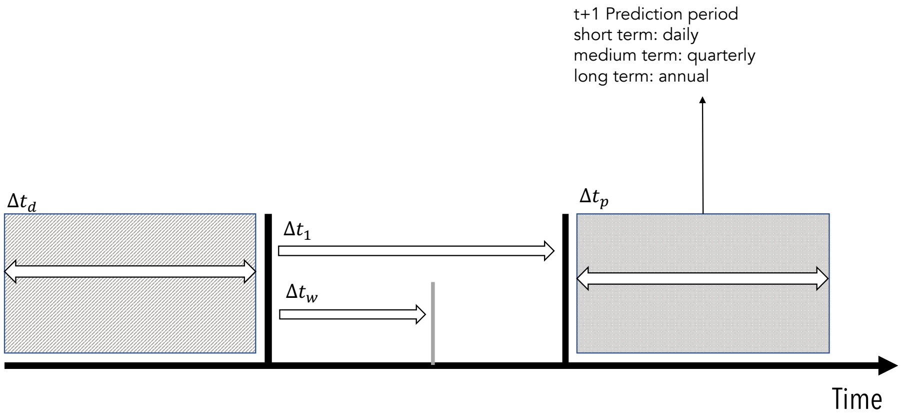


Figure A-3. Prediction and Planning Horizons.

Δt_d: time-series sequence of training examples; Δt_w: warning time; Δt_p: prediction sequence. Note that *t* is current time. Short-Medium-Long Planning Horizon are determined by prediction sequence window.

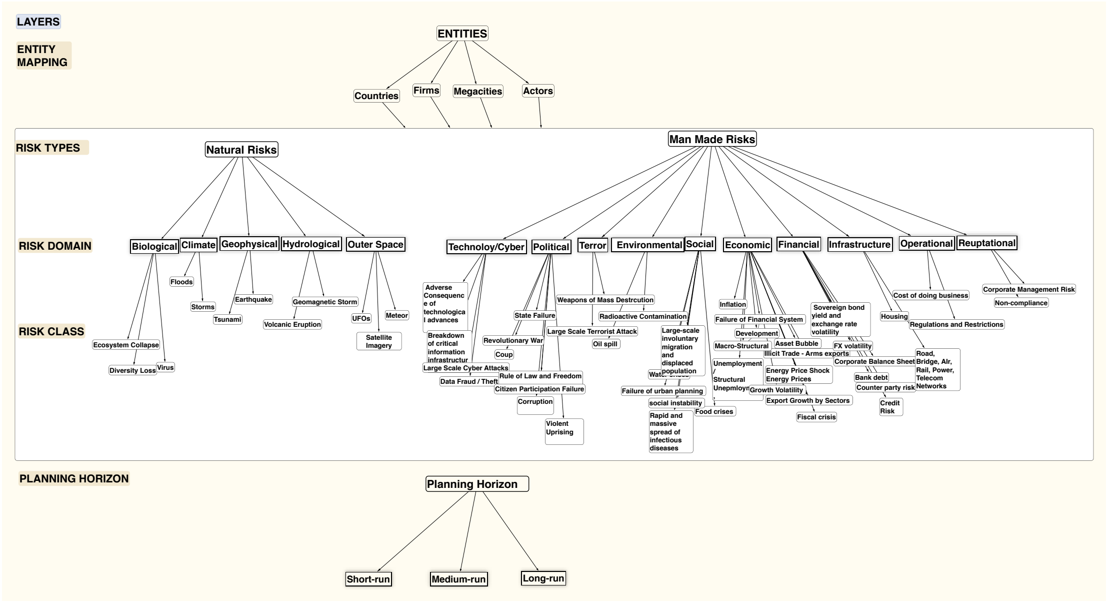


Figure A-4. Common Definitions of Entities, Risk Domains, Risk Classes, and Planning Horizons

Appendix B. System Control Architecture

The rest of this sub-section presents the abstraction of the problem in a high-level. The problems that are of interest are extremely complex in nature. The rest of this section provides some neat engineering abstractions for viewing the complexity of model-building at scale in domains of interest such as geopolitics, finance or economics. This sub-section offers ways to simplify the complexity view of model-building.

Knowledge Base Layer

Knowledge bases (KB) store millions of facts about the world, such as information about people, places and things (generically referred to as entities). Many large-scale KBs have been constructed. Prominent academic projects include YAGO (Suchanek et al. 2004), NELL (Carlson et al. 2010), DBpedia (Auer et al. 2007), and Elementary/ DeepDive (Niu et al. 2012). Commercial projects include Microsoft (AKBC-WEKEX 2012), Google (Angeli and Manning 2012), Facebook (Auer et al. 2007), Walmart (Deshpande et al, 2013), and others. The state-of-the-art formalism in knowledge representation is currently the Web Ontology Language OWL (Staab and Studer 2004). Its most expressive variant, OWL-full, can express properties of relations, but is undecidable. The weaker variants of OWL, OWLlite and OWL-DL, cannot express relations between facts.

YAGO ontology provides an interesting case. The YAGO model uses the same knowledge representation as RDFS: All objects (e.g. cities, people, even URLs) are represented as entities in the YAGO model. Two entities can stand in a relation. (Suchanek et al. 2004). An Informal description for things such as Numbers, dates, strings and other literals are represented as entities as well. This means that they can stand in relations to other entities, Words are entities. Classes are also entities. Relations are entities as well.

Rwanda EXPERIENCED Banking Crisis
Rwanda EXPERIENCED Currency Crisis in 1991
US MEANS United States
China TYPE Country
Country SUBCLASSOF region
Protest SUBCLASSOF Politics
GDPgrowth SUBCLASSOF Macroconomics

Knowledge Vault (KV) is an interesting project worth mentioning. KV was a Google Project that contained three major components. First, system that **extract** triples from a huge number of Web sources. Each extractor assigns a confidence score to an extracted triple, representing uncertainty about the identity of the relation and its corresponding arguments. The second, system learn the **graph based prior** probability of each possible triple, based on triples stored in an existing KB. The third system does **knowledge fusion**. This system computes the probability of a triple being true, based on agreement between different extractors and priors (Dong et al. 2014).

In both cases of KV and YAGO, and many other they are based on natural language programming (NLP) related entity extractions. In this manuscript, the instance of concern are continuous random variables at various time-series types and complex hierarchical structure.

There is essentially a three-step process for the KB construction. Raw data from various information sources are ingest using Django-Docker Container. The Django-Docker environment provides a standardization of data ingestion in the **source tables**. The source tables are the raw data extraction at scale from various sources such as GDELT, IMF, USGS etc. The source table from individual sources are taken as an input to create **secondary tables**. The secondary tables integrate the individual source using a global schema, ontology, and data dictionary. The secondary tables provide a single end-point for accessing the multiple information sources through a single import based on analysts' data preferences. Figure (B-1) and (B-2) provide an abstraction of the 3d data-frame.

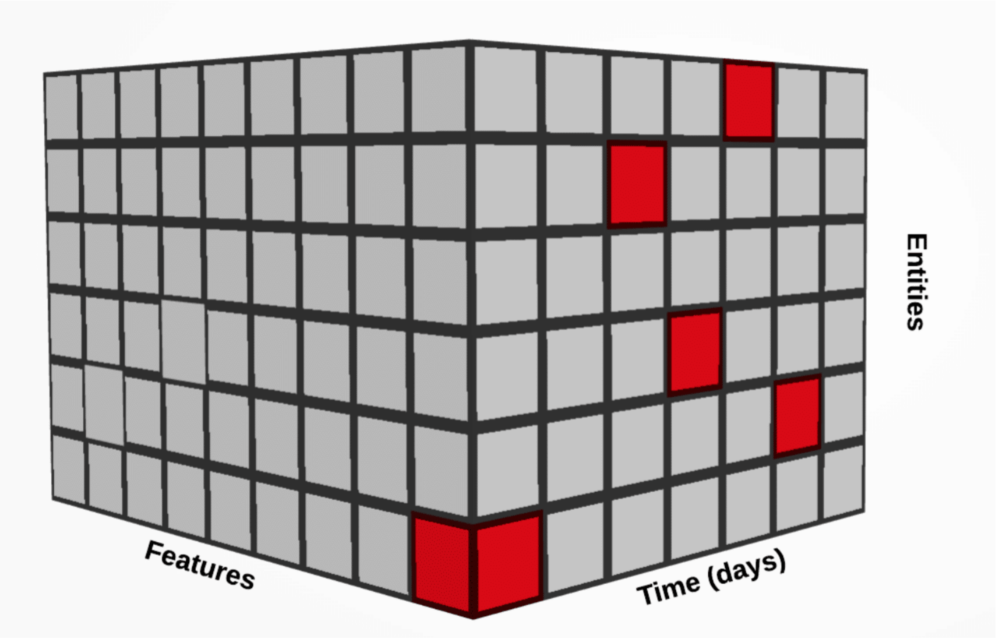


Figure B-1. 3d Database

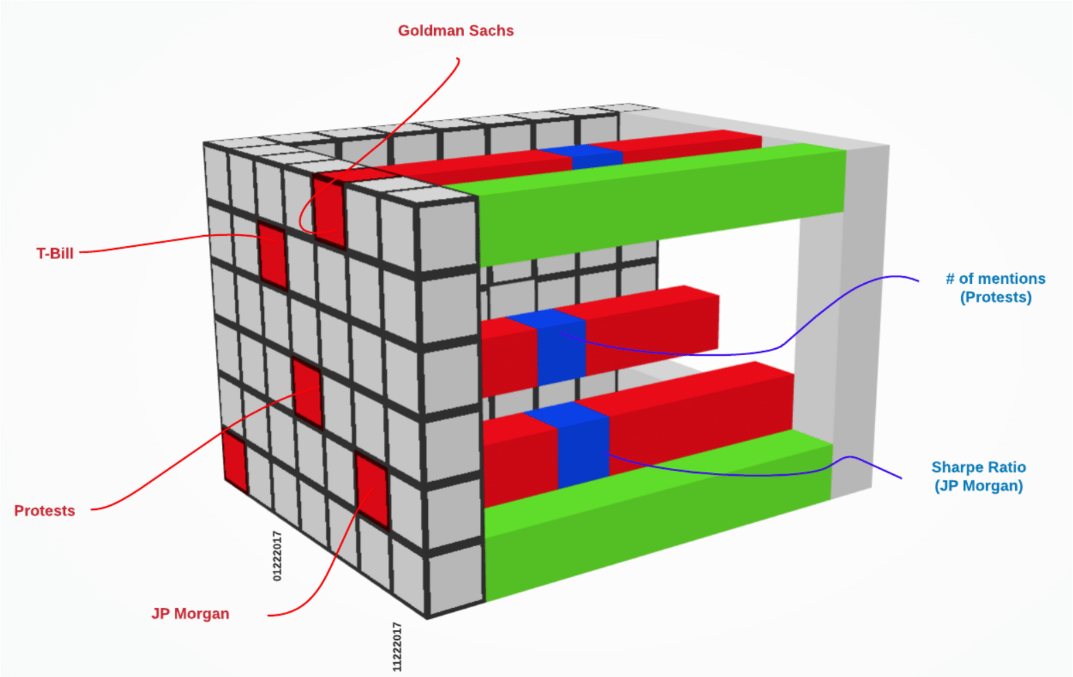


Figure B-2. Abstraction of entity, time, and features

Intelligence Layer

In order to develop a control system for cognition of aggregate location activity, this activity has to be suitably defined depending on its nature and methods of control using a hierarchical control system (Abraham et al. 1989, Ayyub and Hassan 1992, 1992b, 1992c). The hierarchical system classification enables the decomposition of risk domains into risk classes and risk instances for structured analysis. System of countries, cities, and companies exhibit patterns of organized complexity. **Organized complexity** can be observed in a system that involves non-linear differential equations with a lot of interactions among a large number of components and variables that define the system (Weaver 1948).

Organized complexity in nature another interest aspect of complexity in that it can be decomposed into an underlying repeated unit (Flake 1998). For example, economic markets they defy prediction, the pattern recognition capabilities of any of the vertebrates, the human immune system's response to viral and bacterial attack, and the evolution of life on our planet are emergent in that they contain simple units that, when combined for a more complex whole. They are examples of the whole of the system being greater than the sum of greater parts, which is a fair definition of holism – the very opposite of reductionism. They are similar to an ant colony. Although a single ant exhibits a simple behavior that includes a small of task, depending its case, such as foraging for food, caring for the queen's brood, tending to the upkeep of the nest, defending against enemies, or, in the case of the queen, lay eggs, the behavior of the ant colony as a whole is very complex. The ant colony includes millions of workers that can sweep whole regions clean of animal life, and the fungus-growing ants that collect vegetable matter as food for symbiotic fungi and then harvest a portion of the fungi as food for the colony. The physical structure of the colony that ants build often contains thousands of passageways and appears mazelike to human eyes but are easily navigated by inhabitants. The point herein is that an ant colony is more than just a bunch of ants. An organized complexity exists that is challenging to scientists. Knowing how each case in an ant species behaves would not enable a scientist to magically infer that ant colonies possess so many sophisticated patterns of behavior.

Another dimension of this complexity is that agents that exist on one level of understanding are very different from agents on another level; for example, cells are not organs, organs are not animals, animals are not species. The interaction on one level of understanding are often very similar to the interaction on other levels. **Self-similarity** might enhance our understanding of complexity and might help us to unravel complexities associated with predicting the stock market and the weather. The complexity is also **self-organization**, such as the collectives of ant colonies, human brains, and economic markets self-organizing to create enormously complex behavior that is much richer than the behavior of the individual component units. The complexity also exhibits evolution, learning and the adaption found in social systems.

The following system control architecture is inspired cognitive elements of state of the art research and development in Robotics and driverless cars space. In particular, the automotive industry defined a scale of autonomy from Levels 0 to 5, with Level 0 meaning complete manual control of the vehicle and Level 5 meaning that no human driver need be in the vehicle at all. Levels 1 and 2 the vehicle has various automated functions. Of interest is recently proposed **Level 6 autonomy** vehicle that has two “minds”: an internal Mind 1 capable of Level 5 autonomy without external help, and an external Mind 2 which will allow the vehicle to have greater cognition and functionality module (Ayyub et al. 2017). In the ISO26262 functional safety standard a definition of **functional architecture** is presented (ISO 26262:2011). The standard defines a functional concept as, “*specification of the intended functions and their interactions necessary to achieve the desired behavior*”. A functional architecture refers to logical decomposition of the system into components and subcomponents, as well as the data-flows between them. It does so without reference or prejudice to the actual technical implementation of the architectural elements in terms of hardware and software. An analogous term to functional architecture is 'functional view' of the architecture description. This term is recommended by ISO 42010 (ISO 42010:2011) and pertain to the architectural description of software intensive systems, from a functional viewpoint. The combination is often referred to as **functional architecture view (FAV)**. The FAV closely corresponds to the functional. view of the system software architecture, since autonomous systems are highly software intensive. In the simplest of senses, an autonomous driving system may be thought of as a "cognitive driving intelligence" layered on top of a basic "vehicle platform". The cognitive intelligence is responsible for perceiving the environment, generating a feasible motion trajectory through the environment, and manipulating the vehicle platform in order to achieve the desired motion. In this view, the FAV motion control of driverless are split into three attributes: perception of external environment, decision and control of vehicle motion, and finally vehicle platform manipulation (Behere and Törngren 2015).

In similar notion, the autonomous opportunity and threat functional components can be organized in the following manner. The first module is the Perception. Sensing means gathering data on physical variables using sensors, while perception refers to the semantics (interpretation and "understanding") of that data in terms of high level concepts relevant to the task being undertaken. As such, sensing is just one part of an overall perception system. Data fusion component perform a task known as map matching, wherein data objects such as entities are referenced to the map's coordinate system. Localization component is responsible for determining the location of individual risk instance with respect to a global map of earlier instances, with needed accuracy. The clustering and distance measures going forward are used for mapping similarity on time-sequence as a 'semantic' embedding. This practice is not common in for time-series information readily. Lastly, the world model component holds the state of the global environment as perceived by the ego. A graphic illustration is provided in figure (B-3).

Decision and control module provide trajectory generation and prediction for various time-series sequence, adding the knowledge of ensemble learning, followed by diagnostic and fault management as part of any standard model. Reactive control components are used for immediate (or "reflex") responses to unanticipated stimuli from the environment. For example, simulating the impact of infrastructure building on poverty reduction, or other shock-stress testing variables within the confines of the decision and control hierarchy of ensemble models. The actions of the agent can be defined by the agent manipulation module.

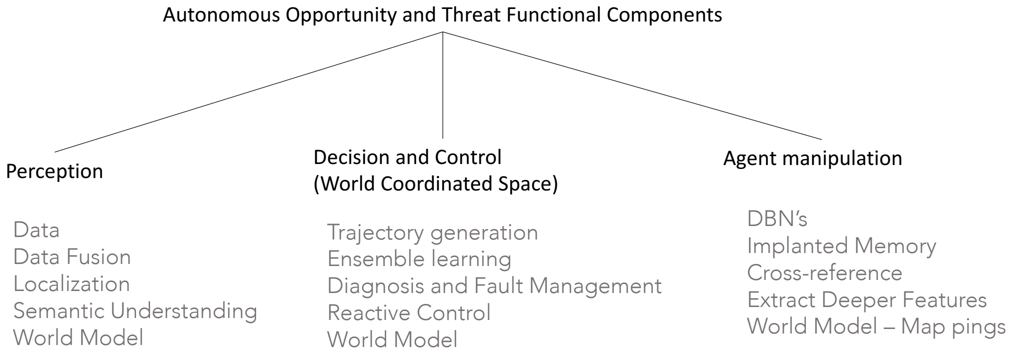


Figure B-3. Autonomous Opportunity and Threat System Functional Components
Source: Inspired by Behere and Törngren 2015.

A multilevel structure, starting with programs to business units to subsidiaries to the entire enterprise as a system with self-similarity, ensures consistency and permits risk aggregation and segregation for examining risk profiles by program, unit, subsidiary, and the entire enterprise using several formats necessary to inform decision makers at various levels. Designing for self-similarity as illustrated in figure (B-4) would drive consistency, offer simplicity in representing a complex framework, and enhance acceptance and embracement of a change of an organizational culture. Such an organization-wide undertaking might require structural changes to the enterprise's organization, such as the formation of a risk management executive committee, appointing a chief risk officer, and creating risk function at appropriate organizational levels.

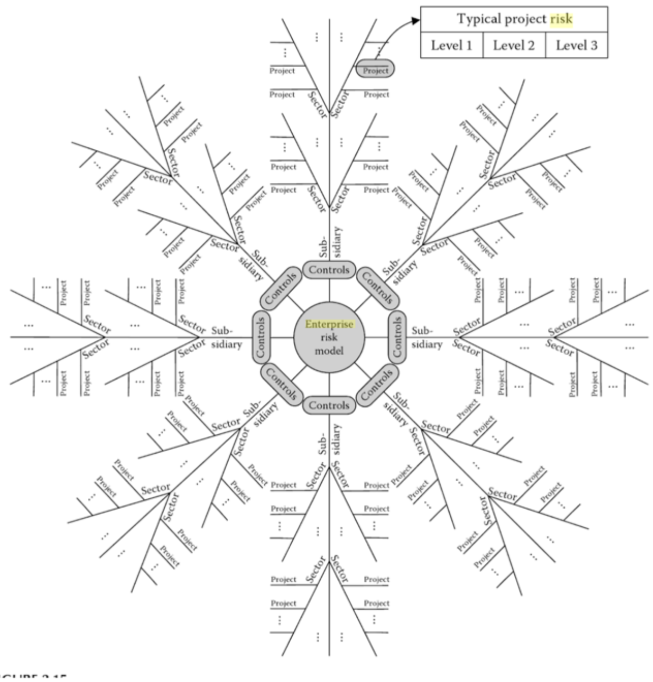


Figure B-4. Self-similarity in structuring enterprise risk *(Ayyub 2010)*

Another inspiration for control architecture for complex dynamic systems is learning from tried and tested techniques in Robotics motion. Robots use a **nested logic** to make sense and make movements. Illustration of this nested logic model framework is provided in figure (B-5). For example, there are different actuators, sensors on each finger. Each of these finger modules have various ensemble models running and project an action for the robot to take. The robot will move a finger or different fingers based on a nested logic, where individual fingers model is a part of the wrist, and the model of the wrist is part of the hand module and so on (Ayyub et al. 1997). Spatio-temporal interest points (Laptev 2005, Willems et al. 2008) and flow in video volumes, or represent short actions by stacks of silhouettes (Blank et al. 2005, Yilmaz and Shah 2005). Approaches to more complex, longer actions employ parametric approaches, such as Hidden Markov Models (Kale et al. 2004), Linear Dynamical Systems (Saisan 2001) or Non-linear Dynamical Systems (Chaudhry et al. 2007), which are defined on extracted features. Active tracking and segmentation method that monitors the changes in appearance and topological structure of the manipulated object are then used in visual semantic graph (VSG) procedure applied to the time sequence of the monitored object to recognize the action consequence (Aloimonos et al. 2014, 2015). In similar sentiment, illustrations for how different risk instances and classes can be controlled and aggregated in a **modular architecture** is shown below.

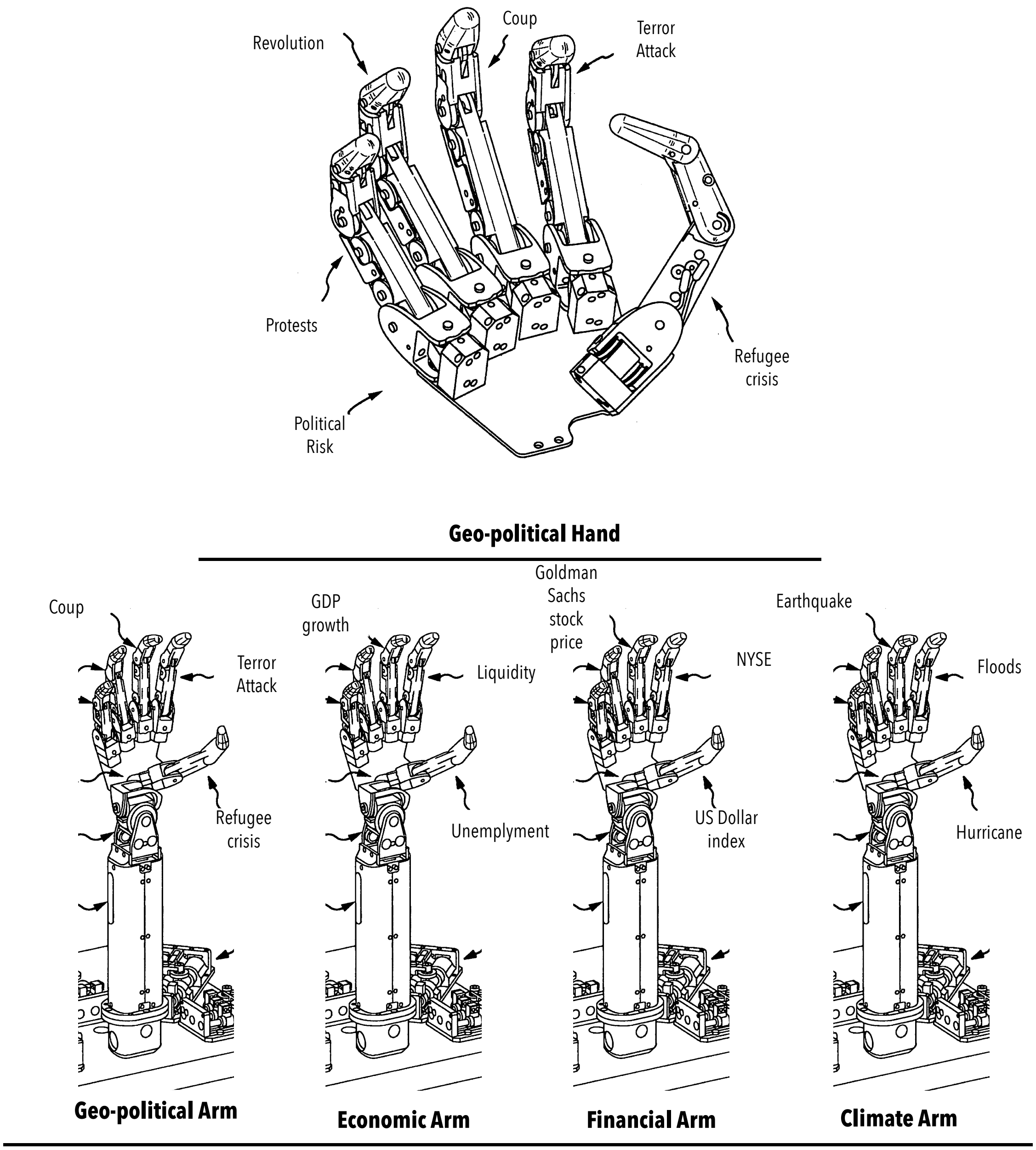


Figure B-5. Abstraction of Robotic Arm Control with Control System in Models for Identification and Prediction of Opportunities and Threats

Box 1. Data System Specification

The real-time risk monitoring uses following components for large-scale data storage and retrieval:

- Open Source Data
- Geospatial, Entity, Domain Features Tags
- Entities, Time, and Dimensions
  - The agent organizes and indexes multiple information sources to entities (Countries, Cities, Companies, Actors), time (annual, quarterly, monthly, daily), and dimension, classes and instances (such as earthquake magnitude, stock market prices, unemployment etc. with associated meta-data) and events in this high dimensional space.
- Signal detection using multiple sensor fusion
  - For example, real-time natural calamity data is accessed (*USGS*), economic facts (*QUANDL, World Bank, IMF, Stock Market, Bitcoin, and hundreds of macroeconomic and microeconomic statistics*), behavioral fluctuations and social movements (*classifying Twitter data, Search Engine Trends*), political-military movements and risks (*GDELT*), involuntary migration (*UN migration databases*) and so on.
- Large Scale Data Pipeline
  - The data pipeline converts data from various sources csv, xls, docs etc. to store data in a postgresSQL source tables.
  - Django-Docker Containers for ingesting data in postgresSQL
  - End-point will convert to schema-less MongoDB database
- o Data Storage and Processing
  The critical operational risk is maximizing resources from cloud services, and other third-party service providers to store and process the data at scale and cost-effectively. However, for processing GPU scaled learning, computational capacity may rely on multiple decentralized server clusters.

Appendix C. Unsupervised Methods

The **curse of dimensionality** is the phenomena whereby an increase in the dimensionality of a data set results in exponentially more data being required to produce a representative sample of that data set. To combat the curse of dimensionality, numerous linear and non-linear dimensionality reduction techniques have been developed. These techniques aim to reduce the number of dimensions (variables) in a data set through either feature selection or feature extraction without significant loss of information. A graphical illustration for the curse of dimensionality is provided in figure (C-1).

Feature extraction is the process of transforming the original data set into a data set with fewer dimensions. Two well known, and closely related, feature extraction techniques are **Principal Component Analysis (PCA)** and **Self Organizing Maps (SOM)**. One can think of dimensionality reduction like a system of aqueducts to make sense of a river of data.

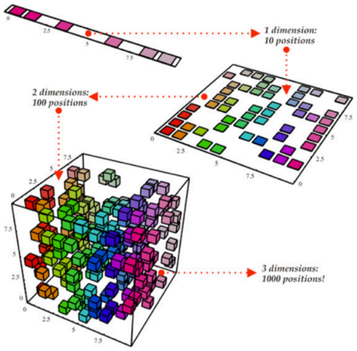


Figure C-1. The Curse of Dimensionality.
Source: BigSnarf 2013. Notation text from Turing Finance.
Notes: With one dimension (top left) there are only 10 possible positions therefore 10 datum are required to create a representative sample which 'covers' the problem space. With two dimensions, there are 10^2 = 100 possible positions therefore 100 datum are required to create a representative sample which 'covers' the problem space. With just three dimensions there are now 10^3 = 1000 possible positions therefore 1000 datum are required to create a representative sample which 'covers' the problem space.

A large number of nonlinear dimensionality reduction techniques that aim to preserve the local structure of data have been proposed, many of which are reviewed by Lee and Verleysen (2007). There are many other dimensionality reduction techniques including: (1) Sammon mapping (Sammon 1969), (2) curvilinear components analysis (Demartines and Herault 1997), (3) Stochastic Neighbor Embedding (Hinton and Roweis, 2002), (4) Isomap (Tenenbaum et al. 2000), (5) Maximum Variance Unfolding (Weinberger et al., 2004), (6) Locally Linear Embedding (Roweis and Saul 2000), and (7) Laplacian Eigenmaps (Belkin and Niyog 2002). Despite the strong performance of these techniques on artificial data sets, they are often not very successful at visualizing real, high-dimensional data. In particular, most of the techniques are not capable of retaining both the local and the global structure of the data in a single map. For instance, a recent study reveals that even a semi-supervised variant of MVU is not capable of separating handwritten digits into their natural clusters (Song et al. 2007).

The Dynamic Time Warp (DTW) was discussed in the main text. Other key alogirthms that are discussed here include clustering algorithms such as t-SNE algorithm, multi-dimensional scaling, or autoencoder. Table (C-1) shows a summary table explanation for these measures.

Table C-1. Summary Table of Dimensionality Reduction and Clustering Algorithms

Model	Definition	Applications	References
Dynamic Time Warping (DTW)	DTW algorithms are used for measuring similarity between two temporal sequences.	temporal sequences of video, audio, and graphics data, automatic speech recognition, speaker recognition, online signature recognition, shape matching application.	Ratanamahatana and Keogh, 2008 ; Hayashi, Mizuhara, and Suematsu, 2005.
t-distributed stochastic neighbor embedding (t-SNE)	t-SNE is a machine learning algorithm for dimensionality reduction a nonlinear dimensionality reduction technique. It is well-suited for embedding high-dimensional data into a space of two or three dimensions for visualization purposes.	computer security research, music analysis, facial expression recognition, cancer research bioinformatics, biomedical signal processing, visualize high-level representations learned by an artificial neural network.	van der Maaten and Hinton, 2008 ; van der Maaten, 2009 ; van der Maaten and Hinton, 2012 ; van der Maaten, 2014 ; Watternberg, 2016.
Multi-dimensional Scaling (MDS)	MDS is a means of visualizing the similarity of individual data based on the distance matrix cases of the datasets. It is a form of non-linear dimensionality reduction. It is well-suited for embedding low-dimensional data into a higher dimensional space.	Mix-marketing models, Psychometrics, Cognitive Psychology, Ecological analysis, physics, political science', biology.	Takane, 2006 ; Cha, 2009 ; Borg and Groenen, 2005 ; Shoben, 1983 ; Young, 1984.
Hierarchical Clustering Analysis (HCA)	HCA is a method of cluster analysis which seeks to build a hierarchy of clusters.	Image recognition, Natural Language Programming (NLP), Robotics, Computer Graphics, data compression, pattern recognition.	Jianbo Shi and Jitendra Malik, 2012 ; Cai et al, 2014 ; Chipman, 2005 ; Balcan and Gupta, 2010.
Autoencoders (AE)	AE is an artificial neural network used for unsupervised learning in feature learning. AE learns a representation (encoding) for a set of data, typically for the purpose of dimensionality reduction. AE is also used in learning generative models of data.	Natural Language, Programming (NLP), Dimensionality Reduction, pretraining deep Network, one-class classification, denoising financial data.	Le, 2015 ; Bao et al, 2017 ; Wang et al, 2012; Lopez-Martin et al, 2017.

t-SNE

t-SNE is capable of capturing much of the local structure of the high-dimensional data very well, while also revealing global structure such as the presence of clusters at several scales (Matten and Hinton 2008). t-SNE is a nonlinear dimensionality reduction technique that is particularly well-suited for embedding high-dimensional data into a space of two or three dimensions, which can then be visualized in a scatter plot.

The t-SNE algorithm comprises two main stages. First, *t-SNE* constructs a probability distribution over pairs of high-dimensional objects in such a way that similar objects have a high probability of being picked, whilst dissimilar points have an extremely small probability of being picked. Second, t-SNE defines a similar probability distribution over the points in the low-dimensional map, and it minimizes the Kullback–Leibler divergence between the two distributions with respect to the locations of the points in the map. Note that whilst the original algorithm uses the Euclidean distance between objects as the base of its similarity metric, this should be changed as appropriate (Matten and Hinton 2008, Matten and Hinton, 2014).

Multi-dimensional Scaling (MDS)

t-SNE is a technique for visualizing low-dimensional data into a higher-dimensional space. **Multi-dimensional Scaling (MDS)** is a technique for information visualization to project high-dimensional data into a low-dimensional space.

MDS is a family of different algorithms, each designed to arrive at optimal low-dimensional configuration (p = 2 or 3). MDS methods include (1) Classical

MDS, (2) Metric MDS, (3) Non-metric MDS. The goal of MDS is following, given pairwise dissimilarities, reconstruct a map that preserves distances (Jung 2013).

MDS is a form of dimensionality reduction and refers to a set of related ordination techniques used in information visualization. It displays the information contained in a distance matrix.

An MDS algorithm aims to place each object in *N*-dimensional space such that the between-object distances are preserved as well as possible. Each object is then assigned coordinates in each of the N-dimensions. The number of dimensions of an MDS plot N can exceed 2 and is specified a priori. Choosing *N*=2 optimizes the object locations for a two-dimensional scatterplot (Borg 2005). Researchers have studied the perception of color in human vision using MDS (Ekman 1954).

Appendix D. Classification Problems

Random time-series data are traditionally used for regression-based problem. This Appendix shows possible ways to use methods to convert such time-series information to class labels. They also Following this, classification problem algorithms such as peak signal detection – z-threshold algorithm and the Bayesian change-point estimation techniques are briefly discussed.

Z-Threshold Peak Detection Algorithm

The foundation for building reasoning and structured inference requires a probabilistic perspective to define tipping-points and anomalous events. The probability density for a given label's time-series features provides avenue to define probability crossings for given thresholds. In this manner, the inputs for machine learning and AI models can also be based converting regression problems into risk and success classification problems. A simple illustration for this case is the random variable for the Price to Earnings (PE) ratio time-series information for Goldman Sachs in figure (D-1). This shows that zooming into the tails of the distribution may provide a different distribution that can be considered thresholds. From a time-series perspective, decision-makers are interested of crossing this threshold in subsequent periods or to label such relatively rarer events in a systematic manner.

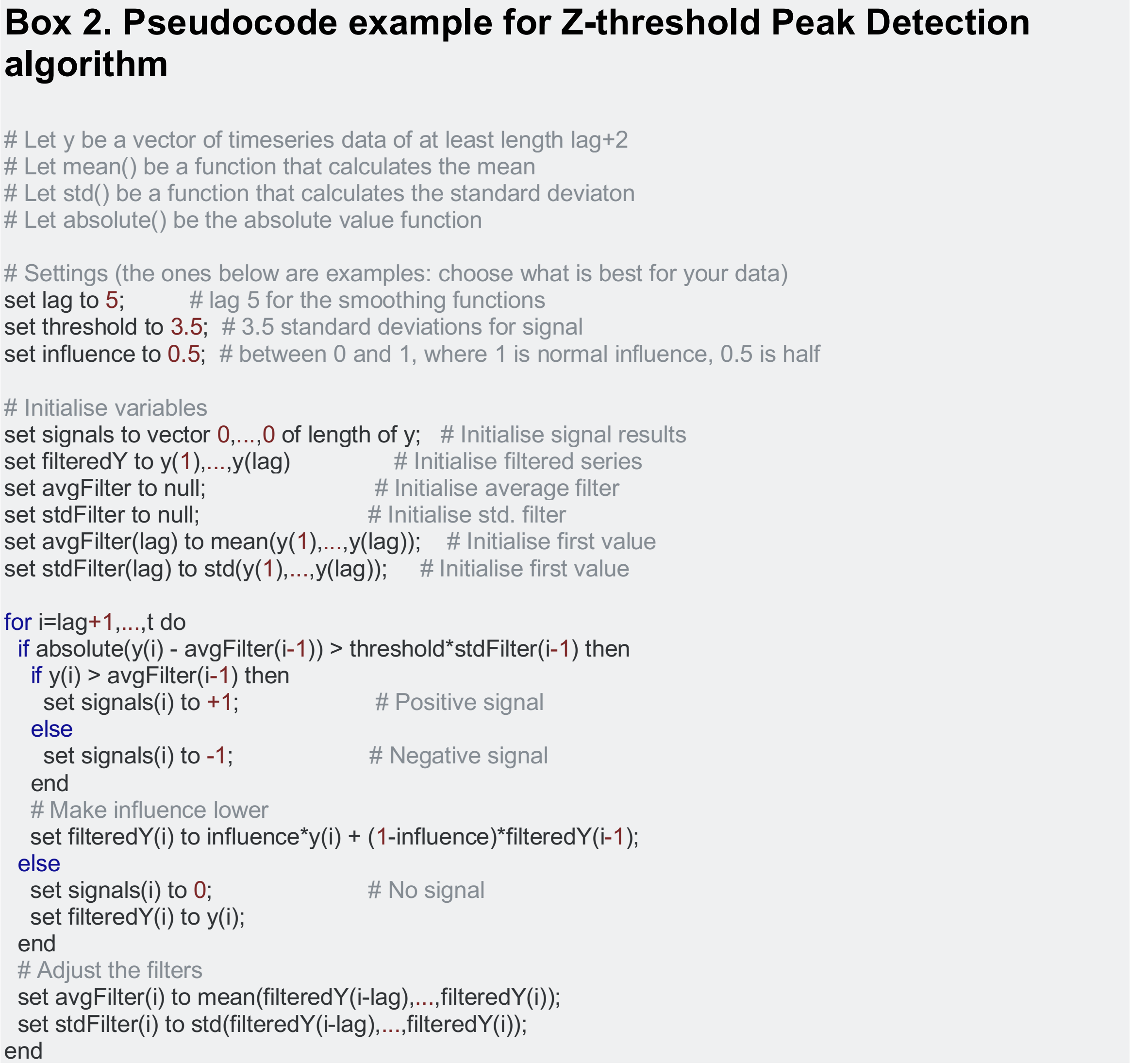
Anomaly Detection with Z-Score Threshold Algorithms is an algorithm that works very well for time series trends. This is available open-source on the web. It is based on the **principle of dispersion**: if a new data point is a given *x* number of standard deviations away from some moving mean, the algorithm signals (also called z-score). The algorithm is very robust because it constructs a separate moving mean and deviation, such that signals do not corrupt the threshold. The author of the algorithm state that "*Future signals are therefore identified with approximately the same accuracy, regardless of the number of previous signals. The algorithm takes 3 inputs: lag = the lag of the moving window, threshold = the z-score at which the algorithm signals and influence = the influence (between 0 and 1) of new signals on the mean and standard deviation.*" For example, a lag of 5 will use the last 5 observations to smooth the data. A threshold of 3.5 will signal if a data point is 3.5 standard deviations away from the moving mean. And an influence of 0.5 gives signals half of the influence that normal data points have. Likewise, an influence of 0 ignores signals completely for recalculating the new threshold: an influence of 0 is therefore the most robust option; 1 is the least. The pseudocode for the peak signal detection algorithm is presented in Box 2.

These peak and trough signals can also be very useful in labelling data. This will be useful as well see to also extend such sequence time series information into labelling classification. For example, if we wish to ask the system simply binary based results for whether GDP will contract, unemployment will increase, or stock price will go up tomorrow. Similarly, more detailed probabilistic measures of the risk of a GDP contraction, or the probability of unemployment increasing can also be computed.

It is essential to manually label by expert opinions the risk label classification. For example, if GDP growth goes negative it is a risk, but if unemployment goes up it is labelled a risk. The manual input for these classifications provide the interpretation of peak and troughs for a given indicator. The tipping points can be defined as failures and labelled more precisely by grouping various risk indicators into risk classes.

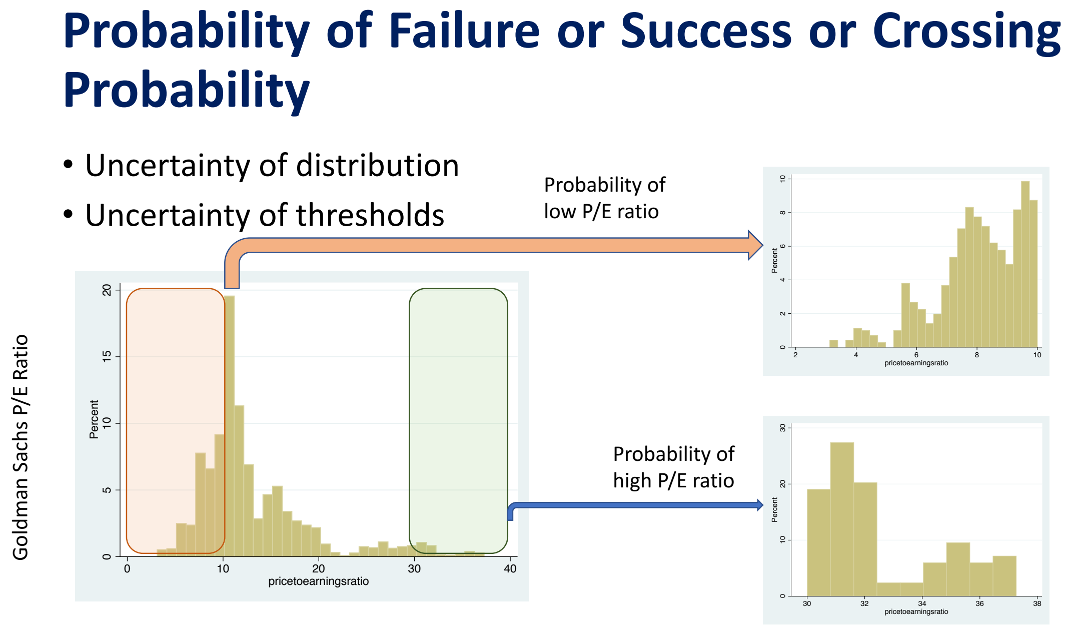
From a system perspective, the following parametrizations are used as standard. For annual data, the lag=2, for quarterly data lag=8, for daily data lag=60. Examples for political threat and risk classification for Protests are presented in Chapter 2 using the z-score algorithm for daily data from GDELT.

A simple case of sensitivity of time-series data to change in parameters is presented below in Figure (D-2). The variable is the percent change in unemployment rate with earlier period. The first graph has the following parameters: lag = 4, threshold = 3, and influence = 0.2. There is a large increase in the unemployment from previous years (in 1st subplot) during the early 1990s, the early 2000s and the 2008 economic crisis. These sudden peaks are detected by the algorithm and appropriately indexed in the 2nd subplot. By changing the lag value to 5 and keeping other parameters same, some of the peaks are lost, and all of the troughs, but retain the 3 major periods of risk (with a shortened risk period for the early 1990s). It is necessary to find a way to tune the lag properly, across indicators, or just assume a standard (which may not work as well).



Box 2. Pseudocode example for Z-threshold Peak Detection algorithm

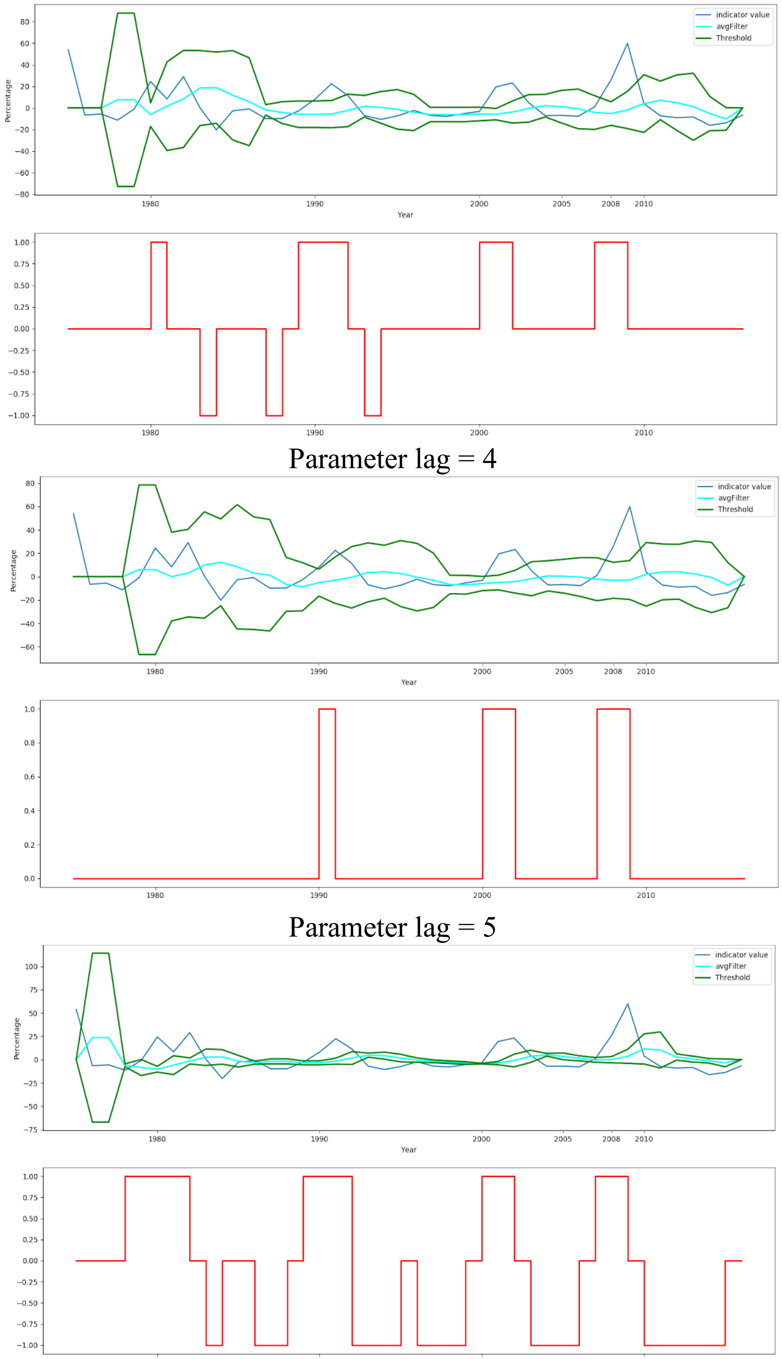
```
# Let y be a vector of timeseries data of at least length lag+2
# Let mean() be a function that calculates the mean
# Let std() be a function that calculates the standard deviaton
# Let absolute() be the absolute value function

# Settings (the ones below are examples: choose what is best for your data)
set lag to 5;          # lag 5 for the smoothing functions
set threshold to 3.5;  # 3.5 standard deviations for signal
set influence to 0.5;  # between 0 and 1, where 1 is normal influence, 0.5 is half

# Initialise variables
set signals to vector 0,...,0 of length of y;   # Initialise signal results
set filteredY to y(1),...,y(lag)                # Initialise filtered series
set avgFilter to null;                          # Initialise average filter
set stdFilter to null;                          # Initialise std. filter
set avgFilter(lag) to mean(y(1),...,y(lag));    # Initialise first value
set stdFilter(lag) to std(y(1),...,y(lag));     # Initialise first value

for i=lag+1,...,t do
  if absolute(y(i) - avgFilter(i-1)) > threshold*stdFilter(i-1) then
    if y(i) > avgFilter(i-1) then
      set signals(i) to +1;                     # Positive signal
    else
      set signals(i) to -1;                     # Negative signal
    end
    # Make influence lower
    set filteredY(i) to influence*y(i) + (1-influence)*filteredY(i-1);
  else
    set signals(i) to 0;                        # No signal
    set filteredY(i) to y(i);
  end
  # Adjust the filters
  set avgFilter(i) to mean(filteredY(i-lag),...,filteredY(i));
  set stdFilter(i) to std(filteredY(i-lag),...,filteredY(i));
end
```



Probability of Failure or Success or Crossing Probability

- Uncertainty of distribution
- Uncertainty of thresholds

Figure D-1. Example of Threshold Crossing Probability (Static Example) - Risk Label Propagation



Parameter lag = 4

Parameter lag = 5

Parameter lag = 2

Figure D-2. Z-threshold peak detection algorithm applied to unemployment rate in the USA. The case of changing the lag parameters is shown above.

Bayesian Change-point Anomaly Detection

A new online algorithm for exact inference of the most recent changepoint is provided where model parameters before and after the changepoint are independent (Adam and MacKay 2007). The probability distribution of the length of the current "run," or time since the last changepoint is computed using a simple message-passing algorithm. The provided by Adam and Mackay is highly modular so that the algorithm may be applied to a variety of types of data, including climate change.

Naïve Bayes Classifier

Common text categorization task, **sentiment analysis,** the positive or negative orientation that a writer expresses toward some object. Many examples are present for example, review of a movie, book, or product on the web expresses the author's sentiment toward the product, while an editorial or political text expresses sentiment toward a candidate or political action. Automatically extracting consumer sentiment is important for marketing of any sort of product, while measuring public sentiment is important for politics and also for market prediction (Jurafsky and Martin 2017). They are useful in various applications such as spam detection and authorship attribution.

The simplest version of sentiment analysis is a binary classification task, and the words of the review provide excellent cues. Consider, for example, the following phrases extracted from positive and negative reviews of movies and restaurants. Words like great, richly, awesome, and pathetic, and awful and ridiculously are very informative cues.

In such contexts, there is also a need for large human corpus of labelled data. Analysts may have to build a large enough training set of labelled data from humans. For the rest of the manuscript, most examples that will be used will be focused on extracting and labelling real-time sentiment data from platforms such as Twitter. The goal of human annotation or classifying sentiments from Twitter is purely to gain more aggregate information about events, organizations, political actors, or classes of risks.

The classification of Twitter data into meaningful aggregation can also be an important signal for monitoring reputation equity risks (Rust et al. 2017). For example, monitoring obsolete components or companies' global sentiments (or by locations and countries) can provide valuable signals for marketing, product launches, and product performance. These information feeds can be another input signal source for monitoring targeted opportunity and risk monitoring.

The goal of **classification** is to take a single observation, extract some useful features, and thereby classify the observation into one of a set of discrete classes.

Appendix E. Empirical Risk Minimization and Optimization

Since frequentist decision theory does not provide any automatic way to choose the best estimator, we need to come up with other heuristic selection principles. In frequentist statistics, a parameter estimate $\hat{\theta}$ is computed by applying an estimator δ to some data $\mathcal{D}$, so $\hat{\theta} = \delta(\mathcal{D})$. The parameter is viewed as fixed and the data as random, which is the exact opposite of the Bayesian approach (Murphy 2012). The uncertainty in the parameter estimate can be measured by computing the sample distribution of the **estimator.** To understand this concept better, imagine sampling many different data sets $\mathcal{D}^{(s)}$ from some true model, p(. |θ ∗), i.e. let

$$\boldsymbol{\mathcal{D}}^{(s)} = \left\{\boldsymbol{x}_i^{(s)}\right\}_{\boldsymbol{i=1}}^{\boldsymbol{N}}$$

where $x_i^{(s)} \sim p(.|\theta *)$, and $\theta*$ is the true parameter. Here *s = 1: S* indexes the sampled data set, and *N* is the size of each such dataset. Now apply the estimate $\hat{\theta}(.)$ to each $\mathcal{D}^{(s)}$ to get a set of estimates, $\{\hat{\theta}(\mathcal{D}^{(s)})\}$.

An estimator is said to be consistent if it eventually recovers the true parameters that generated the data as the sample size goes to infinity, i.e. $\hat{\theta}(\mathcal{D}) \rightarrow \theta*$ as $|\mathcal{D}| \rightarrow \infty$ (where the arrow indicates convergence in probability).

Frequentist decision theory suffers from the fundamental problem that one cannot actually compute the risk function, since it relies on knowing the true data distribution (Russel and Norvig 2010). By contrast the Bayesian posterior expected loss can always be computed since it conditions on the data rather than conditioning on $\theta*$. However, there is one setting which avoids this problem, and that is where the task is to predict observable quantities as opposed to estimating hidden variables or parameters. That is, instead of looking at loss functions of the form $L\ (\boldsymbol{\theta}, \boldsymbol{\delta}(\boldsymbol{\mathcal{D}}))$**,** where $\boldsymbol{\theta}$ is the true but unknown parameter, and $\delta(\mathcal{D})$is our estimator.

Supervised learning can usually be seen as picking one function *f* from a set of possible functions *F*. An obvious question is, how can we tell a good function *f* from a bad one? These notes introduce a general framework that applies for many of the methods for classification and regression that follow (Russel and Norvig 2010, Murphy 2012). First, we introduce the concept of a **loss function L**. Given some particular pair of inputs **x** and outputs **y**,

$$\boldsymbol{L\ (f(x), f(y))}$$

tells us how much it "hurts" to make the prediction *f(x)* when the true output is *y*. Now, let us define the (true) risk

$$\boldsymbol{R_{true}(f)} = \ \mathbb{E}_{\boldsymbol{p}}\ [\boldsymbol{f(x), y}] = \iint \boldsymbol{p\ (x, y)\ L(f(x), y)\ \ dx\, dy}$$

Here *p* is the true distribution over the inputs **x** and **y**. The risk measures how much, on average, it hurts to use *f* as our prediction algorithm.

This can all be made clear by considering an example. Suppose we want to fit a function for predicting if there will be a macroeconomic contraction period i.e. low GDP growth event or very high public debt (% of GDP) event. The input **x** will be the GDP growth: Extremely Low (GDP Growth<0), Low (3<GDP Growth<0), or High (GDP Growth>3), The output **y** will be either, CONTRACTION (when it is extremely LOW) or NOPE (when the country experiences no contraction). The loss function is now a function *L* : {CONTRACTION, NOPE} 2 → $\mathfrak{R}$.

What loss function is appropriate? It is important to realize that this cannot be answered by math. The loss function depends on the priorities of the policymaker or investment decision-maker. For example, if you are an investor who really hates being in a financial crisis but doesn't particularly mind being ready for a crisis environment, you might use a different loss function.

Logistic Regression Cost Function

Traditionally, the loss error function is defined as

$$\boldsymbol{\mathcal{L}\,(\hat{y}, y) = \frac{1}{2}(\hat{y} - \ y)^2}$$

But in logistic regression we don't generally do this. This optimization problem that has not only local minima and so gradient descent may not find the local optima. This function L loss function describes how good y hat is compared to actual value. SGD solves the convex optimization. If we use squared error we want the squared error to be as small as possible. More specifically for logit regression the following loss function is more reliable

$$\boldsymbol{\mathcal{L}\,(\hat{y}, y) = \ (y\ log\,\hat{y} + (1-y)log\ (1-\ \hat{y}))}$$

The **cost function** provides a basis to say how well the model performs on the entire training set. The cost function *J* has the following form

$$\boldsymbol{J\,(w, b) = \frac{1}{m}\sum_{i=1}^{m}\mathcal{L}\left(\widehat{y^{(i)}},\ y^{(i)}\right)}$$

$$\boldsymbol{= -\frac{1}{m}\sum_{i=1}^{m}\left(y^{(i)}\ log\ \widehat{y^{(i)}} + \left(1-y^{(i)}\right)log\ (1-\ \widehat{y^{(i)}})\right)}$$

The loss function is applied to a single training example, and the cost function is the cost of the parameters. Therefore, in training a logistic regression problem, we will try to *w* and *b* that will minimize the overall cost function *J*.

The problem of training a neural network into an optimization problem. Let us set θ a set of all parameters in the model, these are all the parameters in connection matrices as well as the bias vectors, *w* and *b* matrices for all the different layers. We also know $x^{(t)}$ as a training example *t*-th training example in some set of training examples we already have. $x^{(t)}$ is a vector and t-th input vector in the training set. $y^{(t)}$ is the *t*-th prediction example output corresponding to $x^{(t)}$. So $x^{(t)}$ can be the name of the country, and $y^{(t)}$ the associated characteristics associated with that country.

$$arg\min_{\theta}\frac{1}{T}\sum_{t} l(f(x^{(t)};\ \theta), y^{(t)}) +\ \lambda\Omega(\theta)$$

$l(f(x^{(t)};\ \theta), y^{(t)})$ is the loss function

$\Omega(\theta)$ is a regularization that penalizes certain values of θ

In empirical risk minimization, the problem of learning or training the model is about finding the parameters (θ) of the objective function that minimizes the objective function (Murphy 2010). The objective function has two parts. The first part is the average of the loss function *l* that compares the output of the NN with the expect correct answer $y^{(t)}$. *L* is the loss function that compares the output with the label. The term Ω is a regularizer which penalizes certain value that will not work on new examples of the data. The parameter λ is the hyperpatremter that controls the balance between optimization the average loss and optimizing the regularization. Ideally we want to minimize the percentage of error we obtain on the training set. However, this function is not a smooth function, if we want to optimize the number is the right class of the model, the number of time the most likely class output by the neural network is wrong, this function is not smooth. It looks a binary step function, providing a hard function to optimize. Instead, we use loss function that is surrogate for the range that are truly trying to optimize (eg. Upper bound).

Stochastic Gradient Descent

Since the above section casts the neural network as an optimization problem, one common algorithm is the Stochastic Gradient Descent (SGD) algorithm. First, SDG initializes the parameters, connection and biases. For some duration *N* we will iterate over our training set, for each pair $x^{(t)}$ and $y^{(t)}$, to figure out a direction (Δ). for updating the parameters, it is equal to the opposite direction of the gradient.

In addition to Δ, another parameter learning rate α - a hyper parameter multiplied by the direction of updating the parameters. The SGD takes a step in the direction locally that is going to decrease the sum of loss and regularizer, and repeat the whole loop *N* times or training epochs over all the training examples

A gradient descent minimizes an error function, minimizes respect to *w* – the in sample error. In order to capture the error or the gradient of the descent one needs to evaluate the hypothesis at every point in the sample (Russel and Norvig 2010). The equation provides whether error or the direction is in the good direction. GD minimizes

$$E_{in}(w) = \frac{1}{N}\sum_{n=1}^{N} e(h(x_n), y_n\ )$$

In the case of a logistic regression you can get the gradient with respect to that vector, where $e(h(x_n), y_n\ ) = ln\left(1 + \ e^{-y_n w^T x_n}\right)$.

Now iterative steps, where one step is a full epoch (when you have considered all the examples at once) and instead of movement of all in the w space based on all the examples, we are going to try and do it based on one example at a time. This is what makes Stochastic Gradient Descent (SGD). So the standard GD takes a batch of all the examples and does a move at once.

by iterative steps along $-\nabla\ E_{in}$:

$$\Delta w = \ -\eta\nabla\ E_{in}(w)$$

$\nabla\ E_{in}$ is based on all examples (x_n, y_n)

The stochastic aspect is as follows. Pick one example at a time (randomly) and apply GD on that point only. Now think of the average direction where you will descend along. If you take the gradient of the error measure that you are going to minimize. In this case, one example, and the expected value from the entire training set can be easily evaluated. So every step the agent is going alone is accounted for plus the noise. Interestingly, this is identical minus the gradient of the total in-sample error. In expected value we are actually going along the direction we want, except that we involve one example in the computation which is a big advantage and we have a stochastic aspect to the game. And as you repeat, you always get an expected value in that direction and different noises depending on which example. After doing this iteratively after many times, the noise will actually be going along the ideal direction.

Pick one (x_n, y_n) at a time. Apply GD to $e(h(x_n), y_n)$

Average direction:

$$\mathbb{E}_n[-\nabla\, e(h(x_n), y_n)] = \frac{1}{N}\sum_{n=1}^{N} -\nabla\, e(h(x_n), y_n\ ) = -\nabla\, E_{in}$$

This is the randomized gradient descent often referred to as the stochastic gradient descent (SGD). SGD are cheap computation and doing this for each point or a batch GD, the expected value is attractive. There is an aspect of randomization that makes optimization more powerful.

In real world financial, economic, social and political data one is faced with many hills and valleys. One may start at different points of time with different local minima's. In order to at least avoid shallow valleys that the optimization does not get stuck in local minima's. The idea is that because you are not going in the direction that is not deterministic and a random path, with a slight chance that you escape from the local minima. SGD at least helps escaping some silly local minima trap.

One rule of thumb that is useful in practical applications. The learning rate which tells us how far you go, if it is too big then you lose the linear approximation. If it is too small then you are moving too slowly. The exact answer depends on the case and scaling the error up or down. From a practical point of view, a normal error function to start it off could be $\eta = 1$.

Gradient Descent in Logistic Models

The cost function *J* measures how well the parameters w and b are doing in the training set. Or in other words, find *w* and *b* that minimize *J(w, b)*. Note that

J(w, b) is convex (Murphy 2012).

Generally, you initialize to 0 but random initialization could also work. Since the function is convex, we should roughly end up at the same point. GD starts at an initial point and then takes a step in the steepest downhill direction or quickly as possible – in one iteration and then several iterations later converge to the global optima. For example, for a one-dimensional case of finding w,

Repeat {

$$\mathrm{w} := \mathrm{w} - \propto \frac{dJ(w)}{dw}$$

}

Repeatedly do the above till the algorithm converges

where $\propto$ is the **learning rate** and controls how big a step to take at each iteration (Russel and Norvig 2010). The derivate signals the update to the parameter *w*. In order to find the *J(w, b),* we can write

$$\mathrm{w} := \mathrm{w} - \propto \frac{dJ(w,b)}{dw}$$

$$\mathrm{b} := \mathrm{b} - \propto \frac{dJ(w,b)}{db}$$

Training is a minimization problem, by using the negative log-likelihood for the *y*-th element of the output layer. This is also referred to as cross-entropy. By comparing two vectors and one hot encoded vector corresponds to the distance between two variable is cross entropy (Murphy 2012).

L1 and L2 Error Function and Regularizer

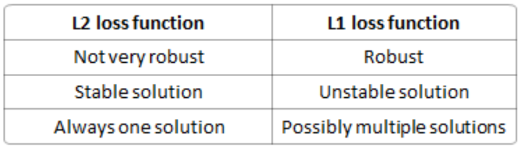
L1-norm loss function is also known as least absolute deviations (LAD), least absolute errors (LAE) (Russel and Norvig 2010). It is basically minimizing the sum of the absolute differences **(S)** between the target value (**Y$_i$**) and the estimated values **(f(x$_i$))**:

$$S = \sum_{i=1}^{n} | \, y_i \; - \; f(x_i) \, |$$

L2-norm loss function is also known as least squares error (LSE). It is basically minimizing the sum of the square of the differences **(S)** between the target value (**Y$_i$**) and the estimated values **f(x$_i$):**

$$S = \sum_{i=1}^{n} \left(\, y_i \; - \; f(x_i) \right)^2$$

The differences of L1-norm and L2-norm as a loss function can be promptly summarized as follows:



L2 loss function	**L1 loss function**
Not very robust	Robust
Stable solution	Unstable solution
Always one solution	Possibly multiple solutions

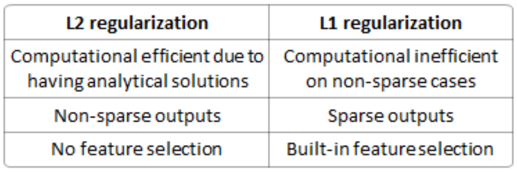
Regularization is a very important technique in machine learning to prevent overfitting. Mathematically speaking, it adds a *regularization term* in order to prevent the coefficients to fit so perfectly to overfit. The difference between the L1 and L2 is just that L2 is the sum of the square of the weights, while L1 is just the sum of the weights (Murphy 2012, Russel and Norvig 2010). L1 regularization on least squares can be written as:

$$w^* = \arg\min_w \sum_j \left(t\left(x_j\right) - \sum_i w_i \ h_i\left(x_j\right) \right)^2 + \lambda \sum_{i=1}^{k} |w_i|$$

L2 regularization on least squares:

$$w^* = \arg\min_w \sum_j \left(t\left(x_j\right) - \sum_i w_i \ h_i\left(x_j\right) \right)^2 + \lambda \sum_{i=1}^{k} {w_i}^2$$

Notice the only difference is in the last term. The difference between their properties can be promptly summarized as follows:



L2 regularization	**L1 regularization**
Computational efficient due to having analytical solutions	Computational inefficient on non-sparse cases
Non-sparse outputs	Sparse outputs
No feature selection	Built-in feature selection

Appendix H. Overfitting

In machine learning and statistics, one common task is to fit a "model" to a set of training data, with the goal of making reliable predictions on unseen test data. **Overfitting** describes a statistical model that captures random error or noise instead of the underlying relationship that we are trying to predict. Overfitting generally occurs as model become more complex with relatively more hyper-parameters than number of observations. Overfitted models have poor predictability power and yield spurious results. Analysts need to be careful to not overfit the data, that is, we should avoid trying to model every minor variation in the input, since this is more likely to be noise rather than true signal.

When we have a variety of models of different complexity (E.g. linear or logistic regression models with different degree polynomials, or neural networks), how to pick the right one?

A natural approach is to compute the misclassification rate on the training set for each method. This is defined as follows

$$err\ (\ f\ ,\mathcal{D}\ ) = \frac{1}{N}\sum_{i=1}^{N} \mathbb{I}\ (\ f\ (x_i) \neq\ y_i$$

where *f (x)* is our classifier or prediction model.

What analysts should really care about is **generalization error**, which is the expected value of misclassification or prediction rate when averaged over future data. This can be approximated by computing the misclassification rate on a large independent test set, not used during model training.

One **epoch** consists of one full training cycle on the training set. Once every sample in the set is seen, you start again - marking the beginning of the 2nd epoch. **Loss function** is usually a function defined on a data point, your prediction and label, and measures the penalty. **Cost function** is usually a big more general object, it might for example be a sum of loss functions over your training set plus some model complexity penalty (regularization). By plotting the test error over different epochs, the analyst can easily interpret whether the model overfits or underfits. If the plot of the error rate vs training sets or epochs shows that error is increasing over the training set, then we are over-smoothing.

Unfortunately, when training the model, we do not have access to the test set (by assumption), so we cannot use the test set to pick the model of the right complexity. In academic settings, we usually do have access to the test set, but we should not use it for model fitting or model selection, otherwise we will get an unrealistically optimistic estimate of performance of our methods. This is one of the "golden rules" of machine learning research.

However, we can create a test set by partitioning the training set into two: the part used for **training** the model, and a second part, called the **validation set**, used for selecting the model complexity. We then fit all the models on the training set, and evaluate their performance on the validation set, and pick the best. Once we have picked the best, we can refit it to all the available data. If we have a separate test set, we can evaluate performance on this, in order to estimate the accuracy of our method. Often, 80% of the data is used for the training set, and 20% for the validation set.

Cross Validation

A simple but popular solution to this is to use **cross validation (CV)**. The idea is simple: split the training data in to *K* folds; then for each fold *k* {1, …, *K*}, we train on all the folds but the *k'th* and test on the *k'th*, in a round-robin fashion (Murphy 2012, Russel and Norvig 2010).

Cross-validation is primarily a way of measuring the predictive performance of a statistical model. Every statistician knows that the model fit statistics are not a good guide to how well a model will predict: high R^2 does not necessarily mean a good model. It is easy to over-fit the data by including too many degrees of freedom and so inflate R^2 and other fit statistics. For example, in a simple polynomial regression one can just keep adding higher order terms and so get better and better fits to the data. But the predictions from the model on new data will usually get worse as higher order terms are added.

One way to measure the predictive ability of a model is to test it on a set of data not used in estimation. Based on the "test set" and the data used for estimation is the "training set". For example, the predictive accuracy of a model can be measured by the mean squared error on the test set (Murphy 2012). This will generally be larger than the MSE on the training set because the test data were not used for estimation. However, there is often not enough data to allow some of it to be kept back for testing. A more sophisticated version of training/test sets is **leave-one-out cross-validation** (LOOCV) in which the accuracy measures are obtained as follows. Suppose there are *n* independent observations, $y_1,\ldots,y_n$.

1. Let observation i form the test set, and fit the model using the remaining data. Then compute the error ($e_i^* = y_i - \hat{y}_i$) for the omitted observation. This is sometimes called a “predicted residual” to distinguish it from an ordinary residual.
2. Repeat step 1 for *i*=1,…,*n*.
3. Compute the MSE from $e_1^*, e_2^*, \dots, e_n^*$. This is called the CV.

This is a much more efficient use of the available data, as you only omit one observation at each step. However, it can be very time consuming to implement (except for linear models). Other statistics (e.g., the MAE) can be computed similarly. A related measure is the PRESS statistic (predicted residual sum of squares) equal to n×MSE (Russel and Norvig 2010).

Variations on cross-validation include **leave-k-out cross-validation** (in which k observations are left out at each step) and **k-fold cross-validation** (where the original sample is randomly partitioned into *k* subsamples and one is left out in each iteration). Another popular variant is the .632+bootstrap of Efron & Tibshirani (1997) which has better properties but is more complicated to implement.

Minimizing a CV statistic is a useful way to do model selection such as choosing variables in a regression or choosing the degrees of freedom of a nonparametric smoother. It is certainly far better than procedures based on statistical tests and provides a nearly unbiased measure of the true MSE on new observations.

However, as with any variable selection procedure, it can be misused. Beware of looking at statistical tests after selecting variables using cross-validation — the tests do not take account of the variable selection that has taken place and so the p-values can mislead (Murphy 2012).

Cross-validation for linear models

While cross-validation can be computationally expensive in general, it is very easy and fast to compute LOOCV for linear models (Murphy 2012, Russel and Norvig 2010). A linear model can be written as

$$Y = x\beta + e$$

then

$$\hat{\beta} = (X'X)^{-1}X'Y$$

the fitted values can be calculated using

$$\hat{Y} = X\hat{\beta} = (X'X)^{-1}X'Y = HY$$

where $H = (X'X)^{-1}X'$ is known as the 'hat matrix' because it is used to compute the Y hat - $\hat{Y}$. If the diagonal values of $\boldsymbol{H}$ are denoted by $h_1, \ldots, h_n$ then the cross-validation statistic can be computed using

$$CV = \frac{1}{n}\sum_{i=1}^{n}[e_i/(1-h_i)^2]$$

where e_i is the residual obtained from fitting the model to all n observations. It is not necessary to actually fit n separate models when computing the CV statistic for linear models (Murphy 2012). This remarkable result allows cross-validation to be used while only fitting the model once to all available observations (Arlot and Celisse 2010).

Cross-validation for time series

When the data are not independent cross-validation becomes more difficult as leaving out an observation does not remove all the associated information due to the correlations with other observations (Russel and Norvig 2010). For time series forecasting, a cross-validation statistic is obtained as follows

1. Fit the model to the data $y_1,\ldots,y_t$ and let $\hat{y}_{t+1} =$ denote the forecast of the next observation. Then compute the $e^*_{t+1} = (y_{t+1} - \hat{y}_{t+1})$for the forecast observation.
2. Repeat step 1 for *t=m,…,n−1* where *m* is the minimum number of observations needed for fitting the model.
3. Compute the MSE from $e^*_{m+1}, \ldots, e^*_n$

Dropout

A critical issue concerning neural networks is the over-fitting problem. It can be attributed to the fact that a neural network captures not only useful information contained in the given data, but also unwanted noise. This usually leads to a poor of generalization (Xiong et al. 2011, Salakhutdinov and Mnih 2008). Deep neural nets with a large number of parameters are powerful machine learning systems. However, overfitting is a serious problem in such networks. Large networks are also slow to use, making it difficult to deal with overfitting by combining the predictions of many different large neural nets at test time (Srivastava et al. 2014).

Dropout is a technique for addressing this problem. The key idea is to randomly drop units (along with their connections) from the neural network during training. This prevents units from co-adapting too much. During training, dropout samples from an exponential number of different “thinned” networks. At test time, it is easy to approximate the effect of averaging the predictions of all these thinned networks by simply using a single un-thinned network that has smaller weights. This significantly reduces overfitting and gives major improvements over other regularization methods.

With limited training data, many complicated relationships will be the result of sampling noise, so they will exist in the training set but not in real test data even if it is drawn from the same distribution. This leads to overfitting and many methods have been developed for reducing it. These include stopping the training as soon as performance on a validation set starts to get worse, introducing weight penalties of various kinds such as L1 and L2 regularization and soft weight sharing (Nowlan and Hinton, 1992).

The term "**dropout**" refers to dropping out units (hidden and visible) in a neural network. By dropping a unit out, we mean temporarily removing it from the network, along with all its incoming and outgoing connections, as shown in the first Figure (E-1) below. The choice of which units to drop is random. In the simplest case, each unit is retained with a fixed probability *p* independent of other units, where p can be chosen using a validation set or can simply be set at 0.5, which seems to be close to optimal for a wide range of networks and tasks ((Nowlan and Hinton 1992, Srivastava et al. 2014). For the input units, however, the optimal probability of retention is usually closer to 1 than to 0.5 (Srivastava et al. 2014).

Each hidden unit in a neural network trained with dropout must learn to work with a randomly chosen sample of other units. This should make each hidden unit more robust and drive it towards creating useful features on its own without relying on other hidden units to correct its mistakes. However, the hidden units within a layer will still learn to do different things from each other.

The dropout neural network model is described as follows. Consider a neural network with L hidden layers. Let $l \in \{1, . . . , L\}$ index the hidden layers of the network. Let $z^{(l)}$ denote the vector of inputs into layer l, $y^{(l)}$ denote the vector of outputs from layer l ($y^{(0)}$ = x is the input) (Srivastava et al. 2014). $W^{(l)}$ and $b^{(l)}$ are the weights and biases at layer l. The feed-forward operation of a standard neural network in Figure (E-2a) can be described as (for $l \in \{0, . . . , L - 1\}$ and any hidden unit *i)*

$$z_i^{(l+1)} = w_i^{(l+1)} y^l + b_i^{(l+1)}$$

$$y_i^{(l+1)} = f\left(z_i^{(l+1)}\right)$$

where *f* is any activation function, for example, *f(x)* = 1/ (1 + exp(−x). Furthermore, with dropout, the feed-forward operation becomes as presented in Figure (2-b).

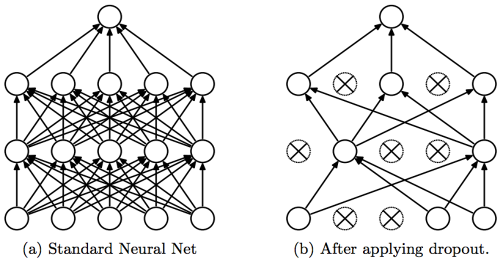
$$r_j^{(l)} \sim Bernoulli(p)$$

$$\tilde{y}_i^{(l)} = r^{(l)} * y^{(l)}$$

$$z_i^{(l+1)} = w_i^{(l+1)} \tilde{y}^l + b_i^{(l+1)}$$

$$y_i^{(l+1)} = f\left(z_i^{(l+1)}\right)$$

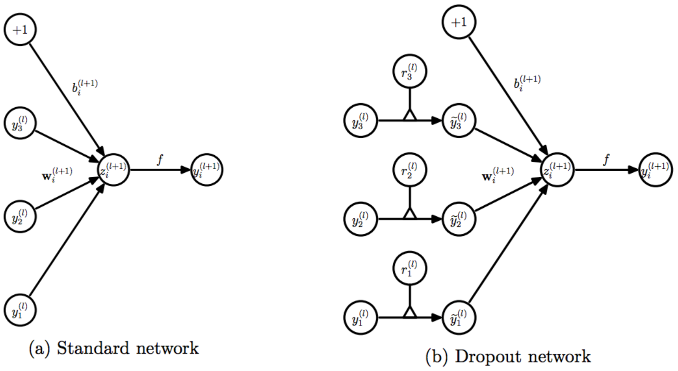
Here $*$ denotes an element-wise product. For any layer l, $r^{(l)}$ is a vector of independent Bernoulli random variables each of which has probability p of being 1 (Srivastava et al. 2014). This vector is sampled and multiplied element-wise with the outputs of that layer, $y^{(l)}$, to create the thinned outputs $\tilde{y}_i^{(l)}$. The thinned outputs are then used as input to the next layer. This process is applied at each layer. This amounts to sampling a sub-network from a larger network. For learning, the derivatives of the loss function are backpropagated through the sub-network. At test time, the weights are scaled as $W_{test}^{(l)} = pW^{(l)}$. The resulting neural network is used without dropout (Srivastava, et al 2014).



(a) Standard Neural Net

(b) After applying dropout.

Figure E-1. Dropout Neural Net Model. **Left:** A standard neural net with 2 hidden layers. **Right:** An example of a thinned net produced by applying dropout to the network on the left. Crossed units have been dropped.
Source: Srivastava et al. 2014.



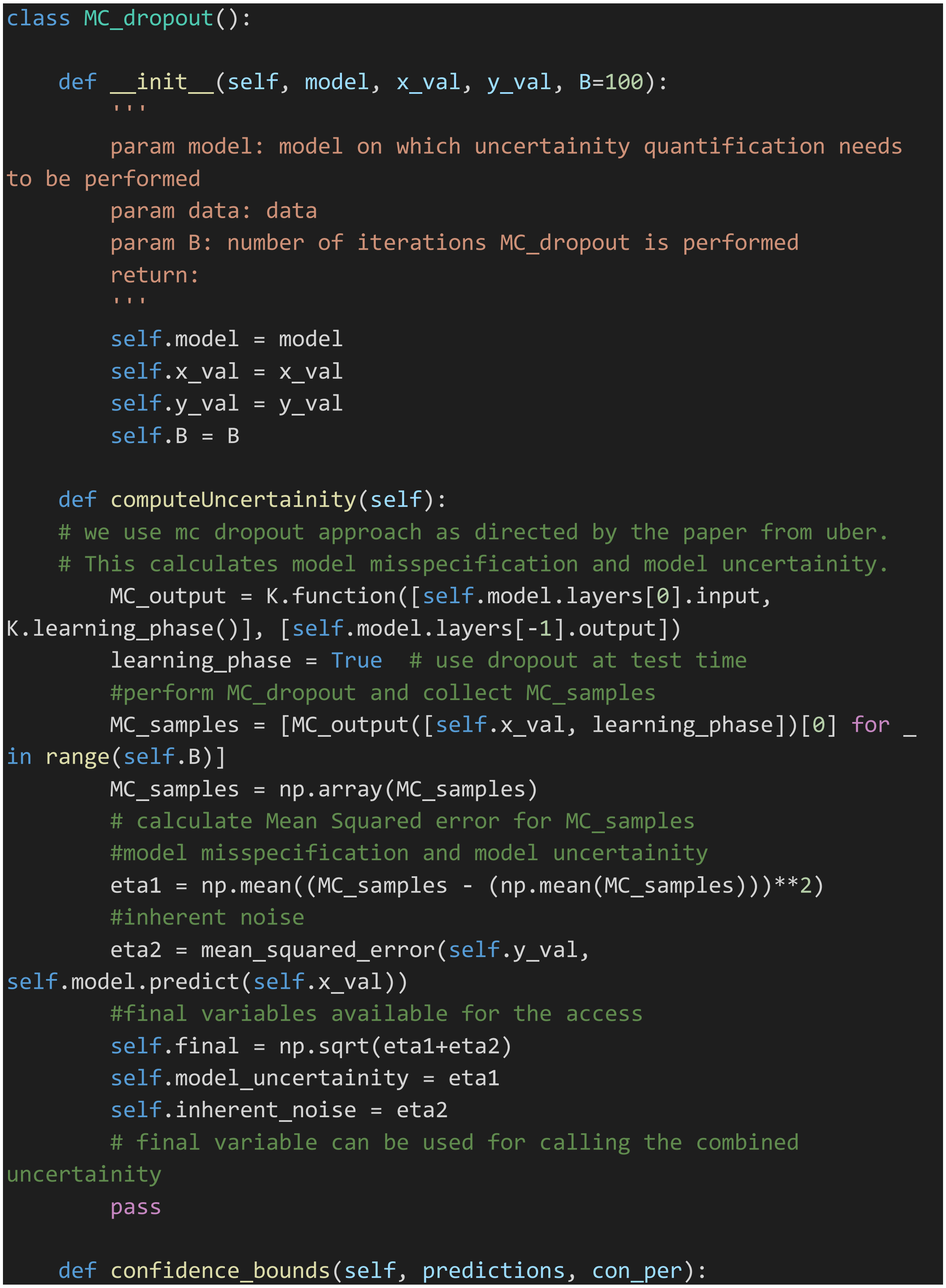
(a) Standard network

(b) Dropout network

Figure. E-2 Comparison of the basic operations of a standard and dropout network
Source: Srivastava et al. 2014.

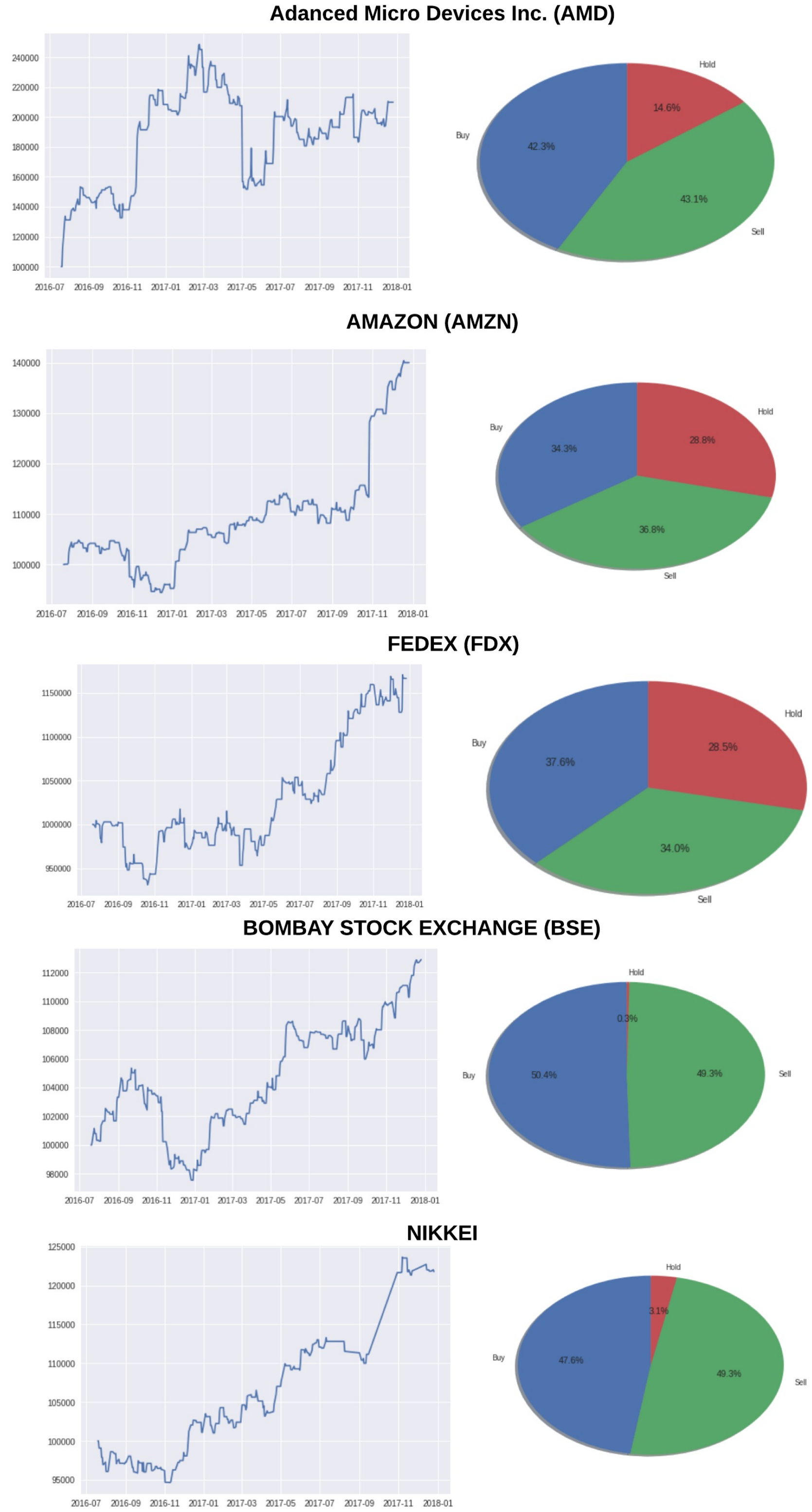
Appendix J. Uncertainty Quantification Class Abstraction

The UQ class implementation uses a monte carlo implementation class and is abstracted into the MC_Dropout.py file. The code is following:



```
class MC_dropout():

    def __init__(self, model, x_val, y_val, B=100):
        '''
        param model: model on which uncertainity quantification needs
to be performed
        param data: data
        param B: number of iterations MC_dropout is performed
        return:
        '''
        self.model = model
        self.x_val = x_val
        self.y_val = y_val
        self.B = B

    def computeUncertainity(self):
    # we use mc dropout approach as directed by the paper from uber.
    # This calculates model misspecification and model uncertainity.
        MC_output = K.function([self.model.layers[0].input,
K.learning_phase()], [self.model.layers[-1].output])
        learning_phase = True  # use dropout at test time
        #perform MC_dropout and collect MC_samples
        MC_samples = [MC_output([self.x_val, learning_phase])[0] for _
in range(self.B)]
        MC_samples = np.array(MC_samples)
        # calculate Mean Squared error for MC_samples
        #model misspecification and model uncertainity
        eta1 = np.mean((MC_samples - (np.mean(MC_samples)))**2)
        #inherent noise
        eta2 = mean_squared_error(self.y_val,
self.model.predict(self.x_val))
        #final variables available for the access
        self.final = np.sqrt(eta1+eta2)
        self.model_uncertainity = eta1
        self.inherent_noise = eta2
        # final variable can be used for calling the combined
uncertainity
        pass

    def confidence_bounds(self, predictions, con_per):
        '''
        param predictions: the predictions on which confidence 
intervals need to be built
        param con_per: the confidence interval for the predcitions in 
percentage
        return: returns High value of the predictions and Low value 
predicitons
        '''
        # the error value based on the normal distribution
        Merror = ((st.norm.ppf((1+(con_per/100))/2)) * self.final)
        # high and low bounds of the predictions
        mH = np.array([predictions + 
Merror]).reshape((len(predictions), 1))
        mL = np.array([predictions - 
Merror]).reshape((len(predictions), 1))
        return mH, mL
```

Appendix K. Additional Figures and Tables



Adanced Micro Devices Inc. (AMD)

AMAZON (AMZN)

FEDEX (FDX)

BOMBAY STOCK EXCHANGE (BSE)

NIKKEI

Figure K-1. Trading Strategies based on Ensemble Deep Learning Model. Y axis is starting investment at $100,000

Table K-1. Company and Sector Labels for S&P500

Symbol	**Name**	**Sector**	**Sectors**
0	MMM	3M Company	Industrials
1	ABT	Abbott Laboratories	Health Care
2	ABBV	AbbVie	Health Care
3	ACN	Accenture plc	Information Technology
4	ATVI	Activision Blizzard	Information Technology
5	AYI	Acuity Brands Inc	Industrials
6	ADBE	Adobe Systems Inc	Information Technology
7	AAP	Advance Auto Parts	Consumer Discretionary
8	AES	AES Corp	Utilities
9	AET	Aetna Inc	Health Care
10	AMG	Affiliated Managers Group Inc	Financials
11	AFL	AFLAC Inc	Financials
12	A	Agilent Technologies Inc	Health Care
13	APD	Air Products & Chemicals Inc	Materials
14	AKAM	Akamai Technologies Inc	Information Technology
15	ALK	Alaska Air Group Inc	Industrials
16	ALB	Albemarle Corp	Materials
17	ALXN	Alexion Pharmaceuticals	Health Care
18	ALLE	Allegion	Industrials
19	AGN	Allergan, Plc	Health Care
20	ADS	Alliance Data Systems	Information Technology
21	LNT	Alliant Energy Corp	Utilities
22	ALL	Allstate Corp	Financials
23	GOOGL	Alphabet Inc Class A	Information Technology
24	GOOG	Alphabet Inc Class C	Information Technology
25	MO	Altria Group Inc	Consumer Staples
26	AMZN	Amazon.com Inc	Consumer Discretionary
27	AEE	Ameren Corp	Utilities
28	AAL	American Airlines Group	Industrials
29	AEP	American Electric Power	Utilities
...	...	...	...
475	V	Visa Inc.	Information Technology
476	VNO	Vornado Realty Trust	Real Estate
477	VMC	Vulcan Materials	Materials
478	WMT	Wal-Mart Stores	Consumer Staples
479	WBA	Walgreens Boots Alliance	Consumer Staples

480	WM	Waste Management Inc.	Industrials
481	WAT	Waters Corporation	Health Care
482	WEC	Wec Energy Group Inc	Utilities
483	WFC	Wells Fargo	Financials
484	HCN	Welltower Inc.	Real Estate
485	WDC	Western Digital	Information Technology
486	WU	Western Union Co	Information Technology
487	WRK	WestRock Company	Materials
488	WY	Weyerhaeuser Corp.	Real Estate
489	WHR	Whirlpool Corp.	Consumer Discretionary
490	WFM	Whole Foods Market	Consumer Staples
491	WMB	Williams Cos.	Energy
492	WLTW	Willis Towers Watson	Financials
493	WYN	Wyndham Worldwide	Consumer Discretionary
494	WYNN	Wynn Resorts Ltd	Consumer Discretionary
495	XEL	Xcel Energy Inc	Utilities
496	XRX	Xerox Corp.	Information Technology
497	XLNX	Xilinx Inc	Information Technology
498	XL	XL Capital	Financials
499	XYL	Xylem Inc.	Industrials
500	YHOO	Yahoo Inc.	Information Technology
501	YUM	Yum! Brands Inc	Consumer Discretionary
502	ZBH	Zimmer Biomet Holdings	Health Care
503	ZION	Zions Bancorp	Financials
504	ZTS	Zoetis	Health Care

Symbol	Name	Sector	Cluster	
0	MMM	3M Company	Industrials	Cluster 8
1	ABT	Abbott Laboratories	Health Care	Cluster 13
2	ABBV	AbbVie	Health Care	Cluster 13
3	ACN	Accenture plc	Information Technology	Cluster 10
4	ATVI	Activision Blizzard	Information Technology	Cluster 8
5	AYI	Acuity Brands Inc	Industrials	Cluster 11
6	ADBE	Adobe Systems Inc	Information Technology	Cluster 9
7	AAP	Advance Auto Parts	Consumer Discretionary	Cluster 11
8	AES	AES Corp	Utilities	Cluster 6
9	AET	Aetna Inc	Health Care	Cluster 0
10	AMG	Affiliated Managers Group Inc	Financials	Cluster 10
11	AFL	AFLAC Inc	Financials	Cluster 10
12	A	Agilent Technologies Inc	Health Care	Cluster 0
13	APD	Air Products & Chemicals Inc	Materials	Cluster 5
14	AKAM	Akamai Technologies Inc	Information Technology	Cluster 6
15	ALK	Alaska Air Group Inc	Industrials	Cluster 14
16	ALB	Albemarle Corp	Materials	Cluster 0
17	ALXN	Alexion Pharmaceuticals	Health Care	Cluster 5
18	ALLE	Allegion	Industrials	Cluster 10
19	AGN	Allergan, Plc	Health Care	Cluster 12
20	ADS	Alliance Data Systems	Information Technology	Cluster 5
21	LNT	Alliant Energy Corp	Utilities	Cluster 4
22	ALL	Allstate Corp	Financials	Cluster 9
23	GOOG L	Alphabet Inc Class A	Information Technology	Cluster 10
24	GOOG	Alphabet Inc Class C	Information Technology	Cluster 13
25	MO	Altria Group Inc	Consumer Staples	Cluster 3
26	AMZN	Amazon.com Inc	Consumer Discretionary	Cluster 10
27	AEE	Ameren Corp	Utilities	Cluster 13
28	AAL	American Airlines Group	Industrials	Cluster 4
29	AEP	American Electric Power	Utilities	Cluster 4
...	...	...	...	...
475	V	Visa Inc.	Information Technology	Cluster 0
476	VNO	Vornado Realty Trust	Real Estate	Cluster 2
477	VMC	Vulcan Materials	Materials	Cluster 3
478	WMT	Wal-Mart Stores	Consumer Staples	Cluster 4
479	WBA	Walgreens Boots Alliance	Consumer Staples	Cluster 12
480	WM	Waste Management Inc.	Industrials	Cluster 0
481	WAT	Waters Corporation	Health Care	Cluster 10
482	WEC	Wec Energy Group Inc	Utilities	Cluster 14
483	WFC	Wells Fargo	Financials	Cluster 14
484	HCN	Welltower Inc.	Real Estate	Cluster 6
485	WDC	Western Digital	Information Technology	Cluster 8

322

486	WU	Western Union Co	Information Technology	Cluster 6
487	WRK	WestRock Company	Materials	Cluster 10
488	WY	Weyerhaeuser Corp.	Real Estate	Cluster 5
489	WHR	Whirlpool Corp.	Consumer Discretionary	Cluster 3
490	WFM	Whole Foods Market	Consumer Staples	Cluster 10
491	WMB	Williams Cos.	Energy	Cluster 3
492	WLTW	Willis Towers Watson	Financials	Cluster 13
493	WYN	Wyndham Worldwide	Consumer Discretionary	Cluster 0
494	WYNN	Wynn Resorts Ltd	Consumer Discretionary	Cluster 13
495	XEL	Xcel Energy Inc	Utilities	Cluster 10
496	XRX	Xerox Corp.	Information Technology	Cluster 4
497	XLNX	Xilinx Inc	Information Technology	Cluster 10
498	XL	XL Capital	Financials	Cluster 10
499	XYL	Xylem Inc.	Industrials	Cluster 13
500	YHOO	Yahoo Inc.	Information Technology	None
501	YUM	Yum! Brands Inc	Consumer Discretionary	Cluster 6
502	ZBH	Zimmer Biomet Holdings	Health Care	Cluster 6
503	ZION	Zions Bancorp	Financials	Cluster 8
504	ZTS	Zoetis	Health Care	Cluster 13

Bibliography

Aach, J. & Church, G. (2001). “Aligning gene expression series with time warping algorithms”. Bioinformatics. Vol. 17, pp. 495-508

Abdi, H. (1994). A neural Network Primer. Journal of Biological System, 2(3), 247-283.

Adams, R.P., D. J.C. MacKay. (2007). “Bayesian Online Changepoint Detection”, arXiv:0710.3742

Agrawal, Ajay, Joshua Gans, and Avi Goldfarb. (2017). “How AI Will Change the Way We Make Decisions”, Harvard Business Review, Decision Making.

Abrigo, Michael R.M. Abrigo, and Inessa Love. (2016). “Estimation of Panel Vector Autoregression in Stata: a Package of Programs”, University of Hawai`i at Mānoa

Ahlquist, J.S. and C. Breunig. (2009). “Country Clustering in Comparative Political Economy”, MPIfG Discussion Paper 09/5, Max Planck Institute for the Study of Societies, Cologne.

American Society of Mechanical Engineers (ASME). (2012). “Forging a New Nuclear Safety Construct: The ASME Presidential Task Force on Response to Japan Nuclear Power Plant Events”.

Anandkumar, Animashree, Rong Ge, Daniel Hsu, Sham M. Kakade, Matus Telgarsky. (2014). “Tensor Decompositions for Learning Latent Variable Models”, Journal of Machine Learning Research 15 (2014) 2773-2832.

Andreasen, M.M. (2008). “Non-linear DSGE Models, The Central Difference Kalman Filter, and The Mean Shifted Particle Filter”.

Antunes, R., and V. Gonzalez. (2015). “A Production Model for Construction: A Theoretical Framework”. Buildings. 5 (1): 209–228. doi:10.3390/buildings5010209

Antunes, R., and V. Gonzalez, and K. Walsh. (2015). “Identification of repetitive processes at steady- and unsteadystate:Transfer function” Proc. 23rd Ann. Conf. of the Int’l. Group for Lean Construction, 28-31 July, Perth, Australia, pp. 793-802.

Arulampalam, M. Sanjeev, Simon Maskell, Neil Gordon, and Tim Clapp. (2002). “A Tutorial on Particle Filters for Online Nonlinear/Non-Gaussian Bayesian Tracking”, IEEE TRANSACTIONS ON SIGNAL PROCESSING, VOL. 50, NO. 2.

Ashvin Ahuja, Kevin Wiseman, Murtaza Syed. (2010). “Assessing Country Risk—Selected Approaches”. Reference Note, Strategy, Policy, and Review Departement, IMF Technical Notes and Manual

Ayyub, Bilal, and George Klir. (2006). “Uncertainty Modelling and Analysis in Engineering and the Sciences, Chapman and Hall Press.

Ayyub, Bilal. (1992). “Generalized treatment of uncertainties in structural engineering, in Analysis and Management of Uncertainty: Theory and Applications”, Ayyub, B. and Gupta, M., Eds., Elsevier, New York, pp. 235-246.

Ayyub, B., McGill, W.L., Kaminskiy, M. (2007). “Critical asset and portfolio risk analysis for homeland security: An all hazards framework”, Risk Analysis, 27(3), 789-801

Ayyub, B., Gupta, M.M., and Kanal, L.N. (1992). “Analysis and Management of Uncertainty: Theory and Applications”, Elsevier New York

Ayyub, B., and Gupta, M.M,Eds. (1997). “Uncertainty Analysis in Engineering and the Sciences: Fuzzy Logic, Statistics, and Neural Network Approach”, Kluwer Academic, Dordrecht.

Ayyub, B.M. (1994). “The nature of uncertainty in structural engineering, in Uncertainty Modelling and Anlaysis: Theory and Applications”, Ayyub, B.M., and Gupta, M.M, Eds., North Holland/Elsevier, Amserdam, pp. 195-210.

Ayyub, Bilal. (2014). “Risk Analysis in Engineering and Economics”. (2014). Taylor and Francis

Baldi, Pierre. (2012). “Autoencoders, Unsupervised Learning, and Deep Architectures”, JMLR: Workshop and Conference Proceedings 27:37–50, 2012 Workshop on Unsupervised and Transfer Learning

Baldi, P. and Z. Lu. (1999). “Complex-Valued Autoencoders”, arXiv:1108.4135

Ball, L. (2014). “Long-term damage from the Great Recession in OECD countries," European Journal of Economics and Economic Policies: Intervention, Edward Elgar, vol. 11(2), pages 149-160.

Ballesteros, M., C. Dyer, and N. A. Smith. (2015). “Improved transition-based parsing by modeling characters instead of words with lstms,” arXiv preprint arXiv:1508.00657.

Bandara, K., C. Bergmeira, and S. Smylb. (2017). “Forecasting Across Time Series Databases using Long Short-Term Memory Networks on Groups of Similar Series”, arXiv:1710.03222v1 [cs.LG] 9 Oct 2017.

Bank of International Settlements (BIS). (2010). “Developments in Modelling Risk Aggregation”, Basel Committee on Banking Supervision.

Bao W, Yue J, Rao Y. (2017). A deep learning framework for financial time series using stacked autoencoders and long-short term memory. PLoS ONE 12(7): e0180944. https://doi.org/10.1371/journal.pone.0180944

Bayes, Thomas & Price, Richard. (1763). "An Essay towards solving a Problem in the Doctrine of Chance. By the late Rev. Mr. Bayes, communicated by Mr. Price, in a letter to John Canton, A. M. F. R. S." (PDF). Philosophical Transactions of the Royal Society of London. 53 (0): 370–418. doi:10.1098/rstl.1763.0053.

Bechtel. (2017). Managing Risks in Mega-EPC Projects – Where do we start?

Ben-David, S., J Blitzer, K Crammer, A Kulesza, F Pereira, JW Vaughan. (2017). "A theory of learning from different domains", Machine learning 79 (1), 151-175

Bengio, Y., N. Boulanger-Lewandowski, and R. Pascanu. (2013). "Advances in optimizing recurrent networks," in Acoustics, Speech and Signal Processing (ICASSP), 2013 IEEE International Conference on. IEEE, pp. 8624–8628.

Bengio, Y., P. Simard, and P. Frasconi. (1994). "Learning long-term dependencies with gradient descent is difficult," IEEE transactions on neural networks, vol. 5, no. 2, pp. 157–166, 1994.

Bengio, Y., Y. LeCun et al. (2007). "Scaling learning algorithms towards ai," Large-scale kernel machines, vol. 34, no. 5, pp. 1–41, 2007.

Bengio, Y. (2009). "Learning deep architectures for AI", Foundations and Trends in Machine Learning, Vol 2:1.

Bengio, Y., I.J., Goodfellow, I. J., and A. Courville, A. (2015). "Deep Learning". Nature, 521, 436-444.

Bertolini, L. and W. Salet. (2007). "Coping with Complexity and Uncertainty in Mega Projects: Linking Strategic Choices and Operational Decision Making", Sustainable Development Challenges for Mega Urban Transport Projects

Bertoluzzo, F. and M. Corazza. (2012). "Reinforcement Learning for automatic financial trading: Introduction and some applications", Working Papers, Department of Economics, Ca' Foscari University of Venice No. 33/WP/2012.

Bertoluzzo, F., and M. Corazza. (2014). "Reinforcement Learning for Automated Financial Trading: Basics and Applications", In S. Bassis et al. (eds.) Recent Advances of Neural Network Models and Applications, Proceedings of the 23rd Workshop of the Italian Neural Networks Society (SIREN), May 23-25, Vietri sul Mare, Salerno, Italy

Bertolini, B. and W. Salet. (2010). "Planning Concepts for Cities in Transition:

Regionalization of Urbanity in the Amsterdam Structure Plan, Planning Theory & Practice", 4:2, 131-146, DOI: 10.1080/14649350307977

Bianchi, F.M., E. Maiorinob, M.C. Kampffmeyera, A. Rizzib, R. Jenssena. (2017). "An overview and comparative analysis of Recurrent Neural Networks for Short Term Load Forecasting", arXiv:1705.04378v1 [cs.NE] 11 May 2017.

Bishop, C. M. (2006). Pattern recognition and machine learning. springer.

Brockmann, Michael, Michael Kalkbrener. (2007). "On the Aggregation of Risk", Risk Analytics & Instruments, Deutsche Bank, Frankfurt.

Brookes, N., and G. Locatelli. (2014). "Power Plants as Megaprojects: Using Empirics to Shape Policy, Planning and Construction Management", *Utilities Policy*, 36. 57 - 66. ISSN 0957-1787, 2014.

Brousseau, Kenneth R., Michael J. Driver, Gary Hourihan, and Rikard Larsson. (2006). "The Seasoned Executive's Decision-Making Style", Harvard Business Review, Organizational Culture.

Burgard, W., A. Karwath, B. Nebel, and M. Riedmiller, "Foundations of AI: Unsupervised Learning", Teaching Slides

Burges, C. (1998). "A tutorial on support vector machines for pattern recognition", Knowledge Discovery and Data Mining 2(2): 121–167.

Breiman, L. (2004). "Population theory for boosting ensembles", Annals of Statistics, Volume 32, Number 1 (2004), 1-11.

Caglar Gulcehre KyungHyun Cho, Yoshua Bengio. (2014). "Empirical Evaluation of Gated Recurrent Neural Networks on Sequence Modeling", arXiv:1412.3555v1 [cs.NE].

Canova, F. and M. Ciccarelli. (2013). "Panel Vector Autoregressive Models A Survey", European Central Bank Working Paper SerieS NO 1507.

Cassisi, C., P. Montalto, M.A. Aliotta, Antonio, A., Canata, and A. (2012). "Similarity Measures and Dimensionality Reduction Techniques for Time Series Data Mining", In book: Advances in Data Mining Knowledge Discovery and Applications Chapter: 3, Editors: InTech, September 2012.

Chan-Lau, Jorge A. (2017). "Lasso Regressions and Forecasting Models in Applied Stress Testing", IMF Working Paper WP/17/108.

Chan, L.-W. and C.-C. Szeto, 1999, "Training recurrent network with blockdiagonal approximated levenberg-marquardt algorithm," in Neural Networks, IJCNN'99.

International Joint Conference on, vol. 3. IEEE, 1999, pp. 1521–1526.

Cho, K., B. Van Merri¨enboer, D. Bahdanau, and Y. Bengio. (2014). “On the properties of neural machine translation: Encoder-decoder approaches,” arXiv preprint arXiv:1409.1259.

Chen Y, Lin Z, Zhao X, Wang G, Gu Y. (2014). “Deep Learning-Based Classification of Hyperspectral Data”. IEEE J Sel Top Appl Earth Observ Remote Sens.; 7(6):2094–107.

Choi, E., M.T. Bahadori, E. Schuetz, W. Stewart, and J. Sun. (2016). “Doctor AI: Predicting Clinical Events via Recurrent Neural Networks”, Proceedings of the 1st Machine Learning for Healthcare Conference: 301–318.

Chen, T. 2014. “Introduction to Boosted Trees TexPoint fonts used in EMF”, University of Washington, Oct. 22 2014.

Chen, T, and C. Carlos Guestrin. (2016). “XGBoost: A Scalable Tree Boosting System”, KDD ’16, August 13-17. (2016). San Francisco, CA, USA

Choi, S., D. Furceri, P. Loungani, S. Mishra, and M. Poplawski-Ribeiro. (2017). “Oil Prices and Inflation Dynamics: Evidence from Advanced and Developing Economies”, IMF Working Paper No. 17/196, 2017.

Coats, D.R. (2017). “Worldwide Threat Assessment of the US Intelligence Community”, Senate Select Committee on Intelligence, Director of National Intelligence.

Chung, J., C. Gulcehre, K. Cho, and Y. Bengio. (2014). “Empirical evaluation of gated recurrent neural networks on sequence modeling,” arXiv preprint arXiv:1412.3555.

Cox, D.R. (1958). “The Regression Analysis of Binary Sequences”, Journal of the Royal Statistical Society. Series B (Methodological), Vol. 20, No. 2

Cox, D.R. (1972). “Regression Models and Life-Tables”, Journal of the Royal Statistical Society, Series B. 34 (2): 187–220. JSTOR 2985181.

Cuadra, L., M.P. Orcid, J.C. Nieto-Borge, and S. Salcedo-Sanz. (2017). “Optimizing the Structure of Distribution Smart Grids with Renewable Generation against Abnormal Conditions: A Complex Networks Approach with Evolutionary Algorithms”, Energies 2017, 10(8), 1097; doi:10.3390/en10081097

Dablemont, S., V. Bellegem, M. Verleysan. (2017). “Forecasting "High" and "Low" of financial time series by Particle systems and Kalman filters”

Dasarathy, B. (1991). “Nearest Neighbor Pattern Classification Techniques”, IEEE Computer Society Press, Los Alamitos, CA.

Davies, A., M. Dodgson, D.M. Gann, and S.C. MacAulay. (2017). “Five Rules for Managing Large, Complex Projects”, Magazine, Research Feature.

Del Moral P., and L. Miclo. (2000). “Branching and Interacting Particle Systems Approximations of Feynman-Kac Formulae with Applications to Non-Linear Filtering”

Del Moral, P., and A. Doucet. (2009). “Particle Methods: An introduction with applications”, HAL-INRIA RR-6991 [50p] (2009).

Del Moral, P., P. Hu, L. Wu. (2011). “On the concentration properties of Interacting particle processes”, HAL-INRIA RR-7677 (2011).

Deng, X., Pheng, L. S., & Zhao, X. (2014). “Project system vulnerability to political risks in international construction projects: the case of Chinese contractors”, *Project Management Journal*, 45(2), 20–33, 2014

Dey, R. and F.M. Salem. (2017). “Gate-Variants of Gated Recurrent Unit (GRU) Neural Networks”, arXiV 1701.05923

DiMatteo, I., Genovese, C. R., and Kass, R. E. (2001). “Bayesian curve-fitting with free-knot splines”, Biometrika 88, 1055–1071.

Ding X, Zhang Y, Liu T, Duan J, editors. (2015). Deep learning for event-driven stock prediction. International Conference on Artificial Intelligence.

Ding, A., and X. He. (2003). “Backpropagation of pseudo-errors: neural networks that are adaptive to heterogeneous noise,” SUBMITTED TO IEEE TRANSACTIONS ON NEURAL NETWORKS 16, vol. 14, no. 2, pp. 253–262, 2003.

Dong, Xin Luna, Evgeniy Gabrilovich, Geremy Heitz, Wilko Horn, Ni Lao, Kevin Murphy ,Thomas Strohmann, Shaohua Sun, Wei Zhang. (2014). “Knowledge Vault: A Web-Scale   Approach to Probabilistic Knowledge Fusion”, Google.

Dybowski, R. and S. Roberts. (2000). “Confidence intervals and prediction intervals for feed-forward neural networks,” in Clinical Applications of Artificial Neural Networks.

Efron, B., 1979, “Bootstrap methods: Another look at the jackknife,” The Annals of Statistics, vol. 7, no. 1, pp. 1–26, 1979.

Ellis C.J., and J. Fender. (2011). “Information Cascades and Revolutionary Regime

Transitions", *The Economic Journal*, Volume 121, Issue 553, Pages 763–792.

Elman, Jeffrey L. (1990). "Finding Structure in Time". Cognitive Science. 14 (2): 179–211. doi:10.1016/0364-0213(90)90002-E.

Erb, Claude B. Campbell R. Harvey Tadas E. Viskanta. (2017). "The Influence of Political, Economic and Financial Risk on Expected Fixed Income Returns"

Espinoza, R. & F. Fornari, and Lombardi, J. Marco. (2009). "The role of financial variables in predicting economic activity", Working Paper Series 1108, European Central Bank.

European Strategy and Policy Analysis System (ESPAS) (2017), The Making of the New Geopolitical Order, Conference Paper

Embrechts, Paul and Giovanni Puccetti. (2009). "Risk Aggregation"

Farmer, R.E. (2012). "The stock market crash of 2008 caused the Great Recession: Theory and evidence", Journal of Economic Dynamics & Control 36 (2012) 693–707

Federal Reserve Board of Governors. (2000). Overview of Risk Management in Trading Activities, 2000

Ferris and M. Solís. (2013). "Earthequale, Tsunami, Meltdown – The Triple Disaster's Impact on Japan, Impact on the World, *Brookings*.

Ferrari, A., P. Ginis, M. Hardegger, F. Casamassima, L. Rocchi et al. (2015). "A Mobile Kalman-Filter Based Solution for the Real-Time Estimation of Spatio-Temporal Gait Parameters," IEEE Transactions on Neural Systems and Rehabilitation Engineering, no. 99.

Fischhendler, I., G. Cohen-Blankshtain, Y. Shuali, and M. Boykoff. (2013). "Communicating mega-projects in the face of uncertainties: Israeli mass media treatment of the Dead Sea Water Canal", *Public Understanding of Science,* Research Article.

Flyvbjerg, B. (2014). "What You Should Know about Megaprojects and Why: An

Flyvbjerg, B. (2006). "From Nobel Prize to Project Management: Getting Risks Right", Project Management Journal, vol. 37, no. 3, August, pp. 5-15.

Flyvbjerg, B. (2009). "Survival of the Unfittest: Why the Worst Infrastructure Gets Built, and What We Can Do about It", *Oxford Review of Economic Policy*, vol. 25, no. 3, pp. 344–367.

Flyvbjerg, B. (2011). "Over Budget, Over Time, Over and Over Again: Managing

Major Projects”, in Peter W. G. Morris, Jeffrey K. Pinto, and Jonas Söderlund, eds., The Oxford Handbook of Project Management (Oxford: Oxford University Press), pp. 321-344.

Flyvbjerg, B. (2012). “Why Mass Media Matter, and How to Work with Them: Phronesis and Megaprojects”, in Flyvbjerg, B., Landman, T, Schram, S., eds., Real Social Science: Applied Phronesis (Cambridge: Cambridge University Press), pp. 95-12.

Fox, Anthony, C. Eichelberger, J. Hughes, and S. Lyon. (2013). “Spatio-temporal Indexing in Non-relational Distributed Databases”, 2013 IEEE International Conference on Big Data.

Freedman, D.A. (2009). Statistical Models: Theory and Practice. Cambridge University Press.

Gal, Y. (2016). “Uncertainty in Deep Learning”, PhD Dissertation, Department of Engineering University of Cambridge Gonville and Caius College September 2016

Gal, Y. and Z. Ghahramani. (2016a). “A Theoretically Grounded Application of Dropout in Recurrent Neural Networks”, arXiv:1512.05287v5 [stat.ML] 5 Oct 2016 University of Cambridge

Gal, Y. and Z. Ghahramani. (2016b). “Dropout as a Bayesian Approximation: Representing Model Uncertainty in Deep Learning”, arXiv:1506.02142v6 [stat.ML] 4 Oct 2016

Gal, Y. and Z. Ghahramani. (2017). “On Modern Deep Learning and Variational Inference”, Advances in Approximate Bayesian Inference NIPS 2017 Workshop; December 8, 2017.

Galindo, and P. Tamayo. (1997). “Credit Risk Assessment using Statistical and Machine Learning: Basic Methodology and Risk Modeling Applications”.

Gamboa, J. (2017). “Deep Learning for Time-Series Analysis”, arXiv:1701.01887v1

Geert Bekaert, Campbell R. Harvey, Christian T. Lundblad, Stephan Siegel. (2015). “Political Risk and International Valuation”

Georgoff, D.M., Murdick, R.G. (1986). “Managers Guide to Forecasting”, Economics, Harvard Business Review

Giordano, F., M. La Rocca, and C. Perna. (2007). “Forecasting nonlinear time series with neural network sieve bootstrap,” Computational Statistics & Data Analysis, vol. 51, no. 8, pp. 3871–3884, May 2007.

Goodfellow, I., Y. Bengio, and A. Courville, Deep Learning. MIT Press. (2016). http://www.deeplearningbook.org.

Goodfellow, I.J., J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, Y. Bengio. (2014). “Generative Adversarial Networks”, Neural Information Processing Systems Conference 5423

Graves A., and J. Schmidhuber. (2005). “Framewise Phoneme Classification with Bidirectional LSTM and Other Neural Network Architectures”

Graves, A., S. Fernández, and J. Schmidhuber. (2005). “Bidirectional LSTM networks for improved phoneme classification and recognition”, Artificial Neural Networks: Formal Models and Their Applications–ICANN 2005. Springer Berlin Heidelberg, 2005. 799-804.

Graves, A., and J. Schmidhuber. (2009). “Offline Handwriting Recognition with Multidimensional Recurrent Neural Networks”, Advances in Neural Information Processing Systems 22, NIPS'22, pp 545–552, Vancouver, MIT Press, 2009.

Graves, A., A.-r. Mohamed, and G. Hinton. (2013). “Speech recognition with deep recurrent neural networks,” in Acoustics, speech and signal processing (icassp), 2013 ieee international conference on. IEEE. (2013). pp. 6645–6649.

Graves, A., and J. Schmidhuber. (2009). “Offline handwriting recognition with multidimensional recurrent neural networks,” in Advances in neural information processing systems. (2009). pp. 545–552.

Graves, A., S. Fernandez, and J. Schmidhuber. (2007). “Multi-dimensional recurrent neural networks,” arXiv preprint arXiv:0705.2011v1, 2007.

Giordano, F., M. La Rocca, and C. Perna. (2007). “Forecasting nonlinear time series with neural network sieve bootstrap,” Computational Statistics & Data Analysis, vol. 51, no. 8, pp. 3871–3884, May 2007.

Goudet, O., D. Kalainathan, P. Caillou, I. Guyon, D. Lopez-Paz, M. Sebag. (2017). “Causal Generative Neural Networks”, arXiv.org > stat > arXiv:1711.08936.

Hadad, N., L. Wolf1, and M. Shahar. (2017). “Two-Step Disentanglement for Financial Data”, arXiv:1709.00199v1 [cs.LG] 1 Sep 2017

Hamilton, J. (1994). “Time Series Analysis”, Princeton University Press. Chapter 13, 'The Kalman Filter'

Harding, R. (2016). “Japan taxpayers foot $100bn bill for Fukushima disaster”, *Financial Times*.

Hagan, M. and M. Menhaj. (1994). “Training feedforward networks with the marquardt algorithm,” IEEE Transactions on Neural Networks, vol. 5,

Harvard Business Review, “The Four Phases of Project Management”, Harvard Business Review Staff, November 03, 2016.

Hassan, T. A. (2013). “Country Size, Currency Unions, and International Asset Returns”, The Journal of Finance, 68: 2269–2308. doi:10.1111/jofi.12081

Hassan, K., M. Maroney, Neal C., M. El-Sady, and T., Ahmad. (2003). “Country risk and stock market volatility, predictability, and diversification in the Middle East and Africa”, Economic Systems, 27, issue 1, p. 63-82.

Hastings, W.K. (1970). “Monte Carlo Sampling Methods Using Markov Chains and Their Applications”, Biometrika. 57 (1): 97–109. doi:10.1093/biomet/57.1.97. JSTOR 2334940. Zbl 0219.65008.

Hubbard, Douglas. (2009). The Failure of Risk Management: Why It's Broken and How to Fix It. John Wiley & Sons. p. 46.

Hayashi, Yuko Mizuhara, and Nobuo Suematsu. (2016). “Embedding time series data for classification”.

Haykin, S.. (1994). Neural networks: a comprehensive foundation. Prentice Hall PTR.

Heskes, T. (1997). “Practical confidence and prediction intervals,” in Neural Information Processing Systems, T. P. M. Mozer, M. Jordan, Ed., vol. 9. MIT Press, 1997, pp. 176–182.

Hinton, G.E. and R.R. Salakhutdinov.(2006). “Reducing the dimensionality of data with neural networks”, Science 313 (5786), 504-507.

Hinton, G.E., S. Osindero, and Y-W. The. (2006). “A fast learning algorithm for deep belief nets”, Neural Computation.

HIM DHARA, “A RIVER UNDER ARREST: A critique of the Luhri Hydro-electric Project on the Sutlej River”, Himachal Pradesh Environment Research and Action Collective, November 2011.

Hochreiter, S. and S., J¨urgen. (1997). Long Short-Term Memory, In Neural Computing.

Hochreiter, S., and J. Schmidhuber. (1999). “Long short-term memory,” Neural computation, vol. 9, no. 8, pp. 1735–1780, 1997. [55] F. A. Gers, J. Schmidhuber, and F. Cummins, “Learning to forget: Continual prediction with lstm,”.

Hochreiter, S., M. Heusel, and K. Obermayer. (2007). “Fast model-based protein homology detection without alignment”. Bioinformatics. 23 (14): 1728–1736.

Holopainen, Markus and Peter Sarlin. (2016). “Toward robust early-warning models: a horse race, ensembles and model uncertainty”, European Central Bank Working Paper No 1900.

Homeland Security. (2011). Risk Management Fundamental

Hopfield, J. J. (1988). “Neural Networks and Physical Systems with Emergent Collective Computational Abilities”, In Neurocomputing: Foundations of Research.

Hubbard, D. (2016). “The Failure of Risk Management: Why It's Broken and How to Fix It”, John Wiley & Sons. p. 46).

Hughes, N. (2014). “Applying reinforcement learning to economic problems”, Australian National University, November 14, 2014.

Ho, T. K. (1995).“Random decision forests”, in M. Kavavaugh and P. Storms (eds), Proc. Third International Conference on Document Analysis and Recognition, Vol. 1, IEEE Computer Society Press, New York, 278–282.

Hwang, J. T.W., and A. A. Ding. (1997). “Prediction intervals for artificial neural networks,” Journal of the American Statistical Association, vol. 92, no. 438, pp. 748–757, 1997.

International Monetary Fund. (2017). Global Financial Stability Report

ISO 31000:2009 Risk management -- Principles and guidelines

Jennings W. (2012) “Executive Politics, Risk and the Mega-Project Paradox”. In: Lodge M., Wegrich K. (eds) Executive Politics in Times of Crisis. The Executive Politics and Governance series. Palgrave Macmillan, London

Jacob, A., M. Rohrbach, T. Darrell, D. Klein. (2017). “Neural Module Networks”

James, G., D. Witten, T. Hastie, and R. Tibshirani. (2013). “An Introduction to Statistical Learning”, Springer Texts in Statistics.

Jiang, Z., D. Xu, and J. Liang. (2017). “A Deep Reinforcement Learning Framework for the Financial Portfolio Management Problem”, arXiv:1706.10059

Jordan, Michael I. (1997). “Serial Order: A Parallel Distributed Processing Approach”. Advances in Psychology. Neural-Network Models of Cognition. 121: 471–495. ISBN 9780444819314. doi:10.1016/s0166-4115(97)80111-2.

Jurafsky, D., and J.H. Martin. (2017). “Naive Bayes and Sentiment Classification”, Speech and Language Processing, Chapter 6.

Kalchbrenner, N., I. Danihelka, and A. Graves. (2015). “Grid long short-term memory,” arXiv preprint arXiv:1507.01526.

Kalman, R. E. (1960). “A New Approach to Linear Filtering and Prediction Problems”, Journal of Basic Engineering. 82: 35.

Kaufman, L. and P. Rousseeuw, P. (1990).“Finding Groups in Data: An Introduction to Cluster Analysis”, Wiley, New York.

Keogh, E. and C.A. Ratanamahatana. (2005). “Exact indexing of dynamic time warping”, Knowledge and Information Systems”, Volume 7, Issue 3, pp 358–386

Kolanovic, M. (2017). “Global Quantitative & Derivatives Strategy”, J.P.Morgan

Khosravi, A., Nahavandi, S., D. Creighton, and Atiya, A.F. (2010). “A Comprehensive Review of Neural Network-based Prediction Intervals and New Advances”, SUBMITTED TO IEEE TRANSACTIONS ON NEURAL NETWORKS

Khosravi, A., S. Nahavandi, and D. Creighton. (2010). “A prediction intervalbased approach to determine optimal structures of neural network metamodels,” Expert Systems with Applications, vol. 37, pp. 2377–2387, 2010

Khosravi, A., S. Nahavandi, D. Creighton, and A. F. Atiya. (2011). “A lower upper bound estimation method for construction of neural network-based prediction intervals,” IEEE Transactions on Neural Networks, vol. 22, no. 3, pp. 337 – 346, 2011.

Kosnic, S. (2005). “Terrorism and its impact on the construction industry”, University of Maryland College Park, Project Management.

Kroese, D.P, and R. Y. Rubinstein. (2008). “Simulation and the Monte Carlo Method”, Second Edition, Wiley Online Library

Krishnan, R., U. Shalit, D. Sontag. (2016). “Structured Inference Networks for Nonlinear State Space Models”, arXiv:1609.09869v2 [stat.ML]

Kulesza, A. and F. Pereira. (2008). “Structured learning with approximate inference”, In NIPS 20, pages 785–792. 2008.

Krizhevsky, A., I. Sutskever, and G. E. Hinton. (2012). “Imagenet classification with deep convolutional neural networks,” in Advances in neural information processing systems. (2012). pp. 1097–1105.

Kyongmin, Y.. (2017). “Model-free prediction of noisy chaotic time series by Deep Learning”, arXiv:1710.01693v1 [cs.LG] 29 Sep 2017.

Larsen, V.H. (2017). “Components of Uncertainty”, *Norges Bank Research*, Working Paper No. 5.

Laufer, A. (1997). “Simultaneous Management: Managing Projects in a Dynamic Environment”, American Management Association

Laeven, Luc and Fabián Valencia, Systemic Banking Crises Database: An Update

Laurens van der Maaten, Geoffrey Hinton. (2008). “Visualizing Data using t-SNE”, Journal of Machine Learning Research 9 2579-2605

Le, Q. V., N. Jaitly, and G. E. Hinton. (2015). “A simple way to initialize recurrent networks of rectified linear units,” arXiv preprint arXiv:1504.00941, 2015.

Locatellia, Giorgio, Giacomo Marianib, Tristano Sainatia, and Marco Grecob. (2017). “Corruption in public projects and megaprojects: There is an elephant in the room!”, International Journal of Project Management, Volume 35, Issue 3, Pages 252-268

LeCun, Y., X. Zhang, and J. Zhao. (2015). “Character-level Convolutional Networks for Text Classification”, arXiv.org > cs > arXiv:1509.01626v3

Lessard, Donald, and Roger Miller. (2001). “Understanding and Managing Risks in Large Engineering Projects”, MIT Sloan School of Management, Sloan Working Paper 4214-01.

Lijuan Cao and Francis E.H. Tay. (2001). Financial Forecasting Using Support Vector Machines, Neural Comput & Applic (2001)10:184–192, Springer-Verlag London Limited.

Lipton, Z.C., J. Berkowitz, C. Elkan. (2015). “A Critical Review of Recurrent Neural Networks for Sequence Learning”, arXiv:1506.00019v4 [cs.LG] 17 Oct 2015

Lockheed Martin. (2017). Sustainability.

Locatellia, Giorgio, Giacomo Marianib, Tristano Sainatia, and Marco Grecob. (2017). “Corruption in public projects and megaprojects: There is an elephant in the room!”, International Journal of Project Management, Volume 35, Issue 3, Pages 252-268.

Lopyrev, K. (2015). “Generating News Headlines with Recurrent Neural Networks”, Stanford Natural Language Processing Report

Loungani, P., S. Mishra, C. Papegeorgiou, and K. Wang. (2017). “World Trade in

Services: Evidence from a New Dataset", IMF Working Paper No. 17/17.

Ma, S., and C. Ji,1998, "A unified approach on fast training of feedforward and recurrent networks using em algorithm," IEEE transactions on signal processing, vol. 46, no. 8, pp. 2270–2274.

Machado, J.A.T., and M.E. Mata. (2015). "Analysis of World Economic Variables Using Multidimensional Scaling", PloS-https://doi.org/10.1371/journal.pone.0121277

MacQueen, J.B. (1967). "Some Methods for classification and Analysis of Multivariate Observations", Proceedings of 5-th Berkeley Symposium on Mathematical Statistics and Probability", Berkeley, University of California Press, 1:281-297

MacKay, D. J. C. (1992). "The evidence framework applied to classification networks", California Institute of Technology.

Matsugu, M., K. Mori, Y. Mitari, and Y. Kaneda. (2003). "Subject independent facial expression recognition with robust face detection using a convolutional neural network", Neural Netw. 2003 Jun-Jul;16(5-6):555-9.

Matzinger and Palter. (2015). "Megaprojects: The good, the bad, and the better", McKinsey Global Infrastructure Report

Mezei, Jozsef and Peter Sarlin. (2016). "RiskRank: Measuring interconnected risk"

Ministry of Economy, Trade and Industry. (2014). Strategic Energy Plan'

Menninghaus A, Mathias, Martin Breunig B, Elke Pulvermuller. (2016). "High-Dimensional Spatio- Temporal Indexing", Open Journal of Databases (OJDB) Volume 3, Issue 1.

Miller Roger, Donald R. Lessard, Pascale Michaud and Serghei Floricel. (2001). "The Strategic Management of Large Engineering Projects: Shaping Institutions, Risks, and Governance", MIT Press.

Mikolov, T., A. Joulin, S. Chopra, M. Mathieu, and M. Ranzato. (2014). "Learning longer memory in recurrent neural networks," arXiv preprint arXiv:1412.7753.

Mikolov, T., M. Karafi´at, L. Burget, J. Cernock'y, and S. Khudanpur. (2010). "Recurrent neural network based language model." in Interspeech, vol. 2. (2010). p. 3.

Mishra, S., and B. Ayyub. (2017). "Entropy for Quantifying Risk and Uncertainty in Economic Disparity"

Mishra, S., and B. Ayyub. (2017). “Entropy-Based Economic Disparity Profiles: The Tale of Four US Cities Baltimore, Detroit, San Francisco, and Washington, DC”
Mishra, S., I. Tewari, and S. Toosi. (2017). “Economic Complexity and the Service Revolution”

Mishra, S., M. Cader, K. Roster, A. Zaccaria, and L. Pietranero. (2017). “Integrating Services in the Economic Fitness Approach”

Mishra, S., P. Loungani, C. Papageiorgiou, and K. Wang. (2017). “World Trade in Services Evidence from a New Dataset”, IMF Working Paper 17/17

Modarres, M. (2017). Probabilistic Risk Assessment and Management: Tools, Techniques, and Applications, Presentation.

Mosteller, F., and D.L. Wallace. (1964). “Inference and Disputed Authorship”

Moon, T., H. Choi, H. Lee, and I. Song. (2015). “Rnndrop: A novel dropout for rnns in asr,” in Automatic Speech Recognition and Understanding (ASRU), 2015 IEEE Workshop on. IEEE. (2015). pp. 65–70.

Murphy, K. (2010). “Machine Learning”, MIT Press

National Council on Measurement in Education. Correlation Coefficient.

Nix, D., and A. Weigend. (1994). “Estimating the mean and variance of the target probability distribution,” in IEEE International Conference on Neural Networks, 1994.

Nuclear Energy Agency. (2017). “Strategic Plan Details Challenges Facing Nuclear Industry”.

Olah, C. (2015). “Understanding LSTM networks”, August 27, 2015

Opitz, T.. (2016). “Modeling asymptotically independent spatial extremes based on laplace random fields,” Spatial Statistics, vol. 16, pp. 1– 18, 2016.

Organization for Economic Co-operation and Development (OECD). (2015a). “Towards a Framework for the Governance of Infrastructure”, Public Governance and Territorial Development Directorate.

Organization for Economic Co-operation and Development (OECD). (2015b). “Infrastructure Financing Instruments and Incentives”.

P´erez-Ortiz, J. A., F. A. Gers, D. Eck, and J. Schmidhuber. (2003). “Kalman filters improve lstm network performance in problems unsolvable by traditional recurrent nets," Neural Networks, vol. 16, no. 2, pp. 241– 250.

Pham, V., T. Bluche, C. Kermorvant, and J. Louradour. (2014). "Dropout improves recurrent neural networks for handwriting recognition," in Frontiers in Handwriting Recognition (ICFHR), 2014 14th International Conference on. IEEE. (2014). pp. 285–290.

Papadopoulos, G., P. Edwards, and A. Murray, "Confidence estimation methods for neural networks: a practical comparison," IEEE Transactions on Neural Networks, vol. 12, no. 6, pp. 1278–1287, 2001.

Pascanu, R., T. Mikolov, and Y. Bengio. (2013). "On the difficulty of training recurrent neural networks," in International Conference on Machine Learning, pp. 1310–1318.

Perolat, J., J.Z. Leibo, V. Zambaldi, C. Beattie, K. Tuyls, and T. Graepel. (2017). "A multi-agent reinforcement learning model of common-pool resource appropriation", arXiv.org > cs > arXiv:1707.06600

Platon, V., S. Frone, A. Constantinescu. (2014). "Financial and Economic Risks to Public Projects", *Procedia Economics and Finance*, Volume 8, Pages 204-210.

Puskorius, G. V., and L. A. Feldkamp. (1994). "Neurocontrol of nonlinear dynamical systems with kalman filter trained recurrent networks," IEEE Transactions on neural networks, vol. 5, no. 2, pp. 279–297.

Qing Li and LiLing Jiang, Ping Li, and Hsinchun Chen. (2015). "Tensor-Based Learning for Predicting Stock Movements", Proceedings of the Twenty-Ninth AAAI Conference on Artificial Intelligence

Rabiner, L., Rosenberg, A. & Levinson, S. (1978). "Considerations in Dynamic Time Warping algorithms for discrete word recognition", IEEE Trans. Acoustics Speech, and Signal Proc., Vol. ASSP-26, pp. 575-582.

Rana R., J. Epps, R. Jurdak, X. Li, R. Goecke, M. Breretonk, and J. Soar. (2016). "Gated Recurrent Unit (GRU) for Emotion Classification from Noisy Speech", arXiv:1612.07778v1 [cs.HC] 13 Dec 2016

Ranger, S. (2014). "iPhone, AI and big data: Here's how Apple plans to protect your privacy | ZDNet". ZDNet. Retrieved 2017-06-27.

Ransbotham, S., D. Kiron, P. Gerbert, and M. (2017). "Reshaping Business with Artificial Intelligence: Closing the Gap between Ambition and Action", *MIT Sloan Management Review*.

Ratanamahatana, Chotirat Ann, and Eamonn Keogh. (2009). “Everything you know about Dynamic Time Warping is Wrong”, University of California Riverside

Rencher, A.C. and F.W. Christensen. (2012). “Chapter 10, Multivariate regression – Section 10.1, Introduction”, Methods of Multivariate Analysis, Wiley Series in Probability and Statistics, 709 (3rd ed.), John Wiley & Sons, p. 19, ISBN 9781118391679.

Rivals, I., and L. Personnaz. (2000). “Construction of confidence intervals for neural networks based on least squares estimation,” Neural Networks, vol. 13, no. 4-5, pp. 463–484, Jun. 2000.

Ruder, S.. (2016). “An overview of gradient descent optimization algorithms,” arXiv preprint arXiv:1609.04747.

Russel, S. and P. Norvig. (1995). “Artificial Intelligence: A Modern Approach”, 3rd Edition, Prentice Hall 1995.

Riedel, Sebastian, Limin Yao, Andrew McCallum, Benjamin M. Marlin. (2014). “Relation Extraction with Matrix Factorization and Universal Schemas”

Safer, Alan M. and Bogdan M. Wilamowski. (2011). “Using Neural Networks to Predict Abnormal Returns of Quarterly Earnings”

Sak, H., A. Senior, and F. Beaufays. (2014). “Long Short-Term Memory Based Recurrent Neural Network Architectures for Large Vocabulary Speech Recognition”, arXiv.org > cs > arXiv:1402.1128

Sakoe,H. and Chiba, S. (1978). Dynamic programming algorithm optimization for spoken word recognition. IEEE Trans. on Acoust., Speech, and Signal Process., ASSP 26, 43-49.

Salazar, J.G. (2000). “Damming the Child of the Ocean: The Three Gorges Project”, Journal of Environment and Development, vol. 9, no. 2, p. 173.

Salehinejad, H., S. Sankar, J. Barfett, E. Colak, and S. Valaee. (2017). “Recent Advances in Recurrent Neural Networks”, arXiv:1801.01078v3 [cs.NE] 22 Feb 2018.

Salet, W., M, Bertolini, L, Giezen. (2012). “Complexity and uncertainty: Problem or asset in decision making of mega infrastructure projects?” International Journal of Project Management in press Google Scholar.

Sarlin, P. (2010). “Visual monitoring of financial stability with a self-organizing neural network”, In: Proceedings of the International Conference on Intelligent Systems Design and Applications (ISDA 10). IEEE Press, Cairo, Egypt, pp. 248–253.

Sarlin, P. (2011). “Sovereign debt monitor: A visual self-organizing maps approach”, In: Proceedings of IEEE Symposium on Computational Intelligence for Financial Engineering and Economics (CIFEr 11). IEEE Press, Paris, France, pp. 1–8.

Sarlin, P. (2013). “Self-organizing time map: An abstraction of temporal multivariate patterns”, Neurocomputing 99(1), 496–508.

Sarlin, P. (2014a). “Data and dimension reduction for visual financial performance analysis”, Information Visualiza- tion, forthcoming, doi: 10.1177/1473871613504102.

Sarlin, P. (2014b). Mapping Financial Stability. Computational Risk Management Series. Springer Verlag.

Sarlin, P., Marghescu, D. (2011). “Visual predictions of currency crises using self-organizing maps”, Intelligent Systems in Accounting, Finance and Management 18(1), 15–38.

Sarlin, P., Peltonen, T. (2013). “Mapping the state of financial stability”, Journal of International Financial Markets, Institutions & Money 26, 46–76.

Sarlin, P.. (2013a). “Decomposing the global financial crisis: A self-organizing time map”, Pattern Recognition Letters 34, 1701–1709.

Sarlin, Peter. (2015). “Macroprudential oversight, risk communication and visualization”, European Central Bank Working Paper Series No 1768.

Schmidhuber. J. (2014). “Deep Learning in Neural Networks: An Overview”, Neural Networks, Volume 61, January 2015, Pages 85-117 (DOI: 10.1016/j.neunet.2014.09.003), published online in 2014.

Schuster, M. and K. K. Paliwal. (1997). “Bidirectional Recurrent Neural Networks”, IEEE TRANSACTIONS ON SIGNAL PROCESSING, VOL. 45, NO. 11, 2673.

Schrage, M. (2017). “4 Models for Using AI to Make Decisions”, *Harvard Business Review*, Technology.

Seal, H.L. (1967). “The historical development of the Gauss linear model”, Biometrika. 54 (1/2): 1–24. doi:10.1093/biomet/54.1-2.1. JSTOR 2333849.

Seber, W. C. J., G. A. F. (1987). Nonlinear Regression. New York: Wiley.
Semeniuta, S., A. Severyn, and E. Barth. (2016). “Recurrent dropout without memory loss,” arXiv preprint arXiv:1603.05118.

Shannon, C., Weaver, W. (1963). A Mathematical Theory of Communication. University of Illinois Press, Champaign.

Shazeer, N., A. Mirhoseini, K. Maziarz, A. Davis, Q. Le, G. Hinton, J. Dean, (2017). “Outrageously Large Neural Networks: The Sparsely-Gated Mixture-of-Experts Layer”. arXIV 23).

Shen F, Chao J, Zhao J. (2015). “Forecasting exchange rate using deep belief networks and conjugate gradient method”, Neurocomputing. 2015; 167(C):243–53.

Smith, C. (2016). “iOS 10: Siri now works in third-party apps, comes with extra AI features”, BGR. Retrieved 2017-06-27.

Solimun. (2014). “Estimating Confidence Interval of Mean Using Classical, Bayesian, and Bootstrap Approaches”, International Journal of Mathematical Analysis Vol. 8. (2014). no. 48, 2375 – 2383.

Sommer, S.C., and C.H. Loch. (2004). “Selectionism and Learning in Projects with Complexity and Unforeseeable Uncertainty,” Management Science, 50(10):1334-1347, 2004

Sovacool, B.K., and C.K. Cooper. (2013). “The Governance of Energy Megaprojects: Politics, Hubris and Energy Security”, Edward Elgar Pub, July 31, 2013

Srivasta, N., Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, Ruslan Salakhutdinov. (1958). “Dropout: A Simple Way to Prevent Neural Networks from Overfitting, Journal of Machine Learning”, Journal of Machine Learning Research 15 (2014) 1929-1958

Srivastava, S. (2013). “Improving neural networks with dropout,” University of Toronto, vol. 182.

Staab, S., and R. Studer, editors. (2004). Handbook on Ontologies. International Handbooks on Information Systems. Springer.

Strauß, S. (2015) “Datafication and the Seductive Power of Uncertainty —A Critical Exploration of Big Data Enthusiasm”, *Information*, 6, 836-847.

Suchanek, Fabian M. (2009). “Automated Construction and Growth of a Large Ontology”, Thesis for obtaining the title of Doctor of Engineering of the Faculties of Natural Sciences and Technology of Saarland University, Saarbr¨ucken, Germany, 2009-01-12

Sundermeyer, M., R. Schluter, and H. Ney. (2010). “LSTM Neural Networks for Language Modeling”

Summers, L. (2014). “U.S. Economic Prospects: Secular Stagnation, Hysteresis, and the Zero Lower Bound", *Business Economics* Vol. 49, No. 2, National Association for Business Economics.

Sundermeyer, Martin, (2014). "Translation modeling with bidirectional recurrent neural networks", Proceedings of the Conference on Empirical Methods on Natural Language Processing, October. 2014.

Sutskever, I., J. Martens, and G. E. Hinton. (2011). "Generating text with recurrent neural networks," in Proceedings of the 28th International Conference on Machine Learning (ICML-11). (2011). pp. 1017–1024

Sutskever, I.. (2013). "Training recurrent neural networks," University of Toronto, Toronto, Ont., Canada, 2013.

Sutskever, I., J. Martens, G. Dahl, and G. Hinton. (2013). "On the importance of initialization and momentum in deep learning," in International conference on machine learning. (2013). pp. 1139–1147.

Syrquin, M. (1978). "The Application of Multidimensional Scaling to the Study of Economic Development", The Quarterly Journal of Economics, Vol. 92, No. 4 (Nov., 1978), pp. 621-639

Taylor, J.W.. (2000). "A quantile regression neural network approach to estimating the conditional density of multiperiod returns." Journal of Forecasting 19, no. 4 (2000): 299-311.

The Department of Treasury. (2012). "The Financial Crisis Response in Charts".

Tibshirani, R. (1996). "A comparison of some error estimates for neural network models," Neural Computation, vol. 8, pp. 152–163.

Tokic, M. and G. Palm. (2011). "Value-difference based exploration: adaptive control between epsilon-greedy and softmax," KI 2011 Adv. Artif. Intell., pp. 335–346.

Tsinaslanidis, Prodromos E., Antonis Alexandridis, Achilleas Zapranis , Efstratios Livanis. (2013). "Dynamic Time Warping as a Similarity Measure: Applications in Finance", Kent Academic Repository

Tax, N., I. Verenich, M. La Rosa, M. Dumas. (2017). "Predictive Business Process Monitoring with LSTM neural networks". Proceedings of the International Conference on Advanced Information Systems Engineering (CAiSE): 477–492. doi:10.1007/978-3-319-59536-8_30.

United Nations Development Program. (2014). "Foresight: The Manual".

United Nations Habitat. (2013). “Planning and Design for Sustainable Urban Mobility”.

van Hinsbergen, C., J. van Lint, and H. van Zuylen. (2009). “Bayesian committee of neural networks to predict travel times with confidence intervals,” Transportation Research Part C: Emerging Technologies, vol. 17, no. 5, pp. 498–509, Oct. 2009.

Veaux, R. D. d., J. Schumi, J. Schweinsberg, and L. H. Ungar. (1998). “Prediction intervals for neural networks via nonlinear regression,” Technometrics, vol. 40, no. 4, pp. 273–282, 1998.

Veeriah, V., N. Zhuang, and G.-J. Qi. (2015). “Differential recurrent neural networks for action recognition,” in Proceedings of the IEEE International Conference on Computer Vision. (2015). pp. 4041–4049

van der Maaten, L.J.P., and G.E. Hinton. (2008). “Visualizing High-Dimensional Data Using t-SNE”, Journal of Machine Learning Research 9(Nov):2579-2605, 2008.

van der Maaten, L.J.P., and G.E. Hinton. (2012). “Visualizing Non-Metric Similarities in Multiple Maps”, Machine Learning 87(1):33-55, 2012.

van der Maaten, L.J.P. (2014). “Accelerating t-SNE using Tree-Based Algorithms”, Journal of Machine Learning Research 15(Oct):3221-3245, 2014.

Vogels, W. (2016). “Bringing the Magic of Amazon AI and Alexa to Apps on AWS. - All Things Distributed”. www.allthingsdistributed.com. Retrieved 2017-11-27.

Wang, Zhiguang, and Weizhong Yan, and Tim Oates. (2017). “Time Series Classification from Scratch with Deep Neural Networks: A Strong Baseline”, GE Global Research

Waterhouse, S.R. (1997). “Classification and Regression using Mixtures of Experts”, Jesus College, Cambridge, and Department of Engineering, University of Cambridge

Wen, R., K. Torkkola, B. Narayanaswamy. (2017). “A Multi-Horizon Quantile Recurrent Forecaster”, 31st Conference on Neural Information Processing Systems (NIPS 2017), Long Beach, CA, USA, arXiv:1711.11053v1 [stat.ML] 29 Nov 2017

Werbos, P. J. (1990). “Backpropagation through time: what it does and how to do it,” Proceedings of the IEEE, vol. 78, no. 10, pp. 1550–1560, 1990.

Wiering, M, and van Otterlo, Martijn (Eds.). (2012). “Reinforcement Learning: State-of-the-Art”, Springer.

Wilkinson, A., and R. Kupers. (2013). “Living in the Futures”, Harvard Business Review, Magazine article

World Bank, “Aggregating Risk Assessment Overview”,

Williams, R.J., and D. Zipser. (1995). “Gradient-based learning algorithms for recurrent networks and their computational complexity,” Backpropagation: Theory, architectures, and applications, vol. 1, pp. 433–486.

World Economic Forum. (2017). Global Risk Report 2017

World Bank. (2017). “Bangladesh Development Update: Sustained Development Progress”.

Wu, Y., M. Schuster, Z. Chen, Q.V. Le, M. Norouzi, W. Macherey, M. Krikun, Y. Cao, and Q. Gao. (2016). “Google's Neural Machine Translation System: Bridging the Gap between Human and Machine Translation”. arXiv:1609.08144.

Yan, X.. (2009). “Linear Regression Analysis: Theory and Computing”, World Scientific, pp. 1–2, ISBN 9789812834119,

Yao, K., T. Cohn, K. Vylomova, K. Duh, and C. Dyer. (2015). “Depth-Gated LSTM”, arXiv.org > cs > arXiv:1508.03790

Yao, K., T. Cohn, K. Vylomova, K. Duh, and C. Dyer. (2015). “Depth-gated lstm,” arXiv preprint arXiv:1508.03790, 2015.

Yin H, Jiao X, Chai Y, Fang B. (2015). Scene classification based on single-layer SAE and SVM. Expert Systems with Applications; 42(7):3368–80.

Yue J, Mao S, Li M. (2016). A deep learning framework for hyperspectral image classification using spatial pyramid pooling. Remote Sensing Letters; 7(9):875–84.

Zhang, X., J. Zhao, and Y. LeCun. (2015). “Character-level Convolutional Networks for Text Classification”, arXiv.org > cs > arXiv:1509.01626

Zheng, Yi, Qi Liu1, Enhong Chen, Yong Ge, and J. Leon Zhao. (2016). “Time Series Classification Using Multi-Channels Deep Convolutional Neural Networks”,

Zhu, L. and N. Laptev. (2017). “Deep and Confident Prediction for Time Series at Uber”, arXiv:1709.01907v1 [stat.ML] 6 Sep 2017

Zhu, X., P. Sobihani, and H. Guo. (2015). “Long short-term memory over recursive structures,” in International Conference on Machine Learning. (2015). pp. 1604–1612.