University of London
Imperial College of Science, Technology and Medicine
Department of Computing

Machine Learning Applications In Finance: Some Case Studies

Wei Zhen

Submitted in part fulfilment of the requirements for the degree of
Doctor of Philosophy in Computing of the University of London and
the Diploma of Imperial College, January 2019

This is to certify that to the best of my knowledge, the content of this thesis is my own work, or otherwise specified in acknowledgements or in footnotes. Neither this thesis nor the original work contained within has been submitted to another institution for a degree.

The copyright of this thesis rests with the author and is made available under a Creative Commons Attribution Non-Commercial No Derivatives licence. Researchers are free to copy, distribute or transmit the thesis on the condition that they attribute it, that they do not use it for commercial purposes and that they do not alter, transform or build upon it. For any reuse or redistribution, researchers must make clear to others the licence terms of this work.

Abstract

The first part of this thesis discusses the application of artificial intelligence to stock price prediction and the building of another artificial intelligence model for stock price prediction during mergers and acquisitions. It is demonstrated that a combination of Long Short-Term Memory/Recurrent Neural Network + Convolutional Neural Network + Attention (LSTM/RNN+CNN+Attention) trained on daily- and minute-stock price and volume data are able to predict future stock prices and stock price changes three times better than traditional non-deep learning methods. Further research has been done on stock price prediction during Mergers and Acquisitions for technology unicorns, finding that when more information on M&A has been added into the input dataset, the accuracy can be improved by about 2%. It is the first time that a deep learning algorithm has been applied to modelling volume and price/price change, and to M&A circumstances. In the second part of this thesis, a Support Vector Machine (SVM) method is proposed, which enables adverting agencies to assess the effectiveness of their advertisement rapidly, quantitatively and without bias. This is crucial work for business/equity trading because when the product/service of a company sells well, then the corresponding company stock prices rise. The success of an advertising agency in their ability to produce compelling advertisement is therefore related to stock price prediction. This research explores models that outperform existing research; even the basic machine learning model improves the accuracy of a prediction to purchase from 70% up to 80%. The last part of this thesis incorporates a deep learning method to encrypt the data going between input nodes and the rest of the network. This method has the advantage of simultaneously preventing sensitive data being intercepted during the transmission from the network edge over insecure transports, and also anonymize incoming data in a way that could help satisfy privacy laws such as General Data Protection Regulation (GDPR). This research has provided a new data privacy and security structure that is more cost efficient than existed deep learning model done solely locally or remotely.

Acknowledgements

I would like to express my gratefulness to my supervisor Guo Yike who accepted my application in the first place and offered me the opportunity to study artificial intelligence. He has given me unwavering support whenever I needed it. I would also like to thank Chao Wu who frequently gave me advice on my research; JingQing Zhang for giving helpful advice on deep learning techniques. I would like to thank my parents who are my strongest support during my life, who give me the courage to become better everyday, to take on and overcome challenges like this PhD; I would also like to thank my husband Yuan Gao who gives me his love and support in our relationship.

Dedication

To my dearest Baba and Mama.

‘May all your minima be global.’
-Ian Goodfellow

Contents

Abstract **i**

Acknowledgements **iii**

1 Introduction **1**

1.1 Motivation And Objectives 1

1.2 Contributions 5

1.3 Statement Of Originality 7

1.4 PhD By Publications 7

2 Background Theory **9**

2.1 ARMA Family 9

2.1.1 ARMA 10

2.1.2 SARMA 13

2.1.3 VAR 14

2.2 Machine Learning Models 15

2.2.1 Linear Regression 16

2.2.2 Least Absolute Shrinkage and Selection Operator (LASSO) Regression 17

2.2.3 Random Forest Regression 17

2.2.4 Support Vector Regression 18

2.2.5 Limitations of Baselines 19

2.3 Deep Learning Models 21

2.3.1 Multilayer Perceptron (MLP) 22

2.3.2 Convolutional Neural Networks (CNN) 23

2.3.3 Recurrent Neural Networks (RNN) 27

2.3.4 Attention+RNN 30

2.3.5 CNN+RNN 32

2.3.6 Attention+CNN+RNN 32

2.4 Data Privacy and Security 33

3 Stock Price Prediction **36**

3.1 Data Collection 36

3.1.1 Machine Learning 36

3.1.2 Deep Learning 37

3.2 Data Analysis and Refinery 40

3.2.1 Error Calculation 42

3.2.2 Special Days 42

3.2.3 Data Refinery 51

3.3 Methodology and Implementation 54
3.3.1 Implementation 54
3.4 Experiment and Evaluation 56
3.4.1 Machine Learning 56
3.4.2 Deep Learning 57
3.5 Discussion 64
3.5.1 Dataset 64
3.5.2 Novel Methods 65
3.6 Conclusion 66
3.6.1 Deep Learning 67

4 Stock Return Prediction Before Merger and Acquisition 68

4.1 Introduction 68
4.2 Data Collection 69
4.3 Data Analysis 70
4.3.1 Machine Learning 72
4.3.2 Deep Learning 76
4.3.3 Discussion 81
4.3.4 Conclusion 85

5 Electroencephalography(EEG) for Advertisement Impact Assessment 86

5.1 Introduction 87

5.2 Methodology 90
5.2.1 Data Collection 90
5.2.2 Questionnaire Collection 92
5.2.3 Modeling 93
5.2.4 Support Vector Machine 101
5.3 Result 102
5.3.1 Behavior Result 102
5.3.2 Cross-Validation And Learning Curve 104
5.4 Discussion 112
5.5 Conclusion 115

6 Data Privacy And Security 116

6.1 Introduction 117
6.2 Method: Dropping Activation Outputs with Localized First-layer Deep Network 118
6.2.1 Non-invertible Activation Functions 118
6.2.2 Invertible Activation Functions 121
6.3 Mathematical Explanation 124
6.3.1 Dropping Connections 125
6.3.2 Dropping Activation Outputs 128
6.4 Experimental Studies 130

6.4.1 Non-invertible Activation Functions . . . 131
6.4.2 Invertible Activation Functions - Dropping Activation Outputs . . . 134
6.4.3 Invertible Activation Functions - Dropping Connections . . . 137
6.4.4 Autoencoder . . . 138
6.4.5 Privacy Preserving . . . 139
6.4.6 Speed . . . 140
6.5 Conclusion . . . 141

7 Conclusion **143**
7.1 Summary of Thesis Achievements . . . 143
7.2 Applications . . . 144
7.3 Future Work . . . 145

Appendix A Figures **147**

Appendix B Crunchbase Data **191**

Appendix C Acquisition valuation **196**

Bibliography **196**

List of Tables

3.1 Training and testing data and the corresponding accuracy 38

3.2 The fields of information provided in the model dataset 39

3.3 Daily Data Partition 40

3.4 Minute Data Partition 40

3.5 A comparison between non-Christmas and Christmas periods on trading volume, close price, and the highest and lowest price change 45

3.6 Comparison within weekdays on trading volume, close price and the highest and lowest price change for Apple. 47

3.7 Comparison within weekdays on trading volume, close price and the highest and lowest price change for IBM. 48

3.8 Comparison within weekdays on trading volume, close price and the highest and lowest price change for Microsoft. 49

3.9 A comparison of Apple during the day Britain voted to leave the European Union, and the day Donald Trump was elected, compared with other normal days of the year on trading volume close price and the highest and lowest price change 51

3.10 A comparison of IBM during the day Britain voted to leave the European Union, and the day Donald Trump was elected, compared with other normal days of the year on trading volume close price and the highest and lowest price change 52

3.11 A comparison of Microsoft during the day Britain voted to leave the European Union, and the day Donald Trump was elected, compared with other normal days of the year on trading volume close price and the highest and lowest change 53

3.12 Machine learning Apple 2018 January to May data MAPE prediction when the training dataset is over different intervals 56

3.13 CNN-LSTM-Attn model settings use the minute dataset prediction 58

3.14 CNN-LSTM-Attn model settings use the minute dataset prediction 60

3.15 MAPE (%) on the 2013-2018 dataset. CNN-LSTM-Attn achieves best overall performance with lowest error rate 9.21%. 61

3.16 Comparison among different DNNs on the dataset. 62

3.17 Comparison between different length of input on the dataset 63

4.1 The MAPE error during Apple acquisitions 74

4.2 The MAPE error during Amazon acquisitions 74

4.3 The MAPE error during Google acquisitions 75

4.4 The MAPE error during Facebook acquisitions 75

4.5 The MAPE error of all acquisitions that has a valuation on the Crunchbase 76

4.6 CNN-LSTM-Attn acquisition model using the minute dataset prediction . 77

4.7 CNN-LSTM-Attn acquisition model monthly MAPE comparison 78

4.8 CNN-LSTM-Attn acquisition model settings use the minute dataset prediction 79

4.9 Comparison among different DNNs on the dataset. 80

4.10 Comparison between different length of input on the dataset 81

4.11 Comparison among the CNN+RNN+Attn model on each company. 84

5.1 Price/cost of different EEG devices 91

5.2 The output dataset list 94

5.3 The EEG band dataset list 97

5.4 Accuracy prediction of the ranked answer to different type of product at different thresholds 104

5.5 Likelihood of purchasing the car in the combined answer model at different thresholds 105

5.6 Likelihood of purchasing the food in the combined answer model at different thresholds 106

5.7 Likelihood of purchasing the digital in the combined answer model at different thresholds 107

5.8 Likelihood of purchasing the clothes in the combined answer model at different thresholds 108

5.9 Likelihood of purchasing in the combined answer model at different thresholds 109

6.1 Randomly setting activation outputs to zero in first hidden layer during feed-forward propagation over 100 trails 136

6.2 Randomly setting activation outputs to zero in first convolutional layer during feed-forward propagation over 100 trails 136

C.1 The 87 companies that are acquired with a valuation listed on Crunchbase 197

C.2 The 87 companies that are acquired with a valuation on listed Crunchbase 198

C.3 The 87 companies that are acquired with a valuation on listed Crunchbase 199

C.4 The 87 companies that are acquired with a valuation on listed Crunchbase 200

List of Figures

2.1 An illustration of dependence and independence in $ARMA(2,2)$. Given $\{x_{t-2}, x_{t-1}, \epsilon_{t-1}, \epsilon_{t-1}, \epsilon_t\}$, x_t is independent with other previous values. . . . 13

2.2 An illustration of dependence and independence in $SARMA(2,2) \times (1,1)_S$. x_t not only depends on previous neighboring values and also takes seasonality into account. 14

2.3 The variance of Support Vector Regression may become unstable to the right. 20

2.4 An example of a three layer MLP, from machinelearningmastery.com/when-to-use-mlp-cnn-and-rnn-neural-networks/ 23

2.5 Sources from http://deeplearning.net/tutorial/lenet.html 24

2.6 Sources from http://deeplearning.net/tutorial/lenet.html 24

2.7 Computing the output values of a discrete convolution, sources from [DV16] 26

2.8 Computing the output values of a discrete convolution for $N = 2, i_1 = i_2 = 5, k_1 = k_2 = 3, s_1 = s_2 = 2$, and $p_1 = p_2 = 1$, sources from [DV16] 27

2.9 Computing the output values of a 3×3 max pooling operation on a 5×5 input using 1×1 strides, sources from [DV16] 28

2.10 LSTM figure, sources from [GBCB16] 29

2.11 The repeating module in an LSTM contains four interacting layers, source from http://colah.github.io/posts/2015-08-Understanding-LSTMs/ 30

2.12 Global attentional model at each time step t, the model infers a variable-length alignment weight vector at based on the current target state ht and all source states $\bar{\mathbf{h}}_\mathbf{s}$. A global context vector $\mathbf{c_t}$ is then computed as the weighted average, according to $\mathbf{a_t}$, over all the source states, source from [LPM15] 31

2.13 CNN-LSTM-Attn network structure. The CNN-LSTM-Attn consists of four sub-networks, CNN, RNN with LSTM, global attention mechanism and fully connected decoder. 33

2.14 Feed-forward propagation in server; x is the input data, y is the output value, $P(y|x)$ is the probability of y given x. 35

3.1 The figure of the training and testing dataset 37

3.2 Figure of the data partition in deep learning training 38

3.3 Project structure. All deep neural network models are implemented on TensorLayer and the base model. Given the model and refined data, the controller defines the process for training, validation, testing and inference. Applications, including analysis and visualisation, are built upon prediction results. 54

4.1 The summation of the two curve 83

5.1 Data pre-processing workflow: the input dataset is first augmented, before both input and output datasets have features extracted and bootstrapped to increase the size of the dataset; the resulting larger dataset is split into training and test datasets, on which SVM classification is applied; the results of the classification are used to label the data and fed into the cross-validation and learning. 96

5.2 Cross validation curves when only ranked answers are selected, with a threshold of 4. 110

5.3 Cross-validation curves when thresholds of ranked answers is 4 and binary answers threshold is 0.4, their weight is equal at 0.5, and the whole dataset thresholds is 0.4 111

6.1 Spliting neuron network during feed-forward propagation. 119

6.2 Dropping activation output during feed-forward propagation. 123

6.3 Dropping connection during feed-forward propagation. 123

6.4 Reconstructed input data from rectifying output by using $\hat{x} \approx (a-b)*W^{-1}$ (Right); Original input data x (Left); Activation output $a = max(0, x * W + b)$. The KL Divergence between x and $\hat{x}$ for digit 2 and 6 are 1.91 and 1.88. 132

6.5 Reconstructed input data from rectifying outputs by $\hat{x} \approx (a - b) * W^T$: Original input data x (Left); Reconstruct from outputs of first hidden layer (Middle, the KL Divergence between x and $\hat{x}$ (1 st layer) are 2.32, 1.49, 1.51 for digit 1, 2, 6); Reconstruct from outputs of second hidden layer (Right, the KL Divergence between x and $\hat{x}$(2nd layer) are 2.63, 1.58, 1.56 which are higher than $\hat{x}$ (1st layer)). 133

6.6 Reconstructed input data from ramp activation outputs by using $\hat{x} \approx (a - b) * W^T$ (Middle and Right) and Original input data x with $v = 0.2$(Left); The KL Divergence between x and $\hat{x}$ ($v = 0.2$), which are 2.35, 1.54, 1.55 for digit 1, 2, 6, is smaller than that between x and $\hat{x}$ ($v = 0.05$), which are 2.42, 1.63, 1.62. 134

6.7 Equation 6.24 and 6.2 are used to reconstruct input data from first sigmoid hidden layer after dropping activation outputs. The first column is the original input data x; the dropping probabilities increase from 0.5% to 2% from second to forth column. Note that 0.5% is only 4 neurons in the layer of 800 neurons. The KL Divergence between x and $\hat{x}(p = 0.5\%)$ are 2.03, 1.34, 1.35 which are similar with $\hat{x}(p = 1\%)$ (2.02, 1.35, 1.35) and $\hat{x}(p = 2\%)$ (2.04, 1.35, 1.36). 137

6.8 Reconstructed input data from first sigmoid hidden layer after drop activation outputs by using Equation 6.24 and 2.25. Original input data x (The first column); From the second to third columns, the dropping probabilities increase from 0.25% to 0.5%. The KL Divergence between x and $\hat{x}(p = 0.25\%)$ are 2.34, 1.51, 1.45, and $\hat{x}(p = 0.5\%)$ are 2.49, 1.52, 1.46, which are higher than Fig.6.7 138

6.9 Reconstructed input data from sigmoid outputs by using Equation 6.5 (Right); Original input data x (Left). The number of neuron is the half of the number of input data. 139

A.1 Apple Daily and Minute data of price change between 05/2017-05/2018 . . 148

A.2 Apple Daily and Minute data of close price between 05/2017-05/2018 . . . 149

A.3 Apple Daily and Minute data of volume between 05/2017-05/2018 150

A.4 Apple monthly and weekly data of price change in October 2017 151

A.5 Apple monthly and weekly data of close price in October 2017 152

A.6 Apple monthly and weekly data of log volume in October 2017 153

A.7 Apple monthly and weekly data of volume in October 2017 154

A.8 Christmas period and non-Christmas period Apple data of price change in October 2017 155

A.9 Christmas period and non-Christmas period Apple data of close price in October 2017 156

A.10 Christmas period and non-Christmas period Apple data of volume in October 2017 157

A.11 Apple weekday data of price change, close price and volume in May 2017 . 158

A.12 Apple weekday data of price change, close price and volume in May 2017 . 159

A.13 Apple weekday data of price change, close price and volume in May 2017 . 160

A.14 Apple weekday data of price change, close price and volume in May 2017 . 161

A.15 Apple weekday data of price change, close price and volume in May 2017 . 162

A.16 Apple weekday data of price change, close price and volume each month in 2017 163

A.17 Apple weekday data of price change, close price and volume each month in 2017 164

A.18 Apple weekday data of price change, close price and volume each month in 2017 165

A.19 Apple weekday data of price change, close price and volume each month in 2017 166

A.20 Apple weekday data of price change, close price and volume each month in 2017 167

A.21 Apple weekday data of price change, close price and volume each February between year 2012 to year 2017 168

A.22 Apple weekday data of price change, close price and volume each February between year 2012 to year 2017 169

A.23 Apple weekday data of price change, close price and volume each February between year 2012 to year 2017 170

A.24 Apple weekday data of price change, close price and volume each February between year 2012 to year 2017 171

A.25 Apple weekday data of price change, close price and volume each February between year 2012 to year 2017 172

A.26 Apple,IBM and Microsoft data of price change in the day Britain voted to leave the European Union and Donald Trump being elected as the USA president 173

A.27 Apple, IBM and Microsoft data of close price in the day Britain voted to leave the European Union and Donald Trump being elected as the USA president 174

A.28 Apple, IBM and Microsoft data of volume in the day Britain voted to leave the European Union and Donald Trump being elected as the USA president 175

A.29 Amazon 1 months before and after announcement daily 176

A.30 Amazon 3 months before and after announcement daily 177

A.31 Facebook 1 months before and after announcement daily 178

A.32 Facebook 3 months before and after announcement daily 179

A.33 Google 1 months before and after announcement daily 180

A.34 Google 3 months before and after announcement daily 181

A.35 Apple 1 months before and after announcement daily 182

A.36 Apple 3 months before and after announcement daily 183

A.37 Apple 6 months before and after announcement daily 184

A.38 Amazon 6 months before and after announcement daily 185

A.39 Facebook 6 months before and after announcement daily 186

A.40 Google 6 months before and after announcement daily 187

A.41 Apple 1 months before and after announcement minute 188

A.42 Apple 3 months before and after announcement minute 189

A.43 Apple 6 months before and after announcement minute 190

B.1 The Apple acquisition list 192

B.2 The Amazon acquisition list 193

B.3 The Google acquisition list 194

B.4 The Facebook acquisition list 195

Chapter 1

Introduction

1.1 Motivation And Objectives

In 2017, Forbes forecast that the valuation of the AI industry would rise from 644 million now to 126 billion in 2025 [Pre18]. There is increasing attention to the use of AI in a broad range of technologies and applications, some of which are merely extensions of earlier techniques while others are wholly new industries. e.g Travel, transport and logistics, retail, automotive and assembly, oil and gas, chemicals, media and entertainment, basic materials, agriculture, packaged goods, banking, healthcare systems and services, public and social sectors, telecommunications, pharmaceuticals and medical products, insurance, advanced electronics/semiconductors, aerospace and defense, finance. Following this trend, this thesis explores AI's applications in the financial industry.

A report by eFinancial predicted that 230,000 finance-related jobs could disappear by 2025, of which 90,000 jobs will disappear from the asset management industry, 58,000 from the securities services, 45,000 from sales and trading, 24,000 jobs from wealth management, and 4,000 from investment banking [Cla17].

In the literature, deep neural networks have been applied to stock price prediction. Because trading strategies vary based on institutions and individuals' profit targets, strategies, and time scale for making a profit; deep neural networks have been modified to achieve different trading strategies. For example, in 2007, Kwon published a stock forecasting paper using a hybrid approach combined together with deep recurrent neural network method [KM07]. In the hybrid approach, the input dataset is technical analysis signals such as relative strength index (RSI), moving average (MA), moving average convergence and divergence (MACD), etc. This paper has demonstrated that technical signals work in a larger sample with a longer period of testing. The KDD conference paper "Stock Price Prediction via Discovering Multi-Frequency Trading Patterns" has been published by Zhang and Aggarwal in 2017, with the lowest error rate described in the paper at 6% [ZAQ17]. Inspired by Discrete Fourier Transform (DFT), Zhang proposed a novel State Frequency Memory (SFM) recurrent neural network to capture multi-frequency trading patterns from past market data. The SFM decomposes the hidden states of memory cells into multiple frequency components, each of which model a particular frequency of latent trading pattern underlying the fluctuation in stock price.

In business, any big sales reduction indicates a very big drop in stock price. For example, in the BBC news "Four reasons that Apple shares have been falling", it states that one of the reasons for stock price reduction is concerns about the China-US trade war. Investors fear that purchasing power will be significantly reduced because no one from China would be able to purchase Apple products [She18]. As research papers have shown, there is a positive correlation between advertising and the product sales figures, so, how to efficiently advertising the products/services have been a focus of advertising agencies [AFL84] [Cla76]. In the past, surveys and questionnaires have been used to find customer preferences and efficiency of advertising. This type of research method is biased, not efficient, nor general, and costly. In order to solve this problem, a better experimental design has been proposed using the machine learning method Support Vector Machine (SVM) to

quantify the advertisement's impact on purchase power, this new experimental design uses a low-cost device called a single electrode EEG headset to collect EEG (electroencephalogram, i.e. brain waves) signal from a subject as an input dataset. The experiment then uses ranked and binary answers from questionnaires as the output dataset. Questions in this dataset are carefully selected from current literature.

There is increasing attention to data privacy and security. Especially in March of 2018, when it was reported that Cambridge Analytica had used data scraped from Facebook to target ads intended to sway the presidential vote. In the Accenture 2018 Latest Thinking, it outlines increasing trends of awareness and research into cyber-security, along with growth in profile of information privacy across a wide range of applications, including for healthcare, consumer sector, financial sector, and internet security.

Deep learning, also known as deep neural networks [DY+14b], has proven to be successful for classification and regression tasks. Once well established, deep learning models exhibited promising performance, which is desired by many real-time prediction/decision-making applications [RDGF16], automatic speech recognition, image recognition, natural language processing.

Today, more and more deep learning models are being developed for trading, driven by the availability of new sources of data. For example, Twitter, which advertises itself as being able to 'get in-the-moment updates on the things that interest you', is a new source of data for public sentiment towards current affairs. A paper by Costa, Michele and Marshall has shown that the Twitter data is correlated with the stock movements [CMMS18]. Similarly, in 2017, Weng conducted research using Google search queries to make stock market predictions [Wen17] [PMS13]. While these two examples apply data that is publicly available to stock price prediction, there are potentially many more sources of data that are privately owned that contain useful information that could be used to build trading indices (i.e. commodity index). However, such private data may come from

privileged or commercially sensitive data that a source provider would likely not wish to expose to the public or third parties. In these cases, data privacy and security is needed to ensure that no data is exposed to third parties.

Other applications of deep learning over distributed sources of data face hardware limitations. It is computationally expensive to implement deep learning algorithms entirely on users' local devices (sensors, mobiles, and even laptops). As a result, the typical approach is to collect data locally first and then transmit the data to a remote server or cluster, where deep networks for training or prediction can be applied. However, as these applications sometimes deal with sensitive data, such patients' clinical data (like fMRI images) and sensitive environment data (like video monitoring in public space), there are similar concerns for data privacy. Both when sending these data over the internet, in the case of the data being intercepted; and concerns about a company using methods to de-anonymize the data. For example, it would cause severe consequences if a company that handles a lot of users' health-care data like DNA, EHR, and fMRI, was to be invaded by malicious hackers. It is therefore desirable to secure the collection of sensitive data being transmitted from the user, or edge of the network where the data is generated or collected, to the servers where deep learning models are run.

There has been some research on conducting neural network prediction on encrypted data rather than the original data in the situation so that users or sensors will not share them with remote servers [BPTG15] [GLN12]. However, their approach brings in extra computation cost, and it would not be safe once the key for encryption was hacked.

The motivation of this thesis is to explore an existing trading strategy, but leveraging deep learning methods to improve it, and use the strategy to explore price and volume movement during mergers and acquisitions. Secondly, as advertising has a big impact on customer purchase power, this thesis also explores general and scalable methods to quantify the impact of advertising, to help design advertisements in the way which can

better promote sales, and indirectly impact the share price of a company. The third reason for this research is to explore data privacy and security methods; in light of the increasing number of data leaks, especially in the financial industry, a method to help reduce the chance and impact of leaks that can be used in all trading houses and institutions would benefit their customers by reducing the chance of data leaks due to being hacked, or data lost in transmission.

1.2 Contributions

The trading strategies discussed in this thesis use trading volume to make stock price predictions. It is based on previous research. In 2000, Tarun Chordia published a paper on trading volume and cross-autocorrelations in stock returns, finding that volume is a significant determinant in stock return [CS00]. In 1987, Jonathan M also published a survey paper on the relationship between price change and trading volume [Kar87].

In this thesis, an existing trading strategy will be analyzed and explored; then an appropriate deep neural network is added to minimize the prediction error. Then, the prediction models for mergers and acquisitions in the technology industry is explored and studied, with a new model built to improve accuracy.

The next chapter of the thesis is focused on quantitatively assessing the impact of advertisement, so that it can serve businesses and advertising agency to improve the sales of a product. The contribution of this chapter is a method that is unbiased and general enough to be applied to any advertised product and business. The envisioned future is one where when an advertisement company can use analysis of recorded electroencephalogram (EEG) to help pinpoint which elements of an advertisement is the most effective. For example, in an ad for children, which colours, objects, animals or other elements are most likely to attract attention.

Targeting the issues in the private financial dataset, we proposed a novel deep network architecture, in which users don't need to reveal their original data to the model. Instead of the traditional approach shown in Figure 2.14, we split the layers between the local device and the server, and migrate the first layer to local device (as shown in Figure 6.1). We apply the activation functions for the first layer locally and then transfer the outputs from the first layer to the server. We then use a "dropping activation output" method to make the output non-invertible, which randomly drops some outputs from the activation function. In this method, feed-forward propagation and data encryption are combined into one process.

We investigated both invertible and non-invertible activations and their performance in privacy protection to prove that this method can make sure the original data cannot be recovered from the transmitted data. Concretely, the main contributions of this thesis are listed as follow:

- We introduce a new deep network architecture that combines neural network and data encryption to solve privacy and data security problem. With privacy-preserving feed-forward propagation, a server (model provider) can serve the user without directly accessing the user's data.

- We proved and evaluate that dropping some activation outputs can encrypt data for invertible activation function. We also evaluate that a ramp function is better than a rectifier in term of encryption.

- We propose privacy-preserving error-back propagation, by which the server can train a neural network for the user without accessing the user's data.

- With this method, data compression for data transmission is available when the number of neurons is less than the size of input data.

1.3 Statement Of Originality

I declare that the content of thesis is composed by myself, and the work it presents is my own. All use of the previously published work of others has been listed in the bibliography. And the chapter four of the thesis has not been published.

1.4 PhD By Publications

- A Data-driven Methodology for Social Issues with a case on left-behind children in China, Chao Wu, GuoLong Wang, Yue Liu, Ye Zhou, David Birch, **Zhen Wei**, Hong Mi, Yike Guo, PLOS ONE (PONE-D-18-20581) (2018 under review)

  This paper uses machine learning methods to analyze the left-behind children in China. I built machine learning models. This paper is not included in the PhD thesis.

- Using Support Vector Machine on EEG for Advertisement Impact Assessment, **Zhen Wei**, Chao Wu, Xiaoyi Wang, Akara Supratak, Hao Dong, Yike Guo, Frontiers in Neuroscience 12(2018):76

  This paper is original research about the use of machine learning method to assess advertisement impact. This paper is in chapter five of the PhD thesis.

- Improved accuracy stock price change prediction model using trading volume, **Zhen Wei**, Chao Wu, YiKe Guo, Future of Information and Communications Conference (FICC) in Singapore (2018)

  This paper presents a better stock price prediction than traditional machine learning methods in the literature. This paper is about Financial stock price prediction, and is one subsection of the chapter three of the paper.

- Conditional Image Synthesis Using Stacked Auxiliary Classifier Generative Adversarial Networks, Zhongwei YAO, Hao DONG, Fangde Liu, **Zhen Wei**, Yike Guo, Future of Information and Communications Conference (FICC) in Singapore (2018)

  This paper is about computing vision, and I am the fourth author of this paper. This paper is not included in the PhD thesis.

- Dropping Activation Outputs with Localized First-layer Deep Network for Enhancing User Privacy and Data Security, Hao Dong, Chao Wu, **Zhen Wei**, Yike Guo. IEEE Transactions on Information Forensics & Security, 10.1109/TIFS.2017.2763126 (2017)

  This paper is about the data privacy and security, and though I am the third author of the article, I was responsible for the mathematics proof behind the idea. This paper is on the sixth chapter of the PhD thesis.

Chapter 2

Background Theory

Stock price prediction can be defined as a time series problem, where given a sequence $x_1, x_2, ..., x_t$ of stock observations (e.g. price and volume) at a specific time, a prediction of $x_{t+\delta}$ with forecasting horizon Δ needs to be made. This chapter provides an overview of the literature and a detailed introduction of methods used for stock prediction. In this chapter, classical time series analysis methods and machine learning models will first be introduced and implemented as baselines, then, fundamental deep neural network architectures that this thesis's proposed models are based on are introduced. Concepts introduced include Multilayer Perceptron, Convolutional Neural Networks (CNN), Recurrent Neural Networks (RNN), Long Short Term Memory (LSTM), Attention+RNN, CNN+RNN, and Attentional+CNN+RNN. Traditional data privacy and security methods are also introduced.

2.1 ARMA Family

The general Autoregressive Moving Average (ARMA) model was first introduced by Peter Whittle in his thesis in 1951 [Whi53][Whi63]. It was made popular by George E.P.Box

and Gwilym Jenkins in 1970 for time series analysis [BJRL15]. ARMA is a tool for understanding and predicting future values in a time series, and as its name suggests, consists of two parts: an autoregression (AR), and a moving average (MA).

In the literature, Chordia and Swaminathan previously conducted research on stock return versus trading volume prediction using Vector Autoregression (VAR) [CS00]; in 1998, Easley and Ohara used stock prices and option volume to find evidence of informed trading using the Auto-Regressive Integrated Moving Average (ARIMA) model [EOS98] [PL05]. In research by Jegadeesh, Thaler, Keim, Rozeff, Kinney Jr, Gultekin, and Gultekin, each have shown from different perspectives that the stock return has a seasonal effect [Jeg91] [DBT87] [Kei83] [RKJ76] [GG83]. The difference between ARIMA and ARMA is that ARIMA introduces another hyper-parameter to handle non-stationarity. Mathematically it is: $ARIMA(p, d, q) = ARMA(p + d, q)$. Stationarity here means that the mean and variance are not constant, indicating that there might be seasonality.

2.1.1 ARMA

Autoregressive moving average, or $ARMA(p, q)$, consists of two parts: an autoregressive $AR(p)$ model and a moving-average $MV(q)$ model. Typically, ARMA is a linear model and can be used to understand and predict values in time series.

Autoregressive Model

If Gaussian random noise is assumed, the conditional probability of x_t can be written as

$$p(x_t|x_1, ..., x_{t-1}) \sim \mathcal{N}(f(x_1, ..., x_{t-1}), \sigma^2) \tag{2.1}$$

$$x_t = f(x_1, ..., x_{t-1}) + \epsilon_t \tag{2.2}$$

where ϵ_t is independent white noise and f is a regression function. In an autoregressive

model, given an integer p, the $AR(p)$ model is a linear combination of previous values with a random noise term, and is defined as:

$$x_t = c + \sum_{i=1}^{p} \psi_i x_{t-i} + \epsilon_t \tag{2.3}$$

where $c, \psi_1, ..., \psi_p$ are parameters and $\epsilon_t \sim \mathcal{N}(0, \sigma^2)$. According to Equation 2.2, future estimation x_t linearly depends on past observation $\{x_{t-p}, ..., x_{t-1}\}$. With a lag operator, the model can be equivalently written as:

$$x_t = c + \sum_{i=1}^{p} \psi_i B^i x_t + \epsilon_t \tag{2.4}$$

As classic AR is a linear model, kernel functions can be applied to incorporate nonlinear dependence relations. This is because usually a "kernel" is used to refer to the kernel trick, a method of using a linear classifier to solve a non-linear problem. For example, the regression function can be formatted as:

$$x_t = c + \sum_{j=1}^{h} \psi_j \phi_j(x_{t-1}, ..., x_{t-p}) + \epsilon_t \tag{2.5}$$

Moving Average Model

The moving-average model captures dependence relations between variables and unobserved noise. The conditional probability is defined as $p(x_t|\epsilon_{t-q}, ..., \epsilon_{t-1})$. As ϵ_i are unobservable, the interpretability of the moving-average model is less convincing than the autoregressive model. Given a integer q, the $MV(q)$ is defined as:

$$x_t = \mu + \epsilon_t + \sum_{i=1}^{q} \theta_i \epsilon_{t-i} \tag{2.6}$$

where μ is an expectation of x_t, $\theta_1, \theta_2, ..., \theta_q$ are parameters and $\epsilon_{t_1}, ..., \epsilon_{t-q}$ are white

noise term. The $MV(q)$ model is typically a linear regression of current expectation with current and previous white noise. It can be equivalently written with lag operator:

$$x_t = \mu + \epsilon_t(1 + \sum_{i=1}^{q} \theta_i B^i) \tag{2.7}$$

Note that B^i is equal to $\sum_{i=1}^{q} \frac{\epsilon_{t-i}}{\epsilon_t}$

ARMA

The $ARMA(p, q)$ as illustrated in Figure 2.1, is a summation of $AR(p)$ and $MV(q)$, modelling conditional probability $p(x_t|x_{t-1}, ..., x_{t-p}, \epsilon_t, ..., \epsilon_{t-q})$, it is $\epsilon_t, ..., \epsilon_{t-q}$ as it is shown written in Equation 2.8, defined as:

$$x_t = c + \epsilon_t + \sum_{i=1}^{p} \psi_i x_{t-i} + \sum_{i=1}^{q} \theta_i \epsilon_{t-i} \tag{2.8}$$

$$= c + \sum_{i=1}^{p} \psi_i B^i x_t + \epsilon_t(1 + \sum_{i=1}^{q} \theta_i B^i)$$

Alternatively the ARMA can be written with lag operator:

$$\psi(B)x_t = c + \theta(B)\epsilon_t \tag{2.9}$$

where

$$\psi(B) = 1 - \sum_{i=1}^{p} \psi_i B^i$$

$$\theta(B) = 1 + \sum_{i=1}^{q} \theta_i B^i$$

ARIMA

The autoregressive integrated moving-average model $ARIMA(p, d, q)$, as a generalization

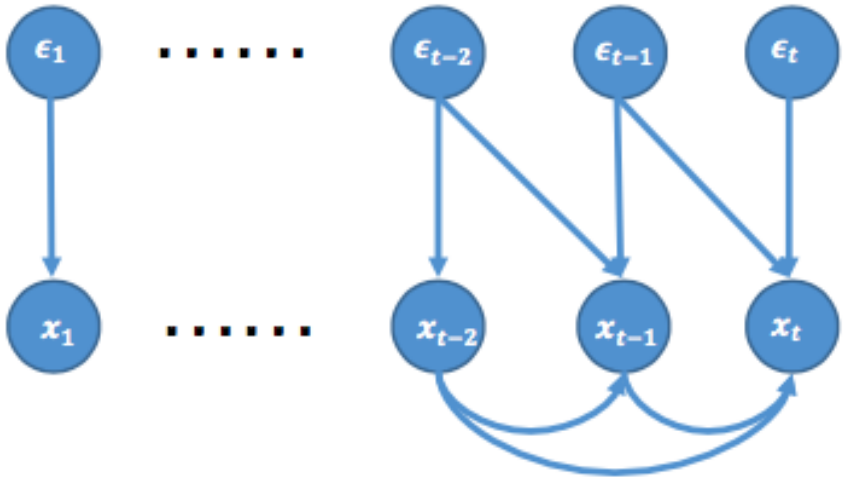


Figure 2.1: An illustration of dependence and independence in $ARMA(2,2)$. Given $\{x_{t-2}, x_{t-1}, \epsilon_{t-1}, \epsilon_{t-1}, \epsilon_t\}$, x_t is independent with other previous values.

of ARMA, introduces another hyper-parameter d to handle non-stationarity.

$$\psi(B)(1-B)^d x_t = c + \theta(B)\epsilon_t \tag{2.10}$$

2.1.2 SARMA

Seasonal ARMA shows periodic features. As trading can potentially be significantly affected by holidays and calendar seasons. In ARMA, x_t is independent with $\{x_1, ..., x_{t-p-1}\}$, if given $\{x_{t-p}, ..., x_{t-1}\}$. SARMA, as shown in Figure 2.2, introduces a group of hyper-parameters $(P,Q)_S$, which defines dependence between x_t with $\{x_{t-S}, ..., x_{t-PS}, \epsilon_{t-S}, ..., \epsilon_{t-QS}\}$. The SARMA denoted by $SARMA(p,q) \times (P,Q)_S$, can be defined as:

$$\Psi(B^S)\psi(B)x_t = c + \Theta(B^S)\theta(B)\epsilon_t \tag{2.11}$$

where

$$\Psi(B^S) = 1 - \sum_{i=1}^{P} \psi_i B^{iS}$$

$$\Theta(B^S) = 1 + \sum_{i=1}^{Q} \Theta_i B^{iS}$$

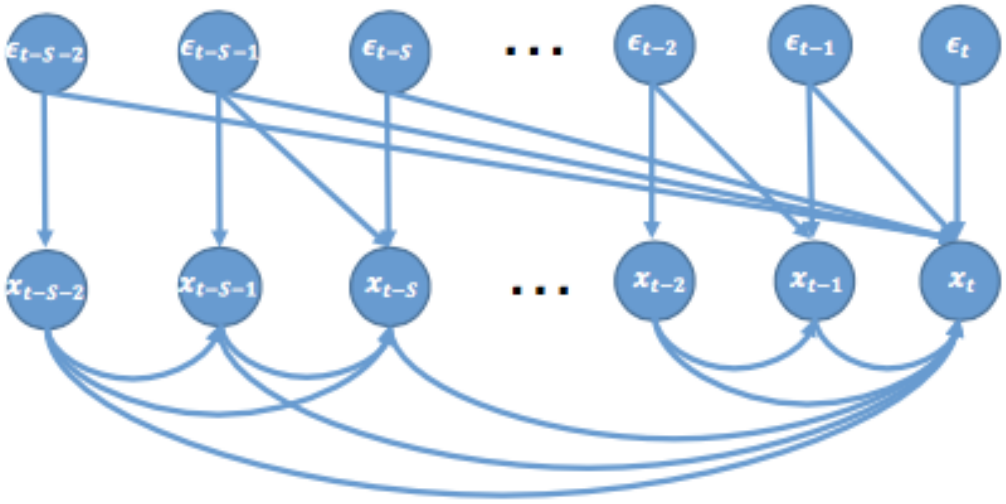


Figure 2.2: An illustration of dependence and independence in $SARMA(2,2) \times (1,1)_S$. x_t not only depends on previous neighboring values and also takes seasonality into account.

SARIMA Similarly, SARIMA can be denoted as $SARIMA(p,d,q) \times (P,D,Q)_S$, and defined as:

$$\Psi(B^S)\psi(B)(1-B)^d(1-B^S)^D x_t = c + \Theta(B^S)\theta(B)\epsilon_t \tag{2.12}$$

2.1.3 VAR

Vector autoregressive models (VAR) are a special case of vector ARMA and is capable of capturing linear interdependencies among multiple time series (see also [BJRL15][Lüt05]). Given $K \times T$ multivariate time series $\{X_1, X_2, ..., X_T\}$, VAR can be defined as:

$$X_t = A_1 X_{t-1} + A_2 X_{t-2} + ... + A_p X_{t-p} + \varepsilon_t \tag{2.13}$$

where A_i is $K \times K$ coefficient matrix and $\varepsilon_t \sim \mathcal{N}(0, \Sigma_K)$.

In this thesis, VAR, ARIMA and SARMA models are selected to test the dataset. Previously, Jung and Body have applied VAR to indices and government rates in 1996 and achieved a prediction accuracy of about 82% [MSA00]. VAR is chosen because there are multiple variables that are considered while modelling. The ARIMA model has been used on the New York Stock Exchange (NYSE) and Nigeria Stock Exchange (NSE) datasets and achieved an accuracy of about 76% in research by Ayo, Adebiyi and Adewumi in 2014 [AAA14]. ARIMA was selected because the stock price, in general, goes up due to inflation. Therefore, the mean and variance of the stock price are not constant; SARMA was picked because there are seasonality effects on the stock price and it would be wise to consider this to be able to make a more accurate prediction. In the article "stock market seasonality: international evidence", Gultekins shows that the seasonality mostly appears to be caused by the disproportionately large January returns in most countries and the April returns in the UK, which coincide with the tax year in those respective countries [GG83]. However, in a later model, this ceases to work due to both costly calculations and also due to having not enough data as the seasonality is considered on an annual base.

2.2 Machine Learning Models

In 2015, Cakra and Trisedya published the paper "Stock Price Prediction using Linear Regression Based on Sentiment Analysis", in the experiment, it was shown that prediction models using previous stock price and hybrid features as predictor gave the best prediction with 0.9989 and 0.9983 coefficient of determination [CT15]. Bollen, in his research, concluded that individuals' opinions on social media such as Twitter could predict DJIA value with 87.6% accuracy [BMZ11]. In 2007, research conducted by Talyor and Francis applied Support Vector Machines (SVM) to technical analysis to drive stock price pre-

diction, achieving an accuracy of about 70% [IT07]. In addition, SVM was also applied to stock price index movement on the Istanbul Stock Exchange National 100 Index and achieved an accuracy about 71.52% accuracy [KBB11]. Lasso Regression has even wider applications in the stock market; Roy et al. conducted experiments on the Goldman Sachs Group Inc. stock value to determine the efficiency of the LASSO model and achieved an accuracy of 69.89% [RMBA15]. Li and Chen also published a paper on "Forecasting Macroeconomic Time Series: LASSO-based Approaches and their Forecast Combinations with Dynamic Factor Models", which evaluates the predictive abilities of this category of LASSO-based approaches for forecasting twenty important macroeconomic variables. Moreover, this type of LASSO combined-forecast approach is significantly better than either dynamic factor model forecasts or the Naive random walk benchmark [LC14].

2.2.1 Linear Regression

Linear regression is a model that assumes the relationship between a dependent variable and an independent variable is linear [NKNW96] [SL12] [MPV15]. While there can be multiple independent variables, in this model, there is only one: volume. Mathematically, it can be written as

$$P_h - P_l = f_l(V) \tag{2.14}$$

where P_h is the highest price at that time, and P_l is the lowest price at that time, V represents the volume and f_l is the trained linear function.

In this model, the linear regression function from Python Sklearn is used, with default options.

2.2.2 Least Absolute Shrinkage and Selection Operator (LASSO) Regression

LASSO regression uses Least Squares to find the best fit model [Tib96] [Han09]. Least Squares regression is written mathematically as $min\frac{1}{N}\sum_{i=1}^{N}(P_{h,i} - P_{l,i} - f_S(V_i))^2$, where i corresponds to the time step. LASSO regression differs from linear regression by the fact that residual/norm β is modulated and the coefficient in front of the norm λ is 0.1 in this case. Mathematically, it can be written as

$min\{\frac{1}{N}\sum_{i=1}^{N}(P_{h,i} - P_{l,i} - f_S(V_i))^2 - \lambda||\beta||\}$.

2.2.3 Random Forest Regression

In general, classification is an example of pattern recognition, and regression is for estimating the relationships among variables; Random Forest Regression and Random Forest classification are both examples of these.

Random Forest regression is a type of regression that uses an ensemble of decision trees that a subset of data is classified with. The regression residual sum of squares (RSS) is calculated as $RSS = \sum_{left}(Y_{train} - Y_L)^2 + \sum_{right}(Y_{train} - Y_R)^2$ where Y_L is the mean Y-value for the left node and Y_R is the mean Y-value for the right node. Similar to the linear model, there is only one independent variable, which is volume [LW$^+$02] [SLT$^+$03] [Seg04]. Mathematically, it can be written as

$$P_h - P_l = f_S(V) \tag{2.15}$$

where P_h is the highest price at that time, and P_l is the lowest price at that time, V represents the volume and f_S is the trained Random Forest regression function.

In this model, the linear regression from Python Sklearn.ensemble is used, with default options.

2.2.4 Support Vector Regression

Support Vector Regression predicts or estimates the output based on the inputs variables and Support Vector Classification classifies the outputs. Support Vector Machine is a machine learning model used for classification and regression analysis [Qui87] [Alt92] [Ho95] [RSTK03]. When SVM is used for classification, it separates a given set of binary labeled training data with a hyperplane that is maximally distant from them. If we assume the input data is $\mathbf{x^j} = (\mathbf{x_1^j}...\mathbf{x_n^j})$ being the realization of the random vector $\mathbf{x^j}$ while ϕ is the map mapping the feature space to a label space y, where label space contains many vectors, and is $(\mathbf{x^1}, \mathbf{y^1}), ...(\mathbf{x^m}, \mathbf{y^m})$; the SVM learning algorithm finds a hyperplane $(\mathbf{w}, \mathbf{b})$ such that the quantity:

$$\gamma = min_t y^i \{< \mathbf{w}, \phi(\mathbf{x^i}) > -\mathbf{b}\}$$

is maximized. In this equation, the dimension of ϕ is in dimension of $n \times m$ and $< \mathbf{w}, \mathbf{f}(\mathbf{x^i}) > -\mathbf{b}$ corresponds to the distance between point $\mathbf{x^i}$ and the decision boundary. γ is the margin and b is a real number. The kernel of this function is $\mathbf{K_{ij}} =< \phi(\mathbf{x^i}), \phi(\mathbf{x^j}) >$. Given new data x to classify, a label is assigned according to its relationship to the decision boundary, and the corresponding decision function is written as $f(x) = sign(< w, \phi(x) > -b)$

2.2.5 Limitations of Baselines

VAR

In the baseline, correlation is considered using the VAR model. VAR performs better in very short-term forecasting than a longer-term prediction. This means a lag of 1 prediction (a 15-minute interval) is more stable and accurate than longer lags. The VAR model can be easily influenced by a small amount of noise in previous observations, and the predicted sequence has higher variance. The minimum time scale is 1- minute time slices, but time slices from 1 minute up to 10 minutes have been tested, and it was found that a model built on them did not converge, which was caused by the low volume of data and small price changes when considering such a short period of time. It seems from the previous calculation/tests that a general rule exists that the more data there is, the more accurate it becomes for stock price prediction.

SARIMA

SARIMA integrates daily seasonality with autoregression and a moving average, which models the repeating daily pattern effectively and efficiently. However, SARIMA with daily seasonality is insensitive to special conditions such as special days like celebration days or national holidays. As these days are not taken into consideration mathematically in SARIMA, they end up being treated like an ordinary day, leading to high errors on or around these days.

Linear Regression

Linear regression limits the relationship of one or more variables to a linear relationship and is a very basic way of modelling. Linear regression is sensitive to outliers and requires

the data to be independent. However, in the finance, trading volume is related to how the price fluctuates, and can even be related to what day of the week it is. Linear regression is not necessarily the best representation for underlying data in systems that are non-trivial.

Support Vector Machine Regression

The performance of Support Vector machine regression relies on the choice of the kernels. A linear kernel is more stable for very short-term prediction but fails to achieve satisfactory accuracy in longer forecasting. Support Vector Machine regression has a variance that may cause instability, as shown in Figure 2.3, the further the prediction. This is a challenge for stock prices which are both hard to predict and is based on many factors like local and world events that develop over time and add noise, making it difficult to choose suitable kernels.

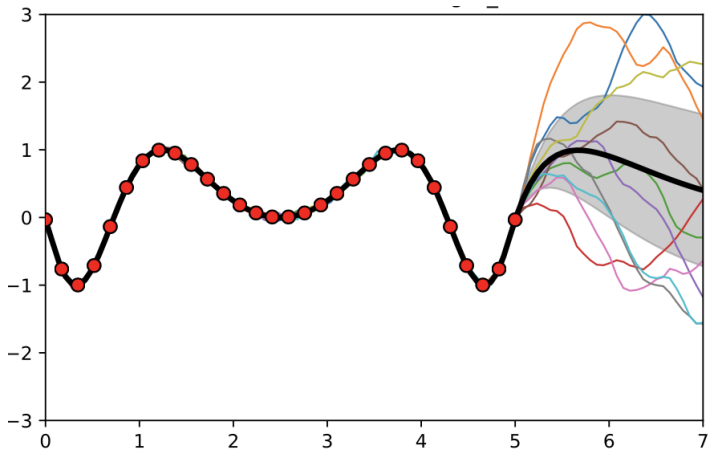


Figure 2.3: The variance of Support Vector Regression may become unstable to the right.

LASSO Regression

LASSO regression has shortcomings that if predictors are correlated, then it will only predict one of them. There are many different LASSO selection methods, and different software will get different LASSO result. There is no guarantee that the model that has been selected by the LASSO is not over-fitting.

2.3 Deep Learning Models

Deep learning, or Deep Neural Networks (DNNs), also known as Deep Structured Learning or Hierarchical Learning [Dec86] [Sch16], is part of the family of machine learning techniques [Nas07] [GH88]. Deep learning is inspired by systems that mimic the structure and function of the brain called an artificial neural networks [CM17] [GBCB16]. The network learns by using computational power to process a massive amount of data. In the network, the nodes/neurons act like sensors to learn and abstract the features; "deep" in this context means it is composed of multiple processing layers learning representations of data with multiple levels of abstraction. The layers include both visible layers (input and output) and hidden layers (designed to extract features and feed-forward the operations).

With bigger datasets and stronger computing power available now, more and more complex challenges can be solved by using Deep Learning models. Deep Learning models can produce results comparable to and in many cases, superior to human experts [KSH12]. There are many different deep learning models that have been applied to computer vision, e.g. Multilayer Perceptron (MLP), convolutional neural network (CNN), Deep Belief Networks (DBN), Generative adversarial networks. Recurrent neural network (RNN) have been used in natural language processing, audio generation and many other fields [VDOD16] [GPAM$^+$14] [KH09] [RMC15] [YZWY17]. Recent research has also explored

Long Short Term Memory (LSTM) for stock price forecasting [LYS09] [AMM05] [SPW98] [KSN06] [KM07] [ZAQ17]. Unlike standard feed-forward neural networks, LSTM has feedback connections and can process not only single data points (such as images) but also entire sequences of data (such as speech or video) [HS97]. In order to train a deep learning model, back-propagation is used as the method to calculate a gradient, where the gradient is used to calculate the weight of the network [GBCB16].

2.3.1 Multilayer Perceptron (MLP)

Multilayer Perceptron, or MLP, is a fully connected and dense network. MLP can be either a deep learning or a shallow learning model. An MLP model has at least three layers of nodes, and each node is a neuron that uses a nonlinear activation function apart from the input nodes. Examples of activation functions include sigmoids, rectifier and softplus. MLP utilizes supervised learning technique named back-propagation for training [Ros61] [RHW85]. The simplest MLP is a three-layer neural network, and it is shown in Figure 2.4. The mathematics equation for MLP is:

$$y = f^{(2)}(\mathbf{W^{(2)}}(\mathbf{f^{(1)}}(\mathbf{W^{(1)}}\mathbf{x} + \mathbf{b^{(1)}})) + \mathbf{b^{(2)}}) \tag{2.16}$$

where the input is vector $\mathbf{x}$, the output is $\mathbf{y}$, $\mathbf{W}$ and $\mathbf{b}$ are vector parameters, $\mathbf{W}$ represents the weight matrices and $\mathbf{b}$ is the noise, activation function is f. In this finance model, two cases are considered: one where $\mathbf{x}$ is a vector representing volume, and another representing volume and price. $\mathbf{y}$ is the predicted price.

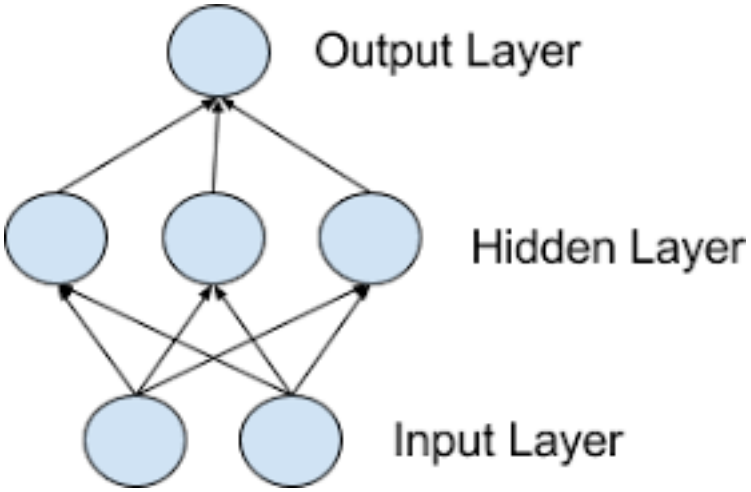


Figure 2.4: An example of a three layer MLP, from machinelearningmastery.com/when-to-use-mlp-cnn-and-rnn-neural-networks/

An example of a three layer MLP, sources from
https://machinelearningmastery.com/when-to-use-mlp-cnn-and-rnn-neural-networks/

2.3.2 Convolutional Neural Networks (CNN)

Convolutional Neural Networks (CNN) was originally inspired by the study of a cat's visual cortex, which contains a complex arrangement of cells [HW68] working together to process vision. Neurons are grouped together into different regions receiving inputs from different parts of the visual field, with other neurons acting as local filters over the input space and are used to exploit the strong spatially local correlation present in natural images. As shown in Figure 2.5, spatially-local correlation is represented as a local connectivity pattern between neurons of adjacent layers. Imagine the layer $m-1$ is the input retina, and the layer m has visual field of width 3 in the input retina, which means it is connected to 3 adjacent neurons in the retina layer and so to the similar connectivity with the layer $m+1$. In Figure 2.6, it shows that the three units in the layer m has the same feature map. A feature map means units share the same parameterization (weight vector and bias); this further indicates that weights of the same colour are shared and

numbered to be identical. Mathematically, it can be written as

$$h^{(k)} = tanh((W^{(k)} \times x) + b^{(k)}) \tag{2.17}$$

where k means the k-th feature map.

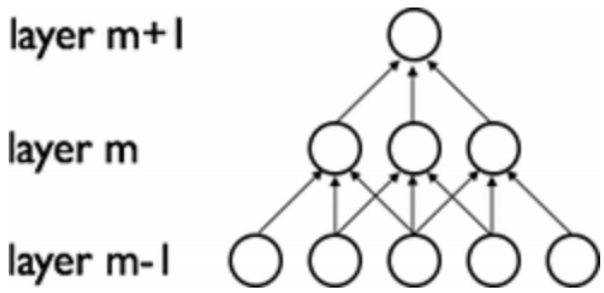


Figure 2.5: Sources from http://deeplearning.net/tutorial/lenet.html

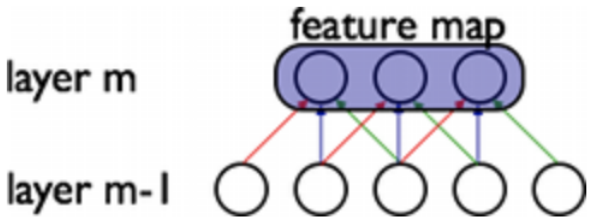


Figure 2.6: Sources from http://deeplearning.net/tutorial/lenet.html

In the research, CNN has demonstrated an outstanding capability for processing grid-like data [L+89]. CNN methods can be used in many circumstances in 1-D time-series samples with regular time intervals as well as 2-D grid pixels [GBCB16]. Mathematically, the 1D-signal CNN is written as:

$$o[P] = f[P] * g[P] = \sum_{u=-\infty}^{\infty} f[u]g[P-u] = \sum_{u=-\infty}^{\infty} g[u]f[P-u] \tag{2.18}$$

The mathematical equation of an extended 2-D grid can be written as:

$$o[V,P] = f[V,P]*g[V,P] = \sum_{u=-\infty}^{\infty}\sum_{v=-\infty}^{\infty} f[u,v]g[V-u,P-v] = \sum_{u=-\infty}^{\infty}\sum_{v=-\infty}^{\infty} g[u,v]f[V-u,P-v] \tag{2.19}$$

In both equations, V represent the trading volume and P represents the price of the stock.

The CNN model is a variant of MLP, and the main differences are the convolution operation and the pooling, described below.

Convolution Operation

As mathematically described in Equations 2.18 and 2.19, the beauty of the neural network is its ability to perform affine transformations: a vector is received as input and is multiplied with a matrix to produce an output, note a bias vector is usually added before passing the result through a non-linearity. Therefore, it can perform N-D convolutions.

The simplest discrete 2-D convolution is a linear transformation, see Figure 2.7, the light blue grid is called the input feature map, and the kernel is the dark blue grid with the sub-numbers.

In the CNN, kernels defining a discrete convolution has a shape corresponding to some permutation of $(n, m, k_1, ..., k_N)$ where n is the number of the output feature maps, m is the number of input feature maps, and k_j is the kernel size along axis j. The output size of a convolutional layer along axis j, represents as o_j are:

- i_j: input size along axis j
- k_j: kernel size along axis j
- s_j: stride(distance between two consecutive positions of the kernel) along axis j

- p_j: zero padding(number of zeros concatenated at the beginning and at the end of an axis) along axis j

In a different example shown in Figure 2.8, the kernel is 3×3, input pad is 5×5 with a 1×1 border of zeros using 2×2 strides.

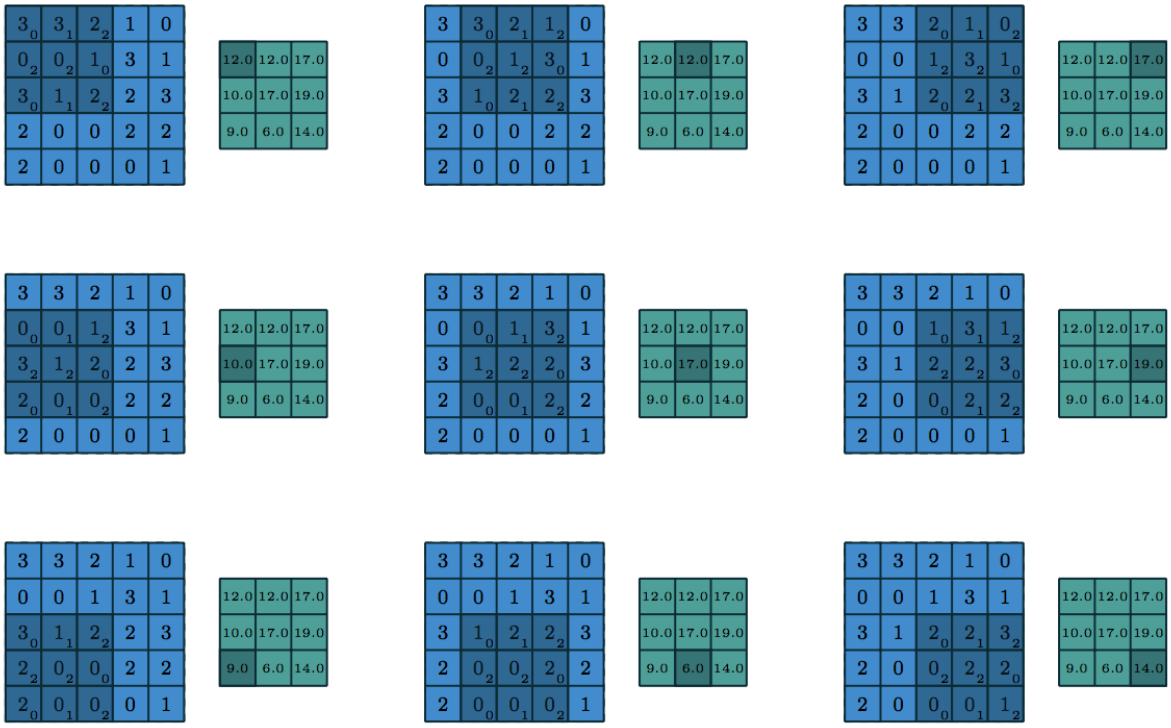


Figure 2.7: Computing the output values of a discrete convolution, sources from [DV16]

Pooling

Pooling in some senses is similar to discrete convolution. Pooling operates to reduce the size of feature maps by using some function to summarize subregions. One such function maximizes the value, and it called maximum pooling, see Figure 2.9. The output size o_j of a pooling layer along axis j are:

- i_j: input size along axis j

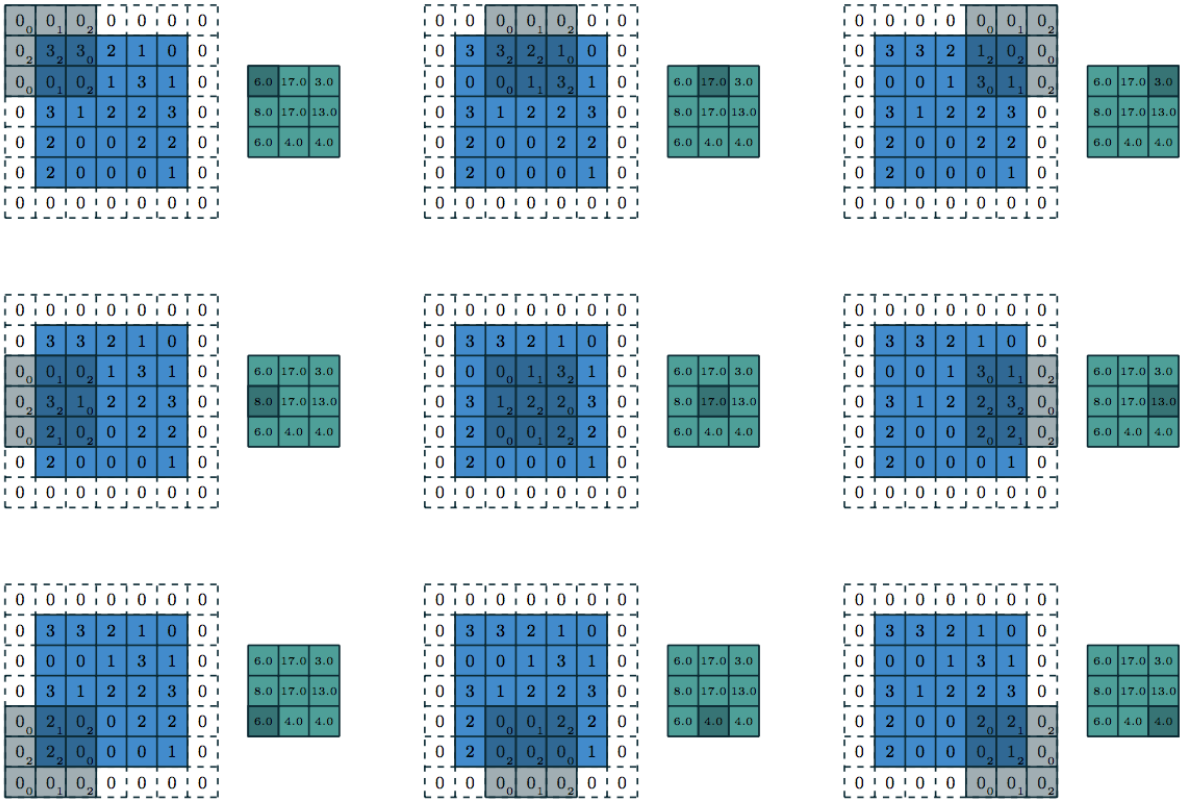


Figure 2.8: Computing the output values of a discrete convolution for $N = 2, i_1 = i_2 = 5, k_1 = k_2 = 3, s_1 = s_2 = 2$, and $p_1 = p_2 = 1$, sources from [DV16]

- k_j: poling window size along axis j
- s_j: stride (distance between two consecutive positions of the pooling window) along axis j

2.3.3 Recurrent Neural Networks (RNN)

Recurrent neural networks (RNN) are a family of neural networks for processing sequential datasets [GBCB16]. They differ from CNNs where the convolution operation allows a network to share parameters across time; the recurrent formulation results in the sharing of parameters through a very deep computational graph, which can be shown in Figure

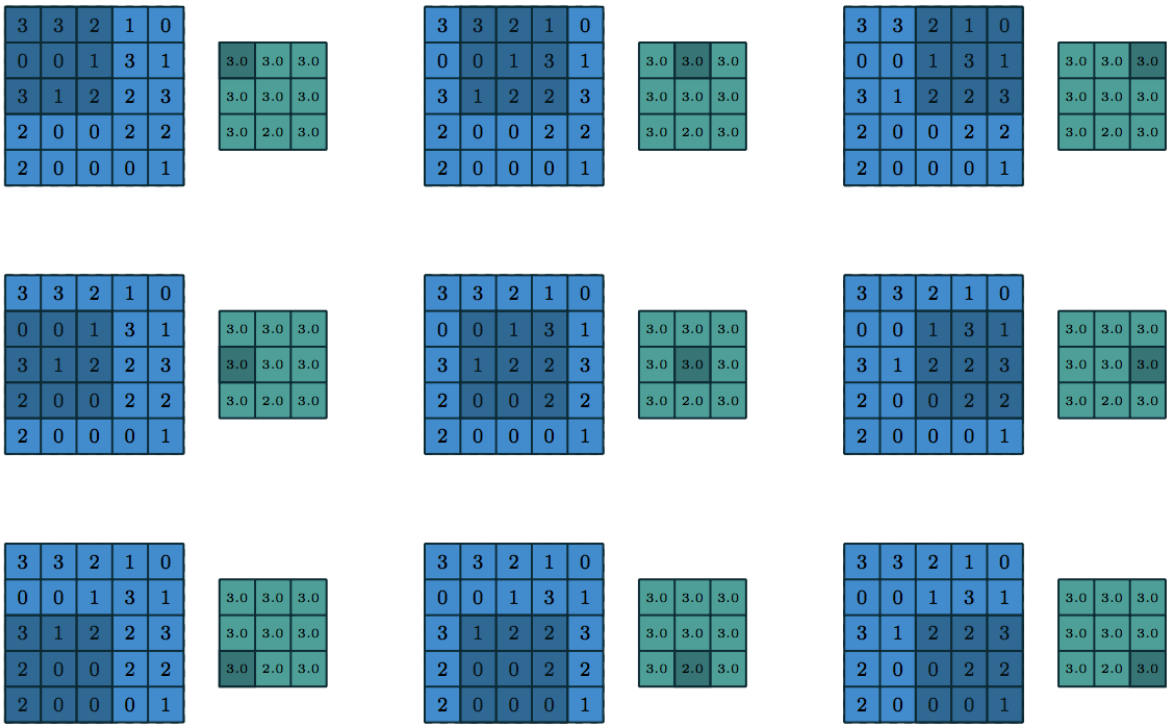


Figure 2.9: Computing the output values of a 3×3 max pooling operation on a 5×5 input using 1×1 strides, sources from [DV16]

2.10. In the figure, x is the input, which is a sequence of values, L is loss, y is the training target, input-to-hidden is matrix U, hidden-to-hidden is matrix W and hidden-to-output is matrix V. The connection from the previous output $y^{(t-1)}$ to the current state $h^{(t)}$ means that it is a conditional recurrent neural network.

There are a wide range of applications for RNN, including: handwriting recognition, speech recognition, stock prediction, and text prediction [GLF+09] [SSB14] [LW15] [PL05] [SPW98].

Long Short Term Memory (LSTM)

LSTM, or Long Short Term Memory networks, were first proposed by Hochreiter and Schmidhuber in 1997 [HS97]. An LSTM adds memory to an RNN to address the short-

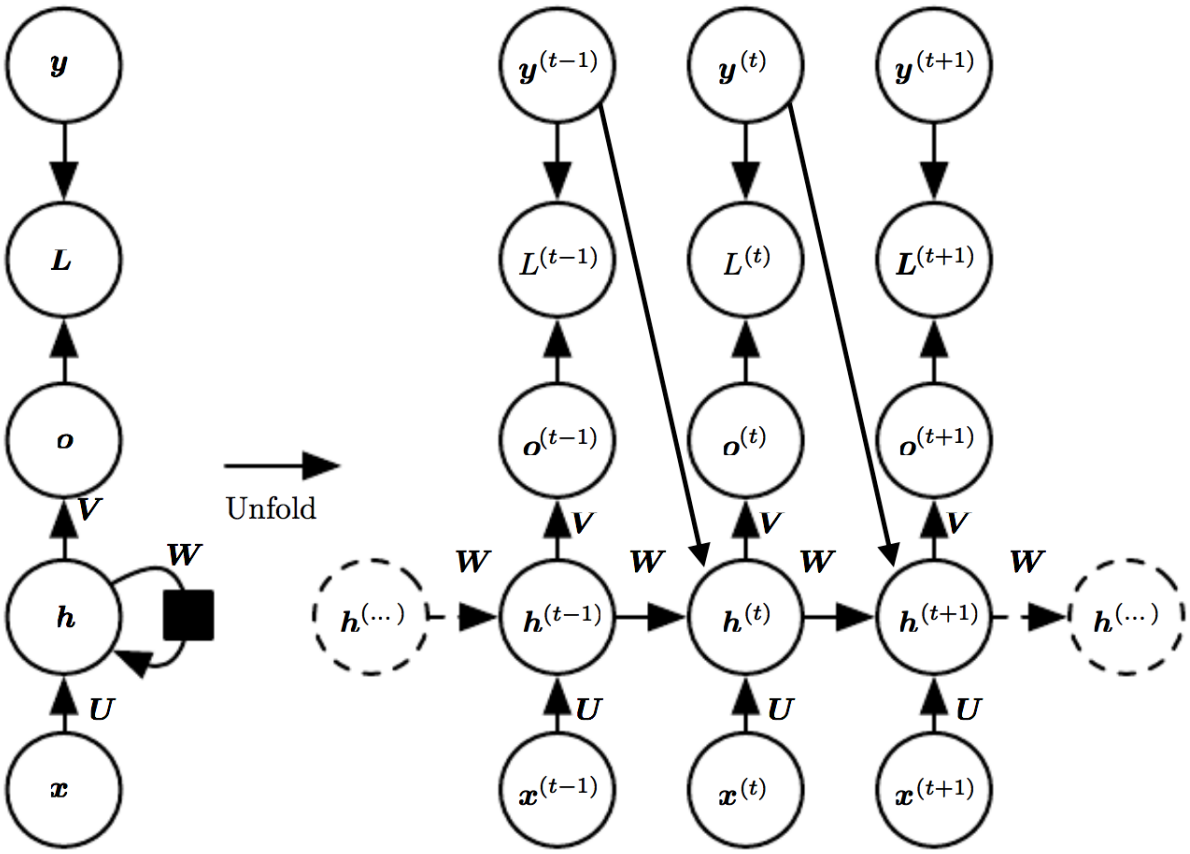


Figure 2.10: LSTM figure, sources from [GBCB16]

coming that using just the information available to an RNN around time step t may not be sufficient to model the process, and more information from further back in time may be necessary or beneficial [GBCB16]. An LSTM is represented as each matrix U in Figure 2.10, and information can be added or removed to the cell state and regulated by structures called gates. The structure of the LSTM is shown in Figure 2.11. In the diagram, the whole vector is carried by each line from the output of one node to the input of the others. The pink circle represent point-wise operations: vector addition or multiplication. The yellow boxes are learned neural network layers. Merging lines denote concatenation, while forking lines denote its content being copied to different locations.

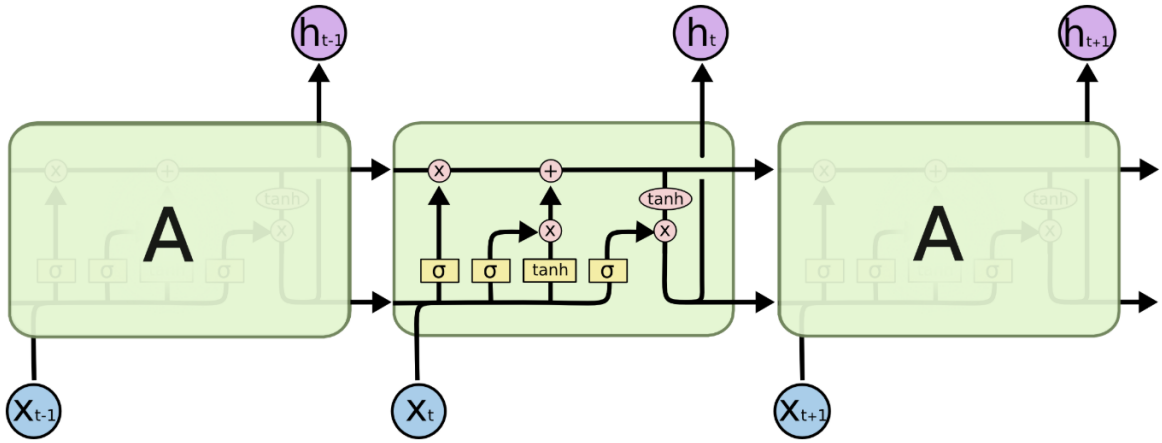


Figure 2.11: The repeating module in an LSTM contains four interacting layers, source from http://colah.github.io/posts/2015-08-Understanding-LSTMs/

2.3.4 Attention+RNN

In the recent Neural Network research, 'attention' has gained popularity because it can solve problems caused by sequences being too long and having not enough memory space in an RNN [GBCB16]. Uses for 'attention' in RNN models in the literature include translation and alignment of text, speech recognition, and natural language description of features of images [BCB14] [XBK+15] [VHGK14].

In the research described by this thesis, the attention mechanism used is described in the article 'Effective Approaches to Attention-based Neural Machine Translation', by Minh-Thang Luong in 2015 [LPM15]. The mechanism can be shown in Figure 2.12. Mathematically, the scale from the previous cells are calculated as

$$score(\mathbf{h_t}, \bar{\mathbf{h_s}}) = \mathbf{h_t}^{\mathbf{T}} \bar{\mathbf{h_s}} \tag{2.20}$$

It is then combined with the current cell $\mathbf{h_t}$, which gives the global aligned weight $\mathbf{a_t}$ as

$$\mathbf{a_t} = \mathbf{softmax}(\mathbf{W_a} \tilde{\mathbf{h_t}}) \tag{2.21}$$

The alignment vector $\mathbf{a_t}$ will then be used to derive a context vector $\mathbf{c_t}$ which is a weighted linear combination of all source hidden states and their corresponding alignment vectors.

$$\mathbf{c_t} = \sum_{\mathbf{s}} \mathbf{a_t h_s} \tag{2.22}$$

The attention hidden state $\hat{\mathbf{h}}_\mathbf{t}$ is the concatenation of $\mathbf{c_t}$ and $\mathbf{h_t}$, and will be transmitted to the FC decoder. $\bar{\mathbf{h}}_\mathbf{s}$ is the source hidden state. $\tilde{\mathbf{h}}_\mathbf{t}$ is the concatenation layer to combine the information from both vectors to produce an attentional hidden state as follow:

$$\tilde{\mathbf{h}}_\mathbf{t} = \mathbf{tanh}(\mathbf{W_c}[\mathbf{c_t}; \mathbf{h_t}]) \tag{2.23}$$

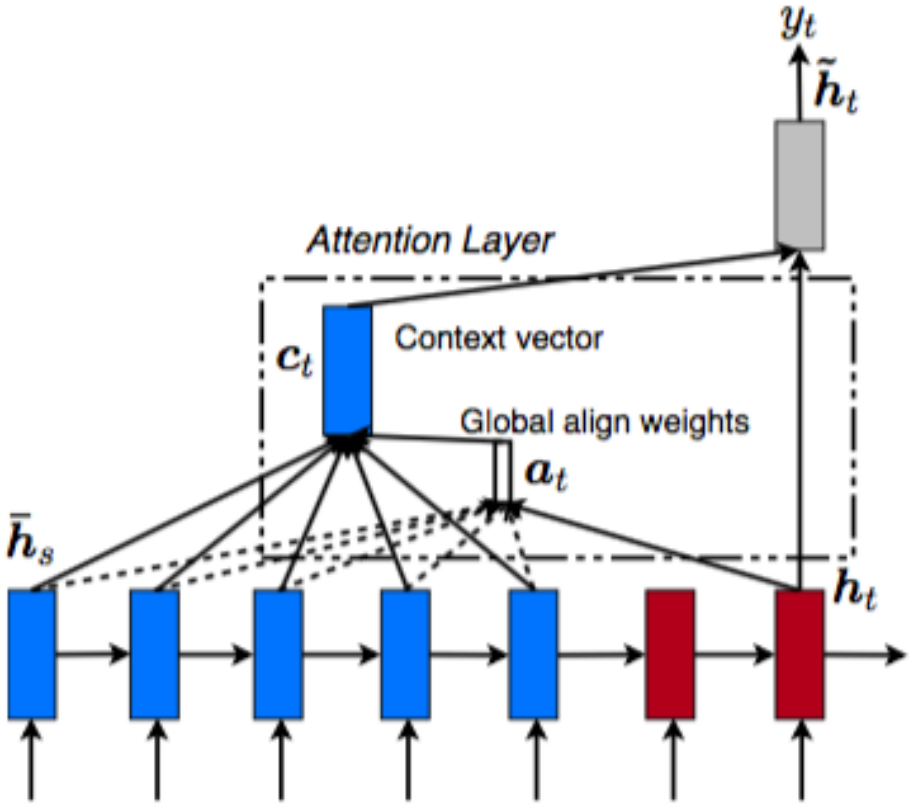


Figure 2.12: Global attentional model at each time step t, the model infers a variable-length alignment weight vector at based on the current target state ht and all source states $\bar{\mathbf{h}}_\mathbf{s}$. A global context vector $\mathbf{c_t}$ is then computed as the weighted average, according to $\mathbf{a_t}$, over all the source states, source from [LPM15]

2.3.5 CNN+RNN

In the literature, a combined CNN+RNN model has proven to perform better than the RNN/LSTM model alone in image caption/classification, video, speech and traffic modelling applications [WYM+16] [FLLL16] [SHB+12], therefore it is worth trying to apply it to stock price prediction.

In stock price prediction, the training dataset is very large. The question of how to extract the key features from each sequence is crucial to predict stock price accurately. CNNs can be used to solve this problem. A number of 2D CNN with a mixed of activation functions (i.e. ReLU and max pooling) is applied to reduce the length of the sequence and extract some features as a representation of the data. A CNN is able to capture patterns in a local region, process that sequence, and then feed into the RNN with LSTM.

2.3.6 Attention+CNN+RNN

In the literature, Wang et al. have applied attention+CNN+RNN model to multi-label image classification, and have found that accuracy is improved versus a CNN+RNN model [WYM+16]. In other research, when applied to other applications, Attention+CNN+RNN have been proven to be a better model than CNN+RNN and Attention+RNN models [YTC+15] [YSXZ15]. However, this method has yet to have been applied to finance.

This model is a combination of CNN, RNN and Attention applied to stock price prediction and is shown in Figure 2.13

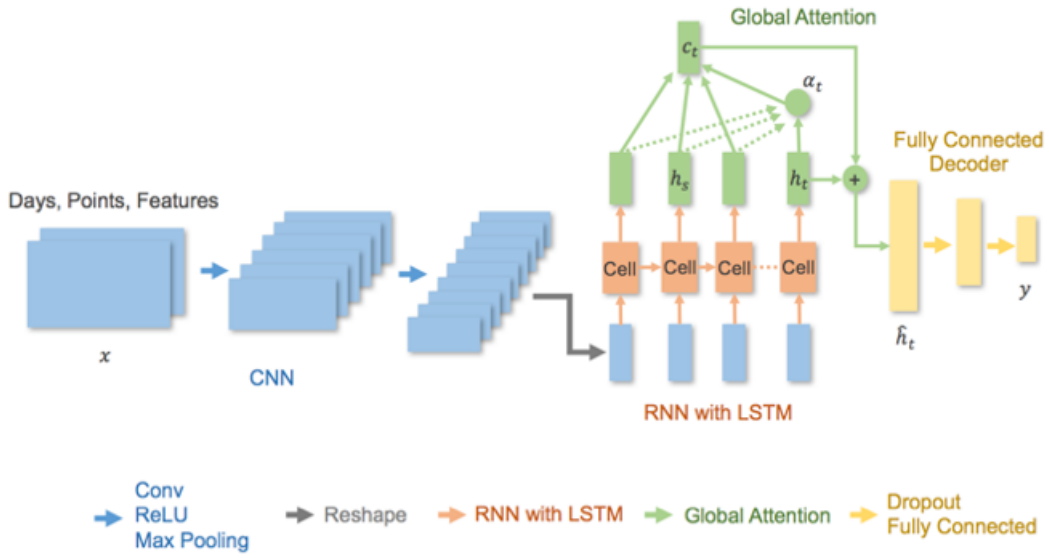


Figure 2.13: CNN-LSTM-Attn network structure. The CNN-LSTM-Attn consists of four sub-networks, CNN, RNN with LSTM, global attention mechanism and fully connected decoder.

2.4 Data Privacy and Security

Information privacy, also known as Data Privacy or Data Protection, is the relationship between the collection and dissemination of data, technology, and the public expectation of privacy, legal and political concerns surrounding them [Mic13]. Concerns over privacy in digital and non-digital forms exist wherever personally identifiable information or other sensitive information is collected, stored, used, and finally destroyed or deleted. One of the privacy issues that may arise in the financial industry is increasing information use. In the front office, there is trading data, and in the back office, there is other privileged information, potentially including the customer's salary, transaction data, and due diligence data. There is a rising sensitivity over data protection across all financial institutions.

Figure 2.14 illustrates a traditional deep network's architecture, which sends data from a local device to a server, using encryption to secure the data transmission. At the server,

the data is decrypted, and the original data is processed to find the result.

The general equation of a single layer can be written as

$$a = f(x * W + b) \tag{2.24}$$

where x is a row vector of original input data, W is the weight matrix, b is the row vector of biases, a is a row vector of activation outputs and f reflects the activation function such as sigmoid, hyperbolic tangent, softplus, and rectifier [GBB11], max-out [GWFM$^+$13].

From a security point of view, the activation output a can be captured by a network sniffer, the weight matrix W and bias b can be acquired by hacking or decompiling the software. As a result, given a, W, b and f, the input data x can be fully reconstructed by Equation 2.25 [1], where W^{-1} is inverse matrix of W and f^{-1} is the inverse activation function. Note W and f are both invertible. Therefore, encrypting the activation outputs a is possible for data privacy and security.

$$x = (f^{-1}(a) - b) * W^{-1} \tag{2.25}$$

[1] If the number of activation outputs equals or greater than the number of input data

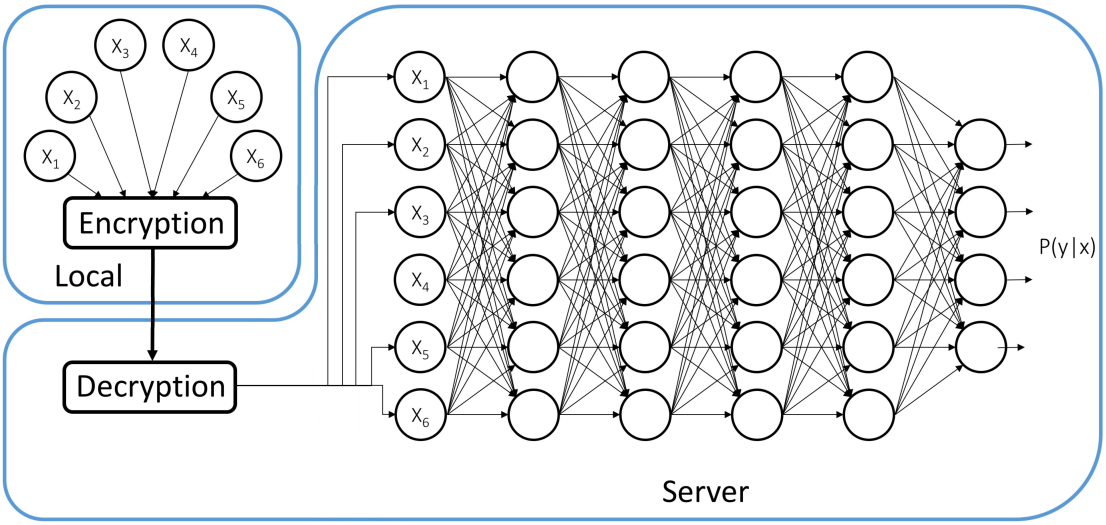


Figure 2.14: Feed-forward propagation in server; x is the input data, y is the output value, $P(y|x)$ is the probability of y given x.

Chapter 3

Stock Price Prediction

3.1 Data Collection

3.1.1 Machine Learning

Data for three companies over a period of February 2012 to August 2017 is collected from the Bloomberg Terminal. The companies are Microsoft, IBM and Google. Data is collected for each company at two different timescales: one with a data point every minute, and another with a data point every day. The dimensions of both timescale datasets include the date, time, open price, close price, highest price, lowest price, and volume that is being traded.

In the machine learning model, the input data is the volume column of this dataset, and the output data is the absolute value of the price change, which is calculated by subtracting the lowest price from the highest price for each data point.

As shown in Table 3.1 and Figure 3.1, the machine learning model is trained using different lengths of the timescale between January 2018 and May 2018. The results after

training and then validated using the testing dataset suggest that the highest accuracy does not necessarily come from having the largest training dataset. Moreover, it has also been noticed that the closer the training dataset is to the testing dataset, the higher the accuracy is likely to be. This makes sense in finance as the next day's stock price is more likely related to the previous day's stock price rather than the previous month's or previous year's stock price. In finance, the stock price is time-series dataset, so it would not be appropriate or useful to pick a training dataset that is ahead of the test dataset in time.

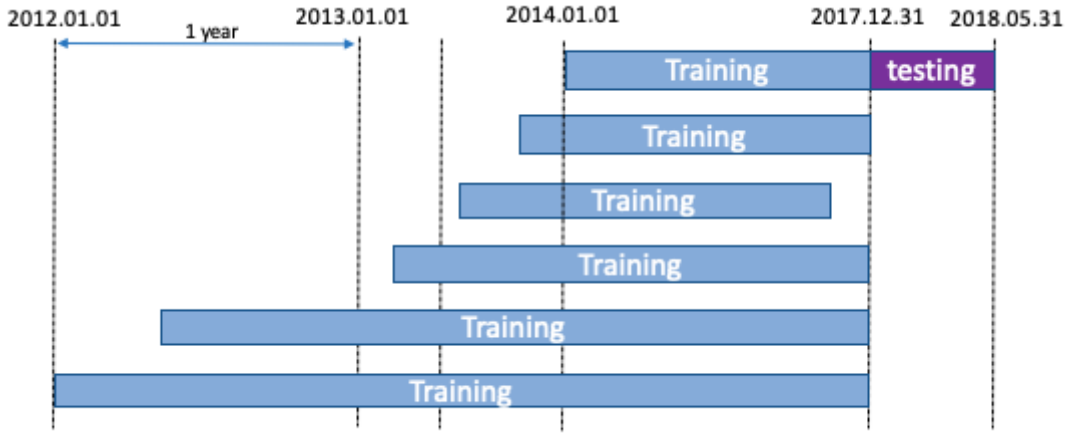


Figure 3.1: The figure of the training and testing dataset

3.1.2 Deep Learning

The stock price data is downloaded from Bloomberg in both minute and daily timescales; three companies are selected: Microsoft, Google and IBM. The fields provided in the dataset are shown in Figure 3.2 and Table 3.2. Similar to as described in the previous sections, multiple periods of time are picked to use as the training dataset to test the deep learning model. In Table 3.2, the dates are enumerated to: Monday as 1, Tuesday as 2, Wednesday as 3, Thursday as 4, Friday as 5, the day Britain voted to exit EU (23/06/2016)

Table 3.1: Training and testing data and the corresponding accuracy

Training time interval	Testing time interval	Accuracy prediction
01/2017-12/2017	01/2018-05/2018	74.0000%
11/2016-12/2017	01/2018-05/2018	76.0000%
02/2016-10/2017	01/2018-05/2018	70.0000%
07/2014-12/2017	01/2018-05/2018	75.0000%
01/2013-12/2017	01/2018-05/2018	79.4000%
01/2012-12/2017	01/2018-05/2018	83.0000%
01/2012-12/2016	01/2018-05/2018	75.0000%
01/2012-02/2017	01/2018-05/2018	77.0000%

as 6, the period of one week before and one week after (16/06/2016-29/06/2016) Britain voted to exit EU as 7, the day (08/11/2016) Donald Trump was elected as the 45th president of the USA as 8, the period of one week before and one week after the day Donald Trump was elected as 9, and the Christmas period of one week before and one week after Christmas (17/12-02/01) as 10.

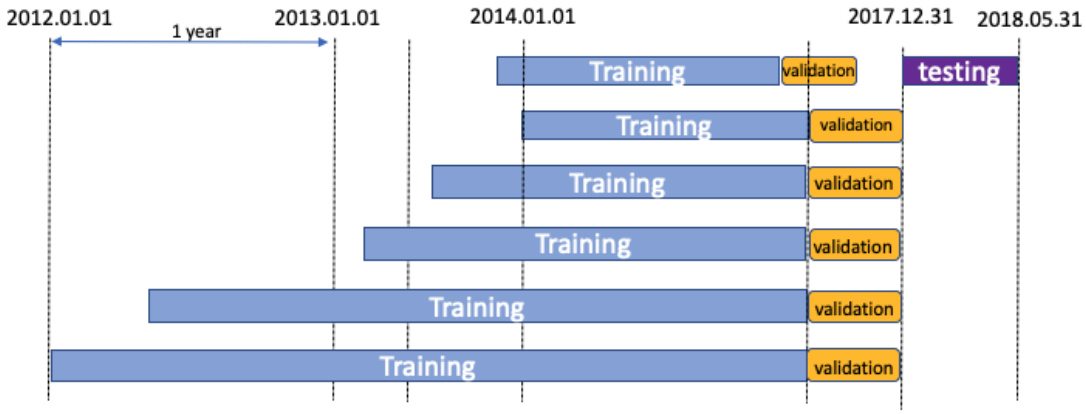


Figure 3.2: Figure of the data partition in deep learning training

Table 3.2: The fields of information provided in the model dataset

Fields	**Descriptions**	
Local Time	Local time (British Standard Time) with 15-minute time intervals. Normally, one and only data point for each 15-minute time slice.	
Input dataset	Days are classified into several categories, and valid ID includes:	
	1	Monday
	2	Tuesday
	3	Wednesday
	4	Thursday
	5	Friday
	6	Result day of Britain voted exit EU(23/06/2016)
	7	1 week before and 1 week after the result of Britain voted to exit EU(16/06/2016-29/06/2016)
	8	The day Donald Trump is selected as the 45th President of the United States(08/11/2016)
	9	1 week before and 1 week after the day Donald Trump is selected as the 45th President of the United States(01/11/2016-17/11/2016)
	10	Christmas period, 1 week before the Christmas holiday start and 1 week after the Christmas holiday(17/12-02/01)
Volume	The number of trading at that moment	
Close Price	The price closes at that moment	
Price change	The difference of the highest and lowest price at that moment	

Because the stock price dataset information is a time sequence, the data is divided into three parts: a training dataset is built using data from 01/2015 to 12/2015, a validation dataset is built using data from 01/2016 to 05/2016, and a testing dataset is built using data from 06/2016 to 12/2016. The daily data is shown in Table 3.3 and the minute data is shown in Table 3.4. The full batch size means the sum of the daily and minute data points.

Table 3.3: Daily Data Partition

company	data type	start day	end day	full batch size
Apple/IBM/Microsoft	train data	01/2012	05/2017	1394
	valid data	06/2017	12/2017	14
	test data	01/2018	05/2018	55

Table 3.4: Minute Data Partition

company	data type	start hour	end hour	start day	end day	full batch size
Apple/IBM/Microsoft	train data	09:30	16:00	01/2012	05/2017	34516
	valid data	09:30	16:00	06/2017	12/2017	3694
	test data	09:30	16:00	01/2018	05/2018	2574

3.2 Data Analysis and Refinery

Figures A.1, and A.2 show that daily and minute data are generally consistent in peaks and troughs. For example, daily and minute price change data peaks are both around

February of 2018; The price generally trends up in the daily and minute data between 05/2017-05/2018, and the troughs are around June 2017. However, while they are similar, they do not always line up, for example, the peak in daily volume is around February 2018, but the peak of the minute volume is around June 2017.

In general, Figures A.1, A.2, and A.3 show that there are periodic patterns to price change, close price, and volume in the daily dataset for the time period 05/2017-05/2018. The minute dataset also shows the same trends and general patterns as the daily dataset, but moreover shows more patterns within smaller timescales than the daily dataset. This prompts us to plot the figures of monthly and weekly price change, close price, log volume and volume, seen in Figures A.4, A.5, and A.6.

Figures A.4, A.5, A.6, and A.7 show patterns and trends in the monthly and weekly timescale for price change, close price, log volume and volume. A period of October 2017 is used for illustrative purposes.

The monthly price change between $01/10/2017 - 28/10/2017$ is shown as Figure A.1a and no strong patterns are apparent in the weekly data within the four weeks of that month. Zooming into the dataset of that month to $02/10/2017 - 06/10/2017$, the five-day price change is shown as Figure A.1b. It shows that the daily price change decreases in the morning, and then fluctuate up and down until at the end of the day, the stock price change rises above the minimum price change of the day. There is no readily apparent pattern in the weekly and daily close price, as shown in Figure A.5. Weekly log(Volume) and volume likewise don't show any obvious patterns. Daily log(Volume) and volume show a pattern where volume and log(Volume) fall at the beginning of the day and then rise again at the end of the day. See Figure A.6 and Figure A.7. The periodic pattern appears both in the daily dataset and in the weekly dataset, this periodic pattern motivates further study, which is discussed later in this section.

3.2.1 Error Calculation

Reduce Mean

In the results, the daily error and minute error are calculated using the equations below, where P_i represents the price, and $\hat{P}_i$ represents the predicted price, i can either be daily data or minute data.

$$MAPE_i = \frac{|P_i - \hat{P}_i|}{P_i} \tag{3.1}$$

MAPE is selected as it is common method used in finance research [WWZG12] [MO97] citekim2016new. This method is most appropriate when one of the input data is price change, which is a positive number. Note the accuracy is more likely to be lower than MAPE in this model because of how the accuracy is calculated; in equation form this is:

$$accuracy = \frac{\text{number of correct predictions}}{\text{total number of predictions}} \tag{3.2}$$

Mean Square Error

MSE, or Mean Square Error, is a measure of the quality of the estimator. It is always a positive value. Under normal circumstances, the smaller the MSE_i is, the better the estimator is [WS08]. It is mathematically written as

$$MSE_i = \frac{1}{n}\sum_{i=1}^{n}(P_i - \hat{P}_i)^2 \tag{3.3}$$

3.2.2 Special Days

In the previous analysis, trading data features have been analyzed on a daily time scale as well as on a weekly time scale. However, there are some special-case days where the

series may behave differently. As seen in Table 3.2, several special days are defined: the Christmas period, the day and corresponding close period of Donald Trump being elected as the president of the USA, and the day and the corresponding close period of Britain voting to exit the European Union. In addition to the special days, the workdays also have their own class as there is also variability due to the days. Some measurements to help us find the difference between workdays and special days are defined below:

1. The dataset spans from January 2012 to May of 2018. Two datasets are used, one with data points every day (daily data), and one with data points every minute (minute data).

2. Peak hours: The moments that have the highest 25% of the selected data feature

3. Percentage difference of all hours:

$$\frac{mean(x_t|t \in T^{all}_{workdays}) - mean(x_t|t \in T^{all}_{specialdays})}{mean(x_t|t \in T^{all}_{workdays})} \tag{3.4}$$

   where $T^{all}_{workdays}$ is a set of data with workdays, and $T^{all}_{specialdays}$ is a set of data with special days.

4. Traders can trade assets at different time scale, for example, they can do daily trading or weekly trading

Christmas Period

Table 3.5 is a comparison between non-Christmas and Christmas periods on trading volume, close price and the highest and lowest price change. This table also makes a comparison between three tech companies: Apple, IBM and Microsoft. It shows that, in general, the price change is more volatile in the daily dataset than the minute dataset; volume is

more volatile in the minute dataset than the daily dataset; and the close price percentage difference mean is sometimes negative, which means the stock price during the Christmas is higher than the stock price during a none-Christmas period, while volume and price change all have positive numbers.

Figure A.9 shows a comparison of a Christmas period and a non-Christmas period for volume, close price and price change. The red line is the top 25% of the dataset in each case (volume, price and price change). In Figure A.8 it shows that the price change is less volatile during the Christmas period compared to the non-Christmas period, especially the top 25% is below 0.5 during the Christmas period, while above 0.5 during the none-Christmas period.

Weekday Differences

Plots for Apple data for the weekdays cases are shown in Figures A.11, A.12, A.13, A.13, A.14, and A.15. In general, it shows that the price change trend is consistent to the volume trend, as seen in Figure A.13a and A.13c, which indicates that the maximum price change and volume are on Wednesday around 11:00 am on the 17th of May 2017. Comparing the price change and volume of the weekdays shows that Wednesdays have the most volatile price change and volume; while Tuesday and Thursday have a flatter price change and volume. This result is also consistent with the finding that many traders make weekly trades on Tuesday and Thursday [Har86] [CS00]. Figure A.15c shows that there is a larger volume of trading around 10:00 am than other times of the day on every Friday in May 2017.

The first Monday, Tuesday, Wednesday, Thursday and Friday of every month in 2017 are plot in Figures A.16, A.17, A.18, A.19, and A.20. July, November and December are the most volatile months of the year with respect to the volume and price change, seen

Table 3.5: A comparison between non-Christmas and Christmas periods on trading volume, close price, and the highest and lowest price change

			MEAN
Apple	**Daily**	**Volume percentage difference**	1.5000%
		Close price percentage difference	1.8000%
		Price change percentage difference	5.1000%
	Minute	**Volume percentage difference**	20.2000%
		Close price percentage differe000nce	-2.0000%
		Price change percentage difference	7.0000%
IBM	**Daily**	**Volume percentage difference**	1.0000%
		Close price percentage difference	4.3000%
		Price change percentage difference	12.6000%
	Minute	**Volume percentage difference**	12.4000%
		Close price percentage difference	5.0000%
		Price change percentage difference	8.0000%
Microsoft	**Daily**	**Volume percentage difference**	1.4000%
		Close price percentage difference	-7.7000%
		Price change percentage difference	4.6000%
	Minute	**Volume percentage difference**	17.7000%
		Close price percentage difference	-8.0000%
		Price change percentage difference	3.0000%

in Figure A.20a and A.20c. Price change and volume fluctuate most in the morning in November and December, while price change and volume fluctuate most in the afternoon in July. January is the flattest month of the year. In the price Figures A.20b, A.17b, A.18b, A.19b, and A.20b, it is show that the January is normally the lowest stock price of the year, and prices increasing gradually in following months. In three pairs of months in 2017: April and May; in September and October; and in November and December, stock prices were very close to each other.

The close price and volume for the first Monday, Tuesday, Wednesday, Thursday, and

Friday for every February from 2012 to 2017 are shown as Figures A.21, A.22, A.23, A.24, and A.25. As can be observed in Figures A.21a, A.22a, A.23a, A.24a, and A.25a, the more recent the year, the more volatile the price change was. However, the volume shows the opposite trend, with volume reducing over time, as can be seen in Figures A.21c, A.22c, A.23c, A.24c, and A.25c; with the highest volume in the year 2013. This result is consistent with the current economic situation: in the 2012 and 2013, the economy was is in recovery, and the public was confident in the stock market and invested heavily; while 2018 is part of what some consider a ten years cycle from the last crisis in 2008, people are not as confident of the current economic market [Roo13] [Bai18] [Fri18]. Figures A.21b, A.22b, A.23b, A.24b, and A.25b also show that the price increased slowly in the February of year 2012 and 2013 every day until 2015. In year 2016, the February price dropped. 2017 and 2018 are then two consecutive increase years.

In this section, the tables are used to see the percentage difference of mean values each day in the minute data for Apple, IBM and Microsoft. These are the indicators used for picking inputs to models, and more importantly, these are the quantitative methods used to draw trading patterns.

Table 3.6 is a comparison within weekday trading volumes, close price, and highest and lowest price change for Apple. Table 3.7 shows the same for IBM, and Table 3.8 shows the same data for Microsoft.

In Table 3.6, the price change has a higher daily and minute percentage mean difference than volume and price change. Also, there can be no certainty of whether the percentage mean difference is positive or negative in the daily and minute dataset. Besides, even the highest percentage mean difference is below 20%.

Table 3.6: Comparison within weekdays on trading volume, close price and the highest and lowest price change for Apple.

Company	Days	Percentage difference	Minute Mean	Daily Mean
AAPL	**Monday**	**Volume percentage difference**	0.7000%	-0.0194%
		Close price percentage difference	0.1770%	-0.0460%
		Price change percentage difference	-1.7700%	-10.8590%
	Tuesday	**Volume percentage difference**	-3.1000%	-0.2100%
		Close price percentage difference	0.0140%	0.1270%
		Price change percentage difference	-4.2700%	-5.1550%
	Wednesday	**Volume percentage difference**	-6.4300%	-0.2640%
		Close price percentage difference	-0.4600%	-0.3770%
		Price change percentage difference	-6.8200%	-1.5600%
	Thursday	**Volume percentage difference**	2.9500%	0.1327%
		Close price percentage difference	-0.2500%	-0.0990%
		Price change percentage difference	0.7290%	5.0050%
	Friday	**Volume percentage difference**	-1.4300%	-0.1300%
		Close price percentage difference	0.5470%	0.4630%
		Price change percentage difference	9.2500%	9.3038%

Table 3.7 is the weekday trading volume, close price, price change minute and daily mean for IBM. Compared to Table 3.6, there is no significant difference to the minute/daily mean between Apple and IBM, all the numbers are below 10% in the Apple case.

Table 3.7: Comparison within weekdays on trading volume, close price and the highest and lowest price change for IBM.

Company	Days	Percentage difference	Minute Mean	Daily Mean
IBM	**Monday**	**Volume percentage difference**	5.9500%	0.2670%
		Close price percentage difference	-0.5020%	-0.4050%
		Price change percentage difference	4.0910%	4.6548%
	Tuesday	**Volume percentage difference**	-1.9600%	-0.1600%
		Close price percentage difference	-0.3600%	-0.2950%
		Price change percentage difference	-0.1600%	-0.7110%
	Wednesday	**Volume percentage difference**	-4.2100%	-0.1330%
		Close price percentage difference	-0.2780%	-0.2000%
		Price change percentage difference	-4.6000%	-6.1280%
	Thursday	**Volume percentage difference**	-1.0900%	-0.1220%
		Close price percentage difference	-0.6100%	-0.5100%
		Price change percentage difference	-5.4100%	-5.1650%
	Friday	**Volume percentage difference**	-2.9000%	-0.1176%
		Close price percentage difference	-0.4300%	-0.3322%
		Price change percentage difference	3.5400%	4.3076%

Table 3.8 shows the same data but for Microsoft. The volume percentage difference in the Friday minute dataset is about -20.5%, this result is also consistent with the previous figure that shows Fridays having higher volatility compared with other days of the week. All other figures are lower in the Microsoft data compared to Apple and IBM.

Table 3.8: Comparison within weekdays on trading volume, close price and the highest and lowest price change for Microsoft.

Company	Days	Percentage difference	Minute Mean	Daily Mean
Microsoft	Monday	**Volume percentage difference**	8.3000%	0.3900%
		Close price percentage difference	0.5200%	0.4320%
		Price change percentage difference	6.1600%	3.3350%
	Tuesday	**Volume percentage difference**	3.3460%	0.1100%
		Close price percentage difference	0.6840%	0.7500%
		Price change percentage difference	1.6600%	0.8500%
	Wednesday	**Volume percentage difference**	3.0000%	0.0520%
		Close price percentage difference	0.1700%	0.1580%
		Price change percentage difference	-1.7500%	0.0960%
	Thursday	**Volume percentage difference**	0.6000%	-0.1050%
		Close price percentage difference	0.3950%	0.4730%
		Price change percentage difference	-3.6000%	-0.7300%
	Friday	**Volume percentage difference**	-20.4900%	-0.7900%
		Close price percentage difference	0.6100%	0.4700%
		Price change percentage difference	-2.5000%	-4.1700%

Political special days: the Day Britain Voted to Leave the European Union& the Day Donald Trump was elected as President of the USA

Table 3.9 is a comparison of Apple trading volume, close price, and price changes on the day Britain voted to leave the EU, the day of Donald Trump's election, as compared with normal trading days. Table 3.10 is the same period for IBM, and Table 3.11 is the same period for Microsoft. Comparing means from normal days to those for these two events, it is very obvious that there are volume, close price and price change percentage differences. In the minute mean for both cases, volume differences were notably large. In the daily mean, the price change percentage difference is instead more significant than the other two.

Comparing the Brexit and USA election cases, the days of the USA election appear to have had a greater influence on volume, close price, and price change than the Brexit event. One likely reason is that Apple, being a US company, is seen to be more impacted by US events than non-US events. It is also observed that closing price and price change percentages went in opposite directions for the Brexit and the Election cases, so apparently, have opposite influences.

Similarly to the Apple table, the tables for Microsoft (Table 3.11) and IBM (Table 3.10) both show a bigger change in the daily and minute mean during the USA election and Brexit events compared to regular weekdays, however, Microsoft's changes are not as significant when compared to Apple and IBM, as shown in Figures A.28, A.27, and A.26.

In conclusion, the price change for three companies are seen to fluctuate more during the day of the USA election, and on the other hand, the volume during the day of the USA election is more volatile than during the day Britain voted to leave the EU.

Table 3.9: A comparison of Apple during the day Britain voted to leave the European Union, and the day Donald Trump was elected, compared with other normal days of the year on trading volume close price and the highest and lowest price change

			Daily Mean	**Minute Mean**
Apple	**The day Britain Votes to Leave the European Union**	**Volume percentage difference**	1.6560%	38.7000%
		Close price percentage difference	11.7860%	12.1400%
		Price change percentage difference	20.5108%	17.5900%
	Donald Trump elected president day	**Volume percentage difference**	2.2100%	43.9300%
		Close price percentage difference	-1.8410%	-1.3620%
		Price change percentage difference	-35.6100%	-21.8100%

3.2.3 Data Refinery

Validation

Due to some mechanical fault of the machine or special event such as trading being halted, if the exchange for trading is partly closed between the normal open 9:00-16:30 hour, resulting in data missing for part of the day, the whole day is eliminated for consideration in the model.

Table 3.10: A comparison of IBM during the day Britain voted to leave the European Union, and the day Donald Trump was elected, compared with other normal days of the year on trading volume close price and the highest and lowest price change

			Daily Mean	**Minute Mean**
IBM	**The day Britain Votes to Leave the European Union**	**Volume percentage difference**	0.5600%	11.4900%
		Close price percentage difference	12.4100%	12.4300%
		Price change percentage difference	-10.0500%	0.6640%
	Donald Trump elected president day	**Volume percentage difference**	1.2600%	18.8700%
		Close price percentage difference	8.8900%	8.9750%
		Price change percentage difference	-6.7500%	-6.6500%

De-duplication

In data de-duplication, we eliminate duplicate copies of repeating data points. In the minute dataset, data is sampled every 15 minutes, which means there is supposed to be one and only one data point in each 15-minute time slice. However, the raw data is not necessarily clean and may contain duplicate data points. To deal with this, data points are first grouped into 15-minute slices; then an average is taken for groups with multiple data points.

The minimum time scale possible is 1-minute time slices, but time slices between 1-minute and 10-minutes have been tested and found that the models built on them do not converge,

Table 3.11: A comparison of Microsoft during the day Britain voted to leave the European Union, and the day Donald Trump was elected, compared with other normal days of the year on trading volume close price and the highest and lowest change

			Daily Mean	**Minute Mean**
Microsoft	**The day Britain Votes to Leave the European Union**	**Volume percentage difference**	-1.0310%	-13.4600%
		Close price percentage difference	-1.6700%	-1.6420%
		Price change percentage difference	-14.4300%	-12.5800%
	Donald Trump elected president day	**Volume percentage difference**	0.2030%	9.6700%
		Close price percentage difference	-20.4300%	-20.4000%
		Price change percentage difference	-34.7160%	-37.0300%

which could be caused by the lack of volume within such a short period, or due to the price change being too small to be used in calculation. From the previous calculation/test, it seems that there is a general relationship between accuracy and quantity of data for the stock price prediction.

Feature Selection

Based on the literature review and the physical meaning of the data in the model, the volume, highest and lowest price of the day, the close price and the day ID are selected to be combined together in different combinations in the model. To avoid duplication, the

models' input dataset is carefully selected.

3.3 Methodology and Implementation

3.3.1 Implementation

The implementation of deep neural networks is based on TensorLayer [DSM+17], a high-level library for TensorFlow [AAB+16] described as "taking advantage of functional programming and object-oriented programming and provides agile project development with simple structure, friendly interfaces, minimum redundancy and sufficient flexibility".

Figure 3.3 demonstrates the structure of the project.

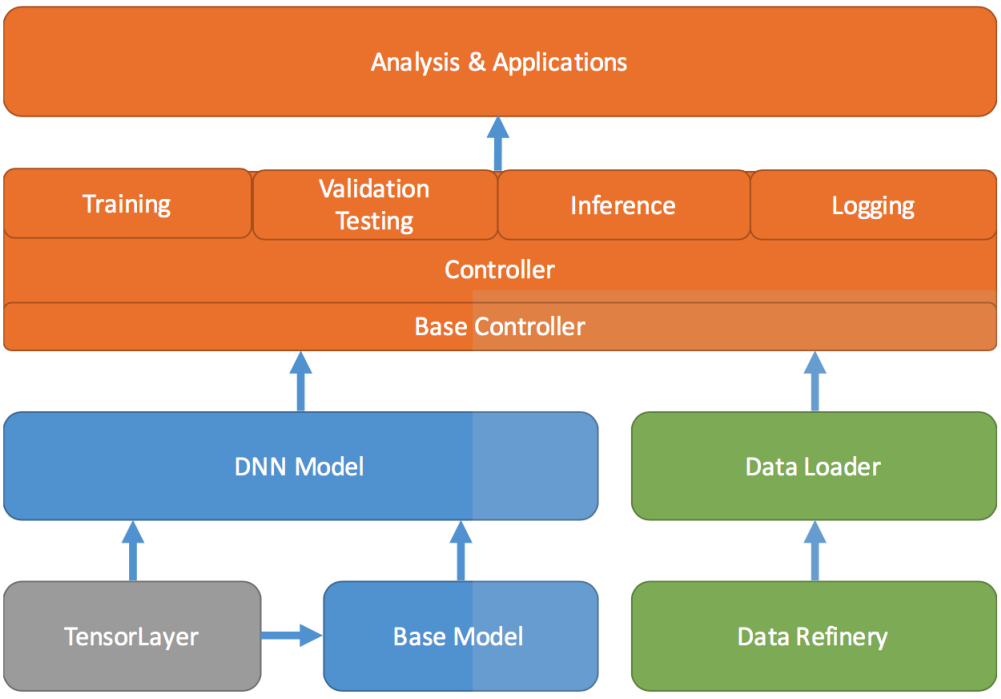


Figure 3.3: Project structure. All deep neural network models are implemented on TensorLayer and the base model. Given the model and refined data, the controller defines the process for training, validation, testing and inference. Applications, including analysis and visualisation, are built upon prediction results.

1. The green section supports data refinery, data analysis and data loading for controllers. Results of data analysis and methods of data refinery have been discussed in the previous chapter.

2. In the blue section, we firstly define a base model, which integrates several fundamental interfaces including placeholders, training operations, loss functions, network structure and models. New deep neural networks inherit the base model and acquire all features of the superclass. The only modification needed is to update the network structure. Besides, methods can be overridden if any other interfaces need updating. For example, we can first build a simple RNN for the prediction problem, and then other models can share the same definitions of input, output, loss functions and optimisers, and only need a new network structure. This means the RNN here can have different settings that change some variables in the model; for example, the activation function can be easily switched between RELU, Sigmoid and others.

3. The orange section implements controllers and applications; this means this part is a combination of data and the deep learning structure. Similarly, the base controller defines the training, testing and inference process. Some functionalities such as model saving/loading can also be implemented within the base controller. Generally, there should be one controller for each DNN model. Most controllers share exactly the same process for training/testing/inference, so we only need to update some basic settings, e.g. the path of the directory for logging.

3.4 Experiment and Evaluation

3.4.1 Machine Learning

As shown in the previous data collection process, the evaluation of the training on different datasets uses the machine learning methods that are listed in Table 3.12. The results in this table indicate that minute dataset results are generally better than the daily dataset result. The table also shows that Random Forest regression is better than linear regression, and linear regression is better than LASSO regression. The table shows that the best training dataset is not necessarily the largest dataset; it is also not necessarily the closest training dataset to the test dataset. However, training dataset with longer periods helps to reduce the error. The table shows that there is no significant difference between the three machine learning methods (Linear regression, Lasso Regression and Random Forest regression) after calculating the ANOVA.

Table 3.12: Machine learning Apple 2018 January to May data MAPE prediction when the training dataset is over different intervals

training time interval	Linear Regression		Lasso Regression		Random Forest regression	
	daily	minute	daily	minute	daily	minute
01/2012-12/2012	53.4000%	52.9000%	53.7300%	26.5100%	50.7000%	56.3100%
01/2013-12/2013	59.9900%	61.0000%	50.2000%	45.1000%	55.3400%	57.8000%
01/2014-12/2014	44.9000%	46.7000%	41.2000%	48.2000%	40.4100%	50.1230%
01/2015-12/2015	30.2300%	30.9000%	39.6000%	37.8000%	34.5000%	32.1800%
01/2016-12/2016	39.6000%	39.9000%	30.1000%	38.9000%	30.4500%	32.1800%
01/2017-12/2017	13.7000%	13.6000%	27.0000%	31.0000%	23.6100%	27.8300%
01/2012-12/2014	51.2000%	50.3000%	28.4000%	26.1000%	37.1000%	32.1900%
01/2014-12/2017	29.9700%	32.2000%	24.1000%	23.8000%	23.2000%	21.34 00%
01/2012-12/2017	34.4000%	38.2000%	26.7000%	25.9000%	19.8720%	20.4500%

3.4.2 Deep Learning

In the Deep Learning model, the following different input datasets are tested:

- The input data is a two-dimensional vectors of the day ID and close price. The output data is the close price.
- The input data is a two-dimensional vector of the day ID and price change. Price change is calculated by subtracting the lowest price from the highest price. The output data is the price change.
- The input data is a three-dimensional vector of the day ID, volume, and the price change. Price change is calculated by subtracting the lowest price from the highest price at that timescale. The output data is the price change.
- The input data is a two-dimensional vector of the day ID, volume, and the close price. The output data is the close price.
- The input data is a three-dimensional vector of the day ID, volume, and the log(close price). The output data is the log(close price).

Table 3.13 is the result of using the different input and output datasets with the Deep Learning model. The table indicates that though inputs have different dimensions, the error for each case does not differ by much, and results generally around 11% MAPE error. Note the lowest MAPE error is the model with Days ID+Price change+log(Volume)+Volume, and thus the deep learning model used in later sections are based on this. Though log(volume) and volume are correlated, in deep learning model, it does not make too much difference in term of MAPE prediction. The table shows that there is no significant difference between all these deep learning methods in terms of the accuracy of prediction for one sample t test.

Table 3.13: CNN-LSTM-Attn model settings use the minute dataset prediction

Model	Predictions MAPE
Days ID+Volume	10.7000%
Days ID+Close price	11.0000%
Days ID+Close price+Volume	10.9800%
Days ID+Close price+log(Volume)	10.4200%
Days ID+Close price+log(Volume)+ Volume	10.5100%
Days ID+Price change	11.0900%
Days ID+Price change+Volume	11.0860%
Days ID+Price change+log(Volume)	10.4000%
Days ID+Price change+log(Volume)+Volume	10.1000%
Days ID+log(Close price)	10.9000%
Days ID+log(Close price)+Volume	10.9020%
Days ID+log(Close price)+log(Volume)	11.1000%
Days ID+log(Close price)+Volume+log(Volume)	10.4300%
Days ID+price change+Close price+Volume	10.7200%
Days ID+price change+Close price+Volume+log(Volume)	11.1200%

Model Settings

The minute dataset is used in a CNN-LSTM-Attn model, with details shown in Table 3.14. The table shows the input dataset; the model, which is a combination of CNN and RNN, with Attention added to the RNN; and the output dataset. The prediction can be changed accordingly based on the time scale. The table also notes how initial conditions are set up. In this setting, the CNN has been set up with only one layer because the input

dataset is small in dimension, and one layer of CNN is enough to extract the necessary features. In the CNN, 32 filters are used because the number of filters is the power of 2, and each filter is 5×13 because there are five days, and 13 makes the 130 data points a week evenly divisible. A single layer LSTM is used because multiple layers were tried and tested, and a single layer found to perform good enough. Two layers of Dropout and Denselayer are used as this is a classic configuration.

As shown in Table 3.15 of MAPE (%)the machine learning methods were all worse than the deep learning method, with a difference of about one-third of the error from the machine learning methods. It also shows that the longer the predictions period is, the higher the error becomes. In fact, the machine learning error rate at 15 minutes is approximately the same as the deep learning method at about 60 minutes, which is about 9.2%. It is also worth noting that the error rate MAPE in the machine learning and ARMA family models are increasing significantly over time, and the average is about 3%. The ANOVA test gives a P-value of 0.010381, which means there is a significant difference. A two-sample t-Test has been run to each of the different machine learning methods, and it turns out that the LSTM deep learning method MAPE result is significantly different to the machine learning methods; while there is no significant difference between each of the two machine learning methods.

Table 3.16 of MAPE (%) shows the error in prediction for the different types of deep learning methods, which include: LSTM, CNN+LSTM, LSTM+Attn and CNN+LSTM+Attn. In the table, 15-minutes predictions can reach an error of about 8%. Prediction error grad-

Table 3.14: CNN-LSTM-Attn model settings use the minute dataset prediction

Layers	**Settings**
Input	One week observations by default. The size (5, 130, 4) refers to five days, 130 data points a week and four features per each week (mathematically a combination of trading volume, type of day, price change, log(volume)). We use mini-batch with batch size of 100.
CNN	One-layer CNN with ReLU and max pooling. The first convolutional layer has 32 filters and each filter is 5×13 with strides of 1×1. The first max pooling layer has a filter size 3 with strides of 1.
RNN	Single-layer LSTM as cell with 512 units of hidden state. Two-layers of Dropoutlayer and Denselayer. The first Dropout layer keeps the probability at 0.8, and the first DenseLayer has units 64; The second DropoutLayer also has the keep at 0.8, and the units are the dimension of the output dataset, which is 1.
Output	If the dimension is 1, this means it will predict the return after 15 minutes; if the dimension is 8, that means the prediction is the return in 2 hours.
Note	Parameters are all initialized randomly with a uniform distribution between [-0.05, 0.05]. The Adam optimiser is applied for parameter optimisation (Kingma and Ba (2014)). MSE is used as a loss function in the early training stage, then MAPE is calculated. The learning rate is initially set as 0.005 and exponentially decays at a rate of 0.8 every two epochs.

ually increases with how long into the future it is predicting. The CNN+LSTM model has the lowest prediction errors at 15-minutes predictions, LSTM and CNN+LSTM+Attn have the same prediction error at 30-minutes prediction. The MAPE error rate is not significantly different across deep learning models. The overall lowest error rate is 9.21% in the CNN+LSTM+Attn model. Note the difference between the error rate in the consec-

Table 3.15: MAPE (%) on the 2013-2018 dataset. CNN-LSTM-Attn achieves best overall performance with lowest error rate 9.21%.

Predictions	SARIMA	**VAR**	**LR**	**RFR**	**SVR**	CNN-LSTM-Attn
15 minutes		9.0100%	10.4600%	9.4100%	9.5800%	**7.3400%**
30 minutes		13.1000%	15.0100%	13.2900%	13.6800%	**8.5200%**
45 minutes		16.5000%	18.9700%	17.0200%	17.3200%	**8.9400%**
60 minutes		19.3000%	21.6500%	19.9900%	20.5400%	**9.2100%**
75 minutes		22.0100%	27.6500%	24.0120%	25.8700%	**9.4000%**
90 minutes		26.0200%	30.1200%	28.5600%	29.6700%	**9.8400%**
105 minutes		29.8900%	36.2300%	31.1200%	34.2100%	**10.1400%**
120 minutes		34.4500%	40.1200%	35.2100%	39.3200%	**10.3400%**
overall	23.0000%	21.2850%	25.0263%	22.3270%	23.7740%	**9.2100%**

utive time gap is less than 1% in every model listed in the table. It is also worth noting that even the highest error rate in the table is 11.62%, which is significantly smaller than the ARMA family and traditional machine learning models. ANOVA has been run to test the statistically significant, the p-value is 0.4068024, and there is no significant difference between different deep learning methods.

Table **??** is a comparison between different length of input on the minute testing set. The $CNN + LSTM + Attn^1$ model is used on datasets of one day (26 data points), one week (130 data points), one month (420 data points), and three months (1260 data points). The results show that using one-week observation as an input gives the lowest MAPE error, while the use of one-day observation as input is also very low. However, the three-

Table 3.16: Comparison among different DNNs on the dataset.

Predictions	**LSTM**	CNN+LSTM	LSTM+Attn	CNN+LSTM+Attn
15 minutes	7.7900%	**7.3100%**	7.6500%	7.3400%
30 minutes	8.8400%	**8.5200%**	8.5700%	**8.5200%**
45 minutes	9.5200%	8.9900%	9.2400%	**8.9400%**
60 minutes	10.0400%	9.3000%	9.8700%	**9.2100%**
75 minutes	10.5200%	9.4600%	9.9900%	**9.4000%**
90 minutes	10.7800%	9.9100%	10.1200%	**9.8400%**
105 minutes	11.0500%	10.1900%	10.7600%	**10.1400%**
120 minutes	11.6200%	10.4500%	10.8900%	**10.3400%**
overall	10.0200%	9.2665%	9.6363%	**9.2100%**

month input gives the highest among all the results. The 15-minute prediction gives the lowest MAPE error than other time length predictions. The p-value in the ANOVA test is 0.1535, which means there is no significant difference between all the CC+LSTM+Attn method when the input dataset length is different.

Table 3.17: Comparison between different length of input on the dataset

Predictions	$CNN + LSTM + Attn^1$	$CNN + LSTM + Attn^2$	$CNN + LSTM + Attn^3$	$CNN + LSTM + Attn^4$
15 minutes	7.4100%	**7.3400%**	7.3800%	8.0100%
30 minutes	8.5700%	**8.5200%**	8.5400%	9.9800%
45 minutes	9.0100%	**8.9400%**	9.3400%	10.0100%
60 minutes	9.3100%	**9.2100%**	9.2600%	10.1100%
75 minutes	9.4900%	**9.4000%**	9.4500%	10.2300%
90 minutes	9.9200%	**9.8400%**	9.8700%	10.4700%
105 minutes	10.2000%	**10.1400%**	10.1700%	11.5200%
120 minutes	10.4100%	**10.3400%**	10.3800%	11.6700%
overall	9.2900%	**9.2100%**	9.2987%	10.2500%

1. Previous one-day observations as input (26 data points).

2. Previous one-week observations as input (130 data points).

3. Previous one-month observations as input (420 data points)
when consider each month 30 days.

4. Previous three-month observations as input (1260 data points)
when consider each month 30 days.

3.5 Discussion

3.5.1 Dataset

The result is relatively accurate given a very big dataset from year 2012 to 2017, especially as the tables of the volume/price/price change mean percentage difference shows that it is not significantly different between the Christmas and non-Christmas period and weekdays; the only differences being the Brexit and USA election periods having relatively high ups and downs. To understand how well the model works when these two periods removed, the same test was run without data fro Brexit and USA election periods, resulting in a MAPE error of 9.201%, meaning the prediction is quite accurate.

It turns out that the weekly input data points are better than daily, monthly and three-month input data points. This result has only been tested on the best model that was selected and was not tested on other models that used all the input dimensions. However, as shown in Table 3.16, nearly all deep learning model have reduced the error rate to below 10%.

This research only targets three companies: Apple, IBM, and Microsoft; and there was some limitation in the training dataset. First, these are all technology companies; and secondly, they are all headquartered in the USA. It's possible that other kinds of companies from other sectors or those who are not based in the USA could behave differently. Other factors that are not accounted for include: transaction costs, and stock splits, which causes a sudden step-change in the stock value and quantity, which a model would need to account for.

In the literature, no research has previously been done to use the Day ID (class of the day) as an input, and to correlate volume and price/price change [Kar87] [JJ88] [AP88]. When designing the model, the Day ID is set to be 1,2,3,4..., this may not have been the

optimum mapping and may have brought noise into the model. However, a similar model has been set up by Wei Bao in the stock price trading [LZW+18], and Day ID is set up this way in other kinds of experiment, for example, it has been applied to Deep Sequence Learning with Auxiliary Information for Traffic Prediction [LZW+18].

3.5.2 Novel Methods

In the LSTM+RNN+Attn model, the attention has been selected based on a paper from the literature [LPM15]; there are three methods described paper, including other methods to set attention. However, as Table 3.16 shows, the method of adding attention only causes the model to reduce the error by 0.4%. Therefore, it is not necessary to address the type of attention added to the model as it does not significantly affect the results.

In the literature, stock price prediction algorithms have been built in many different ways. For example, Wei Bao uses a technical analysis method as input dimension dataset, then applies a deep learning method on top; the resulting MAPE error is 2% [BYR17]. The inspiration of using volume, price change, the close price is from the past research, and in the literature, deep learning methods have not yet been applied as the model [CCG05] [CS00][Sun03].

3.6 Conclusion

Tables 3.10, 3.11 and 3.9 show that the minute mean in general is larger than the daily mean in volume percentage difference, close price percentage difference, and price change percentage difference during the special days like Brexit and USA presidential election. As shown in Table 3.5, in the other special days like Christmas, the volume percentage difference is more significant across all companies (IBM/Microsoft/Apple) than the close price percentage difference and price change percentage difference.

A weekdays comparison has been done in Table 3.8, it also shows that the minute means has a bigger difference than the daily mean; and most importantly, volume and price change percentage difference is more significant than the close price percentage difference. It also shows that Monday and Friday are more significant than Tuesday, Wednesday and Thursday, indicating that daily trading strategies are more likely to make a bigger profit on Monday and Friday.

In this research, deep learning methods are shown to be better than machine learning methods and ARM family methods, with machine learning methods and ARM family methods being about the same accuracy, as seen in Table 3.15. In the table, the prediction of different time intervals, including 15 minutes, 30 minutes, 45 minutes, 60 minutes, 75 minutes, 90 minutes, 105 minutes, and 120 minutes are shown. It shows that the longer the time interval is, the less accurate the stock price prediction is. In the machine learning methods, VAR has the least error rate compared to other machine learning methods, which is 21.285%. Across machine learning methods, accuracy is below 80% while the deep learning method has accuracy rates above 90%.

3.6.1 Deep Learning

In Table 3.13 15 models have been tested, and each model is the combination of the input dimensions of Day ID, Volume, Close price, log(Volume), Price change, and log(Close price). After running a comparison of the deep learning CNN+LSTM+Attn model with different input datasets, it is shown that the the input of
Days ID+Price change+log(Volume)+Volume gives the lowest prediction MAPE. This table also shows that the difference input datasets don't have a huge difference in prediction accuracy in general, with MAPE of all cases being around 11%.

Table 3.16 shows that in general, the attention combined together with CNN and RNN model works best in the deep learning model for stock prediction than the other deep learning models across most time intervals. This matches findings elsewhere in the literature [YTC+15] [XBK+15][MHG+14][VSP+17]. Though in the 15 minutes time interval, CNN+LSTM model is better than the CNN+RNN+Attention model; in the 30 minutes time interval, CNN+LSTM is the same as the CNN+RNN+Attention model with the error in 8.52%. Table 3.16 also indicates that the longer the time interval is, the less accurate the stock price prediction is. The average for price prediction error across all the time intervals is about 9.21%, which means the accuracy of the stock price prediction is above 90%.

In the deep learning model, Table 3.17 shows that the use of one-week of observations as input dataset gives the highest accuracy. The three-month observations as input has a stock price prediction accuracy slightly lower than 90%. The accuracy of one-day, one-week and one-month input datasets are all above 90% and are therefore not much different.

Chapter 4

Stock Return Prediction Before Merger and Acquisition

4.1 Introduction

Financial institutions and traders are always searching for the most accurate algorithm to decide how to trade. Much related research has been done in the literature [KCH01][EVDHM05]. To find a machine-learning based algorithm that can replace human decision making to execute a trade, an accurate and representative dataset and environment is needed to train the model.

Mergers and Acquisitions have always been very interesting to the public, so much so that financial institution have separate investment banking divisions that are set up to deal with M&A-related issues. Many examples of further study in this field appear in the literature.

In this research, an algorithm is trained to predict share price change 6 months after a merger or acquisition is announced to the public. An algorithm is trained on the time

sequence dataset for each company. At the end, an overall algorithm is trained to predict the price change within 6 months after the announcement of the acquisition.

4.2 Data Collection

Technology companies make strategic acquisitions. It happens more and more frequently in the top four Tech Unicorns (Amazon, Apple, Facebook, and Google). A dataset has been collected that includes four big deals: Amazon acquired Whole Foods on the 16th June 2017 at a valuation of 13.7 billion dollars; Google acquired the HTC Pixel Phone division on the 21st September 2017 at an valuation of 1.1 billion dollars; Apple acquired Beats Electronics on the 28th May 2014 at a valuation of 3 billion dollars; and Facebook acquired WhatsApp on the 19th February 2014 at a valuation of 19 billion dollars.

As can be found on the website Crunchbase (see first Appendix). There were 228 acquisitions by Google, 75 acquisitions by Facebook, 101 acquisitions by Apple, and 79 acquisitions by Amazon. It can be noted that many of the acquisitions has no valuations recorded on Crunchbase; of all the companies acquired by Apple, Amazon, Google and Facebook, only 87 of them have recorded valuations, likely because they are notably large acquisitions and so values have been widely publicized. These 87 deals are collected as a dataset for analysis as we have the valuations for them.

As some of the acquired companies are privately owned, they are not listed on the exchange and don't have public stock price data available. Therefore, the four unicorns: Facebook, Google, Apple and Amazon stock prices are collected instead. Different time lengths of data are collected: one month before and after announcing the news on a daily and minute basis; three months before and after; and six months before and after.

4.3 Data Analysis

As can seen in the one-month daily Figures A.35, A.31, A.33, and A.29; three-months daily Figures A.36, A.32, A.34, and A.30; and six-months daily Figures A.37, A.39, A.40, and A.38; the one- and three-months daily figures seem to not show the whole pattern, therefore, the six-months daily or minute dataset will be used to make the analysis.

Further exploring the patterns in the one-month daily dataset, it can be seen that close price gradually increased after the acquisition news was announced, however, Amazon is the only case where daily close price went up and down multiple times while Google, Facebook and Apple stock close prices all gradually rose the whole month after the announcement of the news. Figures can be seen in A.35b, A.29b, A.33b, and A.31b. Most of the high volume in the one-month daily dataset is between 30 and 50 days for each company. Note that the acquisition announcement is around days 40 and 45, this pattern is very obvious in Google, Facebook and Amazon, as can be seen in Figures A.29a, A.33a, and A.31a. In the case of Apple (Figure A.35a), it can be seen that the lowest volume is around days 40 to 50, and is higher than the lowest volume for any other period. As seen in Figures A.35c, A.29c, A.33c, and A.31c, the lowest price change during the 40 to 50 days is higher than the lowest price change for other periods.

In the three-months daily dataset, the stock volume of Facebook, Google, Apple, and Amazon is high around the 50th, 120th, 125th and 200th day, as can be seen in Figures A.30a, A.34a, A.32a, and A.36a. The close price shows a uptrend in Figures A.30b, A.34b, A.32b, and A.36b. Price change shows the low point is around the 175th day, and high point around 75th day, see Figures A.30c, A.34c, A.32c, and A.36c.

In the six-months daily dataset, the close price in Figures A.38b, A.40b, A.39b, and A.37b show that the close price rises until the 300th day, and then remains constant for a while until the 400th day, before continuing to rise again until the 500th day. In the price

change figures, as seen in Figures A.38c, A.40c, A.39c, and A.37c, it shows there is a noticeably high low price change happening between the 300th and 400th days. There is a significantly high volume between the 300th and 400th day for Apple and Facebook Figures in A.39a and A.37a.

One-month, three-months and six-months minute datasets are plotted in Figures A.41, A.42, and A.43. From the volume Figures A.41a, A.42a, and A.43a, it can be shown that there is abnormal volume 3 months before and 1 month after the acquisition. Figure A.41b best shows the close price trend, while, due to having too many data points, Figures A.42b and A.43b don't as clearly show this trend. The same can be found for the price change with detailed changes visible in Figure A.41c, showing how the dataset changes most around the acquisition announcement date.

From all the observations above, the six-months daily data will be used in the machine learning model, and three time-scale datasets: one-month, three-months, and 6-months minute datasets will be used to test the deep learning model, see Table 4.7.

In the machine learning model, the first model is trained and tested on each company's data. Note that each company has tens of acquisitions, those acquisitions can be listed chronologically and if assumed to be independent, the dataset can be split into a training dataset and a testing dataset using different subsets of these acquisition events. The second model combines the data for all four companies, and lists all acquisitions by these four companies that have a valuation on Crunchbase. The combined dataset is similarly split for training and testing datasets. The second model is a deep learning model, and will be trained with the output being price change, while the input dataset can be different combinations of: volume, price change, close price, and/or log(Volume).

4.3.1 Machine Learning

The six-month daily dataset can be used to build an algorithm to make stock price change or close price prediction. Table 4.1, 4.2, 4.4 and 4.3 show MAPE errors listed for: support vector machine, linear regression, and random forest model. The table also has the comparison for different time scales of one-month, three-months and six-months, moreover, the table also compares the MAPE error to different output datasets for volume, close price and price change. In the input dataset, variables are not correlated, which is a useful property. In general, Google has lower MAPE errors in each category compared to the other three companies. As in the previous chapter, the Random Forest (RF) has the lowest MAPE rate compared to other machine learning methods, and the six-month time scale gives a lower MAPE error compared to one-month and three-month times scales.

The model is first tested using 101 Apple acquisitions, shown in Table 4.1. In this Table, one-month and three-month time scales have the lowest MAPE error in the close price. The second lowest is price change in the one-month time scale, but volume in the three-month time scale. In the six-month time scale, price change is less than the volume, volume is less than the close price. The lowest MAPE error in Apple is 24.14%.

The Amazon acquisitions include 79 deals, shown in Table 4.2. In one-month, three-month and six-month trained machine learning methods (SVR, LR,RF), they all follow the trends that the MAPE error in the price change is less than the MAPE error in the close price, and the close price MAPE error is less than the MAPE error in volume. Also the six-month case is less than the three-month case, and three-month case is less than the one-month case. Though it happens that sometimes the six-month volume MAPE error is larger than the three-month price change and close price in the Support Vector Regression (SVR) model.

Google has 228 acquisitions, shown in Table 4.3. Random Forest applied to this data has

the lowest MAPE error at 23.3% in the six-month scale for volume. This is the lowest value out of all companies. In the one-month case, all machine models (SVR, LR, RF) have MAPE errors for volume that are larger than the MAPE errors for close price error and close price. Similarly in the three-month case, the MAPE errors gradually decrease from volume, close price, then price change. However, in the six-month case, volume MAPE errors are lowest in RF and SVR, but LR has the largest error for close price and price change.

Facebook has 75 acquisitions, shown in Table 4.4. The overall MAPE error in Facebook is higher than the other three companies. In this table, the highest MAPE error is about 40% in the one-month LR volume. In the one-month category, the MAPE error decreases over volume, close price, and price change in all machine learning models. In the three-month and six-month cases, the close price has higher MAPE error than the volume, and the volume has higher MAPE error than the price change.

The final table shows the 87 acquisitions that include all the companies that have acquired a company with a valuation publicly listed on Crunchbase, see Table 4.5. In this table, all the MAPE errors are generally lower than when data is trained for individual companies. The lowest MAPE error is in the six-month price change category for Random Forest.

Table 4.1: The MAPE error during Apple acquisitions

			SVR	LR	RFR
Apple(101 acquisitions)	1 month	Volume	37.5100%	34.2200%	32.1100%
		Close price	31.8100%	33.2200%	30.0200%
		Price change	35.1200%	37.3100%	31.3200%
	3 months	Volume	30.5100%	32.5200%	28.3100%
		Close price	27.8200%	31.2100%	26.0300%
		Price change	31.1100%	32.9100%	29.0300%
	6 months	Volume	27.2100%	28.6200%	26.0300%
		Close price	28.3100%	30.1200%	27.2100%
		Price change	25.9100%	26.1200%	24.1400%

Table 4.2: The MAPE error during Amazon acquisitions

			SVR	LR	RFR
Amazon(79 acquisitions)	1 month	Volume	34.5000%	36.2000%	32.4000%
		Close price	32.7000%	34.5000%	30.5000%
		Price change	31.6000%	35.3100%	30.1200%
	3 months	Volume	28.5000%	30.3100%	27.0200%
		Close price	27.0300%	31.2000%	26.1200%
		Price change	26.3100%	30.7200%	25.0400%
	6 months	Volume	27.2100%	28.6200%	24.3100%
		Close price	26.3200%	28.1200%	25.2100%
		Price change	25.0300%	26.5100%	24.4100%

Table 4.3: The MAPE error during Google acquisitions

			SVR	LR	RFR
Google(228 acquisitions)	1 month	Volume	33.5000%	35.7000%	31.6000%
		Close price	34.7000%	35.5000%	32.6000%
		Price change	31.6000%	34.7000%	30.3000%
	3 months	Volume	27.8000%	30.1000%	27.1000%
		Close price	26.7000%	30.4000%	25.7000%
		Price change	25.8000%	30.0000%	25.1000%
	6 months	Volume	24.2000%	28.0000%	23.3000%
		Close price	25.3000%	27.6000%	24.5000%
		Price change	24.6000%	26.7000%	24.0000%

Table 4.4: The MAPE error during Facebook acquisitions

			SVR	LR	RFR
Facebook(75 acquisitions)	1 month	Volume	38.1000%	39.2000%	37.4000%
		Close price	36.2000%	37.1000%	35.1000%
		Price change	35.6000%	36.9000%	33.6000%
	3 months	Volume	32.9000%	34.3000%	31.3000%
		Close price	33.1000%	34.2000%	31.7000%
		Price change	31.3000%	32.7000%	29.7000%
	6 months	Volume	27.2000%	29.4000%	26.4000%
		Close price	28.3000%	29.4000%	26.7000%
		Price change	27.1000%	28.6000%	26.3400%

Table 4.5: The MAPE error of all acquisitions that has a valuation on the Crunchbase

			SVR	LR	RFR
The 87 acquisitions	1 month	Volume	31.1000%	32.2000%	30.4000%
		Close price	32.2000%	34.0000%	31.3000%
		Price change	30.6000%	31.9000%	30.0000%
	3 months	Volume	28.9000%	29.3000%	28.0000%
		Close price	29.1000%	30.2000%	28.6000%
		Price change	27.0000%	28.1000%	26.7000%
	6 months	Volume	26.2000%	27.4000%	25.7000%
		Close price	25.3000%	26.9000%	25.0000%
		Price change	24.1000%	25.6000%	23.4000%

4.3.2 Deep Learning

Given that the combined 87 acquisition dataset has the lowest MAPE error at every time scale (one-month, three-month and six-month) compare to datasets for individual companies (Facebook, Google, Apple, Amazon), the combined six-month minute dataset has been used to test a deep learning model. The model is a combination of Convolution, LSTM, and Attention. In this model, the six-month minute dataset before the acquisition announcement is the training dataset and the first three-month minute dataset after the announcement is the validation dataset, and the second three-month minute dataset is the testing dataset. The model has 15 different combinations of the input variables, and the MAPE error has been calculated for each input dimensions and shown in Table 4.6. In this Table, Days ID+price change+Close price+Volume+Log(Volume) gives the lowest MAPE error. Though Volume and Log(Volume) are correlated, as there are not that many input variables, it is has been decided that it would be fine to use both of them. As seen in the table, there is not much difference in the prediction MAPE across all the different input. There is no statistically significant difference between the MAPE for the

difference deep learning methods through one-sample t-test.

Table 4.6: CNN-LSTM-Attn acquisition model using the minute dataset prediction

Model	Predictions MAPE
Days ID+Volume	5.7000%
Days ID+Close price	6.0000%
Days ID+Close price+Volume	6.9800%
Days ID+Close price+log(Volume)	5.4200%
Days ID+Close price+log(Volume)+ Volume	6.5100%
Days ID+Price change	6.0900%
Days ID+Price change+Volume	5.6000%
Days ID+Price change+log(Volume)	6.4000%
Days ID+Price change+log(Volume)+Volume	6.1000%
Days ID+log(Close price)	5.9000%
Days ID+log(Close price)+Volume	5.9000%
Days ID+log(Close price)+log(Volume)	6.1000%
Days ID+log(Close price)+Volume+log(Volume)	6.4300%
Days ID+price change+Close price+Volume	6.2000%
Days ID+price change+Close price+Volume+log(Volume)	5.1000%

Though the machine learning model shows that the six-month daily dataset gives a better prediction than the three-month and one-month daily input dataset, however, as the comparison of the daily and minute dataset don't necessary show the same amount of the trend's details, it is also worth testing the one-month, three-month and six-month minute datasets with the deep learning model. This can be seen in Table 4.7.

Table 4.7: CNN-LSTM-Attn acquisition model monthly MAPE comparison

CNN-LSTM-Attn Model	**1 month**	**3 months**	**6 months**
Predictions MAPE	8.71%	7.03%	5.12%

The settings for the deep learning model are shown in Table 4.8. The CNN-LSTM-Attn model has an input size of $(1, 26, 5)$ which represents one day of trading: as trading starts in the morning at 9 : 30 and ends in the afternoon at 16 : 30, with 15-minute samples, there are 26 sample points per day; and each sample point has 5 input dimension: the type of day, price, price change, volume and log(volume). With this setting, the initial conditions are set according to the Adam optimiser based on the paper by Kingma and Ba in 2014. In the setting, the CNN layer has 16 filters because the dataset is significant smaller during the Merger and Acquisition than the stock prediction case; the filter size is 5×13 as 13 is a factor of 26, and it's multiplied by 5, the number of input dimensions. In the RNN, 256 units of hidden state comes from the classic setting. The defaults are also used for the probability of the Dropout layer at 0.8 and Dense layer with 32 units.

The above setting and results are all based on the prediction of the next 15 minutes; however, the prediction ranges can change based on the output dimension, for example, the dimension can be 2 to predict the next 30 minute's results or 3 for the next 45 minutes. Table 4.9 shows the prediction from the next 15 minutes up to the next 120 minutes at 15 minute intervals. In this table, different types of deep learning models are compared: LSTM, CNN+LSTM, LSTM+Atten and CNN+LSTM+Attn. As shown in the table, at the 15-minute time scale, the lowest MAPE error is 5.08% for the CNN+LSTM+Attn

Table 4.8: CNN-LSTM-Attn acquisition model settings use the minute dataset prediction

Layers	**Settings**
Input	One day observations by default. The size (1, 26, 5) refers to one day, 26 data points a day and 5 features per each day point (mathematically a combination of trading volume , type of day, price change, close price, log(volume)). We use mini-batch with batch size 20.
CNN	One-layer CNN with ReLU and max pooling. The first convolutional layer has 16 filters and each filter is 5×13 with strides 1×1. The first max pooling layer has a filter size 3 with strides 1.
RNN	Single-layer LSTM as a cell with 256 units of the hidden state. Two-layers of Dropout layer and Dense layer. The first Dropout layer has keep probability of 0.8, and the first Dense layer has 32 units; The second Dropout layer also has the keep at 0.8, and the units are the dimension of the output dataset, which is 1.
Note	Parameters are all initialized randomly by uniform distribution between [-0.05, 0.05]. An Adam optimiser is applied for parameter optimisation (Kingma and Ba (2014)). MSE is used as a loss function in the early training stage, followed by MAPE. The learning rate is initially set as 0.005 and exponentially decays with a rate of 0.8 every two epochs.

deep learning model. The MAPE error increases with the time scale. The average of the MAPE error for CNN+LSTM+Attn is 7.75%. The p-value is 0.296, and there is no significant difference found with the ANOVA test.

Table 4.9: Comparison among different DNNs on the dataset.

Predictions	**LSTM**	**CNN+LSTM**	**LSTM+Attn**	**CNN+LSTM+Attn**
15 minutes	6.7900%	6.3400%	**5.0800%**	5.100%
30 minutes	7.5000%	7.2000%	6.6000%	**6.5000%**
45 minutes	8.2000%	7.6300%	7.2000%	**7.0100%**
60 minutes	9.0400%	8.2000%	7.9300%	**7.7100%**
75 minutes	9.5000%	8.9000%	8.3400%	**8.0900%**
90 minutes	10.0800%	9.0100%	9.0000%	**8.9000%**
105 minutes	10.5000%	9.6000%	9.5000%	**9.2000%**
120 minutes	11.0200%	10.0100%	9.8900%	**9.4800%**
overall	9.0790%	8.3610%	8.0300%	**7.7500%**

Table 4.9 shows that CNN+LSTM+Attn is the best model among all the deep learning models. Table 4.10, shows the MAPE error for different input dataset points. For example, $CNN + LSTM + Attn^1$ means the input data is 26 data points, which represents the previous day of observations. The table shows that there is not much difference when the input dataset is one-day, one-week or one-month. Also, this table shows that the acquisition model MAPE error is generally less than the model that predicts the stock price. In the test, ANOVA has a p-value 0.0425, which means there is a significant difference in result for different lengths of the input dataset, which means different time lengths make a difference. Note that when a two-sample t-test is run between each pair of methods, it shows that the p-value is 0.0209 between the previous one-day observation as input and one-month observation as input, which means there is a significant difference between these two. But the one-day versus one-week input has a p-value of 0.362; the one-day versus one-month input has p-value 0.08, which means none of these two is

significantly different.

Table 4.10: Comparison between different length of input on the dataset

Predictions	$CNN + LSTM + Attn^1$	$CNN + LSTM + Attn^2$	$CNN + LSTM + Attn^3$
15 minutes	**5.1000%**	6.3400%	7.8000%
30 minutes	**6.5000%**	7.2000%	8.4000%
45 minutes	**7.0100%**	7.9400%	9.3000%
60 minutes	**7.7100%**	8.1000%	9.6000%
75 minutes	**8.0900%**	8.9000%	9.7000%
90 minutes	**8.9000%**	9.2000%	9.9800%
105 minutes	**9.2000%**	9.6000%	10.1000%
120 minutes	**9.4800%**	9.8700%	10.3000%
overall	**7.7500%**	8.3900%	9.4000%

1. Previous one-day observations as input (26 data points).
2. Previous one-week observations as input (130 data points).
3. Previous one-month observations as input (420 data points)
when consider each month 30 days.

4.3.3 Discussion

The inspiration for researching the relationship between price and volume during a merger and acquisition period comes from past research done on option volume and stock price. In 2009, Jerayr Halebian, Cynthia E Devers published an article that developed a framework to empirically organise and review recent findings during the acquisition period [HDM+09].

In the research that has been done by Globe and Devra, there was a strong positive correlation between the level of merger and acquisition activity and the volume of trading [GW87]. Therefore, an empirical framework has been explored that uses machine learning and deep learning methods in the second chapter of this research.

Machine Learning

The methods used in this chapter are some of the most basic machine learning methods; other machine learning methods can be applied as well. For example, Schumaker has used the text analysis in the stock market prediction [SC09].

The MAPE error result in this model is larger than the MAPE error in the model from the previous chapter, as seen in Tables 3.15, 4.5, 4.4, 4.1, 4.2, and 4.3. The reasons behind this are because the deal size could vary over a wide range, which means it has a very high variance and volatility. It is also not general enough in terms of the time scale and the trading environment to be able to train accurately; many factors can influence the stock price. To get a more accurate model, a large dataset covering these variables is needed. Conversely, the low accuracy of the "all acquisitions" case is likely because of the high variation in the types of acquisitions each company makes, so a model trained on all the data is less able to generalise. This idea has been applied in research done by Arnold M. Rose in 1951 factoring in even rumours in the stock market [Ros51].

In general, in order to reduce the bias of the model, the machine learning model that is used to test each company case could shuffle the training and testing dataset, and check the accuracy of the prediction. Similar shuffle methods have been used in the literature, for example, on research using Twitter to predict stock price [Zha13], and on S&P 500 index prediction by Li Xiang Shen and Han Tong [SL04].

One challenge here is that some acquisitions happen close to each other, meaning the

effect on the stock price of one acquisition may coincide with another acquisition, causing the data to be hard to train with, or effects of one acquisition masking the other, as can be seen in Figure 4.1.

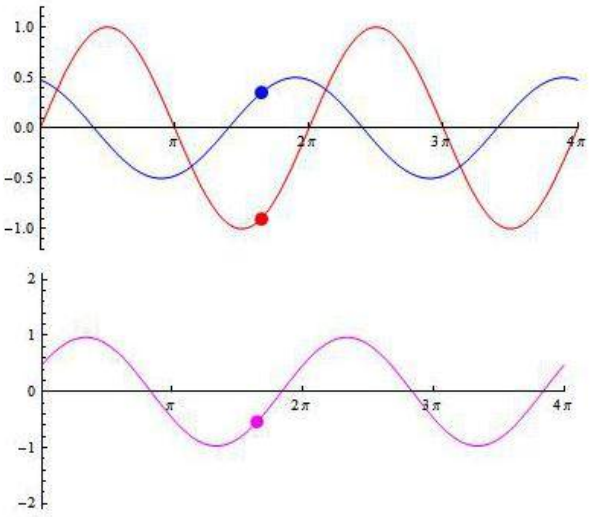


Figure 4.1: The summation of the two curve

Discussion

The research outlined in this section is novel in the literature, likely because the comparison between M&A cases needs the model, the control, and the input variables to be very well controlled, and data sources for that has not available until very recently.

Deep Learning

Though the deep learning methods are only done to the combined all-acquisitions model, and not tailored to each individual company, 7.75% is a fairly good result compared with the previous chapter in stock return prediction.

The reason behind applying a deep learning model on the all-acquisitions case rather than each individual company is that the all-acquisition machine learning model has a lower MAPE error than that for the individual companies. However, it is not always the case that a combined dataset is better than individual ones. Think of the fundamentals of model building; it is common to use massive datasets to find the function that matches the x to y (i.e.$f(x) = y$). In the literature, batch size has been split for a job shop schedule problem in an MRP environment resulted in a smaller MAPE [JPL99]; Sergey Loffe and Christian Szegedy from Google have improved model accuracy through normalisation via mini-batches [IS15]; and Schaap, Marcel G. and Leij, Feike J showed database factors that influence the accuracy of the result in 1998 [SL98].

Therefore additional experiments have been done, and the result can be found in Table 4.11. The ANOVA has been run and achieves a p-value of 0.646; this means there is no significant difference between these tech firms MAPE predictions.

Table 4.11: Comparison among the CNN+RNN+Attn model on each company.

Predictions	**Apple**	**Facebook**	**Amazon**	**Google**
15 minutes	5.1200%	5.9800%	4.9700%	**4.7800%**
30 minutes	5.9000%	7.0100%	5.6100%	**5.5000%**
45 minutes	6.7800%	7.4500%	6.2300%	**6.1200%**
60 minutes	7.2400%	8.0100%	7.1200%	**6.9800%**
75 minutes	7.5600%	8.4800%	7.9800%	**7.6200%**
90 minutes	8.7800%	8.9600%	8.2300%	**8.3400%**
105 minutes	9.4500%	9.6700%	9.3400%	**9.0100%**
120 minutes	9.9100%	10.0200%	9.8900%	**9.2300%**
overall	7.6025%	8.1975%	7.4200%	**7.1975%**

4.3.4 Conclusion

As with the previous chapter, this chapter demonstrates that the deep learning model is better than the machine learning model, and the deep learning model MAPE error is about a quarter of the machine learning model. According to Table 4.11 and Table 4.10, it shows that though in the machine learning model all-acquisitions case is better than the case for each company, however, in the deep learning model, Google is the lowest MAPE error at about 7.2%. The second and third best are Amazon and Apple, with 7.42% and 7.6% respectively, all still better than the all-acquisition case. But Facebook has the worst MAPE error at 8.2%. This result also shows that machine learning and deep learning model don't necessarily end up with the same results.

One explanation could be that the training and testing datasets are smaller in this case, but the variance in this dataset just happens to be smaller than the other dataset.

It also turns out that one-day input dataset is better than the one-week input dataset, and one-week input dataset is better than the one-month input dataset, as seen in Table 4.10. Quantitatively speaking, it shows that machine learning methods differ a great deal from all the deep learning methods, but within the different deep learning methods, there is little significant difference.

Chapter 5

Electroencephalography(EEG) for Advertisement Impact Assessment

In the literature, AM Joshi and DM Hanssens have shown that advertising raises investor expectations about the product's financial performance [JH09]. TJ Chemmanur has also published a paper on the increases in stock price during the advertising year [CY19]. These pieces of research all show that advertisements can help stock prices rise. R Ohme has conducted research into the brain's frontal cortex activation in reaction to TV advertisements [ORWC10]. It is, therefore, an exciting prospect to investigate the use of machine learning method to find unbiased general EEG methods for assessing advertising impact and design a method that can reliably help build advertising that is able to boost the stock price.

Data previously collected by Zhejiang University drives this research, and thus the choice to use EEG rather than the other methods for assessing advertising impact (e.g. eye-tracking) was chosen. Experimental design, ethics review, and data collection were all conducted by Zhejiang University. The machine-learning and data analysis was done on data produced by the experiment were, however, conducted by the author of this thesis

and is the subject of this chapter.

5.1 Introduction

Advertising plays a critical role in marketing. Every year, companies allocate a significant proportion of marketing budget to attempt to quantify the impact of their advertising, particularly video advertising on TV and on the web, which receives wide viewership [Bra14] [Joh15]. However, current methodologies, including both direct observation (questionnaires and focus groups prior to starting of the advertising campaign), and indirect (trends in sales or consumer interest during and after a campaign), tend to have practical or experimental challenges that reduce the effectiveness of the assessment [Gol90] [ES98] [Duc96] [OS11] [LR09].

Direct observation methods include questionnaires and focus groups, where yet to be released advertisements (or multiple versions of an advertisement) are shown to a select group of viewers selected to be representative of the advertisement's intended audience. The viewers answer questions and provide feedback during a survey or engage in discussion with the organisers; then the results or discussion are analysed by the advertising team to try to assess how well the advertisement fulfils the criteria of their campaign [GKQ07].

However, direct approach methods are subject to the same kinds of challenges as other experimental psychology approaches: experimental biases are introduced due to the experimental environment (typically an office room with multiple participants) being different to how a viewer would normally view the advert (often in the comfort of one's own home), leading to a different state of mind of the viewer. There is a tendency for respondents to feel obliged to give more favourable reviews under experimental conditions of being under observation than they would typically do. The way the experiment is carried out in groups can also result in peer-pressure and group-dynamics altering the responses of

individuals, leading to participants reporting attitudes or preferences that may not truly represent their own when they are in private [DPGA02] [SL09] [MGB04]. The analysis of survey feedback or discussion too can introduce experimenter-bias where the opinions of the experimenters impact the evaluation of the results; and the size and cost of conducting the study and analysing the results is often cost-prohibitive past certain scales (and when considering TV advertising for national release, the size of these focus studies are often a tiny proportion of eventual viewership, resulting in uncertainty of the statistical relevance of the results)

Indirect observation methods of assessing advertising impact involve inferring advertising impact based on the result of an advertising campaign [SMS11]. These methods avoid the inaccuracies imposed by artificial experimental conditions of direct observation methods but suffer from their own challenges. By looking only at the results and effects of the advertisement campaign, and because it is only possible to look at aggregated effects (such as the impact on sales or customer interest in a product), it is often difficult to identify if, why, and how a particular aspect of the advertisement causes an impact. Furthermore, due to the life-cycle of an advertising campaign, it is only possible to infer impact during or after the launch of an advertising campaign, meaning in many cases, much or all of the cost of the campaign having been sunk, limiting adaptability in the event of lacklustre response to the campaign [SK04] [KWW92] [GMK98].

In recent decades, research in neuroscience has brought new understanding and tools that can change how the impact of advertising can be assessed and has formed the new field of neuromarketing, in which recent neuroscience and experimental psychology tools and understanding are being applied to marketing. One key hypothesis in neuromarketing is that consumers' decisions can be driven more by emotion than by a careful comparison of product benefits or differentiators. Therefore, measuring an advertisement's emotional impact on an individual could correlate well with the impact of the advertisement.

In neuromarketing-based advertisement impact assessment, biometrics are gathered from individuals participation in the study, and these biometrics are used to assess impact, rather than voluntarily self-reported information from surveys or discussion. The recorded data includes biometrics such as eye-tracking, facial coding, Galvanic skin response and electrodermal activity, and EEG. EEG is a non-invasive electrophysiological recording of brain activity, using electrodes placed along the scalp. EEG has multiple advantages over other methods of measuring brain activity in that it has a high temporal resolution, is non-invasive, quick to instrument and tolerant to subject movement, and low cost with the use of single electrode equipment. In 2010, Murugappan found in a study of human-computer interaction, a correlation between a user's emotion and EEG, providing useful information in understanding a user's reaction to advertisements [MRS+10]. Therefore, an increasing amount of research into neuromarketing has turned to EEG as a key sensor in measuring emotion. Lucchiari and Pravettoni observed that EEG signals with a frequency of 16-31Hz (i.e. Beta wave) could be modulated by the experience of pleasure when a consumer was presented with a favourite brand [LP12]. In 2016, Wang, Chang and Chuang found that a narratives structure in video commercials induced higher EEG signals with a frequency band of 4-7 Hz (i.e. higher Theta) power of the left frontal region resulting in a higher preference for branded products [WCC16].

In psychology research, a person's emotion can be quantified through self-reported measures such as liking (valence) and excitability (awaken) [PD06] [SVMN06]. Questionnaires can be used to gather this type of information. According to the AIDA model, four quantified metrics are used to characterise the experience for a consumer watching an advertisement: attention, interest, desire, and action [SJ25].

In the literature, Support Vector Machine has been widely used on EEG data; research on EEG based emotion recognition using frequency domain features and Support Vector Machine (SVM) was done by X. Y. Wang et al [WNL11]. Research on EEG-based emotion

recognition in music listening using Support Vector Machine(SVM) was done by Y. P. Lin et al [LWW+09]. Though other regression in binary results can also be used to build this model, in the literature, Support Vector Machine is the most widely used method in this field. Also, Support Vector Machine is suitable because of the sparse dataset the experiment uses. The technique used will build a prediction model based on several different brainwaves, which include frequency band less than 4Hz (i.e. Delta), frequency band between 4-7Hz (i.e. Theta), frequency band 8-15Hz (i.e. Alpha), frequency band 16-31Hz (i.e. Beta) and frequency bigger than 32Hz (i.e. Gamma).

In this thesis, we tested the hypothesis that it is possible to use low-cost EEG equipment to collect brainwaves of subjects viewing advertisement, and to apply the latest methods from neuromarketing, and machine learning as a more accurate method of assessing advertisement impact and the likelihood of a person purchasing the advertised product than the current state-of-the-art.

5.2 Methodology

This section will describe the method used to collect data from a single-electrode wearable EEG device, self-reported measures for the impact of an advertisement, and train a predictive model against the data using SVM [PD06] [SVMN06] [CdBV+14]. The device names and prices/costs are listed in table 5.1. The NeuroCARE1.0 has been used in the experiment. This device is reliable based on its report [ACL+08].

5.2.1 Data Collection

In the experiment, thirty right-handed male participants aged 20-35 from the University participated in this experiment as paid volunteers. Thirty participants were in the ex-

Device type/Name	**Price/Cost**
Brain Products ActiCHamp (32-160 channels)	$40000
ANT Neuro eego rt / eego sports (64 channels)	$25000+
mBrainTrain (24 channel)	$1000-$25000
OpenBCI (8-16 channels)	$99-$1000
NeuroCAR1.0	$80

Table 5.1: Price/cost of different EEG devices

periment; each of them is given 4-5 advertisements, to create a sample size of 450, big enough to carry out statistical analysis. The experiment was carried out in China, and the participants are bi-lingual in Chinese and English speaker. The participants had normal or corrected vision without any histories of neurological/mental diseases. The purpose of choosing bi-lingual participants is to avoid language-barrier caused by the advertisement's language content being in Chinese or English. We selected male participants to avoid biases from the gender-specificity of the advertised products. Many products are specifically targeted at a particular gender, for example, male clothing and female clothing are targeted differently; as are hair and cosmetic products. We constructed a database of 220 TV advertisements from four gender-neutral or male-targeted products: cars (55 ads), digital products (55 ads), clothing (55 ads) and food (55 ads), which were randomly selected from Youku.com (one of the largest online media websites). Each advertisement was 15-20 seconds in length. The video resolution and audio volume of each video were normalised to the same level using professional video and audio editing software. In each test, 4-5 advertisements were randomly selected from the 220 TV advertisements database. The shortcomings and future research because of this design will be discussed in section 5.4.

Before the experiment, the volunteers received detailed instructions on all the tasks they

would perform. Each participant was fitted with a single-electrode EEG headset by an experimenter, was seated comfortably in a lab room at 1.20 meters from a 19-inch PC monitor, and shown five advertisements randomly selected from every genre for a total of 20 advertisements. An E-prime system was used to control the presentation of the stimuli.

The experiment consisted of 4 blocks, each containing 20 trials. During every trial, the volunteers were presented with the advertising for about 15-20 seconds. The advertisement was followed by an evaluation questionnaire, including the willingness of the participant to purchase the advertised product (yes or no), and whether they like an advertisement (7-point Like scale).

Each volunteer performed two practice trials before the start of the formal experiment. The frontal EEG was recorded with the single-channel dry electrode-device and system (NeuroCAR1.0, Neuromanagement Lab, Zhejiang University, China). The integrated chip of the device was the ThinkGear (NeuroSky, Wuxi, China). The sampling rate of the device for gathering EEG signals was 512 HZ and the data saved into a computer. Four trials' data were rejected due to voltage abnormality, and in total, 450 sets of brainwave data were used in our study.

5.2.2 Questionnaire Collection

The participants were asked to fill out a questionnaire with 22 questions asking about different aspects (objective and subjective) of the advertisement, or their experience. The questionnaire is listed in Table 5.2. The questionnaires are chosen to record different aspects of the advertising can impact the viewer's response to the advertising, which may indicate the likelihood of the viewer wanting to purchase the product. The aspects are chosen from several studies in the literature that each focus on one of two aspects of

an advertisement's impact. M. Vaismoradi, Kimberly and Klaus show that the audio and visual fidelity of the advertisement (Y2 and Y15 of the questionnaire) had an impact [VTB13]. The content impact is covered by questions Y1, Y5 and Y14. The brand quality is covered in (Y6, Y7 and Y8). The overall feeling of the advertisement (Y3 and Y4) was found to make an impact on a customer's decision to purchase or not [Dah02] [NSAH12]. The conscious decision reported by customers on whether they would make the purchase or not (Y9 and Y10). Padgett and Douglas Allen showed that advertising memorability also plays a role in purchase power (Y11 and Y12) [PA97]. Other impacts include: Celebrity endorsement (Y18)[SSS13] [BN$^+$13]; the feature of children or cartoons (Y20 and Y21) [BSW72] [FSR$^+$91]; the language (Y13) [NB08]; the narrative style and story-telling (Y16, Y17, Y22) [McQ02] [PM02]; and sexual-appeal (Y19) is widely established to have an impact on advertisement [SBB90] [WSN15].

Eleven of the questions had binary answers of 0 or 1; another eleven were ranked answers from 1 to 7. The questions are listed in each column as output data.

5.2.3 Modeling

Raw EEG data is first augmented into frequency domain EEG signal before creating a larger dataset necessary for analysis. In this thesis, the frequency domain EEG signal is listed in Table 5.3. Further details of input data X is listed in Section 5.2.3

The output data is labeled as Y. Each row of output data Y has 23 dimensions, and each dimension represents a feature, and all features are independent. $Y_{450\times22}$ can be written as $Y_{450\times22} = [Y_1, Y_2, ..., Y_{22}]$ where Y_i represents each column vector of Y, and $i = 1, 2, ..., 22$, see Table 5.2.

The purpose of this research is to use SVM to train a machine learning model that can find the map f, where $Y = f(X)$, the reason to use Support Vector Machine(SVM) is

	Output feature
$Y_1(1-7)$	Content Quality
$Y_2(1-7)$	Image Quality
$Y_3(1-7)$	Excitement
$Y_4(1-7)$	Attractiveness
$Y_5(1-7)$	Easiness for understanding
$Y_6(1-7)$	Clearness of the brand
$Y_7(1-7)$	Brand awareness
$Y_8(1-7)$	Familiarness of the brand
$Y_9(1-7)$	Willingness to buy
$Y_{10}(1-7)$	Intention to further learn the product
$Y_{11}(1-7)$	likeliness of memorize the ads content the next day morning
$Y_{12}(0/1)$	If the brand of the product can be memorized the next day
$Y_{13}(0/1)$	Chinese language or not
$Y_{14}(0/1)$	If it is moving
$Y_{15}(0/1)$	If it has significant vision and sound impact
$Y_{16}(0/1)$	If it is interesting
$Y_{17}(0/1)$	If it surprises you
$Y_{18}(0/1)$	Whether it has celebrities
$Y_{19}(0/1)$	If it is sexy
$Y_{20}(0/1)$	If it has children
$Y_{21}(0/1)$	If it has cartoon
$Y_{22}(0/1)$	If it is a story telling ads

Table 5.2: The output dataset list

explained in section 2.2.4. Once found, any new input EEG signals can be translated into a prediction of an advertisement's impact on a consumer, and their likelihood of either a positive impression or willingness to buy the product.

The workflow of this research is shown in Figure 5.1. Due to the format of the raw EEG signal data, data extraction is applied to amend the input data into the applicable format. Feature selection is conducted on input data, and label selection is conducted on output data. Among the thirty participants, data corresponding to twenty-seven participants is used as sample data. 10% of the sample data is then separated as a testing dataset, with the remaining 90% used for the training dataset. Bootstrapping is used to augment the training dataset. The data corresponding to the remaining participant is used as an out-of-sample test. The out-of-sample test is carried out by selecting each person in the thirty participants and then averaged the thirty out-of-sample as judgment.

Support Vector Machine is applied as the machine learning method to predict the label given output data. Cross-validation and learning curve are used to check for over-fitting/under-fitting. Each component in the workflow is explained in the following sections.

Feature Extraction

Raw EEG signal is a voltage over time signal. Hence, a crucial process is needed to break the raw EEG signal into constituent frequencies (High alpha/beta/gamma/delta waves). It is crucial because the majority of the informational content in EEG signals are in the frequency domain, so breaking up the signal into constituent frequencies is most appropriate to apply SVM to classify EEG frequency domain data. According to the research that has been done by Hao Dong [DSP+18], frequency data is better for feature extraction than the raw time-domain data. Feature extraction, specifically Fast

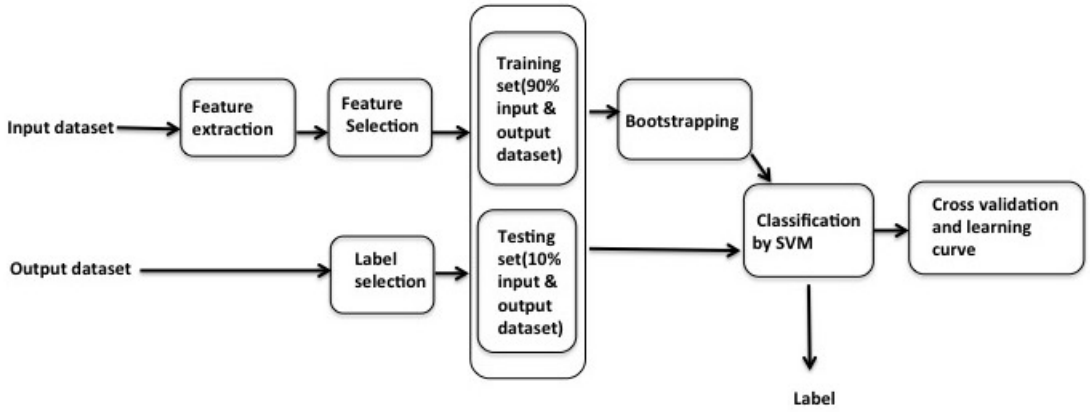


Figure 5.1: Data pre-processing workflow: the input dataset is first augmented, before both input and output datasets have features extracted and bootstrapped to increase the size of the dataset; the resulting larger dataset is split into training and test datasets, on which SVM classification is applied; the results of the classification are used to label the data and fed into the cross-validation and learning.

Fourier transform (FFT) is used to transform the raw EEG signal from time domain into frequencies domain [PDC+94] [VL92] [HJB84]. We computed the EEG frequency band power using traditional EEG frequency band definitions (Delta: 1-3Hz; theta: 47Hz, alpha I: 89Hz, alpha II: 1012Hz, beta I: 1317Hz, beta II: 1830Hz, gamma I: 3140Hz, gamma II: 4150Hz), and these are time signals for each frequency band. Because the band EEG signal is collected from the four different product types, and with the duration of each advertisement being different, data is also needed to be normalised. Timestamps are used to represent the duration, and advertisement lengths are normalised by cutting advertisements to the same length as the shortest advertisement in its category (it is 24 in cars, 13 in clothing, 28 in digital, and 16 in food).

After normalisation and extraction, the raw EEG signal is transformed into a 450×13 dimension vector, where each of the elements themselves contains a vector of data corresponding to the advertisement length. The 13 features include wavelength, time, and signal quality. X is a 13 columns, 450 rows vector, $X_{450\times13}$ can be represented by 13 columns, i.e. $X_{450\times13} = [X_1, X_2, ...X_{13}]$, each vector is X_i where $i = 1, 2..., 12, 13$, then

	Input feature	Meaning
X_1	Time Stamp	A sequence of numbers indicate the time index of raw signal
X_2	Signal Quality	Value ranges from 0 to 255
X_3	Raw	Voltage
X_4	Attention	Intensity of a user's level of mental focus/attention, value ranges from 0 to 100
X_5	Meditation	Level of a user's mental calmness/relaxation, value ranges from 0 to 100
X_6	Delta	<4 Hz
X_7	Theta	≥ 4 Hz and <8 Hz
X_8	Low Alpha	$7.5 - 9$ Hz
X_9	High Alpha	$9.5 - 12.5$ Hz
X_{10}	Low Beta	$12 - 15$ Hz
X_{11}	High Beta	$15 - 18$ Hz
X_{12}	Low Gamma	$30 - 80$ Hz
X_{13}	High Gamma	> 80 Hz

Table 5.3: The EEG band dataset list

each column vector of X is listed in Table 5.3. However, in the experiment, selected columns X are used in the analysis, they are Delta, Theta, Low Alpha, HighAlpha, Low-Beta, HighBeta, LowGamma and HighGamma, Raw, Mediation and Attention. Further selection details are in section 5.2.3.

Feature Selection And Label Selection

The map $Y = f(X)$ can be expressed as $label = f(features)$. It is implied by the equation that a different X and Y will result in different f, and different certainty profiles. Feature extraction and selection is, therefore, a very important process in machine learning.

Feature selection can be classified into three types: flat features, stream features, and structured features [CL12] [TK91] [JKP94]. In the experiment, each feature of the input data is independent, hence features in the experiment are classified as flat.

The 13 features are grouped into different combinations to test their ability to predict purchasing chance. In the end, 11 features give the best prediction in the model. These grouped-features also corresponding to the literature that is related to emotion: Delta, Theta, LowAlpha, HighAlpha, LowBeta, HighBeta, LowGamma and HighGamma [Kli99] [T$^+$02]. Attention, Meditation and Raw values (see Table 5.3 are also considered in the model because their physical meaning is related to emotion. Output data Y is the information of questionnaire answers. In this research, we are using the customer's questionnaire score to determine whether participants will strike an emotion to purchase the product. Each label Y represents a different response/measure to the advertisement; if it is 1, it meant positive feedback is linked to emotion. Hence, we label the overall emotion as 1 (positive), and each label in the output data Y is independent. Different people can be influenced by a different emotion, so we built a model that considers all emotions equally and then make the prediction. Questionnaire answers can be classified into two types: binary answers (yes or no answers) and ranked answers (values ranked between one and seven). All labels are selected and tested independently to see their significant impact on emotion. Due to the questionnaire answer types, labels are then grouped as binary answers, ranked answers and a combination of binary and ranked answers, each group's signification is also tested to the purchase possibility through emotion.

Data Augmentation Via Bootstrapping

The original dataset contains 450 samples, and is a sparse dataset, and therefore bootstrapping is needed to increase the size of the dataset [ET94] [E$^+$03]. In the experiment, a dataset of twenty-nine people is used as in sample size of 435; while the other participant

data of a sample size of 15 is used for out-of-sample testing. The 435 in sample size is too sparse to divide into a 90% training data (i.e. 391 samples) and a 10% testing data(i.e. 44 samples).

The in-sample data 391 samples are first divided into ten-folds under the condition that each fold has the same ratio of all type of answers in output data Y. Nine of the folds are used for training, while the tenth is used for testing, the ten folds algorithm is shown in Algorithm 2. Bootstrapping is used on each in-sample training dataset. In the bootstrapping process, the Gaussian distribution bootstrapping method has been applied to each column in each training data X, shown in Algorithm 1. The Gaussian process is a method to construct a parametric bootstrap approach from Bayesian non-parametric statistics, which implicitly considers the time-dependence of the data [Efr82]. Many features are different frequency bands. Therefore, Gaussian is suitable also because it can generate continuous new data. However, Gaussian is a probability distribution between 0 and 1, while the data can be numbers even bigger than 1000 and any number between 0 and 1 is too small to be the new bootstrapped number to be used in the experiment, hence in the bootstrapping process, a scalar has been calculated to multiply the probability, and the scalar is calculated as $scale = 0.5 \times (min + max)$, where the min represents the minimum number of that column, and the max represents the maximum number of that column. The new bootstrapped number is the original value plus the scaled Gaussian probability. In this way, the original data spreading will not be affected. The mathematical combination is then applied in the nine-fold training data, which is shown in Algorithm 3. The corresponding output data Y is duplicated to match the size of bootstrapped training data X in each of the bootstrapping processes.

Algorithm 1 Bootstrapping1

[1] Gaussian process bootstrap to each column **Input** : Gaussian model, input data **Xtrain Output** : edited input data **Xp**
Use Gaussian process regression bootstrap the whole **Xtrain** dataset, name as bootstrapping1 Each column in **Xtrain** as i Edit **Xtrain** as **Xp** **Xp** = **Xtrain** select the maximum element $max = max(\mathbf{Xp}_i)$ select the minimum element $min = min(\mathbf{Xp}_i)$ $scale = 0.5 \times (min + max)$ bootstrapping using Gaussian Distribution in $(-1, 1)$ $bootc = scale \times Gaussian_bootstrap(\mathbf{Xp}) + \mathbf{Xp}$ update **Xp** with all new columns $\mathbf{Xp} = bootc$

Algorithm 2 Tenfolds

[1] Ten-folds: training and testing data selection through indexes
Ten-fold the whole dataset into ten sub dataset: same ratio of ones and zeros in nine subsets, combine the rest of unselected zeros and ones into the tenth subset **Input** : input data **X**, output data **Y** **Output** : nine different training input/output dataset $\mathbf{Xtrain_i}$
Find the index of ones in the **Y** $index_ones = index_of_ones(\mathbf{Y})$ Find the index of zeros in the **Y** $index_zeroes = index_of_zeroes(\mathbf{Y})$ Calculate the ratio of ones in the length of **Y** $ratio = length(index_ones)/length(\mathbf{Y})$ After divide the **Y** into ten folds, find the number of ones in each fold
$ones = round(ratio * length(\mathbf{Y})/10)$ After divide the **Y** into ten folds, find the number of zeros in each fold $zeros = round((1 - ratio) * length/10)$ Find the left over ones after first nine folds $ones_left = length(index_ones) - 9 * ones$ Find the left over zeros after first nine folds $zeros_left = length(index_zeros) - 9 * zeros$
In the tenth fold, append the selected index of zeros and ones $\mathbf{Xfold}[10] = append(\mathbf{X}[ones_left], \mathbf{X}[zeros_left])$ $\mathbf{Yfold}[10] = append(\mathbf{Y}[ones_left], \mathbf{Y}[zeros_left])$ i in 0:9 Find the **Y** index of ones in each fold $selection_ones = index_of_ones[i * ones : (i + 1) * ones]$ Find the **Y** index of zero in each fold
$selection_zeros = index_of_zeros[i * zeros : (i + 1) * zeros]$ Find the **Y** index in each fold $selection = append(selection_ones, selection_zeros)$ Find the **Yfold** that is corresponding to the selected column index in **Y** $\mathbf{Yfold}[i] = \mathbf{Y}[selection]$ Find the **Xfold** that is corresponding to the selected column index in each **X** $\mathbf{Xfold}[i] = \mathbf{X}[selection]$
Xtrain and **Ytrain** is achieved by appending any nine of ten subsets in Xfold and Yfold $\mathbf{Xtrain} = append(\mathbf{Xfold}[C(10, 9)])$ $\mathbf{Ytrain} = append(\mathbf{Yfold}[C(10, 9)])$

Algorithm 3 Bootstrapping2

[1] Bootstrapping the training X dataset from tenfolds using mathematical combination **Input** : input data **Xtrain Output** : nine different training input dataset $\mathbf{X2_j}$ Start with the original dataset **Xtrain**, but edit the selected column by the corresponding column in algorithm1 $\mathbf{X2} = \mathbf{Xtrain}$ Each column in **X2** as j
make a copy of $\mathbf{X2}_j$ for editing $column = \mathbf{X2_j}$ r in $range(1, 12)$ generate all unique iterations of $column$ using nCr method, and then replace the corresponding index in the **Xtrain** while the rest of the columns remain the same $newdata = nCriterations(12, r, row)$ $\mathbf{X2_j} = replace_bootstrapping1(\mathbf{Xtrain}[j]_replace\mathbf{Xp}[newdata])$

5.2.4 Support Vector Machine

In the experiment, as each label in the output is independent, and are binary (ranked answers are binarised via a threshold as described below); therefore, a regression that can end up with binary results is tested, i.e. non-linear regression, logistic regression and Support Vector Regression. All three methods have been utilized and proven appropriate in the previous research in EEG emotion detection [WNL11] [BWC15] [SM01]. However, Support Vector Machine is more appropriate on this type of sparse data [WNL11]. Besides, the experiment SVM has been shown to give the highest prediction accuracy result. The number of samples is larger than the number of features in our experiment, therefore the variable is chosen to reflect so; the loss function is set to 'Squared_hinge' because it is commonly used in classification and 'Squared_hinge' is convex and smooth and matches the function $0-1$, which is suitable for our experiment. 'Squared_hinge' is mathematically written as $(1 - yf(x))^2$. The map f is linear. Therefore the kernel is zero. Regularisation is $L2$ as output data is no longer sparse, nor feature selection as features are targeted in output data [Ng04].

In order to combine both binary and ranked answers, the rank answers are binarised using thresholds according to the following rules:

- When the selected labels involve only the ranked answers, the threshold is set to be 3.5, $4(i.e. = (7 + 1)/2)$ and 4.5.
- When the selected labels involve only binary answers, the threshold is set to be 0, $0.5(i.e. = (0 + 1)/2$ and 1.
- When the Threshold is a combination of both types of answers, then the threshold can be written as the equation: $Thre = \alpha * Thre_7 + \beta * Thre_2$, where α and β are the weights

5.3 Result

5.3.1 Behavior Result

In this model, we use labels and EEG signals as variables, and emotion is the hidden variable bridge labels and EEG signals, they can be written as $Y = f(X|E)$, where E is emotion. The result of this experiment shows that if we have an individual's EEG signal, which has been collected from the single electron device after watching the advertising, the accuracy of predicting if this individual would or not make a purchase of the corresponding product in the advertising is around 75% based on the SVM model. In the experiment, each participant has been selected as the out sample data, and then the rest of the participants are in-sample data(90% of in-sample data is the training dataset, while the 10% of the in-sample data is the testing dataset, bootstrapping has been applied to the 90% in sample training dataset.

The accuracy of prediction using SVM over the ranked answers is 77.28%. In this setting, the threshold of ranked answers results is 4. The recall score of this model is 72%, and the F score for this model is 75%. In the same threshold of ranked answers result is 4, each category prediction is: the likelihood of purchasing the car is 63.5%, the likelihood of purchasing the cloth is 92.3%, the likelihood of purchasing the digital is 68.5% and the likelihood of purchasing the food is 82.76%.

The accuracy of prediction using SVM model over combined ranked, and binary answers in 75.4% under the conditions that ranked and binary answers have equal weighting (of 0.5), the threshold of ranked answers is 4, the threshold of a binary answer is 0.4, the threshold of the whole dataset is 0.4. The recall score of this model is 69%, and the F score for this model is 72%. In the same experiment setting, each category prediction is: the likelihood of purchasing the car is 59.6%, the likelihood of purchasing the clothes is

92.3%, the likelihood of purchasing the digital is 64.8% and the likelihood of purchasing the food is 81.03%.

This indicates that EEG collected using single-electrode wearable devices is above 70% accuracy for prediction whether customers would purchase the product after watching the advertisement, and it can achieve higher accuracy prediction and reach about 75%.

The out of sample predictions are tested in two cases, and in each case, the results are the average/mean of picking up different thirty individual participants as out sample test. When it is only ranked answers, if the samples from the remaining participant from all four categories are combined into one dataset, then the prediction of purchasing power can reach 72.4%; if each product category is tested, then the car purchasing power can reach up to 50.09%, the clothing purchasing power can reach 78.01%, the digital products purchase power can reach 56.56% and the food purchase power can reach 69.34%. When it is the combined result of ranked and binary answers, if the samples from the remaining participant from all four categories are combined into one dataset, then the prediction of purchasing power can reach 71.2%; if each product category is tested, then the car purchasing power can reach up to 51.8%, the clothing purchasing power can reach 65.9%, the digital products purchase power can reach 57.2% and the food purchase power can reach 60.62%.

In these results, for the food in Table 5.6, there is a significant difference between the thresholds with a p-value 0.002632 in the ANOVA test, and the two-sample t-test shows that the 0.4 and 0.6 thresholds,0.5 and 0.6 thresholds are significantly different while 0.4 and 0.5 thresholds are not significantly different. In Table 5.5, ANOVA test has a p-value 0.000118, which means there is a significant difference, and the two-sample t-test has a p-value 0.002, which means threshold 0.5 and 0.6 are not significantly different. Digital products have ANOVA test shows p-value is 0.00042, the p-value for the two-sample test is 0.001; clothes have ANOVA test p-value 0.0393 and the two-sample t-test 0.0262; in

Product type	Threshold 3.5	Threshold 4	Threshold 4.5
Car	36.5400%	63.5600%	82.6900%
Food	60.3400%	82.7600%	93.1000%
Digital	38.8900%	68.5200%	87.0400%
Clothes	67.3100%	92.3100%	96.1500%
All dataset	59.5100%	77.2800%	89.8900%

Table 5.4: Accuracy prediction of the ranked answer to different type of product at different thresholds

the whole dataset, the ANOVA test has p-value 0.000111, and the two-sample t-test has p-value 0.0008044, which means there is no significant difference.

Also, in the research, each label Y has been tested, but it is a difficult test, and the result is not balanced, the recall and F scores are both low, that is why those predictions are not mentioned in the results. Different thresholds for each model have also been tested during the experiment. However, they are not good enough to accurately explain the model.

75% is a relatively high result to judge the purchasing power based on the EEG signal after watching a piece of advertising. Though there is no research on the direct link of EEG signals and purchase intention, research has been done on using emotions to quantify the purchasing power. In 2006, John Pawle and Peter Cooper showed that emotional factors to brand decision making range from 63%-85% [PC06] [Uti05].

5.3.2 Cross-Validation And Learning Curve

The two models with the whole dataset are selected to draw the cross-validation and learning curve. The reasons for only selecting these two are they give the best prediction results, and they are directly related to the chapter's hypothesis: predicting the likelihood

Threshold to each type answer in car	Threshold of the car dataset		
	$Thre = 0.4$	$Thre = 0.5$	$Thre = 0.6$
$Thres_7 = 0.4$, $Thres_2 = 3.5$	36.5400%	36.5400%	82.6900%
$Thres_7 = 0.4$, $Thres_2 = 4$	59.6200%	59.6200%	86.5400%
$Thres_7 = 0.4$, $Thres_2 = 4.5$	75.0000%	75.0000%	90.3800%
$Thres_7 = 0.5$, $Thres_2 = 3.5$	36.5400%	36.5400%	90.3800%
$Thres_7 = 0.5$, $Thres_2 = 4$	63.4600%	63.4600%	90.3800%
$Thres_7 = 0.5$, $Thres_2 = 4.5$	80.7700%	80.7700%	92.3100%
$Thres_7 = 0.6$, $Thres_2 = 3.5$	36.5400%	36.5400%	98.0800%
$Thres_7 = 0.6$, $Thres_2 = 4$	63.4600%	63.4600%	98.0800%
$Thres_7 = 0.6$, $Thres_2 = 4.5$	80.7700%	80.7700%	100.0000%

Table 5.5: Likelihood of purchasing the car in the combined answer model at different thresholds

Threshold to each type answer in food	Threshold of the food dataset		
	$Thre = 0.4$	$Thre = 0.5$	$Thre = 0.6$
$Thres_7 = 0.4$, $Thres_2 = 3.5$	60.3400%	60.3400%	96.5500%
$Thres_7 = 0.4$, $Thres_2 = 4$	81.0300%	79.3100%	96.5500%
$Thres_7 = 0.4$, $Thres_2 = 4.5$	91.3800%	91.3800%	98.2800%
$Thres_7 = 0.5$, $Thres_2 = 3.5$	60.3400%	60.3400%	98.2800%
$Thres_7 = 0.5$, $Thres_2 = 4$	82.7600%	82.7600%	98.2800%
$Thres_7 = 0.5$, $Thres_2 = 4.5$	93.1000%	93.1000%	100.0000%
$Thres_7 = 0.6$, $Thres_2 = 3.5$	60.3400%	60.3400%	94.8300%
$Thres_7 = 0.6$, $Thres_2 = 4$	82.7600%	82.7600%	94.8300%
$Thres_7 = 0.6$, $Thres_2 = 4.5$	93.1000%	93.1000%	94.8300%

Table 5.6: Likelihood of purchasing the food in the combined answer model at different thresholds

Threshold to each type answer in digital	Threshold of the digital dataset		
	$Thre = 0.4$	$Thre = 0.5$	$Thre = 0.6$
$Thres_7 = 0.4$, $Thres_2 = 3.5$	40.7400%	40.7400%	94.4400%
$Thres_7 = 0.4$, $Thres_2 = 4$	64.8100%	64.8100%	96.3000%
$Thres_7 = 0.4$, $Thres_2 = 4.5$	81.4800%	81.4800%	98.1500%
$Thres_7 = 0.5$, $Thres_2 = 3.5$	38.8900%	38.8900%	98.1500%
$Thres_7 = 0.5$, $Thres_2 = 4$	68.5200%	68.5200%	98.1500%
$Thres_7 = 0.5$, $Thres_2 = 4.5$	87.0400%	87.0400%	98.1500%
$Thres_7 = 0.6$, $Thres_2 = 3.5$	38.8900%	38.8900%	98.1500%
$Thres_7 = 0.6$, $Thres_2 = 4$	68.5200%	68.5200%	98.1500%
$Thres_7 = 0.6$, $Thres_2 = 4.5$	87.0400%	87.0400%	NA

Table 5.7: Likelihood of purchasing the digital in the combined answer model at different thresholds

Threshold to each type answer in clothes	Threshold of the clothes dataset		
	$Thre = 0.4$	$Thre = 0.5$	$Thre = 0.6$
$Thres_7 = 0.4$, $Thres_2 = 3.5$	65.3800%	65.3800%	100.0000%
$Thres_7 = 0.4$, $Thres_2 = 4$	92.3100%	92.3100%	100.0000%
$Thres_7 = 0.4$, $Thres_2 = 4.5$	96.1500%	96.1500%	100.0000%
$Thres_7 = 0.5$, $Thres_2 = 3.5$	69.2300%	69.2300%	100.0000%
$Thres_7 = 0.5$, $Thres_2 = 4$	92.3100%	92.3100%	100.0000%
$Thres_7 = 0.5$, $Thres_2 = 4.5$	96.1500%	96.1500%	100.0000%
$Thres_7 = 0.6$, $Thres_2 = 3.5$	69.2300%	69.2300%	98.0800%
$Thres_7 = 0.6$, $Thres_2 = 4$	92.3100%	92.3100%	98.0800%
$Thres_7 = 0.6$, $Thres_2 = 4.5$	96.1500%	96.1500%	98.0800%

Table 5.8: Likelihood of purchasing the clothes in the combined answer model at different thresholds

Threshold to each type answer in whole dataset	Threshold of the whole dataset		
	$Thre = 0.4$	$Thre = 0.5$	$Thre = 0.6$
$Thres_7 = 0.4$, $Thres_2 = 3.5$	60.4400%	60.0200%	94.0300%
$Thres_7 = 0.4$, $Thres_2 = 4$	75.4100%	75.8600%	95.0200%
$Thres_7 = 0.4$, $Thres_2 = 4.5$	86.6300%	86.6300%	96.8600%
$Thres_7 = 0.5$, $Thres_2 = 3.5$	59.5100%	59.5100%	97.7300%
$Thres_7 = 0.5$, $Thres_2 = 4$	77.2800%	77.2800%	97.7300%
$Thres_7 = 0.5$, $Thres_2 = 4.5$	89.4500%	89.4500%	97.7300%
$Thres_7 = 0.6$, $Thres_2 = 3.5$	59.5100%	59.5100%	97.7300%
$Thres_7 = 0.6$, $Thres_2 = 4$	77.2800%	77.2800%	97.7300%
$Thres_7 = 0.6$, $Thres_2 = 4.5$	89.4500%	89.4500%	97.7300%

Table 5.9: Likelihood of purchasing in the combined answer model at different thresholds

of customers purchasing the products.

The cross-validation and learning curve is shown in Figure 5.2 and Figure 5.3

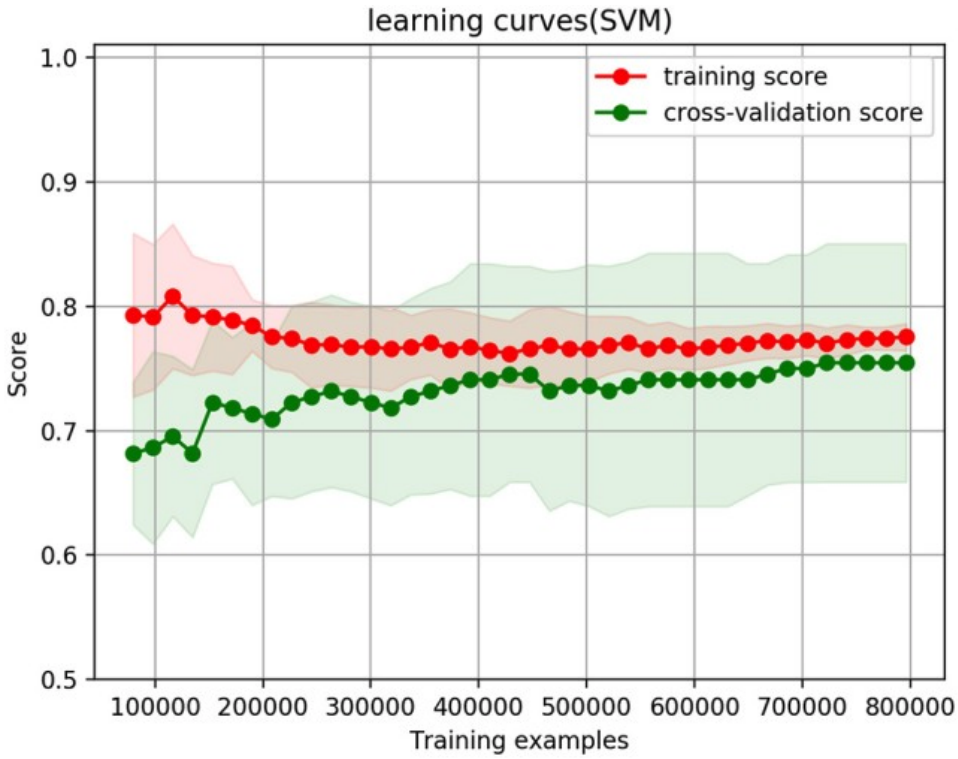


Figure 5.2: Cross validation curves when only ranked answers are selected, with a threshold of 4.

Figure 5.2 shows the cross-validation and learning curve when only the ranked answers are selected. Figure 5.3 shows the cross-validation and learning curve when the thresholds of ranked answers is 4, and binary answers threshold is 0.4, their weight is equal at 0.5, and the whole dataset thresholds is 0.4. In both figures, the X-axis represents the size of training examples; Y-axis represents the accuracy prediction score. The red lines represent the training score, and green lines represent the cross-validation score/testing score.

In the ranked answers only case in Figure 5.2, training and testing curves converge to 77.28%; when the combination of ranked answers and binary answers are selected (ranked answers threshold is 4, and binary answers threshold is 0.4), as shown in Figure 5.3,

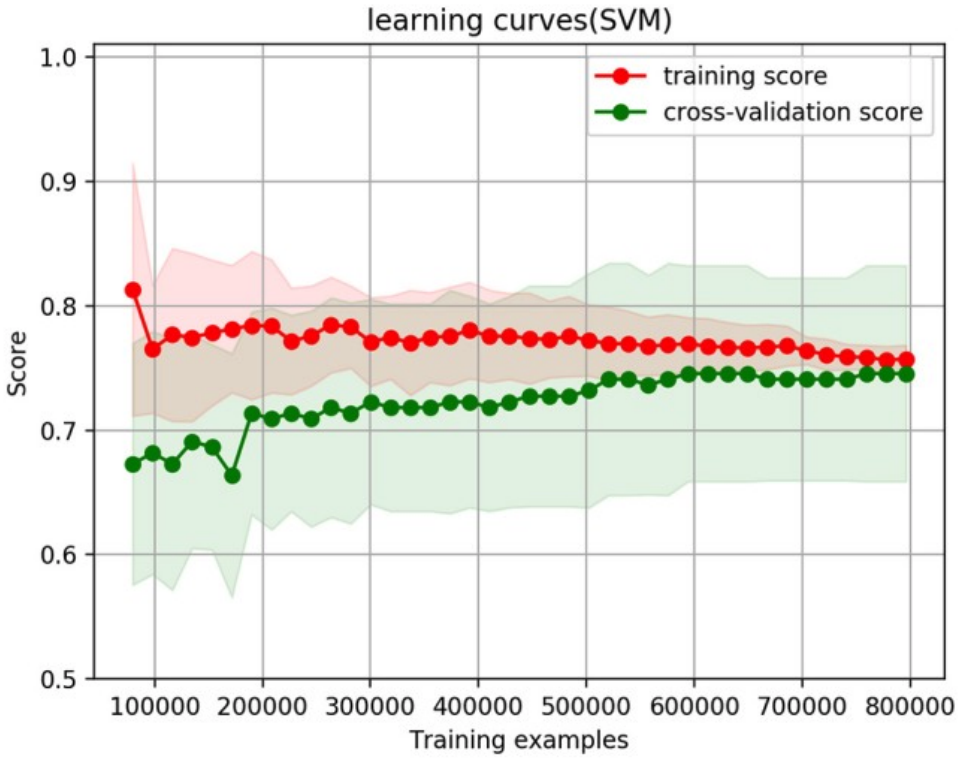


Figure 5.3: Cross-validation curves when thresholds of ranked answers is 4 and binary answers threshold is 0.4, their weight is equal at 0.5, and the whole dataset thresholds is 0.4

the training and testing curves converge to 75.4%. The shaded red and green areas are standard deviations at confidence interval 10% to its corresponding training score and corresponding testing score. Mathematically, they are calculated as $(training_{mean} - training_{std}, training_{mean} + training_{std})$ and $(testing_{mean} - testing_{std}, testing_{mean} + testing_{std})$. Both figures show that the training curve starts with narrow variance, and then gradually increase until training example size reaches 100, at which point the variance of training curve decreases. In testing curves, the variance remains approximately the same from the beginning till the end throughout the training example size increases; it remains the same because the testing data size does not change regardless of how training example size changes. The convergence of training and testing curves means there is no over-fitting and no under-fitting issues of the model; the SVM model we set up and use therefore performs well.

5.4 Discussion

The approximately 75% accuracy prediction and the converging cross-validation curve result together show that the EEG signal and edited AIDA metrics model is a suitable model. This result is sufficient to declare EEG a useful dataset to collect in the advertising industry.

Further improvements can be made on the accuracy of prediction by improving the chosen thresholds and increasing the study sample size rather than relying on bootstrapping. Regarding the threshold, receiver operating characteristic (ROC) can be used in the future work for visually and a more accurately selecting the threshold. The accuracy of prediction may be further improved through selecting both male and female participants and gender-neutral products and their advertisement.

In this research, accuracy has been used rather than Mean absolute percentage error (MAPE) Or mean squared error because it is a binary result, it is either yes or no question represented as 0 or 1.

The statistical results show that the ranked answers with a threshold of 4 have a better result than the combined rank and binary answers, with about 2% more accuracy. It also shows that each category prediction is lower than the whole dataset prediction in both models. It shows that products like cars have the lowest accuracy of prediction and clothes have the highest accuracy of prediction among the four categories. This may be caused by cars being a costly product, and the decision to purchase one depends more on lifestyle and personal circumstance than advertising. In this case, the prediction accuracy can be improved by pre-selecting participants to control for these factors, or by adjusting predictions based on the personal circumstances. In the experiment, the F scores and the recall scores are also good in the model; it happens to be consistent that the whole dataset scores are better than each category dataset. The out of sample test result is not

as good as the sample test, but it is acceptable, which further indicates the model is good.

In the current research, the emotions and opinions of the user have been used as a hidden variable to bridge EEG signal and self-reported metrics to evaluate the impact of advertisement and its power to influence purchasing. In the literature, research has been done on the relationship of Theta wavelength with emotion and the relationship of Beta Wavelength with emotion [LP12] [WCC16]. Considering that EEG signals contain components at many other frequencies, it is worth further investigation of the relationship between EEG signal and emotion, and how they impact a decision to make a purchase.

In past attempts, an AIDA model has been used to quantify emotion metrics in terms of scores for attention, interest, desire and action aspects after watching the advertisement [SJ25]. In this thesis, the idea of the AIDA model and add additional dimensions to quantify these emotions to achieve a more precise measure is followed. The ranked and binary answers have equal waiting in accordance with the AIDA model and literature mentioned in Section 5.2.1 which considers all labels with equal impact. It is also interesting to further execute a non-parametric machine learning model to assess the importance of each label or a grouping of similar labels for purchasing power. More details on the emotion model this experiment uses are described in section 5.2.1.

This is a new method in Neuromarketing. The advantage of this method is to be able to assess the impact fast. The model, once found, can work in real-time on EEG signals, in a highly automated way. Something not possible in traditional methods that involve data collection via surveys or discussion, and then analysis, which requires both time and man-power to accomplish; or assessing the impact through sales figures afterwards, which can be easier to automate, but has much longer cycle times.

The proposed evaluation method also shows less bias due to EEG data being involuntary and therefore, not subject to conscious and experimental bias. Unlike in a focus group

study conducted by an advertisement company, the study in an academic setting removes the potential for study biases in which participants tend to give kinder answers than they may think, out of goodwill or some sense of social obligation [DPGA02] [SL09] [MGB04]. The questionnaire questions are written neutrally and asks similar questions from slightly different angles to validate consistency. Because these questions yield consistent results, it indicates a high likelihood that they represent the true thought of participants. The questions in the questionnaire do not include identifiable or confidential information of any individual; therefore removing potential reasons for participants to hide their true opinion.

Furthermore, the device used in this research is a single-electrode wearable device that is low cost, portable, and easy to use, allowing this kind of data collection to be scaled much more rapidly than machines traditionally associated with functional brain imaging in neuroscience, such as traditional clinical-grade multi-channel EEG setups, and fMRI [Sig15]. Multiple types of EEG devices have been studied in the literature and applied to neuromarketing studies; the results do not show an appreciable improvement in accuracy of multi-channel EEG systems versus single electrode EEG systems [LH02]. Therefore, single electrode EEG devices is sufficient for the experiment, and the ease of use of a simple, self-contained and battery-operated wearable device opens up new kind of customer engagement opportunities where their EEG can be recorded throughout the day in settings that are much more normal than would be the case in a lab setting, removing any influence that may come from the experimental setup.

The most significant part of this research is the extension of modelling emotions with a single-wavelength collected from EEG to multidimensional wavelengths, allowing the extraction of more informational content from EEG than previously attempted.

The result of this research can be further applied to individual consumers behaviour; to allow the advertising industry to tailor their advertisement for maximum impact, and

adapted to work for TV an movie studios to predict viewership rates of movies and TVs from trailers.

5.5 Conclusion

A model that can use EEG signals measured using low-cost consumer-grade EEG headsets taken while a consumer watches an advertisement is proposed. It can rapidly predict the consumer's likelihood of purchasing a product. While further research can be done on the selection of thresholds, and the quality of the result can be improved with the collection of larger datasets, the method as shown is nevertheless easy to deploy, yields rapid results, scales better than any existing method, and introduces less experimental and environmental bias.

Chapter 6

Data Privacy And Security

Data privacy and security have always been an essential part of the financial industry. In finance, there is a vast amount of data in customers; in transactions; and in different asset pricing, due-diligence. How to find and implement a fast and efficient method to work in a real financial system working with real-world data has been a challenge, and it needs to be solved.

Differential privacy has been a very popular topic in finance. Topics cover combining Differential Privacy and Secure Multi-Party Computation with Machine Learning through present a feasible protocol for learning neural networks in a collaborative way while preserving the privacy of each record [CGBL$^+$17]. Sweeney has also published a paper on K-anonymity: a model for protecting privacy, this research is important because it can solve the bank world real-world system Datafly [Swe02]. In addition, many privacy and security setup has been distributed through crypto networks in finance. A model called cryptoNets have been used to encrypted data with high throughput and accuracy in the client's dataset [GBDL$^+$16].

In this chapter, a new deep network architecture that combines neural network and

data encryption is introduced, aiming to solve privacy and data security problem. With privacy-preserving feed-forward propagation, server (model provider) can serve for the user without directly accessing the user's data. We proved and evaluated that dropping a few activation outputs can encrypt data for invertible activation function. We also evaluated that ramp function is better than rectifier in term of encryption. We proposed privacy-preserving error-backpropagation, by which server can train a neural network for the user without accessing users' data. With this method, data compression for data transmission is available when the number of neurons is less than the size of input data.

6.1 Introduction

Deep learning, also known as deep neural networks [DY14a], has been proven to be successful for classification and regression tasks. Once well established, deep learning models exhibited promising performance, which is desirable for many real-time prediction/decision-making applications [edm16], like automatic speech recognition, image recognition, and natural language processing.

For these applications, due to hardware limitations (i.e. the computation capability due to the power consumption limitation), it is very expensive to implement deep learning algorithms entirely on users' local devices (sensors, mobiles, and even laptops). As a result, the typical approach is to collect data locally at first and then transmit the data to a remote server/cluster, and then apply the deep networks for training/prediction. However, such an approach suffers an important privacy issue, especially for those sensitive data, e.g. patients clinical data (like fMRI images) and sensitive environment data (like video monitoring in public space).

There has been some research on using neural network prediction on encrypted data rather than the original data in the situation so that users/sensors will not share them

with remote servers [BG15] [RJ15]. However, their approach brings in extra computation cost, and it would not be safe once the key for encryption is hacked.

6.2 Method: Dropping Activation Outputs with Localized First-layer Deep Network

This chapter introduces a new architecture of a localized first-layer deep network, then introduce the proposed encryption methods for both invertible activation (like sigmoid) and non-invertible activation (like rectifier). Neither two-layer nor multiple layers need to be considered because they will lead to similar results as one-layer. The reasons for this is shown in the mathematical proof. The focus of this chapter will be on a method for encrypting data for invertible activation function.

The proposed data privacy and security architecture decomposes layers of a neural network between local device and server, as shown in Figure 6.1. The local device transfers the activation outputs of the first hidden layer to the server, and the server can only use the activation outputs to compute the result. This architecture can enable privacy protection in both non-invertible and invertible activation functions.

6.2.1 Non-invertible Activation Functions

Several up-to-date activation functions are non-invertible [GBB11] [GWFM$^+$13], for example, rectifier is commonly used in deep neural network, and is able to reach a network's best performance without any unsupervised pre-training on unlabeled data [GBB11]. Its output is a linear function of the inputs, so the vanishing gradient problem can be reduced. The rectifier is shown on Equation 6.1, where $z = x * W + b$. It sets all negative

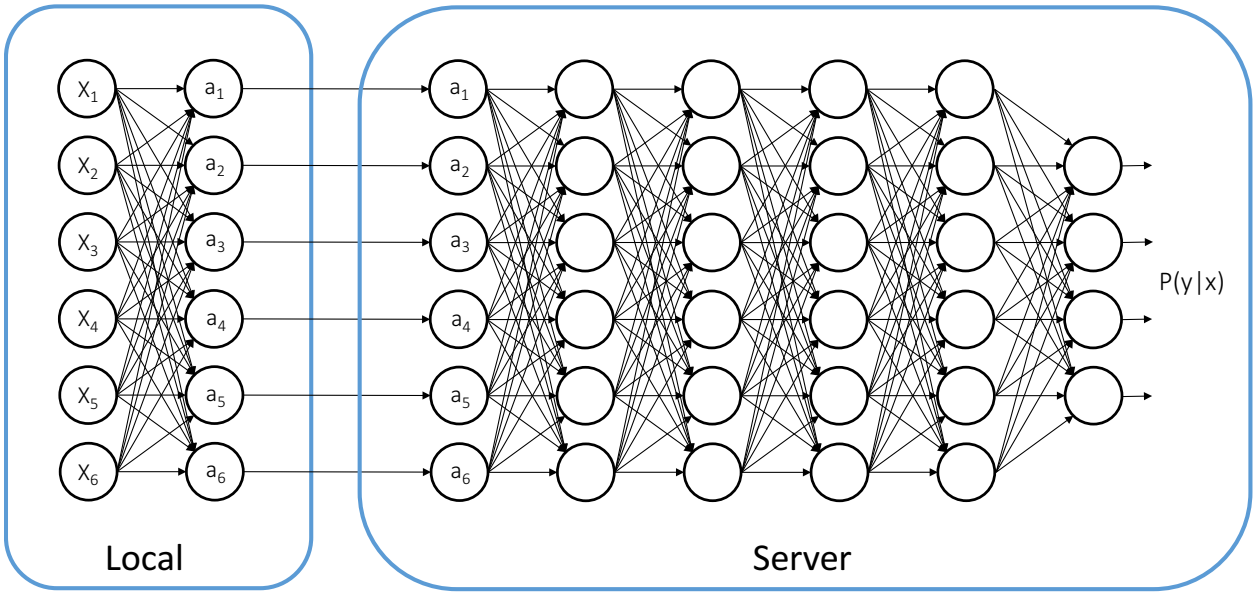


Figure 6.1: Spliting neuron network during feed-forward propagation.

activation outputs to zero.

$$a = f(z) = \begin{cases} 0 & , when\ z < 0 \\ z & , otherwise \end{cases} \tag{6.1}$$

Non-invertible activation functions can naturally encrypt the activation outputs to some degree. For non-invertible activation functions, the best way to reconstruct input data is to use an approximated inverse activation function $\hat{f}^{-1}$ and the transposed weight matrix W^T (instead of W^{-1}), as Equation 6.2 describes. The mathematical explanation can be found in Section 6.3.

$$\hat{x} = (\hat{f}^{-1}(a) - b) * W^T \tag{6.2}$$

However, due to the linearity of a rectifier, the activation outputs is highly related to the scale of feature patterns from inputs. The general features of the original data can be decrypted by Equation 6.2. To alleviate this problem, a ramp activation function is

introduced, as Equation 6.3 shows, where $z = x * W + b$, and v reflects a small value. Smaller v will cut more scale information so that the activation outputs will contain less scale information but more information of feature combinations (i.e. a set of small feature identifiers). The rectifier and ramp functions will be evaluated in Section 6.4.1.

$$a = f(z) = \begin{cases} 0 & , when\ z < 0 \\ z & , when\ 0 <= z < v \\ v & , when\ z >= v \end{cases} \tag{6.3}$$

In the view of information, the non-invertible outputs will result in information loss, so the original data cannot be fully reconstructed. However, from a neural network perspective, the non-invertible function selects features. Therefore, features are not lost but can even improve the accuracy [GBB11]. If the input data is an image, information loss would manifest as blurring and adding noise to the image, for example. Based on this principle, in the rectifying layer, even if just one activation output is turned to zero from negative value, the original input data cannot be reconstructed completely. In practice, due to the sparse property of the rectifying layer, there will be a large portion of rectifying outputs being zero - therefore, the reconstructed data will be distorted completely.

Hence, data encryption and privacy are realized through non-invertible activation functions, as the input of each hidden layer cannot be reconstructed. Even if all parameters of the model are known, the original inputs still cannot be reconstructed by the server. Hence, privacy is preserved for users. Apart from the rectifier and ramp functions, other activation function such as binary step and max-out [GWFM$^+$13] are all non-invertible.

6.2.2 Invertible Activation Functions

Different from non-invertible activation functions, reconstruction is solvable when the activation function is invertible, e.g. sigmoid, hyperbolic tangent, softplus, and others. The sigmoid and inverse sigmoid is shown in Equation 6.4 and 6.5.

$$a = \frac{1}{1 + e^{-(x*W+b)}} \tag{6.4}$$

$$x = (-ln(\frac{1}{a} - 1) - b) * W^{-1} \tag{6.5}$$

Therefore, the only ways to encrypt x is by modifying a, W or b. The modification will bring uncertainty to the neural network. Therefore, the key is to find a modification method which does not affect the final predicting result.

Due to the sparse behaviour of the state-of-the-art neural network, accuracy will not be affected by slightly changing activation outputs and weights. First of all, during training, the proposed method would not compromise the learning results, because both Dropout and Dropconnect are the state-of-the-art way to avoid overfitting. For testing/inferencing, non-invertible activation would not affect the result at all. For invertible activation, dropping a few of activation outputs also would not harm the final result, that is because:

1) From a theory point of view, Dropout training can be considered as training many sub-networks to do the same job; the result is the averaged of all sub-networks, which is one sort of ensemble learning and recently proven to be a Gaussian process [SHK+14] [Gal16]. On the other hand, reasonable uncertainty of neural network during testing would not affect the testing result if the network has uncertainty during training [GG16] [GSB16] [BCKW15].

2) From an experience point of view, the Dropping probability the method required (0.5%) comparing with the Dropping probability during training (usually 20 to 50%) is quite low and would not affect the final averaged result of all sub-networks. Tables 6.1 and 6.2 show the effect with different dropping probabilities using a sigmoid function. As expected, this method would not lead to noticeable performance decreases, as a small dropping probability is enough for encryption.

The sparse behavior of neural network can be achieved by using the methods of Dropout, Dropconnect, and Autoencoder ([HSK+12] [SHK+14] [WZZ+13] [VLL+10]). In this chapter, the idea of Dropping activation outputs and Dropping connections is proposed. The definitions of Dropping activation outputs and Dropping connections are different from Dropout and Dropconnect. In this research, the Dropping activation outputs and Dropping connections are new terms, they are specifically defined under the data privacy and data security environment, the action to Dropping activation outputs is the same as Dropout, and the action to the Dropping connections is the same to the Dropconnect, but both of them applied onto an extra deep neuron layer on the local/clients rather than server/cloud. They are applied in feed-forward propagation, whereas Dropout and Dropconnect are applied in the error backpropagation. A figure of Dropping activation outputs and Dropping connections shown in Figure 6.2 and Figure 6.3.

Dropping activation outputs during the feed-forward process: For encrypting

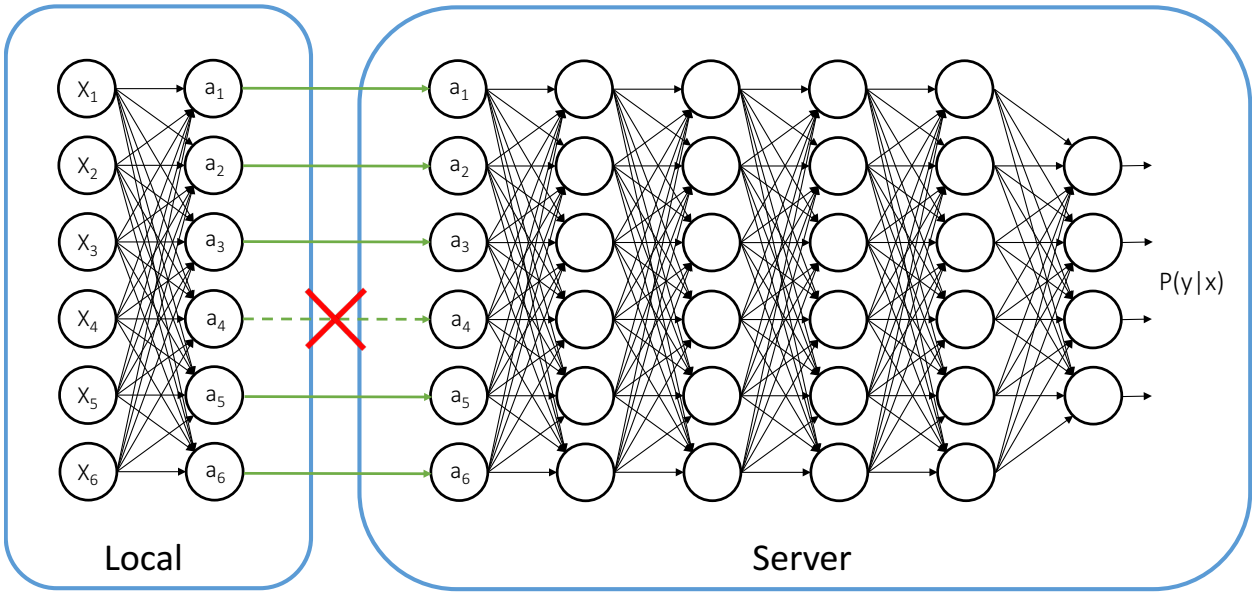


Figure 6.2: Dropping activation output during feed-forward propagation.

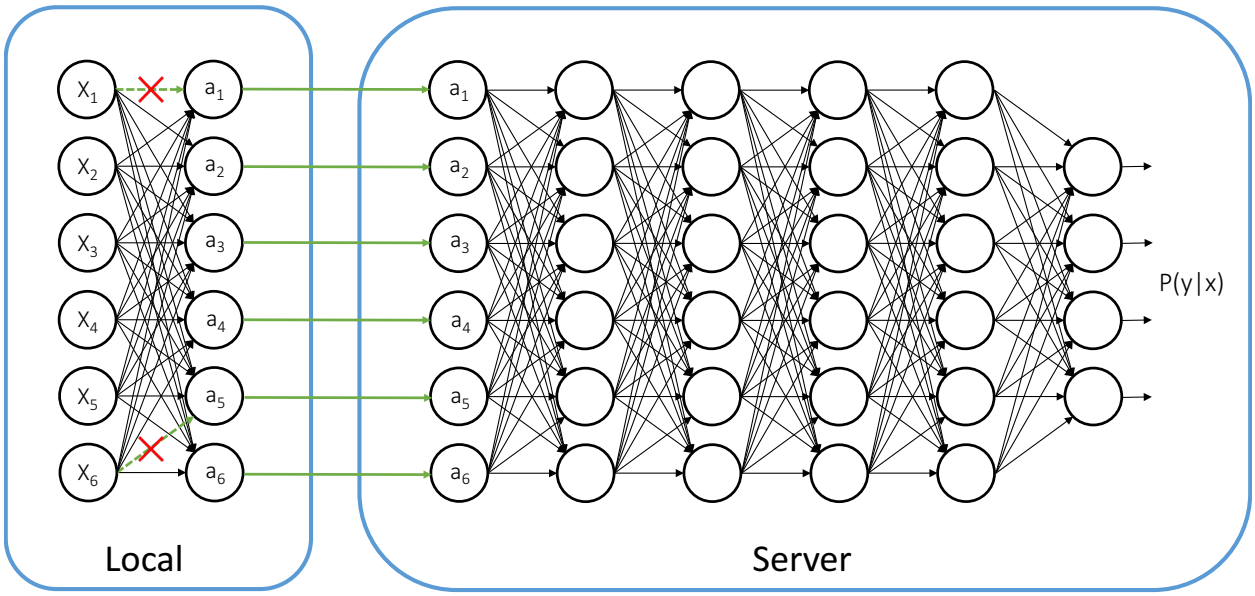


Figure 6.3: Dropping connection during feed-forward propagation.

the data during the feed-forward propagation process, a method called Dropping activation output is proposed. Mathematically the modified activation output $\hat{a}$ is written as

$$\hat{a} = d \odot f(x * W + b) \tag{6.6}$$

where, the element-wise multiplication (Hadamard product) is denoted by $\odot$, and d is a binary vector randomly some activation outputs to zero.

Setting an activation output to zero is equivalent to removing all connections between a neuron and all input data. More details will be described in Section 6.3.

Equation 2.25 and 6.2 can be used to reconstruct the data, and it was found that similar with non-invertible activation, Equation 6.2 can better reconstruct the data.

Dropping connections during the feed-forward process:
Another purposed method is called Dropping connections. In dropping connections, some elements of weight matrix W are set to zero. The modified weight matrix $\hat{W}$ and the modified activation output $\tilde{a}$ are written as

$$\hat{W} = W \odot D \tag{6.7}$$

$$\tilde{a} = f(x * \hat{W} + b) \tag{6.8}$$

where, D is a binary matrix with some elements are zero. However, the experiment shows that this method is not effective enough to encrypt the data.

6.3 Mathematical Explanation

If the activation is linear, we have:

$$a = x * W + b \tag{6.9}$$

$$x = (a - b) * W^{-1} \tag{6.10}$$

We now discuss why dropping connections won't encrypt as good as dropping activation.

6.3.1 Dropping Connections

Assume ΔW is the different between W and $\tilde{W}$, i.e. $\Delta W = W - \tilde{W}$, replace a in Equation 6.10 by Equation 6.9, let $\tilde{a}$ represent the new a that is calculated through the changed W, then the new $\tilde{x}$ can be written as

$$\begin{aligned} \tilde{x} &= (\tilde{a} - b)W^{-1} \\ &= (x * W \odot D + b - b)W^{-1} \\ &= x(W \odot D)W^{-1} \\ &= x(W - \Delta W)W^{-1} \\ &= x(I - \Delta W W^{-1}) \end{aligned} \tag{6.11}$$

Note in ΔW, only a few entries are non-zero; hence, the product of $\Delta W W^{-1}$ is very small, which means it can be neglected. The reason behind it is the product of WW^{-1} is an identity matrix, which means the sum of the product of entries in row W and their corresponding entries in column W^{-1} is 1 when the sum position is diagonal, but the sum is 0 elsewhere. Then $(I - \Delta W W^{-1})$ approximates to an identity matrix. This means the input data is nearly the same as the dropping connection result, written as $\tilde{x} \approx x$, therefore the encrypt performance is poor. Mathematically, it corresponds to an Affine transformation with no change circumstances [Haz01]. This result applies to any c-by-d matrix W, where $c \leq d$. As $\Delta W_{c \times d}$ is a c-by-d matrix with a few entries

change, which means only a few terms are affected in the summation of d terms, therefore, $\Delta W_{c\times d} W_{c\times d}^{-1} \approx 0$, hence $I_c - \Delta W_{c\times d} W_{c\times d}^{-1}$ approximate to identity matrix I_c, which means $\tilde{x}_{1\times c} \approx x_{1\times c}$.

To give an example when W is a 3-by-5 matrix, and $*$ represent entries that are not zero, let $W = \begin{bmatrix} 1 & 3 & 1 & 7 & 2 \\ 2 & 2 & 7 & 5 & 3 \\ 2 & 5 & 3 & 1 & 1 \end{bmatrix}$

- When the changing entry/entries are on the same row in ΔW, then it is the same row that is non-zero in $\Delta W W^{-1}$:

$$\text{if } \Delta W = \begin{bmatrix} 0 & 3 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \end{bmatrix} \text{ or } \begin{bmatrix} 0 & 3 & 0 & 7 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \end{bmatrix}$$

$$\rightarrow \Delta W W^{-1} = \begin{bmatrix} * & * & * \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix} \tag{6.12}$$

$$\text{if } \Delta W = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 \\ 2 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \end{bmatrix} \text{ or } \begin{bmatrix} 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 7 & 5 & 0 \\ 0 & 0 & 0 & 0 & 0 \end{bmatrix}$$

$$\rightarrow \Delta W W^{-1} = \begin{bmatrix} 0 & 0 & 0 \\ * & * & * \\ 0 & 0 & 0 \end{bmatrix} \tag{6.13}$$

$$\text{if } \Delta W = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 3 & 0 & 0 \end{bmatrix} \text{ or } \begin{bmatrix} 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 5 & 3 & 1 & 1 \end{bmatrix}$$

$$\rightarrow \Delta W W^{-1} = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ * & * & * \end{bmatrix} \tag{6.14}$$

- When changing entries are at different row, then those corresponding rows in $\Delta W W^{-1}$ have non-zero entries.

$$\text{if } \Delta W = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 \\ 0 & 2 & 0 & 0 & 0 \\ 0 & 0 & 3 & 0 & 0 \end{bmatrix}$$

$$\rightarrow \Delta W W^{-1} = \begin{bmatrix} 0 & 0 & 0 \\ * & * & * \\ * & * & * \end{bmatrix} \tag{6.15}$$

$$\text{if } \Delta W = \begin{bmatrix} 0 & 0 & 0 & 7 & 0 \\ 0 & 2 & 0 & 0 & 0 \\ 0 & 0 & 3 & 0 & 0 \end{bmatrix}$$

$$\rightarrow \Delta W W^{-1} = \begin{bmatrix} * & * & * \\ * & * & * \\ * & * & * \end{bmatrix} \tag{6.16}$$

W^{-1} is the inverse of W, therefore $WW^{-1} = I_3$ (i.e.

$$W_{11}W_{11}^{-1} + W_{12}W_{21}^{-1} + W_{13}W_{31}^{-1} + W_{14}W_{41}^{-1} + W_{15}W_{51}^{-1} = 1$$

$$W_{11}W_{12}^{-1} + W_{12}W_{22}^{-1} + W_{13}W_{32}^{-1} + W_{14}W_{42}^{-1} + W_{15}W_{52}^{-1} = 0$$

$$W_{11}W_{13}^{-1} + W_{12}W_{23}^{-1} + W_{13}W_{33}^{-1} + W_{14}W_{43}^{-1} + W_{15}W_{53}^{-1} = 0$$

$$W_{21}W_{11}^{-1} + W_{22}W_{21}^{-1} + W_{23}W_{31}^{-1} + W_{24}W_{41}^{-1} + W_{25}W_{51}^{-1} = 0$$

$$W_{21}W_{12}^{-1} + W_{22}W_{22}^{-1} + W_{23}W_{32}^{-1} + W_{24}W_{42}^{-1} + W_{25}W_{52}^{-1} = 1$$

$$W_{21}W_{13}^{-1} + W_{22}W_{23}^{-1} + W_{23}W_{33}^{-1} + W_{24}W_{43}^{-1} + W_{25}W_{53}^{-1} = 0$$

$$W_{31}W_{11}^{-1} + W_{32}W_{21}^{-1} + W_{33}W_{31}^{-1} + W_{34}W_{41}^{-1} + W_{35}W_{51}^{-1} = 0$$

$$W_{31}W_{12}^{-1} + W_{32}W_{22}^{-1} + W_{33}W_{32}^{-1} + W_{34}W_{42}^{-1} + W_{35}W_{52}^{-1} = 0$$

$$W_{31}W_{13}^{-1} + W_{32}W_{23}^{-1} + W_{33}W_{33}^{-1} + W_{34}W_{43}^{-1} + W_{35}W_{53}^{-1} = 1$$

). However, ΔW represents a change of only a few entries in W, which means only a few terms are affected in the summation of five terms, therefore, $\Delta WW^{-1} \approx 0$, hence $I_3 - \Delta WW^{-1}$ approximate to identity matrix I_3, which means $\tilde{x}_{1\times3} = x_{1\times3}$.

6.3.2 Dropping Activation Outputs

For Dropping activation outputs, linear activation is also used in this section. Let the modified activation output be $\hat{a}$ and its corresponding modified input data be $\hat{x}$, then it can be written as

$$\hat{x} = (\hat{a} - b)W^{-1} \tag{6.17}$$

$$= (a - \Delta a - b)W^{-1}$$

$$= (a - b)W^{-1} - \Delta a W^{-1}$$

$$= x - \Delta a W^{-1}$$

$$= x(I - \frac{\Delta a W^{-1}}{x})$$

Hence for any activation output vector **a** in size 1-by-d and weight matrix W that has size c-by-d, where $c \leq d$. If a few entries of **a** change, then the modified vector $\hat{x}$ can be written as a product of 1-by-c original input data x and a matrix $(I - \frac{\Delta a_{1\times d} W^{-1}_{d\times c}}{x_{1\times c}})$, where all diagonal entries are not necessary to be 1. Then $x \not\approx \hat{x}$. Therefore it won't necessary to be no change circumstance in the Affine transformation.

For example, let $a = (a_1, a_2, a_3)$, $\Delta a = (\Delta a_1, \Delta a_2, \Delta a_3)$, and:

$$W^{-1}_{3\times 2} = \begin{bmatrix} w_{11} & w_{12} \\ w_{21} & w_{22} \\ w_{31} & w_{32} \end{bmatrix} \tag{6.18}$$

Then x can be written as:

$$x = (x_1, x_2) = aW^{-1}_{3\times 2} = (a_1, a_2, a_3) \begin{bmatrix} w_{11} & w_{12} \\ w_{21} & w_{22} \\ w_{31} & w_{32} \end{bmatrix} \tag{6.19}$$

$$= (a_1 w_{11} + a_2 w_{21} + a_3 w_{31}, a_1 w_{12} + a_2 w_{22} + a_3 w_{32})$$

$\hat{x}$ can be written as:

$$\hat{x} = (\hat{x_1}, \hat{x_2}) = \hat{a} W^{-1}_{3\times 2} \tag{6.20}$$

$$= (a_1 - \Delta a_1, a_2 - \Delta a_2, a_3 - \Delta a_3) \begin{bmatrix} w_{11} & w_{12} \\ w_{21} & w_{22} \\ w_{31} & w_{32} \end{bmatrix}$$

$$= (a_1 w_{11} - \Delta a_1 w_{11} + a_2 w_{21} - \Delta a_2 w_{21} + a_3 w_{31} - \Delta a_3 w_{31},$$

$$a_1 w_{12} - \Delta a_1 w_{12} + a_2 w_{22} - \Delta a_2 w_{22} + a_3 w_{32} - \Delta a_3 w_{32})$$

Which means

$$(\hat{x_1}, \hat{x_2})$$

$$= (x_1, x_2) \begin{bmatrix} 1 - \frac{\Delta a_1 w_{11} + \Delta a_2 w_{21} + \Delta a_3 w_{31}}{x_1} & 0 \\ 0 & 1 - \frac{\Delta a_1 w_{12} + \Delta a_2 w_{22} + \Delta a_3 w_{32}}{x_2} \end{bmatrix} \tag{6.21}$$

In this case, $\frac{\Delta a_1 w_{11} + \Delta a_2 w_{21} + \Delta a_3 w_{31}}{x_1}$ and $\frac{\Delta a_1 w_{12} + \Delta a_2 w_{22} + \Delta a_3 w_{32}}{x_2}$ are unknown and there is no restriction on them, which means they can be any number, hence it is not an identity matrix.

6.4 Experimental Studies

The MNIST dataset was used to evaluate these methods, which has a training set of 50k, a validation set of 10k and a test set of 10k. Each image is a 28×28 grey-scale digit, with ten classes in total. The MNIST dataset is a widely tested classical machine learning dataset; they are images with ten numbers: from 0 to 9. We adopted an accuracy of classification task to evaluate our model, which is a standard way of evaluating machine learning algorithms. For classification, the definition of accuracy is the percentage of correct prediction on the test set.

Dropping activation outputs was also evaluated on the CIFAR10 dataset, which is a more

challenging dataset compared to MNIST, and has 50k training images and 10k test images. Each image is a 32×32 RGB image, with ten classes in total (airplane, automobile, bird, cat, deer, dog, frog, horse, ship, truck).

6.4.1 Non-invertible Activation Functions

Rectifier The experimental network has three rectifying hidden layers, and each layer has 800 neurons so that the model can be represented as (784-800-800-800-10) from the input layer to the output layer.

Dropout has been applied during training to prevent overfitting. The Dropout probability from the input layer to the first layer is 20%, and the probabilities of other layers are 40%. No weights decay were used. All neural networks were trained by using Adam gradient descent [KB15] with mini-batches of size 500 and 2000 of epochs. This resulted in an accuracy of 98.87% (under the null hypothesis of the pairwise test with $p = 0.05$).

To evaluate the rectifier, Equation 6.22 and 6.23 are used to reconstruct the approximate input data. Figure 6.4 shows the original input data and the reconstructed input data from the activation outputs of first hidden layer by using Equation 6.22. It is clear that the reconstruction failed.

$$\hat{x} \approx (a - b) * W^{-1} \tag{6.22}$$

$$\hat{x} \approx (a - b) * W^{T} \tag{6.23}$$

The input data can be better reconstructed by Equation 6.23. As the middle column of Figure 6.5 demonstrated, even when the reconstructed data were totally distorted, the outline of original input data can still be recognized from the reconstructed data.

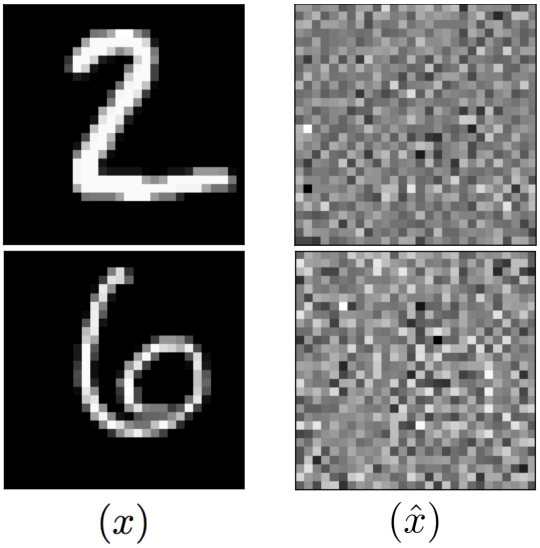


Figure 6.4: Reconstructed input data from rectifying output by using $\hat{x} \approx (a-b) * W^{-1}$ (Right); Original input data x (Left); Activation output $a = max(0, x * W + b)$. The KL Divergence between x and $\hat{x}$ for digit 2 and 6 are 1.91 and 1.88.

Further encryption can be applied by transferring the activation outputs of the second hidden layer, i.e. encrypting the data twice. Right hand side of Figure 6.5 shows the reconstructed input data by Equation 6.23, where the outline of digit cannot be recognized. However, applying more local rectifying layers will lead to higher local computation.

Ramp To let the activation outputs of the first hidden layer contain less scale information, a ramp activation function can be applied. The two test networks are the same as the previous rectifier network other than the activation function of the first layer being a ramp with v of 0.2 and 0.05 respectively. The accuracies of these two networks are both 98.87%, which is approximately the same as the previous pure rectifier neural network.

According to Figure 6.6, smaller v can better encrypt the input data, and the topology of input data cannot be determined from the reconstructed data.

The above experiments can be summarized as below:

- With the proposed architecture, no matter what kind of activation function is used,

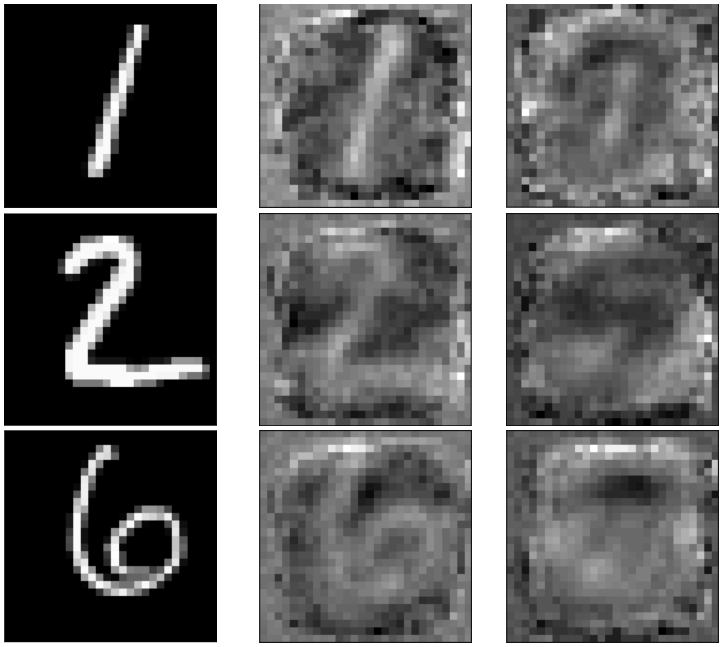


Figure 6.5: Reconstructed input data from rectifying outputs by $\hat{x} \approx (a - b) * W^T$: Original input data x (Left); Reconstruct from outputs of first hidden layer (Middle, the KL Divergence between x and $\hat{x}$ (1 st layer) are 2.32, 1.49, 1.51 for digit 1, 2, 6); Reconstruct from outputs of second hidden layer (Right, the KL Divergence between x and $\hat{x}$(2nd layer) are 2.63, 1.58, 1.56 which are higher than $\hat{x}$ (1st layer)).

without knowing the model parameters in the local device, the original input data cannot be reconstructed from the activation output.

- Even knowing the model parameters on the local device, a single rectifying layer is able to encrypt the input data. Better encryption can be done by applying more rectifying layers before transferring the activation outputs.

- Further encryption to the data can be done through Ramp activation function as it reduces the scale of the feature pattern.

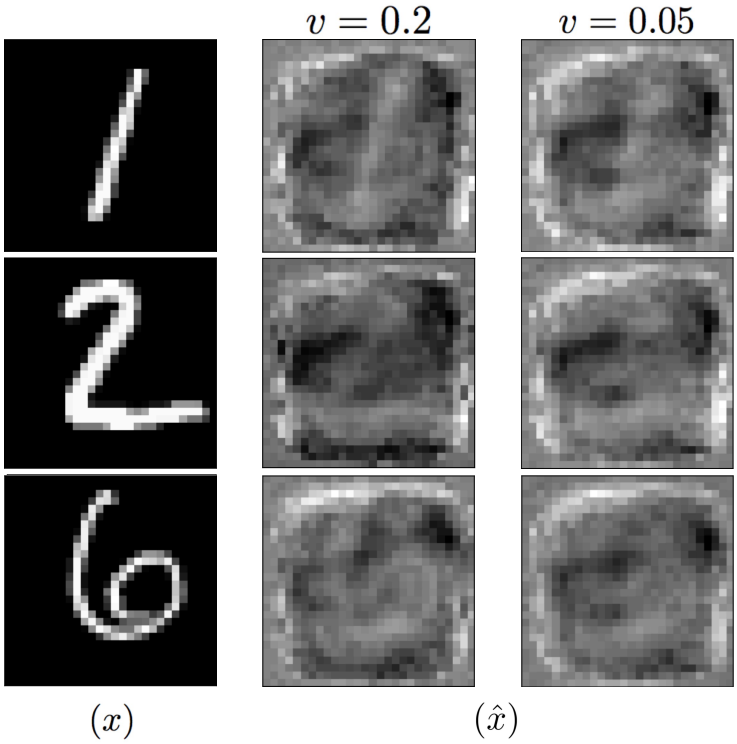


Figure 6.6: Reconstructed input data from ramp activation outputs by using $\hat{x} \approx (a-b) * W^T$ (Middle and Right) and Original input data x with $v = 0.2$(Left); The KL Divergence between x and $\hat{x}$ $(v = 0.2)$, which are 2.35, 1.54, 1.55 for digit 1, 2, 6, is smaller than that between x and $\hat{x}$ $(v = 0.05)$, which are 2.42, 1.63, 1.62.

6.4.2 Invertible Activation Functions - Dropping Activation Outputs

For invertible activation function, the first experiment was done on Dropping activation outputs. A network of 3 hidden layers of 800 units each is created, the activation function is sigmoid, which is invertible. All hyper-parameters and training methods are set to be the same as the previous rectifier neural network.

As a few of activation outputs are set to zero, the inverse sigmoid at Equation 6.5 becomes insolvable, since a number can not be divided by zero. An approximate reconstruction can be achieved by Equation 6.24 and 6.2, i.e. when a is out of range, the approximate

inverse number is set to zero [1].

$$f^{-1}(a) = \begin{cases} 0 & , when\ a \geq 0, a \leq 0 \\ -ln(\frac{1}{a} - 1) & , otherwise \end{cases} \tag{6.24}$$

Figure 6.7 shows the reconstructed input data under dropping probabilities p of 0.5%, 1% and 2%. It is clear that the reconstructed input data was distorted, and no further distortion was found as the dropping probability increase.

Therefore, dropping a few activation outputs is enough to encrypt the input data, and does not lead to noticeable performance decreases as Table 6.1 shows. In addition, adding some noise to a few activation outputs can also lead to the same result. If Equation 2.25 were used for reconstruction, the reconstructed input data will be show in Figure 6.8, which is worse than Figure 6.7.

For CIFAR10, a convolutional neural network (CNN) was used for this task. The network architecture is as follows: N64F5S1> Sigmoid> F3S2> LRN> N64F5S1> ReLU> LRN> F3S2> D384> ReLU> D192> ReLU> D10> Softmax, where N64F5S1 represents a CNN with 64 filters of size 5 and of stride 1, F3S2 represents Maxpooling with size of 3, stride of 2, and D10 represents fully-connective layer with unit of 10, LRN represents local response normalization. The model is trained by 50000 epochs with a learning rate of 0.0001 and Adam gradient descent. Activation outputs were randomly dropped on the 1st convolutional layer. The result can be seen in Table. 6.2. Through the single sample t-test, it shows that the accuracy in different p setting are not significantly different.

The experiments can be summarized as follow:

[1] Expect zero, the following constants have been explored: $\{-10, -1, -0.5, 0, 0.5, 1, 10\}$, we found them all giving similar results as Fig 6.7 shows.

Table 6.1: Randomly setting activation outputs to zero in first hidden layer during feed-forward propagation over 100 trails

p	Accuracy(%)	Standard Deriva-tion	Max/Min Accuracy (%)
0%	**98.7900%**	NA	NA
0.5%	**98.8026%**	0.000221	98.8600% / 98.7400%
1%	**98.8081%**	0.000294	98.8800% / 98.7300%
2%	**98.8093%**	0.000356	98.9200% / 98.7300%
3%	**98.7978%**	0.000373	98.8900% / 98.6900%
5%	98.7721%	0.000419	98.8700% / 98.6500%
10%	98.6991%	0.000500	98.8700% / 98.5700%

Table 6.2: Randomly setting activation outputs to zero in first convolutional layer during feed-forward propagation over 100 trails

p	Accuracy(%)	Standard Deriva-tion	Max/Min Accuracy (%)
0%	**84.5640%**	NA	NA
0.1%	**84.5641%**	0.000409	84.6800% / 84.5100%
0.5%	84.5535%	0.000635	84.7800% / 84.4800%
1%	84.5313%	0.000787	84.7000% / 84.3500%
2%	84.4696%	0.001037	84.7100% / 84.1600%
3%	84.4180%	0.000859	84.2300% / 84.7500%
5%	84.3613%	0.001230	84.6800% / 84.0900%

- For invertible activation functions, dropping a few activation outputs during feed-forward propagation can provide data encryption and privacy.
- To make the function insolvable, the activation value needs to be set out of its reasonable range, for example, setting the value out of $(0, 1)$ for sigmoid, and $(-1, 1)$ for the hyperbolic tangent.
- Instead of setting activation outputs out of its reasonable range, adding small noise

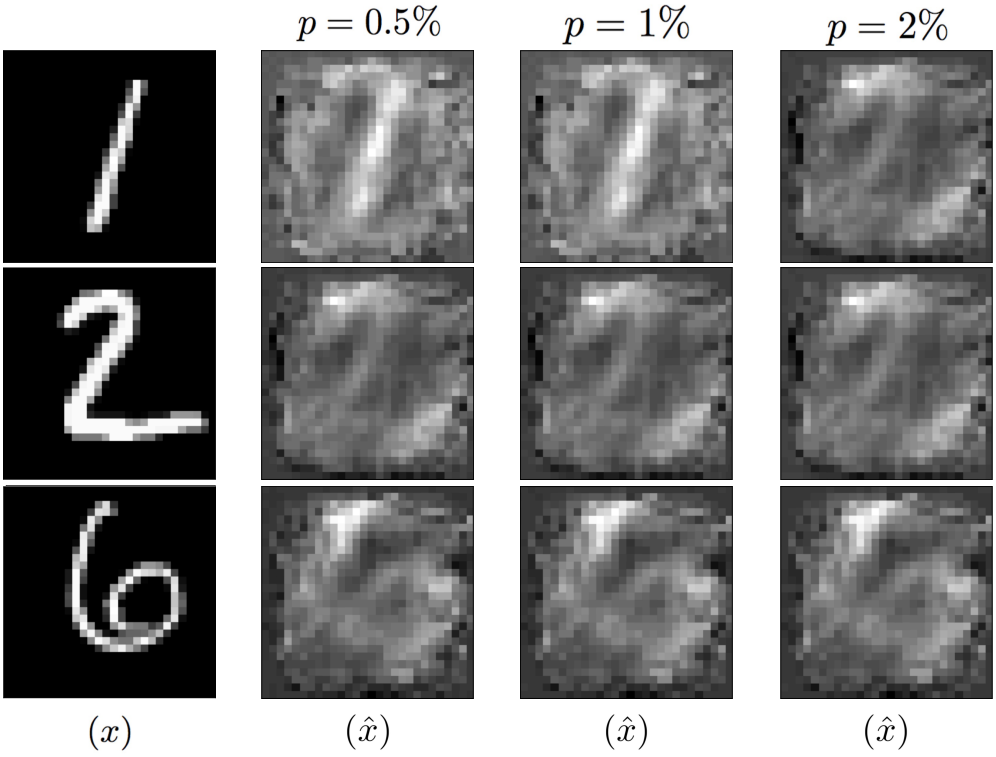


Figure 6.7: Equation 6.24 and 6.2 are used to reconstruct input data from first sigmoid hidden layer after dropping activation outputs. The first column is the original input data x; the dropping probabilities increase from 0.5% to 2% from second to forth column. Note that 0.5% is only 4 neurons in the layer of 800 neurons. The KL Divergence between x and $\hat{x}(p = 0.5\%)$ are 2.03, 1.34, 1.35 which are similar with $\hat{x}(p = 1\%)$ (2.02, 1.35, 1.35) and $\hat{x}(p = 2\%)$ (2.04, 1.35, 1.36).

values to few activation outputs can have a similar impact.

6.4.3 Invertible Activation Functions - Dropping Connections

Randomly setting a part of weight values to zero can indirectly modify the activation outputs. However, according to the experiment using the MNIST dataset, when combining sigmoid function and dropping activation outputs during the feed-forward process, encryption does not appear, and the reconstructed data is almost the same with the original input data.

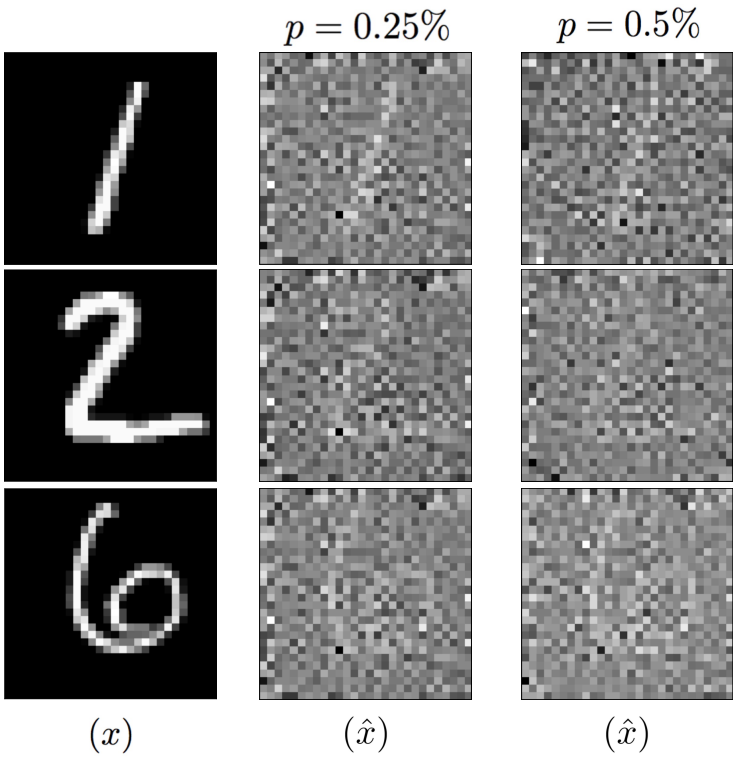


Figure 6.8: Reconstructed input data from first sigmoid hidden layer after drop activation outputs by using Equation 6.24 and 2.25. Original input data x (The first column); From the second to third columns, the dropping probabilities increase from 0.25% to 0.5%. The KL Divergence between x and $\hat{x}(p = 0.25\%)$ are 2.34, 1.51, 1.45, and $\hat{x}(p = 0.5\%)$ are 2.49, 1.52, 1.46, which are higher than Fig.6.7

6.4.4 Autoencoder

Reducing the number of activation outputs can reduce both computation and communication cost for the local device. In that case, an Autoencoder can be applied.

Even for invertible activation function, when the number of activation outputs is smaller than the number of input data, the original input data cannot be reconstructed correctly from its activation outputs, as it uses smaller dimension to represent the original data. The input data can be approximately reconstructed by Equation 6.5.

Figure 6.9 shows the input data, which is reconstructed from Autoencoder with sigmoid

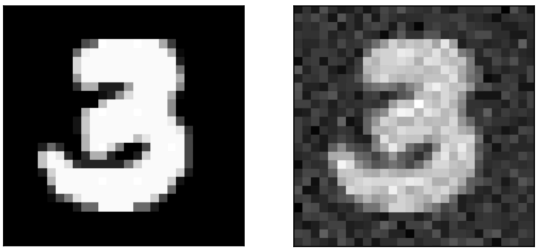


Figure 6.9: Reconstructed input data from sigmoid outputs by using Equation 6.5 (Right); Original input data x (Left). The number of neuron is the half of the number of input data.

activation function. In this case, the number of neurons is half of the number of input data. In other words, half of the information of original input data can be reconstructed, noting the shape of the digit image can still be identified clearly. Therefore, an Autoencoder with non-invertible activation function or dropping activation output is recommended, so as to provide both data encryption and data compression.

6.4.5 Privacy Preserving

In this section, the privacy-preserving property of the above method is evaluated by discussing how it works in a brute-force attack and honest-but-curious model.

Assume M neurons are dropped. As discussed before, the computation raises exponentially; the attackers have to pre-compute possible activation results. Take a sigmoid function as an example, as the outputs of neurons are continuous values, and assume the attacker pre-defines N possible output values, and the algorithm drops M neurons of the outputs, the number of possible combinations is N^M. In the sigmoid experiment when dropping probability is 0.5%, M is four if N is defined to be 100, the number of possible combinations is 100 million, after reconstructing 100 million images, the attacker also needs an algorithm to recognize the "good" image. If a rectifier is used, the attacker would not know where and how many neurons are dropped by this method, which means

the number of possible combinations will become significantly larger.

This can be illustrated by conducting a brute-force experiment on MNIST classifier with sigmoid function. There are 800 outputs, M is 4 (0.5%). These are separated [0, 1] to 101 gaps, i.e. from 0, 0.01, 0.02 to 1. Therefore, there is 104060401 possible input. It takes 5083 hours to compute all possibilities on a Titan X Pascal GPU. Note that, the attacker needs an extra algorithm to select one data point from 104060401 data, but as the attacker do not know the content of data, it is difficult to define the criteria.

In the honest-but-curious model, for invertible activation function, the intact activation outputs can be obtained if local device inputs the same data into the network for multiple times, and the dropping positions do not overlap. The probability of the dropping position does not overlap on second time with the first time is C_{N-M}^{M}/C_{N}^{M}. As M is relatively smaller than N, so the original input data has a high probability of being decrypted if the data is input to the network twice.

As a result, in practice, to prevent such a situation, the same positions need to be dropped if the data is the same. The dropping position, therefore, should be deterministically generated from the data, for example, by using a max value in the data as the random seed to select the dropping position.

It is important to mention that, for non-invertible activation function, as the activation outputs are dropped inherently, even without using dropping activation outputs, the honest-but-curious model can not work.

6.4.6 Speed

To verify the impact of implementing Sigmoid function on user local devices, a system test is conducted to measure it. The Metal framework is used[1] to develop the Sigmoid

[1] https://developer.apple.com/metal/

function on GPU (in Swift language):

```
kernel void sigmoid(const device float
  *inVector [[ buffer(0) ]],
  device float *outVector [[ buffer(1) ]],
  uint id [[ thread_position_in_grid ]]) {
    outVector[id] = 1.0 /
    (1.0 + exp(-inVector[id]));
  }
```

The device used is an iPhone 6 with iOS 10. $NSDate()$ was used to measure the time-lapse of computing the first layer activations. The input used is a vector x where $|x| = 40000$ (200×200 pixels grayscale images), and $Sigmoid(W\dot{x} + b)$ is applied with W (size 40000×1000) and b (size 1000) for 1000 neurons. Below is a sample output of time lapses measured, **1300ms** is calculated to be the average latency induced.

```
D/Sigmoid(601): onActivation: begin
D/Sigmoid(601): onActivation: 1311 ms
D/Sigmoid(601): onActivation: end, 1311 ms
```

6.5 Conclusion

In the above work, a new architecture for a deep network has been proposed, with the localized first layer of the network. How the architecture could support better privacy protection in model prediction is investigated. Invertible activation function through Dropping activation outputs during feed-forward propagation was proven to be able to encrypt the original input data and preserving privacy. The whole encryption process can be improved through combining feed-forward propagation and data encryption into

one process, which means no need for a specialized data encryption process on the local device, nor data decryption process on the server.

During the encryption process, both invertible and non-invertible activation functions have been discussed and mathematically proved possible to do encryption. In error back-propagation, splitting the neural network into a local device and server can provide data privacy during training. In other words, the server is able to provide model learning service by using error backpropagation, without accessing the original input data from the local device.

This process can be applied to the financial industry for those indices that are calculated uses sensitive private datasets. However, this is a general method, and it can be applied to any type of dataset, which means any finance-related data can also use the method.

Chapter 7

Conclusion

7.1 Summary of Thesis Achievements

The achievement of this thesis is several-fold.

First, this thesis builds on a deep learning model to achieve trading strategies that use the volume to predict the stock price or stock price changes. This thesis describes the data selection, refinement and model selection process, as a general strategy that can be used to build any other similar models. The model itself can also be modified based on the length of the asset holding time. Therefore, the return of the stock price can be predicted on the time scale that an asset is held.

The second achievement comes from building a model to explore the pattern in stock price and to test the model during the acquisition period. Not much research exists in the literature due to the challenge of building the dataset; it should be noted that Crunchbase[1] data which was used in this thesis did not exist prior to 2015, so is a relatively new resource

[1] A website that provides technology companies data and information, website is *https* : *//www.crunchbase.com/*

to collect data from.

The third achievement comes from the general, saleable, and unbiased advertisement impact assessment algorithm. By doing so, the analysis can be more detailed, and in the design of advertisements in the future can be guided.

The fourth achievement is migrating the first layer of a deep network to where the data originates (which may be on a local device remote to where the deep learning is being run), applying the activation functions locally, and then use the "dropping activation output" method to make the output non-invertible. The resulting approach is able to make model predictions without accessing sensitive raw data. In doing so, this method may help satisfy requirements of data privacy laws such as GDPR as the resultant data received by the deep learning is not identifiable, yet still valuable for the learning.

7.2 Applications

The most straightforward application is to apply the model as a trading algorithm to help traders and institutions make trading decisions. The decision would be able to make decisions like "at what time to buy, at what time to sell, and at what valuation of the acquisitions, what profit margin to make to take advantage of the merger and acquisition events".

The advertising effect assessment is straightforward and useful to the advertising agencies who can use this information to tailor their advertising to target a specific customer group, therefore, boosting the products/service sales numbers.

The encrypt method provides an option to encrypt the raw dataset no matter what the dataset format is.

7.3 Future Work

Future work can go in two directions. The first is to explore the physical significance of problems that could be solved. The second is to explore the methods that are used to solve these problems.

In the current research, the data collected were for technology companies. A greater variety of companies can be added into the data to expand the model to be more generally applicable, for example, from other industries like semiconductors, clothes, oil and gas and etc.

In the merger and acquisition model, the model was built against a specific company being acquired if the valuation is available on Crunchbase. However, the M&A can be analyzed based on the year it happens and then comparing the difference annually. The regression can be used to fit the model based on the acquisition valuation.

Many cross-section research has been done on option volume and the stock price. For example, in 2009, Richard Roll has worked on a paper about the trading volume ratio on option and stocks,i.e. O/S [RSS10]. The information content of option and equity volume ratio and the future returns have also studied by Travis L. Johnson in 2011 [JS12]. Moreover, Jun Pan published a paper studying the information in option volume for the future stock prices in 2006, which is published by MIT [PP06]. Based on all of this research, machine learning methods can be used to build to study and model the options volume and the stock price.

Patrick Augustin and Mennachem Brenner did research in 2016 on informed options trading prior to takeover announcements. In this paper, abnormal option volumes were used as an indicator of inside-trading before a takeover announcement [ABS19]. Guo and Chen have done work on the informational content of option volume prior to takeovers

[CCG05] [STS11]. David Easley also was able to find an indication of informed traders through option volume and stock prices in 1998 [EOS98]. Therefore, inside trading could be something that can be detected by a model like the one presented in this research too.

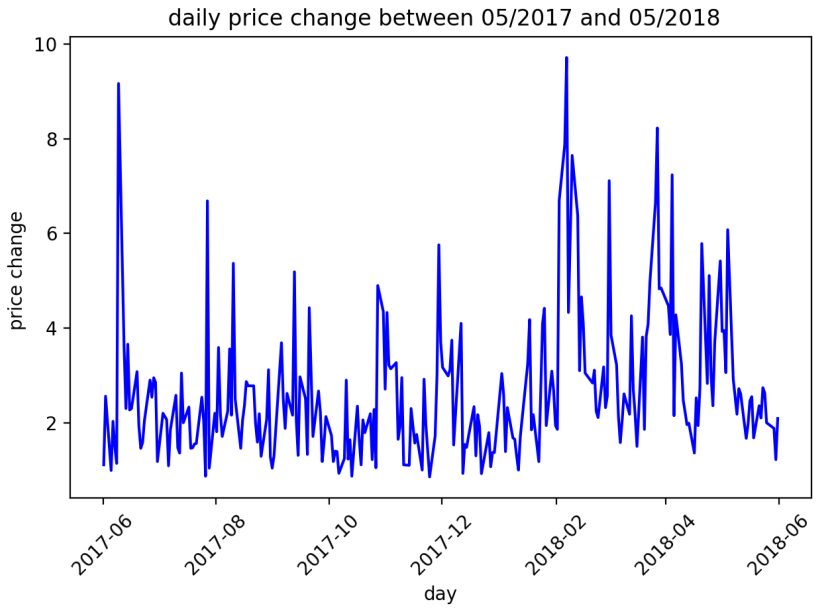
As has been studied in the paper called "inside trading in takeover targets", other algorithms can be applied to the takeover targets, this includes stock return change of the target companies after the announcement has been made [AN12]. The challenges here is that many of the companies are privately traded and do not release enough public data to be used to make predictions.

In the advertising impact assessment chapter, the method used can be considered a relatively basic machine learning model. Further work could be done to extended this research to use deep learning methods if a larger dataset could be collected. If more data can be collected based on more targeted objects or targeted groups of people, better and more accurate analysis can be made.

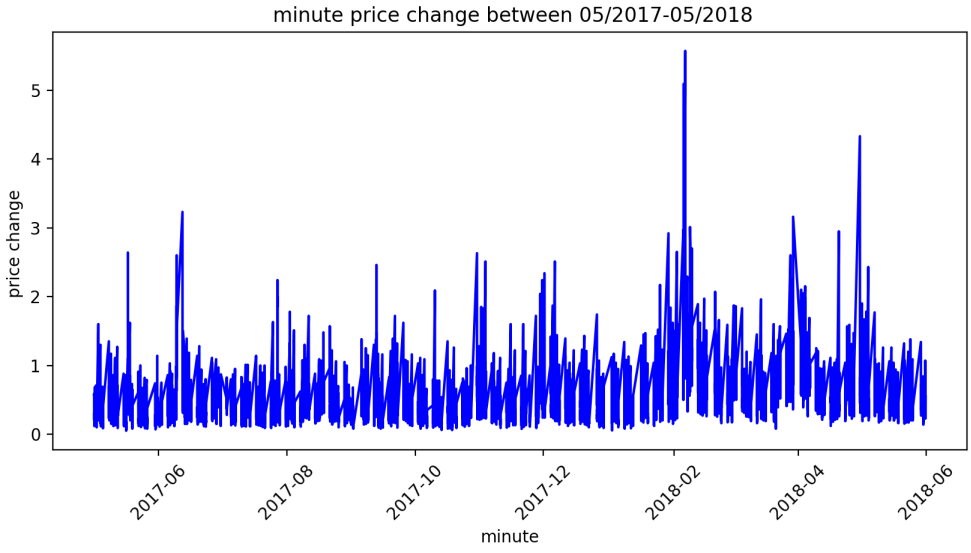
Currently, statistically significant numbers have been done through ANOVA, and t-test in MAPE and accuracy prediction, other quantitative evaluation can also be carried out for the experiments in each chapter.

Appendix A

Figures

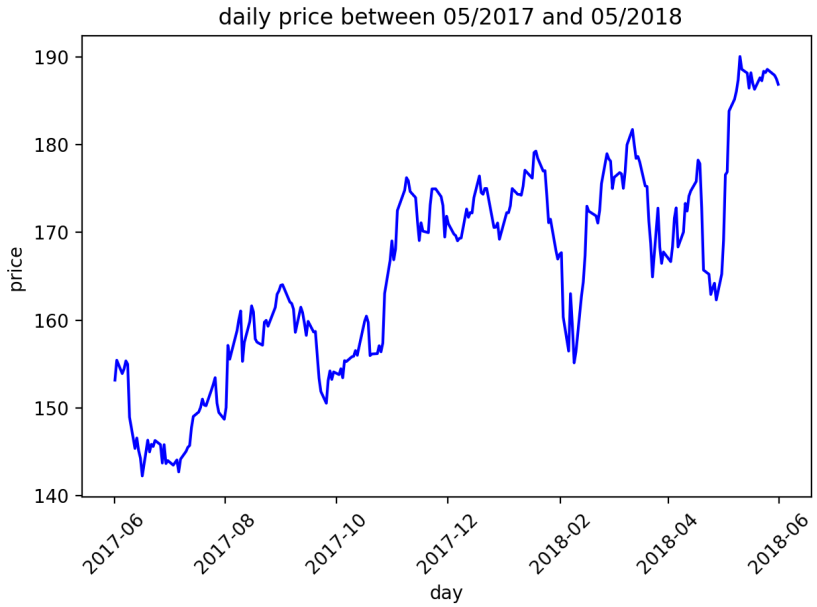


(a) Daily price change

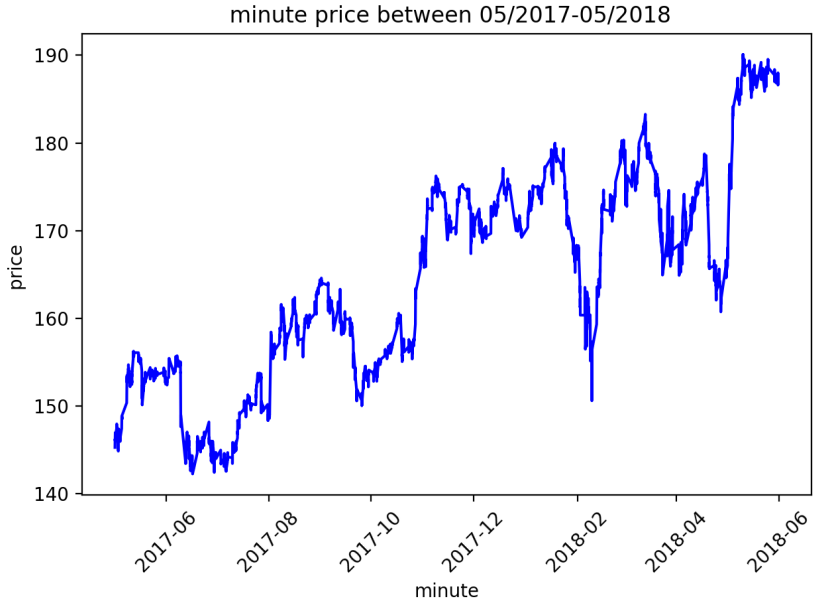


(b) Minute price change

Figure A.1: Apple Daily and Minute data of price change between 05/2017-05/2018

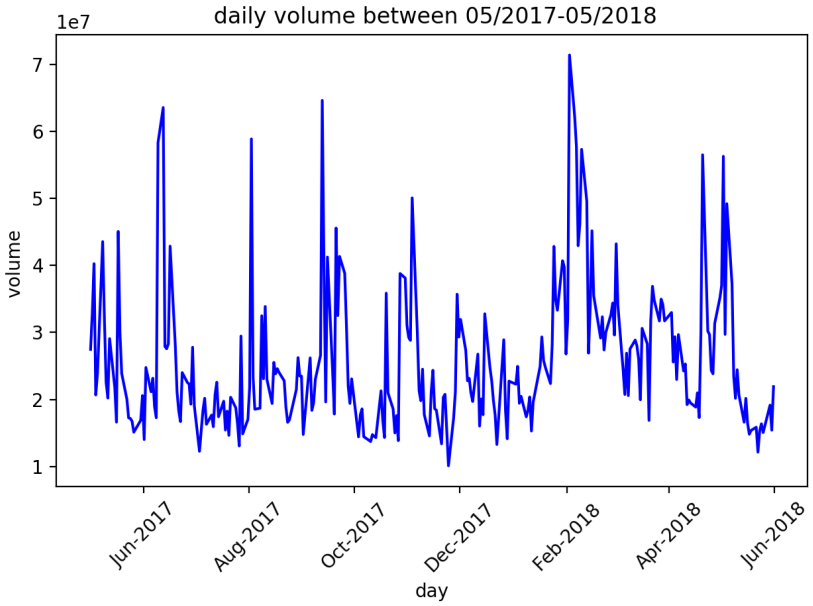


(a) Daily close price

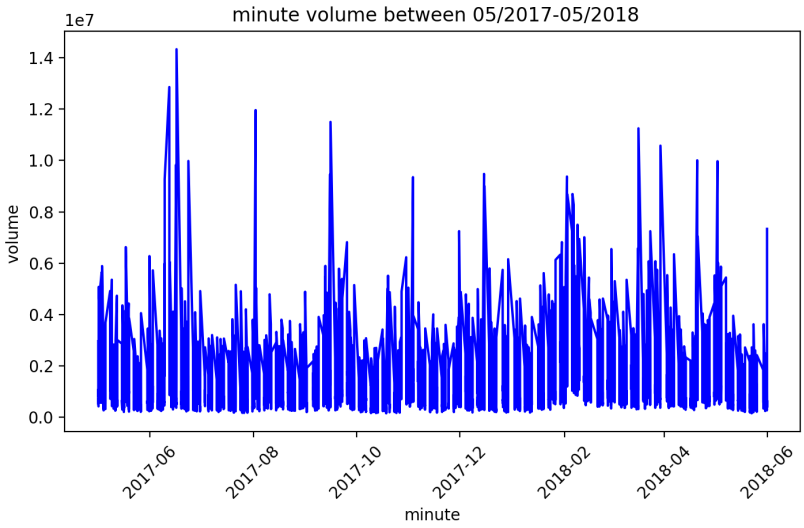


(b) Minute close price

Figure A.2: Apple Daily and Minute data of close price between 05/2017-05/2018

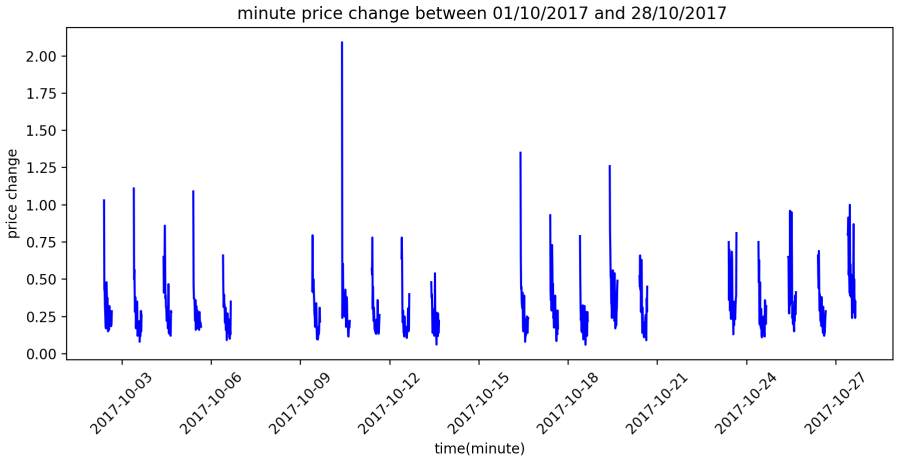


(a) Daily volume

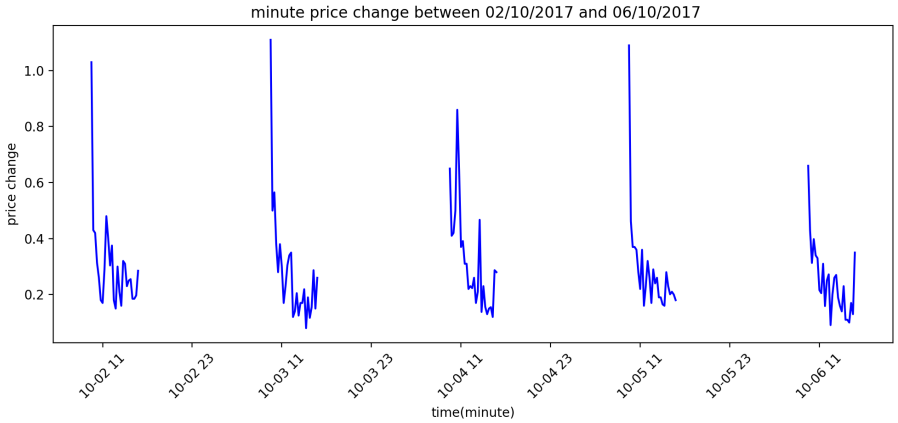


(b) Minute volume

Figure A.3: Apple Daily and Minute data of volume between 05/2017-05/2018

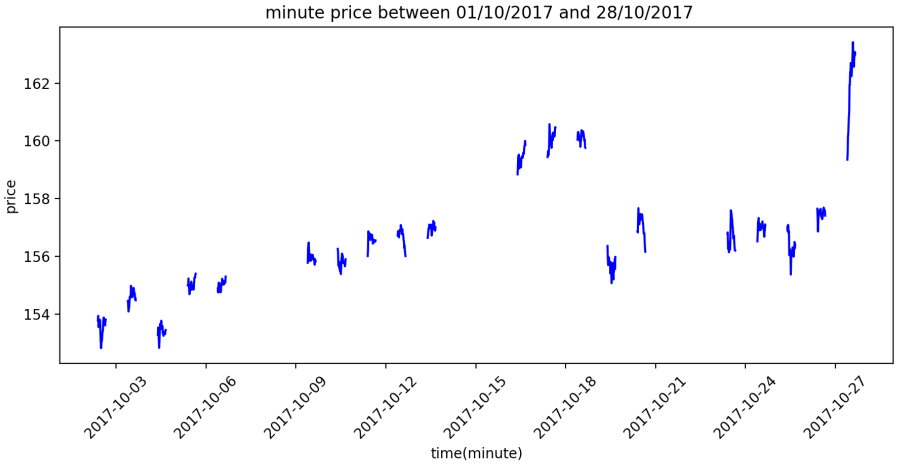


(a) Monthly price change

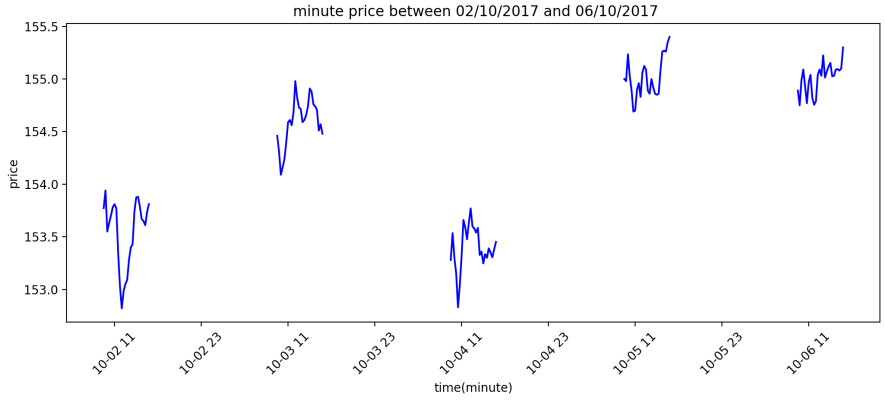


(b) Weekly price change

Figure A.4: Apple monthly and weekly data of price change in October 2017

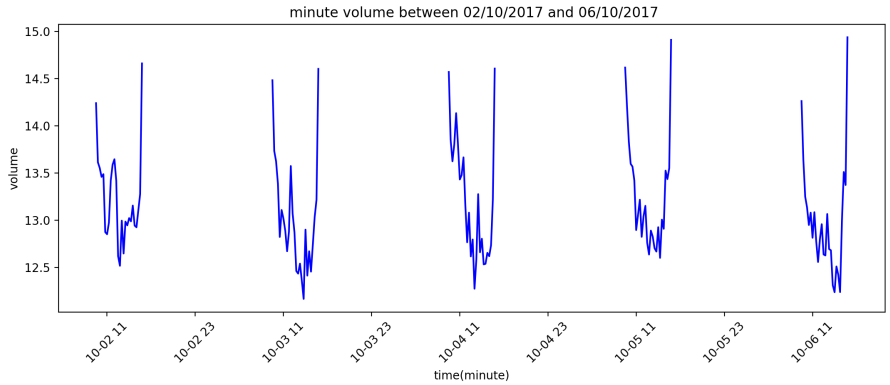


(a) Monthly close price

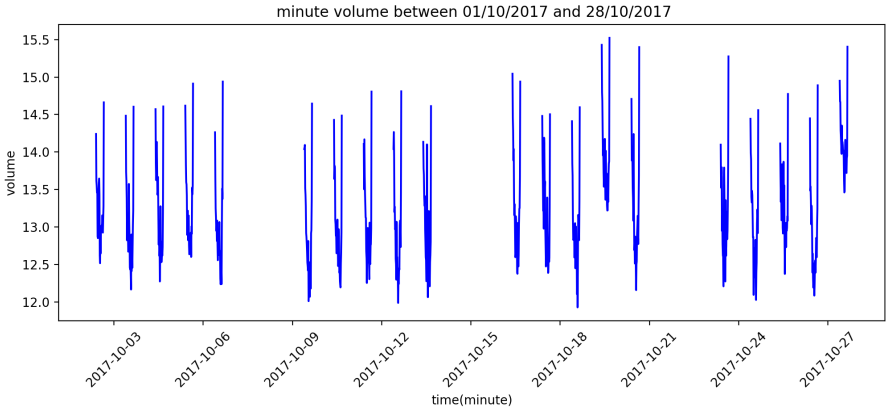


(b) Weekly close price

Figure A.5: Apple monthly and weekly data of close price in October 2017

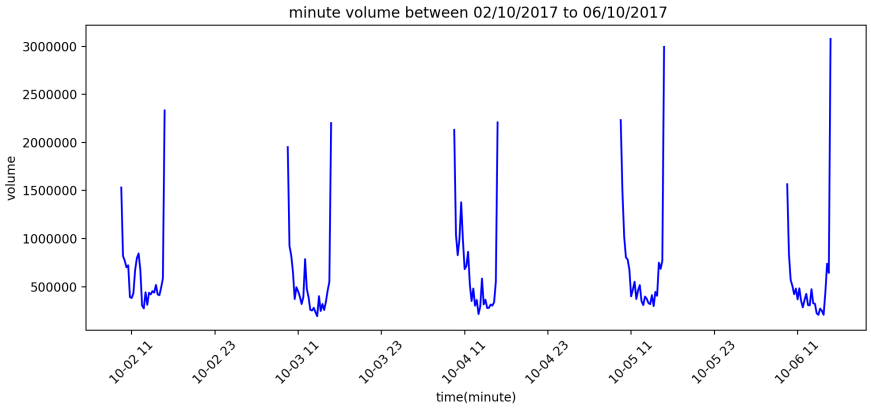


(a) Weekly log(volume)

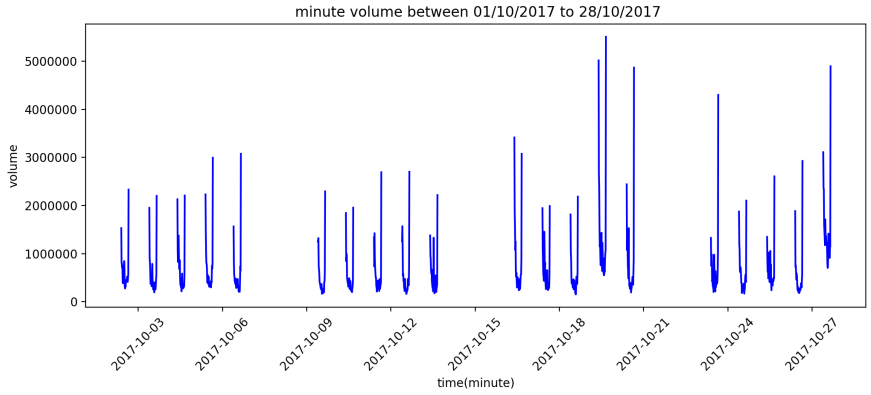


(b) Monthly log(volume)

Figure A.6: Apple monthly and weekly data of log volume in October 2017

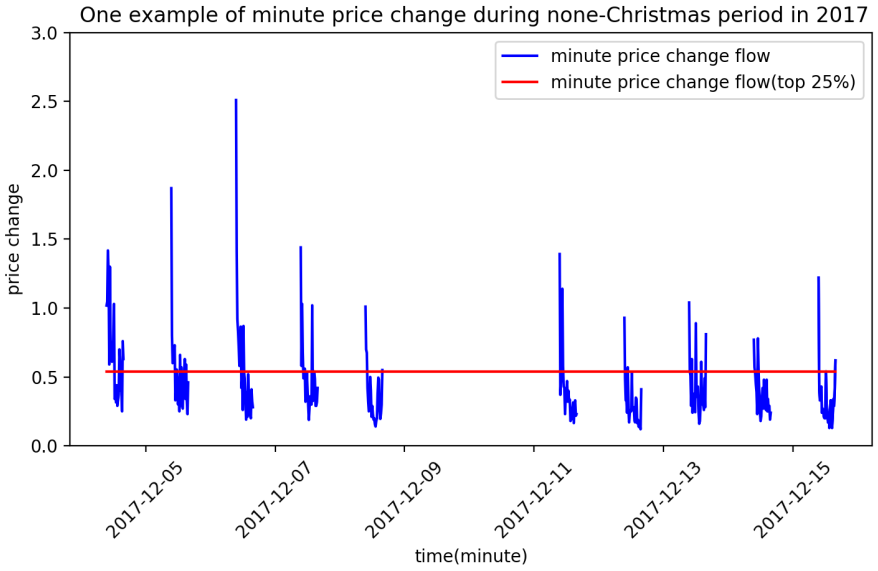


(a) Weekly volume

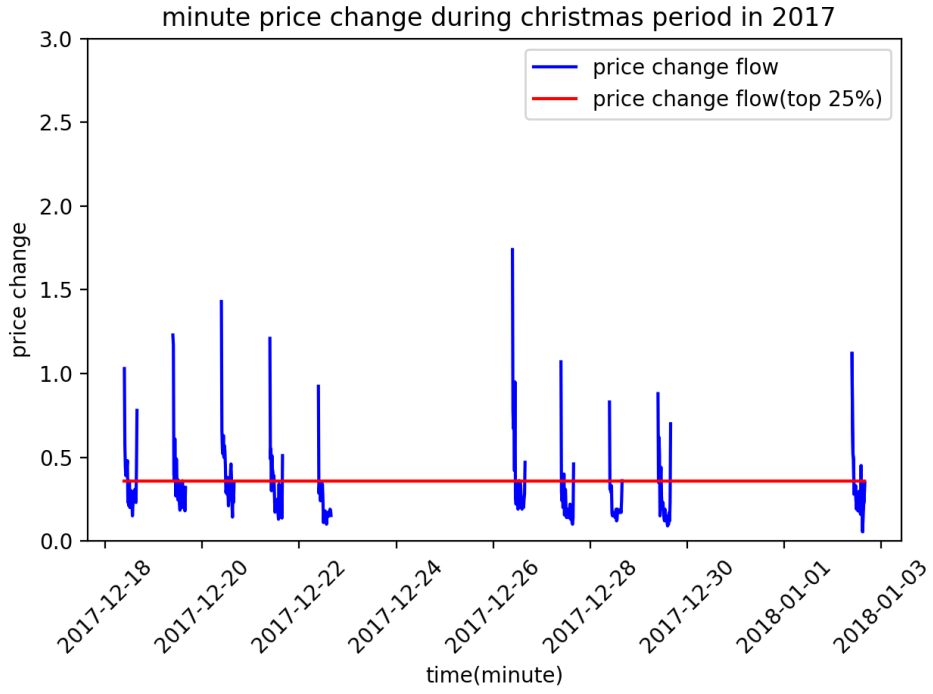


(b) Monthly volume

Figure A.7: Apple monthly and weekly data of volume in October 2017

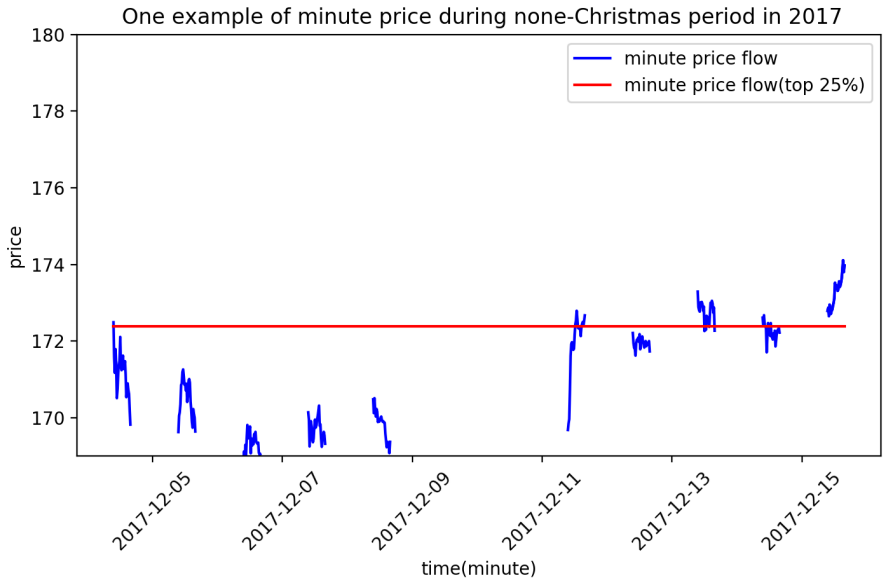


(a) price change

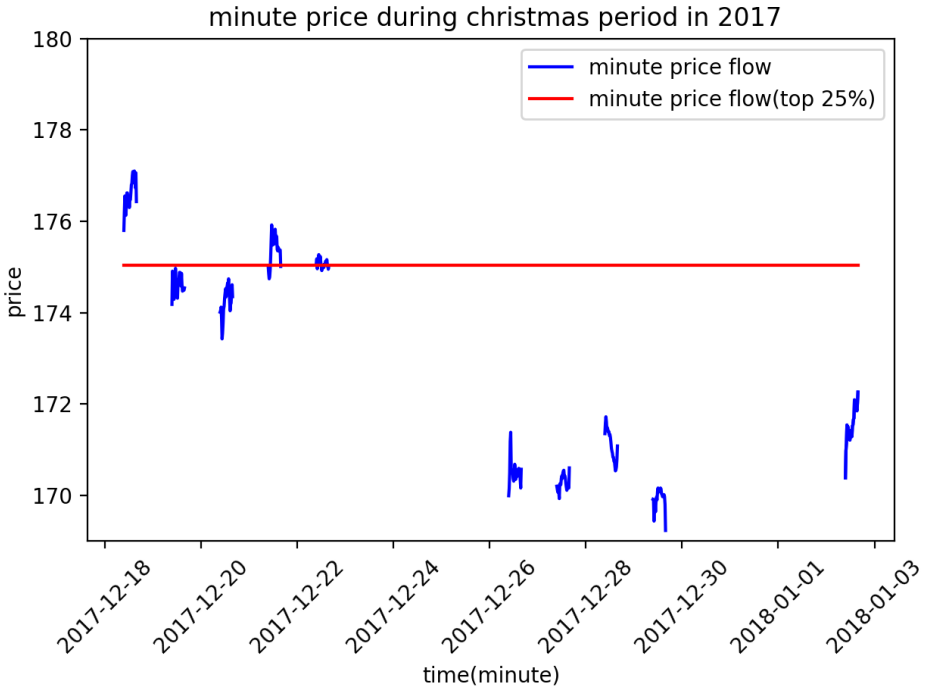


(b) Christmas period price change

Figure A.8: Christmas period and non-Christmas period Apple data of price change in October 2017

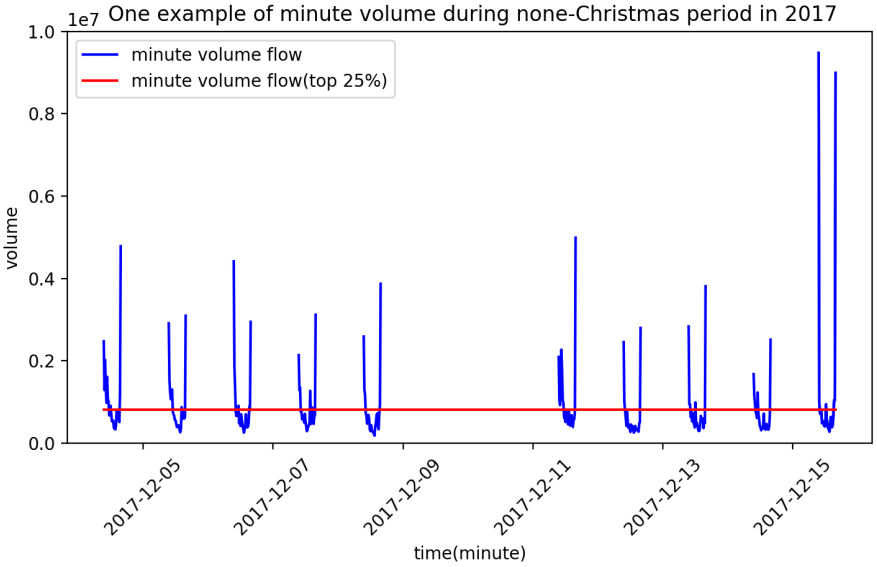


(a) close price

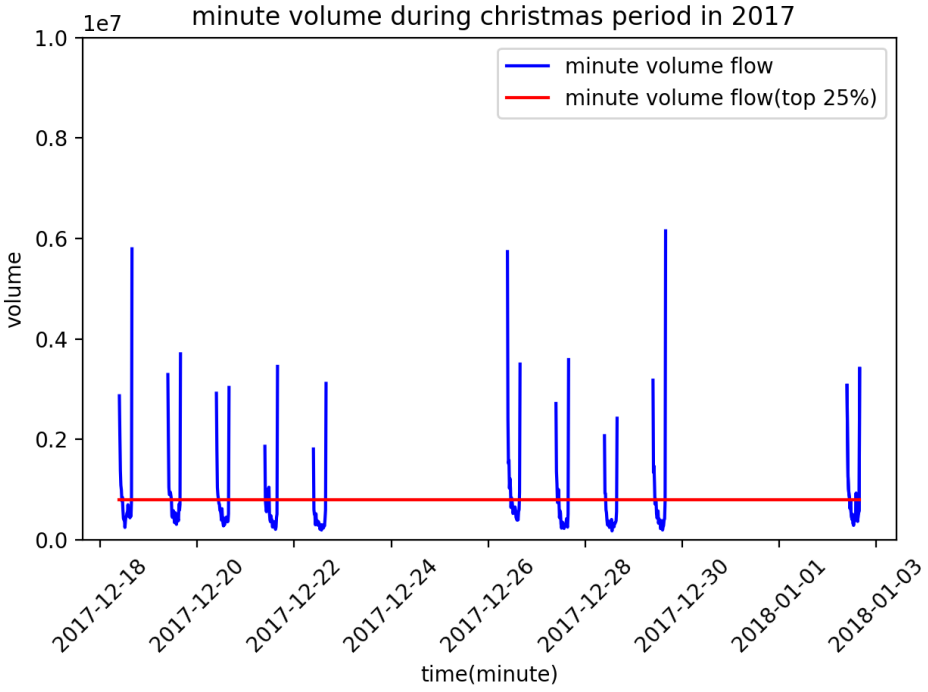


(b) Christmas period close price

Figure A.9: Christmas period and non-Christmas period Apple data of close price in October 2017

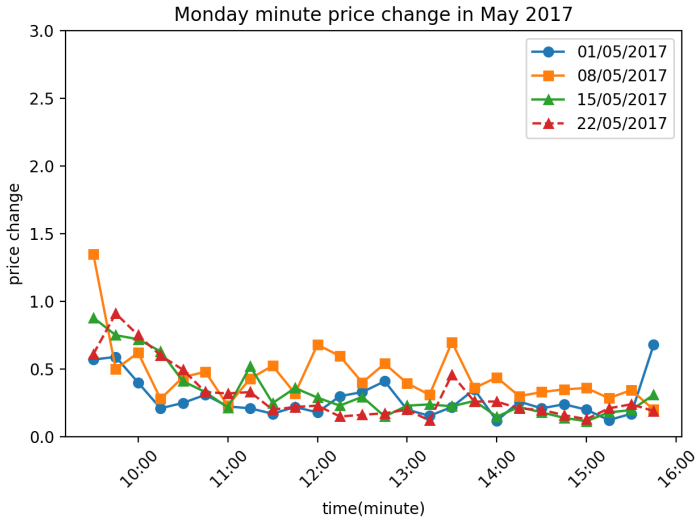


(a) volume

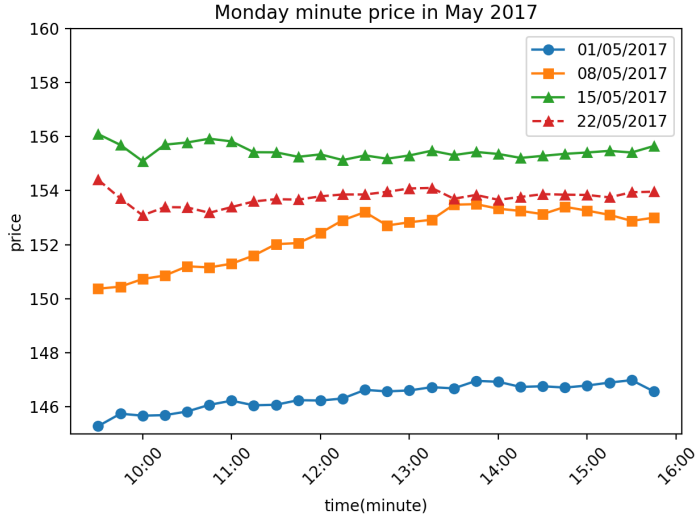


(b) Christmas period volume

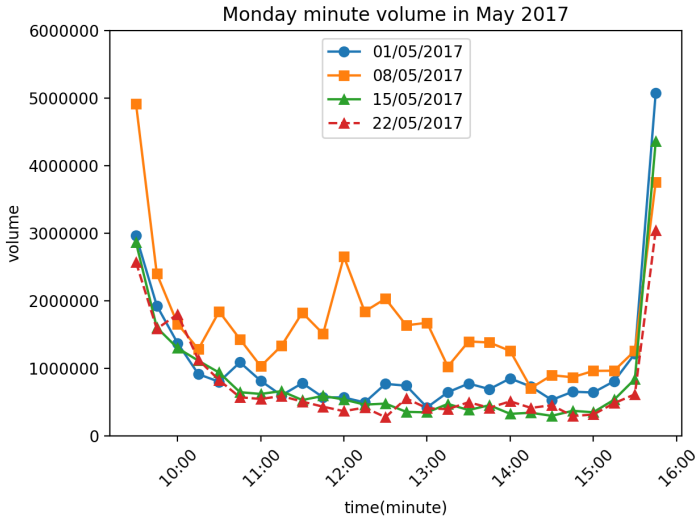
Figure A.10: Christmas period and non-Christmas period Apple data of volume in October 2017



(a) Monday price change

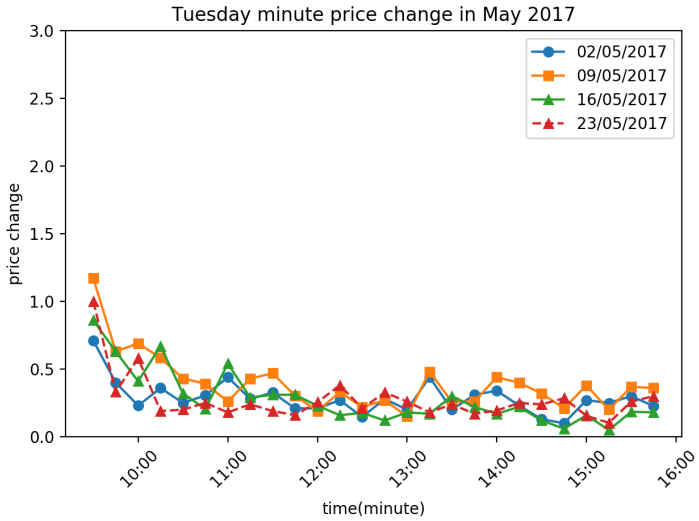


(b) Monday close price

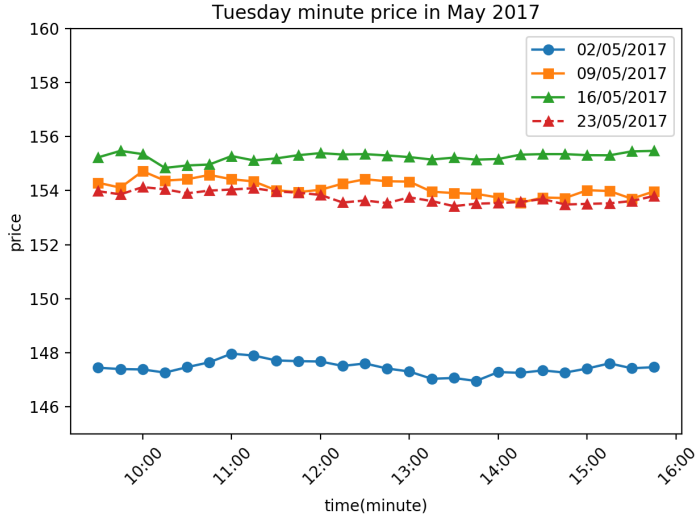


(c) Monday volume

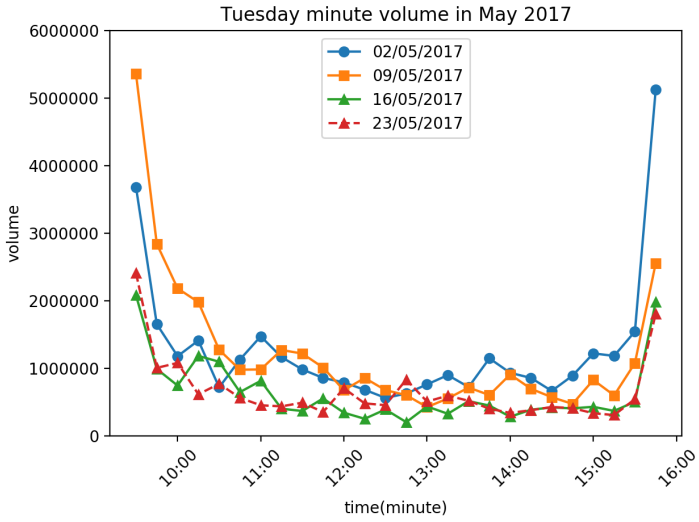
Figure A.11: Apple weekday data of price change, close price and volume in May 2017



(a) Tuesday price change

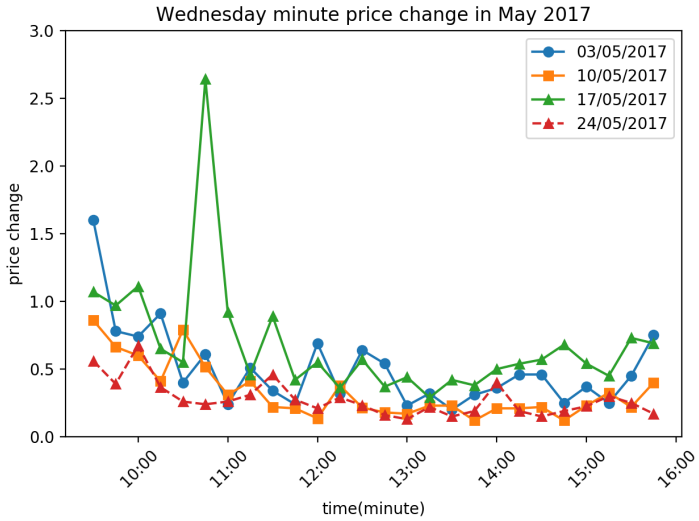


(b) Tuesday close price

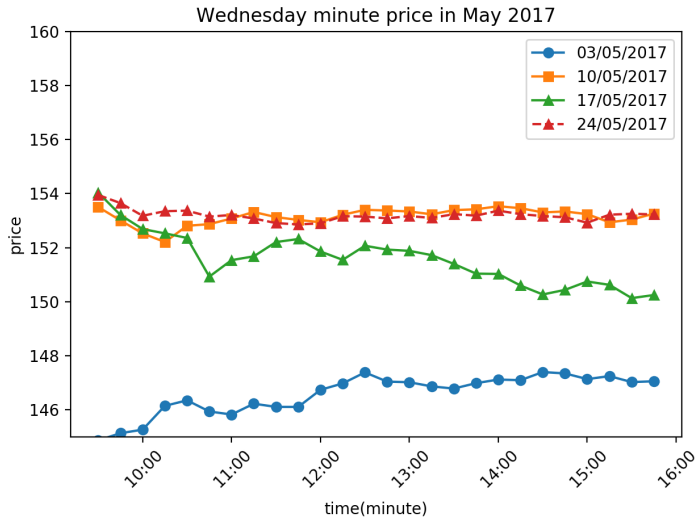


(c) Tuesday volume

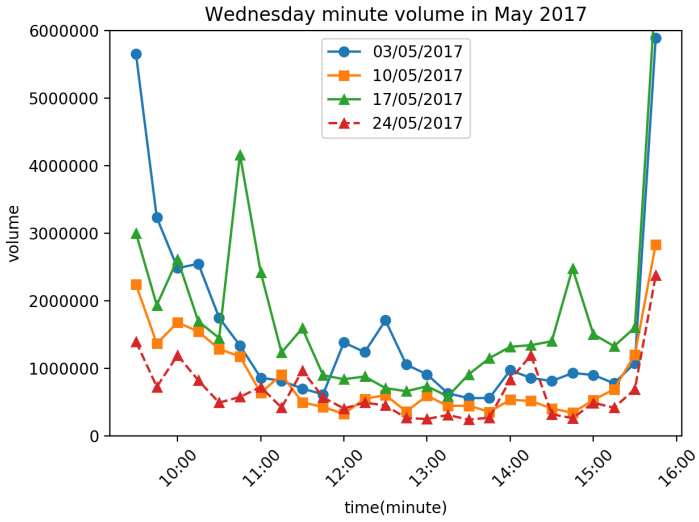
Figure A.12: Apple weekday data of price change, close price and volume in May 2017



(a) Wednesday price change

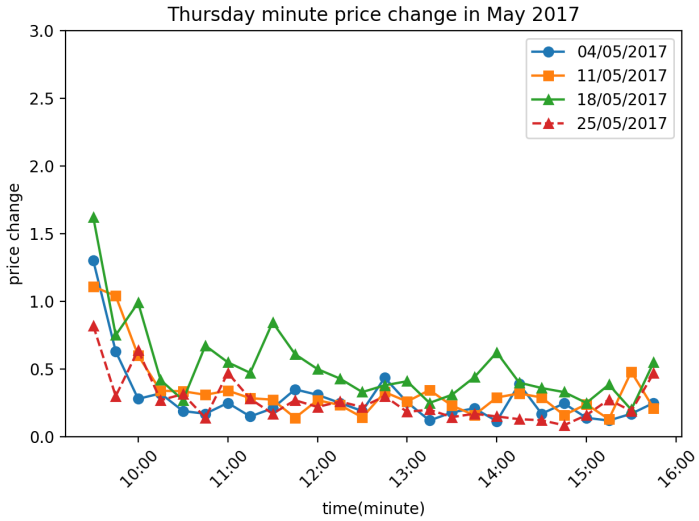


(b) Wednesday close price

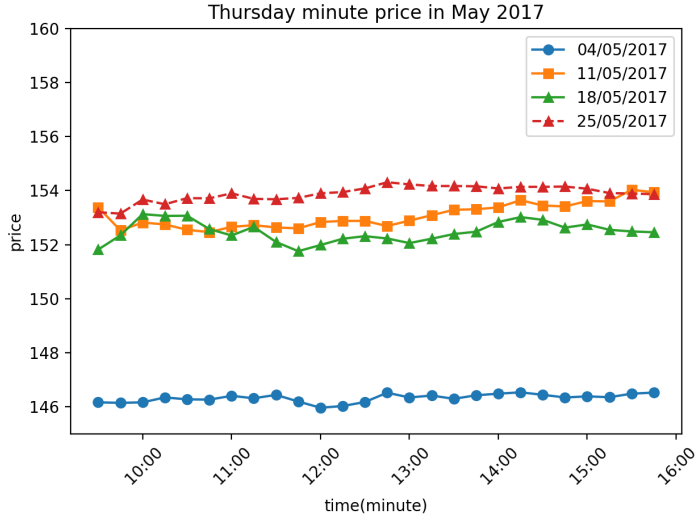


(c) Wednesday volume

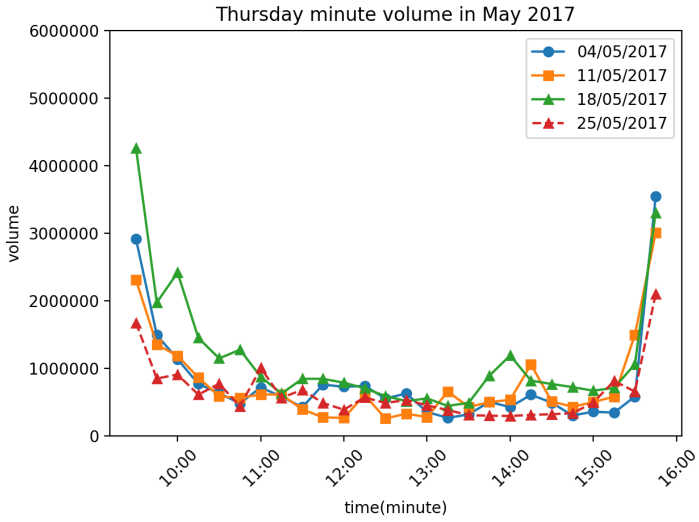
Figure A.13: Apple weekday data of price change, close price and volume in May 2017



(a) Thursday price change

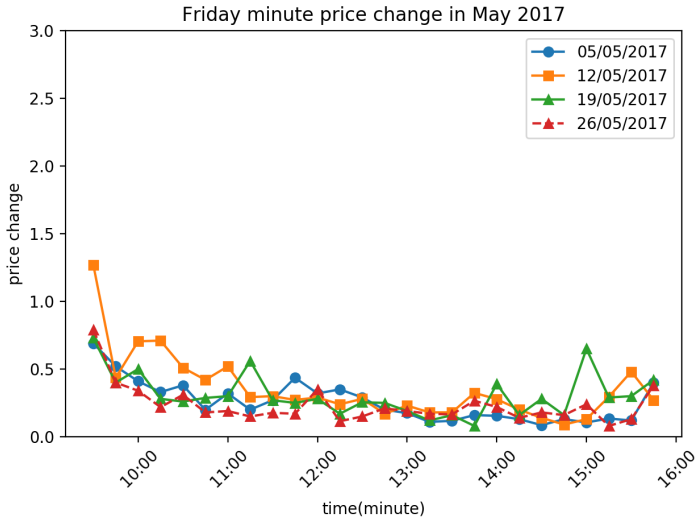


(b) Thursday close price

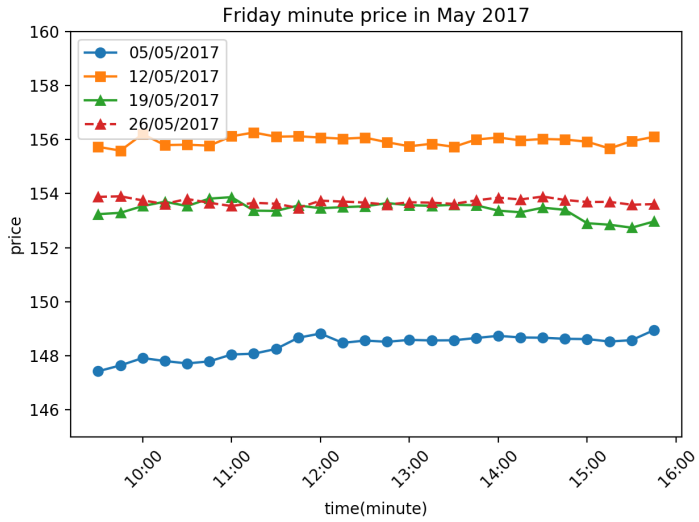


(c) Thursday volume

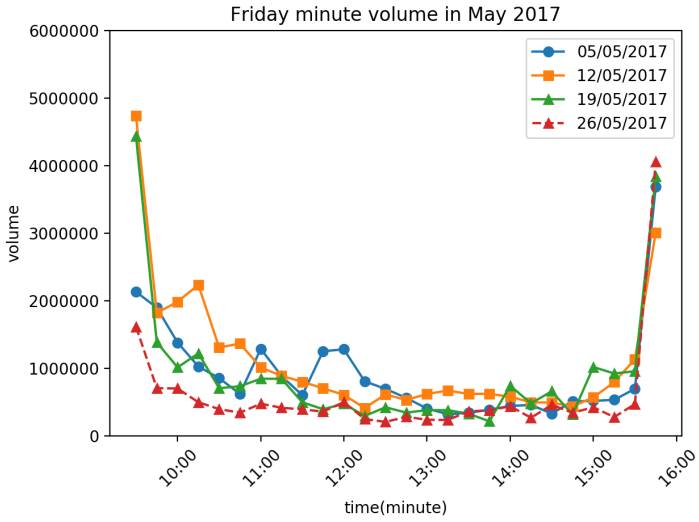
Figure A.14: Apple weekday data of price change, close price and volume in May 2017



(a) Friday price change

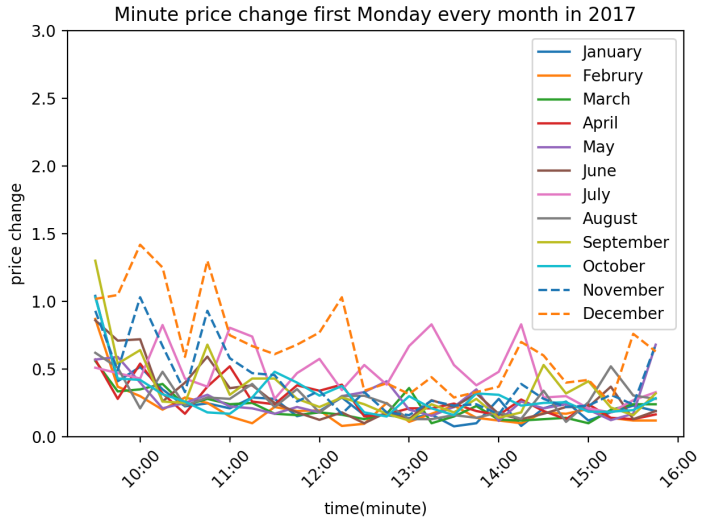


(b) Friday close price

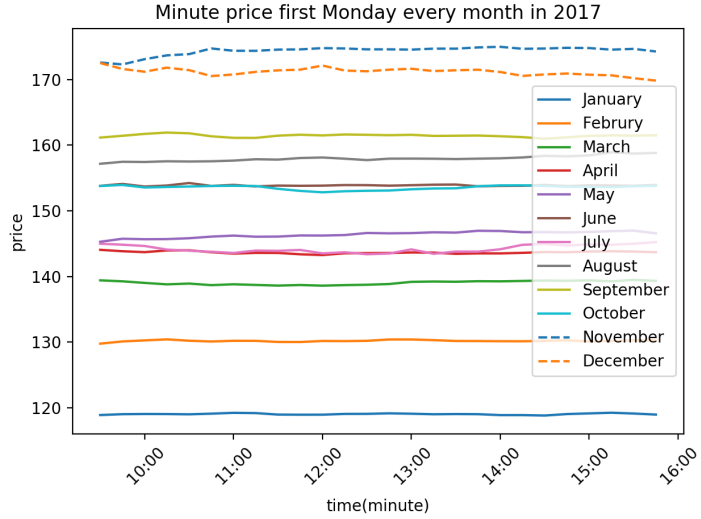


(c) Friday volume

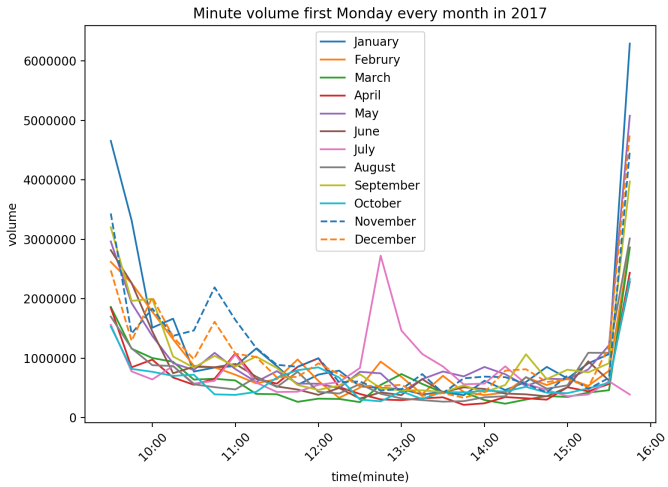
Figure A.15: Apple weekday data of price change, close price and volume in May 2017



(a) First Monday price change every month

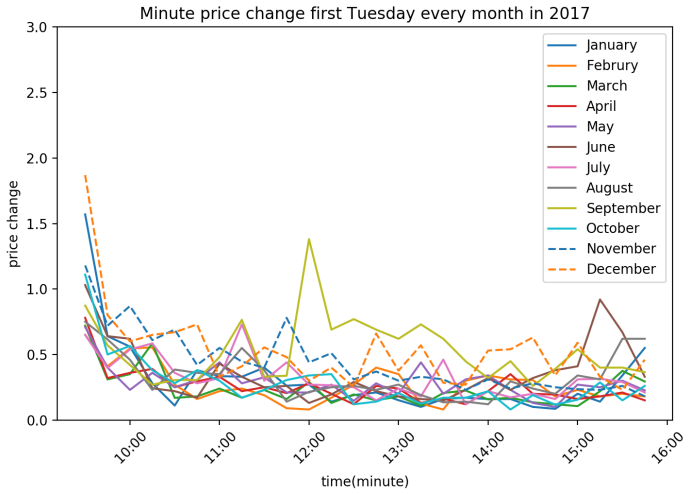


(b) First Monday close price every month

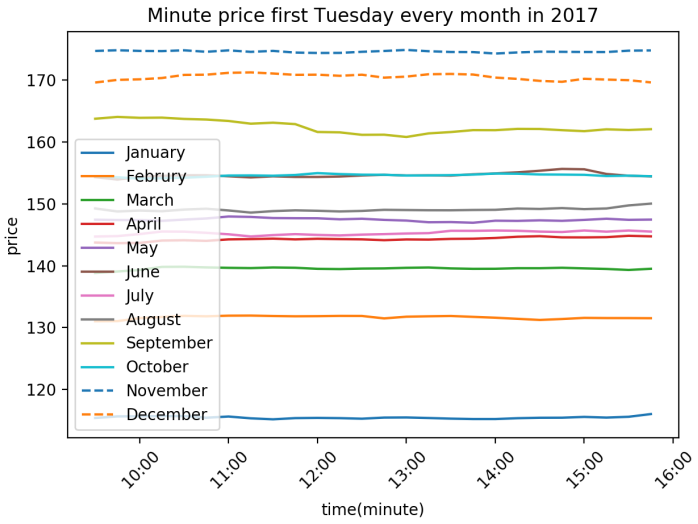


(c) First Monday volume every month

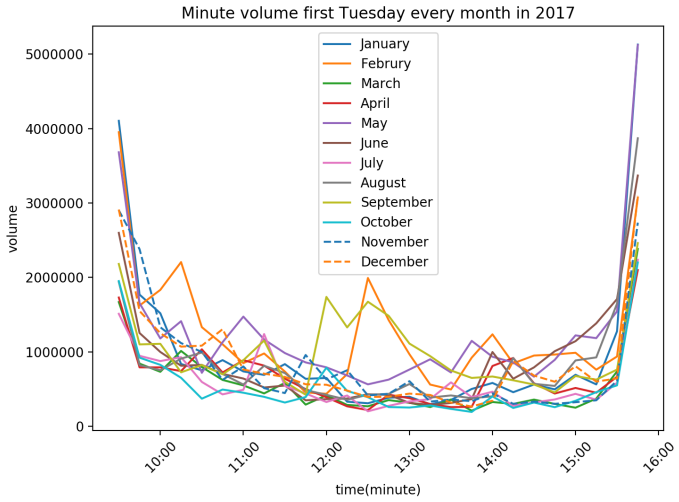
Figure A.16: Apple weekday data of price change, close price and volume each month in 2017



(a) First Tuesday price change every month

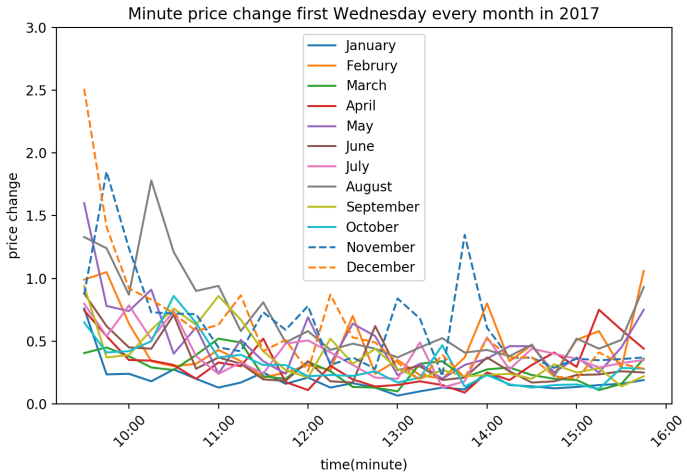


(b) First Tuesday close price every month

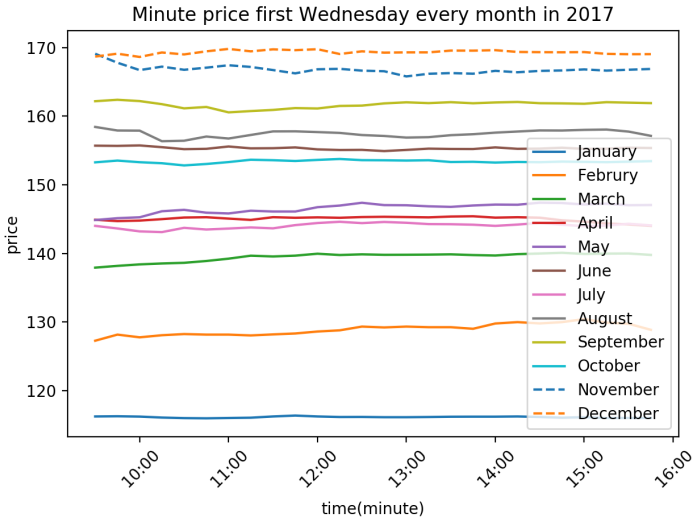


(c) First Tuesday volume every month

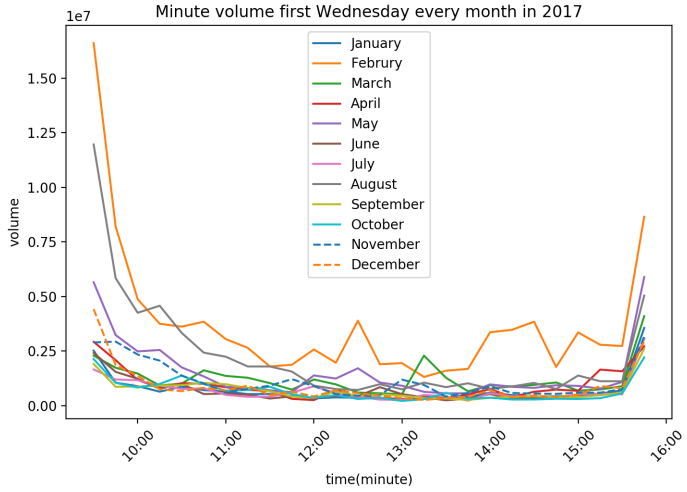
Figure A.17: Apple weekday data of price change, close price and volume each month in 2017



(a) First Wednesday price change every month

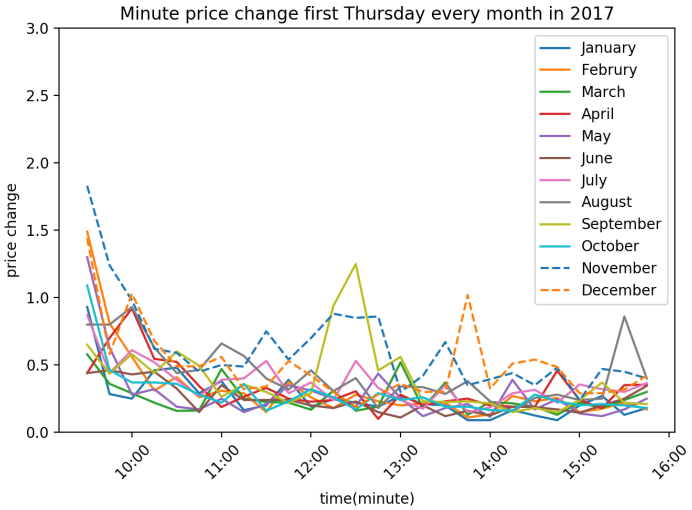


(b) First Wednesday close price every month

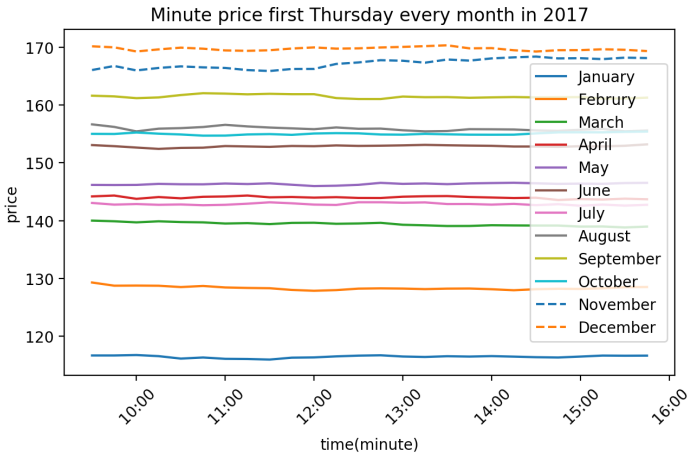


(c) First Wednesday volume every month

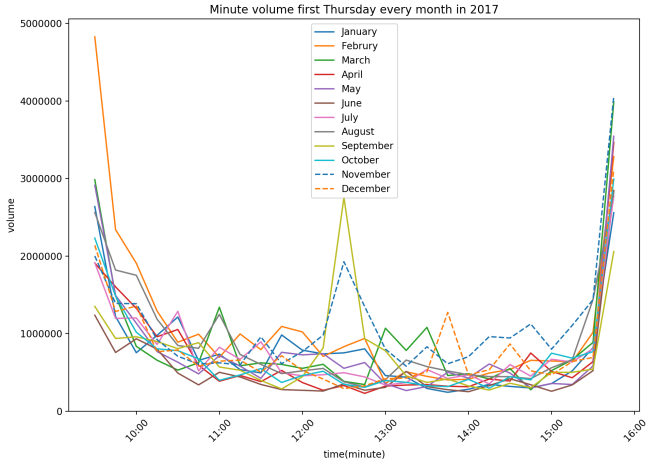
Figure A.18: Apple weekday data of price change, close price and volume each month in 2017



(a) First Thursday price change every month

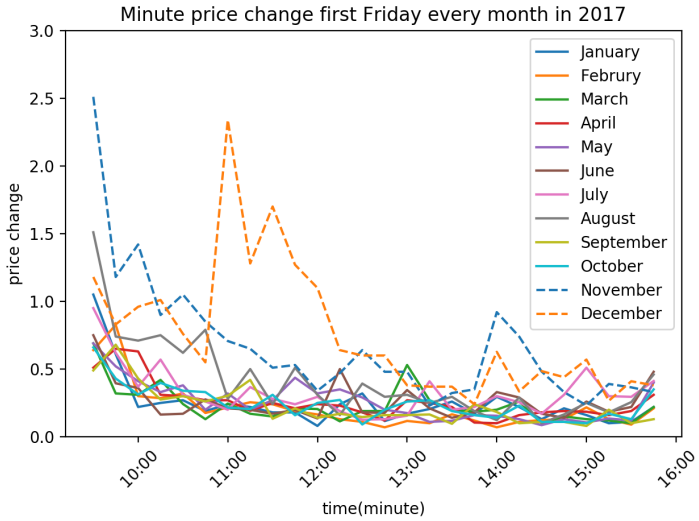


(b) First Thursday close price every month

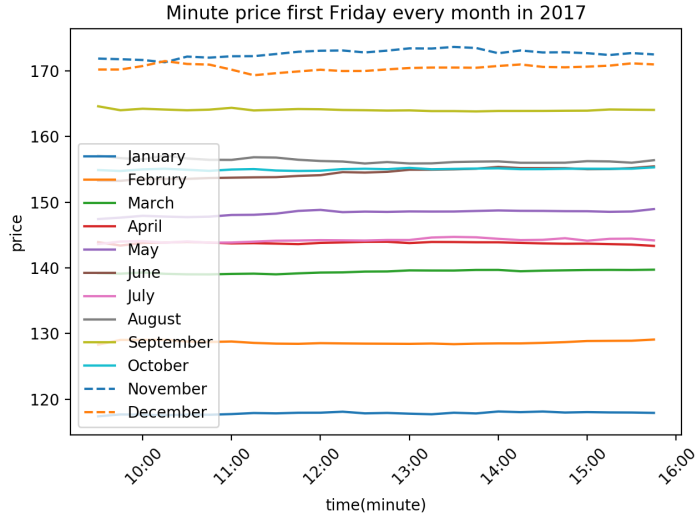


(c) First Thursday volume every month

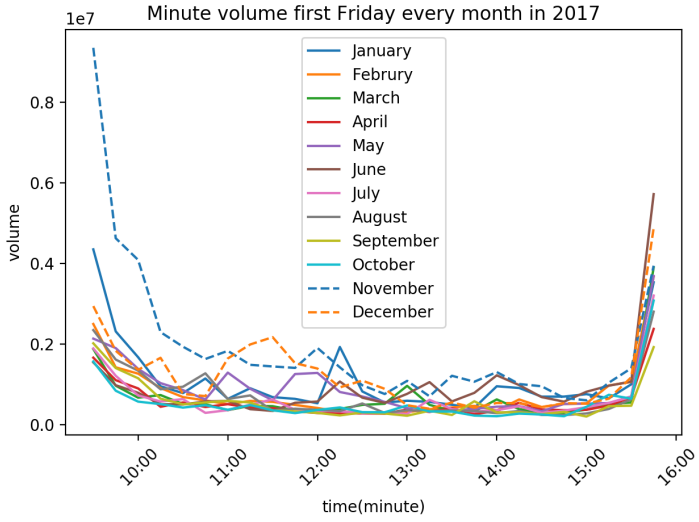
Figure A.19: Apple weekday data of price change, close price and volume each month in 2017



(a) First Friday price change every month

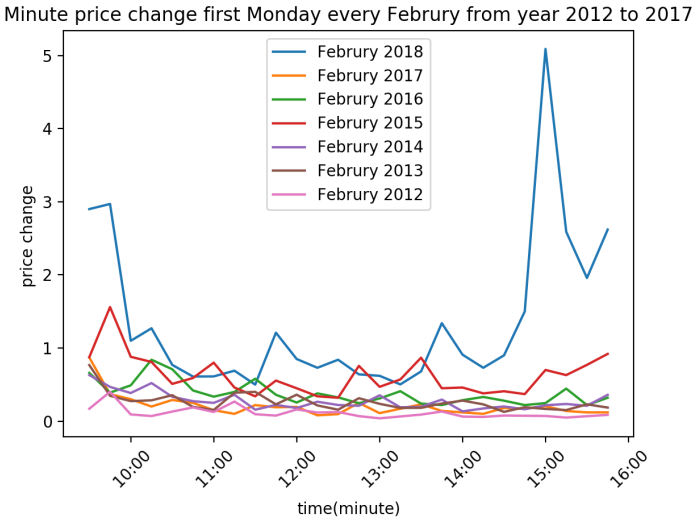


(b) First Friday close price every month

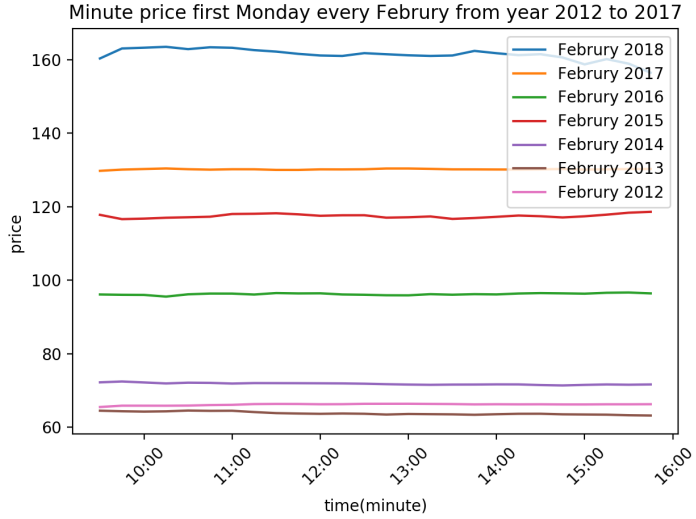


(c) First Friday volume every month

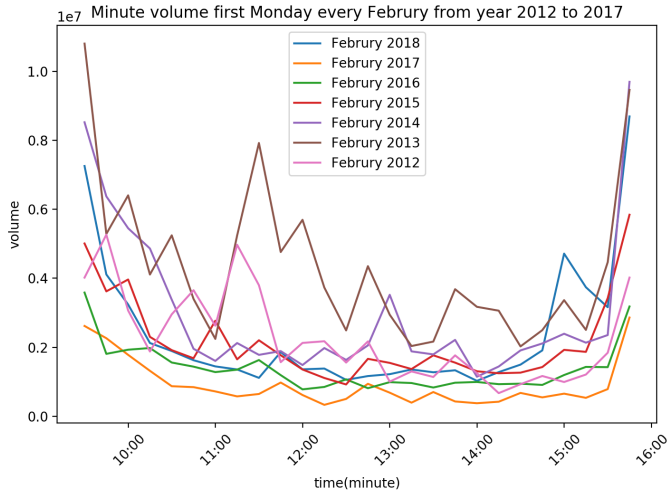
Figure A.20: Apple weekday data of price change, close price and volume each month in 2017



(a) First Monday price change

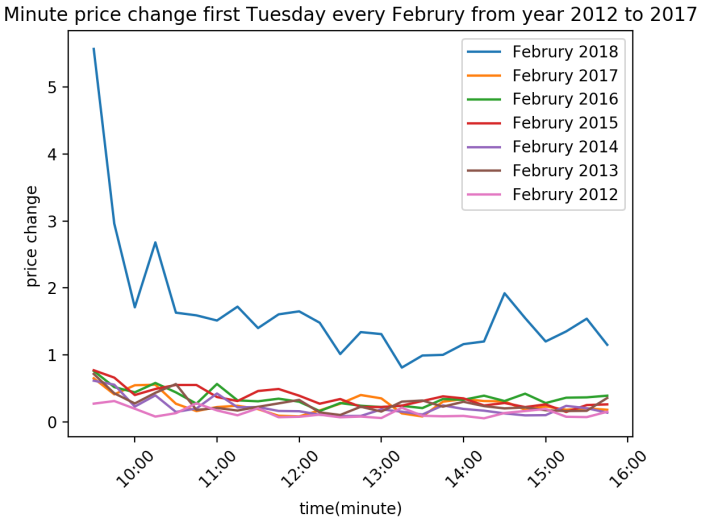


(b) First Monday close price

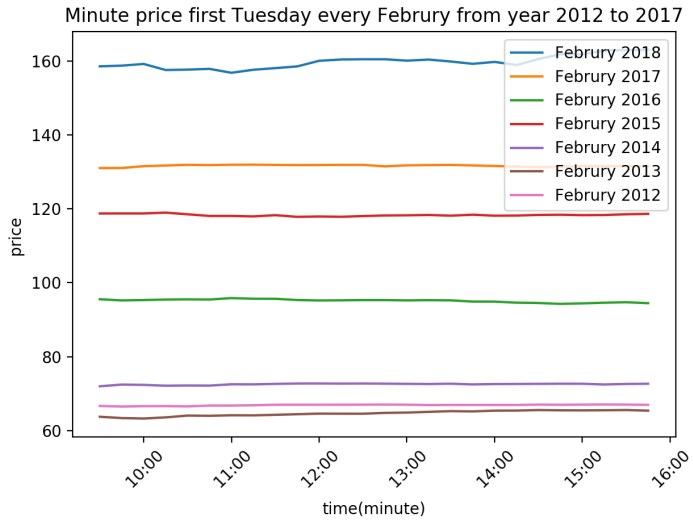


(c) First Monday volume

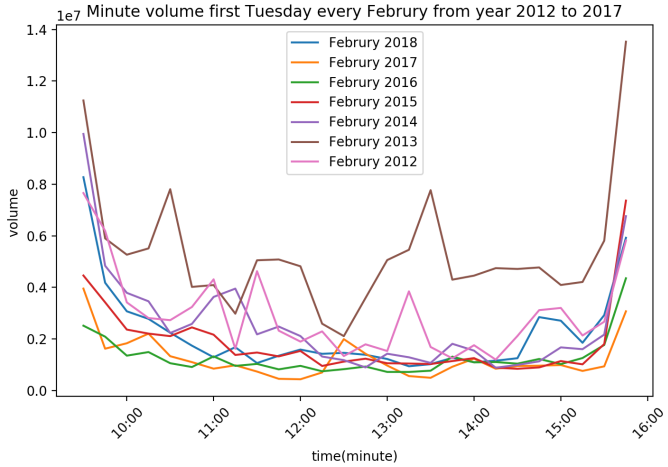
Figure A.21: Apple weekday data of price change, close price and volume each February between year 2012 to year 2017



(a) First Tuesday price change

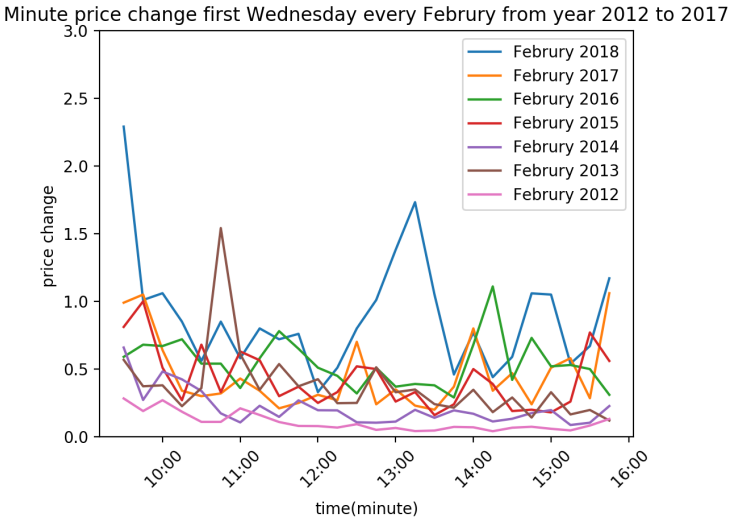


(b) First Tuesday close price

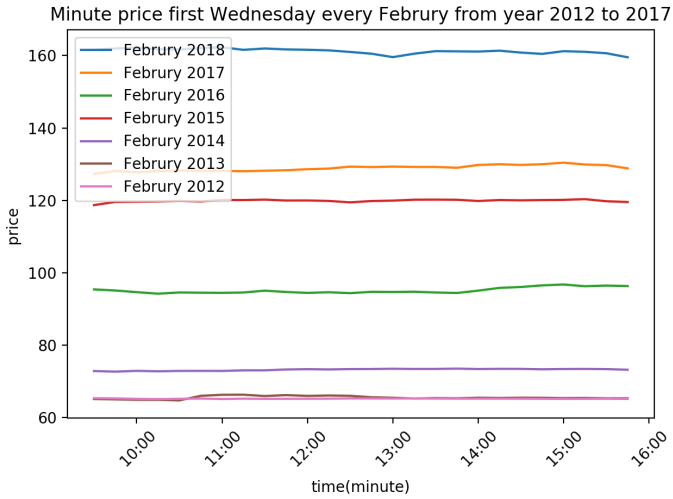


(c) First Tuesday volume

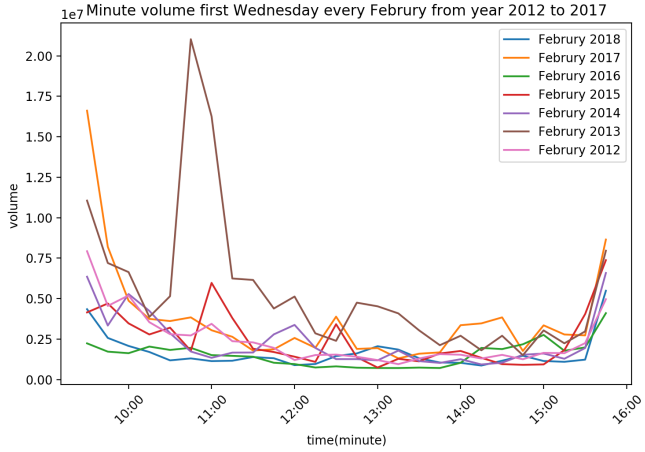
Figure A.22: Apple weekday data of price change, close price and volume each February between year 2012 to year 2017



(a) First Wednesday price change

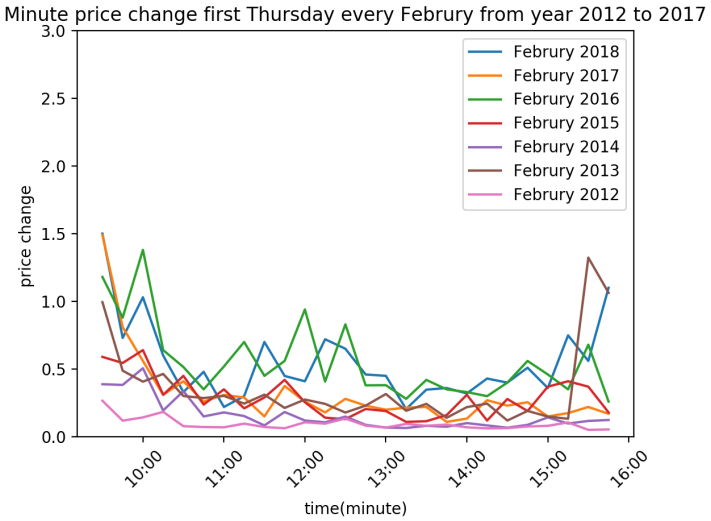


(b) First Wednesday close price

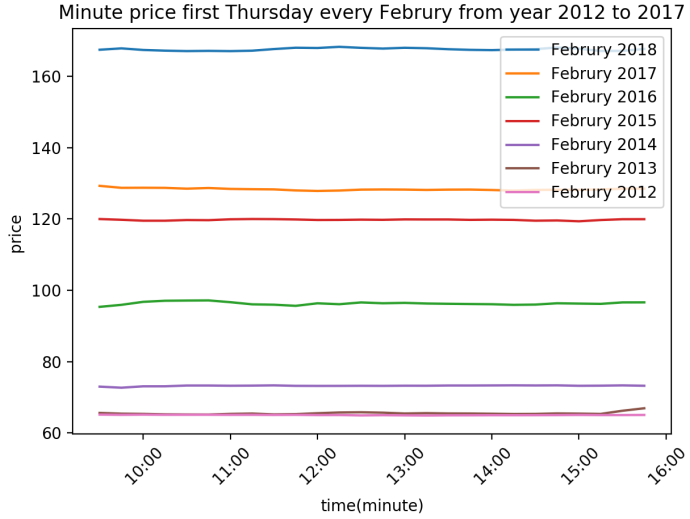


(c) First Wednesday volume

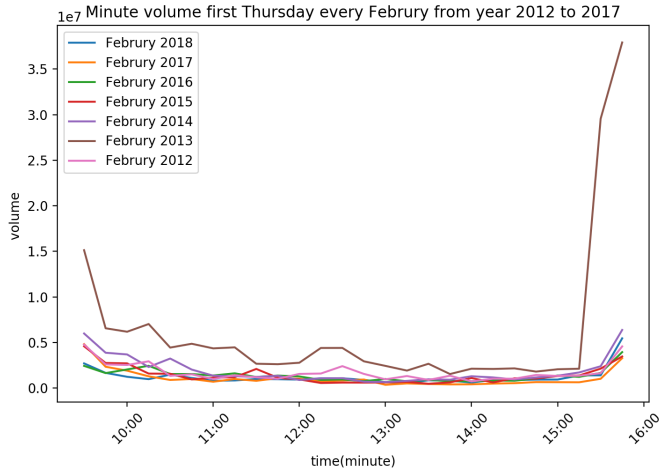
Figure A.23: Apple weekday data of price change, close price and volume each February between year 2012 to year 2017



(a) First Thursday price change

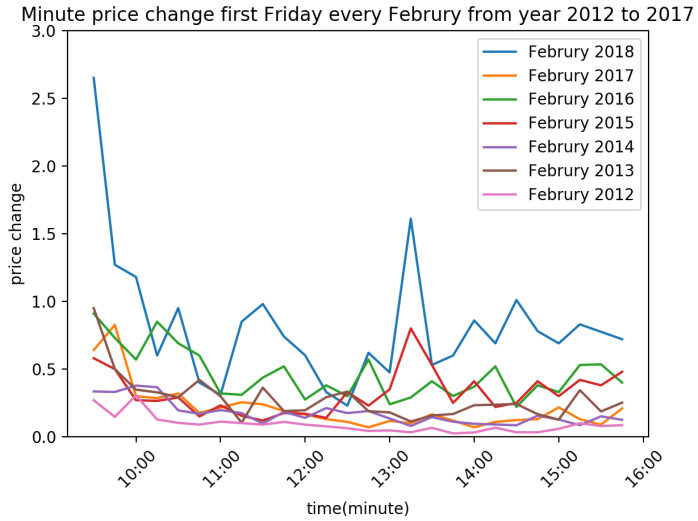


(b) First Thursday close price

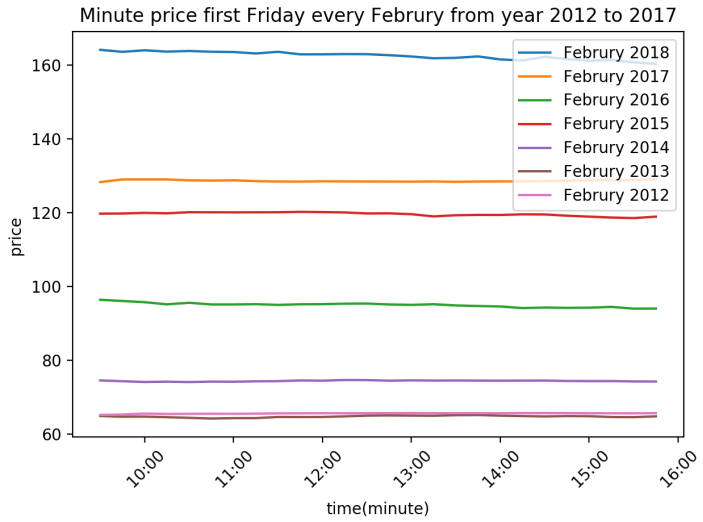


(c) First Thursday volume

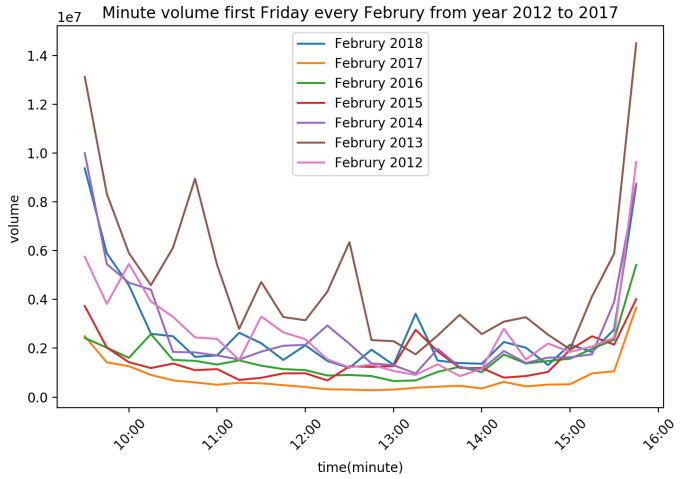
Figure A.24: Apple weekday data of price change, close price and volume each February between year 2012 to year 2017



(a) First Friday price change

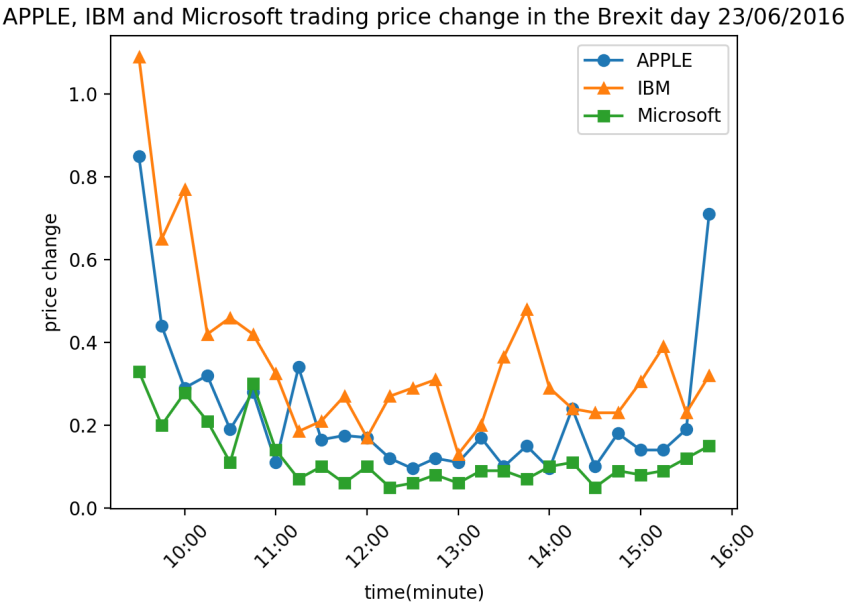


(b) First Friday close price

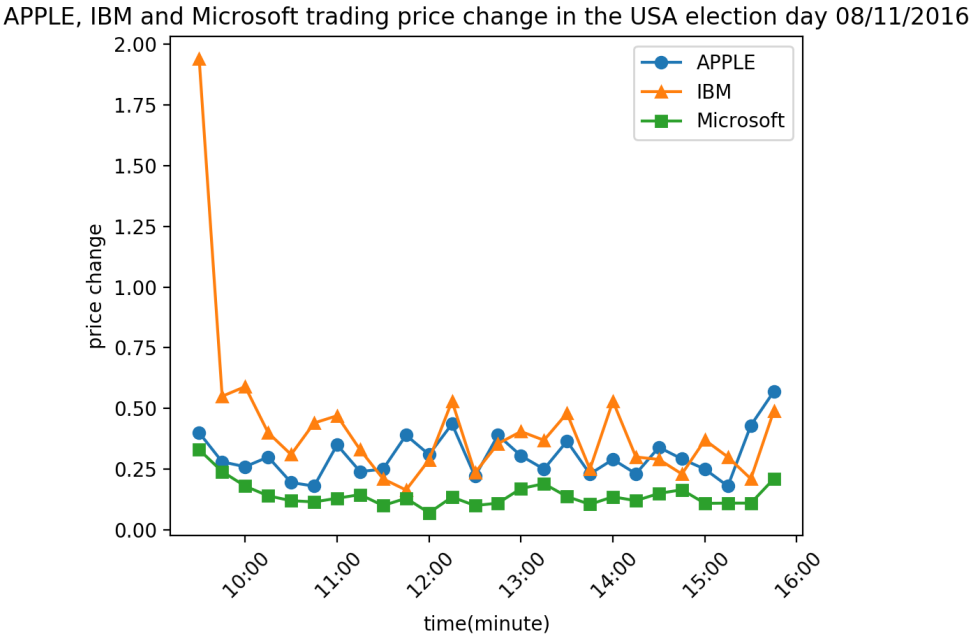


(c) First Friday volume

Figure A.25: Apple weekday data of price change, close price and volume each February between year 2012 to year 2017

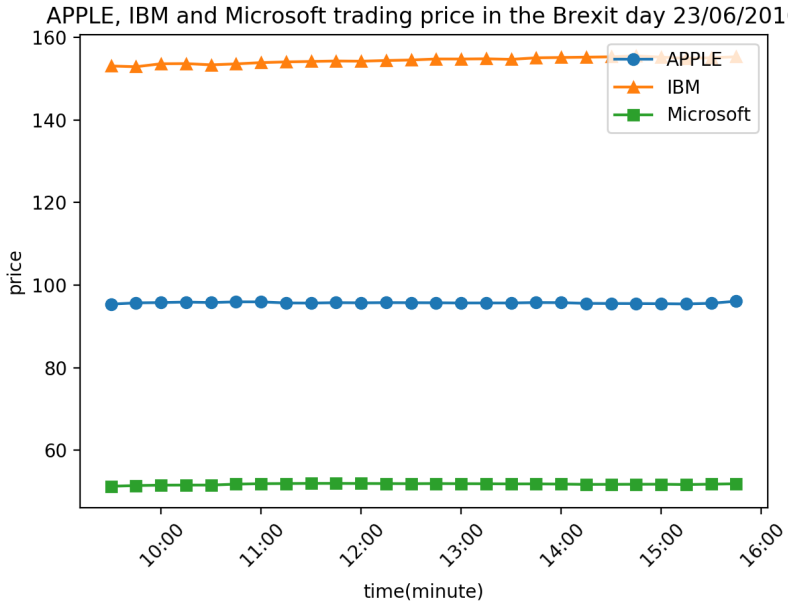


(a) price change when Britain voted to leave the European Union

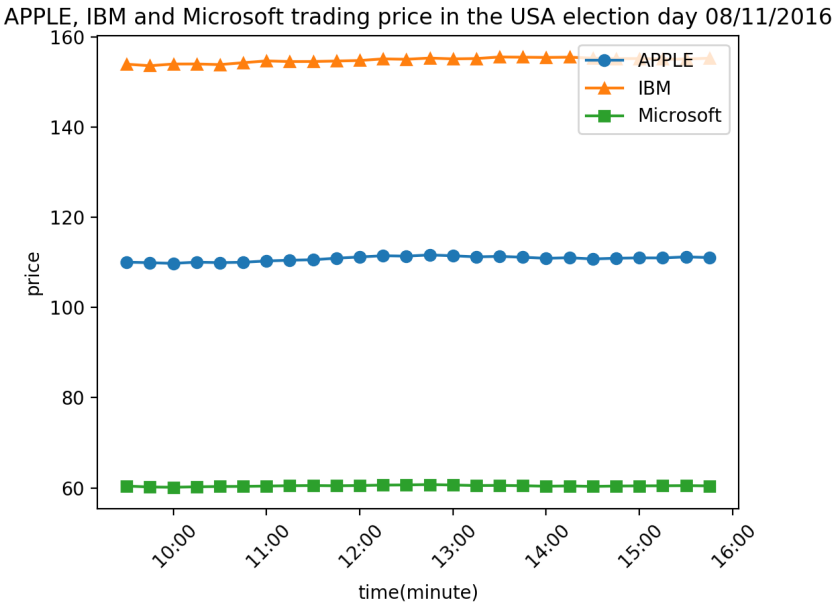


(b) price change when Donald Trump being elected as the USA president

Figure A.26: Apple,IBM and Microsoft data of price change in the day Britain voted to leave the European Union and Donald Trump being elected as the USA president

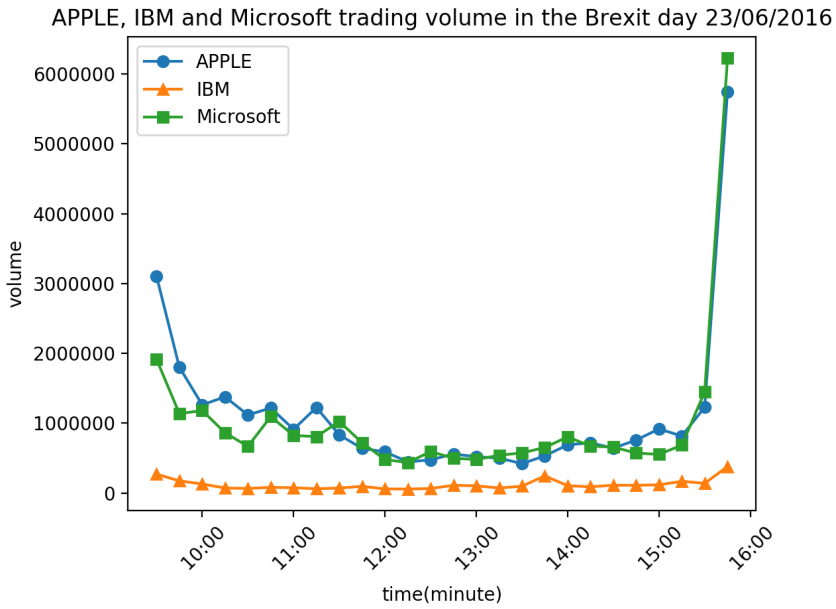


(a) close price when Britain voted to leave the European Union

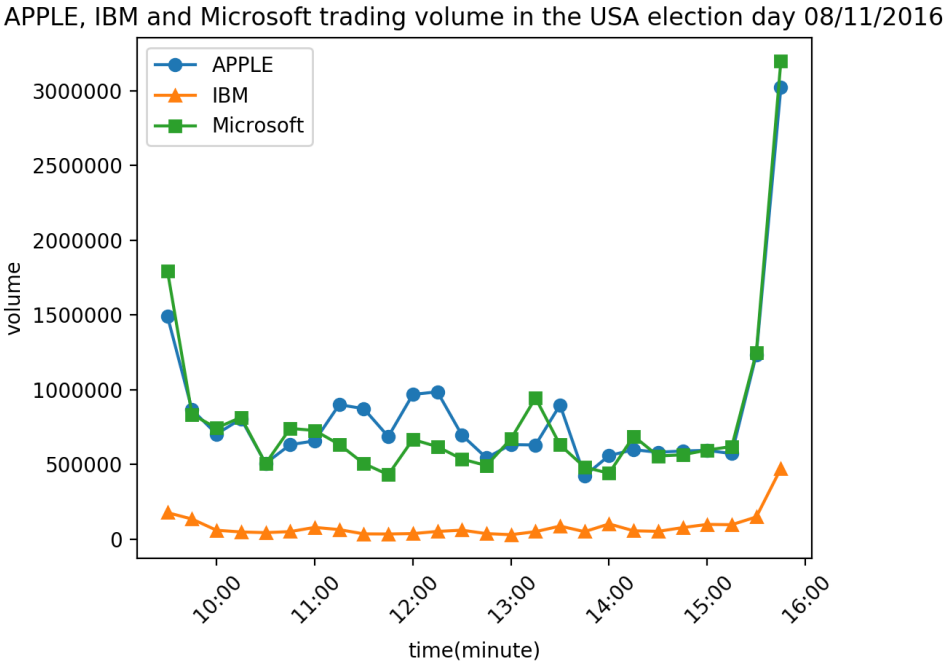


(b) close price when Donald Trump being elected as the USA president

Figure A.27: Apple, IBM and Microsoft data of close price in the day Britain voted to leave the European Union and Donald Trump being elected as the USA president

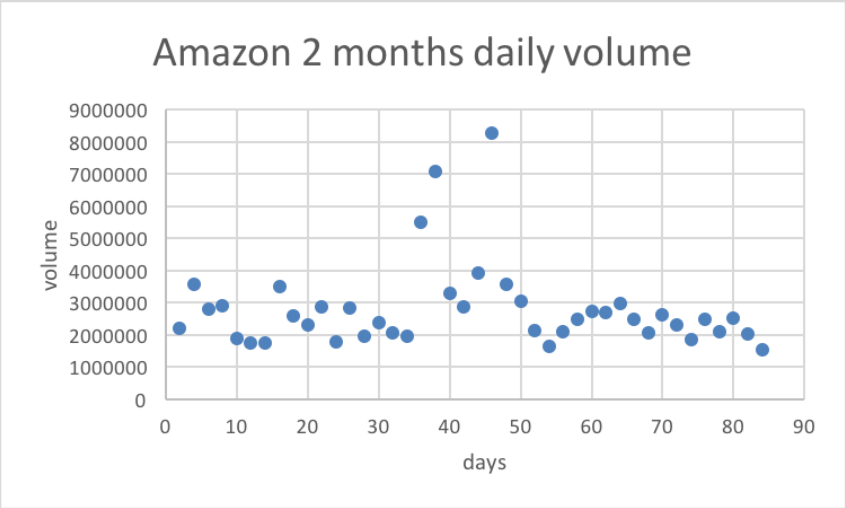


(a) volume when Britain voted to leave the European Union



(b) volume when Donald Trump being elected as the USA president

Figure A.28: Apple, IBM and Microsoft data of volume in the day Britain voted to leave the European Union and Donald Trump being elected as the USA president



(a) Amazon 1 months before and after announcement daily volume

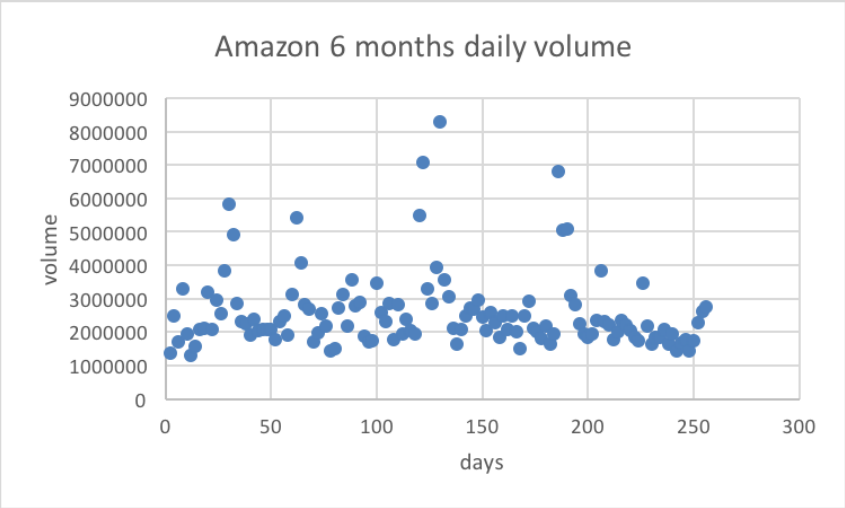


(b) Amazon 1 months before and after announcement daily close price



(c) Amazon 1 months before and after announcement daily price change

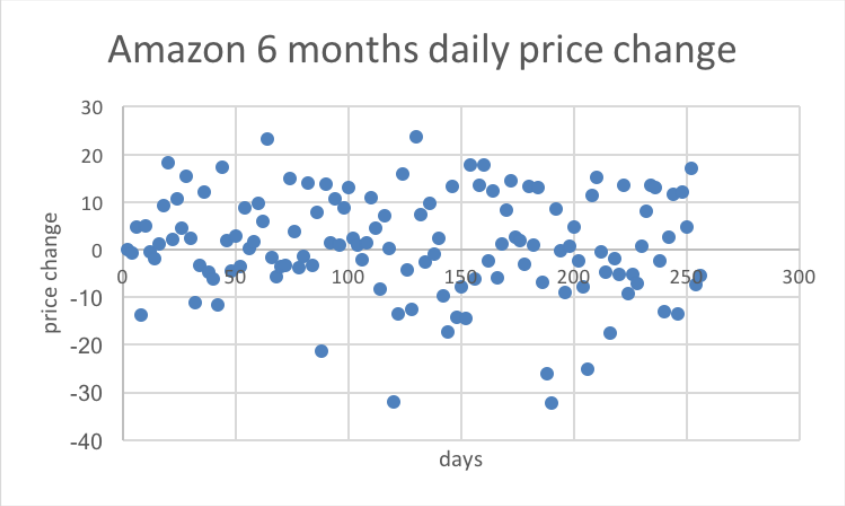
Figure A.29: Amazon 1 months before and after announcement daily



(a) Amazon 3 months before and after announcement daily volume

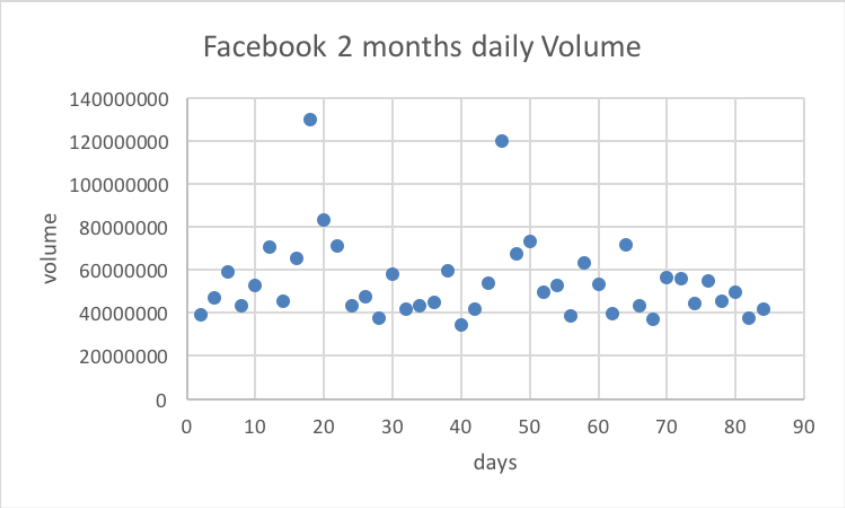


(b) Amazon 3 months before and after announcement daily close price

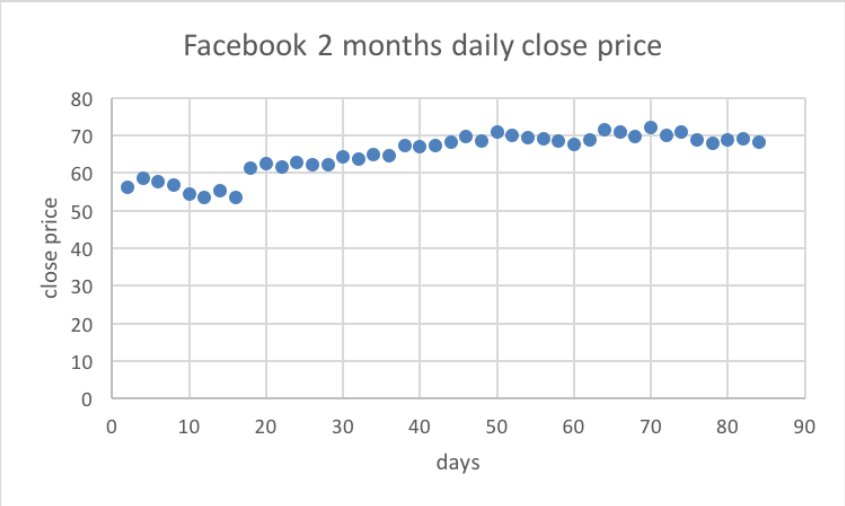


(c) Amazon 3 months before and after announcement daily price change

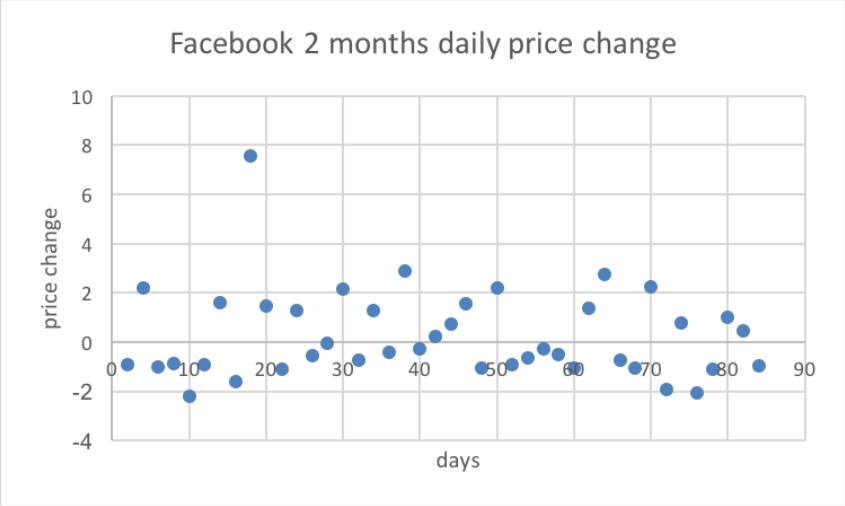
Figure A.30: Amazon 3 months before and after announcement daily



(a) Facebook 1 months before and after announcement daily volume

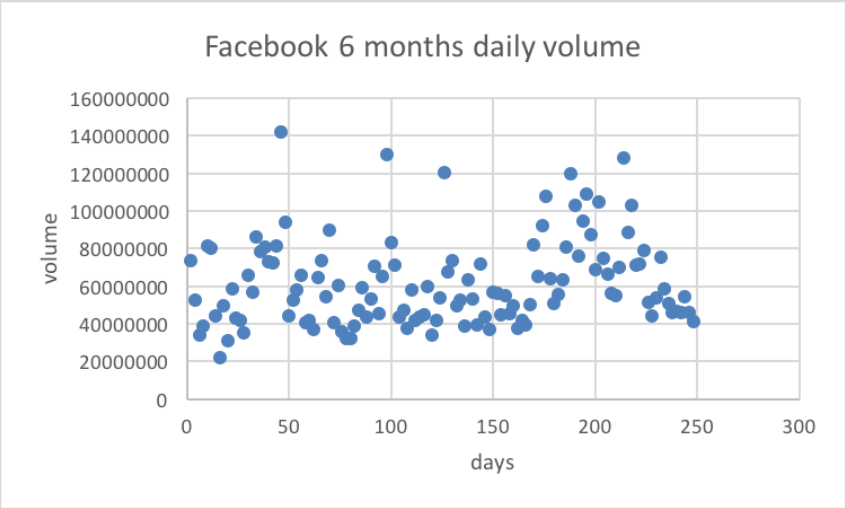


(b) Facebook 1 months before and after announcement daily close price

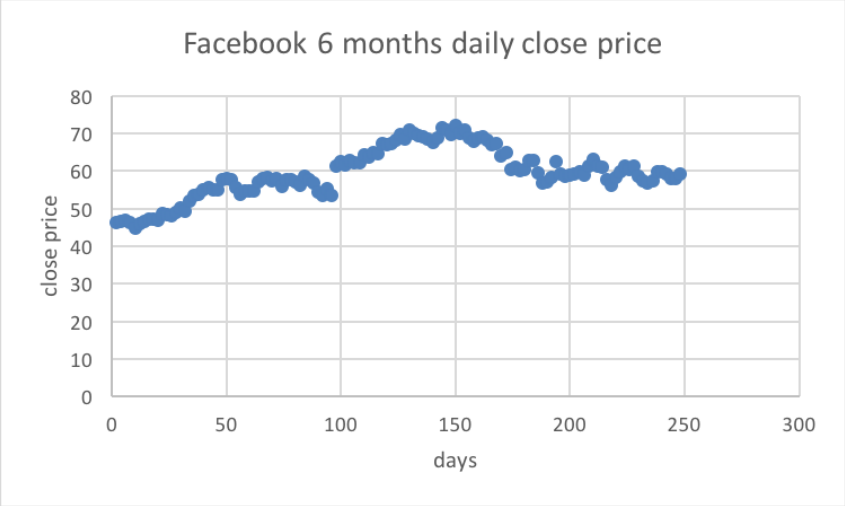


(c) Facebook 1 months before and after announcement daily price change

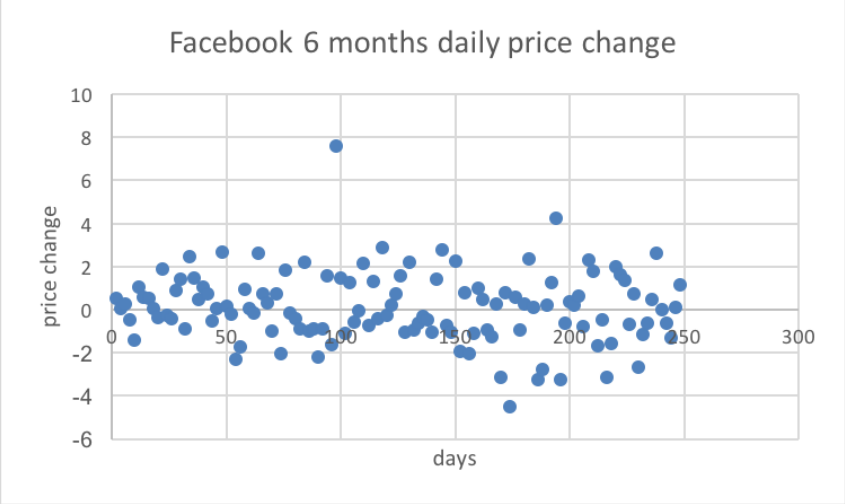
Figure A.31: Facebook 1 months before and after announcement daily



(a) Facebook 3 months before and after announcement daily volume

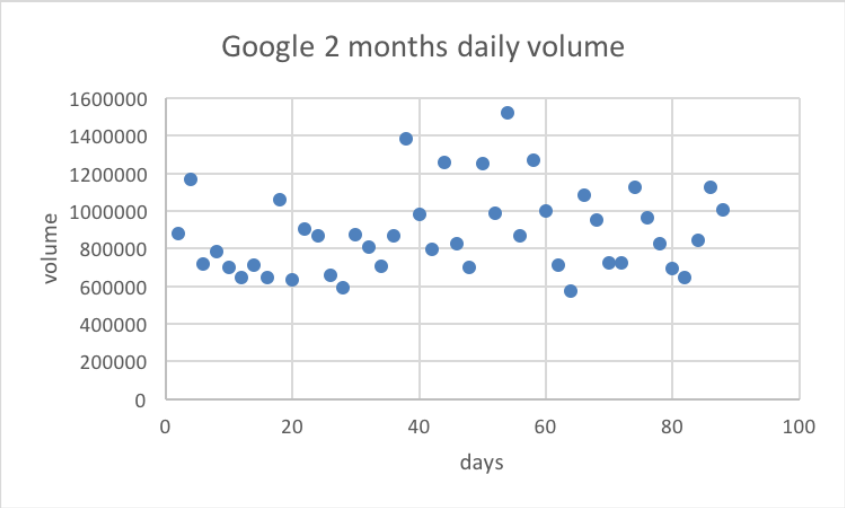


(b) Facebook 3 months before and after announcement daily close price

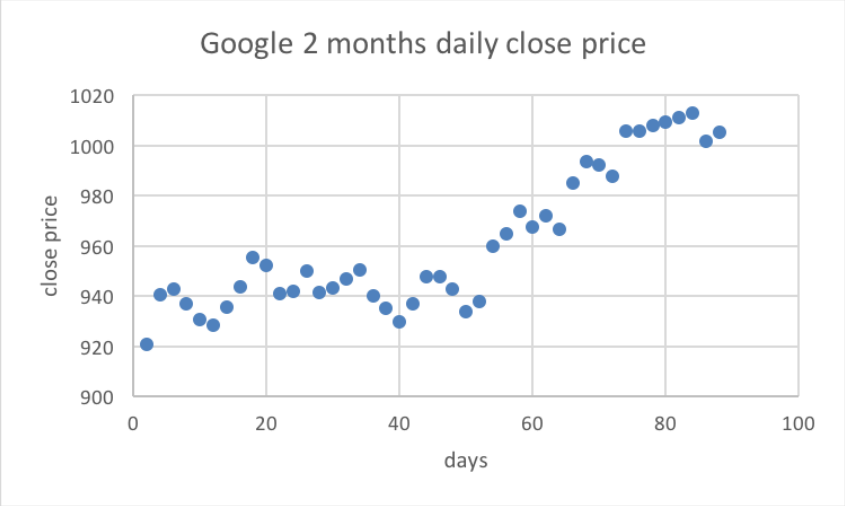


(c) Facebook 3 months before and after announcement daily price change

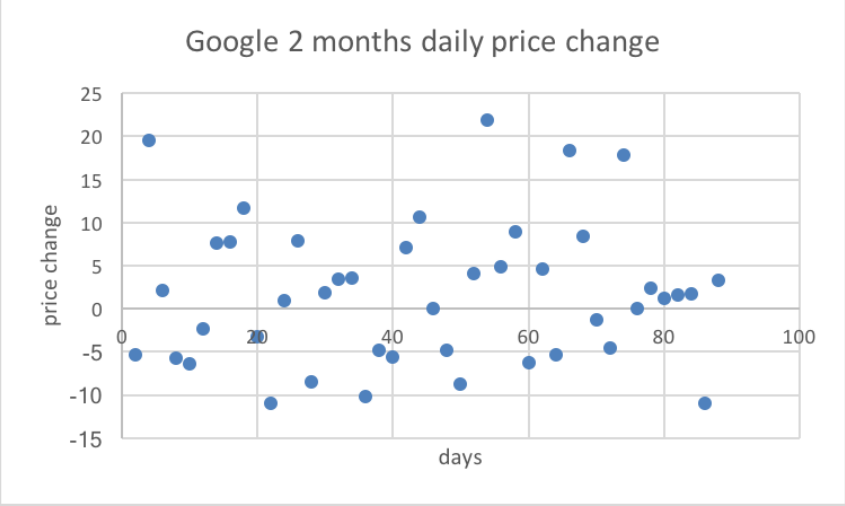
Figure A.32: Facebook 3 months before and after announcement daily



(a) Google 1 months before and after announcement daily volume

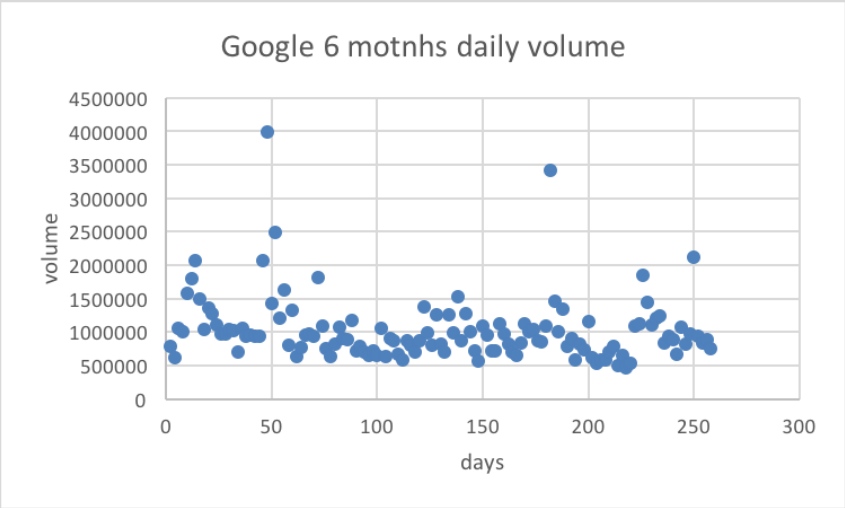


(b) Google 1 months before and after announcement daily close price

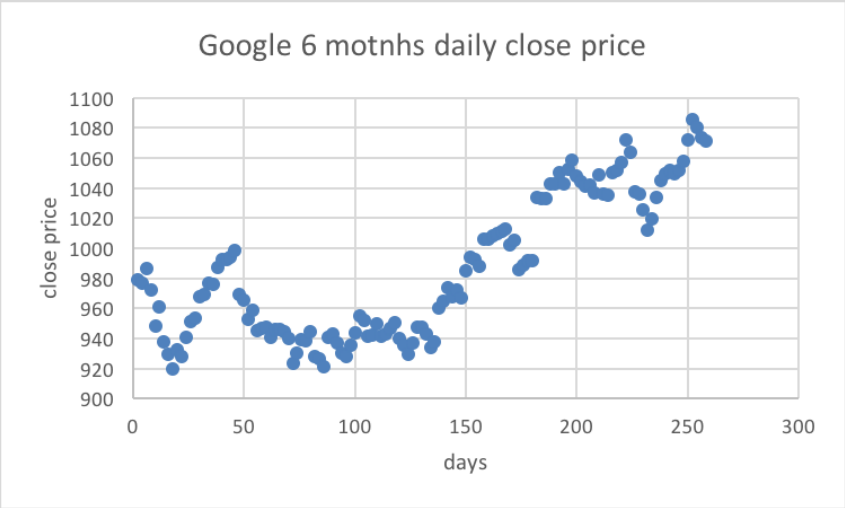


(c) Google 1 months before and after announcement daily price change

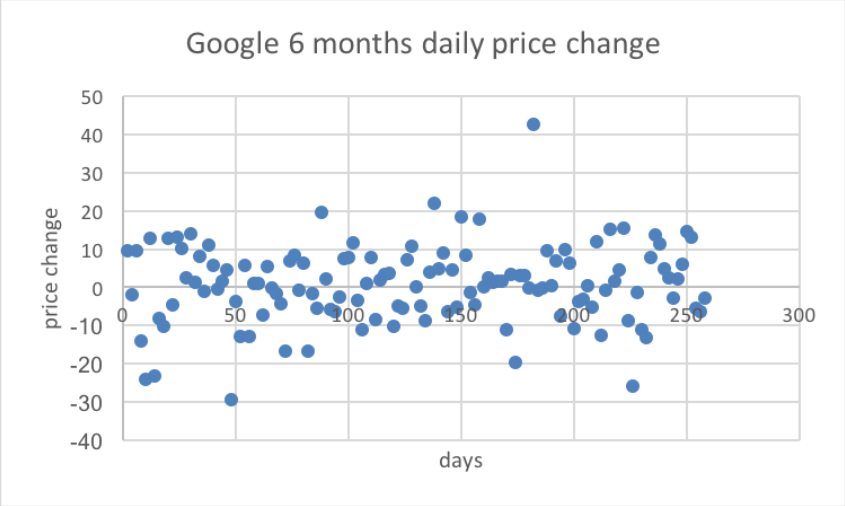
Figure A.33: Google 1 months before and after announcement daily



(a) Google 3 months before and after announcement daily volume

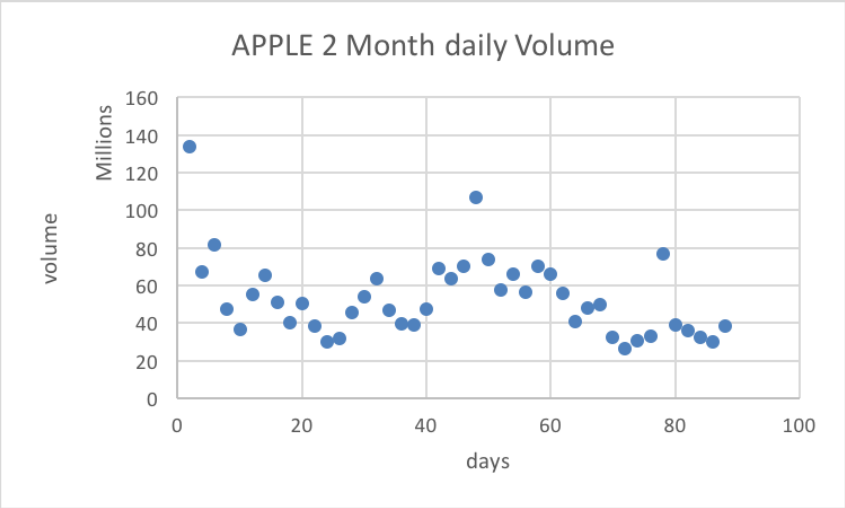


(b) Google 3 months before and after announcement daily close price

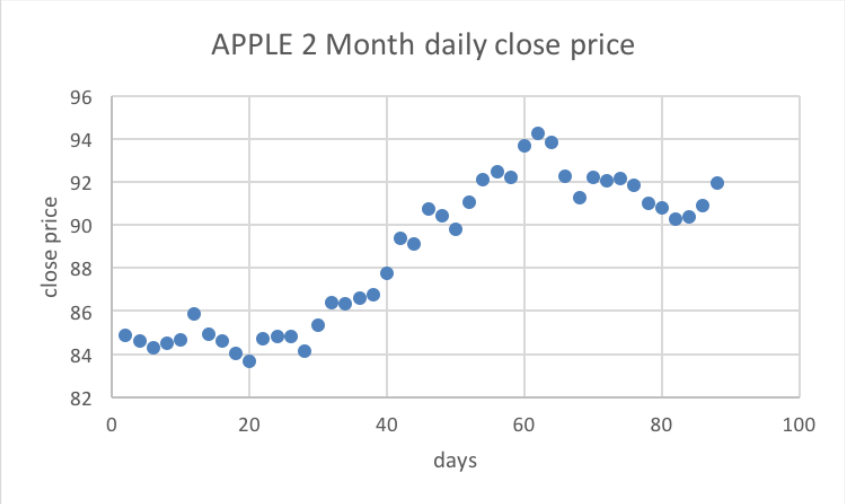


(c) Google 3 months before and after announcement daily price change

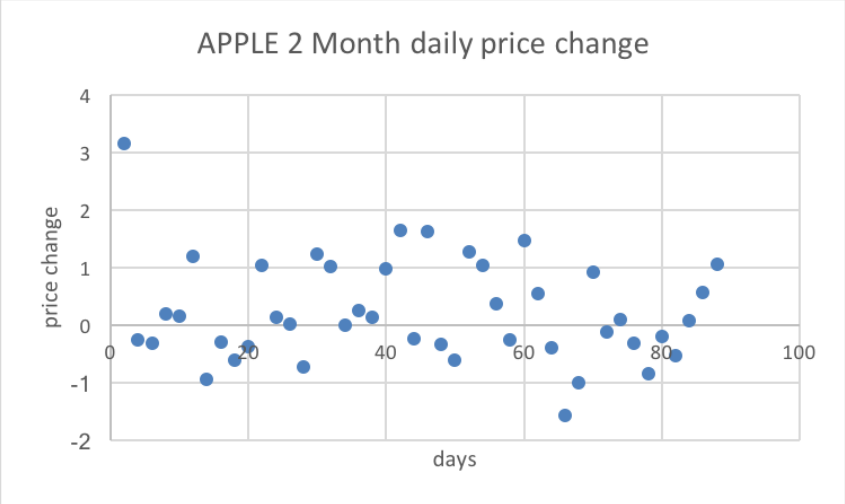
Figure A.34: Google 3 months before and after announcement daily



(a) Apple 1 months before and after announcement daily volume

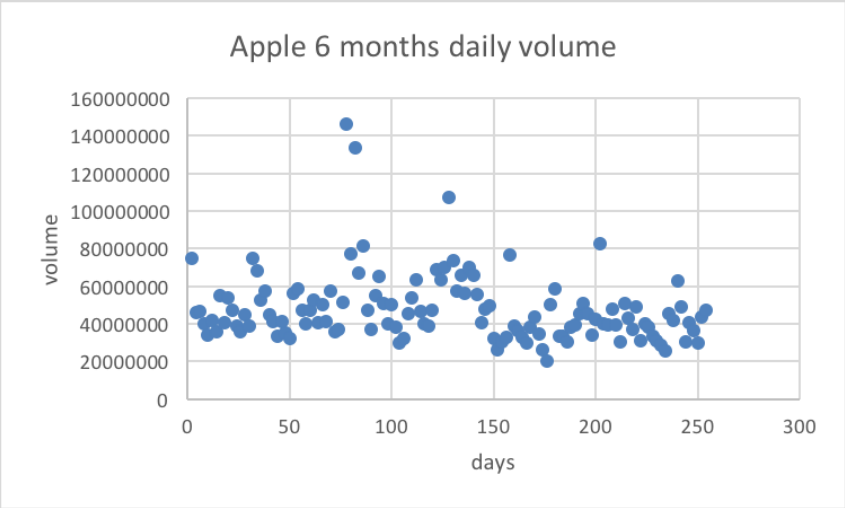


(b) Apple 1 months before and after announcement daily close price

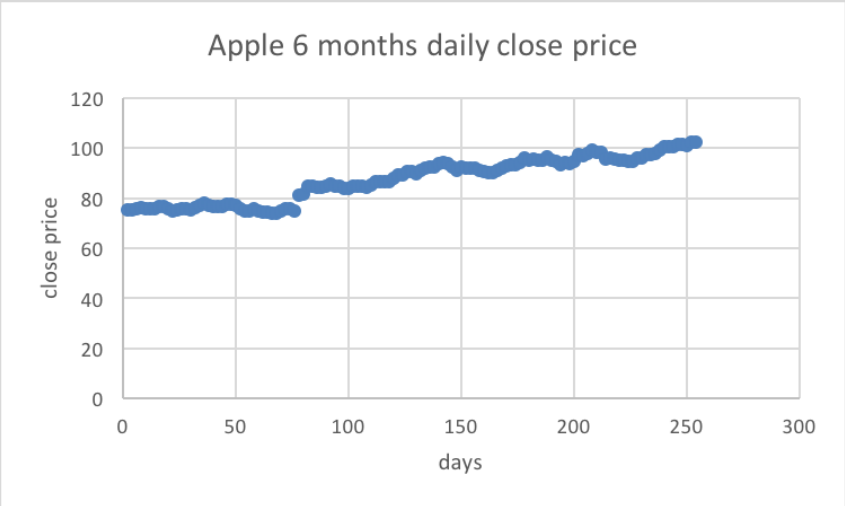


(c) Apple 1 months before and after announcement daily price change

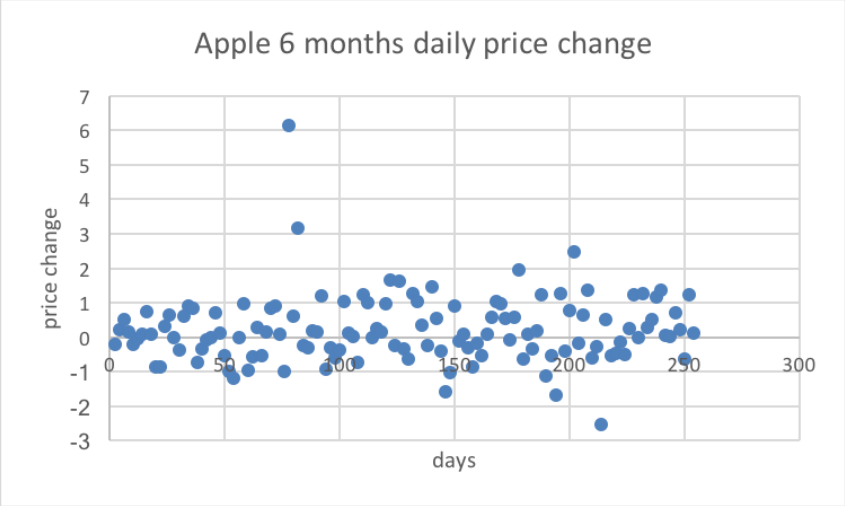
Figure A.35: Apple 1 months before and after announcement daily



(a) Apple 3 months before and after announcement daily volume

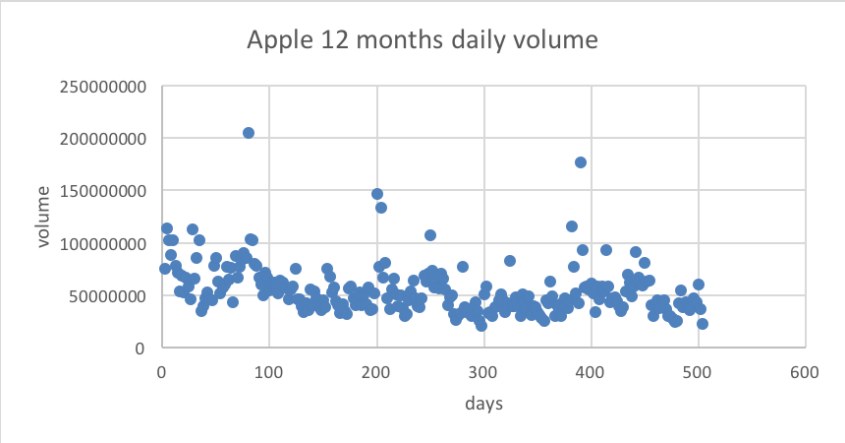


(b) Apple 3 months before and after announcement daily close price

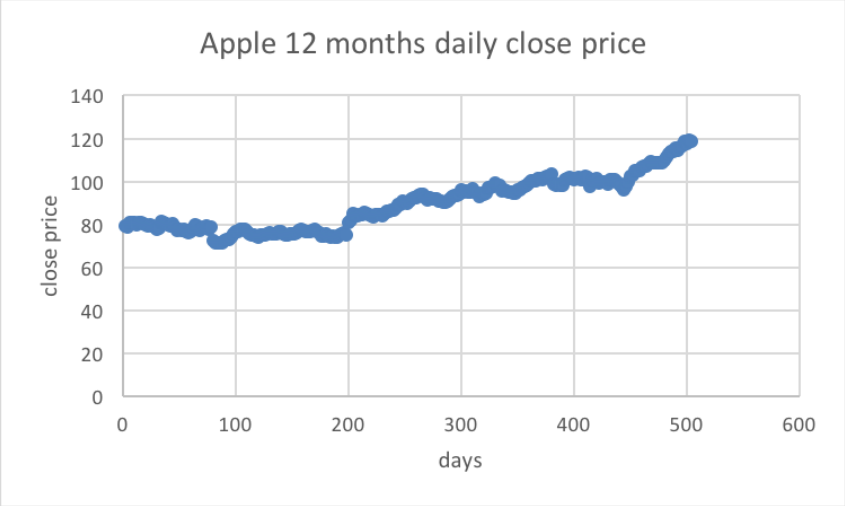


(c) Apple 3 months before and after announcement daily price change

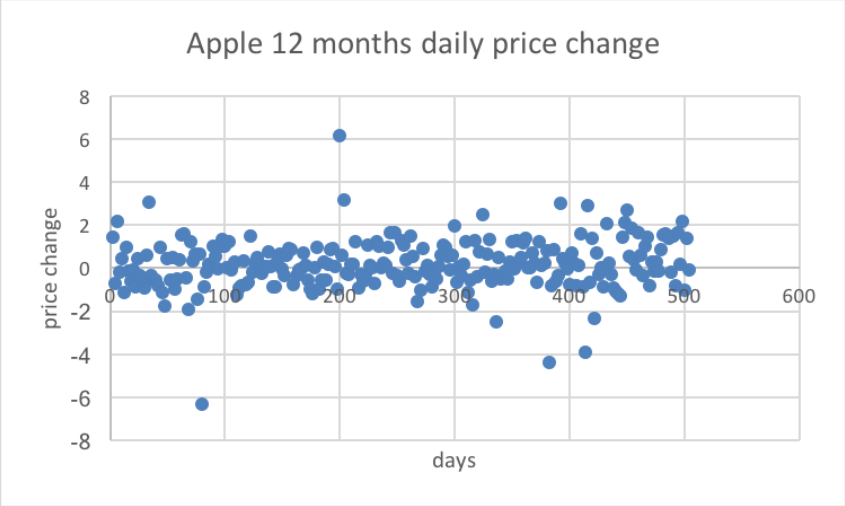
Figure A.36: Apple 3 months before and after announcement daily



(a) Apple 6 months before and after announcement daily volume

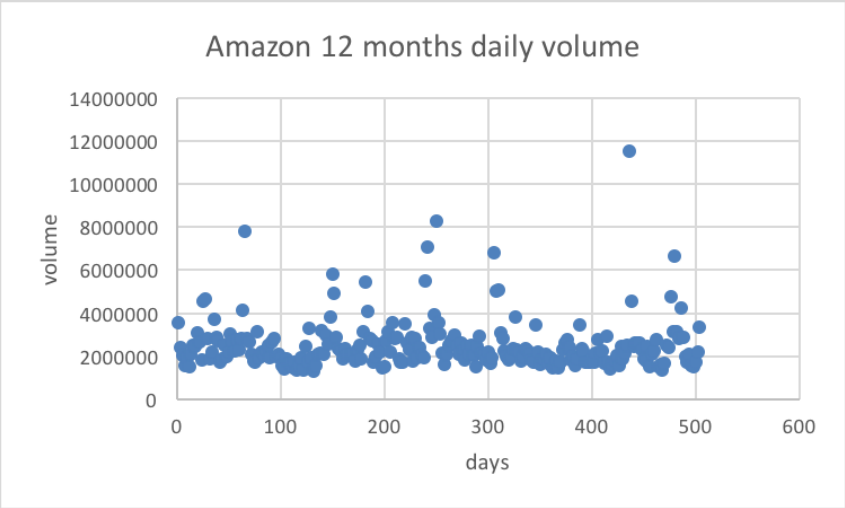


(b) Apple 6 months before and after announcement daily close price



(c) Apple 6 months before and after announcement daily price change

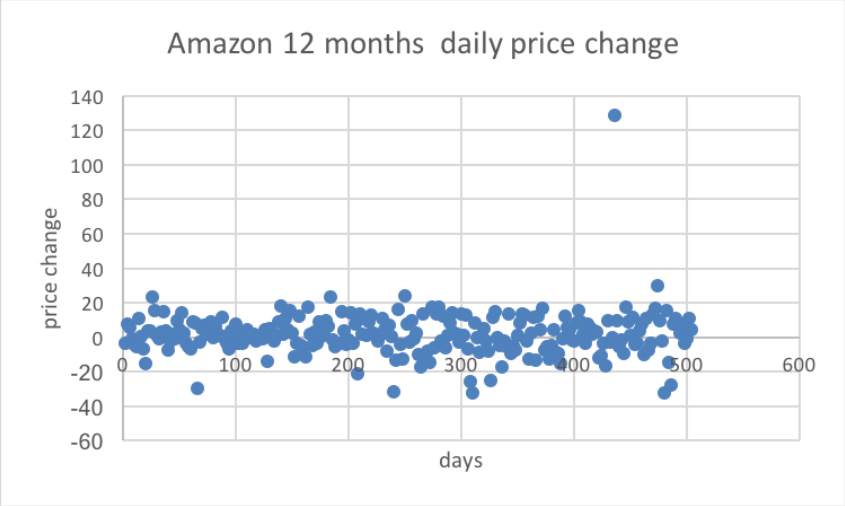
Figure A.37: Apple 6 months before and after announcement daily



(a) Amazon 6 months before and after announcement daily volume

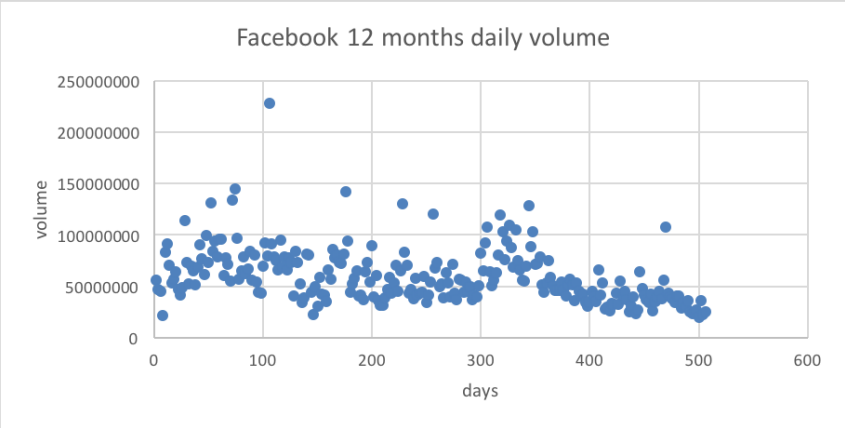


(b) Amazon 6 months before and after announcement daily close price

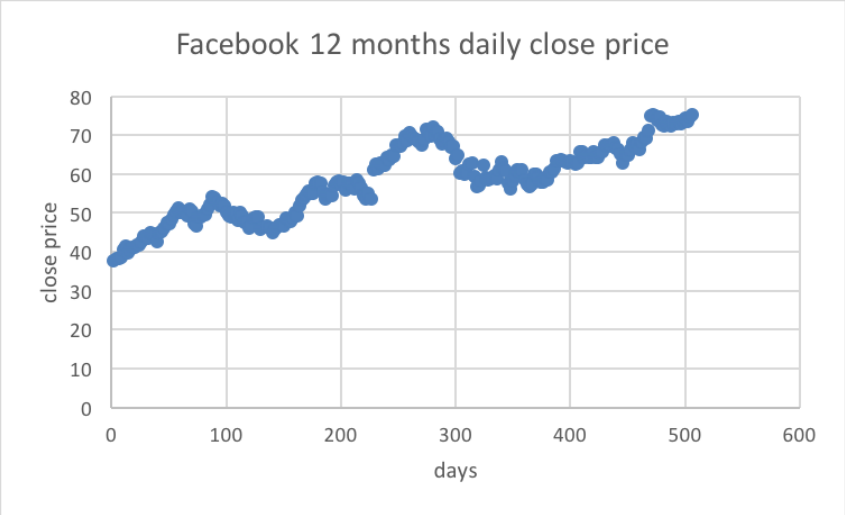


(c) Amazon 6 months before and after announcement daily price change

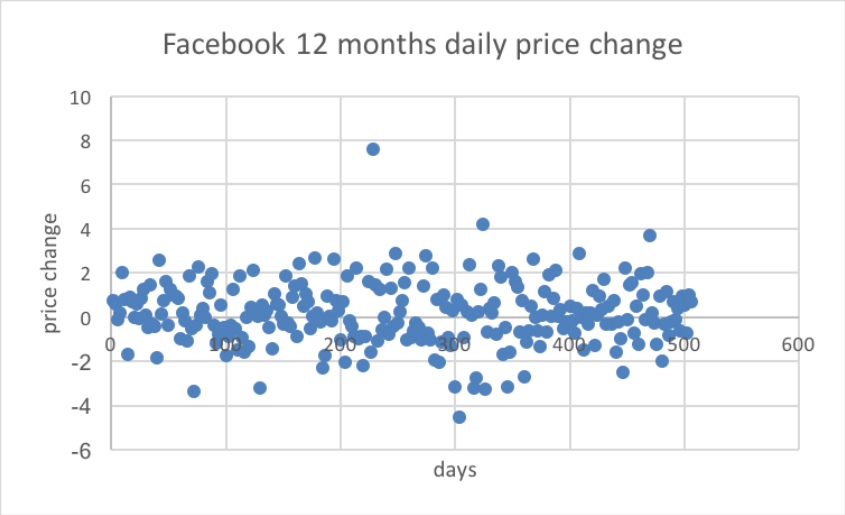
Figure A.38: Amazon 6 months before and after announcement daily



(a) Facebook 6 months before and after announcement daily volume

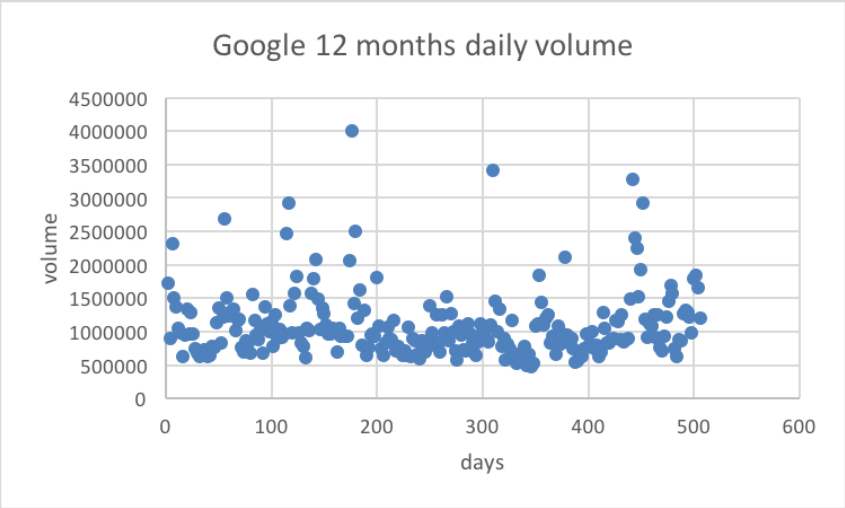


(b) Facebook 6 months before and after announcement daily close price

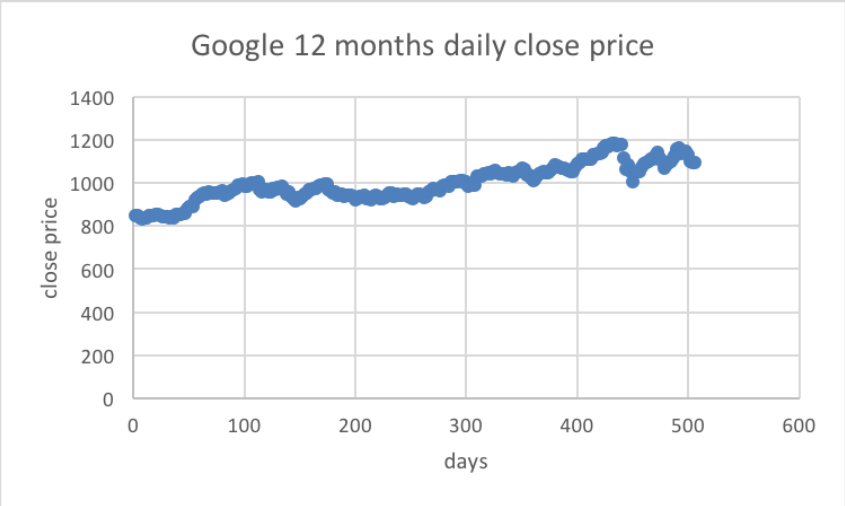


(c) Facebook 6 months before and after announcement daily price change

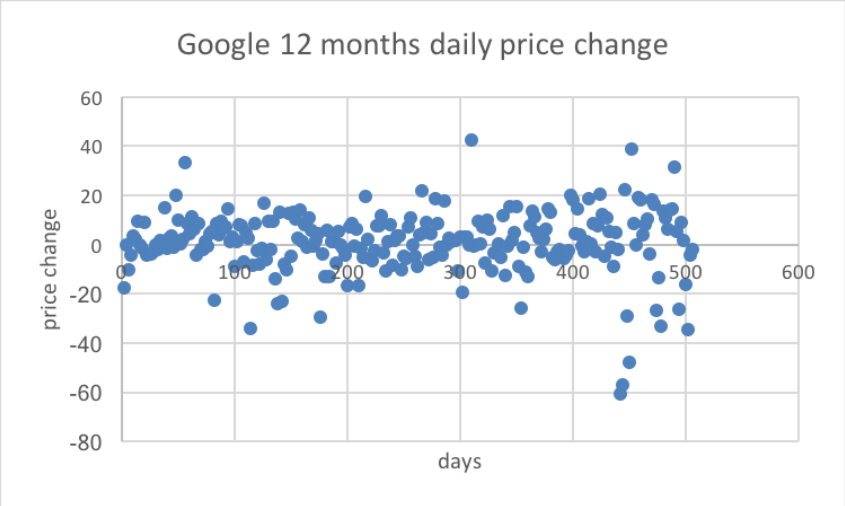
Figure A.39: Facebook 6 months before and after announcement daily



(a) Google 6 months before and after announcement daily volume

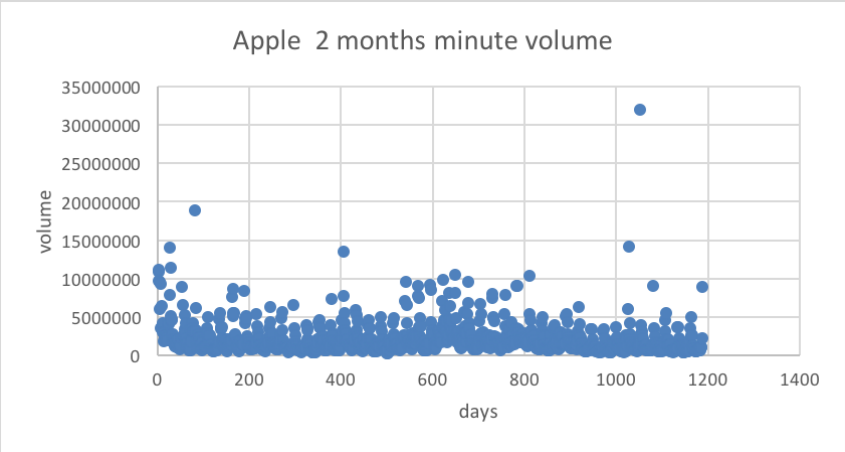


(b) Google 6 months before and after announcement daily close price



(c) Google 6 months before and after announcement daily price change

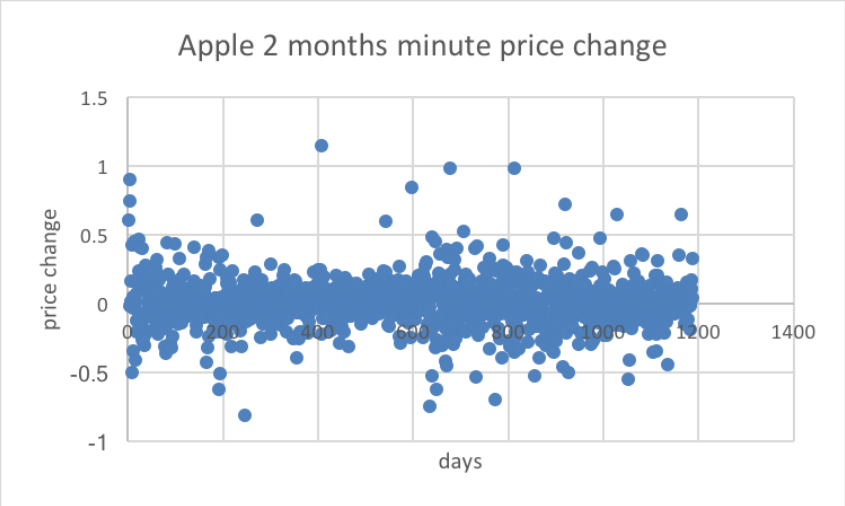
Figure A.40: Google 6 months before and after announcement daily



(a) Apple 1 months before and after announcement minute volume

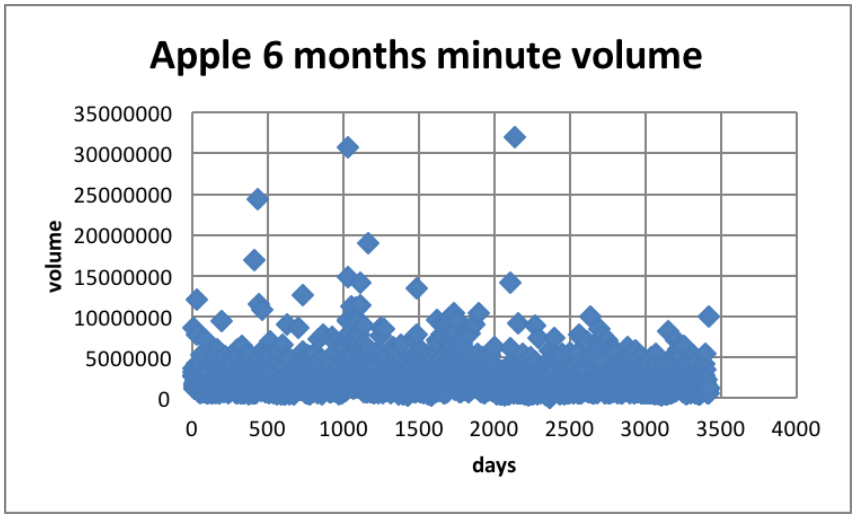


(b) Apple 1 months before and after announcement minute close price

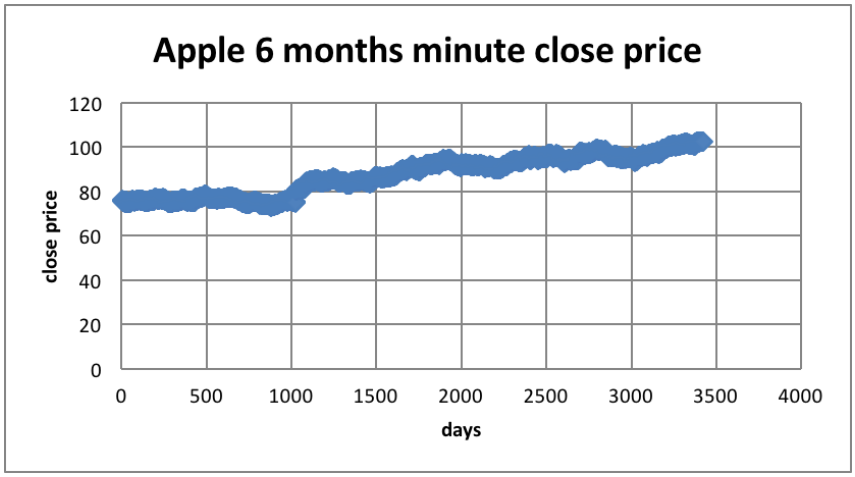


(c) Apple 1 months before and after announcement minute price change

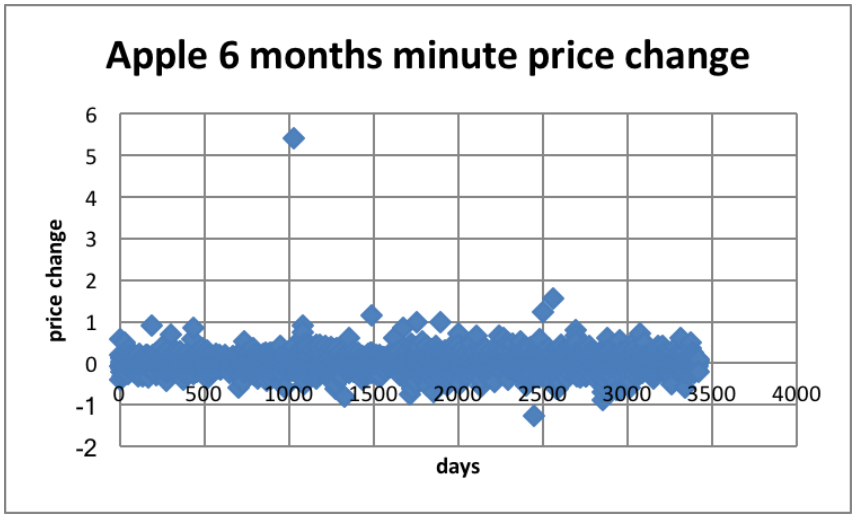
Figure A.41: Apple 1 months before and after announcement minute



(a) Apple 3 months before and after announcement minute volume

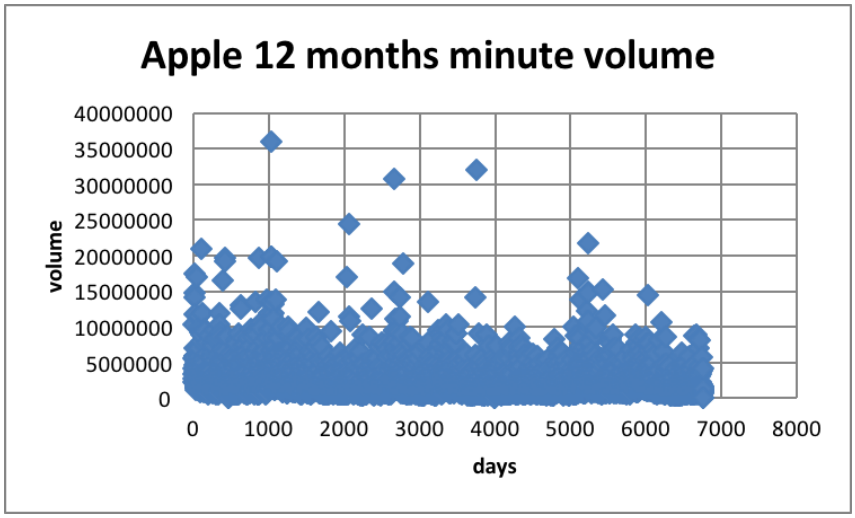


(b) Apple 3 months before and after announcement minute close price

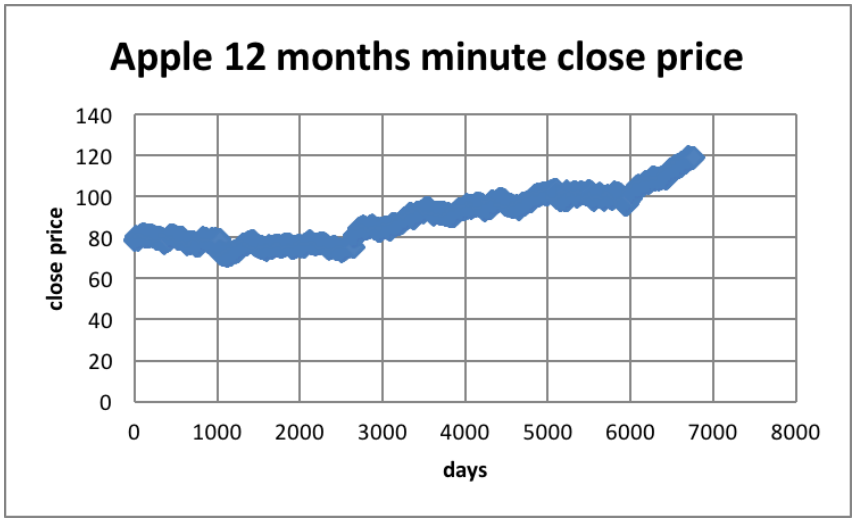


(c) Apple 3 months before and after announcement minute price change

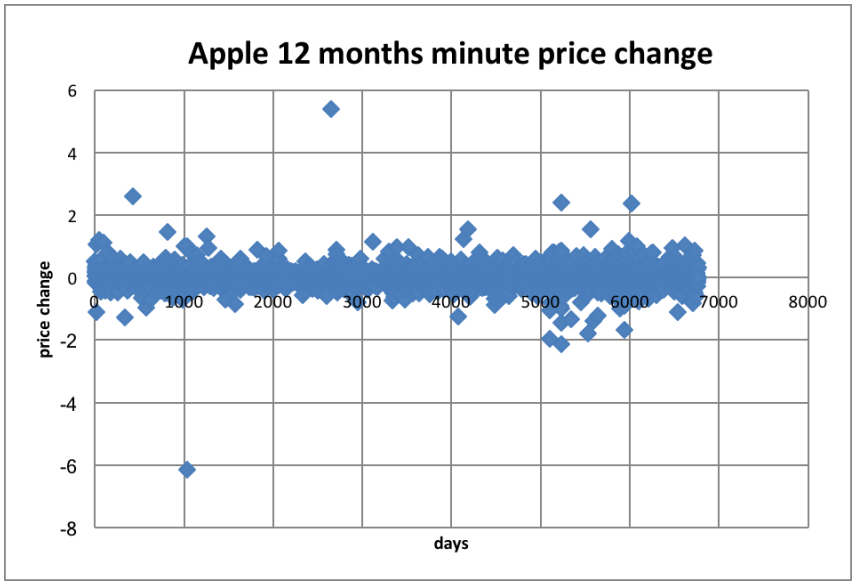
Figure A.42: Apple 3 months before and after announcement minute



(a) Apple 6 months before and after announcement minute volume



(b) Apple 6 months before and after announcement minute close price



(c) Apple 6 months before and after announcement minute price change

Figure A.43: Apple 6 months before and after announcement minute

Appendix B

Crunchbase Data

The figures below are four companies' (Facebook, Amazon, Google, Apple) up-to-date acquisitions. They are downloaded from Crunchbase. Crunchbase is a platform to find business information about private and public companies. The original purpose of the Crunchbase is to track startups, its information include investment and funding, mergers and acquisitions, news, industry trend, founding members and individuals in leadership positions.

crunchbase

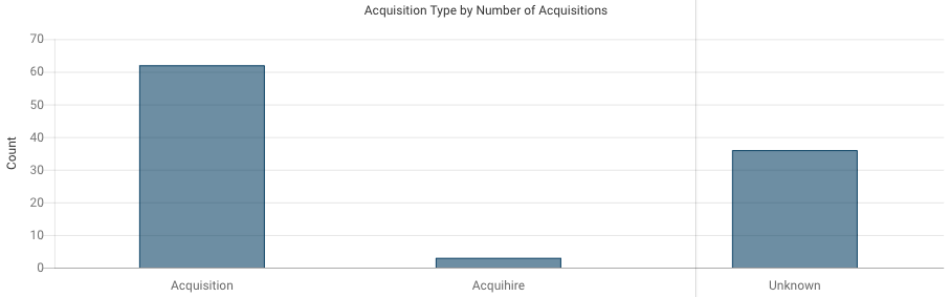
Apple > Acquisitions

Acquisitions

Number of Acquisitions 101

Apple has acquired 101 organizations. Their most recent acquisition was Asaii on Oct 14, 2018.

Which types of acquisition does this organization make most frequently? HIDE



Acquired Organization Name	Announced Date	Price	Transactior
Asaii	Oct 14, 2018	–	Asaii acqu Apple
Dialog Semiconductor	Oct 11, 2018	$600M	Dialog Se acquired
Akonia Holographics	Aug 29, 2018	–	Akonia Ho acquired
Texture by Next Issue	Mar 13, 2018	–	Texture by acquired
Next Issue Media	Mar 12, 2018	–	Next Issu acquired
Shazam Entertainment	Dec 11, 2017	$400M	Shazam Entertainr acquired
Pop Up Archive	Dec 5, 2017	–	Pop Up Ar acquired
Spektral	Dec 1, 2017	$30M	Spektral a Apple
Vrvana	Nov 21, 2017	$30M	Vrvana ac Apple
InVisage Technologies	Nov 9, 2017	–	InVisage Technolog by Apple
PowerbyProxi	Oct 24, 2017	–	PowerbyP acquired
Acquired Organization Name	Announced Date	Price	Transactior
Regaind	Sep 29, 2017		Regaind a

Figure B.1: The Apple acquisition list

crunchbase

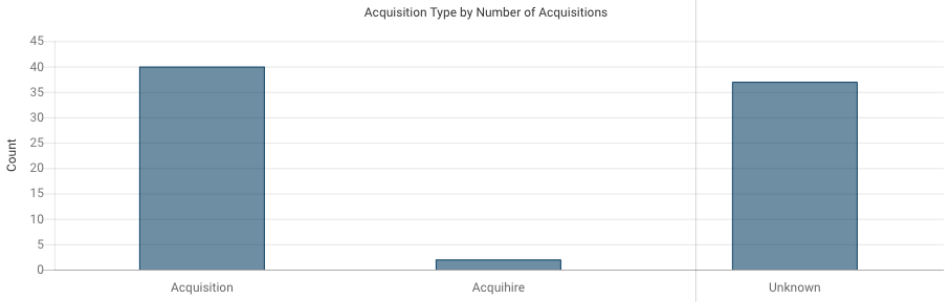
Amazon > Acquisitions

Acquisitions

Number of Acquisitions 79

Amazon has acquired 79 organizations. Their most recent acquisition was PillPack on Jun 28, 2018. They acquired PillPack for $1B.

Which types of acquisition does this organization make most frequently? HIDE



Acquired Organization Name	Announced Date	Price	Transactior
PillPack	Jun 28, 2018	$1B	PillPack a Amazon
Ring	Feb 27, 2018	$1B	Ring acqu Amazon
Blink	Dec 22, 2017	–	Blink acqu Amazon
Immedia	Dec 21, 2017	–	Immedia a Amazon
Body Labs	Oct 4, 2017	–	Body Labs Amazon
Graphiq	Jul 20, 2017	–	Graphiq a Amazon
GameSparks	Jul 8, 2017	$10M	GameSpa by Amazo
Whole Foods Market	Jun 16, 2017	$13.7B	Whole Foo acquired b
Souq	Mar 27, 2017	$580M	Souq acqu Amazon
harvest.ai	Jan 9, 2017	$20M	harvest.ai Amazon
Biba	Nov 23, 2016	–	Biba acqu Amazon
GoButler / Angel.ai	Sep 20, 2016		GoButler / acquired b

Figure B.2: The Amazon acquisition list

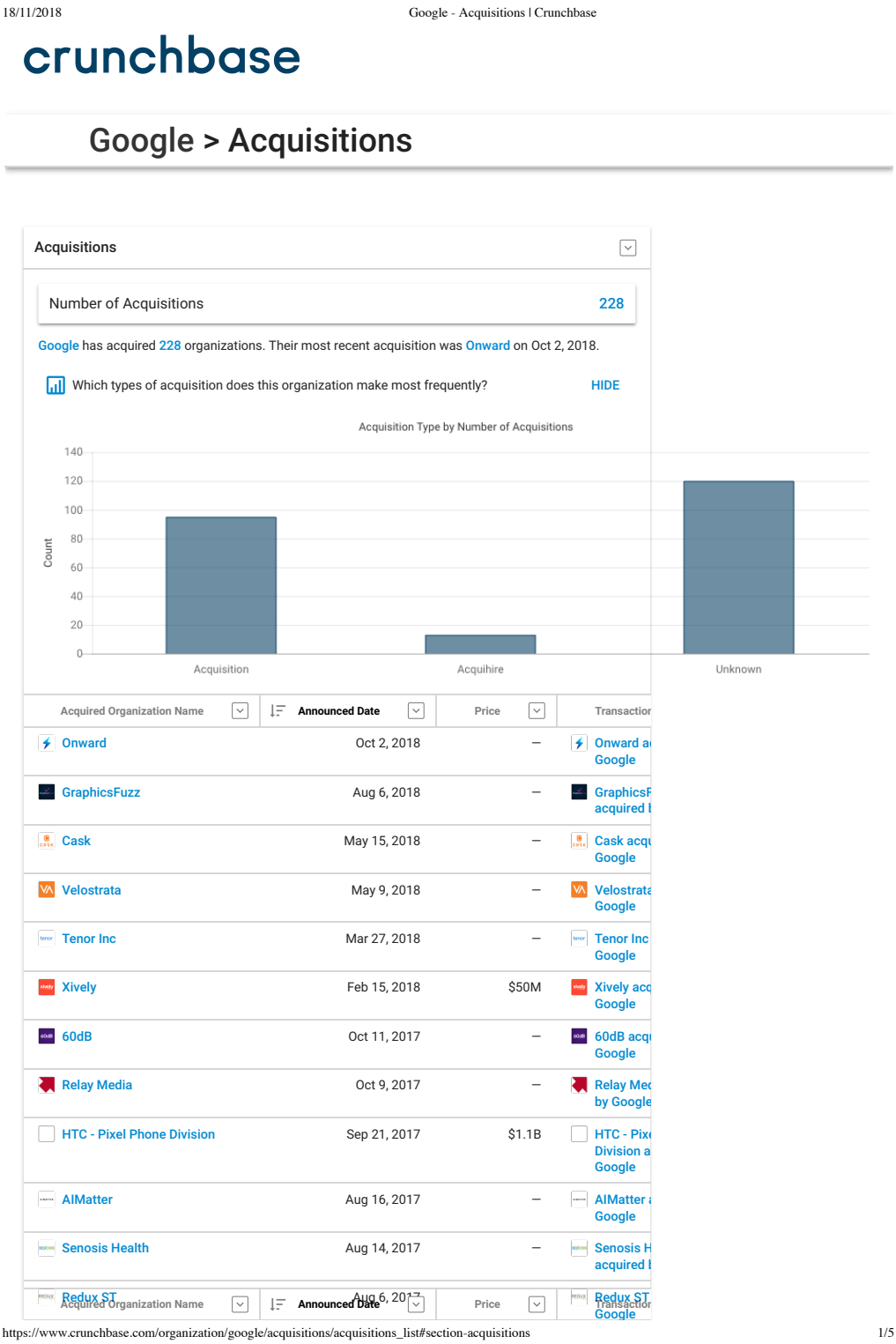


Figure B.3: The Google acquisition list

crunchbase

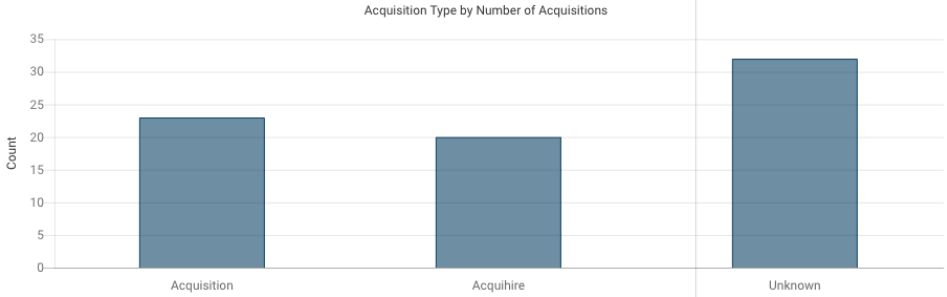
Facebook > Acquisitions

Acquisitions

Number of Acquisitions 75

Facebook has acquired 75 organizations. Their most recent acquisition was Refdash on Oct 1, 2018.

Which types of acquisition does this organization make most frequently? HIDE



Acquired Organization Name	Announced Date	Price	Transaction
Refdash	Oct 1, 2018	–	Refdash a Facebook
Vidpresso	Aug 14, 2018	–	Vidpresso Facebook
RedKix	Jul 26, 2018	–	RedKix ac Facebook
Bloomsbury AI	Jul 2, 2018	–	Bloomsbu acquired
Confirm.io (Acquired by Facebook)	Jan 23, 2018	–	Confirm.io by Facebo by Facebo
tbh	Oct 16, 2017	–	tbh acqui Facebook
Fayteq AG	Aug 11, 2017	–	Fayteq AG Facebook
Ozlo	Jul 31, 2017	–	Ozlo acqu Facebook
Source3	Jul 24, 2017	–	Source3 a Facebook
CrowdTangle	Nov 11, 2016	–	CrowdTan by Facebo
Nascent Objects Inc	Sep 19, 2016	–	Nascent C acquired
Eyegroove	Aug 8, 2016	–	Eyegroove Facebook

Figure B.4: The Facebook acquisition list

Appendix C

Acquisition valuation

The list is the 87 companies that are being acquired and have a valuation on Crunchbase. In the table, the information listed on the tables include the target company, the acquiring company, the date of the acquirement announcement, and valuation.

Table C.1: The 87 companies that are acquired with a valuation listed on Crunchbase

Rank	Acquire company	Acquired company	**Announced date**	**Valuation**
1	**Google**	**Xively**	02/15/2018	$50M
2	**Google**	HTC - Pixel Phone Division	09/21/2017	$1.1B
3	**Google**	**Apigee**	09/08/2016	$625M
4	**Google**	**Bebop**	11/19/2015	$380.2M
5	**Google**	**Oyster**	09/21/2015	$30M
6	**Google**	**Terra Bella**	06/10/2014	$500M
7	**Google**	**Divide**	05/20/2014	$120M
8	**Google**	**DeepMind**	01/26/2014	$500M
9	**Google**	**Nest Labs**	01/2014	$3.2B
10	**Google**	**Schaft**	12/02/2013	$20M
11	**Google**	**FlexyCore**	10/22/2013	$23M
12	**Google**	**Bump Technologies**	09/16/2013	$30M
13	**Google**	**Waze**	06/11/2013	$1.3B
14	**Google**	**Wildfire**	07/31/2012	$350M
15	**Google**	**Meebo**	06/04/2012	$100M
16	**Google**	**Motorola Mobility**	05/22/2012	$12.5B
17	**Google**	**Zagat**	10/26/2011	$151M

Table C.2: The 87 companies that are acquired with a valuation on listed Crunchbase

Rank	Acquire company	**Acquired company**	Announced date	**Valuation**
18	**Google**	**Admeld**	06/2011	$400M
19	**Google**	**Modu**	05/20/2011	$4.9M
20	**Google**	**ITA Software**	04/13/2011	$700M
21	**Google**	**BeatThatQuote.com**	03/07/2011	$37.7M
22	**Google**	**Widevine Technologies**	12/03/2010	$160M
23	**Google**	MentorWave Technologies	09/12/2010	$12M
24	**Google**	**Slide**	08/06/2010	$182M
25	**Google**	**Invite media**	06/02/2010	$81M
26	**Google**	**Gizmofive**	11/09/2009	$30M
27	**Google**	**On2 Technologies**	02/19/2009	$130M
28	**Google**	**Zao Begun**	07/18/2008	$140M
29	**Google**	**Jaiku**	10/09/2007	$12M
30	**Google**	**GrandCentral**	07/01/2007	$60M
31	**Google**	**PeakStream**	06/05/2007	$20.3M
32	**Google**	**FeedBurner**	05/23/2007	$100M
33	**Google**	**Adscape**	02/16/2007	$23M
34	**Google**	**Endoxon**	12/18/2006	$28M
35	**Google**	**YouTube**	10/2006	$1.7B
36	**Google**	**dMarc Broadcasting**	01/01/2006	$102M
37	**Google**	**Android**	08/16/2005	$50M
38	**Google**	**Applied Semantics**	04/28/2003	$102M
39	**Apple**	**Dialog Semiconductor**	10/11/2018	$600M
40	**Apple**	**Shazam Entertainment**	12/11/2017	$400M
41	**Apple**	**Spektral**	12/01/2017	$30M
42	**Apple**	**Vrvana**	11/21/2017	$30M
43	**Apple**	**Lattice**	05/14/2017	$200M
44	**Apple**	**RealFace**	02/19/2017	$2M
45	**Apple**	**Gliimpse**	08/22/2016	$200M
46	**Apple**	**Mapsense**	09/15/2015	$25M

Table C.3: The 87 companies that are acquired with a valuation on listed Crunchbase

Rank	Acquire company	Acquired company	Announced date	Valuation
47	**Apple**	**Concept.io**	07/28/2014	$30M
48	**Apple**	**Beats Electronics**	05/28/2014	$3B
49	**Apple**	**Topsy Labs**	12/02/2013	$200M
50	**Apple**	**HopStop.com**	07/19/2013	$1B
51	**Apple**	**Color Labs Inc**	11/12/2012	$7M
52	**Apple**	**Lala**	12/05/2009	$17M
53	**Apple**	**Xemplar Education**	01/08/1999	$5M
54	**Apple**	**Next**	12/20/1996	$400M
55	**Amazon**	**PillPack**	06/28/2018	$1B
56	**Amazon**	**Ring**	02/27/2018	$1B
57	**Amazon**	**GameSparks**	07/08/2017	$10M
58	**Amazon**	**Whole Foods Market**	06/16/2017	$13.7B
59	**Amazon**	**Souq**	03/27/2017	$580M
60	**Amazon**	**harvest.ai**	01/09/2017	$20M
61	**Amazon**	**Elemental Technologies**	09/03/2015	$500M
62	**Amazon**	**Annapurna Labs**	01/22/2015	$370M
63	**Amazon**	**Twitch**	08/25/2014	$970M
64	**Amazon**	**Evi**	04/17/2013	$26M
65	**Amazon**	**Kiva Systems**	03/19/2012	$775M
66	**Amazon**	**Quidsi**	11/08/2010	$545M
67	**Amazon**	**BuyVIP**	10/07/2010	$70M
68	**Amazon**	**Zappos**	07/22/2009	$1.2B
69	**Amazon**	**LOVEFILM**	02/04/2008	$200M
70	**Amazon**	**Joyo.com**	08/19/2004	$75M
71	**Amazon**	Convergence Corporation	10/04/1999	$23M
72	**Amazon**	**Accept.com**	06/11/1999	$101.7M
73	**Amazon**	**Alexa.com**	05/23/1999	$250M
74	**Facebook**	**liveRail**	07/02/2014	$500M
75	**Facebook**	**Ascenta**	03/28/2014	$20M

Table C.4: The 87 companies that are acquired with a valuation on listed Crunchbase

Rank	Acquire company	Acquired company	**Announced date**	**Valuation**
76	**Facebook**	**Oculus**	03/25/2014	$2B
77	**Facebook**	**WhatsApp**	02/19/2014	$19B
78	**Facebook**	**Branch**	01/13/2014	$15M
79	**Facebook**	**Onavo**	10/14/2013	$120M
80	**Facebook**	**Parse**	04/25/2013	$85M
81	**Facebook**	**Instagram**	04/09/2012	$1B
82	**Facebook**	**Gowalla**	12/02/2011	$3M
83	**Facebook**	**Snaptu**	03/20/2011	$70M
84	**Facebook**	**NextStop**	09/25/2010	$2.5M
85	**Facebook**	**Chai Labs**	08/15/2010	$10M
86	**Facebook**	**Hot potato**	07/28/2010	$10M
87	**Facebook**	**FriendFeed**	08/10/2009	$50M

Bibliography

[AAA14] Adebiyi A Ariyo, Adewumi O Adewumi, and Charles K Ayo. Stock price prediction using the arima model. In *2014 UKSim-AMSS 16th International Conference on Computer Modelling and Simulation*, pages 106–112. IEEE, 2014.

[AAB+16] Martín Abadi, Ashish Agarwal, Paul Barham, Eugene Brevdo, Zhifeng Chen, Craig Citro, Greg S Corrado, Andy Davis, Jeffrey Dean, Matthieu Devin, et al. Tensorflow: Large-scale machine learning on heterogeneous distributed systems. *NIPS*, page 23, 2016.

[ABS19] Patrick Augustin, Menachem Brenner, and Marti G Subrahmanyam. Informed options trading prior to takeover announcements: Insider trading? *Management Science*, page 43, 2019.

[ACL+08] Peter JD Andrews, Giuseppe Citerio, Luca Longhi, Kees Polderman, Juan Sahuquillo, Peter Vajkoczy, Neuro-Intensive Care, Emergency Medicine (NICEM) Section of the European Society of Intensive Care Medicine, et al. Nicem consensus on neurological monitoring in acute neurological disease. *Intensive care medicine*, 34(8):1362–1370, 2008.

[AFL84] Gert Assmus, John U Farley, and Donald R Lehmann. How advertising affects sales: Meta-analysis of econometric results. *Journal of Marketing Research*, pages 65–74, 1984.

[Alt92] Naomi S Altman. An introduction to kernel and nearest-neighbor nonparametric regression. *The American Statistician*, 46(3):175–185, 1992.

[AMM05] Giuliano Armano, Michele Marchesi, and Andrea Murru. A hybrid genetic-neural architecture for stock indexes forecasting. *Information Sciences*, 170(1):3–33, 2005.

[AN12] Anup Agrawal and Tareque Nasser. Insider trading in takeover targets. *Journal of Corporate Finance*, 18(3):598–625, 2012.

[AP88] Anat R Admati and Paul Pfleiderer. A theory of intraday patterns: Volume and price variability. *The Review of Financial Studies*, 1(1):3–40, 1988.

[Bai18] Alex Bailin. World economy at risk of another financial crash, says imf. *Guardian*, 2018.

[BCB14] Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. Neural machine translation by jointly learning to align and translate. *ICLR*, 2014.

[BCKW15] Charles Blundell, Julien Cornebise, Koray Kavukcuoglu, and Daan Wierstra. Weight uncertainty in neural networks. *Phys. Rev.*, 47:777–780, 2015.

[BG15] Popa R.A. Tu S. Bost, R. and S. Goldwasser. Machine learning classification over encrypted data. *NDSS*, page p.4325, 2015.

[BJRL15] George EP Box, Gwilym M Jenkins, Gregory C Reinsel, and Greta M Ljung. *Time series analysis: forecasting and control*. John Wiley & Sons, 2015.

[BMZ11] Johan Bollen, Huina Mao, and Xiaojun Zeng. Twitter mood predicts the stock market. *Journal of computational science*, 2(1):1–8, 2011.

[BN+13] KS BOCHEER, H NANJEGOWDA, et al. The impact of celebrity advertisement on indian customers. *CLEAR International Journal of Research in Commerce & Management*, 4(5), 2013.

[BPTG15] Raphael Bost, Raluca Ada Popa, Stephen Tu, and Shafi Goldwasser. Machine learning classification over encrypted data. In *NDSS*, volume Vol.4324, page p.4325, 2015.

[Bra14] Sarsh Brady. What percent of revenue do publicly traded companies spend on marketing and sales. *Vital*, v.12:23, 2014.

[BSW72] Joan Blatt, Lyle Spencer, and Scott Ward. A cognitive development study of childrens reactions to television advertising. *Television and social behavior*, 4:452–467, 1972.

[BWC15] M Bejaei, K Wiseman, and KM Cheng. Developing logistic regression models using purchase attributes and demographics to predict the probability of purchases of regular and specialty eggs. *British poultry science*, 56(4):425–435, 2015.

[BYR17] Wei Bao, Jun Yue, and Yulei Rao. A deep learning framework for financial time series using stacked autoencoders and long-short term memory. *PloS one*, 12(7):e0180944, 2017.

[CCG05] Charles Cao, Zhiwu Chen, and John M Griffin. Informational content of option volume prior to takeovers. *The Journal of Business*, 78(3):1073–1109, 2005.

[CdBV+14] Yun-Hsuan Chen, Maaike Op de Beeck, Luc Vanderheyden, Evelien Carrette, Vojkan Mihajlović, Kris Vanstreels, Bernard Grundlehner, Stefanie Gadeyne, Paul Boon, and Chris Van Hoof. Soft, comfortable polymer dry electrodes for high quality ecg and eeg recording. *Sensors*, 14(12):23758–23780, 2014.

[CGBL+17] Melissa Chase, Ran Gilad-Bachrach, Kim Laine, Kristin E Lauter, and Peter Rindal. Private collaborative neural network learning. *IACR Cryptology ePrint Archive*, 2017:762, 2017.

[CL12] Md Faisal Mahbub Chowdhury and Alberto Lavelli. Combining tree structures, flat features and patterns for biomedical relation extraction. In *Proceedings of the 13th Conference of the European Chapter of the Association for Computational Linguistics*, pages 420–429. Association for Computational Linguistics, 2012.

[Cla76] Darral G Clarke. Econometric measurement of the duration of advertising effect on sales. *Journal of Marketing Research*, pages 345–357, 1976.

[Cla17] Paul Clarke. How artificial intelligence will eat our finance job-and how to survive. *efinance*, page 12, 2017.

[CM17] Juan Carrasquilla and Roger G Melko. Machine learning phases of matter. *Nature Physics*, 13(5):431, 2017.

[CMMS18] Michele Costa, Laurits Marschall, Seyed Hamed Mirsadeghi, and Alessandro de Sanctis. Investigation of sentiment importance on intraday stock returns. page 90, 2018.

[CS00] Tarun Chordia and Bhaskaran Swaminathan. Trading volume and cross-autocorrelations in stock returns. *The Journal of Finance*, 55(2):913–935, 2000.

[CT15] Yahya Eru Cakra and Bayu Distiawan Trisedya. Stock price prediction using linear regression based on sentiment analysis. In *2015 International Conference on Advanced Computer Science and Information Systems (ICACSIS)*, pages 147–154. IEEE, 2015.

[CY19] Thomas J Chemmanur and An Yan. Advertising, attention, and stock returns. *Quarterly Journal of Finance*, page p.1950009, 2019.

[Dah02] Micael Dahlen. Thinking and feeling on the world wide web: the impact of product type and time on world wide web advertising effectiveness. *Journal of Marketing Communications*, 8(2):115–125, 2002.

[DBT87] Werner FM De Bondt and Richard H Thaler. Further evidence on investor overreaction and stock market seasonality. *The Journal of Finance*, 42(3):557–581, 1987.

[Dec86] Rina Dechter. *Learning while searching in constraint-satisfaction problems*. University of California, Computer Science Department, Cognitive Systems Laboratory, 1986.

[DPGA02] Patrick De Pelsmacker, Maggie Geuens, and Pascal Anckaert. Media context and advertising effectiveness: The role of context appreciation and context/ad similarity. *Journal of Advertising*, 31(2):49–61, 2002.

[DSM+17] Hao Dong, Akara Supratak, Luo Mai, Fangde Liu, Axel Oehmichen, Simiao Yu, and Yike Guo. TensorLayer: A Versatile Library for Efficient Deep Learning Development. *ACM Multimedia*, 2017.

[DSP+18] Hao Dong, Akara Supratak, Wei Pan, Chao Wu, Paul M Matthews, and Yike Guo. Mixed neural network approach for temporal sleep stage classification. *IEEE Transactions on Neural Systems and Rehabilitation Engineering*, 26(2):324–333, 2018.

[Duc96] Robert H Ducoffe. advertising value and advertising on the web-blog@ management. *Journal of advertising research*, page 21, 1996.

[DV16] Vincent Dumoulin and Francesco Visin. A guide to convolution arithmetic for deep learning. *arXiv preprint arXiv:1603.07285*, 2016.

[DY14a] L. Deng and D. Yu. Deep learning: methods and applications. *Foundations and Trends in Signal Processing*, pages 197–387, 2014.

[DY+14b] Li Deng, Dong Yu, et al. Deep learning: methods and applications. *Foundations and Trends® in Signal Processing*, 7(3–4):197–387, 2014.

[E+03] Bradley Efron et al. Second thoughts on the bootstrap. *Statistical Science*, 18(2):135–140, 2003.

[edm16] Divvala S. Girshick R. edmon, J. You only look once: Unified, real-time object detection. *InProceedings of the IEEE conference on computer vision and pattern recognition*, pages 779–788, 2016.

[Efr82] Bradley Efron. *The jackknife, the bootstrap, and other resampling plans*, volume 38. Siam, 1982.

[EOS98] David Easley, Maureen O'hara, and Pulle Subrahmanya Srinivas. Option volume and stock prices: Evidence on where informed traders trade. *The Journal of Finance*, 53(2):431–465, 1998.

[ES98] Michael T Elliott and Paul Surgi Speck. Consumer perceptions of advertising clutter and its impact across various media. *Journal of advertising research*, 38(1):29–30, 1998.

[ET94] Bradley Efron and Robert J Tibshirani. *An introduction to the bootstrap*. CRC press, 1994.

[EVDHM05] Robert J Elliott, John Van Der Hoek*, and William P Malcolm. Pairs trading. *Quantitative Finance*, 5(3):271–276, 2005.

[FLLL16] Yin Fan, Xiangju Lu, Dian Li, and Yuanliu Liu. Video-based emotion recognition using cnn-rnn and c3d hybrid networks. In *Proceedings of the 18th ACM International Conference on Multimodal Interaction*, pages 445–450. ACM, 2016.

[Fri18] Zack Friedman. The next financial crisis will look like this. *Forbes*, 2018.

[FSR+91] Paul M Fischer, Meyer P Schwartz, John W Richards, Adam O Goldstein, and Tina H Rojas. Brand logo recognition by children aged 3 to 6 years: Mickey mouse and old joe the camel. *Jama*, 266(22):3145–3148, 1991.

[Gal16] Yarin Gal. Uncertainty in deep learning. *University of Cambridge*, 2016.

[GBB11] Xavier Glorot, Antoine Bordes, and Yoshua Bengio. Deep sparse rectifier neural networks. In *Proceedings of the fourteenth international conference on artificial intelligence and statistics*, pages 315–323, 2011.

[GBCB16] Ian Goodfellow, Yoshua Bengio, Aaron Courville, and Yoshua Bengio. *Deep learning*, volume 1. MIT press Cambridge, 2016.

[GBDL+16] Ran Gilad-Bachrach, Nathan Dowlin, Kim Laine, Kristin Lauter, Michael Naehrig, and John Wernsing. Cryptonets: Applying neural networks to encrypted data with high throughput and accuracy. In *International Conference on Machine Learning*, pages 201–210, 2016.

[GG83] Mustafa N Gultekin and N Bulent Gultekin. Stock market seasonality: International evidence. *Journal of financial economics*, 12(4):469–481, 1983.

[GG16] Yarin Gal and Zoubin Ghahramani. Dropout as a bayesian approximation: Representing model uncertainty in deep learning. In *international conference on machine learning*, pages 1050–1059, 2016.

[GH88] David E Goldberg and John H Holland. Genetic algorithms and machine learning. *Machine learning*, 3(2):95–99, 1988.

[GKQ07] Brian J Gaines, James H Kuklinski, and Paul J Quirk. The logic of the survey experiment reexamined. *Political Analysis*, 15(1):1–20, 2007.

[GLF+09] Alex Graves, Marcus Liwicki, Santiago Fernández, Roman Bertolami, Horst Bunke, and Jürgen Schmidhuber. A novel connectionist system for unconstrained handwriting recognition. *IEEE transactions on pattern analysis and machine intelligence*, 31(5):855–868, 2009.

[GLN12] Thore Graepel, Kristin Lauter, and Michael Naehrig. Ml confidential: Machine learning on encrypted data. In *International Conference on Information Security and Cryptology*, pages 1–21. Springer, 2012.

[GMK98] Dhruv Grewal, Kent B Monroe, and Ramayya Krishnan. The effects of price-comparison advertising on buyers' perceptions of acquisition value, transaction value, and behavioral intentions. *The Journal of Marketing*, pages 46–59, 1998.

[Gol90] Marvin E Goldberg. A quasi-experiment assessing the effectiveness of tv advertising directed to children. *Journal of Marketing Research*, pages 445–454, 1990.

[GPAM+14] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial nets. In *Advances in neural information processing systems*, pages 2672–2680, 2014.

[GSB16] Raja Giryes, Guillermo Sapiro, and Alexander M Bronstein. Deep neural networks with random gaussian weights: A universal classification strategy? *IEEE Trans. Signal Processing*, 64(13):3444–3457, 2016.

[GW87] Devra L Golbe and Lawrence J White. Mergers and acquisitions in the us economy: an aggregate and historical overview. In *Mergers and acquisitions*, pages 25–48. University of Chicago Press, 1987.

[GWFM+13] Ian J Goodfellow, David Warde-Farley, Mehdi Mirza, Aaron Courville, and Yoshua Bengio. Maxout networks. *arXiv preprint arXiv:1302.4389*, 2013.

[Han09] Chris Hans. Bayesian lasso regression. *Biometrika*, 96(4):835–845, 2009.

[Har86] Lawrence Harris. A transaction data study of weekly and intradaily patterns in stock returns. *Journal of financial economics*, 16(1):99–117, 1986.

[Haz01] Michiel Hazewinkel. Affine transformation. *Encyclopedia of Mathematics, Springer, ISBN 978-1-55608-010-4, 2001*, 2001.

[HDM+09] Jerayr Haleblian, Cynthia E Devers, Gerry McNamara, Mason A Carpenter, and Robert B Davison. Taking stock of what we know about mergers and acquisitions: A review and research agenda. *Journal of management*, 35(3):469–502, 2009.

[HJB84] Michael Heideman, Don Johnson, and C Burrus. Gauss and the history of the fast fourier transform. *IEEE ASSP Magazine*, 1(4):14–21, 1984.

[Ho95] Tin Kam Ho. Random decision forests. In *Document analysis and recognition, 1995., proceedings of the third international conference on*, volume 1, pages 278–282. IEEE, 1995.

[HS97] Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. *Neural computation*, 9(8):1735–1780, 1997.

[HSK+12] Geoffrey E Hinton, Nitish Srivastava, Alex Krizhevsky, Ilya Sutskever, and Ruslan R Salakhutdinov. Improving neural networks by preventing co-adaptation of feature detectors. *arXiv preprint arXiv:1207.0580*, 2012.

[HW68] David H Hubel and Torsten N Wiesel. Receptive fields and functional architecture of monkey striate cortex. *The Journal of physiology*, 195(1):215–243, 1968.

[IS15] Sergey Ioffe and Christian Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. *arXiv preprint arXiv:1502.03167*, 2015.

[IT07] Huseyin Ince and Theodore B Trafalis. Kernel principal component analysis and support vector machines for stock price prediction. *Iie Transactions*, 39(6):629–637, 2007.

[Jeg91] Narasimhan Jegadeesh. Seasonality in stock price mean reversion: Evidence from the us and the uk. *The Journal of Finance*, 46(4):1427–1444, 1991.

[JH09] Amit M Joshi and Dominique M Hanssens. Movie advertising and the stock market valuation of studios: a case of great expectations?. *Marketing Science*, 28(2):239–250, 2009.

[JJ88] Prem C Jain and Gun-Ho Joh. The dependence between hourly prices and trading volume. *Journal of Financial and Quantitative Analysis*, 23(3):269–283, 1988.

[JKP94] George H John, Ron Kohavi, and Karl Pfleger. Irrelevant features and the subset selection problem. In *Machine Learning Proceedings 1994*, pages 121–129. Elsevier, 1994.

[Joh15] Bradley Johnson. Big spenders on a budget: What the top 200 us advertisers are doing to spend smarter. *Advertising Age*, 5, 2015.

[JPL99] Han-Il Jeong, Jinwoo Park, and Robert C Leachman. A batch splitting method for a job shop scheduling problem in an mrp environment. *International Journal of Production Research*, 37(15):3583–3598, 1999.

[JS12] Travis L Johnson and Eric C So. The option to stock volume ratio and future returns. *Journal of Financial Economics*, 106(2):262–286, 2012.

[Kar87] Jonathan M Karpoff. The relation between price changes and trading volume: A survey. *Journal of Financial and quantitative Analysis*, 22(1):109–126, 1987.

[KB15] Diederik Kingma and Jimmy Ba. Adam: A method for stochastic optimization. iclr 2015. *arXiv preprint arXiv:1412.6980*, 2015.

[KBB11] Yakup Kara, Melek Acar Boyacioglu, and Ömer Kaan Baykan. Predicting direction of stock price index movement using artificial neural networks and support vector machines: The sample of the istanbul stock exchange. *Expert systems with Applications*, 38(5):5311–5319, 2011.

[KCH01] Ren Jie Kuo, CH Chen, and YC Hwang. An intelligent stock trading decision support system through integration of genetic algorithm based fuzzy neural network and artificial neural network. *Fuzzy sets and systems*, 118(1):21–45, 2001.

[Kei83] Donald B Keim. Size-related anomalies and stock return seasonality: Further empirical evidence. *Journal of financial economics*, 12(1):13–32, 1983.

[KH09] Alex Krizhevsky and Geoffrey Hinton. Learning multiple layers of features from tiny images. *Techincal Report*, Vol.1(No.4), 2009.

[Kli99] Wolfgang Klimesch. Eeg alpha and theta oscillations reflect cognitive and memory performance: a review and analysis. *Brain research reviews*, 29(2-3):169–195, 1999.

[KM07] Yung-Keun Kwon and Byung-Ro Moon. A hybrid neurogenetic approach for stock forecasting. *IEEE transactions on neural networks*, 18(3):851–864, 2007.

[KSH12] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. In *Advances in neural information processing systems*, pages 1097–1105, 2012.

[KSN06] Nguyen Lu Dang Khoa, Kazutoshi Sakakibara, and Ikuko Nishikawa. Stock price forecasting using back propagation neural networks with time and profit based adjusted weight factors. In *SICE-ICASE, 2006. International Joint Conference*, pages 5484–5488. IEEE, 2006.

[KWW92] Vinay Kanetkar, Charles B Weinberg, and Doyle L Weiss. Price sensitivity and television advertising exposures: Some empirical findings. *Marketing Science*, 11(4):359–371, 1992.

[L+89] Yann LeCun et al. Generalization and network design strategies. *Connectionism in perspective*, pages 143–155, 1989.

[LC14] Jiahan Li and Weiye Chen. Forecasting macroeconomic time series: Lasso-based approaches and their forecast combinations with dynamic factor models. *International Journal of Forecasting*, 30(4):996–1015, 2014.

[LH02] Sourina O. Liu, Y. and M.R. Hafiyyandi. Eeg-based emotion-adaptive advertising. *Humaine Association Conference on Affective Computing and Intelligent Interaction*, pages 731–870, 2002.

[LP12] Claudio Lucchiari and Gabriella Pravettoni. The effect of brand on eeg modulation. *Swiss Journal of Psychology*, 2012.

[LPM15] Minh-Thang Luong, Hieu Pham, and Christopher D Manning. Effective approaches to attention-based neural machine translation. *arXiv preprint arXiv:1508.04025*, 2015.

[LR09] Randall Lewis and David Reiley. Retail advertising works! measuring the effects of advertising on sales via a controlled experiment on yahoo! 2009.

[Lüt05] Helmut Lütkepohl. *New introduction to multiple time series analysis*. Springer Science & Business Media, 2005.

[LW+02] Andy Liaw, Matthew Wiener, et al. Classification and regression by randomforest. *R news*, 2(3):18–22, 2002.

[LW15] Xiangang Li and Xihong Wu. Constructing long short-term memory based deep recurrent neural networks for large vocabulary speech recognition. In *Acoustics, Speech and Signal Processing (ICASSP), 2015 IEEE International Conference on*, pages 4520–4524. IEEE, 2015.

[LWW+09] Yuan-Pin Lin, Chi-Hong Wang, Tien-Lin Wu, Shyh-Kang Jeng, and Jyh-Horng Chen. Eeg-based emotion recognition in music listening: A comparison of schemes for multiclass support vector machine. In *Acoustics, Speech and Signal Processing, 2009. ICASSP 2009. IEEE International Conference on*, pages 489–492. IEEE, 2009.

[LYS09] Xiaowei Lin, Zehong Yang, and Yixu Song. Short-term stock price prediction based on echo state networks. *Expert systems with applications*, 36(3):7313–7317, 2009.

[LZW+18] Binbing Liao, Jingqing Zhang, Chao Wu, Douglas McIlwraith, Tong Chen, Shengwen Yang, Yike Guo, and Fei Wu. Deep sequence learning with auxiliary information for traffic prediction. *arXiv preprint arXiv:1806.07380*, 2018.

[McQ02] Edward F McQuarrie. The development, change and transformation of rhetorical style in magazine advertisements 1954–1999. *Journal of Advertising*, 31(4):1–13, 2002.

[MGB04] Dominika Maison, Anthony G Greenwald, and Ralph H Bruin. Predictive validity of the implicit association test in studies of brands, consumer attitudes, and behavior. *Journal of consumer psychology*, 14(4):405–415, 2004.

[MHG+14] Volodymyr Mnih, Nicolas Heess, Alex Graves, et al. Recurrent models of visual attention. In *Advances in neural information processing systems*, pages 2204–2212, 2014.

[Mic13] MG Michael. *Uberveillance and the Social Implications of Microchip Implants: Emerging Technologies: Emerging Technologies*. IGI Global, 2013.

[MO97] Maureen McNichols and Patricia C O'Brien. Self-selection and analyst coverage. *Journal of Accounting Research*, 35:167–199, 1997.

[MPV15] Douglas C Montgomery, Elizabeth A Peck, and G Geoffrey Vining. *Introduction to linear regression analysis*. John Wiley & Sons, 2015.

[MRS+10] Murugappan Murugappan, Nagarajan Ramachandran, Yaacob Sazali, et al. Classification of human emotion from eeg using discrete wavelet transform. *Journal of Biomedical Science and Engineering*, 3(04):390, 2010.

[MSA00] David McMillan, Alan Speight, and Owain Apgwilym. Forecasting uk stock market volatility. *Applied Financial Economics*, 10(4):435–448, 2000.

[Nas07] Nasser M Nasrabadi. Pattern recognition and machine learning. *Journal of electronic imaging*, 16(4):049901, 2007.

[NB08] Jaime Noriega and Edward Blair. Advertising to bilinguals: Does the language of advertising influence the nature of thoughts? *Journal of Marketing*, 72(5):69–83, 2008.

[Ng04] Andrew Y Ng. Feature selection, l 1 vs. l 2 regularization, and rotational invariance. In *Proceedings of the twenty-first international conference on Machine learning*, page 78. ACM, 2004.

[NKNW96] John Neter, Michael H Kutner, Christopher J Nachtsheim, and William Wasserman. *Applied linear statistical models*, volume 4. Irwin Chicago, 1996.

[NSAH12] Ghulam Shabbir Khan Niazi, Javaria Siddiqui, Burhan Alishah, and Ahmed Imran Hunjra. Effective advertising and its influence on consumer buying behavior. 2012.

[ORWC10] Rafal Ohme, Dorota Reykowska, Dawid Wiener, and Anna Choromanska. Application of frontal eeg asymmetry to advertising research. *Journal of Economic Psychology*, 31(5):785–793, 2010.

[OS11] Michael Ostrovsky and Michael Schwarz. Reserve prices in internet advertising auctions: A field experiment. In *Proceedings of the 12th ACM conference on Electronic commerce*, pages 59–60. ACM, 2011.

[PA97] Dan Padgett and Douglas Allen. Communicating experiences: A narrative approach to creating service brand image. *Journal of advertising*, 26(4):49–62, 1997.

[PC06] J. Pawle and P. Cooper. Measuring emotionlovemarks, the future beyond brands. *Journal of advertising research*, 46(1):38–48, 2006.

[PD06] Karolien Poels and Siegfried Dewitte. How to capture the heart? reviewing 20 years of emotion measurement in advertising. *Journal of Advertising Research*, 46(1):18–37, 2006.

[PDC+94] Walter S Pritchard, Dennis W Duke, Kerry L Coburn, Norman C Moore, Karen A Tucker, Michael W Jann, and Russell M Hostetler. Eeg-based, neural-net predictive classification of alzheimer's disease versus control subjects is augmented by non-linear eeg measures. *Electroencephalography and clinical Neurophysiology*, 91(2):118–130, 1994.

[PL05] Ping-Feng Pai and Chih-Sheng Lin. A hybrid arima and support vector machines model in stock price forecasting. *Omega*, 33(6):497–505, 2005.

[PM02] Barbara J Phillips and Edward F McQuarrie. The development, change, and transformation of rhetorical style in magazine advertisements 1954–1999. *Journal of Advertising*, 31(4):1–13, 2002.

[PMS13] Tobias Preis, Helen Susannah Moat, and H Eugene Stanley. Quantifying trading behavior in financial markets using google trends. *Scientific reports*, 3:1684, 2013.

[PP06] Jun Pan and Allen M Poteshman. The information in option volume for future stock prices. *The Review of Financial Studies*, 19(3):871–908, 2006.

[Pre18] Gil Press. Top 10 hot artificial intelligence (ai) technologies. *Forbes*, 2018.

[Qui87] J. Ross Quinlan. Simplifying decision trees. *International journal of man-machine studies*, 27(3):221–234, 1987.

[RDGF16] Joseph Redmon, Santosh Divvala, Ross Girshick, and Ali Farhadi. You only look once: Unified, real-time object detection. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 779–788, 2016.

[RHW85] David E Rumelhart, Geoffrey E Hinton, and Ronald J Williams. Learning internal representations by error propagation. Technical report, California Univ San Diego La Jolla Inst for Cognitive Science, 1985.

[RJ15] Girshick R Farhadi A Redmon J, Divvala S. Ml confidential: Machine learning on encrypted data. in international conference on information security and cryptology. *In International Conference on Information Security and Cryptology*, pages 1–21, 2015.

[RKJ76] Michael S Rozeff and William R Kinney Jr. Capital market seasonality: The case of stock returns. *Journal of financial economics*, 3(4):379–402, 1976.

[RMBA15] Sanjiban Sekhar Roy, Dishant Mittal, Avik Basu, and Ajith Abraham. Stock market forecasting using lasso linear regression model. In *Afro-European Conference for Industrial Advancement*, pages 371–381. Springer, 2015.

[RMC15] Alec Radford, Luke Metz, and Soumith Chintala. Unsupervised representation learning with deep convolutional generative adversarial networks. *arXiv preprint arXiv:1511.06434*, 2015.

[Roo13] Ben Rooney. 8 things to know about the 2013 bull market. *Business CNN*, 2013.

[Ros51] Arnold M Rose. Rumor in the stock market. *Public Opinion Quarterly*, 15(3):461–486, 1951.

[Ros61] Frank Rosenblatt. Principles of neurodynamics. perceptrons and the theory of brain mechanisms. Technical report, CORNELL AERONAUTICAL LAB INC BUFFALO NY, 1961.

[RSS10] Richard Roll, Eduardo Schwartz, and Avanidhar Subrahmanyam. O/s: The relative trading activity in options and stock. *Journal of Financial Economics*, 96(1):1–17, 2010.

[RSTK03] Jason D Rennie, Lawrence Shih, Jaime Teevan, and David R Karger. Tackling the poor assumptions of naive bayes text classifiers. In *Proceedings of the 20th international conference on machine learning (icml-03)*, pages 616–623, 2003.

[SBB90] Jessica Severn, George E Belch, and Michael A Belch. The effects of sexual and non-sexual advertising appeals and information level on cognitive processing and communication effectiveness. *Journal of advertising*, 19(1):14–22, 1990.

[SC09] Robert P Schumaker and Hsinchun Chen. Textual analysis of stock market prediction using breaking financial news: The azfin text system. *ACM Transactions on Information Systems (TOIS)*, 27(2):12, 2009.

[Sch16] Jürgen Schmidhuber. Deep learning. *Encyclopedia of Machine Learning and Data Mining*, pages 1–11, 2016.

[Seg04] Mark R Segal. Machine learning benchmarks and random forest regression. *Center for Bioinformatics & Molecular Biostatistics*, 2004.

[SHB+12] Richard Socher, Brody Huval, Bharath Bath, Christopher D Manning, and Andrew Y Ng. Convolutional-recursive deep learning for 3d object classification. In *Advances in neural information processing systems*, pages 656–664, 2012.

[She18] Natalie Sherman. Four reasons that apple shares have been falling. *BBC*, 2018.

[SHK+14] Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. Dropout: a simple way to prevent neural networks from overfitting. *The Journal of Machine Learning Research*, 15(1):1929–1958, 2014.

[Sig15] N. B. Signal. Complete solutions for clinical neurodiagnostics neuroscience research. *Of Neurosky, Inc. Neorosky*, 2015.

[SJ25] Edward K Strong Jr. Theories of selling. *Journal of applied psychology*, 9(1):75, 1925.

[SK04] S Shyam Sundar and Sriram Kalyanaraman. Arousal, memory, and impression-formation effects of animation speed in web advertising. *Journal of Advertising*, 33(1):7–17, 2004.

[SL98] Marcel G Schaap and Feike J Leij. Database-related accuracy and uncertainty of pedotransfer functions. *Soil Science*, 163(10):765–779, 1998.

[SL04] Lixiang Shen and Han Tong Loh. Applying rough sets to market timing decisions. *Decision support systems*, 37(4):583–597, 2004.

[SL09] Guo Bin Shen and Shipeng Li. Dynamic search with implicit user intention mining, October 6 2009. US Patent 7,599,918.

[SL12] George AF Seber and Alan J Lee. *Linear regression analysis*, volume 936. John Wiley & Sons, 2012.

[SLT+03] Vladimir Svetnik, Andy Liaw, Christopher Tong, J Christopher Culberson, Robert P Sheridan, and Bradley P Feuston. Random forest: a classification

and regression tool for compound classification and qsar modeling. *Journal of chemical information and computer sciences*, 43(6):1947–1958, 2003.

[SM01] Douglas-Cowie E Westerdijk M Gielen S Schrder M, Cowie R. Acoustic correlates of emotion dimensions in view of speech synthesis. *InSeventh European Conference on Speech Communication and Technology*, 2001.

[SMS11] Gagan Deep Sharma, Mandeep Mahendru, and Sanjeet Singh. Advertisement cause sales or sales cause advertisement: a case of indian manufacturing companies. 2011.

[SPW98] Emad W Saad, Danil V Prokhorov, and Donald C Wunsch. Comparative study of stock trend prediction using time delay, recurrent and probabilistic neural networks. *IEEE Transactions on neural networks*, 9(6):1456–1470, 1998.

[SSB14] Haşim Sak, Andrew Senior, and Françoise Beaufays. Long short-term memory recurrent neural network architectures for large scale acoustic modeling. In *Fifteenth annual conference of the international speech communication association*, 2014.

[SSS13] J Srikanth, M Saravanakumar, and S Srividhya. The impact of celebrity advertisement on indian customers. *Life Science Journal*, 10(9):59–65, 2013.

[STS11] Spyros Spyrou, Andrianos Tsekrekos, and Georgia Siougle. Informed trading around merger and acquisition announcements: Evidence from the uk equity and options markets. *Journal of Futures Markets*, 31(8):703–726, 2011.

[Sun03] Walter Sun. Relationship between trading volume and security prices and returns. *Area Exam Report, Technical Report, MIT Laboratory for Information and Decision Systems*, 2003.

[SVMN06] Edith G Smit, Lex Van Meurs, and Peter C Neijens. Effects of advertising likeability: A 10-year perspective. *Journal of Advertising Research*, 46(1):73–83, 2006.

[Swe02] Latanya Sweeney. k-anonymity: A model for protecting privacy. *International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems*, 10(05):557–570, 2002.

[T+02] Michal Teplan et al. Fundamentals of eeg measurement. *Measurement science review*, 2(2):1–11, 2002.

[Tib96] Robert Tibshirani. Regression shrinkage and selection via the lasso. *Journal of the Royal Statistical Society. Series B (Methodological)*, pages 267–288, 1996.

[TK91] Carlo Tomasi and Takeo Kanade. Detection and tracking of point features. 1991.

[Uti05] Tsai SP. Utility. cultural symbolism and emotion: A comprehensive model of brand purchase value. *International journal of Research in Marketing6*, 22(3):277–291, 2005.

[VDOD16] Aaron Van Den Oord and Sander Dieleman. Heiga zen, karen simonyan, oriol vinyals, alex graves, nal kalchbrenner, andrew senior, and koray kavukcuoglu. *Wavenet: A generative model for raw audio. arXiv preprint*, 2016.

[VHGK14] M Volodymyr, N Heess, A Graves, and K Kavukcuoglu. Recurrent models of visual attention. NIPS, 2014.

[VL92] Charles Van Loan. *Computational frameworks for the fast Fourier transform*, volume 10. Siam, 1992.

[VLL+10] Pascal Vincent, Hugo Larochelle, Isabelle Lajoie, Yoshua Bengio, and Pierre-Antoine Manzagol. Stacked denoising autoencoders: Learning useful representations in a deep network with a local denoising criterion. *Journal of machine learning research*, 11(Dec):3371–3408, 2010.

[VSP+17] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *Advances in Neural Information Processing Systems*, pages 5998–6008, 2017.

[VTB13] Mojtaba Vaismoradi, Hannele Turunen, and Terese Bondas. Content analysis and thematic analysis: Implications for conducting a qualitative descriptive study. *Nursing & health sciences*, 15(3):398–405, 2013.

[WCC16] Regina WY Wang, Yu-Ching Chang, and Shang-Wen Chuang. Eeg spectral dynamics of video commercials: Impact of the narrative on the branding product preference. *Scientific reports*, 6, 2016.

[Wen17] Bin Weng. Application of machine learning techniques for stock market prediction. 2017.

[Whi53] Peter Whittle. The analysis of multiple stationary time series. *Journal of the Royal Statistical Society. Series (Methodological)*, pages 125–139, 1953.

[Whi63] Peter Whittle. Prediction and regulation by linear least-square methods. 1963.

[WNL11] Xiao-Wei Wang, Dan Nie, and Bao-Liang Lu. Eeg-based emotion recognition using frequency domain features and support vector machines. In *International conference on neural information processing*, pages 734–743. Springer, 2011.

[WS08] Dennis Wackerly and William Scheaffer. Mathematical statistics with applications . belmont, ca, usa, thomson higher education. Technical report, ISBN 0-495-38508-5, 2008.

[WSN15] Ralph B Weller, Stanley D Sibley, and Colin Neuhaus. Experimental results concerning the affect of the female model in television commercials on product and brand recall. In *Proceedings of the 1982 Academy of Marketing Science (AMS) Annual Conference*, pages 468–472. Springer, 2015.

[WWZG12] Ju-Jie Wang, Jian-Zhou Wang, Zhe-George Zhang, and Shu-Po Guo. Stock index forecasting based on a hybrid model. *Omega*, 40(6):758–766, 2012.

[WYM+16] Jiang Wang, Yi Yang, Junhua Mao, Zhiheng Huang, Chang Huang, and Wei Xu. Cnn-rnn: A unified framework for multi-label image classification. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2285–2294, 2016.

[WZZ+13] Li Wan, Matthew Zeiler, Sixin Zhang, Yann Le Cun, and Rob Fergus. Regularization of neural networks using dropconnect. In *International Conference on Machine Learning*, pages 1058–1066, 2013.

[XBK+15] Kelvin Xu, Jimmy Ba, Ryan Kiros, Kyunghyun Cho, Aaron Courville, Ruslan Salakhudinov, Rich Zemel, and Yoshua Bengio. Show, attend and tell: Neural image caption generation with visual attention. In *International Conference on Machine Learning*, pages 2048–2057, 2015.

[YSXZ15] Wenpeng Yin, Hinrich Schütze, Bing Xiang, and Bowen Zhou. Abcnn: Attention-based convolutional neural network for modeling sentence pairs. *arXiv preprint arXiv:1512.05193*, 2015.

[YTC+15] Li Yao, Atousa Torabi, Kyunghyun Cho, Nicolas Ballas, Christopher Pal, Hugo Larochelle, and Aaron Courville. Describing videos by exploiting

temporal structure. In *Proceedings of the IEEE international conference on computer vision*, pages 4507–4515, 2015.

[YZWY17] Lantao Yu, Weinan Zhang, Jun Wang, and Yong Yu. Seqgan: Sequence generative adversarial nets with policy gradient. In *AAAI*, pages 2852–2858, 2017.

[ZAQ17] Liheng Zhang, Charu Aggarwal, and Guo-Jun Qi. Stock price prediction via discovering multi-frequency trading patterns. In *Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 2141–2149. ACM, 2017.

[Zha13] Linhao Zhang. *Sentiment analysis on Twitter with stock price and significant keyword correlation.* PhD thesis, 2013.