UiO : Department of Informatics
University of Oslo

Predicting Stocks with Machine Learning

Stacked Classifiers and other Learners Applied to the Oslo Stock Exchange

Magnus Olden

Master's Thesis Spring 2016

Predicting Stocks with Machine Learning

Magnus Olden

29th April 2016

Abstract

This study aims to determine whether it is possible to make a profitable stock trading scheme using machine learning on the Oslo Stock Exchange (OSE). It compares binary classification learning algorithms and their performance. It investigates whether Stacked Ensemble Learning Algorithms, utilizing other learning algorithms predictions as additional features, outperforms other machine learning techniques. The experiments attempt to predict the daily movement of 22 stocks from OSE with 37 machine learning techniques, using selected data spanning over four years.

The results shows that the top performing algorithms outperform Oslo Benchmark Index (OBX). However, several issues regarding the test period and stock prediction in general stops us from drawing an indisputable conclusion whether a long term profitable scheme is likely. The experiments yielded no evidence indicating that stacked ensemble learning outperforms other machine learning techniques.

Contents

1 Introduction **1**
- 1.1 Motivation . . . 2
- 1.2 Goals and Research Questions . . . 2
- 1.3 Outline . . . 3

2 Background **5**
- 2.1 Stock Market . . . 6
  - 2.1.1 Stock exchanges . . . 6
  - 2.1.2 Buying, Selling and making Profit . . . 6
  - 2.1.3 Forces that move stock prices . . . 7
  - 2.1.4 Predictability . . . 8
  - 2.1.5 Oslo Stock Exchange . . . 10
  - 2.1.6 OBX and other indices . . . 10
  - 2.1.7 Available Data . . . 10
  - 2.1.8 Noise in Finance . . . 11
  - 2.1.9 Omitted-Variable Bias . . . 11
  - 2.1.10 Bull, Bear and Market Trends . . . 11
  - 2.1.11 Risk and Reward . . . 12
  - 2.1.12 Ethical concerns with automated trading . . . 12
- 2.2 Machine Learning . . . 12
  - 2.2.1 Sample data and building models . . . 13
  - 2.2.2 Supervised Learning . . . 14
  - 2.2.3 Regression . . . 14
  - 2.2.4 Classification . . . 15
  - 2.2.5 Binary Classification and Performance Measures . . . 16
  - 2.2.6 Over- and under-fitting . . . 17
  - 2.2.7 Cross Validation . . . 17
  - 2.2.8 Time series analysis . . . 18
  - 2.2.9 Normalization . . . 18
  - 2.2.10 Curse of dimensionality . . . 19
  - 2.2.11 Feature Selection . . . 19
  - 2.2.12 Statistics . . . 19
- 2.3 Binary Machine Learning Models . . . 20
  - 2.3.1 No Free lunch Theorem . . . 20
  - 2.3.2 The Perceptron . . . 21
  - 2.3.3 Support Vector Machines . . . 22
  - 2.3.4 Bayes Point Machine . . . 24

2.3.5 Logistic Regression . . . 24
2.3.6 Ensemble Learning . . . 24
2.3.7 FastTree . . . 26
2.4 Predicting the stock market . . . 26
2.4.1 State of the art . . . 26

3 Methodology **29**
3.1 Test Environment . . . 30
3.1.1 Implementation Notes . . . 30
3.2 Data . . . 31
3.2.1 Selection . . . 31
3.2.2 Preprocessing . . . 37
3.2.3 Talking Data . . . 40
3.3 The Algorithms . . . 44
3.3.1 How the modules work . . . 44
3.3.2 Machine Learning Groups . . . 45
3.3.3 Tuning Parameters . . . 49
3.4 Understanding the results . . . 50
3.4.1 Performance Measure . . . 50
3.4.2 Profit Estimation . . . 51
3.4.3 Statistical significance . . . 52
3.4.4 Random as a measure . . . 53
3.4.5 Box Plots . . . 54
3.5 Limitations . . . 54
3.5.1 Limited Depths . . . 54
3.5.2 Simplified Profit Estimation . . . 55

4 Experiments **57**
4.1 How to read the result . . . 58
4.1.1 Diversity and Adjustments . . . 58
4.1.2 Categorizing the stocks . . . 59
4.1.3 Results Overview . . . 60
4.2 Results . . . 63
4.2.1 Groups Compared . . . 63
4.2.2 Learners Compared . . . 67
4.2.3 Standalone . . . 71
4.2.4 Meta ensemble . . . 73
4.2.5 Simple Ensemble . . . 77
4.2.6 Top Performers . . . 79
4.3 Analysis . . . 79
4.3.1 Can you make a profit? . . . 79
4.3.2 Comparing Schemes and algorithms . . . 86

5 Conclusion **91**
5.1 Research Questions . . . 92
5.2 Future work . . . 94
5.2.1 Feature selection . . . 94
5.2.2 Regression and Multi-Class Classification . . . 94

5.2.3 Parameter Optimization 94
5.2.4 Other Problems 95
5.2.5 Time Frames and Markets 95

List of Figures

2.1 A model of supervised learning 15
2.2 A model of Regression [70] 15
2.3 A model of Classification [73] 16
2.4 A model of Under- and Overfitting 18
2.5 Model of Perceptron [21] 21
2.6 Model of Neural Network [72] 22
2.7 Model of an SVM's Kernel [73] 23
2.8 Model of optimal separation [73] 23
2.9 Model of a decision tree [71] 25

3.1 Histogram of days above and below 0% change for each stock 36
3.2 Histogram of days above and below 0.5% change for each stock 36
3.3 Histogram of days above and below 1% change for each stock 36
3.4 Visualization of dividing the data set into two separate sets . 37
3.5 Visualization of k-fold cross validation 38
3.6 Visualization of how the data was made to a time series . . . 41
3.7 NYSE, trends for the entire period 42
3.8 Average of all international indices, trends for the entire period 42
3.9 OBX weighted, trends for the entire period 43
3.10 NOK against all currencies, trends for the entire period . . . 43
3.11 Crude Oil per barrel, trends for the entire period 44
3.12 How a machine learning algorithm is trained in Azure Machine Learning Studio 45
3.13 Illustration of N Standalone Learners, and how they were cross-validated 46
3.14 Illustration of a single simple ensemble learner 47
3.15 Illustration of N Minimal Meta Learners, utilizing N and how they were cross-validated 48
3.16 Illustration of the maximal meta ensemble learners 49
3.17 Plot of low, standard and high normal distributions 54
3.18 Illustration of Boxplot 54

4.1 Groups Compared 65
4.2 Statistical Measures, Groups Compared 67
4.3 All Algorithms Compared 70
4.4 Standalone Learners Compared 72
4.5 Maximal Meta Ensemble Learners 74

4.6 Minimal Meta Ensemble Learners Compared 76
4.7 Simple Ensemble Learners 78

List of Tables

3.1 Stocks included in the experiments 32
3.2 International Indices, country and name 33
3.3 Preliminary results of days included for input 35
3.4 Preliminary results of days forward to predict 35
3.5 Preliminary results of different threseholds 35
3.6 Profit estimate for owning the stock through the entire test period 52

4.1 Categorized by Profit Estimation, Bull, Bear and Normal stocks 60
4.2 Algorithms 61
4.3 Overview of results 62
4.4 Significance Table, Groups Compared 64
4.5 Significance Table, All Algorithms 69
4.6 Significance Table, Standalone Algorithms 71
4.7 Significance Table, Maximal Meta Ensemble Algorithms . . . 73
4.8 Significance Table, Minimal Meta Ensemble Algorithms . . . 75
4.9 Significance Table, Simple Ensemble Learners 77
4.10 Top Performers 79

5.1 Parameters Azure ML algorithms 104
5.2 Preliminary results of days included for input 105

Abbreviations

DNB	Den Norske Bank
DNO	Det norske oljeselskap International
FOE	Fred. Olsen Energy
FRO	Frontline
GJF	Gjensidige Forsikring
MHG	Marine Harvest
NYSE	New York Stock Exchange
NOD	Nordic Semiconductor
NOK	Norwegian Krone
NHY	Norsk Hydro
NAS	Norwegian Air Shuttle
OPERA	Opera Software
ORK	Orkla
OBX	Oslo Benchmark Index
OSE	Oslo Stock Exchange
PGS	Petroleum Geo-Services
REC	REC Silicon
RCL	Royal Caribbean Cruises
SCH	Schibsted serie A
SDRL	Seadrill
STL	Statoil
STB	Storebrand
SUBC	Subsea 7
SVM	Support Vector Machine
TEL	Telenor
TGS	TGS-NOPEC
YAR	Yara International

Acknowledgement

I would like to express my gratitude to my counselor, Kyrre Glette. My sincere thanks to my father Asgeir for editing and spell checking this thesis, to my brother Andreas for helping me with the economy and Domos Labs for allowing me to use their servers for the experiments.

I would also like to thank my fellow students for helping me keep my sanity, and to my friends, family, and girlfriend for all their support.

Chapter 1

Introduction

1.1 Motivation

As long as there have been stock markets, Fortune Hunters and Academics alike have attempted to predict them. Predicting stock markets and individual stocks are interesting and valuable as one can gain both financial benefits and economic insight, driven by numerous factors, stocks are notoriously challenging to predict. Researchers cannot agree upon whether the stock markets are predictable [50] or not [33], studying whether one can predict them is therefore very interesting.

Machine Learning a sub-field of computer science is the study and application of computers that possess the ability to find patterns, generalize and learn without being explicitly programmed. In the recent years, efforts have been put into applying machine learning to stock predictions [44] [5], however there are still many stock markets, machine learning techniques and combinations of parameters that are yet not tested. Some have applied machine learning to the Oslo Stock Exchange [47], Norway's only stock exchange. But the studies are limited, and there are still a great deal of improvement that can be made. The Oslo Stock Exchange may be even more problematic than other Stock Exchanges because of its relative small size and limited number of factors that heavily affects it [55].

Ensemble Learning is a machine learning technique where several individual machine learning algorithms are combined into a single, and hopefully, more powerful algorithm. The initial idea is that several minds are greater than one. There are several ways of combining machine learners, and some have been utilized successfully to predict stocks [22]. Stacking, an ensemble technique works by using a machine learning algorithm to learn from other algorithms prediction. This technique have been applied to stock prediction [59] and showed great promise and a need for further research. These algorithms are computationally heavy, but modern day cloud computing with very few computational restraints allows us to disregard the complexity of these algorithms and apply them to a problems such as stock prediction.

1.2 Goals and Research Questions

To formalize the overall goals of this thesis, four research questions have been formed. The experiments and results presented have been performed to give insight that may help answer the questions. The main goal of this study is to determine whether or not it is possible to make a profitable stock trading scheme using machine learning on the Oslo Stock Exchange. Or as questioned in research question (1.1)

(1.1) Is it possible to make a profit on the Oslo Stock Exchange using machine learning?

It is possible to make a profit on the stock market by trowing darts on table of stocks, one could however not claim that dart trowing is a sensible stock picking strategy that always will yield profit. The same principles applies to machine learning, and we should therefore use several machine learning algorithms applied to several different stocks to, with any form of certainty, determine whether it is possible to make a profit on the Oslo Stock Exchange. Research question (1.2) provides insight into whether stocks are predictable.

(1.2) Does the performance of machine learning algorithms predicting stocks vary?

It might sound obvious that different algorithms using different ideas, mathematics and implementations will perform variously, however if the stock market and individual stocks are governed by a random walk as many claim. Then it is not predictable and the machine learning algorithms should over time and over many stocks see a very similar performance.

As noted, stacked ensemble schemes have shown great promise to improve the performance of normal machine learning algorithms. Leading us to the third research question (1.3)

(1.3) Will Machine Learning Algorithm will perform better when other machine learning algorithms predictions are included as a feature?

The final research question looks into whether this thesis has a set itself up for success. The experiments utilizes binary classification to predict the stock market. Meaning that the machine learning algorithms predicts whether or not a stock will increase with more than a certain threshold the next day. Research question (1.4) questions the validity of the approach.

(1.4) Is Binary Prediction suitable for a stock market problem?

To find an answer to these questions, experiments using 37 machine learners have been tested on 22 ocf the stocks with the highest turnover on the Oslo Stock Exchange. They attempt to predict whether a stock will rise in value the next day using a selected set of input data, and are subsequently tested against Oslo Benchmark Index.

1.3 Outline

This thesis is divided into five chapters: introduction, background, methodology, experiments and conclusion. The Background chapter 2 explains the theory used in the experiments, and provides insight into previous work. The Methodology chapter 3 contains an overview of the

data and explains how the algorithms were implemented. The Experiments chapter 4 presents the results and explains how to understand them, and discusses their implications. The Conclusion chapter 5 attempts to draw some conclusions from the results.

Chapter 2

Background

This chapter describes the financial and machine learning theory that is the basis for all the experiments. Section 2.1 explains stock markets, what affects them, and challenges one faces in predicting them. Section 2.2 describes Machine Learning and its potential and drawbacks. Section 2.3 provides an overview of the Machine Learning models utilized. Section 2.4.1 presents a brief summary of the state of the art regarding predicting stocks with machine learning.

2.1 Stock Market

A stock of a corporation is an equity stake, or more simply: a stock is an ownership share in a corporation. A stock market is an aggregation or gathering of buyers and sellers of stocks and other financial instruments, a place where financial instruments are traded.

2.1.1 Stock exchanges

A Stock Exchange is a stock market where brokers and traders buy, sell or exchange publicly listed financial instruments. Commonly stock exchanges provide a way for brokers and traders to exchange financial instruments. Traditionally stock exchanges were physical places, often referred to as the floor, where stock-brokers and -traders exchanged stocks for other stocks or money. These days nearly all stock trades take place through electronic communication. Most stock exchanges work as an institution that allows for trading certain stocks and other financial instruments through a near instant electronic trading system. Most stock exchanges use a continuous auction principle. This principle includes an instant execution of stock orders as they are received by the market. By operating with the continuous principle and rapid electronic orders, modern day stock exchanges are driven solely by supply and demand.

There are now hundreds of stock exchanges throughout the world. Some stock exchanges, like Oslo Stock Exchange, include all of the listed stocks in a country or region, while others like NASDAQ and NYSE are more specialized in certain types of corporations and industries. As stated, most of the stock exchanges in the world have automated the trading process, but some stock exchanges like NYSE and some other smaller stock exchanges, still have a floor where stocks can be traded.

2.1.2 Buying, Selling and making Profit

When someone acquires any number of a corporation's stock, it is common terminology to say they have entered the market. If the stock they acquired is subjected to price changes, the real value of their investment also changes. When the stock price goes up, they have effectively made money. And it is the opposite way when the stock loses in value. However, these value changes are not realized before the stock is sold. Buying stocks

in the belief that its value will increase is called a long position. A short position is a bet that a stock's value will depreciate. Shorting can be done by selling stocks that you do not currently own, and later repurchase them. If the price of the stock declines, you will repurchase them at a lower price than you sold them for and realize a profit. There are also other forms of making profits in the stock market such as options, but long and short sales are the most important for this thesis.

The stock market is not a zero-sum-game, meaning that all stock markets have historically had an upward movement increasing above inflation. New money goes into the stock market daily. In short this means that by simply buying a random set of stocks, and hold on to them, you are likely to make a profit as the years goes by. Because of this consistent upward movement of stock markets over time, one cannot say that a trading scheme is successful simply because it generates a profit.

An Index is a gathering of stocks. The value of an index is a mathematical construct which typically uses weighted average of the gathered stocks to compute the overall price. These indices can be used as a measuring tool on how well a certain part of a stock market is doing. A Stock Market Index is a measurement of the value of a part of a stock market. It is made up by prices on selected stocks. Stock Market Indices can be useful for comparison with trading schemes, because they, or at least try to, represent the overall movement of the stock market. Outperforming the Stock Market Index is known as beating the market, because Stock Market Indices is a representation of the market.

2.1.3 Forces that move stock prices

Since this is a thesis about predicting stock prices, an important part of it will be attempting to understand and utilize the relationship between stock prices and various factors. One can read countless theses, papers and books on forces that move the stock prices, and still get none the wiser, or at least not fully understand or know half of what goes into the pricing of stocks. Macroeconomics, psychological effects, politics, news, country borders, the corporation's current financial are just some of the factors that affect the price of a stock.

Due to the limitations of this thesis, it would not be feasible to account for all of them, but efforts will be made to a least include the most important short term factors. Movements in stock prices can be looked at both short term and long term. The terms *short term* and *long term* are not academically defined. In this thesis *short term* will be used for time periods of less than 3 months, while *long term* is any amount of time above that.

David M. Cutler tries in his paper "What moves stock prices" to determine what factors that go into the stock price and estimates the fraction of the variance in stock returns that can be attributed to different kinds of

news. His paper is about short term changes. First the paper examines what effect macroeconomic news have on the stock prices. The conclusion is that macroeconomic news cannot explain more than one third of the variance. [28]. In the same paper he also explores political events and other news, and conclude that every type of news they have looked into effects the stock price. Economists like to talk about ideal scenarios; in an ideal world the stock price is fully explicable by a corporation's future cash flow and discount. Unfortunately, the world is far from ideal and numerous research papers such as Cutler's have shown that other factors go into the pricing of stock. [28].

Exchange rates and currencies are two of the more obvious factors for transnational companies. The relationship between stock prices and exchange rates are shown rigorously in [2]. Raw material prices, such as oil prices or aluminum prices, have also been shown to effect the pricing of stocks. [4]. Other stock markets also have to be taken into consideration[[40]. The Volume of stocks being traded[37]. News on the company can make massive impacts on the stock price[15], as can changes in a corporation's management [69]. Macro financial news, such as news about changes in interest rates and changes in inflation can move stock prices with an amplified force[52] [56]. Even speculations on Internet forums [68] may change the volume of traded stock and its prices.

That something seemingly peripheral and insignificant as an Internet forum post can move the stock price of a billion dollar corporation leads us into the perhaps most significant short term factor for stocks, the psychological effects. Stock market trends often begin with bubbles and end in crashes. Some researchers regard this as an example of herd behavior, as investors are driven by greed in bubbles and fear in crashes. Traders join the herd of other traders in rushes to get in and out of the market [19]. Greed, fear and herd mentality are just three examples of psychological effects that play a part in governing the stock market. Other, less intuitive, factors also play a role. As an example stock have been shown to move differently on Mondays and Fridays [27] . And just like a nice sunny day puts a smile on your face, it has also been known to make traders more optimistic [43].

Which forces that move the stock market is no simple question. It has been shown that there are many, many different forces that can change the pricing of stocks. This sub chapter was an attempt to give an overview of some of the most important factors. Now we need to narrow them down to a selection of parameters that may be used within the limitation of a thesis.

2.1.4 Predictability

Are the movements of stock prices predictable? Some researchers suggest that stock prices move by the *theory of random walk*, that is that the future path of the price of a stock is not any more predictable than random numbers [34]. However, *Stock prices do not follow random walks* [50] is the title

of a heavily cited paper. The authors of the paper claim that considerable empirical evidence exists that show that stock returns are to some extent predictable. This means that we can make the basic assumption that past behaviour of a stock's price is rich in information, and may show signs of future behaviour. Some claim to have shown that history is repeated in patterns, and that some of the patterns tend to recur in the future. And since there are patters, it is possible through analysis and modelling to develop an understanding of such patterns. These patterns can further be used to predict the future behaviour of stock prices.[34]
Academically, economists cannot seem to agree with each other on whether or not stock prices move by random walk or not. Both supporters of *random walk theory* [34] and supporters of predictable movements [50] claim to have shown empirically that their theory is correct. And since researchers cannot seem to agree on the predictability of the movements of stock prices, one can investigate the more practical side of this question. There are certainly numerous of anecdotal stories of people succeeding in predicting the stock market; an example is Nicolas Darvas, a Hungarian dancer who in 1960 published the book *How I made $2,000,000 in the stock market*,where he claimed to have recognized patterns in the movements of stocks that eventually lead him to great wealth. Other examples are the thousands of Technical Traders that have made a big impact on the stock market for at least 40 years, and the emerging market of automated trading schemes often known as stock robots. With only anecdotal evidence, one should be careful making generalizations (leave that to the machine learners), and for every successful stock prediction, there might be an opposite story of loss and bankruptcy.

So is the stock market predictable? It depends on which researchers you believe use the most correct methodology for their research, and even then the best answer to the question is, perhaps. What we can conclude is that there are certainly a lot of people that believe that the stock market is predictable, which coincidentally might be what makes the stock market predictable. Traders' belief in themselves and experts might create a self-fulfilling prophecy, like when the magazine *Business Week* recommends a stock, that stock gives abnormally high returns[63].

Even if the entirety of the stock market is not predictable, one can still create profitable trading schemes focusing only on parts of the stock market or certain time periods. For example has a trading scheme focusing on only buying "Business Week's" recommended stock in periods been shown to outperform the overall market[63]. Purely buying and selling stocks based on current oil prices, may perhaps yield great returns. And as previously stated, if currency changes have impact on certain stocks, it might be possible to outperform the market by acting quicker on macroeconomical news than most competitors.

2.1.5 Oslo Stock Exchange

Oslo Stock Exchange (OSE) is Norway's only stock market. Trading is carried out through computer networks, and trading starts at 09:00 and ends at 16:30 Monday through Friday. What separates OSE from other, more well known stock exchanges, is that OSE is, most prominently, a relatively small stock exchange with an overrepresentation of energy companies, and therefore to a large degree is affected by a few factors. The most obvious and most prominent factor is oil prices

oil prices significantly affect cash flows of most industry sectors at the OSE[55]

It is not surprising that oil prices plays a significant role in the pricing of a big part of OSE as Norway's economy is heavily oil dependent. Macroeconomic factors have not been shown by researchers to be as significant as oil prices.

macroeconomical variables affect stock prices, but since we only find weak evidence of these variables being priced into the market, the most reasonable channel for these effects is through company cash flow[55]

OSE being a relatively small stock exchange causes some problems. The biggest problem is the impact that inside information may have; inside information is information that someone gets before the rest of the market, giving them an unfair advantage. Inside trading is illegal, but difficult to unveil, prove and prosecute. Inside trading at OSE is discussed in[31], and is generally a known problem with smaller sized stock exchanges.

2.1.6 OBX and other indices

An index or, in plural form, indices are mutual funds, created with some rules that attempt to match or track the movement of a market. There are many ways that index funds attempt to emulate the market movements, a common way is to use a number of the stocks with the highest turnover to best try to emulate a stock market. Using this philosophy way that OBX index is set up. It consists of parts of the 25 stocks with the highest turnover on the Oslo Stock Exchange [17].

2.1.7 Available Data

To perform Machine Learning, an essential part is data, and preferably lots of it. Luckily there are seemingly infinite amounts of easily available data for stocks. The data comes in all shapes and sizes, from immensely detailed data sets containing everything from volume of trades, detailed information about each trade and lots and lots more. One can also find it in less detailed forms, where there is just a single data point for each year. The most common set of data, and the most easily obtainable for a stock consists of six variables. Time, Open, High, Low, Close and Volume. Open, high, low and close are different bid prices for the stock at different times with

relatively intuitive names. Volume is the number of stocks that changed hand during the time period. Time is marks the date, hour and occasionally minute and second of the closing price. Stock Market Data spans as far back as the 19th century, and comes in resolutions from milliseconds to decades. There are extreme amounts of data regarding the stock market. From macroeconomical data, such as interest, inflation, unemployment rates. There are also exchange rates between every pair of currency, raw material prices such as natural gas, crude oil, coal, metal. And there are several international indices attempting to give an overview of the economy in single country, region or for the entire world economy. One could list available data for anything that may affect the stock market for ages, and a lot of it is available free from different sources, providing anyone with the opportunity of retrieving the data and using it to build models.

2.1.8 Noise in Finance

Noise in financial data can be caused by many factors. It may be uncertainty about future events, technology and trends. It may be expectations not following rational rules. And it may be uncertainty regarding how the relative prices of assets should be set. It may be delays in purchases or random influxes. Whatever the reason for the noise in financial data, the noise represents imperfections that makes the market somewhat inefficient since it effects the data, but cannot be modelled. Noise will always be present in financial data, which is unfortunate, as it makes it difficult to test academic theories and practical models, such as machine learning models, of the financial markets.[13]

2.1.9 Omitted-Variable Bias

Omitted-variable bias is when a model of a system is built with falsely left out essential factors. The model over adjust for the falsely left out factors and builds a model that creates a bias where other factors are either over- or under-estimated. The problem of Omitted-Variable Bias is well known in economic circles, and is generally thought of as one of the most challenging parts of building financial models. Some even goes as far as calling Omitted-Variable Bias the phantom menace of Econometrics [23]. Econometrics being the application of mathematical models to economics for hypotheses testing and forecasting. The problems with Omitted-variable bias is highly related to the problems highlighted in the what moves stock section 2.1.3, as there are so many factors effecting the financial markets that it is nearly impossible to include them all.

2.1.10 Bull, Bear and Market Trends

Market trends are perceived tendencies of a market that moves a certain way over time. Since prices of stocks and other financial data of the future is unknown, market trends can only be set in hindsight. Two common terms in finance are Bull and Bear. These animal names are used to describe

upward and downward market trends. Bull markets or stocks describe an upward trend, while Bear Markets or Stocks describe a downward trend.

2.1.11 Risk and Reward

Risk is a broad expression used in a variety of different ways both in common speech and in finance. Risk is most commonly used to imply the uncertainty of a return of a financial mean and the potential for financial loss. Risk consists of two components; the first component is the possible outcomes that an investment can yield and the other is the likelihood of these possible outcomes. The two components combined is the overall risk. If one wants to yield long term profit in any financial system an integral part is the balance between risk and possible reward. For a stock purchasing scheme, such as the ones that will be presented later in this thesis, we should take note that owning a stock yields more risk than having the money in the bank. There are more possible outcomes, and the likelihood of negative outcomes are greater for stocks than in the bank. On the other hand, splitting your money on several stocks can disperse the risk, as the likelihood of an event that cause several stocks to fall in value is lower than for a single one. Usually coupled with risk is reward, the higher the risk, normally, the higher the possible reward as the results can vary to a greater extent.

2.1.12 Ethical concerns with automated trading

There have been concerns about automated trading, as some see it as an unfair practice. The issue has been discussed in several papers [29], with one of the main concerns being that while researchers have shown that some strategies are beneficial to the market as a whole, many automated trading strategies are not. And these automated traders simply exploit and manipulate the market in order to yield a profit [3]. Another concern is the unfair advantage large banks and other companies that can splash out for servers with low latency connections directly into the stock market. These expensive servers give some an unfair advantage, as much of the profit in today's automated trading is yielded from pure speed. Critics also show the increased likelihood of crashes [6] as a direct consequence of automated trading, and that it is notoriously hard to regulate the ever growing market of automated trading [10]. On the other hand, some researchers claim that the automated trading makes the market more efficient [9], as it makes the market more efficient

2.2 Machine Learning

Machine learning is that domain of computational intelligence which is concerned with the question of how to construct computer programs that automatically improve with experience. [54]

Or in other words it is about constructing machines that adapt and modify their actions or predictions in such a way that they get more accurate. In order to properly understand Machine Learning, it is useful to first understand and define learning. Learning is a function that allows, animals and machines alike, to modify and reinforce or acquire new knowledge, skills, preferences and behaviors [53].

With some basic understanding of learning, we can now move forward to Machine Learning
Machine Learning is simply the implementation of learning capabilities in machines. Phil Simon defined Machine learning is what gives computers the ability to learn without being explicitly programmed [66]

Machine learning is a research field within Computer Science. Most of the research regarding Machine Learning investigates the construction of algorithms that can learn and later predict, a prediction being a statement that some definite outcome is expected. Machine Learning algorithms operate by constructing a model from example data. Machine Learning differs from traditional Computer Science. When traditional computer science writes code specifically for the problem domain and uses a priori knowledge to hard-code rules, machine learning algorithms are not tailored for a single problem domain, the algorithm itself learns a model that fits samples in the data and by extension create its own self-taught rules.

Machine Learning is not a trivial task. After decades of research, no machine can be said to have anywhere near human-like learning capabilities. In Machine Learning: An Artificial Intelligence Approach the authors state that the problem of implanting learning capabilities in computers is highly demanding. And some researchers claim that making machines learn is a most challenging in artificial intelligence [53]

However, success has been attained several times on many types of data and problems. Amongst them Machine Learning have been successfully used for modeling financial time series.[62]

2.2.1 Sample data and building models

A prerequisite for learning is that there is something to learn from, samples are needed. For machines these samples are sets of data. For a Machine Learning algorithm to derive a stable and durable model, it is necessary that the data is suitable. A correlation or a context between the sample data, or a transformation of the sample data and the desired outcome is needed. There may be several problems with the sample data which makes it more difficult to derive a sustainable model for prediction. The data should to be stationary, meaning that the distribution does not change over time. The degree of random noise in the sample data should not be too high. And equal inputs should yield equal outputs.

When data is gathered it is usually divided into two groups. These groups of data sets are called Training set and Test set. The training set is the data that you apply to build the model. This set is used to fit the Machine Learning model by pairing the input with the expected output, and attempting to find some relationship between the variables and thereby minimizing the error. The fitting process is repeated until the model has minimized the errors to the point where it has reached some minimal threshold. When one terminates the process is often decided with cross validation which is further discussed in section 2.2.7. After fitting the model, the Test set is put to use. Since the Test set is unseen by the model we can use it to get a final measure on how well the Machine Learning model fits the data; this measure should indicate how well the model will perform on real-world data.

2.2.2 Supervised Learning

For Supervised Learning there is a set of data that contains both input data and target data, target data being the answer which the algorithm should produce from the input. These two sets of data combined are usually referred to as the training data. The target data can be prepared by experts or normal humans, or in some cases, as with stock market, these targets would be the next, or some incremental jump, forward in time. If we had an infinitely large training set, with every possible input and outcome of a problem, we could have made a look-up table, where we would simply look up the input and then the corresponding output. Unfortunately, it would not be computational feasible, as no computer can store an infinite amount of data, nor is it possible to generate every possible input and output for most real-world problems. This is why supervised learning is useful, because supervised learning can make the computer able to generalize. The Machine Learning algorithm should produce sensible output for the input. By generalizing, the algorithms separate noise, or small errors, in the data from the desired data.

Within supervised learning we can divide the types of problems into two sub groups: Regression, predicting continuous values. Classification, predicting discrete categories.

2.2.3 Regression

As stated, regression is about predicting values. Regression predicts a continuous value. The regression is performed by estimating the relationships between input variables and the output, the relationship being how a certain change in the input would affect the output. An example of regression in machine learning is prediction of brain activity used in [35] where they successfully predicted certain values of brain activity in areas in the brain after some stimuli. Another example could be predicting the future price

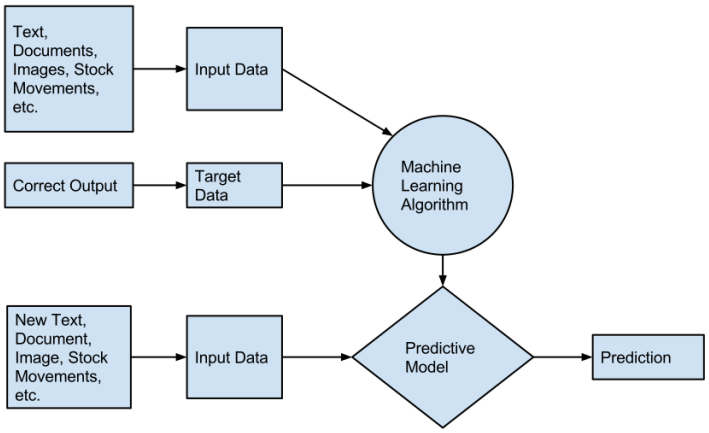


Figure 2.1: A model of supervised learning

of a stock based on current pricing.

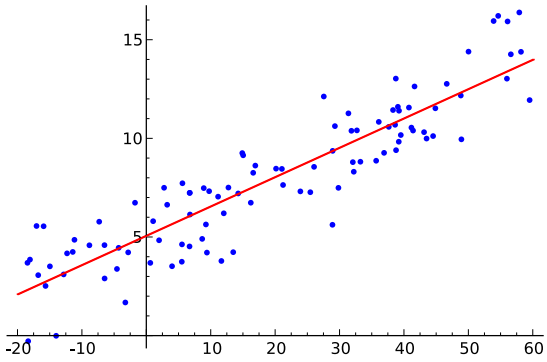


Figure 2.2: A model of Regression [70]

In figure 2.2 a linear regression is shown. The regression attempts to fit a line that can predicts continuous values.

2.2.4 Classification

Classification is the prediction of discrete values. The problem is taking a new input or observation and deciding which, of a number of classes, the input belongs to, knowing each input belongs to exactly one class. It is performed based on training form examples of each class, and is therefore a supervised learning algorithm. An example of classification would be assigning a diagnosis to a patient based on observations, the observations could be white blood cell ratio, symptoms, heart rate, gender

etc. The discrete nature of classification is however not always realistic in real world problems, as some examples might belong to, or be precisely at the border of two or more different classes. There are also problems when it is impossible to categorize each possible input. Another example of classification is forecasting the level of return on stock market index done by [49] where the researchers tested several classification models, and predicted stocks returns with various success. In the study they used the directions of movement of the stock as categories. In figure 2.3 we see an example of classification, a linear separator divides the two classes.

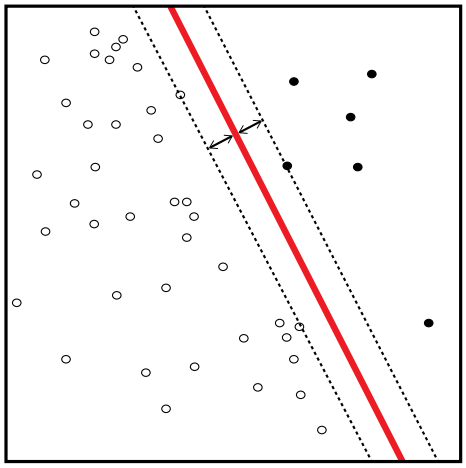


Figure 2.3: A model of Classification [73]

2.2.5 Binary Classification and Performance Measures

Binary, also known as Two-Class, Classification is simply Classification with only two possible outputs, for example True or False. The biggest difference between these binary classification, and classification with more possible outcomes is the way we measure their performance. Some of the more common measures are Accuracy, Precision, Recall and F-Score. Accuracy is simply a measure of the percentage or number of correctly classified samples. This measure does however fall short when two categories are not symmetric, which is when one class has a greater number of samples than the other. In cases like this the classifier will have a high accuracy by simply predicting a majority class every time. The other measures allow us to examine the result in different contexts and in relation to the size of the True and False classes. All of these measures are in the range between 0 and 1, 1 being the optimal result and 0 being the absolutely worst. The formulas for calculating the values are shown in 2.1, 2.2, 2.3, 2.4

$$Precision = \frac{tp}{tp + fp} \tag{2.1}$$

$$Accuracy = \frac{tp + tn}{tp + tn + fp + fn} \tag{2.2}$$

$$Recall = \frac{tp}{tp + fn} \tag{2.3}$$

$$F - Score = 2 * \frac{precision * recall}{precision + recall} \tag{2.4}$$

2.2.6 Over- and under-fitting

One of the most prominent issues for performing Machine Learning Are Over- and Under-fitting. The problems occur because of noise. To better explain Over- and under fitting, we can use the principle of Occam's Razor to deduce how and why these problem occurs. As we know, Machine Learners attempt to build a model so that for a set of inputs, it can provide the wanted output. Occam's Razor states that a model should contain all that is necessary for the modeling but nothing more. In Machine Learning terms this means that the set of inputs should contain exactly what is needed for a good prediction, meaning that the model includes more terms than are necessary or use more complicated approaches than are necessary [41]. Figure 2.4 attempts to show the problems of over- and under-fitting. When the model emphasizes having low error too much, the model creates a decision boundary that is overly complicated and includes the noise. When the model allows for too great of an error, it is not able to properly divide the classes. These problems can be difficult to manage. And unfortunately, nearly every real world data set contains some sort of noise. What happens is that the learner tries to predict the noise, and creates a model which either requires too many inputs or is too complex.

2.2.7 Cross Validation

To dodge the problems of over- and under-fitting and keep the machine learning algorithm's generalization power, cross-validation is used. Cross-validation is a model evaluation method. The issue with simply dividing the data set into one part for testing and one part for building the model is that there is no way of knowing when to stop fitting the model. There is no way of knowing when the model is overfitting and when the model is underfitting. To counter this issue cross validation removes some of the data before training begins. When the training is done, the data that was removed is used to test the performance of the fitted model with unseen data. There are several different ways of cross-validation. One of the more common is known as K-Fold Cross Validation. The data is divided into k sets, and one of the sets is used for testing, while the other k-1 sets are used for training. The process is repeated k times. K-Fold is a thorough cross-validation technique, but has some problems with being time and computational consuming. It is common to set k as 10 [8].

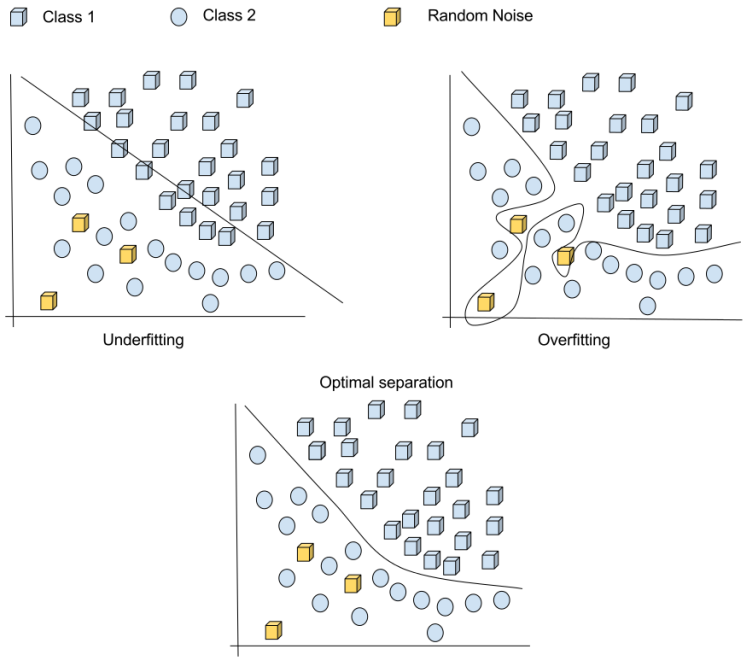


Figure 2.4: A model of Under- and Overfitting

2.2.8 Time series analysis

Time Series are observations ordered in sequences, generally ordered in time. Time Series Analysis differs from other analyses, because the observations are dependent of time. This requires some considerations when performing machine learning, as one cannot randomize the order of the inputs nor can one with certainty claim how many of previous observation a current observation is dependent on. For the latter reason one should perform some preliminary test onto how far back in time one looks when attempting to make a prediction. Studies have shown that machine learning algorithms are fitting for forecast time series [1] and that machine learning can be used fully for both one and two steps of prediction [16].

2.2.9 Normalization

A common problem with data sets is that the scaling is off, essentially meaning that some of the data points are simply much larger than others without necessarily representing a large change. An example from stock markets is the volume of stocks sold, which may vary from zero to millions, while the change of a price of a stock varies with a certain percentage. However, a few percent of change in price might very well be a better indicator for tomorrow's stock price than a change of thousands in volume. These scaling differences may cause problems for machine learning algorithms and it may therefore be wise to apply some feature scaling to the data set prior to using it for machine learning. To further illustrate the problem, we may look at machine learning algorithms that

uses Euclidean distance. If one of the features has a broad range of values such as stock volume, the distance will be governed by that particular feature. A way of handling these problems are known as Normalization, which is the process of uniformly changing the amplitude of a data set, so that no features varies in a greater range than the other.

2.2.10 Curse of dimensionality

As we recall from section 2.1.3, there are many, many factors that may influence the movement of a stock. This is a common issue with many machine learning applications and may lead the designers of machine learning algorithms into the temptation of simply including vast amounts of data to prevent Omitted-Variable Bias. However, including too much data comes with several other problems, one of which is known as the Curse of Dimensionality. The term was introduced by Bellman to the describe the issues that occur when one increases the dimensions of an input. The problem arises because of the exponential increase in volume caused by adding more dimensions [7] [46]. When the complexity of the input rises, the complexity of the underlying patterns may also rise. To extract these more complex patterns we may need more samples. Simply put, a high dimension input may require a high number of samples to build a meaningful model.

2.2.11 Feature Selection

To counter problems such as the curse of dimensionality it is common to use some sort of feature selection. Feature selection is the process of selecting a subset of all the features for use. The benefits of feature selection are many, as it can reduce training time, storage needs and reduce implications of the curse of dimensionality [39]. There are three major ways of performing feature selection; filtering, wrapping and embedding methods. The filter methods analyse the properties of the data set to find the optimal set of features. One such filter method is known as Pearson's correlation and uses a measure of the linear correlation between two variables to find a good set of parameters. Wrapper methods uses a classifier to find the features, and embedded methods are more complex methods where the feature selection is part of the machine learning algorithm.

2.2.12 Statistics

In order to validate that machine learning models are indeed making predictions that are useful, it is common to apply to statistical tests to the results in order to know with certainty that the results are not derived by chance and are indeed with statistical significance outperforming a baseline.

Hypothesis testing and P-Values

A way of testing whether an event has any effect is a hypothesis test. This can be done by first stating a null hypothesis, meaning an initial idea that an event has no effect, and then calculating the p-value. A p-value states how likely the hypothesis is and may be calculated in a number of different ways depending on what type of test is used. Using a predefined threshold α, commonly 0.05, one states that when the P-Value is below α we reject the null hypothesis and claim that there is a statistical significant chance of the event having some effect.

Two Samples Whitney Mann U test

One way of deriving a P-Value is with a Two Samples Mann-Whitney U Test. It is a nonparametric statistical test. The test provides statistics regarding whether it is likely that two groups of numerical values are be from the same population. It does not require the population to have similar values to each other, and can therefore be used when one of the populations have larger values than the other. Also it does not assume any normal distribution and may therefore be used regardless of the distribution of the samples. What the Two sampled Whitney Mann U test tells us is whether or not one can with statistical significance see whether there is a difference between two sets of samples. This is very useful when comparing machine learning algorithms as it gives insight into whether or not the result from two machine learning algorithms are in fact different. It is calculated by:

$$U = n_1 n_2 * \frac{n_2(n_2+1)}{2} * \sum_{i=n_1+1}^{n_2} R_i$$

Where n_1 and n_2 represent the sample sets, R_i the Rank of the sample and U is the Result of the Whitney Mann U test. The test is commonly used because it only assumes random picking from the population, independence within the samples and mutual independence and ordinal measurement scale.

2.3 Binary Machine Learning Models

There are a high number of binary machine learning algorithms; this section will give an overview of a handful of the more common algorithms and a brief explanation of how they work. First, however, a quick explanation why there are so many machine learning algorithms.

2.3.1 No Free lunch Theorem

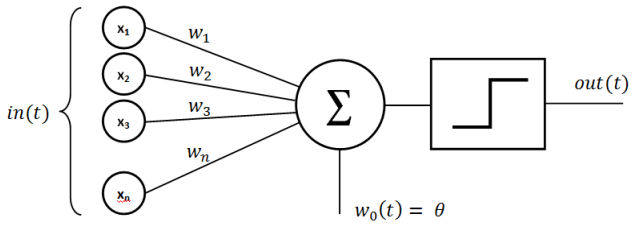
The reason why there are some many machine learning algorithms is known as the No Free Lunch Theorem. The No Free Lunch Theorem for supervised learning states that all supervised machine learning algorithms are equivalent when their performance is averaged across all possible

problems [76]. The implications of the theorem are that there is not one optimal machine learning algorithm for all problems, and that some are more fitting for certain problems than other. For supervised learning the cause of this is that different algorithms handle issues like noise and overfitting differently. There is no way of knowing with certainty how a single machine learning algorithm will perform on a certain problem. And we therefore have to try several algorithms in order to know with certainty whether or not machine learning may solve a problem. [75]

2.3.2 The Perceptron

A perceptron is a machine learning algorithm that can solve limited, simple problems. It is modeled after the neurons in the human brain. It works simply by weighing a number of inputs; if the sum of the inputs times the respective weight is above some predefined threshold, the output is true, otherwise the output is false. A model of the perceptron can be seen in figure 2.5. Starting with random weights, the algorithm is trained by getting fed the input data and the corresponding targets. The weights are adopted with a mathematical function until the algorithm has met some minimal error criteria.

Figure 2.5: Model of Perceptron [21]

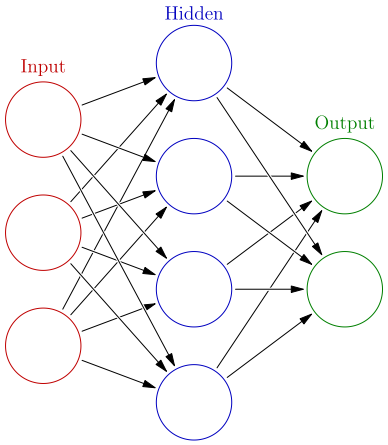


Since the perceptron only can solve linear problems and therefore has quite limited powers of classification, it is more commonly used in machine learning as a basis of other, more complicated algorithms.

Neural Network

One of the most common algorithms that used the perceptron as a basis are the neural networks, sometimes also referred to as multi-layered perceptrons. By combining several layered perceptrons, as shown in figure 2.6, the algorithm can create more complex classes and decision boundaries that are non-linear. The more layers of perceptrons are added, the more complex the classes and the boundaries that separate the classes can be. The algorithm is trained in a similar fashion as the perceptron, but since it contains several perceptrons in layers, the updating of the weights starts from the back, meaning the output nodes and it is updated from back to start with a mathematical function known as the backpropagation function.

Figure 2.6: Model of Neural Network [72]



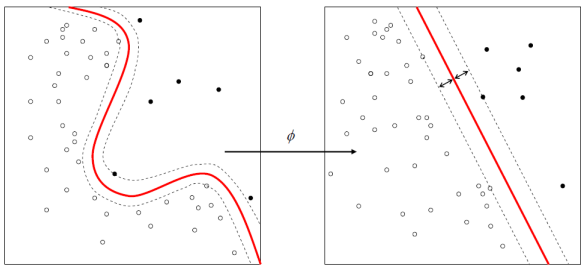
Averaged Perceptron

The Averaged Perceptron Algorithm also utilizes several perceptrons. Instead of connecting the perceptrons, as is done by the neural networks, the Averaged Perceptron uses many versions of the same perceptron. It starts with a perceptron with random weights, and then updates the weights for each input in the same manner as the single perceptron. The Averaged Perceptron, however, stores each version of the weights and then uses the average of all of the version of the weight to make its predictions. The idea is that each of the sets of weights is over-adapted to the last examples it saw and that all of these over adoptions put together are better than the perceptron in itself. Like the single Perceptron the Averaged Perceptron can only separate linear classes, but may be less susceptible to overfitting than its predecessor.

2.3.3 Support Vector Machines

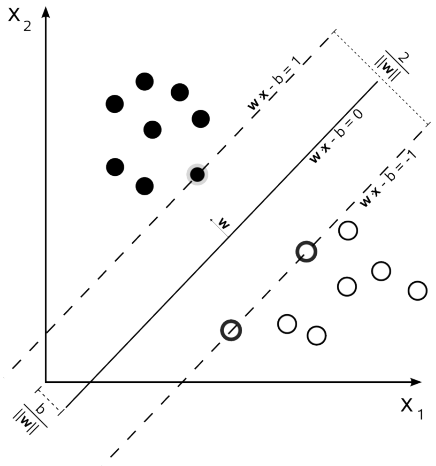
Support Vector Machines, here SVMs, is a classification scheme that, when building the model, utilizes a mathematical function to increase the dimensions of the samples until it can linearly separate the classes in the test set. The mathematical function that increases the dimensions is known as a kernel function; the kernel function transforms the data in a such a way that there is a greater possibility of separable classes. Figure 2.7 shows how a kernel ϕ transforms the data into separable classes.

Figure 2.7: Model of an SVM's Kernel [73]



When the SVM has reached a state where it can linearly separate the classes, it attempts to find the optimal separation. The optimal separation is when the separator is equally far from the closest sample in each class, as shown in figure 2.8. When the SVM has built its model, it can predict on new data by performing the same kernel transformation on the new data and subsequently observe what class it should belong to. SVMs are known for working well on reasonably sized sample sets, but have poorer performance on large sets [51]. SVMs can use both linear and non-linear kernel functions.

Figure 2.8: Model of optimal separation [73]



Pegasos Linear

An extension of the standard SVM, is the Pegasos Linear SVM. The algorithm optimizes the standard SVM, the method alternates between stochastic gradient descent steps and projection step to optimize the kernel [64].

Locally Deep

Another Extension of the SVM is the locally deep SVM. The algorithm speeds up SVMs with non-linear kernel functions prediction while maintaining classification accuracy above an acceptable limit [45]. The algorithm

uses multiple kernels. It aims to learn a different kernel, hence classifier, for each point in feature space.

2.3.4 Bayes Point Machine

Like the SVMs the Bayes Point Machine also utilizes a kernel function. Instead of finding a dimension where there is linear separability and using only this decision boundary, the Bayes Point Machine attempts to separate the classes in several dimensions, and stores each attempted separator. Subsequently it approximates a Bayes-Optimal, or minimal error, intersection of the attempted separators and utilize it to make its predictions. The intersection is a midpoint of the region and bisects the space into two halves of equal volume and is known as the Bayes point. [42]

2.3.5 Logistic Regression

Logistic Regression is, despite its name, not a regression algorithm but a binary classifier. The model estimates probability of a binary outcome based on some features. It works by measuring the relationship of the variables and estimates probabilities using a cumulative logistic distribution. The model is fast to train, but is limited to linear models.

2.3.6 Ensemble Learning

Ensemble Learning utilizes multiple learning algorithms to obtain better predictive powers. The idea behind is that several heads think better than one. A common feature of machine learning problems is that it is impossible to obtain perfect classifiers, which opens up the possibility that different algorithms, with different biases, strengths and weaknesses together yield better results than a single learner. The learners are trained independently and predictions are combined in some way to make the overall prediction. There are several ways of combining the independent learners and ensuring that they do not yield predictions that are too similar to each other.

Perhaps the most common algorithm for ensemble learning is known as Boosting. Boosting incrementally builds the ensemble model by training each new model instance to emphasize samples were the previous models miss-classified [36].

Bagging or Bootstrap aggregating, often abbreviated bagging, uses multiple learners that have equal weight in the ensemble committee. Variance is achieved by training each model on a randomly drawn subset of the data [18].

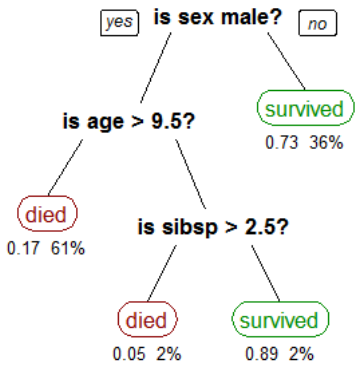
Stacking trains several different machine learning algorithms on all of the available data and then uses a combiner algorithm that is trained to make a final prediction using both the predictions from the and the

available data. Any machine learning algorithm may be used as a combiner, however, single-layer logistic regression is commonly used. [74] Stacking is at times referred to as meta ensemble learners [30] and will be referred to as this in the thesis, just to separate standard machine learning algorithms from stacked algorithms.

Decision Trees

Decision Trees are not Ensemble Learners by their self, but serve as the basis for several ensemble learning algorithms. The Decision Tree is a structure similar to flowcharts, where each node contains a test on a feature, each branch represents the outcome of the test, and every end node contains a class label. An example classifying the chance of survival on the Titanic is shown in figure 2.9. When decision trees are used as a classification algorithm, trees are built by splitting the data set into subsets based on the outcomes regarding a feature and create corresponding sub trees. This is performed until each derived subtree leads to an end node.

Figure 2.9: Model of a decision tree [71]



Decision Forest

The Decision Forest is an algorithm that ensembles decision trees. It creates a multitude of decision trees and the output is the mode of the predictions of the individual trees. It may use several of the ensemble techniques presented, but does most commonly utilize bagging for the random selection of features to construct different decision trees. [61]

Decision Jungle

The Decision Jungle is a modification of the Decision Forest. Instead of allowing only one path to every node in the trees, the Decision Jungle allows for a several paths from the root node to the end nodes. This changes the structure of the trees and complicates the learning process, but does, however, allow the trees to be smaller and consume less memory

and computational power, while also outperforming the Decision Jungle in some cases. The class is found by using directed acyclic graph search. [65]

2.3.7 FastTree

FastTree is yet another extension of the Decision Forest. What differs the FastTree from the aforementioned algorithms is that it uses a nearest neighbor's scheme to reduce the length of the individual trees. It is, as the name suggests, a faster implementation of the Decision Forest, but in certain cases it performs better than the Decision Forest [58].

2.4 Predicting the stock market

As covered, the stock markets are influenced and moved by probably countless factors. Finding out which factors are the most significant is the first of several monumental challenges. By including too many factors for prediction, one is bound to find correlations, but unfortunately not causation. The topic of this is covered in DJ Leinwebers article *Stupid Data Miner Tricks, Overfitting the S&P 500* [48], where the author shows that the stock market S&P 500 were nearly perfectly correlated with the sheep population and butter production in Bangladesh for over 10 years.
As previously covered, we know that there are a lot of factors contributing to a stocks pricing. We also know that the factors that effect the stock market continuously change, leaving a predictor of a stock market in an awkward situation. By including too many factors for the prediction, we are bound do find correlations without causation. By including too few factors for the prediction, we may not be able to predict the market sufficiently well. Adding to an already difficult problem is the ever changing nature of stock markets makes predicting them even more challenging.

2.4.1 State of the art

There are vast numbers of papers and articles about the subject of stock market prediction using Machine Learning. Researcher from Japan have found that that when comparing machine learning algorithms using weekly data on Japanese stock market NIKKEI 225, a SVM outperformed its competitors [44]. The paper used several of the machine learning algorithms presented here, such as Neural Network and Decision Forest, but found that integrating SVM with the other classification methods made it the top performer. The fact that their combining model outperforms the other prediction (forecasting) methods gives hope for this thesis' ensemble learning approach to prediction of stock markets, as ensemble learning methods works by combining machine learning algorithms.

Perhaps the most comprehensive research paper concerning stock market prediction with machine learning algorithms is the paper *Surveying*

stock market forecasting techniques – Part II: Soft computing methods by GS Atsalakis. The paper surveys more than 100 research papers related to using neural networks and other algorithms for stock market prediction and compares the different approaches. The paper concludes that the neural networks has had the highest performance of the machine learning algorithms. But the paper also shows that there are several problems with selecting the appropriate amount of nodes and layers for the neural networks [5].

Ensemble Learning methods have also been attempted and researched for stock market prediction. It has been shown that ensemble learning techniques may represent the stock indices behavior accurately [22].

The research on predicting the stock market with machine learning is somewhat schizophrenic. Some papers state that SVM is the optimal algorithm, while other claim the Neural Network is the best choice and others claim Ensemble Learning Models will outperform them all. This might be an indication that different machine learning algorithms will perform differently on different stock markets and with different inputs, exemplifying the No free lunch theorem, and indicating that results yielded on the Norwegian stock market with a certain set of inputs might not be representative for other stock markets and inputs. It will however still be interesting whether results found in the experiments will support previous findings, which may imply whether these differences are caused by the behavior of different stock markets, the choice of input for the learners or the learners themselves.

Chapter 3

Methodology

This chapter describes how the results were obtained. Section 3.1 gives a brief overview of the test environment. Section 3.2 describes and rationalizes the selection of the data used in the experiments. It also explains the preprocessing of the data. Section 3.3 describes the implementation of the algorithms. Section 3.4 outline how to understand the results and the performance measures. Finally, section 3.5 discusses the limitations of the thesis.

3.1 Test Environment

The aim of the tests carried out for this thesis were to:

Train a selection of two-class ensemble learning algorithms to predict daily stock movement, and compare their performance to each other and non-ensemble machine learning algorithms.

First, stock prediction was made into a two-class prediction problem. This was done by breaking the problem down to whether or not a stock would rise with more than a threshold in N days. After some preliminary testing, a decision was made to use the threshold of 0.05% predicting 1 day, more on the preliminary tests in 3.2.1, making the problem into a true/false question for the machine learning algorithms:

Based on the knowledge we have today, what is the probability that a stock will rise with more than 0.05% by end of the day tomorrow.

The machine learning algorithms were trained using a selection of data that is further discussed in 3.2.1 using data from 5.10.2011 to 30.09.2014, and attempted to predict the daily movements in the period from 1.10.2014 to 1.7.2015.

Simply training and testing the algorithms on one or a few stocks would be an unfair way to evaluate the performance of the algorithms, as stocks may increase or decrease vastly in value in the test period, creating biases. Therefore, nearly all the stocks in OBX were included in testing to avoid biased results.

37 machine learning algorithms and ensemble learning schemes were trained on 22 stocks and their performance evaluated by Accuracy, Precision, Recall, F-Score and a simple Profit estimate. All tests were carried out for every one of the 22 stocks and the mean, median and quartiles of the results are discussed in the results chapter.

3.1.1 Implementation Notes

Implementing Machine Learning Algorithm is time consuming, prone to errors and notoriously difficult to optimize. To streamline the machine

learning process, *Microsoft Azure Machine Learning*© [24] environment and the enclosed Machine Learning Algorithms were used for every algorithm. The environment performs all computations in an auto scaling cloud environment which makes it ideal for computational heavy modelling such as ensemble learning. Using only one environment and no other external libraries or self-made algorithms made the process efficient and less prone to errors and mistakes. The Algorithm modules are covered in details in 3.3. All data preprocessing was done by using Python and R scripts, that are further described in 3.2.2, with the exception of cross-validating which used *Microsoft Azure Machine Learning*© built in module.

3.2 Data

3.2.1 Selection

What and why

As discussed in chapter 2, choosing data for building a model to predict the stock market might be an impossible task. There are countless papers, theses and books written on the subject, and experts seemingly cannot agree upon what an optimal set of data could consist of. There is seemingly no viable way to safeguard the data selection from omitted variable bias discussed in section 2.1.9. Efforts were made in this thesis to choose data that represented the Norwegian stock market as a whole, and also detailed information about the stocks that the machine learning algorithms were to predict. The indices from the G20 stock exchanges were also included, as they give a good representation of the world economic as an entirety. Currency rates were included, as it holds valuable information on how the Norwegian economy is doing compared to other economies. Lastly the Brent Oil price was included, as many Norwegian stocks on the OSE are highly dependent on the oil business.

It may be argued using this much data could lead to overfitting and findings of correlation without causation. However, other will argue that not enough information was included. Data such as macroeconomic data or interest rates is needed to make a valid model of any stock movement. The selection of data is an attempted compromise between too much and too little data, and can surely be criticized both for including and excluding too much data. The stocks in the OBX index, which consists of the 25 most liquid companies on the main index of the OSE, were included, with the exception Aker Solution, Marine Harvest and BW LPG. These companies did major restructuring during the time period (i.e. the companies were divided into smaller parts, changed names and acquired or merged with other companies). And if they were to be included in the model, a serious attempt of adjusting for the restructuring would have been necessary. Adjusting for such restructuring is worth a thesis in itself, therefore these stocks were excluded for simplicity. The stocks included are shown in figure 3.1

Table 3.1: Stocks included in the experiments

DNB	DNO	FOE	FRO	GJF
MHG	NOD	NHY	NAS	OPERA
ORK	PGS	REC	RCL	SCH
SEADR	STL	SUBC	TEL	TGS
YAR	STOR			

For each of the aforementioned stocks the data included is:

- Best Bid Price
- Best Ask Price
- Open
- High
- Low
- Volume
- Closing Price

All these data were included to give a moderately detailed picture of the stocks' daily movements without having too much data.

Also Norwegian indices from OBX were included. Note that these indices represent a weighted average of the same stocks as included but are adjusted on different parameters. This data was included to give a picture of how the Norwegian economy is performing adjusted to some conventionally used indicators.

- OBX International Index
- OBX Price Index
- OBX Total Return Index
- OBX Volume-weighted Index

As discussed in section 2.1.3 oil prices have a massive impact on many stocks listed on OSE. Therefore, it was natural to include oil prices. The decision was made not to include other Raw Material prices, as few or no other raw material prices normally makes much of an impact on the OBX index. Included of raw materials the closing price for:

- Crude Oil, West Texas Intermediate (WTI)
- Crude Oil, NYSE

An attempt of including indicators that represent the world economy as a whole was also made, here by the closing price of the G20 nations highest volume indices. Obviously these indices may be some of the best indicators for the current state of the world economy. All of the included indices can be seen in table 3.2.

Table 3.2: International Indices, country and name

USA	S&P 500	Italy	FTSE MIB
USA	NASDAQ Composite	Russia	RTSI
USA	Dow Jones	India	BSE Sensex
USA	Russell 1000	India	S&P Nifty
USA	Wilshire 5000	Canada	S&P TSX Composite
China	Shanghai Composite	Australia	All Ordinaries
China	Hang Seng (Hong Kong)	Spain	IBEX 35
Japan	Nikkei 225	Mexico	Mexbol IPC
Germany	DAX	South Korea	KOSPI
France	CAC-40	Indonesia	Jakarta
Brazil	Bovespa	Argentina	MERVAL MXX
UK	FTSE 100		

The exchange rate between the Norwegian Kroner and other currencies is an indicator of the Norwegian economy as a whole and was therefore included. The exchange rate towards the Scandinavian neighbors Sweden and Denmark were included, as well as the five largest currencies in the world.

- 100 DKK (Danish krone)
- 1 EUR(Euro)
- 1 GBP (Great Britain Pound)
- 100 SEK (Swedish krone)
- 1 USD (United States Dollar)
- 100 CNY (Chinese Yuan)
- 100 RUB (Russian Rubles)

Granularity

Stock market data and financial data in general exist in resolutions ranging from milliseconds to decades. Higher resolution offers more comprehensive details, but is likely noisier. High resolution financial data are also often not publicly available and reserved for those willing to pay premium prices. Weekly, monthly and yearly data are probably free of the noise often associated with daily and higher resolution data. The downside of using low resolution data is that it greatly limits the amount of data points.

The compromise was to choose daily data points. Not riddled with intraday noise and easily available for free, but not of such a low grain that relevant data became sparse and the training cases to few. Daily data points are by no means perfect, problems arise with weekends, holidays and seasons. Any day is not the same as others, stock markets behave differently on Mondays and Fridays. To counter this problem, the day of the week was also included in the input to the learners.

As discussed in section 2.1.3 there is also noise associated with daily financial data; as an example news and rumors can create noise that extends

beyond intra-day and affects closing prices but are quickly adjusted for the next day.

Time Period

After settling on a daily resolution, the time period of the test data needed to be set. As discussed in chapter 2 using too old data could make the model include outdated information. On the other hand, not having enough data would make it impossible for the machine learners to create an adequately good model for prediction. The financial crisis of 2008 sent OBX down 54.1%. The market adjusted itself and in 2009 the index went up by 68.6% and in 2010 the it went up by 15.67 % up, making the movements in these years highly abnormal. To avoid these abnormal years, data was gathered from 06.10.2011 and to the day of the first experiments for this thesis 01.07.2015, making it a total of 934 days of stock data to train and test the learning algorithms.

Preliminary Tests

There are numerous ways of training a machine learning algorithm. One of the important decisions was to set the time line for the prediction, meaning how many days of data that were used to predict how many days forward. If we were to use 3 days of data to predict 1 day ahead, that would mean that if the current day was a Thursday, data from Tuesday, Wednesday and Thursday would be used to predict the stocks movement on Friday. A simple way of deciding these parameters was to perform some early tests and use the results to set the parameters. Due to time constraints, the preliminary test could not be performed on every stock. These preliminary tests were performed with Schibsted (SCH), which was chosen at random.

The results for the test with regard to the number of days of data to use in the prediction are laid out in table 3.3. 5 days of data yielded the highest profit and precision, which, as is discussed in section 3.4.1, may be regarded as the best indicator of performance for the problem at hand. For that simple reason, 5 days of data were used as input to the learners. More on how the data is structured using 5 days of data as input in section 3.2.2. In the background the curse of dimensionality was discussed. These preliminary results shows that more data, meaning here more days of data, yielded better results till 5 days of data. This indicates that the Curse of Dimensionality problem is, for the problem at hand, less influential than the need several days of data, meaning a great number of features. The choice was therefore taken to not focus on feature selection, but rather attempting a broad range of machine learning algorithms.

Table 3.3: Preliminary results of days included for input

Days	Profit	Accuracy	Precision	Recall	F-Score
1	0.97978	0.582782	0.434879	0.218911	0.247108
3	1.04782	0.580695	0.385825	0.179732	0.210232
5	1.076791	0.588917	0.445489	0.186265	0.232877
10	1.024078	0.583774	0.369036	0.197351	0.223694

The next preliminary test was to test how many days ahead to predict. The most instinctive idea is to predict the next day. It would however be foolish to assume that this is the optimal way without any testing. So the test was performed and the average of all the learners are shown in table 3.4. Predicting 1 day ahead yielded better results in every category than 2 days ahead. As the results for predicting 2 days ahead were worse in every way it was decided that further tests would likely produce even worse results. This is an interesting result as it is an early indication that there is some relationship between the stock movements and the data from the previous day, and it implies that the stocks movement may be, in some degree, predictable.

Table 3.4: Preliminary results of days forward to predict

Days Ahead	Profit	Accuracy	Precision	Recall	F-Score
1	1.076791	0.588917	0.445489	0.186265	0.232877
2	1.004696	0.586374	0.412599	0.112971	0.177375

The final preliminary test that was carried out was to set the threshold for the two classes, meaning how much a stock needs to increase in value before it is considered a positive. The results of the preliminary tests are shown in 3.5. Both 0.05% and 0% would generate convincing results. Notice that in the histograms in the figures 3.3, 3.2, 3.1, above 0% change has the most even distribution, and the other histograms have a rather uneven distribution of positives and negatives. Even distributions are generally considered better for machine learning, as uneven can create biases. However, perhaps the most interesting result is whether or not it is plausible to derive a profit from the predictions, and therefore the final choice of threshold was to set it at 0.5%, as it seemingly has a larger possibility of yielding a profit.

Table 3.5: Preliminary results of different threseholds

Threshold	Profit	Accuracy	Precision	Recall	F-Score
0	0.962631	0.54036	0.554905	0.595143	0.555264
0.05	1.076791	0.588917	0.460851	0.192688	0.232877
0.1	1.019174	0.719808	0.292927	0.073342	0.10494

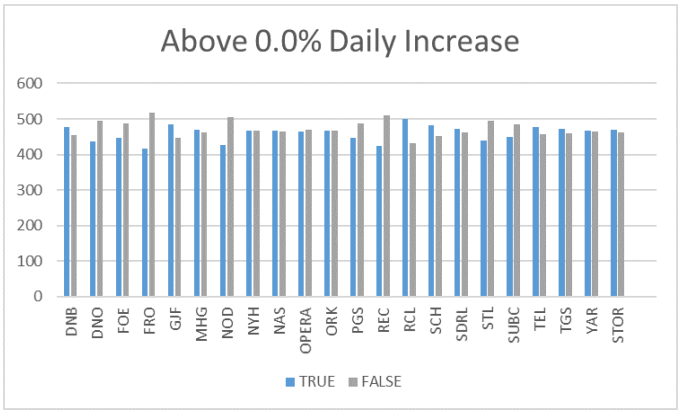


Figure 3.1: Histogram of days above and below 0% change for each stock

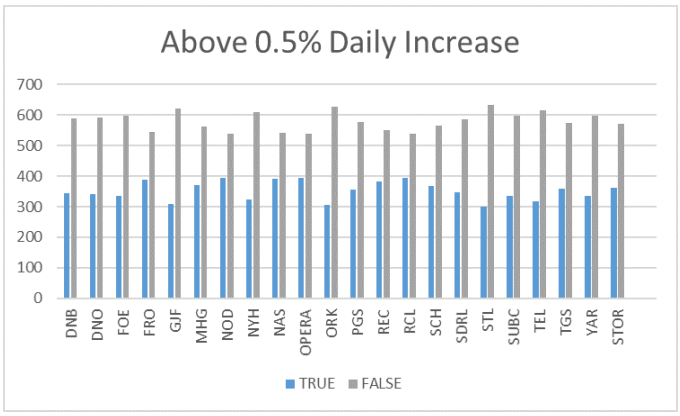


Figure 3.2: Histogram of days above and below 0.5% change for each stock

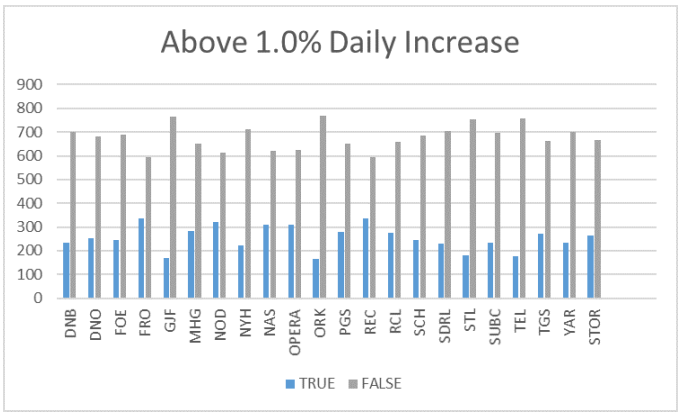


Figure 3.3: Histogram of days above and below 1% change for each stock

Gathering

All stock data were gathered from TITLON [67]. TITLON is a database with financial data from Oslo Stock Exchange for all universities and university

colleges in Norway, managed by the University of Tromsø. Daily Crude Oil prices are made public by U.S. Energy Information department [32] and were gathered from their website . The currency data was collected from OANDA [26] a Canadian-based foreign exchange company providing currency conversion, online retail foreign exchange trading, online foreign currency transfers, and forex information. The International Indices were gathered from Quandl [60], a marketplace for financial and economic data.

3.2.2 Preprocessing

Splitting

First the training data were split into two parts, one for training and cross validating, and another part for Profit Estimation. 80% of the data were used in cross validation, while 20% were used solely for Profit estimation as visualized in figure 3.4.

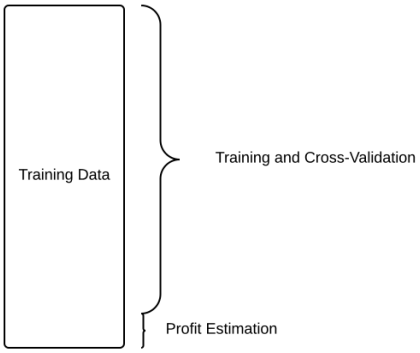


Figure 3.4: Visualization of dividing the data set into two separate sets

After splitting the data, the part reserved for Cross validating, was again split up using K-Fold cross validating as seen in figure 3.5. With K = 10.

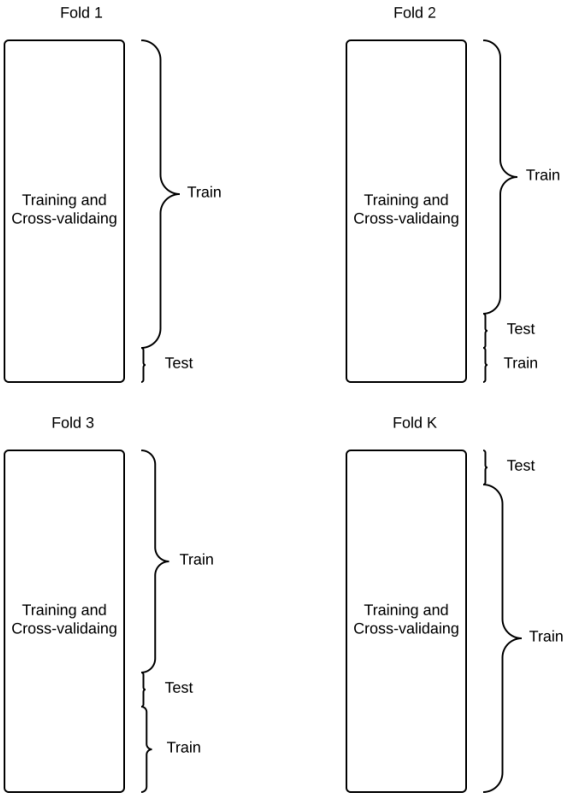


Figure 3.5: Visualization of k-fold cross validation

All of the stand-alone Machine Learning algorithms were trained on the k folds of the training set and the results were evaluated with the target data to get True Positives, False Positives, True Negatives and False Negatives and Scored Probabilities. The Mean of all the folds are used to calculate Accuracy, Precision, Recall and F-Score. Leading us to the next group of the learners. The cross validation was performed by Azure MLs Cross validation module

Time Series

As stocks prices may move in patterns, and are affected by changes happening not only the previous day, but may be affected by changes happening several days ago, simply including data from the previous day may not be sufficient to adequately predict a stock. To counter this problem several days of data were included as input for the learners. After preliminary testing presented in section 3.2.1 the decision was made to use 5 days of data as input, increasing the dimensionality of the input from 187 to 936.

Looking at Change

The data was initially prepared as an array of all the data for each day. However, for data such as Best Bid Price, Best Ask Price, Open, High, Low and Closing Prices for each stocks, currencies and indices, the important information for a machine learning predictor does not lie in the real value. The data that should be included is the change from one point in the to another. Therefore, the real values were not fed to the learners, but rather the change between the days. This was calculated with equation 3.1. Volume (i.e. numbers of stock traded that day) and the day of the week was the only indicator that was fed to the algorithms as real values and not as the change from the previous day.

$$\frac{new - old}{old} = change \tag{3.1}$$

Normalization

As discussed in section 2.2.9, in order to optimize the machine learning process, data should always be preprocessed in a way to avoid biases from anomalies in the data. Normalization can be thought of as an adjustment of values to a common scale, removing the disproportionate force of outliers. Z-Score is one of the more common ways of normalizing data, and was chosen for this thesis. Z-Score is a statistical measurement of a data point relationship with the mean of all the data. This method was chosen because it preserves range, i.e. minimum and maximum, and introduce dispersion of the data set, making it possible to see whether or not a data point is typical or atypical for the data set. Azure Machine Learning's Normalize module was used, where each data point is calculated by:

$$Z = \frac{X - \mu}{\sigma} \tag{3.2}$$

where Z is the standard score, X is the real value, μ is the mean of the dataset and σ is the standard deviation of the dataset

Missing Data

With the choice of using financial data from several countries some problems concerning incomplete data arose. There are different holidays in different countries, meaning that the stock exchange is closed on certain days in different countries. American Stock Exchanges are for example not open on the US Independence Day the 4th of July, while the Norwegian Stock Exchange is open for business this day. Also all stock exchanges are closed on weekends, while the Forex Market, where currencies are traded, is open every day of the year.

Missing data need to be handled for optimal Machine Learning. Since the object of this thesis is to predict stocks listed on the Norwegian Stock Exchange the choice was made to use OBX as the reference, meaning that

data from days when OBX was not open were removed completely from the dataset. This can lead to some biases, especially in the Norwegian summer holiday when the OSE is closed for three weeks while other Stock Exchanges are open. The problem is that since we are looking at change, there is a great likelihood that there have been large changes in the weeks OSE are closed, giving some skewed data.

The data that are missing when OSE was open and other stock exchanges were closed were replaced with zeros, meaning no change. This is not optimal, and both the removing and replacing may lead to biases. However, removing all days where any stock exchange was closed would mean removing a lot of data from18 different countries with several unique holidays. This could have led to data sparseness, which obviously could be worse for the machine Learners.

Target Data

The final decision of using 0.5% positive change as threshold for was taken after the preliminary test 3.2.1. This was performed with the simple equation 3.3

$$target = \begin{cases} 1, & \text{if } change > 0.5 \\ 0, & \text{otherwise} \end{cases} \tag{3.3}$$

3.2.3 Talking Data

Data Structure

Figure 3.6 is an attempt of visualizing one sample input for the machine learning algorithms. Day N is the current day, and day N-1 is the day before and so on. Day N+1 the next day and therefore what we attempt to predict or simply the target. Data for each day consists of normalized data regarding stocks, currencies, indices and raw materials. The stock data consists of more detailed daily data. All the data except the stock volume data and day of the week are prepared as percentage changed. There is a total of 187 features for each date and an extra variable stating the day of the week making it totally 936 features for each target. As mentioned, all data were daily in the time period from 5.10.2011 to 1.7.2015.

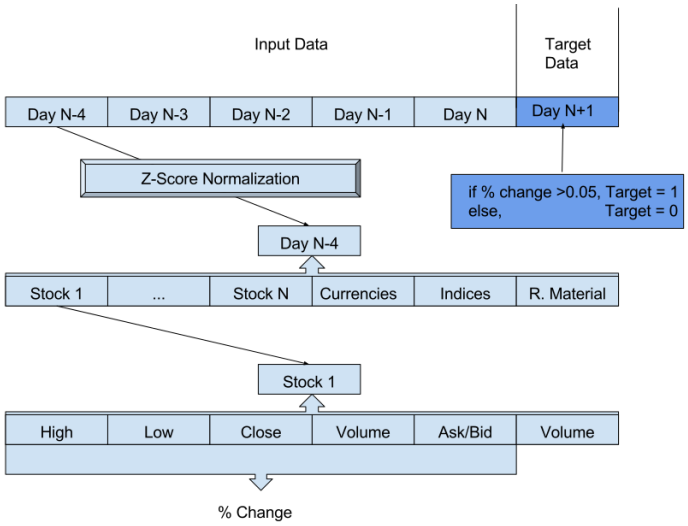


Figure 3.6: Visualization of how the data was made to a time series

Market Trends

When using historical financial data, it is important to note that in most time periods there is generally an upward trend, as most stock markets rise over time. This is also the case for the time period the training data covers in this thesis.

This is, however, a problem for anyone attempting to predict stocks, as sudden shocks in the stock market do occur. There is no such shock in the training data, and therefore any model built with the training data is unlikely to be able to predict such a shock as the financial crisis of 2008. The period after the financial crisis in 2008 is biasedly prone towards an upward trend because there has been worldwide governmental subsidies, bail out packages and low interest rates and other efforts to stimulate economic growth. To understand what the learners should be able to predict, what they are unlikely to predict and what kind of biases they are likely to have, we will have to do some analysis of the training data and the test data.

Since the dataset was divided into one part for testing and another part for training, the datasets need to be reviewed both as a whole and each test set individually to better understand the results.

World Economy

There are many ways to review how the world economy was behaving in the time period of the training data. First let us look at the trends at NYSE, the world's largest stock exchange, which could to some degree reflect the world economy as an entirety. Figure 3.7, shows the movements of NYSE through both the training and test period, the vertical line represent the separation of the sets. As we can see from the figure, NYSE has an upward

trend through the entire sample period. And there is not any particularly obvious difference training and test set.

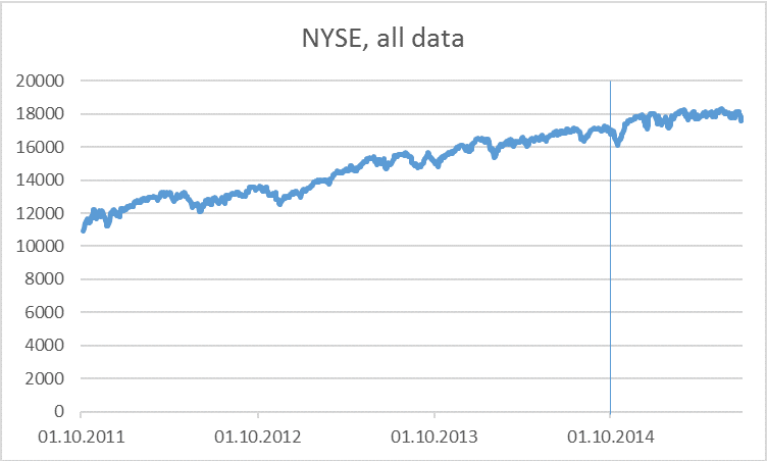


Figure 3.7: NYSE, trends for the entire period

Another indicator is an average of all the G20 stock exchanges. As we can see from figure 3.8, the world economy has also had an upward trend in both the training and test period.

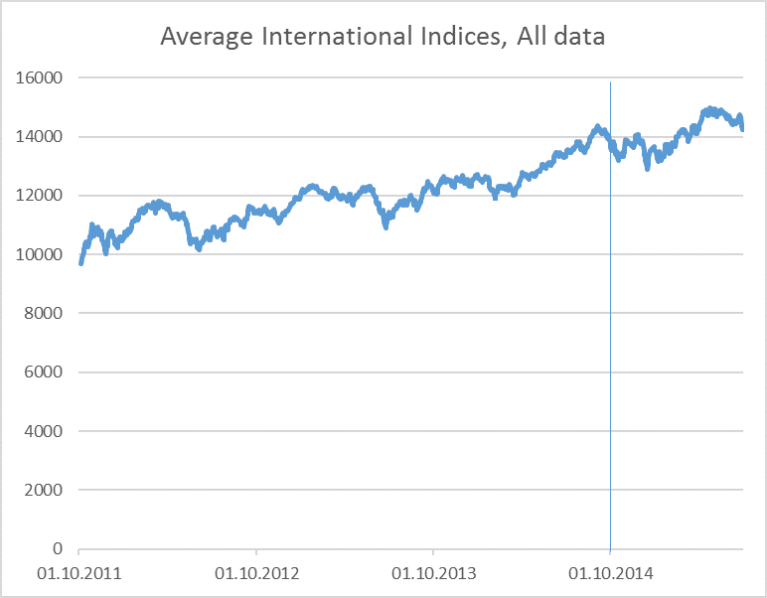


Figure 3.8: Average of all international indices, trends for the entire period

Norwegian Economy

The Norwegian economy, here represented with a weighted version of the OBX index can be seen in figure 3.9, This is a weighted average of the stocks that are attempted predicted in the experiments. It is somewhat similar to

that of the world economy, a general upward trend. There are however two dips in the test period, which may negatively influence the results.

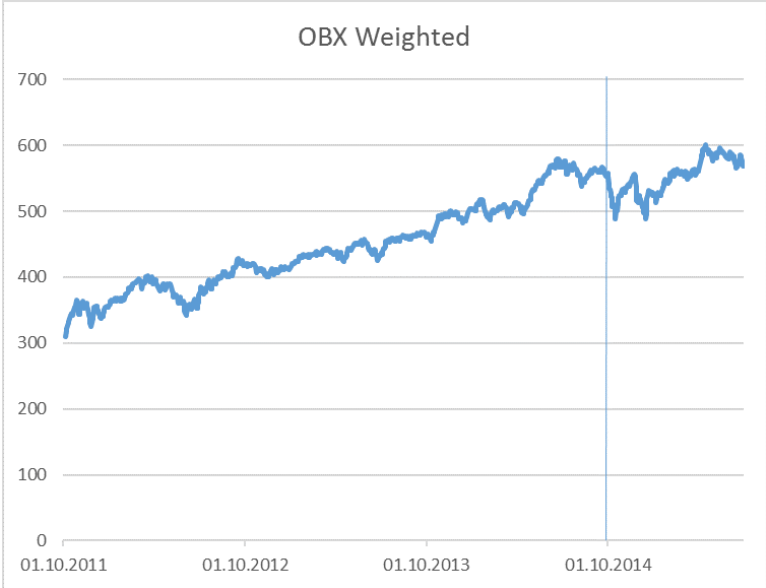


Figure 3.9: OBX weighted, trends for the entire period

When using indicators from foreign economies, comparing the Norwegian economy to foreign economies can be important. Inspecting the Norwegian currency against other may give insight into the movement of the stocks listed on the Norwegian stock market, and can be seen in figure 3.10. We can observe that the Norwegian krone has a particular weak period in the test period, and that nearly all of the other currencies increased in value against NOK in the test period.

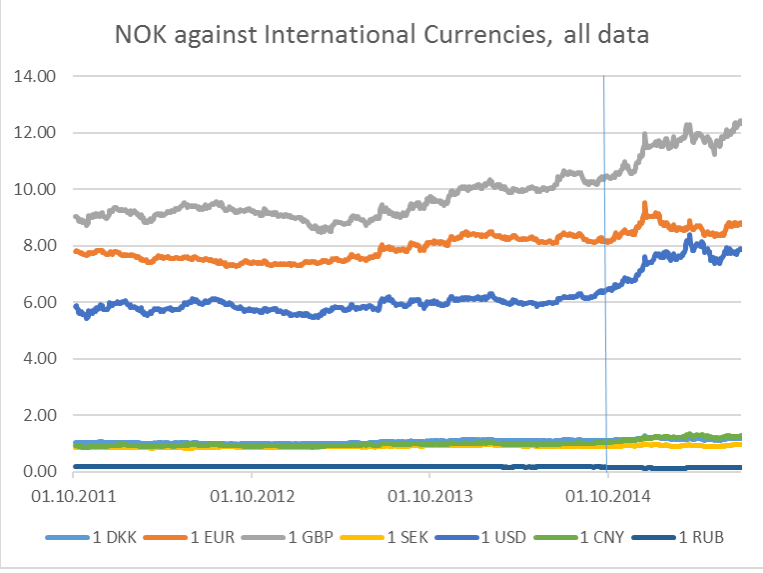


Figure 3.10: NOK against all currencies, trends for the entire period

The Oil price movements in the can be observed in figure 3.11. The Oil price plummeted to below half of its highest value in the training period. This may be the explentaion for the drop in NOK agianst other currencies, and why the Norwegian stock markets underperforms, to some extent, in the test period.

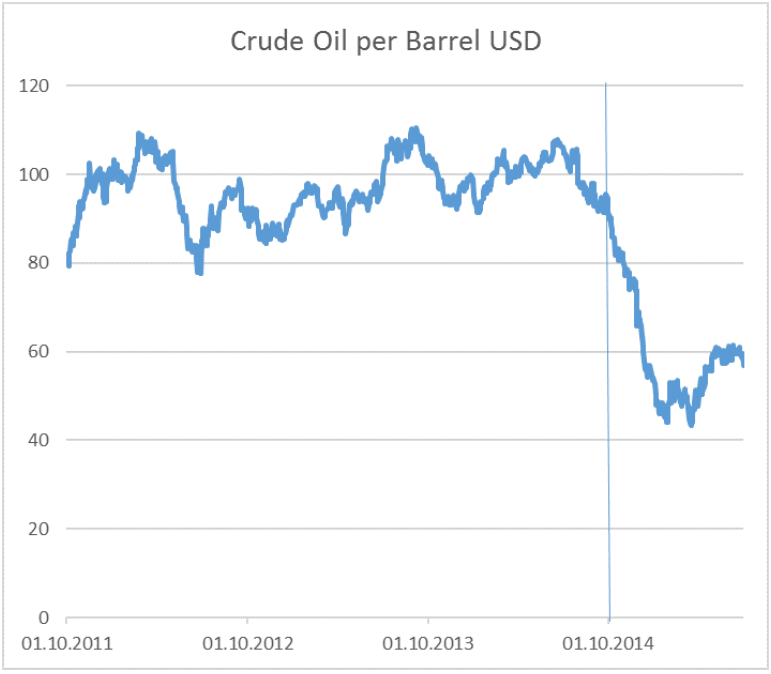


Figure 3.11: Crude Oil per barrel, trends for the entire period

As a final review of the data included, we can observe that the world economy was in an, perhaps unnatural, positive state through both the training and the test period, while the Norwegian economy was in a similar positive state through the training period, but around the time when the test period started, the Norwegian economy stagnated, likely due to the dropping oil price. Since this thesis attempts to predict Norwegian stocks, the fact that the test period is a period of low growth compared to the rest of the world may influence the results negatively.

3.3 The Algorithms

3.3.1 How the modules work

The Azure Machine Learning modules work by connecting a model, like Two-Class Neural Network in figure 3.12 and data into a Cross-Validate Model. After running what is shown in the algorithm, the cross-validated model outputs two sets of data. The first data set is a list of scored probabilities for each input and the second set is performance measures. The performance measures are Accuracy, Precision, Recall and F-Score,

which are further discussed in 3.4.1. The machine learning model that is built can be stored and be used for other input.

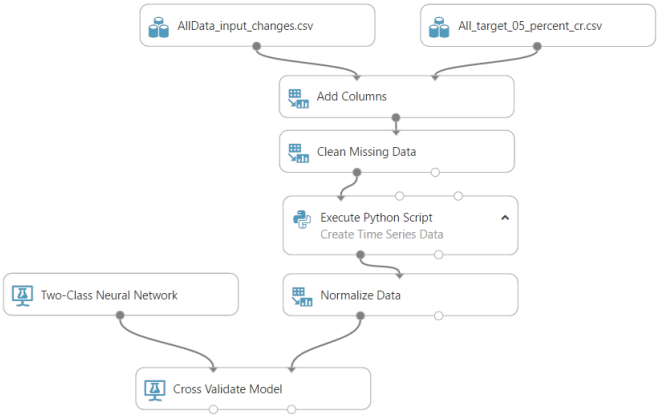


Figure 3.12: How a machine learning algorithm is trained in Azure Machine Learning Studio

3.3.2 Machine Learning Groups

As mentioned, 37 machine learning algorithms were tested. The algorithms are in this thesis divided into three subgroups:

Standalone

First there is what will from now be called the Standalone Machine Learning Algorithms. These are the "normal" machine learning algorithms found in Azure Machine Learning, and also in many other libraries. They are in this thesis called standalone algorithms to separate them from stacked or meta machine learning algorithms. For the purpose of comparing Ensemble Learners, the standalone algorithms are divided into two subgroups.

- **Non-Ensemble Learners:**
- Binary Neural Network (short training)
- Binary Neural Network (long training)
- Logistic Regression
- Locally Deep Support Vector Machine
- Bayes Point Machine
- Averaged Perceptron
- Support Vector Machine

- **Ensemble Learners:**
- Decision Jungle
- Fast Tree
- Decision Forest

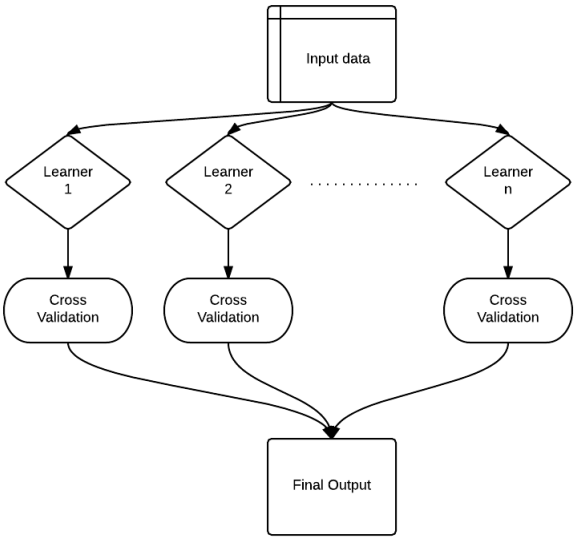


Figure 3.13: Illustration of N Standalone Learners, and how they were cross-validated

These algorithms were fed the training data and were cross-validated as shown in 3.13 and later tested on the test set.

Simple Ensemble Learners

Using only predictions of other algorithms and simple criterias, seven ensemble schemes were created. These are henceforward called *simple ensemble learners*. These algorithms draws inspiration bagging ensemble learners, and uses the output, i.e. the scored probabilities, and their performance measures, i.e. the Accuracy, Precision, Recall and F-Score, from the standalone learners to make its predictions. The seven schemes were created after some simple preliminary testing. They work by waiting for the standalone machine learning algorithms to finish, and afterwards apply a criteria to the combined output to make its prediction as shown in 3.14. The criterias used are shown in list 3.3.2.

- **Criterias:**
- Majority Voting, i.e. over half the learners predicted class 1
- Average Above > 0.5, the mean of the scored probabilities was above 0.5
- Recall * Scored Probability > 0.3 for any of the learners
- Accuracy * Scored Probability > 0.3 for any of the learners
- F-Score * Scored Probability > 0.3 for any of the learners
- Precision * Scored Probability > 0.3 for any of the learners
- Any Above > 0.9 any one of the learners had a scored probability above 0.9

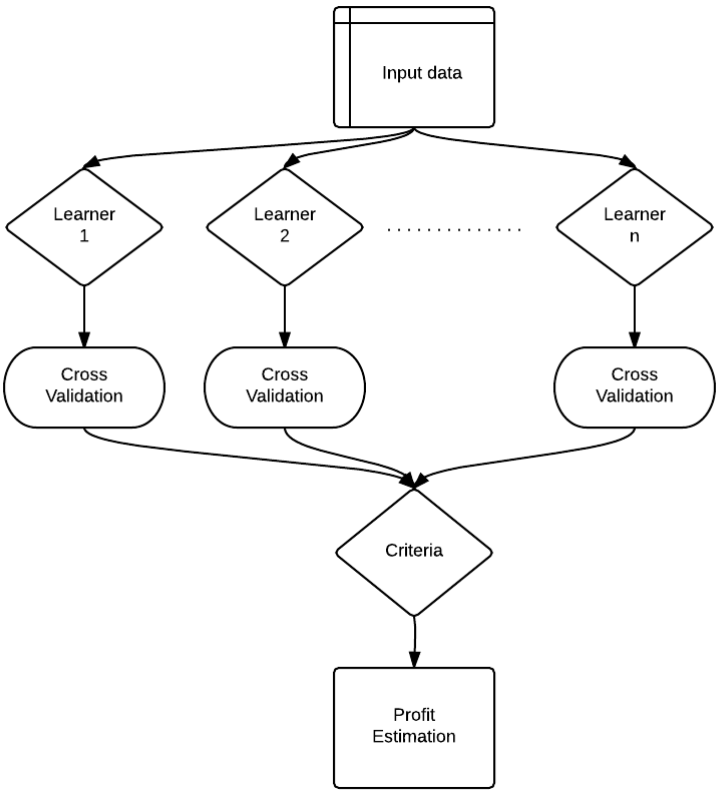


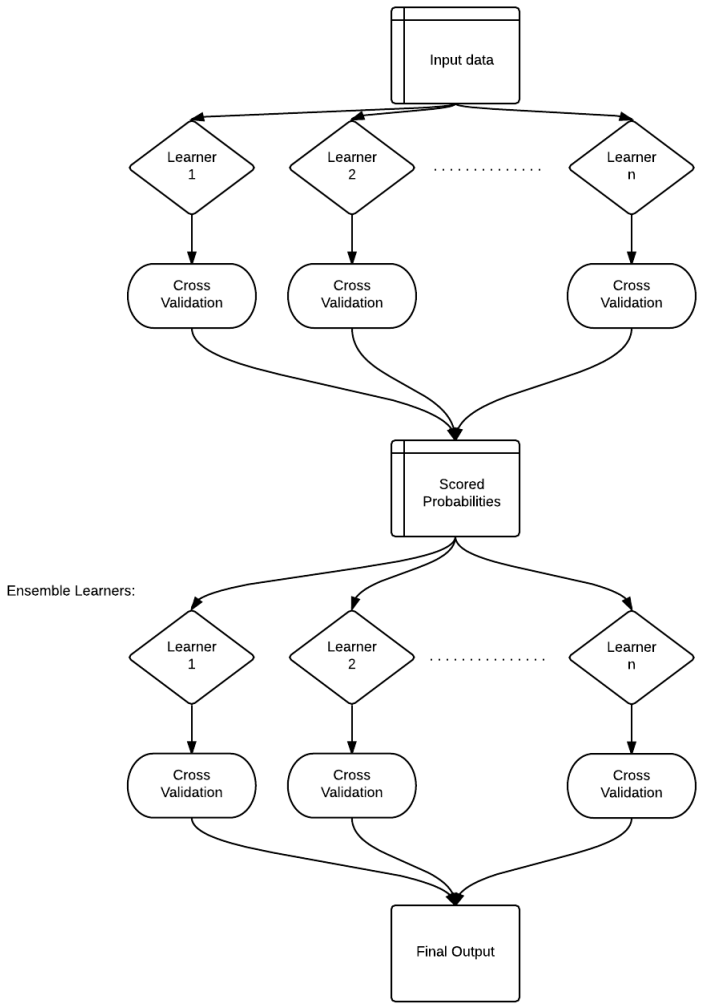
Figure 3.14: Illustration of a single simple ensemble learner

Meta Ensemble Learners

Drawing inspiration from stacked machine learning algorithms, algorithms from now called Meta Ensemble Learners have been created. They work similarly to stacked machine learning algorithms, meaning that they utilize the predictions of other machine learning algorithms as features for their predictions. They are called Meta learners in this thesis, to separate them from the more common stacked machine learning algorithms utilizing a logistic regression as its combiner. The same algorithms that have been tested as standalone learners have been tested using this meta scheme. These can be divided into two subgroups.

The first is The minimal meta ensemble learner. The implementation of these learners uses exclusively the scored probabilities from the standalone learners as their input data. As shown in figure 3.15, these algorithms wait till the completion of every standalone algorithm. Then the algorithm uses the output from the standalone algorithms as its input for learning. The same algorithms that were used for the standalone were used for this minimalistic ensemble learner.

Figure 3.15: Illustration of N Minimal Meta Learners, utilizing N and how they were cross-validated



The next subgroup of ensemble learners are in this study is called maximal meta ensemble learners. These function much like the minimal ensemble learners, in that they wait for the prediction of the standalone machine learning algorithms and use their predictions as input for their learning. However, these algorithms also include the same input data as the standalone is fed. How they work is shown in figure 3.16.

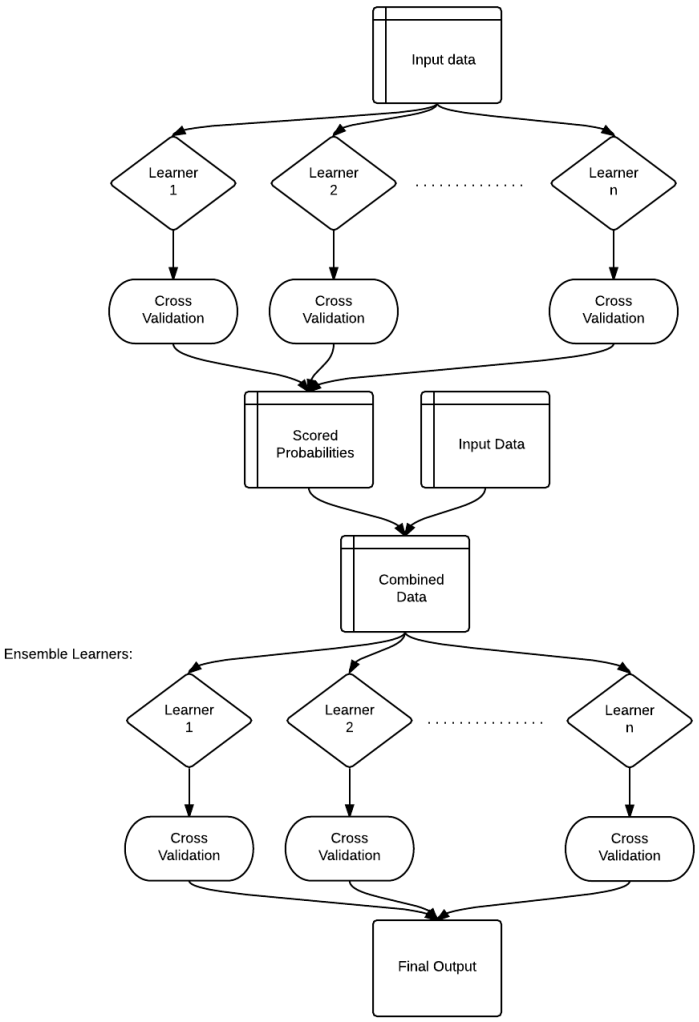


Figure 3.16: Illustration of the maximal meta ensemble learners

Both of the meta ensemble learning schemes were tested with the same algorithms as the standalone, giving us 20 algorithms; 10 for minimal and 10 for maximal. After all the algorithms were trained and evaluated with Accuracy, Precision, Recall and F-Score, a profit estimate was calculated using unseen data.

3.3.3 Tuning Parameters

One of the limitations of this thesis is that the parameters of the algorithms were not fine-tuned or optimized. The parameters were simply set to the default value set by Azure Machine Learning Studio, with the exception of the random seed variable which was set to 0 so that it can be replicated and the neural network which was trained with both 100 and 1000 number of learning iterations. A table of all the possible parameters to tune is included in the appendix 5.1.

3.4 Understanding the results

3.4.1 Performance Measure

In binary classification, some of the most common performance measures are, as we can recall from section 2.2.5, Precision, Accuracy, Recall and F-Score. These are statistical measures that are calculated from the number of true/false positive and negative. For the problem in this thesis, a true positive would be when the learner correctly predicted that a stock would rise with more than 0.05% the next day. False positive would be that the learner incorrectly predicted that the stock would rise. On the other hand, a true negative means that the learner predicted correctly that the stock would not rise, and a false negative means that it predicted incorrectly that the stock would not rise.

If anyone were to trade stock based on the prediction of a binary predictor, it would likely be wise to use an algorithm that rarely predicted a false positive, since a false positive would cause the trader to buy a stock and lose money. For that reason, it may be argued that precision is the best measure, as it is a measure of the ratio between true positives and false positives, or in other words, the ratio of how many times the learner would make a profit compared to how many times the learner would create a loss, meaning that it would minimize the risk of losing money. Because of this, the learner with the highest precision may very well be the best if one wants to make a profit. This would especially be true in bearish periods when the market is not expected to rise.

No risk, no reward. A common phrase in the financial world and in bullish market situations when stocks are expected to rise, higher risk strategies may very well be the most lucrative. Recall measures the ratio of how often the learner correctly predicts that a stock will rise against the number of times it fails or predict that the stock will rise. The higher the recall, the more often the learner will identify growth and thereby profit.

It is easy in retrospect to determine if a period was bullish or bearish, it is however more difficult to know what kind of period it is at the moment. For that very reason both recall and precision may be an inadequate measure as they fail to tell the full story. The F-Score considers both Precision and Recall and is therefore more informative than either of them by themselves. It is calculated by using the harmonic mean of the two values. The learner with the highest F-Score is therefore likely to do pretty well in periods that are bearish, bullish and anywhere in between.

Accuracy is also a more informative measure than precision and re-

call. It simply shows the percentage of correctly classified samples. Much like the F-Score, accuracy is a good measure when there is no a priori knowledge about the future stock market.

All of these measures have been used to evaluate the machine learning algorithms. It is challenging to conclude whether one measure is better than the other. Which measures to use depends on the general trends of the market and the risk profile one is willing to take; it is therefore interesting to examine the learners in regards of all of these measures.

3.4.2 Profit Estimation

The motivation for many financial models and predictions is making a profit. It is therefore interesting to estimate the profit one could expect with the predictions of the machine learning algorithms. There are several ways that the predictors could be incorporated into a financial system, and also many ways that profit could be estimated from these incorporations.

The profit estimation was performed in a simple way that ignores several factors, but should make it possible to evaluate the performance of the learning algorithms. When a learner predicts a scored probability above 0.5 for a day, we assume that the stock is bought at opening and sold at closing. The value starts at 1, meaning that if we never buy a stock, it will remain at 1 and that 1.1 means an increase of 10 %. Every calculation is for an individual stock. It is calculated as in equation 0.

For every day of the training set
  if $ScoredProbability \geq 0.5$ **then**
    $Money+ \leftarrow Money * Change$
  end if

If the learner were to predict a scored probability over 0.5 every day in the test period, it would have yielded a result as shown in table 3.6. Note that this profit estimation does not take into account market slippage, broker commissions and other fees. It is meant to be a purely theoretical measure of performance and not an indication of actual profit.

By performing a profit estimation on every stock included in the experiments, we can divide the stocks into groups based on their performance in the test period. Bearish stocks would be the group that has lost more than 10% of its value, bullish the stocks that have had a value increase of more than 10%, and normal performers is every stock in between. When looking at the profit estimate we should always compare it to how the market as a whole is

performing. Therefore, the average profit estimate is an important measure that will work as a baseline for many of the results.

Table 3.6: Profit estimate for owning the stock through the entire test period

Stock:	Profit Esimate
Average	**1.07**
FOE	0.46
SDRL	0.46
SCH	0.74
OPER	0.77
STL	0.78
SUBC	0.83
NYH	0.92
GJF	0.93
STOR	0.94
DNO	1.00
MHG	1.01
PGS	1.01
ORK	1.07
DNB	1.09
TGS	1.11
TEL	1.22
YAR	1.27
NOD	1.45
RCL	1.46
NAS	1.46
FRO	2.51

3.4.3 Statistical significance

The Mann-Whitney U Test has been used for deciding whether there are significant differences between the results yielded for the different machines learning algorithms and the groups of algorithms. It may be used as a measure, since the assumption of the Mann-Whitney U Test is that there is independence within the sample set and mutual independence, which is true for the independently trained machine learning algorithms used in the experiments.
There are issues with using Mann-Whitney U Test finding out whether or not there are statistical differences between pairs of the results. The Mann-Whitney test is a ranked test that simply shows whether it is likely that two data sets are from different groups. However, it does not take into account that the machine learning algorithms in the experiments conducted are predicting the same stock; this may be an issue if some stocks are more easily predicted than others, as it may bias the results.

A solution to this issue could be to use Wilcoxon signed-rank test,

which could have paired the results from the same stock. However, this test would not have allowed for comparing groups of machine learners, since there are no longer pairs. A solution to the problems of pairs may by using the means or medians of the groups, but that again will lead to new problems; whether or not the mean or median is adequate for the representation of the group.

The main difference between the methods is that the Mann-Whitney is for independent groups and uses only the numerical information. Wilcoxon is for matched groups and also uses information about the pairwise differences. The Mann-Whitney U-test can be used with paired data, and it will simply be less powerful [12]. The Mann Whitney U-Test was still chosen as the test, since most of the results and discussion are about comparing groups of algorithms, and Wilcoxon does not allow this.

3.4.4 Random as a measure

Monkeys have famously been better at picking stocks than expert financial advisors, giving us reason to believe that picking stocks at random may in fact not be an unreasonable way of investing. Research has shown that picking stocks at random in many cases yielded better results than almost any fund and expert advice [11]. Therefore, a set of random stock purchasing schemes were implemented so that we could test against a random procedure as a performance measure. For every stock 3 different normal distributions were tested 10 times, and a profit estimate was made much like the profit estimate for the machine learning algorithms, the difference only being that scored probability was switched with the normally distributed random variable, so that if the random value is above 0.5, then it will buy the stock. The three distributions are plotted in figure 3.17 and are identical, with the exception of different means. The first distribution represents a rather defensive buying strategy and would buy stock roughly 12% of the days. The second would buy 50% of the days and the third would represent a very aggressive strategy and buy 88% of the days One could argue that distribution 1 would do well in bearish periods while distribution 3 would do well in bullish periods.

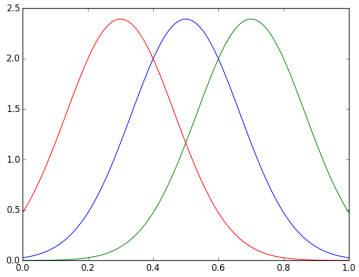


Figure 3.17: Plot of low, standard and high normal distributions

3.4.5 Box Plots

Most of the results are presented in Box Plots. Boxplots is a convenient way of graphically portraying groups of data through their quartiles. Quartiles are the 3 points that divide the group into 4 equally populated groups. As seen in 3.18 the top of the box is the first quartiles, the bottom line of the box is the third quartile, and the line in the middle represents the median. The whiskers represent the range of the data. Flier points are those past the end of the whiskers represented with +.

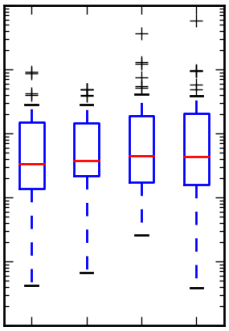


Figure 3.18: Illustration of Boxplot

3.5 Limitations

There are numerous factors that affect how this thesis was conducted. Some of the limitations have direct effects on the results and should be taken into account.

3.5.1 Limited Depths

An attempt was made to include as many machine learning models as possible, for the sake of a broader comparison and to give the meta ensemble learners a rich foundation of data. A total of 37 algorithms were trained and tested and the process was time consuming and computably heavy. This left fewer available resources for

optimization, and little optimization was carried out. This gave the experiments that were carried out a broad scope, unfortunately at the cost of the depth. The effect of not doing an in-depth optimization of each algorithm is that we are unlikely to see the full potential of the algorithms. A low performing algorithm may with a change of parameters become a high performer. We should therefore judge models by their relative performance

The data selection might also affect the outcome of the experiments. Some of the learners might perform better with a large set of data, while other may perform equally or even better with fewer features. Some learners are more prone to overfitting; for these algorithms large input vectors may considerably weaken the prediction power. No attempt of optimize the features for the different algorithms have been made. Feature selection techniques such as Z-Score selection, remains untried. The time consumption of the different learners has not been taken into consideration. Some of the learning algorithms are considerably more time consuming than others, which in many cases would be disadvantageous.

Different machine learning algorithms react differently to normalization techniques, different cross-validation techniques and schemes to handling missing data. No attempt have been made to find a better fit of data preparation for the different algorithms.

3.5.2 Simplified Profit Estimation

Trading stocks is associated with several fees. Each trade in itself have a fee. These fees have not been accounted for. There is also a gap between the asking price and the and bid price forcing a stock trader to accept a less favorable price. Market slippage, meaning the time lag between the sending of the buy or sell to the execution of the order, and other circumstances exhibit why the profit estimation used in this thesis is at best an indication of performance and should not be considered a measure of actual attainable profit.

Using binary classification strategy for stock prediction raises another issue in regards of the profit estimation. The algorithms are simply predicting whether or not the stock will rise with more than 0.05%, it does not take into account how much the stock rises. This has an effect on the profit estimation as the stock might rise 10% one day and 0.06% another day. While predictors are equally correct regarding both days, a predictor that correctly predicted that the stock would rise only on the day of the 10% increase will get a significantly better profit estimate than a predictor only estimating the rise on the 0.06% day.

Chapter 4

Experiments

This chapter describes the results gathered from the experiments. Section 4.1 describes how to read the results. Section 4.2 presents the results. Section 4.3 reflects upon and analyze the results and compares the performance of the machine learning algorithms.

4.1 How to read the result

4.1.1 Diversity and Adjustments

Normally when one is evaluating machine learning algorithms it is common to use a combination of statistical measures and real life performance. The same approach has been used in this thesis. However, profit estimation, which is a type of real life performance, has been emphasized. It is emphasized because like with most applications of machine learning, there is a great desire to have machine learning that can be applied to actual problems.

One of the main reasons that the statistical measures are used is because real life performance measures are riddled with noise and one can easily be fooled by the dumb luck of a machine learning algorithm. This thesis mostly uses profit estimation, and can do so because it analyses the mean of 23 different stock for each learner, severely decreasing the likelihood of chance influencing the results. We know that examining profit estimate of the single machine learning algorithms predictions is not especially interesting as it is prone to biases and chance and may not reveal anything about the algorithms actual prediction power.

In order to make sure that the results are valid within reasonable doubt, Mann-Whitney U tests have been performed on the different results, and a table has been made which shows whether two batches of results can with a significance level of 99.95% certainty distinguish that the two results are inherently different and that there is a significant difference between the sets of results. The tables are laid out so that the intersection between two algorithms or groups of algorithms represents whether there is a significant difference between the two groups of profit estimates. When the p-value is over 0.05, it means that we cannot be more than 95% sure that the results are not the result of noisy data and luck, and these intersections are marked with a capitalized *N* for No. When the p-value is above 0.05 and there is significance it is marked with a capitalized *Y* for Yes in the table. The real P-Values were not included as a measure for readability.

Simply inspecting the real values of the profit yielded has significant downsides, as the performance of the stock in the test period to a high degree influences the result. To separate the performance of the machine learning algorithm from the performance of the stock and

the market in general, all of the results are presented adjusted to the development of entire stock market (*OBX*). The market adjustment means that every profit estimate is subtracted by the profit estimate of owning an OBX index fund through the test period. What this means is that while a learner before the adjustment would have made a profit if its value was above 1. After the adjustment it means that profit is at above 1.0-1.07 = -0.07 and that performing better than the market is above 0.

4.1.2 Categorizing the stocks

In addition to inspecting all the stocks at the same time, the stocks have been categorized by their overall performance throughout the test period. The stocks have been divided into three groups *Bear Stocks*, *Bull Stocks* and *Normal Stocks* as shown in 4.1. The criteria for which category the stock is assigned to is simply whether the stock has increased or decreased with more than 10%, or as we see more formally in equation 4.1

$$Category = \begin{cases} Bull\ Stocks, & \text{if } change > 0.1 \\ Bear\ Stocks, & \text{if } change < -0.1 \\ Normal\ Stocks, & \text{otherwise} \end{cases} \tag{4.1}$$

The categories shown in table 4.1, named *All Stocks*, *Bull Stocks* and *Bear stocks* are grouped together in different plots. The top figure shows the results for all the stocks, the second from the top shows the same profit estimations but only regarding the *Bear Stocks* and the bottom plot shows profit estimates for *Bull Stocks*. *Normal Stocks* plots of these were however not included as they were very similar to the *All Stocks* plots, and may have made the presentation of the results too chaotic. Also the performance of the *Normal Stocks* can be derived from the other results as *All Stocks* is of course the sum of *Bull Stocks*, *Bear Stocks* and *Normal Stocks*.

The reason for including the plots of the Bull and Bear stocks is that it can tell us how the learners act and predict and whether the learners produce defensive or aggressive purchasing strategies. But the perhaps most important insight it gives, is how the algorithms would handle time periods unlike the test period, for example how the predictions would be in very bearish markets. As we remember from section 3.2.3 the test period is a quite normal period, with a slight upward trend, but unfortunately the stock market does not always act in this way. By looking exclusively at the *Bear Stocks* it may give valuable insight into how the learner would act in a crash and the *Bull Stocks* plots gives into how the learner would handle a stock rally. This renders some more certainty in answering a research question on whether it is possible to make a profit, particularly in a long term set-

Table 4.1: Categorized by Profit Estimation, Bull, Bear and Normal stocks

Bear Stocks	**Normal Stocks**	**Bear Stocks**
FOE 0.457203	NYH 0.916528	TGS 1.10581
SDRL 0.457342	GJF 0.930831	TEL 1.221986
SCH 0.738532	STOR 0.935403	YAR 1.274334
OPER 0.770428	DNO 1.002643	NOD 1.451187
STL 0.785388	MHG 1.00723	RCL 1.461378
SUBC 0.827248	PGS 1.008354	NAS 1.461401
	ORK 1.070568	FRO 2.506716
	DNB 1.088113	

ting when both crashes and rallies are bound to occur, and is therefore important in the question of whether it is possible to make a profit or not.

To further understand how the bull and bear stocks should be viewed, we have to look at what is good for a *Bull Stock*. Remember that the average of all the stocks is 1.07 and that adjusted for the market it is obviously 0. Since all of the bull stocks outperform the market as a whole we would expect more of a profit as well. In fact, by simply owning any of the bull stocks throughout the test period we are guaranteed 1.1 - 1.07 = 0.03 above the market. But further scrutinizing the *Bear Stocks* we can observe that the mean of all the bull stocks is 0.427, meaning that we should expect results in this region for high performing algorithms. It is the other way around for the *Bear Stocks*; by owning any of the stocks throughout the period one is guaranteed a loss of at least -0.17 and the mean of the bear stocks adjusted for the market is -0.397, and the results should be inspected with this in mind.

4.1.3 Results Overview

The results that are included, are included as an attempt of answering the research questions posed in this thesis, and a small justification is included for every plot. Most of the results are shown as box plots with an additional mark for the mean. As we recall from section 3.4.5 the box indicates the quartiles, the line the median and it can tell us something about the entirety of the results. All of the results one learner have from every stock are always presented together. Mean is perhaps the most important measure in the box plot for us to look at, as it shows the results of what we could expect to gain if we were to distribute our money equally for the single learner for every stock. This would likely be the most reasonable investment strategy, and is therefore highlighted. The median tells us how large a part of the stocks it is likely to make a profit if we follow the single learner's prediction. The box tells us what we could expect for 50% of

the stocks and the whiskers and outliers show the extremes that one possibly could encounter if you were to put all your money into one stock.
It is important to note that in this chapter the machine learning algorithm types, meaning the standalone learners, minimal and maximal meta ensemble learners, simple ensemble learners are referred to as the machine learning types or most often **groups**. Algorithms such as Neural Network, Pegasos-Linear SVM, Decision Jungle and so on are referred to as **algorithms**. The results are divided into different sections. First there is an overview of the profit estimate for every machine learning algorithm tested on every stock is shown in the first figure 4.1. Later on in the result section, we further examine each group of learners by themselves. Let us take a quick recap of the groups and algorithms presented in chapter 3 and then an overview of all the results presented in section 4.2.In table 4.2 we can observe all the algorithms tested, as we know the same 9 algorithms are tested for both the Meta Ensemble Learners and as Standalone algorithms, and that the so called Simple Ensemble Learners uses 7 different schemes to make predictions.

Table 4.2: Algorithms

Meta Ensemble		**Standalone**	**Simple Ensemble**
Maximal	**Minimal**		
Neural Network			Accuracy
Locally Deep SVM			Precision
Logistic Regression			Sum Above
Decision Forest			F-Score
Decision Jungle			Any Above
FastTree			Recall
Bayes Point Machine			Precision
Averaged Perceptron			
Pegasos Linear SVM			

In table 4.3 we see an overview of the results and how they are presented in the next section 4.2. First all the groups are compared, then the algorithms within the groups are compared. After that we go more in-depth into each group of learners. And in the end we do a summation and scrutinize the absolute top performers from all of the groups and algorithms.

Table 4.3: Overview of results

	Machine Learning Groups	**Performance Measure**	**Stocks**	**Section**
1	Minimal Meta, Maximal Meta, Standalone, Standalone Ensemble, Simple Ensemble, Random Schemes	Profit Estimate Box Plot, Significance Table	All, Bull, Bear	4.2.1
2	Minimal Meta, Maximal Meta, Standalone, Standalone Ensemble,	Bar Chart: Accuracy, Precsion,Recall,F-Score,	All, Bull, Bear	4.2.1
3	Minimal Meta, Maximal Meta, Standalone, Standalone Ensemble	Profit Estimate Box Plot, Significance Table	All, Bull, Bear	4.2.2
4	Standalone, Standalone Ensemble	Profit Estimate Box Plot, Significance Table	All, Bull, Bear	4.2.3
5	Maximal Meta Ensemble	Profit Estimate Box Plot, Significance Table	All, Bull, Bear	4.2.4
6	Minimal Meta Ensemble	Profit Estimate Box Plot, Significance Table	All, Bull, Bear	4.2.4
7	Simple Ensemble	Profit Estimate Box Plot, Significance Table	All, Bull, Bear	4.2.5
8	Minimal Meta, Maximal Meta, Standalone, Standalone Ensemble,	Top Performes Table	All	4.2.6

4.2 Results

4.2.1 Groups Compared

The first results are laid out in 4.1; these results are shown to answer this thesis' hypothesis. The figure is a compact summary of the profit estimates of all the tested algorithms. It shows whether one could expect to make a profit from any of the types of algorithms. It also shows the difference between the groups of algorithms, and whether the meta ensemble learners gained from their additional input. The groups of algorithms are plotted together, meaning that for example the Maximal Meta Ensemble, the Neural Network, Decision Jungle and so on are included in the group. The profit estimates for each one can in the plot be compared with the others and the random distributions. Each data point represents the profit estimate of a single learner's result on a single stock. As an example the one point is Fast Tree for the Minimal Meta Ensemble type for SCH. Table 4.4 shows which of the results plotted in 4.1 that are significant.

Let us inspect figure 4.1 *All Stocks* first. One can observe that three of the algorithm groups, namely Standalone, Maximal Meta Ensemble and Standalone Ensemble learners, have a mean that is slightly better than the market. Observing the p-value table we can see that just in one case we may say with significance that these algorithms outperform the market represented here at 0 by *OBX*. It is however not by much, only 0.06 or 0.6% more profit than *OBX*, a meager result for the algorithm group with the highest mean, namely the Standalone learners. We can however, not say with significance that the Standalone algorithms outperform the Maximal Meta Ensemble Learners or the Standalone Ensemble Learners. We can also observe that the standalone learners have a bigger box, meaning that the distribution is sparser and that the variation within the group is quite large.

After that we see that the random purchasing schemes have a mean lower than OBX, but have both the highest and lowest performers of all the algorithms and schemes. The mean is also, perhaps surprisingly, not much lower than the other algorithms and OBX. The Simple Ensemble Learners and the Minimal Meta Ensemble Learners perform poorer than OBX and significantly poorer than the other machine learning methods.

When observing 4.4 *Bull Stocks* and *Bear Stocks* one immediately notices that the random purchasing schemes vary much more in performance on bull and bear stocks than the learners. The simple ensemble learners also have a greater variation in performance than the other machine learning types. That random purchasing is more influenced by the performance of the stock than the other schemes is not very surprising, and it can be seen as a strength of the machine

learning algorithms that they do not perform badly when used on *Bear Stocks*.

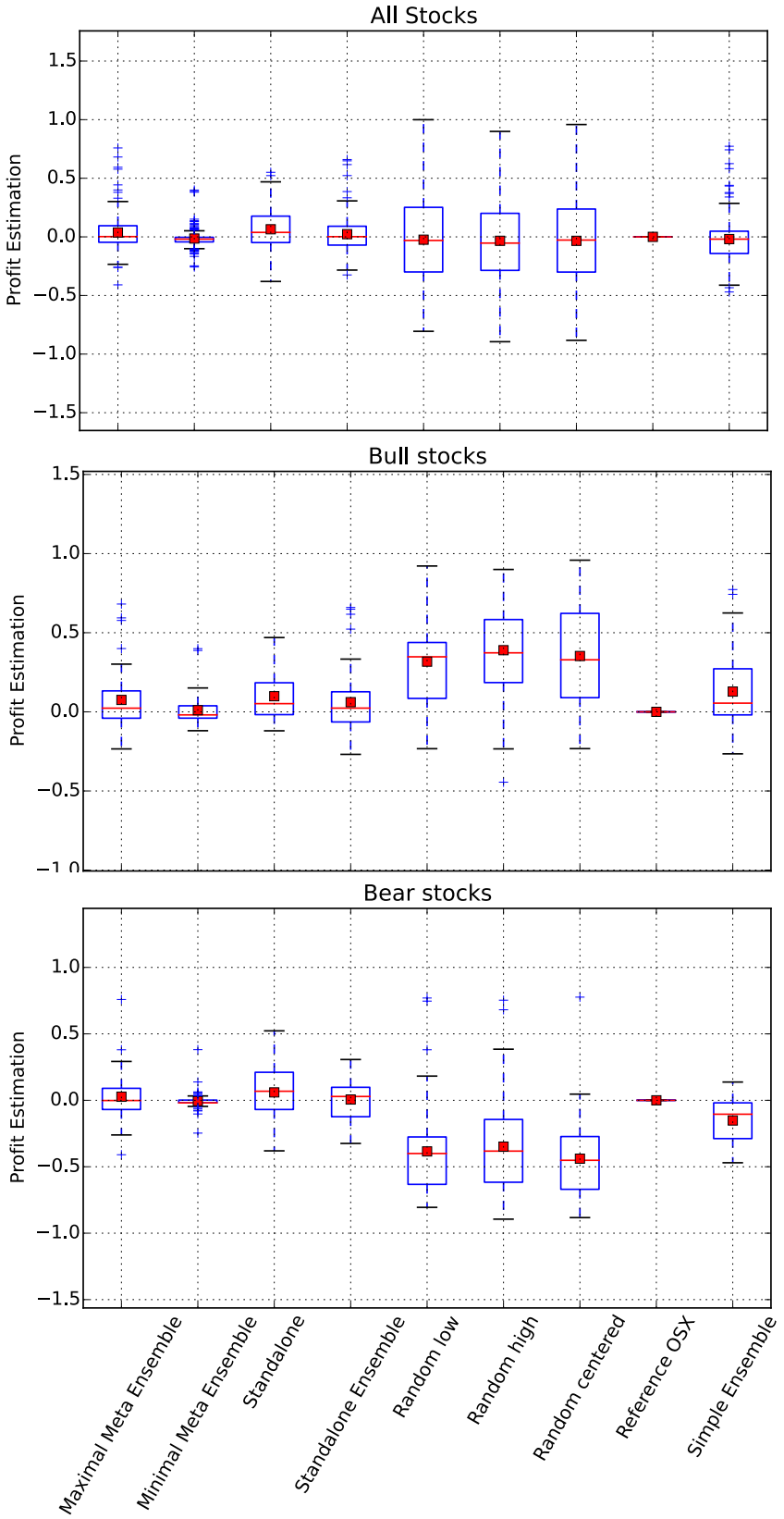
In the *Bull Stocks* plot, the means are generally higher than in the *All Stocks* plot. This comes as no surprise as it is easier to make a profit on rising stocks. What it is more of a surprise is that only the standalone algorithms and the simple ensemble learners have a mean above 0.1. From 2.1.10 we separated the stocks into groups that made a profit larger than 10%. Owning any of these stocks guaranteed a result of at least 0.03 owning an equal share of all of these stocks would yield a 0.43 profit above *OBX*. All of the machine learners and schemes perform below the average results of the stocks. This is an indication that they were to defensive in their purchasing strategy. As we can see the random purchasing strategies perform notably best for the *Bull Stocks*, and best of them all the Random High strategy, which purchases stocks the most often. This can be seen as yet another indication that the learners did not properly manage to separate the bull stocks from the other stocks and purchase these at a higher frequency.

On the other hand we can see that for the *Bear Stocks* there is, yet again quite unsurprisingly, an opposite reaction. There is in general less profit yielded, and there is not much difference between the learners, but the random stocks perform notably poorer. It is impressive that the all of the learners with the exception of the *Simple Learners* managed to yield a profit larger than *OBX*.

Table 4.4: Significance Table, Groups Compared

	Maximal Meta Ensemble	Minimal Meta Ensemble	Standalone	Standalone Ensemble	Random low	Random centered	Random high	OSX
Minimal Meta Ensemble	Y							
Standalone	N	Y						
Standalone Ensemble	N	Y	N					
Random low	Y	Y	Y	Y				
Random centered	Y	Y	Y	Y	N			
Random high	Y	Y	Y	Y	N	N		
OSX	Y	Y	Y	Y	Y	Y	Y	
Simple Ensemble	Y	Y	Y	Y	Y	Y	Y	Y

Figure 4.1: Groups Compared



To further investigate the difference in the groups of machine learning algorithms, this section scrutinizes their precision, accuracy and recall and F-Score for the Minimal and Maximal Meta learners as well as the standalone learners, including the standalone ensemble learners. This is done as an attempt to see whether the meta ensemble learners performed any better because of their extra input from other machine learning algorithms, the top performers from the previous results. The results can be seen in 4.2, where the bars represent the mean and the whiskers the standard deviation. The first noticeable results are that the Accuracy is fairly similar in both the mean and standard deviation, although the maximal meta learners somewhat outperform the others in this category for all the stock types. What makes that result interesting is that, as we remember from the previous sections, the standalone algorithms outperformed the maximal meta learners regarding profit estimation. Adding to that, we can also see that the maximal meta learners have a considerably higher precision as well.

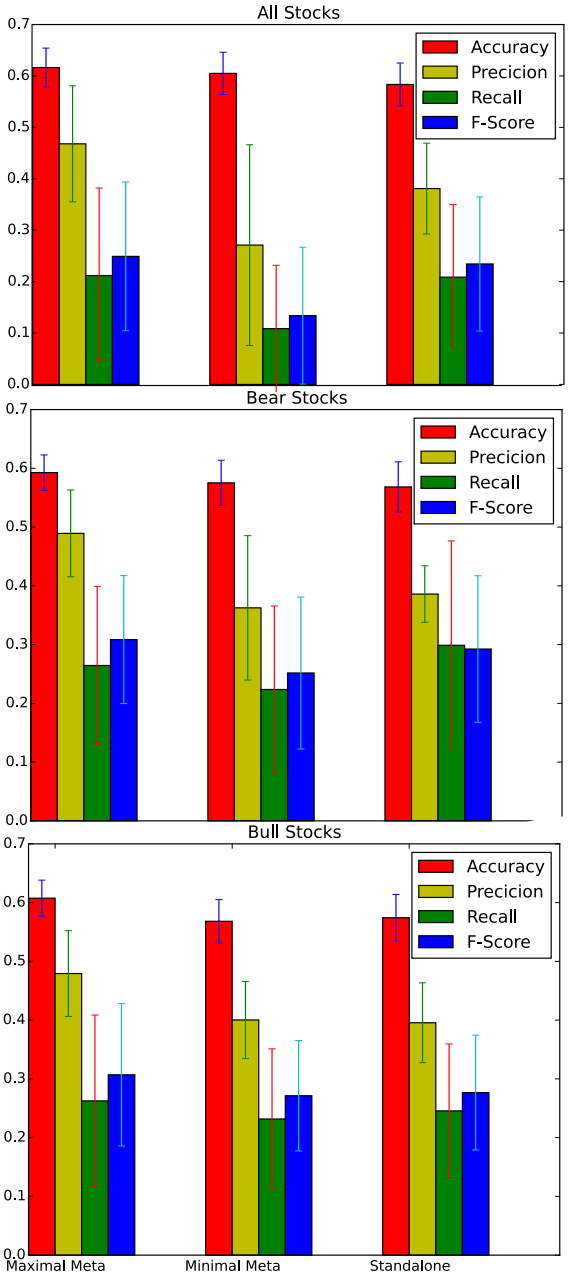
In section 3.4.1 we discussed the statistical measures, and made a seemingly intuitive assumption that a higher precision is better for Bearish periods and higher Recall is better for Bullish periods for a stock market prediction problem, and that Accuracy and F-Score were good measures for the overall time periods. However, comparing the results in 4.2 regarding the Accuracy, Precision, Recall and F-Score for all the stocks, the Bearish stocks and the Bullish Stocks, it is hard to conclude that these assumptions were correct. The standalone algorithms outperform the Maximal Meta algorithms in all of the profit measures, and it does so in a very similar manner for all the stock types. This also goes for the bearish and bullish stocks, even though the Maximal Meta Ensemble Learners have notably higher Precision.

Why the profit measure and the statistical measures are showing different top performers is hard to grasp. One of the main causes could be, as highlighted in the limitations section 3.5, that the performance measures have some quirks in that a correct classification counts as a single true positive/negative for the statistical measures, while for profit estimates, it has a series of degrees that can change the result. It may seem that even though the Maximal Meta Learners classify more correctly, the Standalone learners can make bigger profits or smaller losses on each trade. The fact that the meta learners actually have a higher Accuracy mean that these types of learners may not be any worse than the standalone learners. They may in fact be better at other problems, and simply do not perform perfectly on this exact problem. This is an indication that the Maximal Meta algorithms may have performed better with other thresholds or features.

The minimal meta ensemble learners did not perform very well

regarding the profit estimate, but hold up better for these statistical measures. They have higher Accuracy than the standalone learners for *All Stocks* and *Bear Stocks*, and have a lower or similar precision, recall and F-Score, for all the other categories.

Figure 4.2: Statistical Measures, Groups Compared



4.2.2 Learners Compared

This section shows the results that allow us to compare the performance of the different algorithms in each group of learners.

Comparing Machine Learning Algorithms

Here the results for all the machine learning algorithms are shown, in the same manner as in section 4.2.1, with *All Stocks*, *Bull Stocks* and *Bear Stocks* adjusted for OSX in 4.3 and the p-values in 4.5. These results are shown to see whether there are any differences between the standard machine learning algorithms, and this gives us an idea of how the algorithms compare to each other regardless of the input provided, before we in later sections can see the results of the machine learning algorithms in regards of only one scheme at a time. With all the machine learning algorithms it means that for, for example Binary Neural Network, the results show the performance of the Neural Network as a standalone learner, as a Minimal Meta Ensemble Learner and a Maximal Meta Ensemble Learner combined.

Six of the algorithms have a mean that is above 0, Locally Deep SVM, Decision Forrest, Decision Jungle, Averaged perceptron Pegasos-Linear SVM and the Fast Tree. Of all of them Averaged Perceptron and the Pegasos-Linear SVM have the highest means, but as seen in the p-value table they are not significantly better than several of the other algorithms. The locally deep SVM also performs well, but not well enough to prove that it is significantly better than most of the other algorithms. Logistic Regression performs by far the worst, but does so in a surprisingly steady way. This is due to the algorithm producing a very defensive purchasing scheme, meaning that it nearly never purchased a stock.

The observations to take away from the *Bull Stocks* and *Bear Stocks* plots is that the Locally Deep SVM and the Decision Forest overperform compared to the other stock on the *Bull Stocks*, and that the SVM performs better than on the *Bear Stocks* than on the *Bull Stocks*, which is certainly impressive for the *Bear Stocks*, but perhaps nearly equally unimpressive for the *Bull Stocks*. The fact that none of the algorithms have a mean that is far from zero is interesting. This result can generally mean two things; first that the learners are actually able to make the decision that the stock will fall in value and choose not to buy. This could be a sign of that the machine learning algorithms makes reasonable predictions. The other thing it could mean is that the algorithms simply choose to stay out of the market, i.e. not buy stocks very often, and therefore stay close to zero, which in many cases is not a bad choice, but if the algorithms do this often it all becomes quite meaningless.

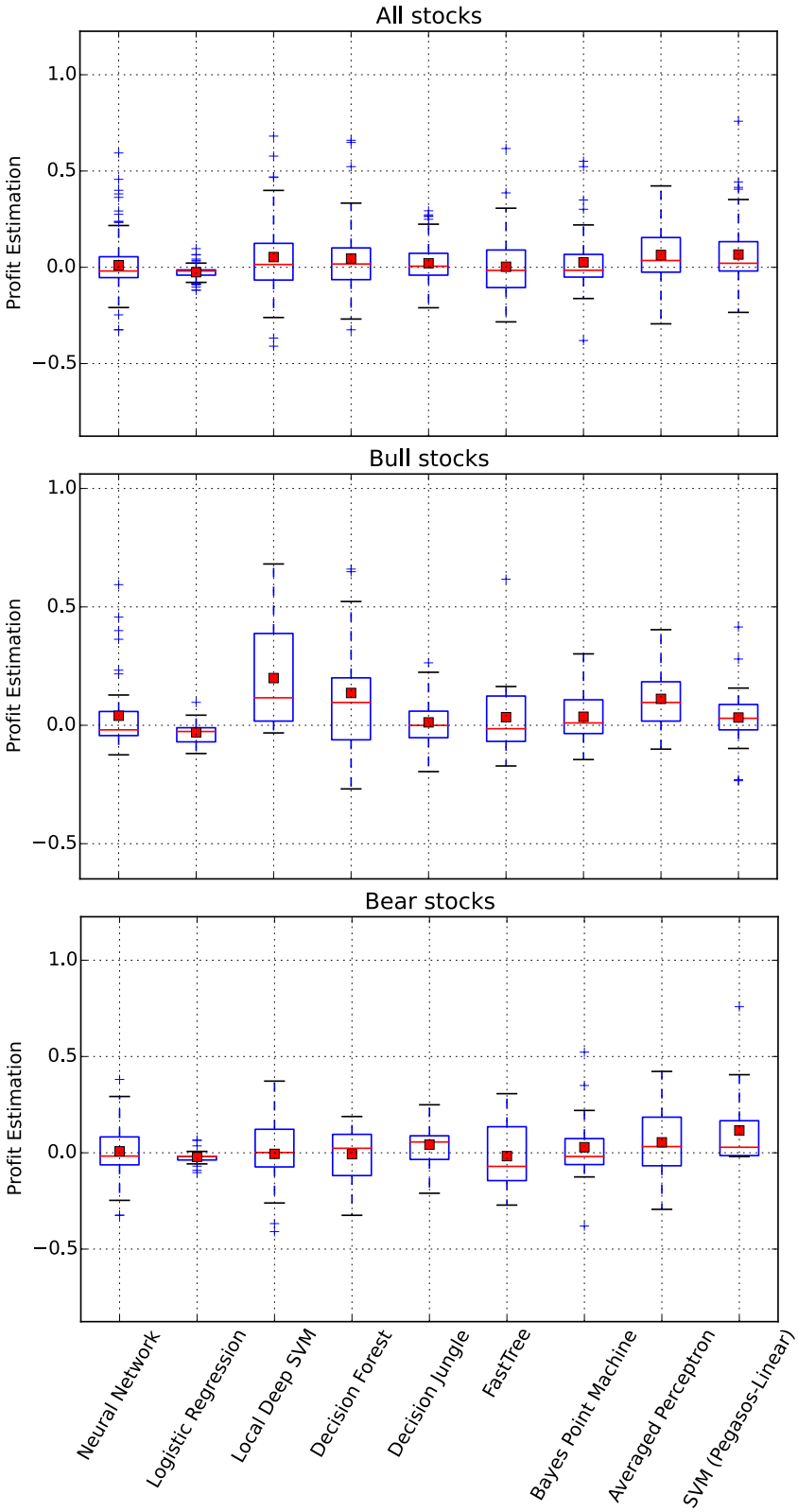
Now that we have reviewed the results from the machine learning algorithms, it is interesting to see whether the results from the machine learning types are similar to the results to the types combined. This might give an indication whether one of the algorithms is generally better at stock market prediction regardless of

inputs; whether some algorithms perform better with certain inputs or if it is perhaps simply luck that separate them. This follows in the next sections.

Table 4.5: Significance Table, All Algorithms

	Neural Network	Logistic Regression	Locally Deep SVM	Decision Forest	Decision Jungle	FastTree	Bayes Point Machine	Averaged Perceptron
Logistic Regression	N							
Locally Deep SVM	N	Y						
Decision Forest	N	Y	N					
Decision Jungle	N	Y	N	N				
FastTree	N	N	N	N	N			
Bayes Point Machine	N	Y	N	N	N	N		
Averaged Perceptron	Y	Y	N	N	Y	Y	Y	
Pegasos Linear SVM	Y	Y	N	N	Y	Y	Y	N

Figure 4.3: All Algorithms Compared



4.2.3 Standalone

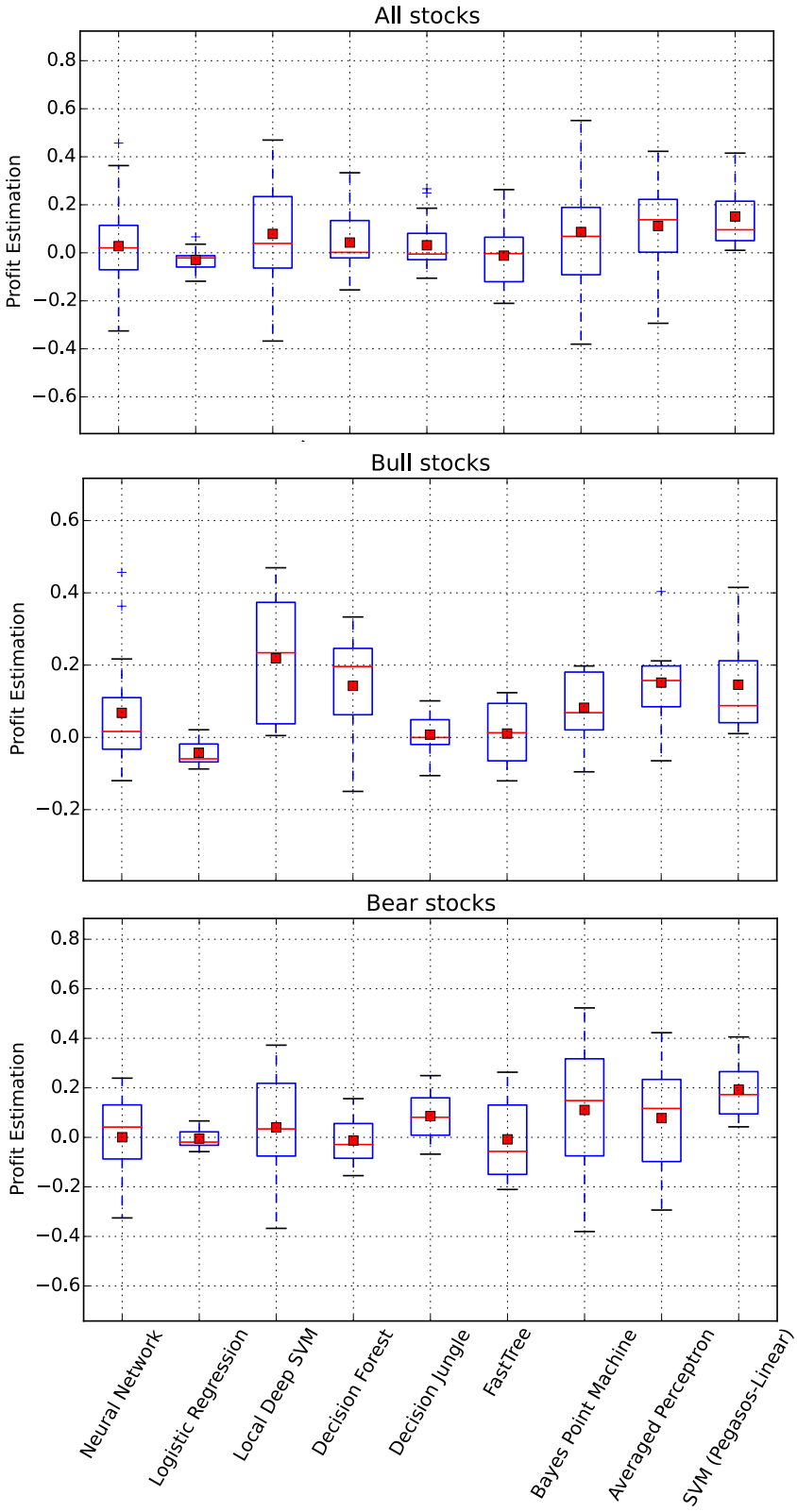
The figure 4.4 shows the standalone learners and the standalone ensemble learners compared to each other. These results are included to give the reader an idea of the performance of the different standard machine learning algorithms with the standalone algorithms. Table 4.2.3 shows the p values of significant differences for the same results as the plot. Note here that in the plot the Y axis is on a slightly different scale than in the previous plots, and that the results may seem skewed when comparing the results here with figure 4.1.

Reviewing the plot, we are quickly able to see that see that the results are not much different from what we saw in the previous section 4.2.2. The biggest difference for all stocks is that the FastTree algorithms mean has fallen below 0 and that the locally deep SVM has gained a meager profit. But other than that there is not much difference. There is in fact a bigger and more noticeable difference of what we can see with the naked eye in the P-Value table. The P-Value table shows that the results from the standalone learners may show that our two top performing algorithms, the Pegasos Linear SVM and Averaged Perceptron, can be shown to have statistically significant different result than the other algorithm except the Bayes Point Machine and each other. This is a better result than we got for all of the machine learning algorithms. And the result comes despite having a third of the sample size. This may tell us that there is less variance in the groups for the standalone learners alone than in the other types of algorithms. This is interesting because it shows in another way that there is in fact a difference between the groups of learners.

Table 4.6: Significance Table, Standalone Algorithms

	Neural Network	Logistic Regression	Locally Deep SVM	Decision Forest	Decision Jungle	FastTree	Bayes Point Machine	Averaged Perceptron
Logistic Regression	Y							
Locally Deep SVM	N	Y						
Decision Forest	N	Y	N					
Decision Jungle	N	Y	N	N				
FastTree	N	N	N	N	N			
Bayes Point Machine	N	Y	N	N	N	Y		
Averaged Perceptron	Y	Y	N	N	Y	Y	N	
Pegasos Linear SVM	Y	Y	Y	Y	Y	Y	N	N

Figure 4.4: Standalone Learners Compared



4.2.4 Meta ensemble

Maximal

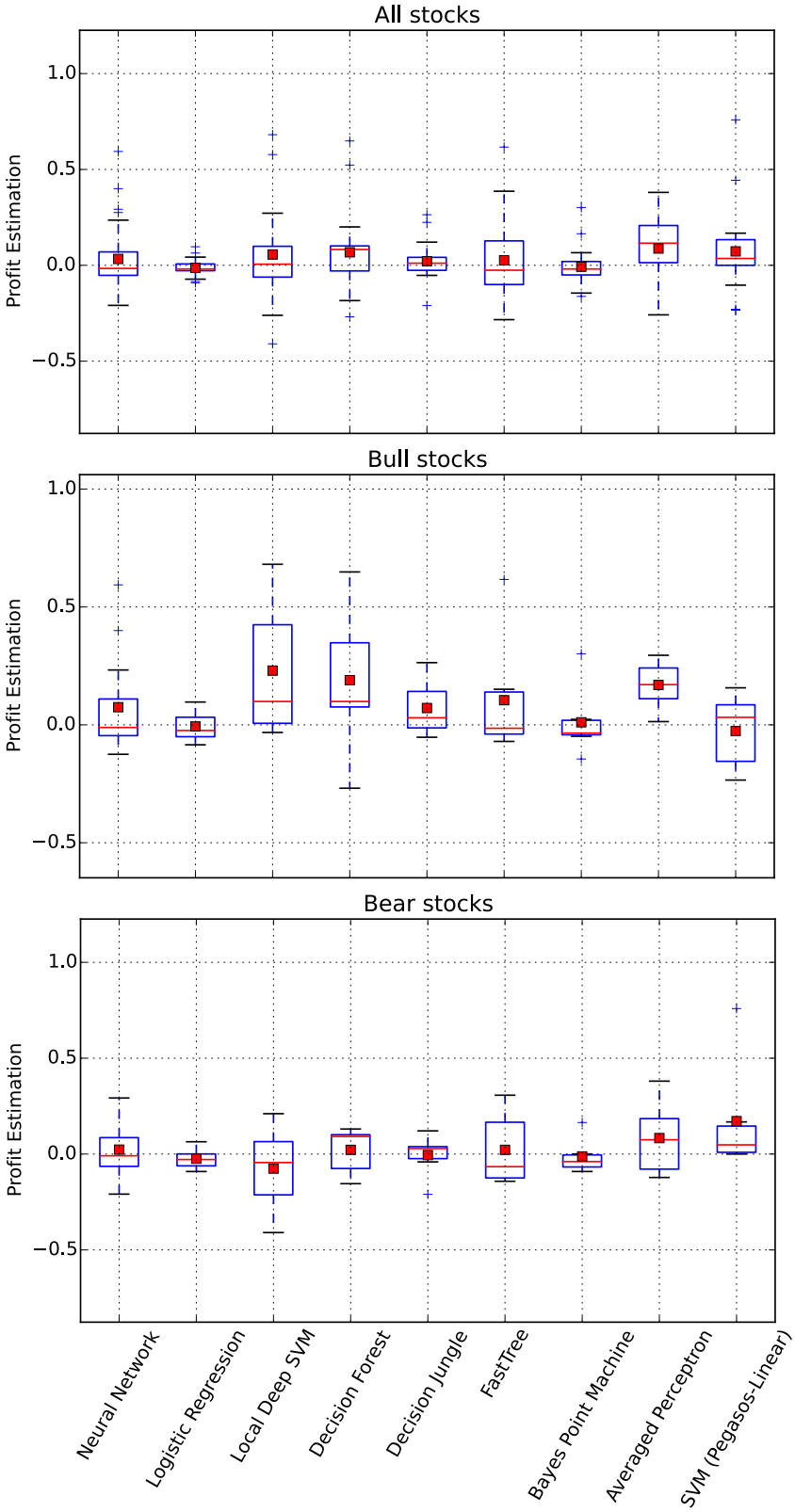
In figure 4.5 we see the results from the Minimal Meta Ensemble Learners, and their associated P-Values in table 4.2.4. These results were included to show the performance of the different machine learning algorithms compared with each other for the maximal meta scheme. Examining the *All Stocks* plot, there are again small differences between the learners, and nearly every algorithm has a mean above 0. Bayes Point Machine and Logistic Regression have a mean below 0.

The Significance table shows that we with statistical significance only may say that over the top performers, yet again the Pegasos Linear SVM and the Averaged Perceptron, are better than the lowest performers. The locally Deep SVM and the Decision Forest perform better in this scheme than in the standalone. For the *Bull Stocks* and *Bear Stocks*, a lot of the results are similar to the standalone scheme and all of the learners in general. Locally Deep SVM and Decision Forest fluctuate the most between the stock types, a clear sign of more aggressive purchasing schemes. Another interesting takeaway from these results is that the Peagsos Linear SVM performs better for the *Bear Stocks* than with the *Bull Stocks*, which is highly unusual and counter intuitive. One may speculate on the reasons for this result; and it is likely due to a defensive strategy and very accurate prediction of the *Bear Stocks*.

Table 4.7: Significance Table, Maximal Meta Ensemble Algorithms

	Neural Network	Logistic Regression	Locally Deep SVM	Decision Forest	Decision Jungle	FastTree	Bayes Point Machine	Averaged Perceptron
Logistic Regression	N							
Locally Deep SVM	N	N						
Decision Forest	N	Y	N					
Decision Jungle	N	N	N	N				
FastTree	N	N	N	N	N			
Bayes Point Machine	N	N	N	Y	N	N		
Averaged Perceptron	Y	Y	N	N	Y	N	Y	
Pegasos Linear SVM	N	Y	Y	N	N	N	Y	N

Figure 4.5: Maximal Meta Ensemble Learners



Minimal

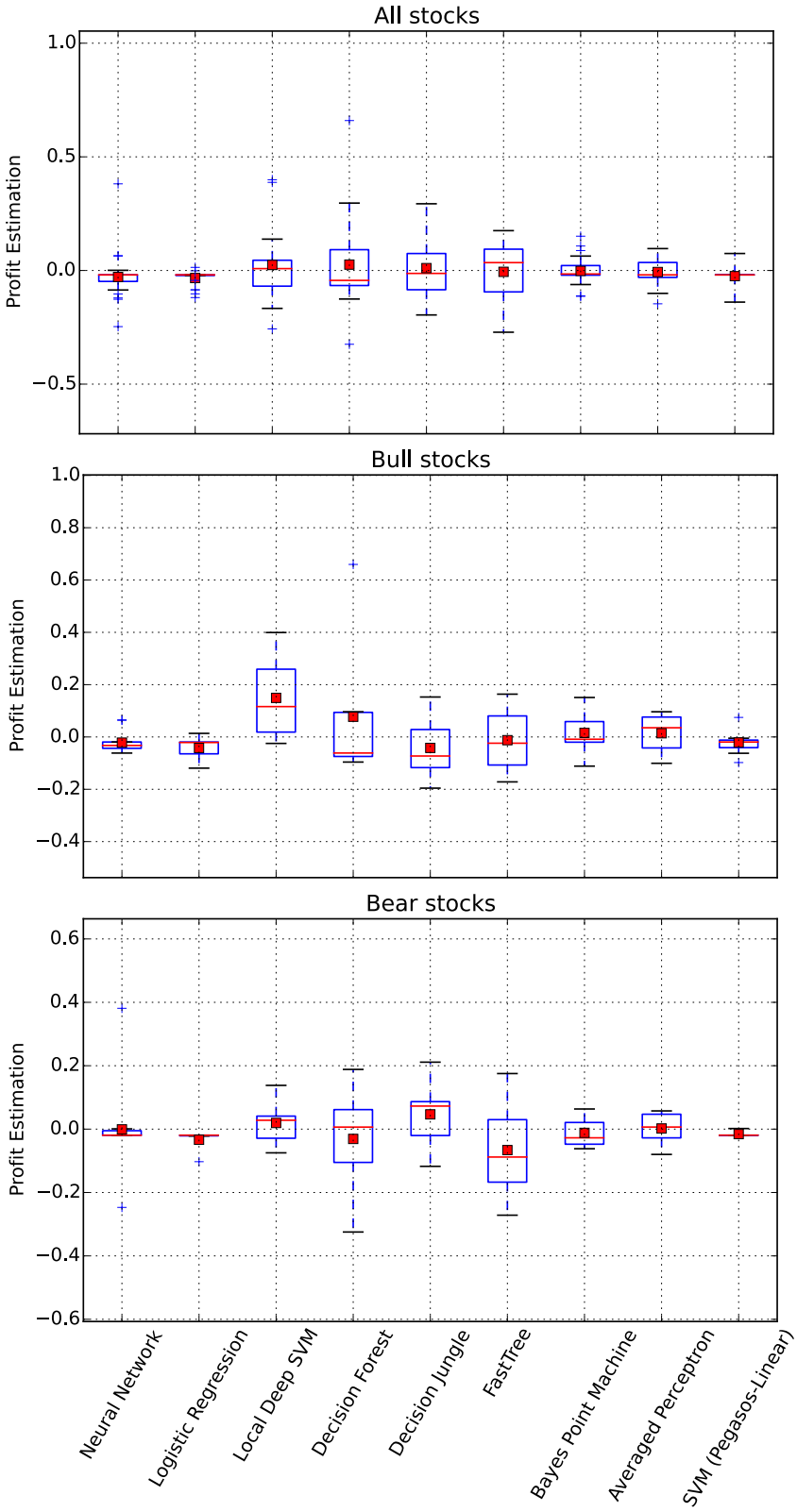
In figure 4.5 we see the results from the Maximal Meta Ensemble Learners, and their associated P-Values in table 4.2.4. These results were included to show the performance of the different machine learning algorithms compared with each other for the minimal meta scheme. As we recall from the section 3.3.2, these algorithms get a lot less input than the others, as its only input is the prediction from the standalone learners. We can recall from 4.1 that the minimal meta algorithms performed poorer than the standalone algorithms and the maximal meta ensemble learners. This is also visible in the plot, as we can see that the majority of the algorithms have a mean below 0, effectively meaning that an index fund is a better choice than using the algorithm.

What we see is small distributions, meaning that the results are similar for every stock. This is a strong indication that the predictors simply too seldom predict a stock rise. This is seemingly a bigger problem for these the minimal meta ensemble schemes than it is for the maximal and standalone. There can be a twofold reason for this: The scored probabilities given to the learning algorithms may simply not give enough information for a meaningful prediction or the scored probabilities from the other algorithms are biased closer to 0, which is likely, as there are more days that the stocks do not rise more than 0.5%. For the *Bull Stocks* and *Bear Stocks*, the same pattern occurs as we have seen for the standalone and maximal algorithms.

Table 4.8: Significance Table, Minimal Meta Ensemble Algorithms

	Neural Network	Logistic Regression	LocallY Deep SVM	Decision Forest	Decision Jungle	FastTree	Bayes Point Machine	Averaged Perceptron
Logistic Regression	N							
LocallY Deep SVM	N	N						
Decision Forest	N	N	N					
Decision Jungle	N	N	N	N				
FastTree	N	N	N	N	N			
Bayes Point Machine	Y	Y	N	N	N	N		
Averaged Perceptron	Y	N	N	N	N	N	N	
Pegasos Linear SVM	N	N	N	N	N	N	N	N

Figure 4.6: Minimal Meta Ensemble Learners Compared



4.2.5 Simple Ensemble

While some save the best for last, the poorest group of learners are to be displayed last here: The Simple Ensemble Learners. The results are shown in figure 4.7 and whether there is any significance in the results is shown in table 4.9. These results were included to provide insight into the different simple ensemble learners. The simple ensemble learners were perhaps given an unfair opportunity. While the other algorithms have been optimized by years of research and a large Microsoft team, these learners had its parameters from a few swift preliminary tests. The problem with this is best shown by the Precision scheme, which nearly never bought any stocks, and definitely should have had a lower threshold for purchasing stocks.

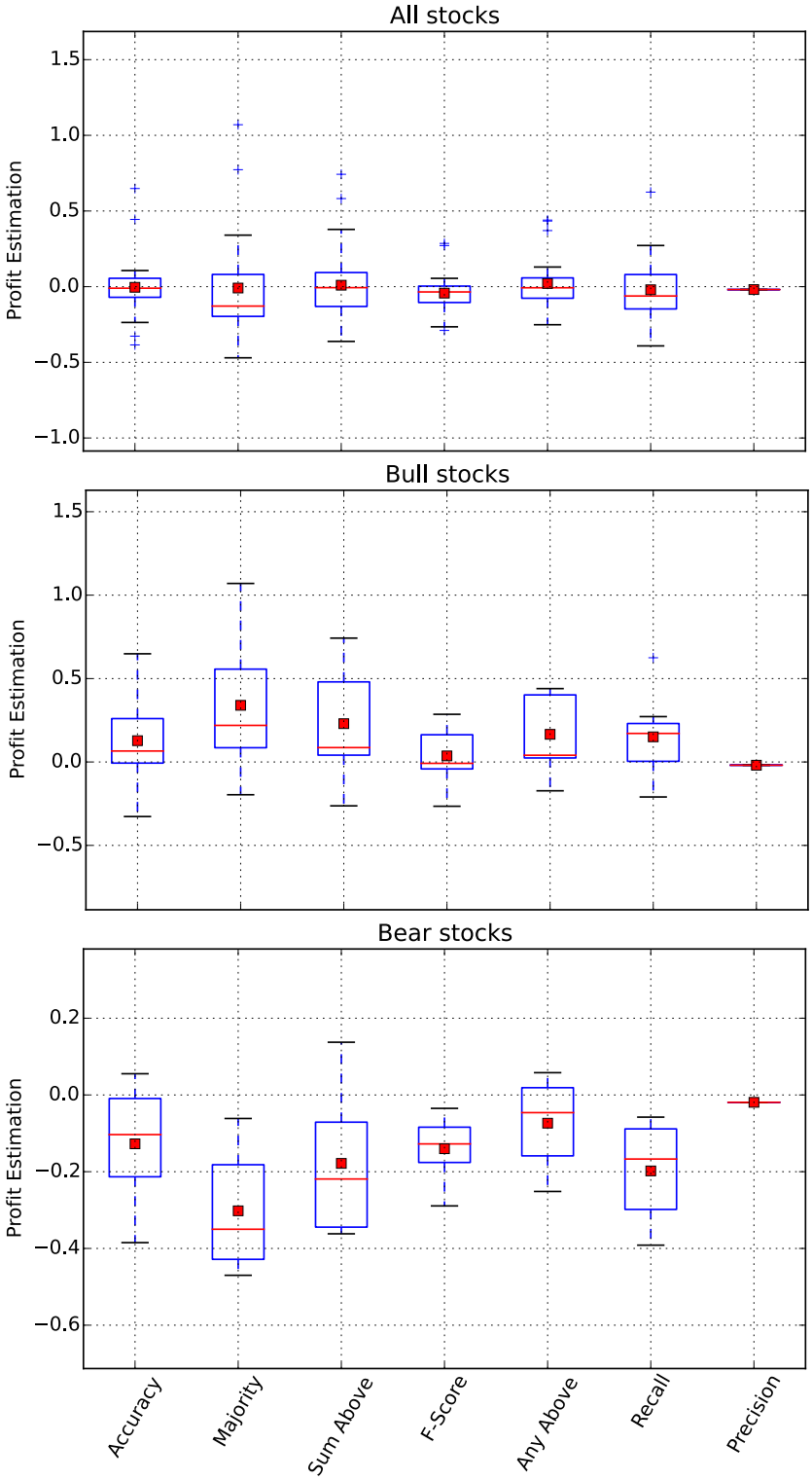
If it as a trend that these simple ensemble schemes for purchasing stocks all had to high thresholds, they would likely perform decently for Bearish stocks. However, one can observe from figure 4.1 and figure 4.7, that this is in fact not the case. As we see, the precision scheme outperforms the other schemes for the *Bear Stocks* by simply not doing anything, which hardly is impressive.

Of the different simple ensemble schemes, only Sum Above and Any Above had a mean above 0. But all of the schemes perform relatively similar, and as we can observe in table 4.9,we can with statistical significance only say that Precision is the worst scheme. We can not statistically separate the results due to the variance within the performances of the schemes on the different stocks, increasing the probability that the results are in fact derived from chance.

Table 4.9: Significance Table, Simple Ensemble Learners

	Accuracy	Majority	Sum Above	F-Score	Any Above	FastTree
Majority	N					
Sum Above	N	N				
F-Score	N	N	N			
Any Above	N	N	N	N		
Recall	N	N	N	N	N	
Precision	Y	Y	Y	Y	Y	Y

Figure 4.7: Simple Ensemble Learners



4.2.6 Top Performers

The 15 overall top performers and their average profit estimate above OSX for all the stocks are shown in table 4.10. These results were included as an attempt of a final summation of the results and to show the potential of profit making. As we can observe, the top performer is not unexpectedly the Pegasos Linear SVM from the standalone group. It yields a profit of more than 10 % more than OBX over the test period. As we can remember from previous sections, the Pegasos Linear SVM has been one of the top performers for all of the machine learning groups. And we can also remember from section 4.2.1 that the standalone algorithms performed the best, so it is anything but shocking that it is one of the top performers. The Averaged Perceptron occupies the two next positions in the table, both as a Standalone Learner and as a Maximal Meta Ensemble Learner, amplifying the belief that the Averaged Perceptron is an algorithm that is well fit for stock market predictions. We can see that all the top 10 performs are from either the Maximal Meta Ensemble Learners or the Standalone Learners, giving us strong reason to believe that these groups are the best for the problem at hand.

Table 4.10: Top Performers

Profit	Algorithm	Group
0.151	SVM (Pegasos-Linear)	Standalone
0.111	Averaged Perceptron	Standalone
0.087	Averaged Perceptron	Maximal Meta Ensemble
0.086	Bayes Point Machine	Standalone
0.078	Local Deep SVM (LDSVM)	Standalone
0.072	SVM (Pegasos-Linear)	Maximal Meta Ensemble
0.067	Decision Forest	Maximal Meta Ensemble
0.056	Local Deep SVM (LDSVM)	Maximal Meta Ensemble
0.042	Decision Forest	Standalone
0.033	Neural Network	Maximal Meta Ensemble
0.031	Decision Jungle	Standalone
0.027	Neural Network	Standalone
0.026	FastTree	Maximal Meta Ensemble
0.025	Decision Forest	Minimal Meta Ensemble
0.024	Local Deep SVM (LDSVM)	Minimal Meta Ensemble

4.3 Analysis

4.3.1 Can you make a profit?

As discussed in the limitations section 3.5, there are several weaknesses with the profit estimate posed in this thesis which makes a definite conclusion to whether one could make a profit with a ma-

chine learning algorithm difficult to postulate. However, the results show that a handful of the algorithms do indeed yield a profit greater than the market, which has been the baseline for the results. When comparing the groups against OSX and Random purchasing strategies in section 4.2.1, we observed that Maximal Meta Ensemble Learners, Standalone and Standalone Ensemble learners outperformed the market with statistical significance. The significance gave reason to believe that there is a possibility of making a profit.

Trading Schemes

To reduce risk, it is common to disperse the money into different stocks and sometimes also different purchasing schemes. This would be wise to implement also for a machine learning implementation of stock trading. As we can see from all of the plots in this thesis, sometimes the algorithm performs very poorly, and by simply trading one stock with one algorithm, there is a big risk of losing most of your money. There is of course also a chance of making a lot of money, but gambling strategies like this are not at all recommended and will likely not hold up over time. One of the more sensible strategies for making a profit would be to spread your money equally to each stock and stick to a strategy. However, the real life problem of fees imperfects markets. Not being able to buy and sell at the wanted price quickly enough, makes the strategy of dividing your money between all stocks available more of an implausible solution.

Top Performers

When observing the overall top performers, we can observe that the Standalone Pegasos Linear SVM beats the market by a staggering 15%, which might be enough of a profit to outrun the fees. Note that fees are normally set at a fixed price, therefore if we were to trade with large amounts of money, each trade would have a relatively lower price than if we trade with small amounts of money. The fees also vary from broker to broker. On the other hand, it is harder to buy and sell large amounts of stocks than small, and this may cause some loss of profit. But the 15% profit over the market yielded by the top performer might very well have been profitable in actual trading. While we can with statistical significance claim that Standalone Pegasos Linear SVM outperforms the market from the first significance table 4.4, we cannot say that it is better than the Standalone Averaged Perceptron or the Bayes Point Machine, and the extra profit might be derived from chance. On the other hand, we know from the same significance table that it is a better performer than the standalone Locally deep SVM, and therefore we can say that it is likely that Standalone Pegasos Linear SVM would yield a profit larger than the locally deep

SVM at 7.8%.

The test period

Whether over 7.8% from the profit estimate is enough to make a profit in the real world is as we know difficult to say, but let us for the sake of argument assume that it would. There is also another issue regarding an adequate answer to whether we could expect a profit. The test period is as discussed in section 3.2.3 a generally positive time period. As we know from history the market has its ups and downs, and it is likely that this test period does not adequately represent any other time period. The Bull and Bear results were included to give better insight into how the predictors would work in crashes and rallies. We have observed in 4.1 and discussed in the talking data section 3.2.3, how they perform surprisingly well for Bear stocks and surprisingly poor for Bull stocks. These results give us reason to believe that the learners, and especially the top performers yield reasonable results even in crash-like scenarios and therefore rarely make a big loss. But we should not forget that there is no justification for choosing this time period, and as we know, the world economy is in constant change and not two periods of time are equal. We should therefore be careful claiming that one could have made a similar profit to the ones in these results in another time period. We should however not forget that the standalone Pegasos Linear SVM and the other top performers did not simply make a profit, they outperformed the market substantially. The fact that they also handled the bear stock without losing money is a strong argument that the most fitting machine learning algorithms would not perform badly even in downward markets.

Lady Luck

If 37 people spun the roulette wheel 22 times each, it is likely that some of them would make a profit. If 37 machine learning algorithms attempts to predict 22 stocks, it is also likely that some of them would make a profit, even if the stock market was entirely random. The fact that some out of many are bound to perform well, is another sign of that we should interpret any results regarding any prediction of any system with noise or general randomness with a pinch of salt. One could talk about probabilities, U test and distributions, but there will always be that insecurity of chance. The results in this thesis are obviously affected by this, however one could argue that the fact that this was not a single buy/sell transaction, but rather a series stretching over nearly a year and hundreds of buy/sell decisions, that it is less prone to luck. And by observing the results with the Whitney Man U test we observed that there was a difference between the profits of all the random purchasing strategies and the best types. The fact that

we can say with 99.95% probability that the top performers are inherently different from the random strategies, gives us an argument that the results are very likely valid, and the chances of chance is playing a trick on us are low. But we should always keep in mind that the Whitney Mann U Test shows that the results are different, but that different is not necessarily better. A scheme that purchased a stock for the first 50% of the days and remained inactive the last 50% of the days would be different from a scheme that bought a random 50% of the days, and their profit estimates would likely be very different. On the other hand, we should not forget that all the experiments are done with 22 stocks, and that this obviously decreases the chance of luck being the main influencer of the result.

Theory bound models

Most of the researchers studying finance and the predictability of the stock market would criticize any machine learning model for drawing conclusions not being based upon a theory. In finance and most other social sciences the normal procedure is to first make a hypothesis upon why, for example, variable a effects variable b, and then test the hypothesis using some statistical test and subsequently draw a conclusion upon whether or not variable a effects variable b. Machine Learning turns this around, as we skip the theory part. As an example one of the machine learning algorithms tested here might through the data discover that when variable a increases by 1% variable b increases by 2%, and use this in its predictions. Finding such correlations may seem great, but it may fool us into thinking that there is a direct cause and effect between variable a and b. It may very well be a third variable not included in the data set that affects variable a, and a fourth variable effecting variable b. Since we obviously will never be able to include all possible data that may affect variable b, it leaves us in an awkward position. The machine learning algorithm has no way of testing that there is cause and effect between a and b, and since it is not bound to any theory, there is no way of justifying the correlation. Because stocks can be affected by near infinite variables, it is almost impossible to conclude without any doubt whether one could make a profit over time or not. On the other hand, one could argue that there is no need for the machine learning algorithms to be able to explain a theory that it bases its predictions upon. For a stock prediction problem, it may be enough to predict correctly 51%, or at whatever percentage it starts yielding a profit, of the time for it to be useful. This is illustrated by the top performers that yielded a more than satisfactory profit on the test period.

Optimization

If one were to predict stock with machine learning algorithms with actual money, the experiments performed here would likely just be a preliminary test to help aid the choice of one or a few algorithms that would be optimized further. There is a great deal of research regarding machine learning on real life problems where great improvements have been made by manually adjusting the parameters of the algorithms. One could take it even further using optimization schemes such as genetic optimization. The machine learning types and algorithms tested here have not been given the advantage of such an enhancement, which certainly can be used as an argument that it is possible for the algorithms to improve their performance. As is pointed out in the limitations 3.5, the experiments here were focused on a broader research trying many algorithms rather than optimization of a single one.

Variable selection

An issue that occurs for any model based prediction is, as discussed in the feature selection section 2.2.11, the input data or features. It is well known in machine learning that feature optimization can improve the performance of a machine learning algorithm [38]. A problem like stock prediction can be, as discussed in the forces that moves stocks section 2.1.3, particularly susceptible for omitted variable bias. Therefore, feature selection could very well have improved the results presented in this thesis. This thesis used quite a high number of features, which may lead to problems of finding correlations without causation. One can actually observe from the results in this thesis that more features is not necessarily better, as the Maximal Ensemble Learners get the same inputs as the standalone learners and some extra inputs, but fail to outperform them. On the other hand, by excluding features, we risk not being able to predict certain things that are being caused by external variables. No matter if one should have included more variables or fewer, it is highly unlikely that the features chosen for this thesis is the optimal set of features. Therefore, it is likely that one could get improved results with another set of features, an argument that it is possible for a machine learning algorithm to yield even greater profits than what is shown here.

Risk and Reward

As we observed in the results there is a surprisingly small difference between the result on the bull and the bear stocks. Investigating the random distributions purchasing strategies made it possible for us with an even greater conviction to say that the algorithms traded quite defensively. This theory of defensive predictions can be shown by observing the low recall in the plot in figure 4.2. The plot shows

that recall is lower than precision in every case, and as was discussed in 3.4.1 this means that the algorithms more often than not predicted that the stock would not rise the next day. One can argue that this is a positive property of the learners as a trading scheme, as it will lower the risk, since fewer purchases provide a lower risk. That they are defensive may be used as an argument that the picking a threshold of 0.05% for the target of the machine learning algorithms was a good decision. The choice was made after it yielded the greatest profits in the preliminary tests.

Regardless of how much risk one would want to take, and how many purchases that are reasonable, it is clear that the algorithms did not predict perfectly. This is easily observable in the plot in figure 4.2, where accuracy, precision, recall and F-Score are quite low for all of the learners. The unfulfilled potential is further illustrated by the random schemes outperforming all of the other algorithms for the bear stocks. It presents an argument that the algorithms predicted too defensively, and that they therefore made a much smaller profit than possible. These additional profits could have made it possible for us to say with more certainty whether one actually could have made a profit using machine learning for the time period. The balancing act of finding the perfect ratio between risk and reward is strenuous. It is doubtful that the thresholds set for the experiments were optimal, and quite certain that improvements can be made to both increase the profit or decrease the risk.
Another choice influencing both the risk and the performance was the choice of a daily resolution. The choice was taken for convenience and because it might have shown interesting results. It may however not be the optimal resolution. As we can see from the results, the learners, with the exception of a few, have not yielded particularly large profits compared to the risk taken. With another resolution the learners may have performed better. Many of the automated trading schemes used today purchase stocks with a much higher frequency than daily purchases. On the other hand, longer term predictions may also have yielded interesting results with lower risk and perhaps also larger profits.

Binary Classification

As discussed, changing the threshold of 0.05% may offer greater results or less risk. This may lead us into the question of whether binary classification is the right choice for attempting to make a profit on the stock market. On one hand one could argue that in the end anything regarding profit in the stock market boils down to one question: Should I own this stock? On the other hand, one may argue that the binary result does not provide enough information to decide whether on should purchase a stock or not. Multi-class classification could provide more information by utilizing classes richer in information,

classes that indicated how much it predicted the stock to rise or fall. This would also allow for trading scheme where one could weigh the amount of money spent on purchasing each stock. This is to some extent possible with two-class classification; one could achieve similar performances by changing the threshold of the outputs, and the trading schemes could have been made by using the scored probabilities. This was to some extent attempted by using the so called simple ensemble learners, but they only looked at one stock at a time and therefore fall short of any scheme looking at all the stocks at the same time.

There are however scenarios where two class classification falls short, namely with shorting and other more complicated ways of trading stocks. Shorting is essentially betting that a stock will fall in value, and makes it possible to make a profit by predicting a stock's decreasing value. Both Multi-Class classification and Regression would have allowed for predictions that would have allowed for shorting. Regression seems perfect for stock trading schemes that use both buying and shorting stocks, as it shows the real value of what it predicts. However, if you say: I will purchase a stock if the regression predicts a result above zero, and short the stock if it predicts below zero, the problem is actually a two class problem, and would yield the same results as the tested algorithms only with a threshold at 0% change for the targets. One could set other thresholds on when to buy, when to sell and when to hold, but the essence of making the problem into a regression problem is that at one point we would have to set thresholds of when to purchase, when to short and when to hold, which boils the problem down to a classification problem. Multi-class classification would have allowed for these different categories, and may therefore be the optimal way of predicting stock. However, by shorting and using more classes, we increase the risk and maybe also the error rate, as it makes the problem more complex. Errors can also be more expensive using a profit estimate, as shorting a rising stock results in a loss.

Similar research

The results presented regarding whether one could make a profit in the stock market seem to coincide with the published papers on the topic. There is however a divide in the research, usually depending on the background of the paper authors. Papers where the authors view the stock market as a problem that can be solved using smart algorithms and historical data are usually from an Artificial Intelligence and Informatics background. These papers often show that there is great promise in applying machine learning for predicting stocks [47][20], and many show results of models that can yield profits. On the other hand, papers where the authors are from an economical background seem to emphasize to a much higher degree the problems with predicting stocks using historical data [14]

. These economists have shown that it is not necessarily possible to say that a test period is representative, and as previously covered, they have even shown that random strategies may perform as well as any other scheme given a long enough test period. There are some researchers that from a Machine Learning background address the problems of ever changing data, such as the stock market, on how to draw knowledge from the decisions boundaries, and problems with any time series data when using machine learning and data mining [77]. The results presented in this chapter also show that there are possibilities, and it seems that one could in fact make a profit using machine learning on the stock market, but there are however some basic underlying problems with prediction stocks one cannot escape.

4.3.2 Comparing Schemes and algorithms

Predicting with predictions

One of the sub goals of this thesis was to implement and test machine learning algorithms that used the predictions of other machine learning algorithms as part of its input. These schemes, called Minimal and Maximal Meta Ensemble and Simple Learners, and their results have been shown. The results may not seem too impressive at first glance, but when one investigates and theorizes around their seemingly low performance, we might see why the methods should not be dropped as a research topic, and that further research is needed to conclusively draw conclusions.

Let us first compare the results of the types of algorithms in figure 4.1 the Standalone learners and Maximal Meta Ensemble Learners, since they were the top performers. We could through the Whitney Mann U test separate the results from these types, and show that they yielded better results than the other types of algorithms. Between these two top performing groups there were however not a statistical difference in the results. We can therefore say that the experiments produced no evidence supporting the statement that the machine learning algorithms got an improved performance by having predictions of other algorithms as an input. There are however also no statistical significant evidence showing that the maximal meta ensemble learners performed any poorer. Therefore, we cannot exclude the chance of the standalone schemes running into a stretch of luck. We can see in the top performer section 4.2.6, that the standalone algorithms in general perform better than their Maximal Meta Ensemble counterparts, but there are a few exceptions. The Decision Forest, FastTree and Neural Network all performed better for the Maximal Meta Ensemble Learner than the standalone. These results might give reason to believe that some algorithms could benefit more from the additional input than others. It is however challenging to grasp why some algorithms perform better with the different

schemes.

An interesting observation made in section 4.2.1 where we investigated the results of the different types using statistical measures, is that even though the Maximal Meta Ensemble Learners outperformed the standalone algorithms in all of the statistical measures, it did not yield higher profits. This not only shows that the limitations of the profit estimates as well as the limitations of the statistical measures, but it also leaves more room for the discussion on whether the Maximal Meta Ensemble Learners may in fact outperform the standalone algorithms on other problems or sets of data, as they did predict correctly more than the standalone learners. And we must also not forget that the Standalone algorithms have gained from being optimized in a much higher degree than the Maximal Meta Learners, as they come default with carefully set parameters. One of the issues that may cause the Maximal Meta Learners not to perform better than its standalone counterpart is that there is a high number of features, in fact over 900. This may cause the predictions from the other algorithms to drown, showing yet another reason that these types of algorithms need to be further researched before one can say with certainty whether they produce better results than standalone algorithms or not.

When comparing the statistical measures of the Minimal Meta Ensemble learner with the others, we can observe that the results are not as poor as the profit estimate would suggest. As an example the minimal meta has higher accuracy than the standalone learners. When looking deeper into the reason why the Minimal Meta Learners performed as poorly as they did in profit estimate, we see that the recall of the Minimal Meta Learners is quite a bit lower than for the other strategies. This shows that the algorithms predicted that the stock would not rise the next day almost every day, which actually could be considered a decent strategy, since it is true for more days than not. These findings show that the Minimal Meta Ensemble Learners may have benefited more than the other types by a different threshold that created a more equal distribution between the true/false. On the other hand, the Minimal Meta Ensemble Scheme might not get enough information to make a good prediction the way they have been tested in this thesis. It is highly likely that its input from other machine learning algorithms predictions are too similar to each other, and that they therefore fail to make adequately good predictions. The schemes could be further researched by drawing inspiration from other Ensemble Learning schemes differentiating the input to the machine learning algorithms. The Maximal Meta Ensemble Learners might also benefit from changes that lead to a more differentiated input.

The so-called Simple Ensemble Learners consisted of 7 different simple schemes of using the output from other machine learning al-

gorithms to make predictions. The inspirations of the schemes came from other Ensemble Learners, the Averaged Perceptron and commonly used Majority Voting. The results that these schemes yielded were however less than impressive, and did in general perform a lot poorer than its counterparts. When we investigate into why they performed so poorly, we can see that all of the schemes performed poorly, and that there was no particularly bright star among them. They did in general perform below average, and we can say with the support of the Whitney Mann U test, that the results were poorer than the other types. It is unlikely that the schemes Majority and Sum Above are simply bad schemes that will never work, as similar implementations exist in for example Averaged Perceptron and Bayes Point Machine as we can recall from section 2.3.2. What is more likely is that these were optimized very poorly, and that different thresholds should have been set. As we discussed with the minimal meta ensemble learner, these learners may also gain from having a more differentiated input. The other schemes using the statistical measures F-Score, Precision, Accuracy and Recall did not manage to convince anyone with their performance, but yet again this could be due to bad parameters and thresholds.

The Algorithms

One of the research questions presented in this thesis whether or not the algorithms would perform differently. Given the random nature of the stock market, a different performance of the algorithms was not given. We have however shown that there is a significant difference between some of the algorithms tested in this thesis. Even though we once again should be careful drawing explicit conclusions, because the test set might not be representative for any other test period, we can say with some conviction that there was a difference in performance.

It is also interesting to investigate the performance of the different algorithms to find out which ones are the most suited for stock prediction and why. The first thing we observed in section 4.2.2 was that there were surprisingly small differences between the performance of the algorithms, and even though the Whitney Mann U Test showed a significant difference between some of the algorithms, the majority were in fact without significant difference. The reason for the small differences might very well be the random nature of the stock market, and perhaps we cannot conclude that all the algorithms perform differently, and that only some of them might have different results.

When comparing the algorithms to each other we noticed that there were meager differences between the top performers. And many of the differences where not significant, which makes an analysis of

these results difficult. An interesting observation is that when we compare *Azure Machine Learning*'s [25] own recommendation onto when to use which algorithms, this does not necessarily coincide with the result yielded in the experiments. Whereas Azure ML recommends the Neural Network as a perfect algorithm for large complex problems where training time is not an issue, such as the one we got at hand, in section 4.2.2 we did observe that the Neural Network was one of the poorest performers. On the other hand, the kind of simplified neural network, namely the Averaged Perceptron was one of the top performers. Why this is the case is hard to grasp, but one might speculate that one of the reasons is that there is so much noise and random movements that the complex structure and decision boundaries of the neural network underperformed.

The Pegasos Linear SVM also seems like a good fit for the problem, as it is recommended for problems with more than 100 features that are linear, and should therefore also be a good fit for our problem. As we have observed it was one of the top performers. The Averaged Perceptron was also recommended for linear problems, which does not coincide with an initial idea of the stock market as a problem with a least some non-linear dependencies. These results are highly interesting as they indicate that the stock market is linear, or at least that linear models will outperform the non-linear models, which is somewhat unexpected. We can however see that this coincides well with previous research on the topic, as there are several studies showing that SVMs are top performers as stock market predictors [20], and it is therefore not that weird that some of the other top performers are linear models.

Chapter 5

Conclusion

This Chapter attempts to draw some conclusion from the result and analysis. Section 5.1 tries to find some meaningful answers to the research questions. Section 5.2 discusses future work.

5.1 Research Questions

This study aimed to determine whether or not it is possible to make a profitable stock trading scheme using machine learning, to compare different machine learning algorithms and their performance, investigate whether a machine learning algorithm improves when given predictions from other algorithms as features and discuss whether it is possible at all to predict the stock market. To go into further detail let us revisit the research questions posed in the introduction of the thesis. The main research question (1.1)

(1.1) Is it possible to make a profit on the Oslo Stock Exchange using machine learning?

As we can recall from the predictability section 2.1.4 predicting stocks is hard, it is very hard. This notion was further highlighted in the analysis section 4.3. There are numbers of well cited papers insisting that the stock market is governed by *random walk* [57]. Some say it is impossible to make any meaningful prediction of the stocks, and that the best strategy of investing is buying stocks at random [11]. A few researchers state that although price changes may not be strictly random, the interdependence is so slight that it is impossible to make a profit [33]. Anyway it is a question that is challenging to answer, and drawing a definite conclusion seems almost impossible. There are simply are too many unknown factors that may influence the stock market.
On one hand, the results yielded show that there are machine learning algorithms that are profitable in the test period, and can be optimized to a much greater extent, which likely will increase potential profit yielded. This shows that there is promise in predicting stocks with machine learning. There are, however, some underlying problems with predicting the stock market and knowing whether the test period is representative for any future time period. What we may conclude is that it is likely that one could apply some of the presented machine learning algorithms to generate a profit if we assume that the test period is representative for other time periods and that we were trading in a near perfect market with low trading fees. Unfortunately, the experiments conducted are not extensive enough to make a final decision on whether or not it is possible to make a profit.

One of the sub-goals of this study was to compare the machine learning algorithms to each other, and to investigate whether ensemble learning schemes could outperform other machine learning

algorithms.
(1.2) Do the performance of machine learning algorithms predicting stocks vary?

As noted when the question first was posed, it may seem obvious that different machine learning algorithms perform differently, however, if the stock market and individual stocks follow a random walk as many claim, it may not be the case. And as we observed in the top performers section 4.2.6, there were some algorithms from some schemes that outperformed the others. But behind the top performers there were relatively small differences between the profit estimates. And if we scrutinize our results even further, we can, as we recall from the previous section, see that there are often not significant differences between many of the presented results. One may therefore easily think that much of the profit may be a result of chance and coming from the limitations of the profit estimate. There are however some patterns that are observable, primarily the fact that the standalone and maximal meta algorithms occupy nearly all the top 15 places, and that a few of the algorithms have occupied some of the top spots from both the standalone scheme and maximal meta scheme. This is an important finding, as it may be used as an argument that it is possible to predict the stock market. This is because one can argue that a market that can be predicted could not be entirely governed by random walk in the test period.

(1.3) Will Machine Learning Algorithm perform better when other machine learning algorithms predictions are included as a feature?

The experiments did not provide any statistical significant evidence that the performance of the algorithms improved when getting the extra features of other algorithms predictions, nor did it show that any ensemble learning scheme outperformed the other algorithms. The initial idea of the ensemble learners being better at handling problems with overfitting have not been proven true; this may be because the cross-validation making overfitting less of an issue. It seems that we may draw the conclusion that these ensemble schemes will not improve their performance by getting the extra features, however, more extensive test should be performed before making a final conclusion.

(1.4) Is Binary Prediction suitable for a stock market problem?

The results presented give reason to believe, or at least imply, that binary prediction is suitable for making a profit predicting the stock market, as the results show a possible profit. One can also say that it is suitable because every profit making stock scheme boils down to the question; should I own this stock or not. On the other hand, the two-class predictor makes exploiting other ways of trading stocks such as

shorting difficult, and may therefore not provide as much profit as possible.

The need for more extensive research appears obvious when attempting to answer all of the research question. There is simply not enough data and tests to conclude with absolute certainty one way or the other. What tests, which experiments and what data will be further discussed in section 5.2.

5.2 Future work

5.2.1 Feature selection

One of the more apparent problems with stock market prediction is attempting to find an optimal set of features. As noted in section 2.1.3 there is almost an infinite number of possible factors that may influence the price of a stock, but by including too many features one can run into other problems such as the curse of dimensionality and finding correlations without causation. Even though the preliminary test found that five days of data yielded better results than one and three days of data, there may be a great deal of room for improvement. Finding an optimal set of features may be a way of improving the performance of the machine learning algorithms. There are several methods for automatic feature selection which may be tested, like the briefly tested Z-Score selection; other feature selection schemes should also be tested. And one could use deeper domain knowledge to select more fitting features, since it is, as noted, highly unlikely that the features selected in this thesis were optimal.

5.2.2 Regression and Multi-Class Classification

Even though we saw that two-class classification is fitting for the stock prediction problem, regression and multi-class classification would give opportunities for shorting stocks and therefore yield profit on sinking stock prices. These more complex trading opportunities are worth looking into, as they may open up to ways of weighing the amount of money put into each purchase and short, which again may lead to an increased profit or decreased risk, depending on how one chooses to set it up.

5.2.3 Parameter Optimization

Every one of the algorithms tested have a wide set of parameters that can be optimized. While the standalone learners have the default set up from azure machine learning, it is likely that they are somewhat modified to a general problem, but it is unlikely that this is perfect for the experiments performed or for stock prediction in general. This is even more the case with the Meta Ensemble Learners and simple, as

they have even more parameters to be set. Perhaps some of the simple ensemble learners would have performed much better simply by changing the threshold. Since machine learning algorithms so often get a big improvement in performance by optimizing parameters, this would be an obvious and smart move for further research, since an increase in performance may imply that one can make a profit in the stock market using machine learning.

5.2.4 Other Problems

For the sub problem of testing and comparing the different machine learning algorithms against each other, it is not enough to use a single data set to determine whether there is a difference. The algorithms should also be tested on completely different data-sets as the performance might vary a great deal. This would also open up for applying different performance measures that may better show the difference between the algorithms.

5.2.5 Time Frames and Markets

It is known that many of the automated trading schemes operating today use quite different time frames and resolutions for their predictions. High frequency trading is getting increasingly popular and might also be suited for machine learning. Also long term predictions may be profitable and may certainly decrease risk, and should therefore also be tested, as it may give a more concise answer to whether it is possible to yield profits over extended time periods on the stock market. Other stock markets can also provide valuable insight into the research questions. Oslo Stock Exchange is a relatively small stock exchange and may therefore be more imperfect, meaning that due to few investors, there is greater slippage in the market and it may therefore be harder to predict. Other stock markets may very well be more applicable for machine learning predictions and should therefore be tested. Applying the machine learning algorithms on different stock markets over other time frames is needed in order to decide with certainty whether it is possible to predict the stock market using machine learning.

Bibliography

[1] Nesreen K. Ahmed et al. *An Empirical Comparison of Machine Learning Models for Time Series Forecasting*. 2010.

[2] Richard A Ajayi and Mbodja Mougou. 'On the dynamic relation between stock prices and exchange rates'. In: *Journal of Financial Research* 19.2 (1996), pp. 193–207.

[3] James J Angel and Douglas McCabe. 'Fairness in financial markets: The case of high frequency trading'. In: *Journal of Business Ethics* 112.4 (2013), pp. 585–595.

[4] Nicholas Apergis and Stephen M Miller. 'Do structural oil-market shocks affect stock prices?' In: *Energy Economics* 31.4 (2009), pp. 569–575.

[5] George S Atsalakis and Kimon P Valavanis. 'Surveying stock market forecasting techniques–Part II: Soft computing methods'. In: *Expert Systems with Applications* 36.3 (2009), pp. 5932–5941.

[6] Edgar Ortega Barrales. 'Lessons from the Flash Crash for the Regulation of High-Frequency Traders'. In: *Fordham J. Corp. & Fin. L.* 17 (2012), p. 1195.

[7] Richard Bellman. *A Markovian decision process*. Tech. rep. DTIC Document, 1957.

[8] Yoshua Bengio and Yves Grandvalet. 'No unbiased estimator of the variance of k-fold cross-validation'. In: *The Journal of Machine Learning Research* 5 (2004), pp. 1089–1105.

[9] Evangelos Benos and Satchit Sagade. 'High-frequency trading behaviour and its impact on market quality: evidence from the UK equity market'. In: *Bank of England. Quarterly Bulletin* 52.4 (2012), p. 370.

[10] Bruno Biais, Thierry Foucault et al. 'HFT and market quality'. In: *Bankers, Markets & Investors* 128 (2014), pp. 5–19.

[11] Alessio Emanuele Biondo et al. 'Are random trading strategies more successful than technical ones?' In: *PloS one* 8.7 (2013), e68344.

[12] ZW BIRNBAUM et al. *ON A USE OF THE MANN-WHITNEY STATISTIC*. 1955.

[13] Fischer Black. 'Noise'. In: *The journal of finance* 41.3 (1986), pp. 528–543.

[14] Olivier Blanchard, Changyong Rhee and Lawrence Summers. *The stock market, profit and investment*. Tech. rep. National Bureau of Economic Research, 1990.

[15] Werner FM Bondt and Richard Thaler. 'Does the stock market overreact?' In: *The Journal of finance* 40.3 (1985), pp. 793–805.

[16] Gianluca Bontempi, Souhaib Ben Taieb and Yann-Aël Le Borgne. 'Machine learning strategies for time series forecasting'. In: *Business Intelligence*. Springer, 2013, pp. 62–77.

[17] Oslo Børs. *OBX*. Accessed: 31.03.2016. URL: http://www.oslobors.no/markedsaktivitet/#/details/OBX.OSE/overview.

[18] Leo Breiman. 'Bagging predictors'. In: *Machine learning* 24.2 (1996), pp. 123–140.

[19] Markus Konrad Brunnermeier. *Asset pricing under asymmetric information: Bubbles, crashes, technical analysis, and herding*. Oxford University Press, 2001.

[20] Li-Juan Cao and Francis EH Tay. 'Support vector machine with adaptive parameters in financial time series forecasting'. In: *Neural Networks, IEEE Transactions on* 14.6 (2003), pp. 1506–1518.

[21] cdipaolo. *Perceptron*. Accessed: 15.04.2016. URL: https://github.com/cdipaolo/goml/tree/master/perceptron.

[22] Yuehui Chen, Bo Yang and Ajith Abraham. 'Flexible neural trees ensemble for stock index modeling'. In: *Neurocomputing* 70.4 (2007), pp. 697–703.

[23] Kevin A Clarke. 'The phantom menace: Omitted variable bias in econometric research'. In: *Conflict Management and Peace Science* 22.4 (2005), pp. 341–352.

[24] Microsoft Corporation. *Azure Machine Learning*. Accessed: 17.04.2016. URL: https://studio.azureml.net/.

[25] Microsoft Corporation. *Azure Machine Learning Cheat Sheet*. Accessed: 17.04.2016. URL: https://azure.microsoft.com/en-us/documentation/articles/machine-learning-algorithm-cheat-sheet/.

[26] Oanda Corporation. *Oanda, Currencies*. Accessed: 17.04.2016. URL: oanda.com.

[27] Frank Cross. 'The behavior of stock prices on Fridays and Mondays'. In: *Financial analysts journal* 29.6 (1973), pp. 67–69.

[28] David M Cutler, James M Poterba and Lawrence H Summers. 'What moves stock prices?' In: *The Journal of Portfolio Management* 15.3 (1989), pp. 4–12.

[29] Michael Davis, Andrew Kumiega and Ben Van Vliet. 'Ethics, finance, and automation: A preliminary survey of problems in high frequency trading'. In: *Science and engineering ethics* 19.3 (2013), pp. 851–874.

[30] Saso Džeroski and Bernard Ženko. 'Is combining classifiers with stacking better than selecting the best one?' In: *Machine learning* 54.3 (2004), pp. 255–273.

[31] Espen Eckbo and David C Smith. 'The conditional performance of insider trades'. In: *The Journal of Finance* 53.2 (1998), pp. 467–498.

[32] U.S. Energy Information Administration (EIA). *EIA, Raw material prices*. Accessed: 17.04.2016. URL: eia.com.

[33] Eugene F Fama. 'Random walks in stock market prices'. In: *Financial analysts journal* 51.1 (1995), pp. 75–80.

[34] Eugene F Fama. 'The behavior of stock-market prices'. In: *Journal of business* (1965), pp. 34–105.

[35] Elia Formisano, Federico De Martino and Giancarlo Valente. 'Multivariate analysis of fMRI time series: classification and regression of brain responses using machine learning'. In: *Magnetic resonance imaging* 26.7 (2008), pp. 921–934.

[36] Yoav Freund, Robert Schapire and N Abe. 'A short introduction to boosting'. In: *Journal-Japanese Society For Artificial Intelligence* 14.771-780 (1999), p. 1612.

[37] A Ronald Gallant, Peter Eric Rossi and George Tauchen. 'Stock prices and volume'. In: *Review of Financial studies* 5.2 (1992), pp. 199–242.

[38] David E Goldberg and John H Holland. 'Genetic algorithms and machine learning'. In: *Machine learning* 3.2 (1988), pp. 95–99.

[39] Isabelle Guyon and André Elisseeff. 'An introduction to variable and feature selection'. In: *The Journal of Machine Learning Research* 3 (2003), pp. 1157–1182.

[40] Yasushi Hamao, Ronald W Masulis and Victor Ng. 'Correlations in price changes and volatility across international stock markets'. In: *Review of Financial studies* 3.2 (1990), pp. 281–307.

[41] Douglas M Hawkins. 'The problem of overfitting'. In: *Journal of chemical information and computer sciences* 44.1 (2004), pp. 1–12.

[42] Ralf Herbrich, Thore Graepel and Colin Campbell. 'Bayes point machines: Estimating the bayes point in kernel space'. In: *IJCAI Workshop SVMs*. 1999, pp. 23–27.

[43] David Hirshleifer, Tyler Shumway et al. 'Good day sunshine: Stock returns and the weather'. In: *Journal of finance* 58.3 (2003).

[44] Wei Huang, Yoshiteru Nakamori and Shou-Yang Wang. 'Forecasting stock market movement direction with support vector machine'. In: *Computers & Operations Research* 32.10 (2005), pp. 2513–2522.

[45] Cijo Jose et al. 'Local deep kernel learning for efficient nonlinear svm prediction'. In: *Proceedings of the 30th international conference on machine learning (ICML-13)*. 2013, pp. 486–494.

[46] Eamonn Keogh and Abdullah Mueen. 'Curse of dimensionality'. In: *Encyclopedia of Machine Learning*. Springer, 2011, pp. 257–258.

[47] Jan Ivar Larsen. 'Predicting stock prices using technical analysis and machine learning'. In: (2010).

[48] David J Leinweber. 'Stupid data miner tricks: overfitting the S&P 500'. In: *The Journal of Investing* 16.1 (2007), pp. 15–22.

[49] Mark T Leung, Hazem Daouk and An-Sing Chen. 'Forecasting stock indices: a comparison of classification and level estimation models'. In: *International Journal of Forecasting* 16.2 (2000), pp. 173–190.

[50] Andrew W Lo and A Craig MacKinlay. 'Stock market prices do not follow random walks: Evidence from a simple specification test'. In: *Review of financial studies* 1.1 (1988), pp. 41–66.

[51] Stephen Marsland. *Machine learning: an algorithmic perspective*. CRC press, 2014.

[52] Grant McQueen and V Vance Roley. 'Stock prices, news, and business conditions'. In: *Review of financial studies* 6.3 (1993), pp. 683–707.

[53] Ryszard S Michalski, Jaime G Carbonell and Tom M Mitchell. *Machine learning: An artificial intelligence approach*. Springer Science & Business Media, 2013.

[54] Tom M Mitchell et al. *Machine learning. WCB*. 1997.

[55] Randi Næs, Johannes Skjeltorp and Bernt Arne Ødegaard. 'What factors affect the Oslo Stock Exchange'. In: *Norges Bank (Central Bank of Norway), Working Paper* (2009).

[56] Douglas K Pearce and V Vance Roley. *Stock prices and economic news*. Tech. rep. National Bureau of Economic Research, 1985.

[57] James M Poterba and Lawrence H Summers. 'Mean reversion in stock prices: Evidence and implications'. In: *Journal of financial economics* 22.1 (1988), pp. 27–59.

[58] Morgan N Price, Paramvir S Dehal and Adam P Arkin. 'FastTree: computing large minimum evolution trees with profiles instead of a distance matrix'. In: *Molecular biology and evolution* 26.7 (2009), pp. 1641–1650.

[59] Bo Qian and Khaled Rasheed. 'Stock market prediction with multiple classifiers'. In: *Applied Intelligence* 26.1 (2007), pp. 25–33.

[60] Quandl. *Quandl Financial and Economic Data*. Accessed: 17.04.2016. URL: quandl.com.

[61] J. Ross Quinlan. 'Induction of decision trees'. In: *Machine learning* 1.1 (1986), pp. 81–106.

[62] AN Refenes et al. 'Financial time series modelling with discounted least squares backpropagation'. In: *Neurocomputing* 14.2 (1997), pp. 123–138.

[63] Rajiv Sant and Mir A Zaman. 'Market reaction to Business Week 'Inside Wall Street'column: a self-fulfilling prophecy'. In: *Journal of Banking & Finance* 20.4 (1996), pp. 617–643.

[64] Shai Shalev-Shwartz et al. 'Pegasos: Primal estimated sub-gradient solver for svm'. In: *Mathematical programming* 127.1 (2011), pp. 3–30.

[65] Jamie Shotton et al. 'Decision jungles: Compact and rich models for classification'. In: *Advances in Neural Information Processing Systems*. 2013, pp. 234–242.

[66] Phil Simon. *Too Big to Ignore: The Business Case for Big Data*. John Wiley & Sons, 2013.

[67] University Of Tromsø. *Titlon, financial database*. Accessed: 17.04.2016. URL: titlon.uit.no.

[68] Robert Tumarkin and Robert F Whitelaw. 'News or noise? Internet postings and stock prices'. In: *Financial Analysts Journal* 57.3 (2001), pp. 41–51.

[69] Jerold B Warner, Ross L Watts and Karen H Wruck. 'Stock prices and top management changes'. In: *Journal of financial Economics* 20 (1988), pp. 461–492.

[70] Wikipedia. *Linear Regression Analysis*. Accessed: 17.04.2016. URL: https://en.wikipedia.org/wiki/Regression_analysis.

[71] wikipedia. *Decision Tree*. Accessed: 15.04.2016. URL: https://en.wikipedia.org/wiki/Decision_tree_learning.

[72] wikipedia. *Neural Network*. Accessed: 15.04.2016. URL: https://en.wikipedia.org/wiki/Artificial_neural_network.

[73] wikipedia. *SVM*. Accessed: 15.04.2016. URL: https://en.wikipedia.org/wiki/Support_vector_machine.

[74] David H Wolpert. 'Stacked generalization'. In: *Neural networks* 5.2 (1992), pp. 241–259.

[75] David H Wolpert. 'The supervised learning no-free-lunch theorems'. In: *Soft Computing and Industry*. Springer, 2002, pp. 25–42.

[76] David H Wolpert and William G Macready. 'No free lunch theorems for optimization'. In: *Evolutionary Computation, IEEE Transactions on* 1.1 (1997), pp. 67–82.

[77] Qiang Yang and Xindong Wu. '10 challenging problems in data mining research'. In: *International Journal of Information Technology & Decision Making* 5.04 (2006), pp. 597–604.

Appendix

Table 5.1: Parameters Azure ML algorithms

Algorithm	Parameter Name	Value
Averaged Perceptron		
	Learning Rate	1
	Maximum number of iterations	10
	Random number seed	0
	Allow Unknown categorical levels	True
Bayes Point Machine		
	Number of training iterations	30
	Include bias	True
	Allow unknown values in categorical features	True
Boosted Decision Tree		
	Maximum number of leaves per tree	20
	Minimum number of samples per leaf node	10
	Learning Rate	0.2
	Number of trees constructed	100
	Random number seed	0
	Allow unknown categorical levels	
Decision Forrest		
	Resampling method	Bagging
	Number of decision trees	8
	Maximum depth of the decision trees	32
	Number of random splits per node	128
	Minimum number of samples per leaf node	1
	Allow unknown categorical features	True
Decision Jungle		
	Resampling method	Bagging
	Number of decision DAGs	8
	Maximum depth of the decision DAGs	32
	Maximum width of the decision DAGs	128
	Number of optimization steps per decision DAG layer	2048
	Allow unknown values for categorical features	True

Table 5.2: Preliminary results of days included for input

Algorithm	Parameter Name	Value
Locally Deep Support Vector Machine		
	Create trainer mode	Single Parameter
	Depth of the tree	3
	Lambda W	0.1
	Lambda Theta	0.01
	Lambda Theta Prime	0.01
	Sigmoid sharpness	1
	Depth of the tree	1; 3; 5; 7
	Lambda W	0.1; 0.01; 0.001
	Lambda Theta	0.1; 0.01; 0.001
	Lambda Theta Prime	0.1; 0.01; 0.001
	Sigmoid sharpness	1.0; 0.1; 0.01
	Feature normalizer	Min-Max normalizer
	Number of iterations	15000
	Number of iterations	10000; 15000; 20000
	Random number seed	0
	Allow unknown categorical levels	True
Logistic Regression		
	Optimization Tolerance	0.0000001
	L1 regularization weight	1
	L2 regularization weight	1
	Memory size for L-BFGS	20
	Random number seed	0
	Allow unknown categorical levels	True
Neural Network		
	Hidden layer specification	Fully-connected case
	The initial learning weights diameter	0.1
	Learning rate	0.1
	The momentum	0.0
	The type of normalizer	Min-Max normalizer
	Number of learning iterations	100 And 1000
	Shuffle examples	true
	Random number seed	0
	Allow unknown categorical levels	True
Support Vector Machine		
	Number of iterations	1
	Lambda	0.001
	Normalize features	True
	Project to the unit-sphere	False
	Random number seed	0
	Allow unknown categorical levels	True