ERASMUS UNIVERSITY ROTTERDAM

Erasmus School of Economics

Master Thesis Research Proposal [programme Econometrics and Management Science; Quantitative Finance]

A one-sided anomaly perspective in portfolio construction, by machine learning risk premia estimation.

Name Student: Alexander Debelle
Student ID number: 427410

Supervisor: Prof. Dr. Dick van Dijk
Second assessor: Dr. Rasmus Lönn

Date final version: April 8, 2022

List of Abbreviations

ANN	Artificial neural network
CAPM	Capital asset pricing model
DNN	Deep neural network
ES	Expected shortfall
FFNN	Feedforward neural network
GBT	Gradient-boosted tree
KNN	Kth nearest neighbors
LR	Linear regression
LSTM	Long short-term memory
MFCM	Multi-frequency combination method
MSE	Mean square error
RF	Random forest
VaR	Value-at-risk

Table of Contents

1 Introduction

In the vast universe of assets, there exists persistent pricing patterns, which continue to exist despite highly liquid markets facilitating efficient supply and demand. In an efficient market, these pricing patterns should be arbitraged away, however, they continue to prevail, and hence, are referred to as asset pricing anomalies. These anomalies have become the focal points of asset pricing studies, which beyond the underlying question of market inefficiency, have generated numerous competing theories for their existence.

An example of this is the size anomaly, where small firms, by market capitalization, continue to out-perform large firms. The size anomaly was introduced to the world in the seminal paper, Banz (1981), and then decades later, with the immense amount of literature it generated, this pricing pattern would be expected to have been arbitraged away, hence referring to it as an anomaly.

Justification of the size anomaly could be that smaller firms take up a much smaller portion of their respective market, and therefore, have more room to grow than larger firms. Alternatively, asset pricing literature pursues a deeper understanding of these anomalies, based of the assertion that the differences in returns can be explained by viable risk factors, often constructed from the anomaly itself. Fama and French (1993) perform the first dissection of this goal, where they construct a risk factor portfolio from the anomaly, made by taking the average return of the small assets minus the average return of the large assets, while making these two asset spaces invariant to the value anomaly. Then, this risk factor, or a combination of it with many other risk factors, attempt to explain the differences in returns found within the anomaly, with the goal of uncovering certain risk dimensions which the enable the anomaly to persist.

However, the research conducted in this thesis takes a stride away from traditional asset pricing, which focuses on explaining the existence of anomalies as well as using them to construct factor models, and instead accepts their existence, and questions what their implications are in the field of portfolio management. To begin, a key observation of asset pricing anomalies is that every one has a definitive characteristic: they categorize two sides to a universe of assets, where one side exhibits higher average returns than its adjacent category in the anomaly. Therefore, as a practical decision in portfolio construction, it may be of interest to only consider the asset space made up of the expected higher return side of the anomaly, rather than ignoring the anomaly, and considering all assets together. In essence, an anomaly can be used as a selection device to reduce the dimension of the universe of assets being considered.

This leads to the main research question presented in this thesis: should an investor, when creating long-only optimal portfolios as subsets of an asset space, strictly consider the assets in the expected higher return category of an anomaly? Or, does disregarding the anomaly, and considering all assets together in both sides of the anomaly, result in the same or higher average returns for the long-only optimal portfolios?

The term optimal throughout this thesis refers to achieving a higher level of mean return. This optimality metric is required for comparison purposes, as it is the standard metric used when measuring the difference between the two sides of an asset pricing anomaly. The condition arises from traditional asset pricing, which focuses on explaining these return differences of the anomaly through factor models. However, this argument from asset pricing of an anomaly existing because of a difference in the exposer to risk, if true, could lead to the portfolio's in the higher expected return category attaining a higher level of risk, along with a higher return. Therefore, it is also of interest to investigate the differences in risk between the two portfolios when they are constructed out-of-sample, to fully determine whether the anomaly should be ignored or not.

Furthermore, in the main research question, the optimal portfolio method selects a subset of assets, rather than deciding on non-zero weights over all of the assets, such as traditional Markowitz mean-variance theory. This optimal portfolio method, based on selecting a subset of assets, is largely inspired by the recent, machine learning-based portfolio construction methods used by Krauss, Do, and Huck (2017), Fischer and Krauss (2018), and Gu, Kelly, and Xiu (2020). These three papers display strong, out-of-sample returns, where mean-variance theory has been shown to lack a similar consistency by DeMiguel, Garlappi, and Uppal (2009).

Additionally, this question can be applied to each of the numerous anomalies found in asset pricing, or generally, any condition that separates a universe of assets into two categories where one is expected to achieve a higher mean return. Examples of this are for the latter, a split based on industry sectors, and for the former, the five anomalies which the risk factors are built off in the standard five-factor model from Fama and French (2015): market beta, size, book-to-market value, operating profitability, and investment. Each of these five anomalies splits the universe of assets into two categories, where one is expected to, on average, outperform the other, such that the question presented in this thesis remains pertinent. As well, the same holds true for any combination of multiple splits across anomalies or general conditions, which demonstrates the depth of the question.

The first step in answering the main research question is to identify an anomaly in recent years. Three of the most common anomalies are analyzed: size, value and momentum; each conducted with separate 1, 6 and 12 month holding periods. The size anomaly is defined as the average return of small firms minus the average return of large firms, measured by market capitalization. The value anomaly uses the book-to-market ratio and takes the average return of value firms minus the average return of growth firms. Lastly, the momentum anomaly takes the average return of assets which exhibited the largest return over the recent chosen time period, minus the average return of assets, which exhibited the smallest return over the recent chosen time period. For the momentum anomaly, there is expected to be a one month reversal effect, such that the following month for the high momentum assets exhibits poor returns (Asness, Moskowitz, & Pedersen, 2013). Therefore, for the 1 month momentum holding period, the anomaly is expected to be negative, and then for the 6 and 12 month holding periods, the return calculation is delayed by a month to account for this reversal effect.

The universe of assets used for the analysis is given by the Russell 3000 index membership lists, as it is the largest index, covering roughly 98% of U.S. companies, for a thorough and market wide perspective into the anomalies. As well, in order to maximize the presence of the three anomalies, as demonstrated in the introductory papers; Fama and French (1992) and Jegadeesh and Titman (1993), upper and lower decile portfolios of the Russell 3000 index are used. Lastly, the anomaly which is most present in recent years is chosen for the remainder of the analysis. This is found to be the momentum anomaly with a 12 month holding period.

The second step in answering the main research question is to create a pseudo out-of-sample testing set, which the anomaly exists in. This is taken to be eight recent years from the start of 2013 to the end of 2020. The reason that pseudo is inserted in front of the out-of-sample testing set, is because it is not truly out-of-sample, as it is chosen based on the existence of an anomaly. Therefore, the testing set used in this thesis, makes the assumption that a researcher or investor has an accurate prediction about an asset pricing anomaly existing in the future. In doing so, this enables testing the main research question of this thesis, such that if an anomaly were to exists, would the two portfolios converge in their mean return? On the contrary, if this pseudo out-of-sample testing set was not used, a guess would have to be made about the anomaly existing in the future. Then, if this guess were to be incorrect, the research is now invalid, as the main research question can no longer be examined.

Next, two categories of assets are defined, which are used as the asset spaces when forming the optimal portfolios. The first category is the 300 assets which belong to the expected higher return decile portfolio, being the ones which exhibited the highest return over the past 12 months. The second category consists of all 600 assets in both decile portfolios. Only 600 assets are used, rather than the entire Russell 3000 constituents list, as this guarantees that the second category will have a lower mean return than the first category, which is required to test the main research question. Essentially, at the start of every year, the universe of assets considered is only 600, then the anomaly is used to cut these assets into two halves. This is why decile portfolios of the Russell 3000 index are used, to enable a strong dissimilarity of returns between the two sides of the anomaly, while still being able to utilize a large number of assets when constructing the optimal portfolios.

Following, throughout the testing set, two versions of optimal portfolios are made. The first consists of a subset of assets, formed by only considering the 300 assets in the expected higher return category of the anomaly. The second consists of a subset of assets, formed by considering the entire 600 assets in the all assets category. Lastly, the main research question can then be answered. Such that, if the average return of the first optimal portfolio is larger than the average return of the second optimal portfolio, then this shows that the anomaly should be used to split the asset space before the optimal portfolio construction. Formally, only the assets in the expected higher return category of the anomaly should be considered as the asset space, when forming long-only, optimal portfolios as a subset of it. On the contrary, if the average return of the second optimal portfolio is the same or larger than the average return of the first portfolio, the anomaly can simply be ignored.

As for the construction of optimal portfolios as a subset of the asset spaces defined by the anomaly, the method used focuses on measuring and ranking equity risk premiums. This is fundamentally a task of prediction, as risk premium is the conditional expectation of future realized excess returns. Traditional asset pricing uses linear models with a large focus on factor construction for their estimation. However, with the tie made between risk premia and prediction, the vast field of machine learning, which often specializes in predictive tasks, can instead be applied to risk premia estimation, where the constraint of linear models, based off of few factors no longer applies.

Machine learning, as defined by Gu et al. (2020), resides in: "(a) a diverse collection of high-dimensional models for statistical prediction, combined with (b) so-called "regularization" methods for model selection and mitigation of overfit, and (c) efficient algorithms for searching among a vast number of potential model specifications." These characteristics enable machine learning models to cover rich, often nonlinear specifications in functional form, along with unique, and even occasionally unconstrained requirements for the feature set. The complex nature of equity risk premiums presents a very well suited environment to apply these models, and through their enhanced flexibility, brings hope of improved estimation and predictive accuracy.

Therefore, because of the ambiguous structure of equity risk premiums, various forms of machine learning models are applied. Each model is used to rank the assets, then construct optimal portfolios in order to answer the main research question of this thesis, by taking the top $k = \{5, 10, 20, 40\}$ number of assets in equally weighted, 1/N portfolios. Six of the most publicised machine learning prediction models are used: elastic net autoregressive linear regression (LR), Kth nearest neighbors (KNN) regression, random forest (RF), gradient-boosted tree (GBT), feedforward neural network (FFNN), and long short-term memory (LSTM) network.

The main research question, as it investigates the one-sided optimal portfolio potential of asset pricing anomalies, is of practical relevance, however, a second, theoretical contribution is made. This is the introduction of a novel forecasting methodology, called the multi-frequency combination method (MFCM), which is applied in two forms as the seventh and eighth machine learning models used for optimal portfolio construction. The introduction of MFCM leads to the second question of this thesis: does applying MFCM increase the accuracy of a forecasting model, and in turn, the performance of the optimal portfolios built from its predictions?

A brief description of MFCM is given as follows, with an in-depth, theoretical explanation presented in section 5, Methodology. MFCM works by taking a higher frequency dataset of the original time series and by applying overlapping temporal aggregation, expands it into multiple time series of decreasing frequency. It then constructs a forecast for each of the time series at different horizons, and lastly uses an activation function with the true desired horizon values as the endogenous variable and the varying horizon forecasts as the exogenous variables, to aggregate these forecasts together into a single point estimate.

This differs from standard forecasting combination techniques, which involve combining multiple models over the same horizon and frequency, where instead here the combination is made over the same model, but trained on different frequency data with varying forecast horizons. The justification for this method comes from the autocorrelation nature of time series data, such that if the regular data has dependence through autocorrelation, then the estimated forecasts of different horizons should also have dependence through autocorrelation. Additionally, the objective of MFCM achieving higher accuracy than a traditional single frequency forecast, is derived from its ability to find new patterns by utilizing different frequencies of the same time series through temporal aggregation.

However, the goal of MFCM is not the methodology in itself, but instead, to display that it is possible to combine forecasts over multiple horizons, in similar, parametric ways as the more traditional forecast combination techniques used over different models.

The second question to test if applying MFCM can increase accuracy is of both theoretical and practical relevance. It's of theoretical relevance, because MFCM is a new methodology which may potentially increase the accuracy of a point estimate, and if successful, could be expanded on in the scientific community. It also has practical relevance in a situation where a practitioner only wants to use one type of model for forecasting a time series as it closely resembles the distribution of data. This constraint disables the opportunity to use traditional forecast combination methods, as they rely on a multi-model framework. However, with MFCM, the practitioner is able to take advantage of the benefits of forecast combination while only using a single model. Additionally, the varying horizons of forecasts produced by MFCM can be used for aligning operational and strategic decision making, in lieu of only using their data for improving the single point estimation.

After the analysis is complete, six key findings are presented in the data. First, comes from the asset pricing anomaly analysis, which shows that there is a statistically significant decrease, both in terms of intercept and slope, of the size and value anomalies when taking a 12 month holding period over the last 28 years. The momentum anomaly, on the other hand, has been quite prominent over the past decade.

Second, the main research question is answered, finding that with a complex and accurate enough machine learning model, the optimal portfolios from the all assets category actually give a higher or close to identical mean weekly return than the optimal portfolios from the assets in the expected higher return side of the anomaly. Additionally, for this most complex model, 88% of the risk metrics for the all assets portfolios show less risk than their respective portfolio constructed from the anomaly's expected higher return asset space. Hence, in terms of average return as well as risk, the asset pricing anomaly can be ignored. However, this is only the case when applying the most complex machine learning model, being MFCM with a LSTM network as its activation function. Therefore, for all seven less complex models, the portfolios constructed from the expected higher return side of the anomaly achieved a higher mean weekly return than the portfolios constructed from all assets together. Accordingly, in order to ignore the asset pricing anomaly, an investor must be confident in the complexity and accuracy of their forecasting model to isolate the optimal assets.

Third, it is found that MFCM constructs the best performing portfolios by average return. Additionally, for MFCM that uses an LSTM network as its activation function, all of its portfolios have a mean weekly return which is higher than both naive 1/N portfolios, for all assets together and the anomaly's expected higher return assets.

Fourth, from the accuracy analysis it is clear that classification accuracy is the determining factor in the performance of the optimal portfolios, not mean squared error (MSE). Therefore, when constructing optimal portfolios as a subset of an asset space based on risk premia estimation, it is rather the probability of having a positive risk premia that is more important than the actual predicted value of the risk premia itself.

Fifth, out of the six standard models, elastic net LR had the lowest MSE, but LSTM had the highest classification accuracy. Additionally, out of the six standard models, LSTM is the only one where all eight of its portfolios beat their respective naive 1/N portfolio's mean weekly return.

Sixth, MFCM has by far the highest MSE due to large outlier errors in its predictions, but it also has by far the highest classification accuracy. This leads to MFCM being more suited towards a classification problem, rather than a regression problem.

The remainder of this thesis is organized as follows. Section 2 provides an overview of relevant and related literature, and section 3 covers the data sample. Section 4 introduces the null hypothesis, and Section 5 discusses the methodology. Section 6 presents the results and examines key findings, and lastly, Section 7 concludes and provides directions for further research.

2 Literature

2.1 Asset Pricing Anomalies

Anomalies in asset pricing date back over half a century to the capital asset pricing model (CAPM) for market betas introduced in Sharpe (1964). Following this were four other seminal papers, the first three being; Banz (1981), Fama and French (1992), and Fama and French (1993). These introduced size and value anomalies in asset pricing, and the ability to turn them into risk factors, providing much deeper insights than the CAPM alone. The fourth paper, by Jegadeesh and Titman (1993), introduced momentum as the first anomaly which is not a firm characteristic, but instead a characteristic of the asset price itself. These return patterns are considered anomalies, because in an efficient market they should be arbitraged away through price discovery by supply and demand, but despite this, they prevail even in today's highly liquid markets. Additionally, the anomalies may still exist as they are driven by genuinely relevant risk dimensions, such as small firms having a larger chance of default, to the point where the anomalies represent real risk premia.

A large-scale and more recent example of this is given by Asness et al. (2013), who performs an analysis of the value and momentum anomalies jointly across eight diverse markets and asset classes. They find consistent evidence of the value and momentum anomalies across all markets studied, and over a vast amount of assets, including government bonds, currencies and commodities. Additionally, they discover striking correlations with each anomaly across seemingly unrelated markets based on the anomaly portfolios. This suggests the existence of common global factors relating to value and momentum, and displays the magnitude of existence for these anomalies.

This thesis investigates three of the most commonly cited anomalies in asset pricing: size, value and momentum. However, there is an absence of literature regarding forming optimal portfolios as a subset of one side of a pricing anomaly. This may have vast implications in portfolio construction, and answers the practical question: should all assets, or only the assets in the higher expected return side of the anomaly be considered when constructing optimal portfolios as subsets of an asset space?

2.2 Multi-Frequency Combination Method

As for literature relating to the development of MFCM, it begins with Mincer and Zarnowitz (1969), who first proposed using a forecast as an exogenous variable in a regression. However, they use a single variable regression not to improve predictive performance, but instead to evaluate the accuracy of a forecasting model. Following was Granger and Ramanathan (1984), who pioneered using a regression as a method for combing forecasts over multiple models. Next, Coulson and Robins (1993) use forecasts as exogenous variables in a regression to enhance predictive performance. However, the forecasts used are of time periods in the past, where the true value is already known. Such as, at time T, using the forecast that was constructed for time T-1 to predict the value at time T+1. This leads them to conclude that the forecasts do not enhance predictive performance and should be taken out of the regression.

Kline (2004) investigates multi-step time series forecasting, and states three approaches: (1) the iterative method that uses a single step ahead model to iteratively create forecasts; (2) the independent method that uses a dedicated dataset to forecast each forecast horizon; and (3) the joint method that uses a single dataset to forecast all forecast horizons. MFCM uses the second approach, being an independent method, which involves multiple datasets of different frequencies to forecast each horizon separately. However, MFCM could simply be adjusted to use either of the other two approaches, but the second approach is chosen to exploit how different frequency datasets display different patterns in the time series.

Kourentzes, Petropoulos, and Trapero (2014) introduce the multiple aggregation prediction algorithm methodology, MAPA, which is the starting piece of MFCM proposed in this thesis. They consider demand forecasting, and begin by applying non-overlapping temporal aggregation to create multiple datasets of different frequencies. Next, they apply exponential smoothing to the data which enables them to separate each time series into its time series components, being level, trend and seasonality. They then construct different forecasts for each time series component over the varying frequencies and matching forecast horizons. After this, because of the highly linear demand data, they bring each of the time series components to the desired horizon by an additive or multiplicative structure, and then add them together to get the final forecast.

A following paper to Kourentzes et al. (2014) is done by Petropoulos and Kourentzes (2014), who demonstrate using the MAPA algorithm without the separation into time series components. They again use the linear structure of demand to multiplicatively expand all frequency forecasts to the same horizon. For example, with a monthly forecast, when quarterly is the desired horizon, the monthly forecast is multiplied by three to bring it to the quarterly horizon. The MFCM is a generalization of the MAPA extension used in Petropoulos and Kourentzes (2014).

The differences between MFCM and related literature begins by expanding on Granger and Ramanathan (1984), who show a regression to be successful for combining forecasts from multiple models. MFCM then expands on this by using a regression to combine multiple horizons of forecasts from the same model. Following is an expansion of Coulson and Robins (1993), where instead of using past forecasted values as predictors for out-of-sample horizons, use future forecasted values. Lastly, and most recent, is the expansion done by Petropoulos and Kourentzes (2014) of the MAPA algorithm introduced in Kourentzes et al. (2014). They assume linearity of data, and use multiplication to bring the lower frequencies to the desired horizon, then use traditional multi-model combination techniques. MFCM is a generalization of this model, where instead a complex structure is assumed to the different forecasted horizons, which directly relates them to the desired horizon. This structure can take the form of any parametric or non-parametric model, such as a neural network or Kth nearest neighbours regression.

MAPA additionally uses non-overlapping temporal aggregation, which causes every higher frequency used to be divisible by each frequency lower than it. An example of this as applied by Petropoulos and Kourentzes (2014), is starting with monthly data, and temporally aggregating to quarterly, semi-annually and annually. Requiring this divisibility constraint causes all the time series to have observations falling at the same occurrences as the highest frequency. Then, when constructing a forecast at time t, each frequency can utilize the most recent observation. However, this also means that any frequency which does not follow the divisibility rule of non-overlapping temporal aggregation, such as every 5 months in the example, cannot be used. Therefore, by applying overlapping temporal aggregation in MFCM this issue is no longer present, where now, any frequency can be used since the overlapping structure causes their observations to always align.

2.3 Machine Learning Portfolio Construction

The machine learning models used for forecasting in this research are largely inspired by three papers. The first is by Krauss et al. (2017), who apply deep neural networks (DNN), gradient-boosted trees (GBT) and random forests (RF) to a simple daily trading strategy of S&P 500 assets. Their trading strategy covers a period from December 1992 until October 2015, and forms daily classification-based, long-short portfolios. This results in average daily percentage returns prior to transactions costs of 0.33 for the DNN, 0.37 for the GBT and 0.43 for the RF. However, they find that the returns of these strategies degrades in recent years starting from 2001, likely due to the markets growing to be more efficient with respect to these machine learning methods.

An extension of Krauss et al. (2017) is done by Fischer and Krauss (2018), who use a very similar methodology but instead focus on applying a long short-term memory (LSTM) neural network. The data, training and testing sets, and number of long-short portfolio assets analyzed are all the same. They find that the LSTM network results in an average daily percentage return prior to transaction costs of 0.46, which far outperforms the other neural network, DNN, of 0.32. However, a similar result is found with the LSTM as with all the models in Krauss et al. (2017), such that the returns in the years following 2009 are again arbitraged away.

The third machine learning paper is Gu et al. (2020), who conduct a large-scale comparative analysis of forecast-based machine learning methods for measuring risk premiums in asset pricing. Since they are measuring risk premium, regression-based models are used instead of classification and they consist of; linear regression, generalized linear models with penalization, dimension reduction via principal components regression and partial least squares, regression trees (including boosted trees and random forests), and feed-forward neural networks. Overall, the regression trees and neural networks perform the best in terms of R-squared, and neural networks give the highest Sharpe ratio when forming long-short decile portfolios.

As well, their analysis consists of a vast feature evaluation using more than 900 predictors. They find the most informative category across all models to be price trend variables, including stock momentum, industry momentum, and short-term reversal. The second and third best performing categories are liquidity variables and return volatility respectively.

Lastly, the trading strategy used in this paper is based on machine learning forecasting models to develop portfolios generating short-term arbitrage. Initial works of this involve Huck (2009) who forms weekly forecasts using Elman neural networks, then conducts a pairs based long-short trading strategy with an optimal ten assets. Furthermore, the trading strategy used in this paper is largely inspired by Krauss et al. (2017), Fischer and Krauss (2018), and Gu et al. (2020), who all deploy similar portfolio optimization strategies based on risk premium forecasts constructed through machine learning models.

The relations and differences in the optimal portfolio construction applied in this thesis to the relevant literature begins with the trading strategy implemented being largely similar to the one used by Krauss et al. (2017). However, due to their, as well as Fischer and Krauss (2018) findings that the portfolios constructed through these forecast-based, machine learning methods degrades following 2009, only more recent data is used in the testing sets, to better ensure the veracity of the results found in this thesis

Three more notable differences from Krauss et al. (2017) and Fischer and Krauss (2018) are: (1) the use of the Russell 3000 index instead of the S&P 500 index; (2) two additional models in the form of a LR, and a KNN regression are applied; and (3) the use of regression instead of classification. This third difference relates to forecasting risk premium as routinely done in asset pricing, and as well to using the most standard framework of MFCM, which uses a regression model when aggregating the multi-frequency forecasts together. However, to transform MFCM into a classification problem requires only a simple expansion by using any classification model as the activation function, such as a logistic regression.

As for Gu et al. (2020), weekly frequency of data is used instead of monthly, and additionally less of an emphasis is put on predictors. Only momentum predictors are used in this thesis, which are consistently the most important predictors in Gu et al. (2020), and the only ones used in both Krauss et al. (2017) and Fischer and Krauss (2018). Another extension, and one inspired by Fischer and Krauss (2018), is the use of a recurrent neural network, the long short-term memory network, along with the standard feedforward neural network, as used in Gu et al. (2020). Lastly, a focus is put on portfolio returns instead of R^2 or Sharpe ratios, as this is a requirement in answering the main question of this thesis.

3 Data and Computing

3.1 Data

The Russell 3000 index is used, which consists of the 3,000 largest U.S. traded companies, which represents about 98% of all U.S. publicly traded, incorporated equity securities (FTSE-Russell, 2021). This choice is motivated by it being the largest index of U.S. companies which provides a thorough and market wide perspective into asset pricing anomalies. The Russell 3000 index is rebalanced on the last Friday of June every year, and the total data analyzed in this thesis consists of every asset on the Russell 3000 index from July 1991 until December 2020. However, four-year intervals of the index are used, meaning that the assets analyzed on the Russell 3000 index are updated every four years, instead of yearly. This is applied because FTSE Russell will only give a researcher access to a certain number of membership lists, and as of July 2020, these became licensed content and have since been removed from the Compustat database.

Using the past Russell 3000 indexes in this way eliminates survivorship and selection bias in the data as similarly done in Krauss and Stübinger (2017). This is accomplished because at every point in time, the assets in consideration are from the most recent past Russell 3000 membership list, such that the modeling framework does not know if these assets will still be in the Russell 3000 index in the future. Additionally, the start date of 1991 is chosen as it falls after the great moderation period in the 1980s (Davis & Kahn, 2008).

Pricing data is acquired from CRSP where weekly frequency is used, however, a higher frequency, chosen to be daily, is used for the MFCM. Stock splits are accounted for by dividing each asset's price at every point in time by CRSP's cumulative factor to adjust price, *cfacpr*, variable.

Additionally, delisting codes in CRSP are used to account for returns gained through bankruptcy, liquidation, mergers, acquisitions, or any other event which requires an asset to be delisted from a stock exchange. Specifically, the CRSP variable delisting return, *dlret*, is used to calculate an asset's return of its final week when delisted.

As well retrieved from CRSP is monthly data for calculating the market capitalization of each company as required in constructing the size anomaly. This is done by making use of CRSP's database specific labels, permco and permno. These are used because they stay the same throughout a company's life, regardless of any name changes, ticker symbol changes or declaration of Chapter 11 Bankruptcy. Each company has a single permco code assigned to it, but can have multiple permno codes, representing all traded securities of that company, such as common shares, preferred shares, etc. Therefore, calculating market capitalization of a company at any point in time requires taking each permno code's price multiplied by its shares outstanding. Then summing together all the individual permno's market capitalizations, gives the total market capitalization of the company, which is used in the size and value anomalies.

The value anomaly additionally requires the book value of each company, which is retrieved from Compustat at a quarterly frequency. It is then translated to a monthly frequency by simple repetition, taking a quarterly value to be the book value of equity of its following three months. This is then applied for calculating the book to price ratio as required in the value anomaly. The book to price ratio is used instead of the standard price to book ratio to account for negative book values, where by using the former ratio a negative book value does not interfere with if a company is a growth or value asset. The last anomaly considered, momentum, requires monthly price returns which are found in CRSP.

3.2 Software and Hardware

All code and data handling is entirely conducted in Python 3.8 (Van Rossum & Drake, 2020), relying on packages pandas (McKinney, 2010) and numpy (Walt, Colbert, & Varoquaux, 2011). Tree-based ensemble models, including RF and GBT, use the package XGBoost (Chen & Guestrin, 2016). Neural networks, including FFNN and LSTM, are conducted in Keras (Chollet et al., 2015) on top of Google TensorFlow (Abadi et al., 2016). Then, LR and KNN regression use the package sci-kit learn (Pedregosa et al., 2011).

LR, KNN, RF and GBT regression models are trained on an Intel i7-10750H CPU. However, the neural networks, FFNN and LSTM, make use of CUDA's platform for parallel computing on a NVIDIA Quardro T2000 GPU.

4 Null Hypothesis

An anomaly categorizes an asset space which achieves a higher average return than the adjacent category in the anomaly. The anomaly can then be used as a selection device to reduce the dimension of the universe of assets being considered. Therefore, the main question is: should an investor, when creating long-only optimal portfolios as subsets of an asset space, strictly consider the assets in the expected higher return category? Or, does disregarding the anomaly, and considering all assets together in both sides of the anomaly, result in the same or higher average returns for the long-only optimal portfolios? The null hypothesis of this question goes along with the standard practice in portfolio construction, which ignores the anomaly, such that considering all assets together is beneficial when constructing an optimal portfolio.

In order to answer this question, a simple forecast-based portfolio strategy is implemented multiple times, while increasing the complexity of the machine learning model used when constructing the forecasts. Applying this method instead of using only a single model gives a greater chance of accepting the null hypothesis. The goal then is to find a certain complexity level that intercepts the returns of the two optimal portfolios constructed as subsets of the two asset spaces; all assets and the anomaly's expected higher return assets. From here on, the anomaly's expected higher return asset space is referred to as anomaly assets.

It is important to note that the all assets category, is made up of both sides of the anomaly. Hence, it has twice as many assets as the anomaly assets category, and includes every single asset that is in the anomaly assets category. A graphical representation of the null hypothesis is given in Figure 1.

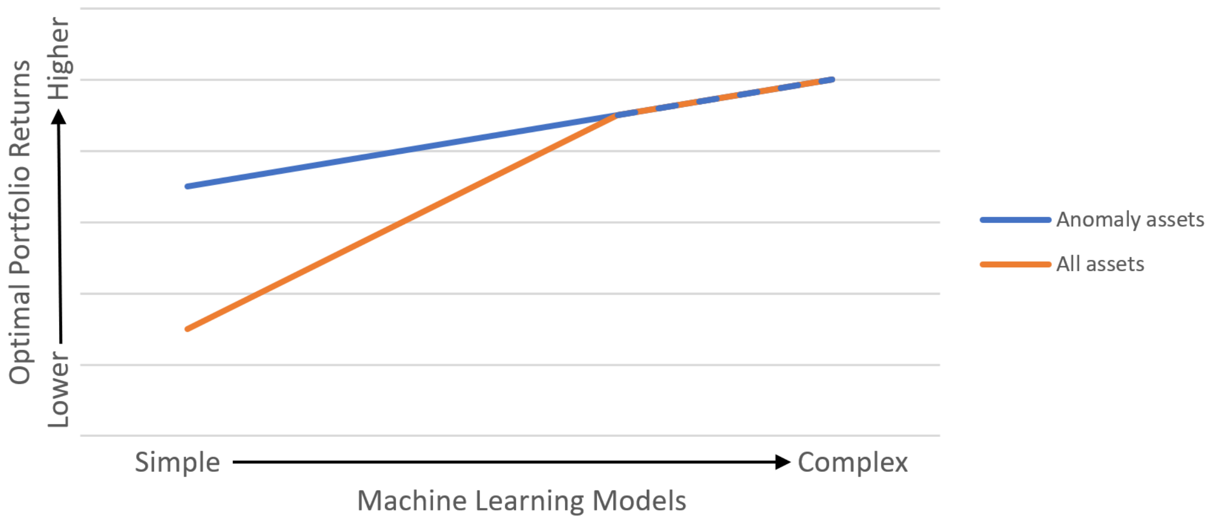


Figure 1: Null hypothesis: advanced machine learning forecasting methods cause convergence in the optimal portfolios.

In Figure 1, it is seen that when using the simply machine learning models to forecast risk premiums, which the optimal portfolios are built off of, the anomaly effect still prevails. The justification of this falls in line with the predictions being very close to random. Hence, when selecting a subset of assets from each category for the optimal portfolios, by the law of large numbers, the optimal portfolio returns will converge to the average return of their respective category. This results in the anomaly effect still being present.

As the complexity of the machine learning model increases, due to the complex nature of stock returns as confirmed by Krauss et al. (2017) and Gu et al. (2020), the predictions become more accurate and less random. Then, the machine learning models are able to accurately predict, and select, the assets which give the highest returns. Since the all assets category includes every asset in the anomaly assets category, and the assets in the anomaly assets category give a higher average return, the models will choose the same subset of assets in each optimal portfolio. This leads to both optimal portfolios exhibiting the same returns, hence the two lines in Figure 1 converge.

However, this is only the null hypothesis as done in standard practice, but indeed these two portfolios may not converge. Therefore, if the null hypothesis is not rejected, then the anomaly can simply be ignored as a preselection device when constructing optimal portfolios as a subset of the asset space, conditional that a complex enough forecasting model is used. However, if the null hypothesis is rejected, this would lead to an interesting implication in portfolio construction; the anomaly should be used as a preselection device, such that only the family of assets in the expected higher return category of the anomaly should be considered as the asset space, when forming optimal portfolios as a subset of it.

5 Methodology

5.1 Finding an Anomaly

The core question of this research relies on the fact that an anomaly exists in the data. Therefore, the first step in this methodology is to locate an anomaly which is then used as the focal point of the analysis. To do this, three of the most commonly researched anomalies in asset pricing are investigated, being size, value, and momentum (Back, 2017). Then the one which is most prominent in recent years is chosen for the remainder of the analysis.

This is done by first collecting the Russell 3000 membership lists since 1991. Then for each membership list, collect its ticker symbols, and transfer them into the association permno codes in CRSP used on the specific Russell rebalance day. Doing this ensures that the permno codes represent all the securities used throughout the Russell 3000 index since 1991. Aggregating this list to include only unique permno codes, it is then used to find, at a monthly frequency, all market capitalization, book to price ratio, and momentum data from July 1991 to December 2020 as described in section 3.1, Data.

Once this data is in place, the actual anomalies are calculated as follows. Each month, two equally weighted decile portfolios are constructed, with the first portfolio containing the 300 assets in the expected higher return side of the anomaly, and the second portfolio containing the 300 assets in the expected lower return side of the anomaly. Decile portfolios are used to enforce the effect of the anomaly as much as possible. For size and value, this is seen from the introductory paper, Fama and French (1992), in Tables I and IV for size and value respectively.

For momentum, the decile effect is as well shown in the introductory paper, Jegadeesh and Titman (1993), in Table III. For all three of these anomalies, it is seen that the difference between the decile portfolios consistently exhibits the largest impact of the respective anomaly.

This displays the benefit of using the Russell 3000 index as opposed to a smaller index for anomaly analysis. Where two decile portfolios can be taken in order to enforce the anomaly to the greatest extent, while still including a large amount of 600 assets for analysis. Additionally, since the Russell 3000 index consists of approximately 98% of all U.S. stocks, the difference between these two decile portfolios shows the magnitude of the anomaly throughout the U.S. stock market.

As well, the universe of assets considered every month to form these decile portfolios consists of only the assets in the most previous Russell 3000 index. Hence, these portfolios are rebalanced each month, with every four years considering an updated universe of assets given by the Russell 3000 index. Lastly, the anomaly is the difference between the average returns of the two decile portfolios. The larger this difference, the more prominent the anomaly is in the Russell 3000 index.

Additionally, this process is repeated for 6 month and 12 month holding periods for each of the size, value and momentum anomalies. As for momentum, its previous return, which is used to decide the decile portfolios, is always calculated by going back the same length of time as the holding period. As standard in momentum literature, the most recent monthly return is ignored when calculating an asset's 6 or 12 month return. This is done because of the existence of a 1 month reversal in stock returns, which could be related to microstructure or liquidity issues (Asness et al., 2013). This one month reversal anomaly, is also expected to be seen when calculating the one month momentum anomaly, which could still be used as the anomaly of interest in this thesis.

To conclude, nine anomalies are analyzed in total; 1 month, 6 month and 12 month, for each of the value, size and momentum anomalies. Then, the anomaly which is most prominent in the data in recent years is chosen for the remainder of the analysis. Applying this choice forces the existence of an anomaly in the out-of-sample testing set, which is required to test the null hypothesis of this thesis.

5.2 Optimal Portfolio Construction

Once the anomaly is chosen, two separate universes of assets are defined; all assets and anomaly assets, with 600 and 300 assets respectively. The assets in each universe are rebalanced at the same interval as the chosen anomaly; 1, 6 or 12 months, to provide consistent enforcement of the anomaly.

The trading strategy conducted is very similar to the ones implemented in Krauss et al. (2017), Fischer and Krauss (2018), and Gu et al. (2020). Two differences lie in the use of a weekly frequency, and as well holding the assets throughout the week from Monday open to Friday close, called the intraweek returns. The former difference is used to increase the training data available compared to the standard monthly returns used in asset pricing, as well as proving a suitable framework to use the MFCM. The latter difference is applied to enforce practicality in the trading strategy, such that the previous Friday close price is then possible to use as a predictor.

Therefore, the trading strategy is as follows. Every weekend, construct forecasts for each asset in the universe predicting the return from Monday open to Friday Close. Following, take the k assets with the highest predicted weekly return, form an equally weighted portfolio and buy them at the Monday open price, then sell them at the Friday close price. Since intraweek returns are being predicted, the machine learning forecast models are trained on Monday open to Friday close returns. Then, as similarly done in Krauss et al. (2017) and Fischer and Krauss (2018), a total of 26 return observations, representing 26 weeks or half a year of data, are used as predictors.

Using only momentum predictors in a standard configuration provides an unbiased proving ground for each of the models, because some require different forms of predictors to perform their best. For example, a LR can't have features highly correlated which results in multicollinearity, and each of these predictors must be covariance stationary. Then for DNN these conditions do not apply due to their overparameterization (Veaux & Ungar, 1994). However, also due to their overparameterization, having too many predictors may lead to overfitting and higher variance in out-of-sample testing. As for the recurrent neural network, LSTM, it requires each predictor to come as a sequence of time series observations, meaning for full functionally of LSTM a single predictor cannot be added to the set. Then, adding each predictor as a lengthy sequence contradicts the overfitting problem of a DNN, and the multicollinearity issue of a LR.

Lastly, the non-parametric models; RF, GBT and KNN regression, do not have any of these requirements for a feature set. Hence, by adding more features, and at the magnitude required by LSTM, it will unconditionally benefit them in-sample with the chance of increasing out-of-sample accuracy, while possible hindering the parametric models.

As for the number of assets, k, tested in the optimal portfolios, motivation comes from Krauss et al. (2017) who implemented a similar methodology testing values of $k = \{10, 50, 100, 150, 200\}$, and finding $k = 10$ to give the largest average return by a substantial margin. Additionally, Fischer and Krauss (2018) implemented an identical strategy but using an LSTM network, and again finding $k = 10$ to give the largest average return by almost double of the second place portfolio. Lastly, Huck (2009) as well applied a similar trading strategy, testing values of $k = \{1, ..., 45\}$, and finding $k = 3$ to give the largest average return. Since an asset pricing anomaly is defined by the difference in average returns of the respective two categories, a focus on average returns has to be made in order to analyze the null hypothesis of this thesis. Therefore, since it is clear from the previous literature that a lower value of k gives a higher average return, the values of k tested here is the set; $\{5, 10, 20, 40\}$.

Two important notes regarding optimal portfolio construction are: first, since the past 26 weeks of observations are used as predictors, assets need to be at least 26 weeks old to be used in an optimal portfolio; second, a no short selling constraint is applied as it does not add prominence to the main research question. However, the research question would still hold if short selling were allowed, where the question would then change to: should an investor only go long in assets on the higher expected return side of the anomaly, and only go short in assets in the lower expected return side of the anomaly? Although interesting, both questions, with and without short selling, give identical repercussions, hence short selling is ignored.

5.3 Training and Testing

The testing set used is the most recent years where the chosen anomaly is present, ideally over at least five years of data. This is enforced because of the findings of Krauss et al. (2017) and Fischer and Krauss (2018), who find their machine learning forecasting methods becoming considerably less effective following 2009, as mentioned in section 2, Literature.

Training takes place over two separate universal models, with both cross-sectional and time series components: (1) trained on the 600 assets in the all assets category, and (2) trained on 300 assets in the anomaly assets category. Therefore, each universal model is trained over every asset in its category at once. Then, the time series component comes from each asset in the category including the entire life of its time series, back to maximum 1991, in order to only train models on data following the great moderation of the 1980s (Davis & Kahn, 2008).

Training two universal models is done since these sets are the only assets each model is able to select for the optimal portfolios, which in practice is what the model is trained on (Gu et al., 2020). Therefore, every training set, up to a point in time before testing, has the endogenous variable of the intraweek returns of each asset in the respective category, over the entire life of the asset back to maximum 1991. Then the exogenous predictors for each endogenous observation consist of the 26 intraweek returns of the respective asset.

The models are retrained at the same interval as the assets are adjusted in the anomaly, being either 1, 6 or 12 months. This enables training a model on only the assets in the current category, which are available to be selected in the optimal portfolios. Additionally, an expanding window is used, such that each time the models are retrained, the starting date of the data stays the same. An expanding window is chosen, because at each training interval, the field of assets changes as these are the new ones which define the anomaly decile portfolios of the Russell 3000 index. Therefore, even with a standard moving window, since the life of all assets do not go back to 1991, the number of observations trained on will still be different in each training interval. Additionally, a preference is made to have as much data as possible when training the models, which an expanding window allows for.

As often applied along with machine learning models, a validation set of data is isolated taking place right before the testing set, which is used for hyperparameter optimization. The goal of a validation set is to simulate an out-of-sample test of the models, while comparing different hyperparameter specifications. Hyperparameters in machine learning are parameters which must be set by the user, for example, the number of hidden nodes in a layer of a neural network or the number of nearest neighbors in a Kth nearest neighbor regression. The validation set takes place over the three years prior to the start of the testing set. After validating, the models are retrained on the training set and the validation set before the true out-of-sample testing.

5.3.1 Scaling

When training any of the models: KNN, FFNN, or LSTM; scaling of the data plays a crucial role. In a KNN regression, this is simply to enforce standardization of the distance metric, such that all features are used with an equal weight throughout the model.

For the neural networks, FFNN and LSTM, scaling of the training data is necessary to have stable error gradients, such that the coefficient weight values don't change dramatically after each observation is fed into the model.

Due to intra weekly asset returns having a distribution with extremely thick tails, and some very large outliers, a scaling technique that is robust to outliers is applied to the data:

$$x_{\text{scaled}} = \frac{x - \text{median}(x)}{\text{quantile}(x, 75) - \text{quantile}(x, 25)} \tag{1}$$

The reason (1) is robust to outliers is first, that it subtracts the median instead of the mean from every observation in the numerator. Second, it relies on quantiles as a proxy for the variance in the denominator. Both of these metrics are not influenced by outliers.

It is necessary here to take the precautions for outliers, because if using a more standard scalar in the neural network literature, such as the MinMax scalar between 0 and 1, the outliers would cause the vast majority of observations to be almost identical. Then, there would be little variation in the data for the models to learn from.

In order to mitigate the effect of larger in absolute value weekly returns taking up most of the weighting when minimizing the sum of squared errors, the scaling done in (1) is as well applied to the training data for LR, RF and GBT.

As a final precaution for mitigating asset return's thick tails when training the models, outliers are removed for observations where the endogenous intra week return is greater in absolute value than 75%. This accounts for approximately 0.1% of the training data.

5.4 Analysis

There are two main forms of analysis conducted in this thesis. The first is to answer the main research question; which universe of assets to consider. This is done by constructing the true, out-of-sample graph as hypothesised by Figure 1 in section 4, Null Hypothesis.

To determine if the difference between the weekly returns of each of the two optimal portfolios, being all assets and anomaly assets, is statistically significant, White's reality check test is applied. Additionally, six risk metrics are used for risk analysis of the optimal portfolios: standard deviation of returns, maximum drawdown, historical 95 and 99 percentage one week value-at-risk (VaR), and historical 95 and 99 percentage one week expected shortfall (ES).

The second form of analysis conducted is a comparison of the accuracy of all models by MSE. This involves comparing the new MFCM introduced in this thesis. The models trained using the all-assets universe is used for accuracy comparison, as they are trained on approximately twice as much data as the anomaly assets model. An additional focus is made on the classification accuracies of the regression models, such that if a positive return is predicted it is given a value of one, and if a zero or negative return is predicted, it is given a value of zero.

5.4.1 White's Reality Check

For finding statistical significance between portfolios or when comparing them to a return of zero, White's reality check, as introduced by White (2000), is applied. This test accounts for data snooping, which occurs when a set of data is used more than once for purposes of model comparisons or inference. In doing so, there is the possibility that a satisfactory result may be obtained simply due to chance rather than any merit from a particular model. This is often an unavoidable problem when testing with time series data, as typically only a single history that measures a certain phenomenon of interest is available for the analysis. Therefore, to account for the data snooping problem, White's reality check is applied, by testing against simulated observations found through bootstrapping with replacement.

Three forms of White's reality check are used, the first is by comparing a single model to a benchmark return of zero. The second, is by comparing all models together to determine if any of the models are able to give consistently significant results. The third, is by comparing each of the high category optimal portfolios to their respective all category optimal portfolios. An explanation of this test is as follows.

To begin, a benchmark is chosen, which for forms one and two is simply against having a zero return in every week. For form three, the benchmark is the all category optimal portfolio, as the goal is to test if its respective high category optimal portfolio generates higher returns. Next, the mean returns of the portfolios against their respective benchmark are found.

$$\bar{f}_x = \frac{1}{417}\sum_{t=1}^{417} \ln(1+f_{x,t}) - \ln(1+b_{x,t}) \quad \text{for} \quad x = \{1, ..., N\} \tag{2}$$

In (2) 417 is the number of weeks in the out-of-sample testing set, $f_{x,t}$ is the return for portfolio x at time t, and ln() represents the natural logarithm. Then, $b_{x,t}$ is the benchmark return, which for forms one and two is simply taken to be zero. In form three, $f_{x,t}$ is the high category optimal portfolio, and $b_{x,t}$ is its respective all assets portfolio. Lastly, N is 64 for testing all portfolios in forms one and two, and N is 32 for testing the high category portfolios against their respective all category portfolio in form three. $\bar{f}_x$ will later be used to compare it against the simulated portfolio returns.

Next, the bootstrapping process is applied, which creates, as used in this thesis, a total of 10,000 simulated portfolio return distributions.

$$\bar{f}^*_{x,i} = \frac{1}{417}\sum_{t=1}^{417} \ln(1+f_{x,\theta_i(t)}) - \ln(1+b_{x,t}) \quad \text{for} \quad x = \{1, ..., N\} \text{ and } i = \{1, ..., 10000\} \tag{3}$$

In (3) $\theta_i(t)$ represents the sampled time period used in the bootstrapped simulation. However, due to the dependent nature of time series data, a stationary bootstrap by taking single observations without regards to their dependent structure is not suitable. Therefore, the bootstrapping method as introduced by Politis and Romano (1994) is applied. This method produces blocks of random length having a geometric mean equal to $1/q$, where q is taking to be in between 0 and 1. As recommended by White (2000), a value of $q = 0.5$ is taken. Formally, this enables every observation chosen in the bootstrap to have a 50% chance of being the observation that chronologically follows the current bootstrapped observation. As well, then it has a 50% chance of being another random draw from the sample.

Following, the test imposes asymptotic normality of the means, and constructs the distribution values given below.

$$\begin{aligned} \bar{V}_x &= \sqrt{n}\bar{f}_x \quad \text{for} \quad x = \{1, ..., N\} \\ \bar{V}^*_{x,i} &= \sqrt{n}(\bar{f}^*_{x,i} - \bar{f}_x) \quad \text{for} \quad x = \{1, ..., N\} \text{ and } i = \{1, ..., 10000\} \end{aligned} \tag{4}$$

In (4) the expected value of $\bar{f}_x$ is taken to be zero, as the benchmark return series, $b_{x,t}$, which is the original return series's expected value, $f_{x,t}$, has already been subtracted from it. $\bar{V}^*_{x,i}$ represents the simulated distribution values, which has the expected value of mean returns of the portfolios against their respective benchmark, $\bar{f}_x$ as given in (2).

The first and second forms of White's reality check are used for a test on a single portfolio, without regards to the other 63 portfolios. The null hypothesis of this test is that the portfolio returns do not outperform the benchmark return, which is zero or the all category optimal portfolio returns for first and second form respectively. Additionally, the p-value for this test is found by determining how many of the simulated distribution values of portfolio return's means, are greater than the actual distribution value of the portfolio return's mean.

$$\begin{aligned} \text{p-value}_x &= \text{M}_x/10000 \quad \text{for} \quad x = \{1, ..., N\} \\ \text{M}_x &= \sum_{i=1}^{10000} I[\bar{V}^*_{x,i} > \bar{V}_x] \quad \text{for} \quad x = \{1, ..., N\} \end{aligned} \tag{5}$$

In (5) $I[]$ represents the indicator function, where the value is one if the condition in the brackets is met, and zero otherwise. Therefore, it is seen that if very few of the simulated distribution values of portfolio return's means, $\bar{V}^*_{x,i}$, are higher than the actual distribution value of the portfolio return's mean, $\bar{V}_x$, then the actual mean portfolio return is not due to chance.

The third form of White's reality check is used to test the null hypothesis that the portfolio among the 64 with the highest mean return is able to outperform the benchmark returns of zero. It tests this while accounting for data snooping by considering the randomness of not just the highest mean return strategy, but all other 63 strategies at the same time. To begin, it defines the distribution value of the highest mean return model.

$$\bar{V1} = \max_{x \in \{1,....,64\}} (\bar{V}_x) \tag{6}$$

Following, for every one of the 10,000 simulations, it finds which of the 64 portfolios had the maximum mean return.

$$\bar{V1}^*_i = \max_{x \in \{1,....,64\}} (\bar{V}^*_{x,i}) \quad \text{for} \quad i = \{1, ..., 10000\} \tag{7}$$

Lastly, the p-value of this test is found by determining how many of the simulated distribution values of portfolio return's means, are greater than the actual distribution value of the portfolio return's mean.

$$\begin{aligned} \text{p-value} &= \text{M}/10000 \\ \text{M} &= \sum_{i=1}^{10000} I[\bar{V1}^*_i > \bar{V1}] \end{aligned} \tag{8}$$

In (7) and (8) for each simulation, by considering all 64 portfolios when finding the maximum, it enables testing the highest mean return portfolio, $\bar{V}1$, against any randomness that could have risen across all 64 portfolios. Specifically, $\bar{V}1$ is compared against the bootstrapped portfolio whose randomness resulted in the highest mean return out of the 64 portfolios in each simulation.

5.5 Machine Learning Models

Figure 1 in section 4, Null Hypothesis, depicts on the x-axis machine learning models of increasing complexity applied to the optimal portfolios. As machine learning models are vast in their formulations, judging their complexity level across multiple families of models requires a subjective opinion. A machine learning model's complexity here is defined based on certain criteria, being; linear constraint, flexibility, parametric or non-parametric, number of parameters, and additional machine learning capabilities such as boosting or regularization. Therefore, in total, eight models are applied, and listing them from least complex to most complex gives; elastic net autoregressive LR, k-th nearest neighbors regression, random forest, gradient-boosted trees, feedforward neural network, long short-term memory network, MFCM with a linear activation function, and MFCM with an optimal activation function. For MFCM, the model that constructs forecasts using different frequencies is chosen to be the model of the previous six, which had the highest MSE accuracy in the validation set.

Each of these models, requires a degree of hyperparameter tuning. To accomplish this, the tournament style HYPERBAND algorithm is applied as introduced by Li, Jamieson, DeSalvo, Rostamizadeh, and Talwalker (2018). The algorithm is non-parametric and formulated on the idea of successive halving, which through an iterative procedure is applied to the number of observations and at the same time, to the number of model hyperparameterizations. Each iteration begins with a tournament style game played between all remaining models, where only a certain proportion make it to the next round. This proportion is decided by the hyperparameter of HYPERBAND, called the factor, which is chosen to be three, as this is the value suggested by Li et al. (2018). The factor controls the proportion of models which make it to the next round, and as well the rate at which amount of observations used in training the models grows by. Therefore, after each iteration, the number of model candidates is divided by three, and the number of observations used in the training set is multiplied by three. HYPERBAND continues to iterate until the number of candidate models is less than or equal to three, at which point every observation is used in the training set and a final model is chosen.

Using HYPERBAND as opposed to a standard gridsearch algorithm enables testing a much larger set of hyperparameters in a way that greatly reduces both runtimes and memory requirements. Additionally, two slight variations are made to the HYPERBAND algorithm in order to accommodate using time series data. The first is that no cross validation is applied during the tournaments. This is because in time series data, it is not possible in practice to train data on future observations of a testing set. Therefore, at each tournament iteration the testing dataset remains the same and takes place after the training data. Second, at each iteration, the models are always trained using the most recent data up until the testing set. This is applied, as again in practice, it is generally preferred to train time series models on the most recent set of data, up to the out-of-sample testing date.

Brief descriptions of each of the machine learning models applied is given throughout the remainder of this section. For variable notation, as described in section 5.2, Optimal Portfolio Construction, each of the models use intraweek returns between Monday open to Friday close for both the endogenous predicated variable, and the 26 lags of exogenous predictors. Throughout the model descriptions, the 26 exogenous predictors at time $t+1$ are given by the vector $\mathbf{X}_{t+1} = [X_t, X_{t-1}, ..., X_{t-25}]'$, and the endogenous, one period ahead forecast is given by a function unique to each machine learning model represented by $\hat{f}(\mathbf{X}_{t+1})$. Lastly, $f(\mathbf{X}_{t+1}) = X_{t+1}$ represents the in-sample, one period ahead endogenous observation of $\mathbf{X}_{t+1}$.

5.5.1 Elastic Net Linear Regression

The first model used is the standard econometrics LR with an elastic net imposed for regularization, as introduced in Zou and Hastie (2005). This model is characterized as follows:

$$\begin{aligned} \hat{f}(\mathbf{X}_{t+1}) &= \mu + \sum_{j=1}^{26} \hat{\phi}_j X_{t+1-j} + \varepsilon_{t+1} \\ \hat{\boldsymbol{\phi}} &= \underset{\boldsymbol{\phi}}{\operatorname{argmin}}(||f(\mathbf{X}_{t+1}) - \mathbf{X}'_{t+1}\boldsymbol{\phi}||^2 + \alpha\rho||\boldsymbol{\phi}||^2 + \alpha(1-\rho)||\boldsymbol{\phi}||_1) \end{aligned} \tag{9}$$

In (9), μ is the intercept, ε_{t+1} is the error term, and ϕ_j's are the regression coefficients. When solving the regression coefficients, α is the regularization parameter, and ρ is the weighting applied between the L1 and L2 regularizations.

The goal of the regularization parameters is to reduce in-sample overfitting and bias, which in turn should reduce out-of-sample variance. It does this by shrinking the regression coefficients towards zero, by placing a penalty on them when minimizing the sum of squared errors, α, which is proportionate to how large they are in absolute value.

The two standard regularization techniques are Lasso and Ridge. Lasso uses a L1 regularization by considering the absolute value of the coefficient. Applying this has the tendency to force the coefficient to be exactly zero. On the other had, Ridge uses a L2 regularization by considering the squared value of the coefficient. Applying this has the tendency to move the coefficient towards zero, but not exactly zero. The elastic net is a combination of Lasso and Ridge, which by the weighing parameter, ρ, determines the split between the two regularization methods. Elastic net is used, because the weighting parameter, ρ, can be taken as a hyperparameter, and then HYPERBAND can decide which form of regularization should be selected. The hyperparameters tested for the elastic net LR are given in Table 1 at the end of Methodology section.

5.5.2 K-th Nearest Neighbors Regression

The second model is the non-parametric KNN regression as introduced by Altman (1992), and is built off the K-th nearest neighbors algorithm introduced by Fix and Hodges (1951). Quite simply, this model considers the current 26 lags, and then finds the k sets of concurrent 26 lags in the training data which are the closest to the current 26 lags. Then, as an estimate of the future value, it takes the average of the k sets of lag's one period ahead values:

$$\hat{f}(\mathbf{X}_{t+1}) = \frac{1}{k}\sum_{i=1}^{k} f(\mathbf{X}^*_{t+1,i}) \tag{10}$$

In (10), $\hat{f}(\mathbf{X}_{t+1})$ is the forecasted value and $f(\mathbf{X}^*_{t+1,i})$ is the one period ahead value of the i^{th} set of lag's in the training set, which are the closest distance to the current set of lags at time t.

Distance between lags is found by the Euclidean metric given by:

$$\begin{aligned} \mathrm{D}(\mathbf{X_{t+1}}, \mathbf{X^*_{t+1,i}}) &= \sqrt{(\mathbf{X_{t+1}} - \mathbf{X^*_{t+1,i}})^2} \\ &= \sqrt{(X_t - X^*_{t,i})^2 + (X_{t-1} - X^*_{t-1,i})^2 + ... + (X_{t-25} - X^*_{t-25,i})^2} \end{aligned} \tag{11}$$

Additionally, a weighted KNN regression is also tested, which when making a prediction, gives weights to each endogenous observation as the inverse of its distance metric. Therefore, the observations with the smallest distance from the current set of lags are given larger weights:

$$\begin{aligned} \hat{f}(\mathbf{X}_{t+1}) &= \frac{1}{k}\sum_{i=1}^{k} w_i f(\mathbf{X}^*_{t+1,i}) \\ w_i &= \frac{\mathrm{D}(\mathbf{X_{t+1}}, \mathbf{X^*_{t+1,i}})^{-1}}{\left(\sum_{i=1}^{k} \mathrm{D}(\mathbf{X_{t+1}}, \mathbf{X^*_{t+1,i}})\right)^{-1}} \end{aligned} \tag{12}$$

The main hyperparameter of the KNN regression is the number of k nearest neighbors. The set of values of k tested for the hyperparameter optimization are given in Table 1.

5.5.3 Random Forest

Random forests as introduced by Breiman (2001), consist of combining many decorrelated decision trees built off different samples of data. The goal of a RF is to group observations which behave similarly by their feature space in each tree, and then derive the final forecast by averaging each tree's individual forecast. Subsampling of the dataset is done before constructing each tree to reduce the in-sample bias of trees, and additionally, decorrelation of the trees is done by subsampling the feature space throughout each tree.

A decision tree has three special characteristics; nodes representing each data split made within the tree, leaves representing the final nodes which house the endogenous observations, and lastly, rules which define a pathway to each leaf. At each node in a tree, a split is made in the data by a feature chosen from a random subsample of features, this causes the trees to be decorrelated. In a regression tree, the feature chosen at the node, as well as the value it is split on, are found by minimizing the MSE of the partitioned data, given by:

$$\min_{j,s}\left[\min_{c_1} \sum_{\mathbf{X}_{t+1}\in R_1(j,s)} (f(\mathbf{X}_{t+1}) - c_1)^2 + \min_{c_2} \sum_{\mathbf{X}_{t+1}\in R_2(j,s)} (f(\mathbf{X}_{t+1}) - c_2)^2\right] \tag{13}$$

In (13), j is the feature being split on, s is the split point in the feature, $R_i(j,s)$ is the i^{th} region made by the feature and split point, and c_i is the value minimizing the MSE of the observations in region i. The regions, as well as optimal minimum values for c_1 and c_2 are given by:

$$\begin{aligned} &R_1(j,s) = \{\mathbf{X}_{t+1} | X_{t+1,j} \leq s\} \quad \text{and} \quad R_2(j,s) = \{\mathbf{X}_{t+1} | X_{t+1,j} > s\} \quad \text{for } j \in \{1,2,...,26\} \\ &\hat{c}_1 = \text{ave}(f(\mathbf{X}_{t+1}) | \mathbf{X}_{t+1} \in R_1(j,s)) \quad \text{and} \quad \hat{c}_2 = \text{ave}(f(\mathbf{X}_{t+1}) | \mathbf{X}_{t+1} \in R_2(j,s)) \end{aligned} \tag{14}$$

The decision tree keeps repeating this process of splitting on the optimal features until the maximum depth is reached. At this point, each endogenous observation is partitioned to belong to a single leaf, determined by the rule leading to each leaf. Following, the forecast for a single tree is given by:

$$\begin{aligned} \hat{g}_i(\mathbf{X}_{t+1}) &= \sum_{k=1}^{K_i} \theta_{k,i} I[\mathbf{X}_{t+1} \in C_{k,i}] \\ \theta_{k,i} &= \frac{1}{|C_{k,i}|} \sum_{\mathbf{X}_{t+1} \in C_{k,i}} f(\mathbf{X}_{t+1}) \end{aligned} \tag{15}$$

In (15), i represents the tree, K_i is the total number of leaves in tree i, $C_{k,i}$ is the rule in tree i which leads to leaf k, and $|C_{k,i}|$ is the number of endogenous observations in the leaf belonging to rule $C_{k,i}$. Finally, the forecast given by the RF is the average of every tree's individual forecast, with T representing the total number of trees in the RF:

$$\hat{f}(\mathbf{X}_{t+1}) = \frac{1}{T} \sum_{i=1}^{T} \hat{g}_i(\mathbf{X}_{t+1}) \tag{16}$$

A RF is a non-parametric model, but has five key hyperparameters which require tuning. These are; number of decision trees in the forest, maximum depth of each tree, data subsample per tree, feature subsample per node, and a regularization term on the weights in each leaf. The hyperparameters tested in the HYPERBAND algorithm are given in Table 1. These are chosen by motivation from the hyperparameters used and found to be optimal in both Krauss et al. (2017) and Gu et al. (2020), as well as the recommendations by Chen and Guestrin (2016) in the XGBoost Python package.

5.5.4 Gradient-Boosted Trees

The GBT model is an extension of a RF with boosting, which is a machine learning technique introduced by Schapire (1990). Boosting works by sequentially applying weak learners to repeatedly re-weighted versions of the training data (Hastie, Tibshirani, & Friedman, 2009). After each boosting iteration, observations which have a larger forecasting error have their weights increased, while observations which have a smaller forecasting error have their weights decreased. Therefore, the weak learner applied at each iteration focuses on observations that had a larger forecasting error in the previous iterations.

The GBT model applies boosting to decision trees, identical to the ones used in a RF. In a regression setting, GBT begins by fitting a very simple estimation, plainly relying on the mean of the endogenous variable. Next, it calculates the estimation errors by MSE of the mean model and fits a decision tree to predict these errors. Once these predictions are made, it makes a second fitting to the data by adding the predicted errors, multiplied by a learning rate, to the estimations of the first mean model. This is the boosting step, such that by predicting the errors instead of the actual data, it focuses on the observations which resulted in the largest error in the previous step.

This iterative process of fitting a model to the errors, then adding its error estimates to the endogenous estimations of the previous iteration is continued until the desired number of trees is reached. At this point, a strong learner is built by sequentially applying weak learners to repeatedly re-weighted versions of the training data. Lastly, an out-of-sample forecast is made by an additive model, which works in the same steps of iteratively going through all the weak learners until the chosen number of trees is reached.

A simple definition of the boosting process in GBT is given by:

$$\hat{f}(\mathbf{X}_{t+1}) = \overline{f(\mathbf{X}_{t+1})} + \alpha \sum_{i=1}^{T} \hat{h}_i(\mathbf{X}_{t+1}) \tag{17}$$

In (17), $\overline{f(\mathbf{X}_{t+1})}$ is the starting estimate by taking the mean of the endogenous variable over each observation in the training set. Next, $\hat{h}_i(\mathbf{X}_{t+1})$ is the estimation at iteration i of the error terms from the previous iteration. Lastly, α is the learning rate which determines how quickly the estimate is adjusted throughout every iteration.

The hyperparameters of GBT are the same as the RF with the addition of a learning rate. The hyperparameters tested in the HYPERBAND algorithm are given in Table 1, and once again are chosen by motivation from both Krauss et al. (2017) and Gu et al. (2020), as well as the recommendations by Chen and Guestrin (2016) in the XGBoost Python package.

5.5.5 Feedforward Neural Network

Artificial neural networks (ANN) categorize a family of highly flexible models, with mathematical underpinnings, inspired from the body's nervous system, introduced by McCulloch and Pitts (1943). Their flexibility goes far enough that ANNs are proven to be "universal approximators", having the ability to approximate any smooth predictive association (Hornik, Stinchcombe, & White, 1989).

This thesis applies two types of ANNs, with the first one being the FFNN, which is the most standard ANN formulation. A FFNN is composed of three types of layers; an input layer taking in all the predictors, a hidden layer connecting the input layer to the output, and lastly, an output layer giving the prediction. Each layer has a series of neurons, or nodes, with the number of them in the hidden layer chosen as a hyperparameter. As well chosen as a hyperparameter is the number of hidden layers, excluding zero as this would then form a generalized linear model. When the number of hidden layers is greater than one, these ANNs are categorized as deep neural networks.

The nodes from one layer connect to the nodes of the previous layer by an activation function. This function takes as input a regression equation of all the nodes on the previous layer including an intercept, or called the bias parameter in ANNs. Using the formulation in this thesis, starting from the input layer, and going to a single node in the hidden layer is given as:

$$\begin{aligned} a_j &= \sum_{i=1}^{26} \beta_{j,i}^{(1)} X_{t+1-i} + \beta_{j,0}^{(1)} \\ z_j &= h(a_j), \quad \text{for} \quad j = 1, ..., M \end{aligned} \tag{18}$$

In (18), z_j represents the value taken by node j in the hidden layer, and $h()$ is the node's activation function. a_j then forms a linear regression of the predictor values, with $\beta_{j,i}^{(1)}$ being the regression coefficient for input node j of layer (1), and input predictor i. Note that the input layer is considered layer (0), and lastly, $\beta_{j,0}^{(1)}$ is the bias parameter.

An activation function can take any form, with a standard being the hyperbolic tangent function, which outputs a value between negative one and one, and can be seen as the level at which the given node is firing in the network. (18) is repeated for all hidden layers, taking the previous layer as input, until the network arrives to the output layer. Since the focus of this thesis is on time series prediction in a regression framework, the output layer simply has the identity function, $h(a) = a$, as its activation function. For a FFNN with one hidden, the output node, and prediction, is given by:

$$\hat{f}(\mathbf{X}_{t+1}) = \sum_{j=1}^{M} \beta_j^{(2)} z_j + \beta_0^{(2)} \tag{19}$$

In (19) z_j is the value taken at the j^{th} node in the previous hidden layer with M hidden nodes. $\beta_j^{(2)}$ represents the regression coefficient of hidden node j of layer (2), and lastly, $\beta_0^{(2)}$ is the bias parameter of layer (2).

There are four required hyperparameters in a FFNN: number of hidden layers, number of nodes in each hidden layer, activation function used by each layer, and learning rate. The hyperparameters chosen for testing in optimization are motivated by Krauss et al. (2017) and Gu et al. (2020), and given in Table 1. The number of nodes per hidden layer follows the geometric pyramid rule as recommended by Masters (1993), with the first hidden layer having the same number of nodes as predictors. Additionally, a dropout rate regularization is applied to reduce overfitting, where a percetage of hidden nodes are randomly set to zero during each epoch of training, as introduced by Srivastava, Hinton, Krizhevsky, Sutskever, and Salakhutdinov (2014).

As for training the FFNN, the stochastic gradient decent optimizer is used with a momentum parameter included of 0.9, which decreases the chance of getting stuck in a local optimum. As well, an early stopping callback is applied, which stops training the FFNN if the cumulative loss does not improve over ten epochs, which if initiated, decreases runtimes and prevents overfitting.

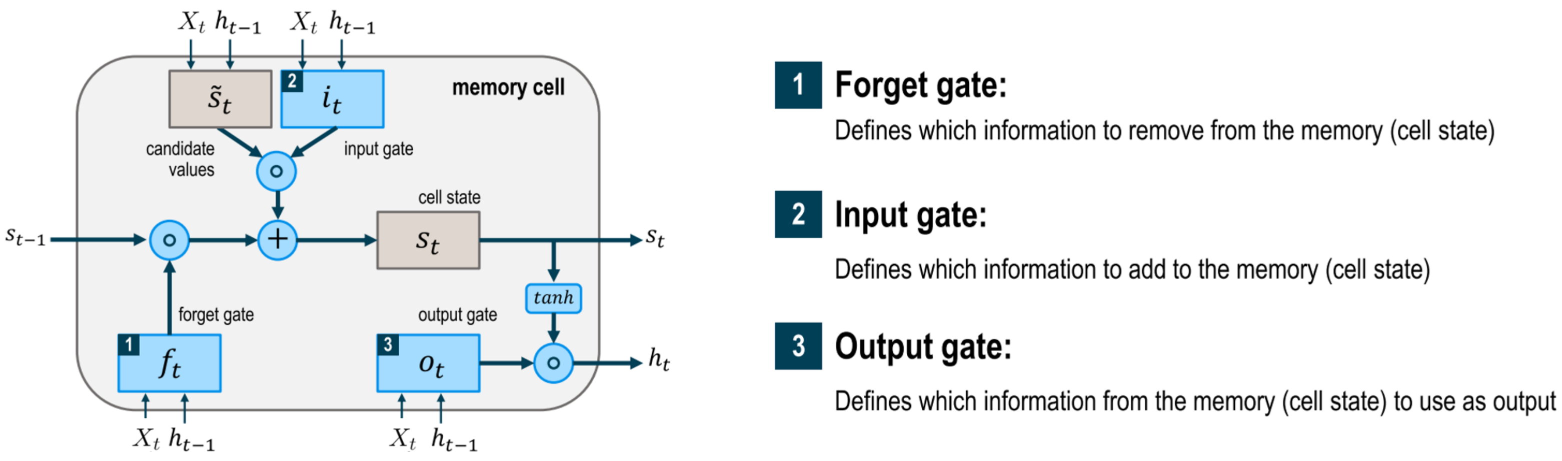
The batch size of a neural network is the number of observations trained at the same time. Keskar, Mudigere, Nocedal, Smelyanskiy, and Tang (2017) recommends a batch size to the power of two between 32 and 512, but mentions this varies largely with different data, and is also highly dependent on the learning rate. Therefore, to analyze batch size in combination with the learning rate, the first training and validation sets are used in an exhaustive gridsearch. It is seen that the batch size play little role in the accuracy of a model, however, a larger batch size substantially reduces runtimes. Therefore, a batch size of 4096 is chosen, as runtimes do not decrease very much with batches larger than this.

5.5.6 Long Short-Term Memory Network

The LSTM network, as introduced by Hochreiter and Schmidhuber (1997), falls into the class of recurrent neural networks whose "underlying topology of interneuronal connections contains at least one cycle" (Medsker & Jain, 2000). LSTM networks are designed specifically to learn long-term dependencies, where there is a gap in observations between the past relevant information and the current prediction. Additionally, through their ability to forget information, they are able to overcome previous issues with recurrent neural networks, i.e., vanishing and exploding gradients when training through backpropagation (Sak, Senior, & Beaufays, 2014).

LSTM networks require that the predictors follow sequences over consecutive points in time. Thus, the feature set used in this thesis works as having a single predictor, with a sequence of 26 weekly observations. Additionally, the LSTM network has an input layer, one or more hidden layers, and an output layer. Both the input and output layers are the same as in the FFNN, however, the difference is contained in the hidden layer(s), as they are made up of a memory cell iteratively operating across the input sequence.

The description of a memory cell follows Fischer and Krauss (2018) and Olah (2015). To begin, two general formulations are; first, that each LSTM hidden layer has a single memory cell with its own parameters, and second, before moving onto the next layer, the memory cell iteratively runs through the entire input sequence. Furthermore, a memory cell has three gates adjusting and maintaining its cell state s_t: a forget gate (f_t), an input gate (i_t) and an output gate (o_t). Figure 2 gives the structure of a memory cell.



X_t h_{t-1} X_t h_{t-1}
$\tilde{s}_t$
2 i_t
memory cell
candidate values
input gate
cell state
s_{t-1}
s_t
s_t
tanh
forget gate
output gate
1 f_t
3 o_t
h_t
X_t h_{t-1} X_t h_{t-1}
1 Forget gate:
Defines which information to remove from the memory (cell state)
2 Input gate:
Defines which information to add to the memory (cell state)
3 Output gate:
Defines which information from the memory (cell state) to use as output

Figure 2: LSTM memory cell structure (Fischer & Krauss, 2018).

At every timestep t in the sequence, each gate is presented with two inputs; the predictor, X_t, at timestamp t in its sequence, and the output of the memory cell in the previous timestep, h_{t-1}. Once these are given, each gate performs a different purpose in a specific order, starting with the forget gate which defines the information that is removed from the current cell state. Following, the input gate defines the information added to the cell state, and lastly, the output gate defines which information from the cell state is used as output. This output is used in two ways, first, as input for the memory cell in the following $t+1$ timestamp, and second, as input to the next layer in the network.

To begin notation of a LSTM layer, all of the variables are defined as follows, with p representing the number of hidden nodes:

- X_t is the input scalar value at timestep t.
- $W_{f,X}$, $W_{\tilde{s},X}$, $W_{i,X}$, and $W_{o,X}$ are $p \times 1$ weight vectors.
- $W_{f,h}$, $W_{\tilde{s},h}$, $W_{i,h}$, and $W_{o,h}$ are $p \times p$ weight matrices.
- b_f, $b_{\tilde{s}}$, b_i, and b_o are $p \times 1$ bias vectors.
- f_t, i_t and o_t are $p \times 1$ vectors for the activation values of the respective gates.
- s_t and $\tilde{s}_t$ are $p \times 1$ vectors for the cell states and candidate values.
- h_t is a $p \times 1$ vector for the output of the memory cell.

The number of hidden nodes p, determines the deminsion of the cell state s_t, as well as the dimension of the output h_t.

The forget gate in a memory cell determines which information should be removed from its previous cell state, s_{t-1}. This is decided based on two inputs; the current predictor in the sequence, X_t, and the output vector of the memory cell from the previous timestep, h_{t-1}. The sigmoid function is applied ranging from 0, completely forget, to 1, completely remember:

$$f_t = \text{sigmoid}(W_{f,X}X_t + W_{f,h}h_{t-1} + b_f) \tag{20}$$

In (20), the input inside the brackets takes the form of a $p \times 1$ vector, and the sigmoid function is applied to each element of the vector separately.

As for the input gate, the LSTM network uses it to determine which information should be added to the cell state, s_t. This is done by first constructing candidate values, $\tilde{s}_t$, by using the hyperbolic tangent function, tanh, ranging from -1 to 1 for what information could be added to each element in the cell state. As well, the activation values, i_t, for the candidate values are found by the sigmoid function, representing what percentage of each of the candidate values should be added to the cell state:

$$\begin{aligned} \tilde{s}_t &= \tanh(W_{\tilde{s},X}X_t + W_{\tilde{s},h}h_{t-1} + b_{\tilde{s}}) \\ i_t &= \text{sigmoid}(W_{i,X}X_t + W_{i,h}h_{t-1} + b_i) \end{aligned} \tag{21}$$

Following, the updated cell state is based on the results of the first two gates, with $\circ$ denoting the Hadamard (elementwise) product:

$$s_t = f_t \circ s_{t-1} + i_t \circ \tilde{s}_t \tag{22}$$

In the output gate, first the sigmoid function is applied to the gate inputs, X_t and h_{t-1}, which determines the percentages of the updated cell state vector, s_t, that are used for the memory cell's output at time t. Additionally, the tanh function is applied to the updated cell state to ensure values between -1 and 1:

$$\begin{aligned} o_t &= \text{sigmoid}(W_{o,X}X_t + W_{o,h}h_{t-1} + b_o) \\ h_t &= o_t \circ \tanh(s_t) \end{aligned} \tag{23}$$

The output gate in (23) is what enables a LSTM network to carry over long term dependencies. This is done by the output gate still taking as input the previous iterations output, h_{t-1}. Doing so enables the carry over of crucial information throughout every iteration in the memory cell, without having to run it through the forget gate or input gate, in such a way that it is still taken as input after the updates are applied to current iterations cell state, s_t.

Finally, once the entire predictor sequence has been run through, the final LSTM hidden layer is connected to the output layer by using the identity function, $h(a) = a$, as its activation function.

$$\hat{f}(\mathbf{X}_{t+1}) = \sum_{i=1}^{p} \beta_i h_{i,t} + \beta_0 \tag{24}$$

In (24) $h_{i,t}$ is the i^{th} hidden node output from the memory cell at time t after the entire sequence of 26 observations has been iterated through. Therefore, only the last cell in the LSTM network feeds input into the output layer. Lastly, β_i is the regression coefficient of the i^{th} hidden node from the 26th iteration, and β_0 is the bias parameter.

The hyperparameters tested are very similar to the ones used for FFNN. A slight difference is that only the hyperbolic tangent activation function is used. This is because it is the only activation function that gives LSTM the ability to make use of parallel computing through CUDA, which results in upwards of 1000% faster runtimes. For training the LSTM, the stochastic gradient decent optimizer is again used with a momentum parameter of 0.9, and as well an early stopping callback of ten epochs. The same exhaustive grid search is applied for the learning rate and batch size as FFNN with similar findings, such that a batch size of 4096 is chosen. The hyperparameters tested in optimization are motivated by Fischer and Krauss (2018) and Bao, Yue, and Rao (2017), and are given in Table 1.

5.5.7 Multi-Frequency Combination Method

Temporal Aggregation:

MFCM is a novel methodology introduced in this paper, built off of applying overlapping temporal aggregation to a higher frequency time series and then combining forecasts at different horizons. The first step in MFCM is to use a higher frequency time series than the original, where the higher frequency is divisible by the original frequency. For example, quarterly to monthly or daily, or as used in this thesis, weekly to daily. Consider the original time series has a frequency of n, and the new time series uses a frequency of $p > n$.

Next, temporal aggregation is applied to the higher frequency time series until it reaches the lower frequency, specifically $p/n - 1$ times. Temporal aggregation works by iteratively reducing the frequency of a time series by one unit at every iteration, creating a new time series at each cycle. For example, with daily being the higher frequency, the first iteration generates a new time series with a frequency of every two days, and the second iteration generates a new time series with a frequency of every three days. This is repeated $p/n - 1$ times, as the resulting lowest frequency found by temporal aggregation is then the original, n frequency time series.

Additionally, overlapping temporal aggregation is applied, which enables every frequency, between the lowest and highest frequencies, to be used. On the other hand, non-overlapping temporal aggregation requires that each frequency used must follow the divisibility rule as described at the end of the literature review section 2.2.

Specifically, overlapping temporal aggregation means that each frequency in the temporal aggregation step generates the same number of time series as the associated frequency. However, each of these time series with the same frequency begins one observation later than in the highest frequency time series, similar to a moving window. This is required because when constructing a forecast at time t, each frequency found through temporal aggregation must use the most recent observation of the time series at time t. Therefore, this ensures that there are no holes in the lower frequency series, such that they all have observations at the same time as every observation of the highest frequency. The constraint that p is divisible by n ensures this is possible. An example of this is with the frequency of every three days, three time series are created all with the same three day frequency starting at day zero, day one, and day two.

A benefit to this overlapping form of temporal aggregation is that all of the non-highest frequency time series now have multiple time series, each of the same frequency, to use when training a model. For example, for the lowest, original frequency of every five days, it now has five separate time series which are all used to train a model rather than just a single time series. Although this is advantageous, the overall benefit of temporal aggregation falls in line with the identification of series patterns and characteristics, which are unique to each frequency of the time series.

This is further emphasized by the features used in MFCM, where the standard MFCM formulation enforces the same set of predictors on each of the different frequencies, as the original time series. In the example in this thesis, the original, lower frequency time series uses the past 26 lags of itself for prediction. Therefore, because of the varying frequencies, each of these lags goes back a different distance in time. For example, the original time series goes back 26 weeks, but the daily time series goes back 26 trading days, which captures dependence structures over different lengths of time. This frequency matching feature creation, along with the different frequencies of time series that the models are trained on, is what conceivably enables MFCM to identify series patterns and characteristics unseen to a time series of a single frequency.

For example, considering two frequencies of the same time series, where the second one is twice as frequent as the first. Then, when feeding the higher frequency time series into a complex model, such as a neural network, it observes all the same time points in the lower frequency time series, with the addition of an extra time point in between each observation. Lastly, the neural network then uses the datapoints in between the lower frequency time series to derive additional causality, not seen by a model trained on only the lower frequency time series. Through this example, it is apparent that MFCM is not found from a statistical basis, but rather from an ad hoc machine learning perspective.

Forecasting:

MFCM is then finalized by each time series forecasting the same horizon as their associated frequency, as traditionally done in time series estimation, which results in forecasts through the future life of the time series until the desired horizon of the original frequency. Then, by applying an activation function, these forecasts are aggregated together to predict the time series observation of the original frequency's horizon:

$$\hat{f}(\mathbf{X}_{t+1}) = h\Big(\hat{f}_1(\mathbf{X}_{t+\frac{1}{p}}),\ \hat{f}_2(\mathbf{X}_{t+\frac{2}{p}}),\ \ldots\ ,\ \hat{f}_{p-1}(\mathbf{X}_{t+\frac{p-1}{p}}),\ \hat{f}_p(\mathbf{X}_{t+1})\Big) \tag{25}$$

In (25) $h()$ is the activation function, which can take on any predictive model, such as a LR or a LSTM network, and $t+1$ is the longest horizon given by the original, lowest frequency n. Then $\hat{f}_i(\mathbf{X}_{t+\frac{i}{p}})$ is the frequency forecasting model conducted using model i of frequency $p-i+1$ at horizon i/p.

As an application in this thesis, each of the frequency forecasting models, $\hat{f}_i(\mathbf{X}_{t+\frac{i}{5}})$ for $i \in \{1,2,3,4,5\}$, uses the one model of the six covered in sections 5.5.1 to 5.5.6 that results in the lowest MSE in the validation sets. Despite the same type of model for each frequency being used, each undergoes their own hyperparameter optimization process through the HYPERBAND algorithm. This supports a better fit of the model to each frequencies specific statistical distribution. Additionally, two forms of MFCM are applied, with the first using a LR activation function, called linear MFCM:

$$\hat{f}(\mathbf{X}_{t+1}) = \beta_0 + \beta_1\hat{f}_1(\mathbf{X}_{t+\frac{1}{5}}) + \beta_2\hat{f}_2(\mathbf{X}_{t+\frac{2}{5}}) + \beta_3\hat{f}_3(\mathbf{X}_{t+\frac{3}{5}}) + \beta_4\hat{f}_4(\mathbf{X}_{t+\frac{4}{5}}) + \beta_5\hat{f}_5(\mathbf{X}_{t+1}) + \varepsilon_{t+1} \tag{26}$$

The second form of MFCM uses the activation function which is the same model as the one used for the frequency forecasting models, called optimal MFCM. If hyperparameters are used in the activation function, they are as well tuned in the validation set. Figure 3 shows a flow chart model of the standard forecasting approach and the MFCM forecasting approach, both used in this thesis.

Training:

The procedure for training MFCM with an activation function which requires hyperparameter tuning follows four steps: (1) Train multi-frequency models on the training set, and use the validation set to optimize their hyperparameters; (2) Train optimal multi-frequency models on the training and validation sets, then construct forecasts for them over both the training and validation sets; (3) Train the activation function on multi-frequency models forecasts constructed in the training set, and use multi-frequency models forecasts of the validation set to optimize the hyperparameters of the activation function; and (4) Train the optimal activation function over the multi-frequency models forecasts of both the training and validation sets. If no hyperparameter optimization is required for the activation function, step (3) is skipped. As well, the robust scaling using equation (1) is applied to both the starting training data, and the forecasts which the activation function is trained on.

After these four steps have been accomplished, out-of-sample testing is done by first constructing forecasts from the already fit multi-frequency models, each using data starting from the most recent observation. Then take these forecasts as inputs into the already fit activation function, and create the forecast for the desired endogenous variable.

It can be argued that validation of the activation function is only done half out-of-sample, since the forecasts from the multi-frequency models that it is validating on are trained using both the training and validation set. The reason for this, is that now the inputs to the activation function, being the forecasts from the multi-frequency models, are being constructed using the same model fits which will be used in out-of-sample testing. Additionally, utilizing in-sample accuracy of the multi-frequency models as inputs in the activation function when validating it out-of-sample, gives a higher possibility of the multi-frequency forecasts being accurate enough that the activation function can find causality between them and the endogenous variable.

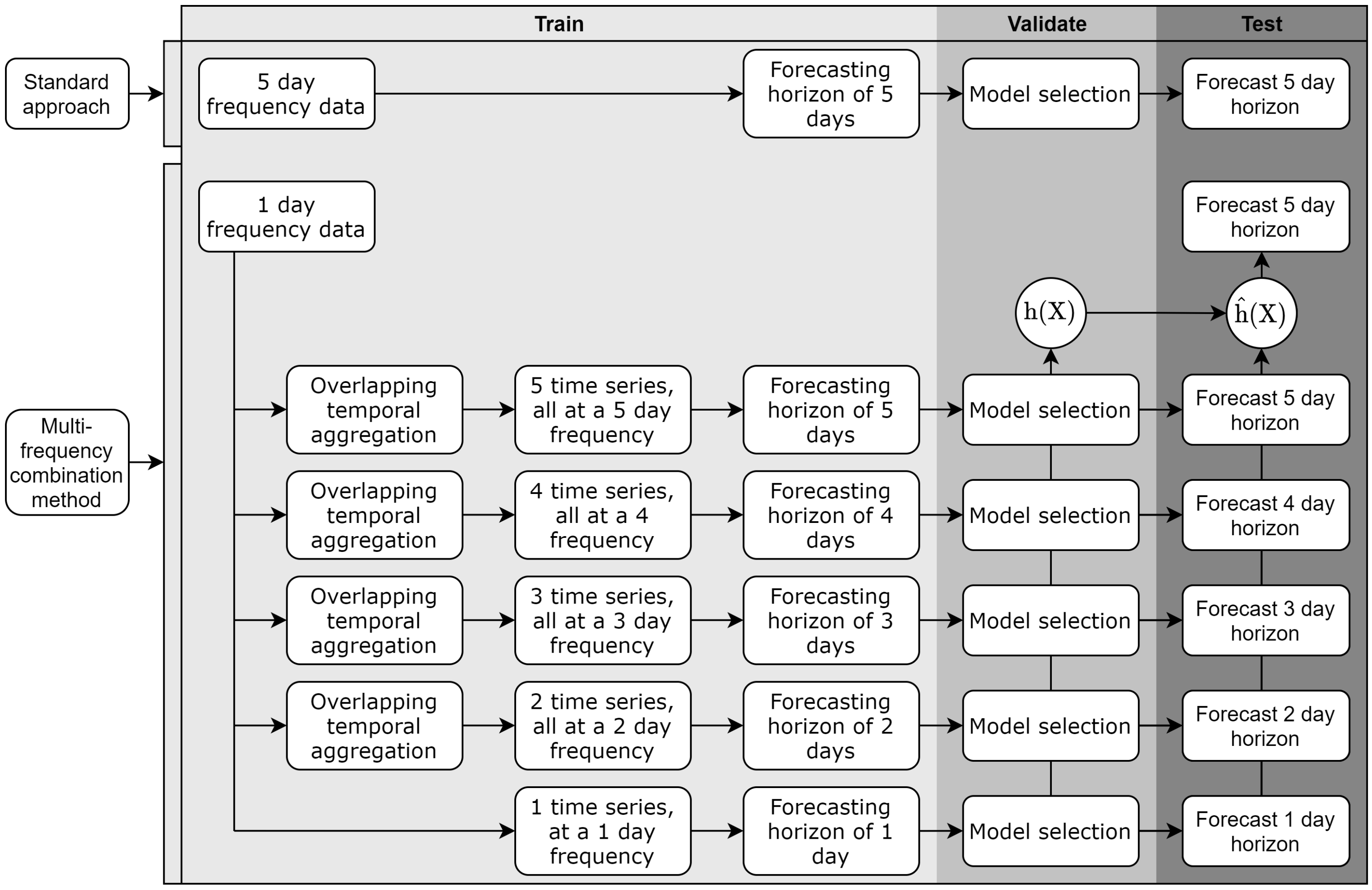

Train
Validate
Test
Standard approach
5 day frequency data
Forecasting horizon of 5 days
Model selection
Forecast 5 day horizon
Multi-frequency combination method
1 day frequency data
Forecast 5 day horizon
h(X)
ĥ(X)
Overlapping temporal aggregation
5 time series, all at a 5 day frequency
Forecasting horizon of 5 days
Model selection
Forecast 5 day horizon
Overlapping temporal aggregation
4 time series, all at a 4 frequency
Forecasting horizon of 4 days
Model selection
Forecast 4 day horizon
Overlapping temporal aggregation
3 time series, all at a 3 day frequency
Forecasting horizon of 3 days
Model selection
Forecast 3 day horizon
Overlapping temporal aggregation
2 time series, all at a 2 day frequency
Forecasting horizon of 2 days
Model selection
Forecast 2 day horizon
1 time series, at a 1 day frequency
Forecasting horizon of 1 day
Model selection
Forecast 1 day horizon

Figure 3: Standard forecasting versus MFCM forecasting.

Table 1: Hyperparameters tested for each model in the HYPERBAND algorithm.

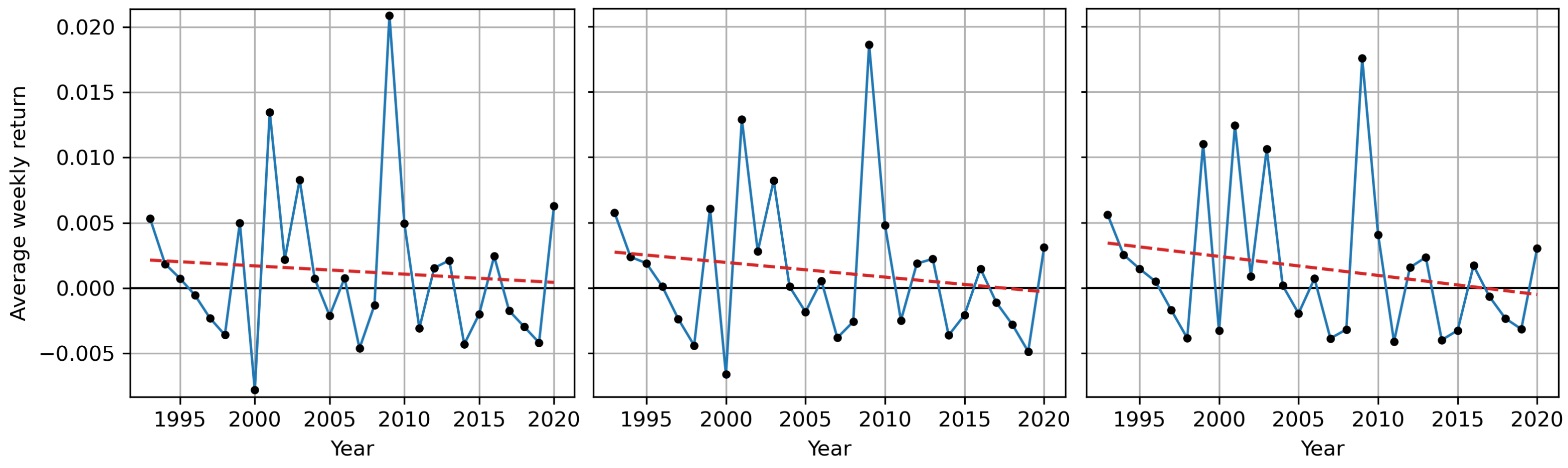
Model	Hyperparameter	Values Tested
LR	Alpha regularization	{0.1, 0.25, 0.5, 0.75, 1}
	L1 weight	{0.1, 0.25, 0.5, 0.75, 0.9}
KNN	Nearest neighbors	{5, 10, 20 ,40, 60}
	Prediction weights	{uniform, distance}
RF	Number of trees	{100, 300, 600}
	Max depth	{2, 3, 4, 6, 9, 11, 14}
	Feature subsample by level	{0.2, 0.5}
	Data subsample	{0.3, 0.6}
	L2 regularization	{0.001, 0.5}
GBT	Number of trees	{200, 500}
	Max depth	{2, 3, 4, 6, 8, 10}
	Feature subsample by level	{0.2, 0.5}
	Data subsample	{0.3, 0.6}
	L2 regularization	{0.001, 0.5}
	Learning rate	{0.07, 0.3}
FFNN	Number of hidden layers	{1, 2, 3, 4}
	Number of nodes per hidden layer	{[26, 12, 6, 3], [4, 4, 4, 4]}
	Epochs	{200, 500}
	Learning rate	{0.0000001, 0.00001, 0.0001}
	Batch	4096
	Activation function	{sigmoid, ReLU}
	Hidden node dropout rate	{0.3, 0.6}
	L2 regularization	0.0001
	Early stopping	10
LSTM	Number of hidden layers	{1, 2, 3}
	Number of nodes per hidden layer	{[26, 12, 6], [4, 4, 4]}
	Epochs	{100, 400}
	Learning rate	{0.0000001, 0.00001, 0.0001}
	Batch	4096
	Activation function	tanh
	Hidden node dropout rate	{0.3, 0.6}
	L2 regularization	0.0001
	Early stopping	10

6 Results

6.1 Anomaly Analysis

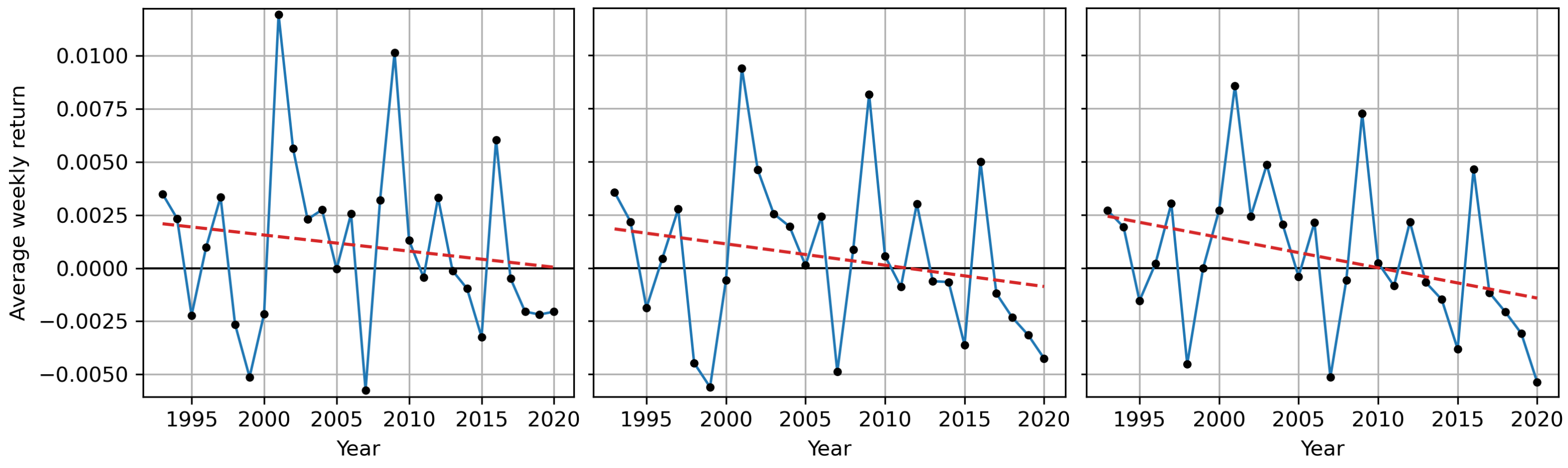
The first step for analysing the null hypothesis of this thesis is to find an asset pricing anomaly in the Russell 3000 index. Nine anomalies are tested in total; 1 month, 6 month and 12 month holding periods, for each of the size, value and momentum anomalies. Then, the anomaly which is most prominent in the data in recent years is chosen for the remainder of the analysis.

The anomalies are calculated, as described in section 5.1, Finding an Anomaly, and presented with their average weekly returns, for each year from 1993 to 2020, in Figures 4 to 6. Also included are their regression lines of time on yearly returns, which shows the trend of the anomaly since 1993.



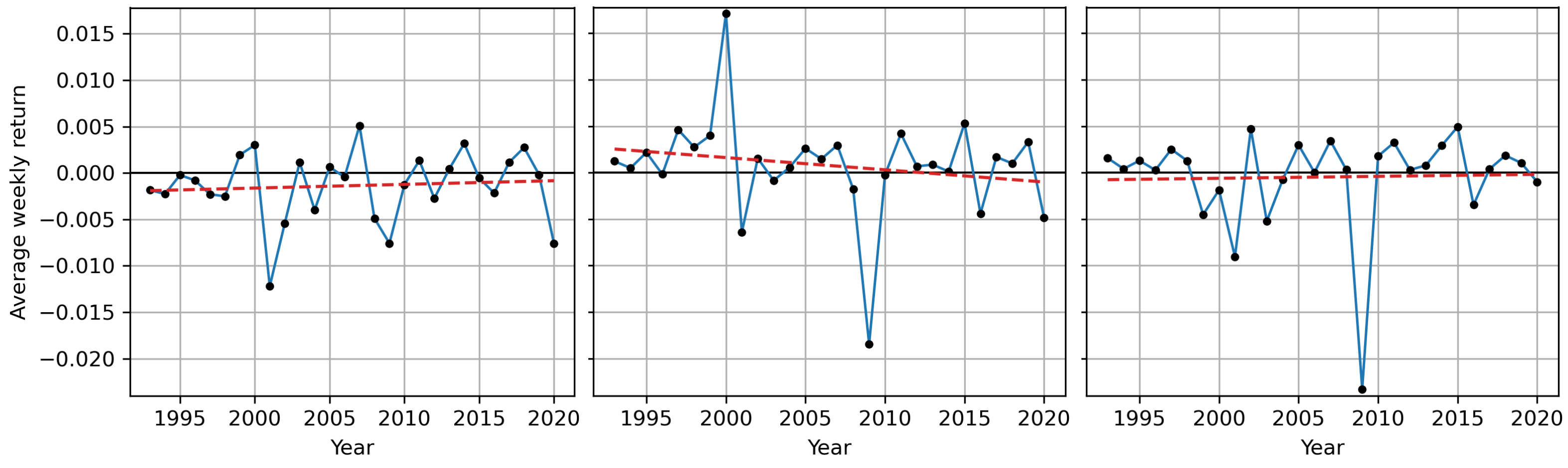
(a) 1 month (b) 6 month (c) 12 month

Figure 4: Size anomaly from 1993 to 2020 for 1, 6 and 12 month holding periods, formed through decile portfolios of the Russell 3000 index.



(a) 1 month (b) 6 month (c) 12 month

Figure 5: Value anomaly from 1993 to 2020 for 1, 6 and 12 month holding periods, formed through decile portfolios of the Russell 3000 index.



(a) 1 month (b) 6 month (c) 12 month

Figure 6: Momentum anomaly from 1993 to 2020 for 1, 6 and 12 month holding periods, formed through decile portfolios of the Russell 3000 index.

To begin with, the size anomaly is in effect when the values in Figure 4 are positive, meaning the average weekly return of the small asset's decile portfolio, is larger than the average weekly return of the large asset's decile portfolio. However, in recent years there is no consistent sign of the size anomaly being present for any of the 1, 6 or 12 month holding periods. By the regression lines in each of the graphs, it appears that the effect of the size anomaly has been decreasing over time.

Additionally, the value anomaly in Figure 5, which is the difference between the value and the growth decile portfolios, shows a similar finding. For its 12 month holding period, 11 out of the 14 years from 1993 until 2006, all show the value anomaly in effect. Yet, in recent years there is no consistent sign of the value anomaly being present; in fact, 7 out of the 8 recent years show that growth firms have been achieving higher average returns than value firms. This is displayed by the decreasing regression lines in all three graphs.

The low signs of the value anomaly may be related to Asness et al. (2013) finding; "Our broader set of portfolios generates much larger cross-sectional dispersion in average returns than those from U.S. stocks only". By broader set of portfolios, they are referring to asset portfolios from European and Asian markets, stating that the U.S. market showed the value anomaly the least out of all markets. This is as well seen in their Table I, Panel A. Additionally, their sample period goes up to 2011. Where it is seen in all three graphs in Figure 5, the years following 2011 show the value anomaly much less than the years before 2011.

As for the momentum anomaly in Figure 6, which is the difference between the high momentum and the low momentum decile portfolios, the 1 month holding period exhibits a strong reversal pattern throughout the dataset. This is as expected from standard momentum literature, and may be due to microstructure or liquidity issues (Asness et al., 2013). However, the 6 and 12 month holding periods both show clear signs of the momentum anomaly, especially the 12 month holding period over the past decade. Notably, for the 12 month holding period, the regression line is consistently negative. This is to do with the large outlier of 2009 following the financial crisis, and if this observation is removed the regression line becomes positive. Specifically, the mean of the momentum anomaly with a 12 month holding period goes from -0.0005 to 0.0004 with and without the 2009 outlier respectively.

To analyze how all nine anomalies have been evolving since 1993, regressions are made of time on their returns. Additionally, to increase significance of the regression coefficients, weeks are regressed on weekly returns, instead of years on yearly returns. The results of these regressions are given in Table 2. As well included, are three mean values of the anomaly's weekly returns: over the entire dataset, up to 2010, and after 2010. The split at 2010 is applied to isolate the last decade which may be used in the final testing set when forming optimal portfolios.

Considering the first three rows of Table 2 for the size anomaly to begin with, the striking result that was first seen in Figure 4 is again realized, such that the effect of the size anomaly has been decreasing over the past 28 years. This is first seen by the last three columns containing the average weekly returns. Here, for each of the 1, 6 or 12 month holding periods, the average return from 1993 to 2010 is positive, while the average return from 2011 to 2020 is negative. The minimum difference between the average weekly returns of these two time periods is 29 basis points for the one month size anomaly, displaying the magnitude of this change over the past decade.

Table 2: Regressions of weeks on weekly, close-to-close, returns for each of the nine anomalies. Each p-value column refers to the coefficient to its left. The last three columns contain the average weekly returns of each anomaly over different time intervals.

Anomaly		slope	p-value	intercept	p-value	Average weekly return		
						all	1993-2010	2011-2020
Size	1 month	-0.000001	0.431	0.0023	0.118	0.0013	0.0023	-0.0006
	6 month	-0.000002	0.145	0.0029	0.029	0.0012	0.0024	-0.0008
	12 month	-0.000003	0.057	0.0036	0.005	0.0015	0.0028	-0.0009
Value	1 month	-0.000002	0.328	0.0022	0.096	0.0011	0.0018	-0.0002
	6 month	-0.000002	0.164	0.0019	0.109	0.0005	0.0012	-0.0009
	12 month	-0.000003	0.045	0.0025	0.031	0.0005	0.0014	-0.0012
Momentum	1 month	0.000001	0.615	-0.0020	0.184	-0.0014	-0.0019	-0.0004
	6 month	-0.000002	0.169	0.0025	0.088	0.0008	0.0008	0.0008
	12 month	0.000000	0.804	-0.0008	0.589	-0.0005	-0.0013	0.0011

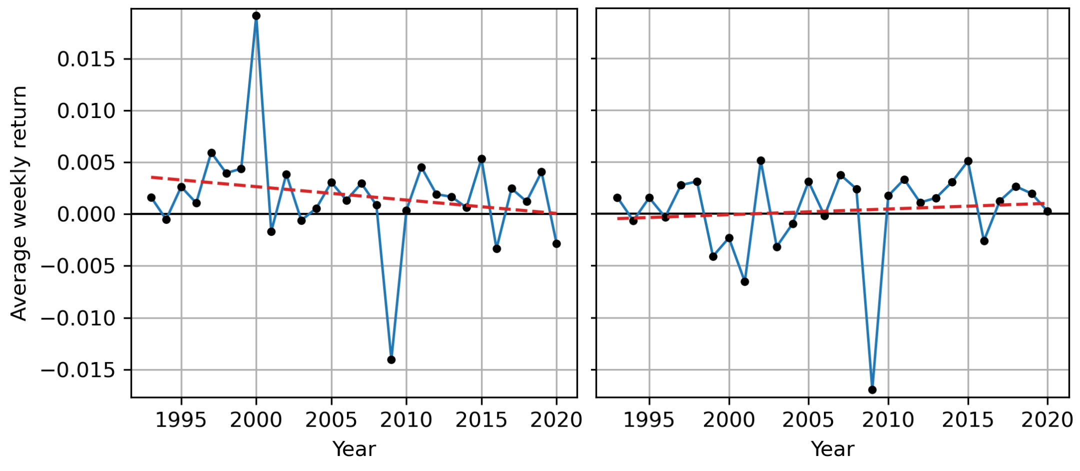
Additionally, all three slope coefficients are negative, with the 12 month holding period having the steepest negative slope and a significance at the 10% level. As well, all three intercepts, representing the anomaly effect at the start of the testing period, are positive, with the 6 month holding period significant at the 5% level, and the 12 month holding period significant at the 1% level.

A similar finding holds true for the value anomaly, where all average weekly returns are positive for the period from 1993 to 2010, and negative for the period from 2011 to 2020. Once again, these differences are quite large with the one month holding period having the minimum difference of 20 basis points in average weekly returns. Additionally, the three slope coefficients are negative, with the 12 month holding period being the steepest negative and significant at the 5% level. As well, all three intercepts are positive, with the 1 month holding period significant at the 10% level, and the 12 month holding period significant at the 5% level.

As for the momentum anomaly, the 1 month holding period exhibits a reversal pattern, with all three time periods of average weekly returns being negative. The only significant coefficient for momentum is the 6 month holding period intercept at the 10% level, showing little evidence of variation for the momentum anomaly over time. Additionally, the 6 month holding period has a consistent positive average weekly return throughout the testing set. As for the 12 month holding period, its average weekly return is negative from 1993 to 2010, but positive and greater than the 6 month holding period from 2011 to 2020.

It is seen by the final column in Table 2 that the 6 month and 12 month holding periods for the momentum anomaly are the most present out of the nine anomalies over the past decade. Therefore, one of these anomalies will be selected for the remainder of the analysis. However, since the optimal portfolios in this thesis are constructed using intraweek returns, it is crucial that these anomalies are as well present for intraweek returns over the final testing period. To analyze this, two more graphs are presented in Figure 7, showing the average intraweek returns of both the 6 month and 12 month holding periods for the momentum anomaly.

It is seen in Figure 7 that the 12 month holding period for the momentum anomaly exists in every year of 2010 to 2020, excluding 2016. Therefore, since the momentum anomaly with a 12 month holding period is the most prominent of the nine anomalies tested in recent years, for both weekly and intraweek returns, it is used for the remainder of the analysis in this thesis.



(a) 6 month (b) 12 month

Figure 7: Momentum anomaly using intraweek returns from 1993 to 2020 for 6 and 12 month holding periods, formed through decile portfolios of the Russell 3000 index.

Specifically, the pseudo out-of-sample testing set for the optimal portfolios takes place over the eight most recent years from the start of 2013 to the end of 2020. The year 2016 may not have the anomaly present, but it is included to provide practical relevance when answering the main research question, such that in practice there may be certain time periods when the anomaly does not hold.

Additionally, by beginning the testing set in 2013, this leaves the three years prior for the first validation set; {2010,2011,2012}, which doesn't involve validating the models on the year 2009. This is because as seen in Figures 6 and 7, 2009 exhibited a strong reversal pattern in the momentum anomaly, when following the financial crisis starting in September 2008. Lastly, since the 12 month holding period is used, models are retrained every 12 months after the 600 assets are updated in the two decile momentum portfolios.

6.2 Portfolio Analysis

In total, 64 portfolios are tested. This includes eight models, with both linear MFCM and optimal MFCM. Then, for each model there are eight separate portfolios; {5,10,20,40} portfolio sizes for both the anomaly assets category and the all assets category. For the MFCM models, it is found that the LSTM network achieves the lowest average MSE across the validation sets, hence it is used for both the frequency forecasting models in MFCM, and the activation function model in optimal MFCM.

All model runtimes and optimal hyperparameters for each year, and for anomaly and all asset classes are given in Appendix A.1. A few brief notes on these results is that MFCM has substantially the largest runtimes, with an average over both the validation and final model runtime of 2 hours and 29 minutes. KNN is as well quite slow taking an average of 1 hour and 9 minutes. The remaining models from slowest to fastest are: LSTM, FFNN, GBT, RF, LR.

For optimal hyperparameters a few interesting patterns emerge. First, LR tends to select a rather low amount of regularization, with the weight in elastic net moving closer to a L2 penalty. Second, all 14 KNN models use a uniform distance metric with the maximum available 60 nearest neighbors, showing its preference for reducing in-sample bias for less out-of-sample variance. Third, every GBT model selects the shallowest possible trees at a max depth of 2. This was as well found to be the case by Krauss et al. (2017). Additionally, every GBT model selects the lowest learning rate of 0.07. Fourth, 15 out of the 16 models for LSTM choose a node setup with the most amount of layers, given by 3. FFNN on the other hand doesn't seem to have a consistent preference.

All 64 portfolios are tested over the eight years of out-of-sample data as described in section 5, and their results are given in Table 3 to Table 10. Full tables with additional portfolio metrics are given in Appendix A.2.

Table 3: Elastic net LR out-of-sample portfolio results. All values throughout Table 3 to Table 10 are stated in standard decimal form, and the number after the model abbreviation in the second row of the last eight columns represents the portfolio's number of assets. The mean weekly return is taken over the entire out-of-sample testing set, and the p-value of White's reality check (WRC) test is with the null hypothesis of weekly returns being greater than zero. Lastly, the annualized return is the average of the eight cumulative yearly returns over the out-of-sample testing set.

	Anomaly assets				All assets			
Metric	LR5	LR10	LR20	LR40	LR5	LR10	LR20	LR40
Mean weekly return	0.0034	0.0052	0.0034	0.0032	0.0001	-0.0011	-0.0003	0.0006
WRC p-value	0.4337	0.0918	0.0995	0.0566	0.7419	0.8154	0.7233	0.5542
Annualized return	0.02	0.2	0.136	0.143	-0.127	-0.133	-0.079	-0.021

Table 4: KNN regression out-of-sample portfolio results.

	Anomaly assets				All assets			
Metric	KNN5	KNN10	KNN20	KNN40	KNN5	KNN10	KNN20	KNN40
Mean weekly return	0.0011	0.0014	0.0025	0.0031	-0.001	-0.0018	-0.0005	0
WRC p-value	0.5093	0.3863	0.1253	0.0345	0.8684	0.8907	0.754	0.6423
Annualized return	-0.008	0.029	0.101	0.146	-0.21	-0.183	-0.087	-0.04

Table 5: RF out-of-sample portfolio results.

	Anomaly assets				All assets			
Metric	RF5	RF10	RF20	RF40	RF5	RF10	RF20	RF40
Mean weekly return	0.0002	0.0015	0.0019	0.0029	-0.0045	-0.0039	-0.0004	-0
WRC p-value	0.6556	0.3843	0.1948	0.0427	0.9714	0.9667	0.7121	0.6363
Annualized return	-0.063	0.028	0.07	0.134	-0.292	-0.235	-0.069	-0.038

Table 6: GBT out-of-sample portfolio results.

	Anomaly assets				All assets			
Metric	GBT5	GBT10	GBT20	GBT40	GBT5	GBT10	GBT20	GBT40
Mean weekly return	0.0014	0.0019	0.0026	0.0029	-0.0038	-0.0027	-0.0015	-0.0003
WRC p-value	0.5011	0.2952	0.1155	0.0546	0.9621	0.9312	0.863	0.6947
Annualized return	-0.009	0.051	0.107	0.13	-0.317	-0.217	-0.134	-0.057

Table 7: FFNN out-of-sample portfolio results.

	Anomaly assets				All assets			
Metric	FFNN5	FFNN10	FFNN20	FFNN40	FFNN5	FFNN10	FFNN20	FFNN40
Mean weekly return	-0.0003	0.0015	0.0025	0.0027	0.0007	0.0013	0.0007	0.0016
WRC p-value	0.7218	0.3174	0.1021	0.0693	0.6924	0.5314	0.5354	0.2934
Annualized return	-0.073	0.042	0.105	0.118	-0.096	-0.017	-0.019	0.045

Table 8: LSTM network out-of-sample portfolio results.

	Anomaly assets				All assets			
Metric	LSTM5	LSTM10	LSTM20	LSTM40	LSTM5	LSTM10	LSTM20	LSTM40
Mean weekly return	0.0052	0.0037	0.0033	0.0034	0.0037	0.0031	0.0025	0.0025
WRC p-value	0.0453	0.0519	0.0478	0.0176	0.1093	0.0704	0.0849	0.0542
Annualized return	0.227	0.166	0.151	0.165	0.146	0.135	0.108	0.113

Table 9: MFCM method with linear activation function out-of-sample portfolio results.

	Anomaly assets				All assets			
Metric	$\text{MFCM}^{\text{L}}5$	$\text{MFCM}^{\text{L}}10$	$\text{MFCM}^{\text{L}}20$	$\text{MFCM}^{\text{L}}40$	$\text{MFCM}^{\text{L}}5$	$\text{MFCM}^{\text{L}}10$	$\text{MFCM}^{\text{L}}20$	$\text{MFCM}^{\text{L}}40$
Mean weekly return	0.008	0.005	0.0031	0.0032	0.0025	0.0033	0.0019	0.0024
WRC p-value	0.0001	0.003	0.0162	0.0096	0.2226	0.0579	0.1575	0.0616
Annualized return	0.443	0.259	0.153	0.158	0.084	0.151	0.077	0.11

Table 10: MFCM method with optimal activation function out-of-sample portfolio results.

	Anomaly assets				All assets			
Metric	$\text{MFCM}^{\text{O}}5$	$\text{MFCM}^{\text{O}}10$	$\text{MFCM}^{\text{O}}20$	$\text{MFCM}^{\text{O}}40$	$\text{MFCM}^{\text{O}}5$	$\text{MFCM}^{\text{O}}10$	$\text{MFCM}^{\text{O}}20$	$\text{MFCM}^{\text{O}}40$
Mean weekly return	0.0043	0.004	0.0036	0.0034	0.005	0.0048	0.004	0.0032
WRC p-value	0.1611	0.0448	0.0329	0.0271	0.0049	0.001	0.0025	0.0089
Annualized return	0.149	0.179	0.165	0.159	0.254	0.253	0.206	0.161

For a baseline comparison of the portfolios, the naive 1/N portfolio of intra week returns has a mean weekly return of 0.0026 and 0.0014 for the anomaly assets and all assets categories respectively. This portfolio is proven to be surprisingly difficult to consistently outperform in out-of-sample data by DeMiguel et al. (2009).

To begin the analysis, the general performance of the portfolios is considered from Tables 3 to 10. First, looking at the all assets category, elastic net LR, KNN regression, RF and GBT models, have 14 out of their 16 portfolio's mean weekly return negative, with the remaining two only positive by a small margin. These reside far below the naive 1/N return, showing that having the low momentum assets in the all assets category can greatly hinder the returns of portfolios constructed from this category. The portfolios constructed in the anomaly assets category do give higher mean weekly returns for these first four models, but still only half of them are able to achieve a higher mean weekly return than the respective naive 1/N portfolio.

For the neural networks and MFCM models, 31 out of their 32 portfolios all have positive mean weekly returns for both the anomaly assets and the all assets categories, with the negative value coming from FFNN in the anomaly category with 5 assets. However, for FFNN, only 2 out of its 8 portfolios achieve higher mean weekly returns than the respective naive 1/N portfolios. This comparison is starkly different for LSTM network, MFCM^L and MFCM^O, which have all 24 of their portfolios achieving a higher mean weekly return than the respective naive 1/N portfolios.

The highest mean weekly return is awarded to MFCM^L with an asset size of 5, which gives a 0.8% mean weekly return. This results in an annualized geometric return of 44.3%, and an increase in 18.8 times the initial portfolio value over the testing set, while excluding transaction costs. However, the MFCM^O, which uses a LSTM network as its activation function, seems to be the most consistent, with all eight of its portfolio's mean weekly returns falling in between 0.32% to 0.48%.

As for if these returns are significantly greater than zero, Table 11 presents how many of the 4 portfolios in each category give a significant result through White's reality check.

Table 11: How many of the 4 portfolios for each model over the two categories give significant p-values for White's reality check. The asterisks represent the largest significance level in the group, with *, **, and *** representing 10%, 5% and 1% significance levels respectively.

Asset category	LR	KNN	RF	GBT	FFNN	LSTM	MFCM^L	MFCM^O
Anomaly assets	3*	1**	1**	1*	1*	4*	4**	3**
All assets	0	0	0	0	0	3*	2*	4***

From Table 11 it is seen that elastic net LR, KNN regression, RF, GBT, and FFNN all don't have any of their four portfolios with significantly positive returns in the all assets category. Elastic net LR does the best of these first five models, with 3 out of its 4 portfolios in the anomaly assets category achieving significantly positive returns, at a maximum 10% significance level. However, the remaining three models: LSTM network, MFCM^L and MFCM^O, produce far more consistently positive returns, with 20 out of their 24 portfolios being significant. Notably, the best performing model is MFCM^O, which produces all four of its portfolios in the all assets category with a positive return at the 1% significance level.

Applying the third form of White's reality check, as described in section 5.4.1, gives a p-value of 12.6%. This result fails to reject the null hypothesis that the strategy with the highest mean return among the 64 portfolios, being MFCM^L with 5 assets, is able to outperform the benchmark returns of zero, while accounting for data snooping for all other 63 portfolios in each simulation. However, this third form of White's reality check only enables testing on the single model with the highest returns, being MFCM^L. Additionally, there are 27 other portfolios that it is testing against which are all significantly non-random by the individual White's reality tests, with 7 of them significant at the 1% level. Then, since these portfolios are already proven to be non-random, it is reasonable that the third form of White's reality check is substantially difficult test to pass, when it must outperform not only itself, but the 27 other significant portfolios.

To answer the main question of this thesis, the null hypothesis graph of Figure 1 is recreated with the out-of-sample returns, and is presented in Figure 8. As well, the weekly hit rate of each portfolio, given by the percentage of weeks which have a positive return, and a White's reality check test comparing if the anomaly assets portfolio has higher weekly returns than the all assets portfolio are given in Tables 12 and 13.

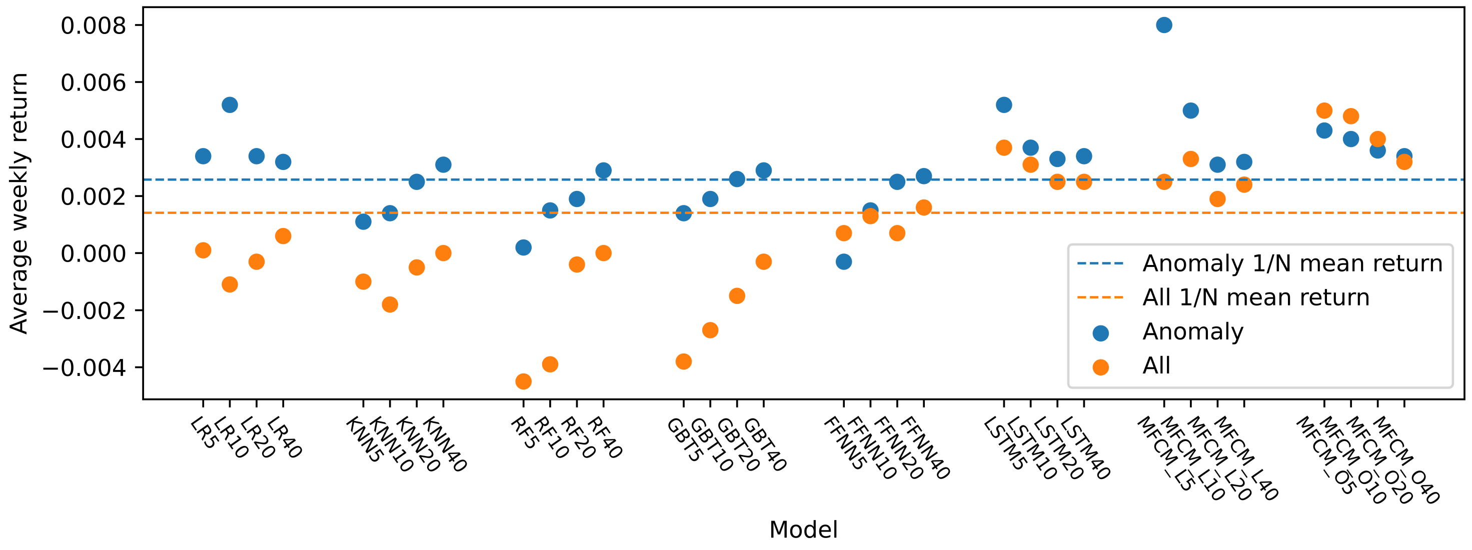


Figure 8: Null hypothesis graph with real world data.

Table 12: Weekly hit rates, given by the percentage of weeks which have a positive return, for the all and anomaly portfolios. As well, White's reality check (WRC) p-value, testing if the anomaly assets portfolio has higher weekly returns than the all assets portfolio. For models: LR, KNN, RF and GBT.

	LR				KNN				RF				GBT			
Metric	5	10	20	40	5	10	20	40	5	10	20	40	5	10	20	40
All hit rate	48.9	50.8	52.3	55.2	51.1	52	54.4	58.3	50.4	52.5	53.2	56.1	49.4	51.1	54.9	55.9
Anomaly hit rate	46	45.8	46.5	51.8	46.3	43.4	47.7	49.4	44.6	46	50.6	49.6	44.6	45.6	47.2	50.1
WRC p-value	0.2	0.02	0.02	0.04	0.03	0.01	0.02	0.01	0.02	0	0.05	0.02	0.01	0	0	0.01

Table 13: Weekly hit rates, given by the percentage of weeks which have a positive return, for the all and anomaly portfolios. As well, White's reality check (WRC) p-value, testing if the anomaly assets portfolio has higher weekly returns than the all assets portfolio. For models: FFNN, LSTM, MFCM^{L} and MFCM^{O}.

	FFNN				LSTM				MFCM^{L}				MFCM^{O}			
Metric	5	10	20	40	5	10	20	40	5	10	20	40	5	10	20	40
All hit rate	50.8	52.3	56.1	57.1	57.3	57.6	57.8	55.6	57.5	55.5	58.4	57.7	53.1	54.3	55.5	58.9
Anomaly hit rate	47.5	47.2	47.5	51.6	52	54.7	55.2	56.8	51.4	53.8	54.3	56.2	56.5	56.5	59.1	58.4
WRC p-value	0.46	0.29	0.08	0.21	0.28	0.39	0.32	0.26	0	0.14	0.16	0.25	0.74	0.73	0.67	0.5

The first overall observation from Figure 8 is that it actually carries out the characteristics of the null hypothesis graph in Figure 1. This is seen by the first four, least complex models: elastic net LR, KNN regression, RF and GBT; the mean weekly return of their anomaly assets portfolios are consistently, and substantially higher than the mean weekly return from their all assets portfolios. This visual outcome is as well represented in the White's reality check p-values of Table 12. Here, all 16 portfolio comparisons are significant at the 5% level, accepting the alternative hypothesis that the anomaly assets portfolio gives higher returns than the all assets portfolio.

Following these observations, is that the next four, most complex, models, being FFNN, LSTM network, MFCM^{L} and MFCM^{O}, have their returns between the anomaly assets and all assets portfolios much closer together. This is seen visually from Figure 8, but as well by the p-values of White's reality check in Table 13. Here, 14 out of the 16 portfolio comparisons fail to reject the null hypothesis that the two portfolios give the same returns. Additionally, for LSTM network, MFCM^{L} and MFCM^{O}, weekly hit rates are all generously above 50% with values ranging from 51.4% to 59.1%, and a mean value across all 24 portfolios of 56%.

A curious observation from Figure 1 is that for the first five models which perform poorly, having the largest number of assets, being 40, in the portfolio almost always gives the highest mean weekly return. However, for the last three models which perform far better, the opposite is seen, where a lower number of assets gives a higher mean weekly return. This would seem that it is a consequence of having more predictive ability, such that with higher accuracy, the models can put less emphasis on diversification towards the mean naive 1/N return, and more emphasis on selecting the assets which are forecasted to give higher returns.

The most interesting observation from Figure 8 comes from the most complex model; MFCM^{O}. It is seen that 3 out of its 4 portfolio comparisons exhibit the all assets portfolio achieving a higher mean weekly return than its respective anomaly assets portfolio. A higher value may be explained by MFCM^{O} in the all assets category having twice as many assets as MFCM^{O} in the anomaly assets category. Then, even though half of these assets are made up of the low momentum decile which produce on average worse returns, there may be some weeks were certain of these assets achieve high returns that the model is able to predict.

Therefore, MFCM^{O} is shown to be the level of model complexity required to initiate a convergence between the two portfolios, as suggested to be possible by the null hypothesis of this thesis in Section 4. Accordingly, the null hypothesis is not rejected, such that if using a complex enough model when predicting risk premiums to construct portfolios, there is no need to differentiate between the two sides of an asset pricing anomaly when optimizing average returns, and instead the entire universe of assets can be considered.

However, MFCM^{O} is a rather complex model to construct with much longer runtimes consisting of training and finding the optimal hyperparameters of six different LSTM networks. Then, it is seen from Figure 8 that for all seven other models, the anomaly assets portfolio consistently performs better than its counterpart all assets portfolio, excluding FFNN5 which still performed far worse than the naive 1/N portfolios. Therefore, in order to ignore the asset pricing anomaly, an investor must be confident in the complexity of their forecasting model to isolate the optimal assets, regardless whether the anomaly assets or all assets are being considered.

6.2.1 Risk Analysis

A question remains about the risks associated with the two types of portfolios, such that the anomaly assets portfolios may have a higher expected return because their asset pool carries more risk. In order to analyze this, it is noted that each model contains four pairs of portfolios, one for each number of assets; {5, 10, 20, 40}, with the pairs being made with the anomaly assets portfolio and its respective all assets portfolio. Therefore, it is of interest to see for each of these pairs of portfolios, which one in the pair is found to carry more risk. By the no arbitrage argument that a higher return should yield higher risk, it would be expected that the portfolios constructed from the anomaly assets category should carry more risk. Then, for the portfolios constructed from the all assets category, by including the lower expected return assets, this should reduce the risk exposure.

Table 14 contains the number of pairs for each model, where the anomaly assets portfolio has a higher risk than the all assets portfolio. This is analyzed over six risk metrics standard to the risk management literature, being; standard deviation, value-at-risk 95%, value-at-risk 99%, expected shortfall 95%, expected shortfall 99%, and maximum drawdown. A complete list all the risk and return metrics is given in the appendix section A.2.

Table 14: For each model there are four pairs of portfolios, one for each number of assets; {5, 10, 20, 40}. This table contains the number of pairs for each model, where the anomaly assets portfolio has higher risk than its respective all assets portfolio, for six risk metrics. The bottom row, gives the percentage total of how many risk metric pairs, out of the 24 for each model, show higher risk for the anomaly assets portfolios.

Metric	LR	KNN	RF	GBT	FFNN	LSTM	MFCM^{L}	MFCM^{O}
Standard deviation	2	0	0	0	0	4	1	4
VaR 95%	0	0	0	0	0	4	0	4
VaR 99%	0	0	0	0	0	3	0	4
ES 95%	0	0	0	0	0	4	0	4
ES 99%	0	0	0	0	0	4	0	4
Maximum drawdown	0	0	0	0	1	0	0	1
Percent of total metrics	8	0	0	0	4	79	4	88

From Table 14 an interesting phenomenon occurs, such that for the first five models which performed the worst in terms of their average return from Figure 8, as well as linear MFCM, the anomaly assets portfolios are substantially less risky than the all assets portfolios. The range of percentage of risk metrics, where the anomaly assets portfolio in the pairs has more risk than its respective all assets portfolio, is from only 0% to 8% across the six models. This observation goes against the expectation that the portfolios constructed from the expected higher return category should give more risk. Therefore, when using these six models, by adding the lower expected return assets to the asset space, this actually increased the risk of the portfolios.

However, for the two models that solely use LSTM networks, being LSTM and the only model to have average returns converge, optimal MFCM, the opposite finding is observed. Such that 79% and 88%, for LSTM and optimal MFCM respectively, of the portfolio pair's risk metrics show that the anomaly assets portfolios have higher risk than the all assets portfolios. For these two models, the expectation of risk exposer is realized, such that by adding the lower expected return assets to the asset space, this decreases the risk of the portfolios.

This is quite a valuable finding for the most predictive model, optimal MFCM, as ignoring the anomaly, and adding the lower expected return assets, not only increases returns in three out of its four portfolios, but decreases 88% of the risk metrics associated with these portfolios.

As well, a further interesting discovery is that the enforcement of a linear structure on the activation function, in linear MFCM, does not carry the same risk benefits as the more complex structure on the activation function, when using a LSTM network in optimal MFCM.

Therefore, for optimal MFCM, this provides additional evidence that the anomaly should be ignored, and that it is beneficial to consider all assets together. Where now, beyond having the average returns of the portfolios converge from Figure 8, the portfolios constructed from the all assets category also have a further substantial risk advantage, with 88% of their risk metrics showing less risk compared to their respective anomaly assets portfolios.

Accordingly, considering that the optimal MFCM model has all eight of its portfolio's mean returns greater than both of the naive 1/N portfolios, as well as its ability to converge average returns and decrease risk exposer, optimal MFCM is taken to construct the best performing portfolios. Figure 9 shows the cumulative return of the four optimal MFCM portfolios constructed considering all assets, compared to the baseline all assets naive 1/N portfolio. This plot displays a similar finding to Krauss et al. (2017), Fischer and Krauss (2018), and Huck (2009), where having a lower number of assets in the portfolio gives higher returns, but as well a higher standard deviation in returns.

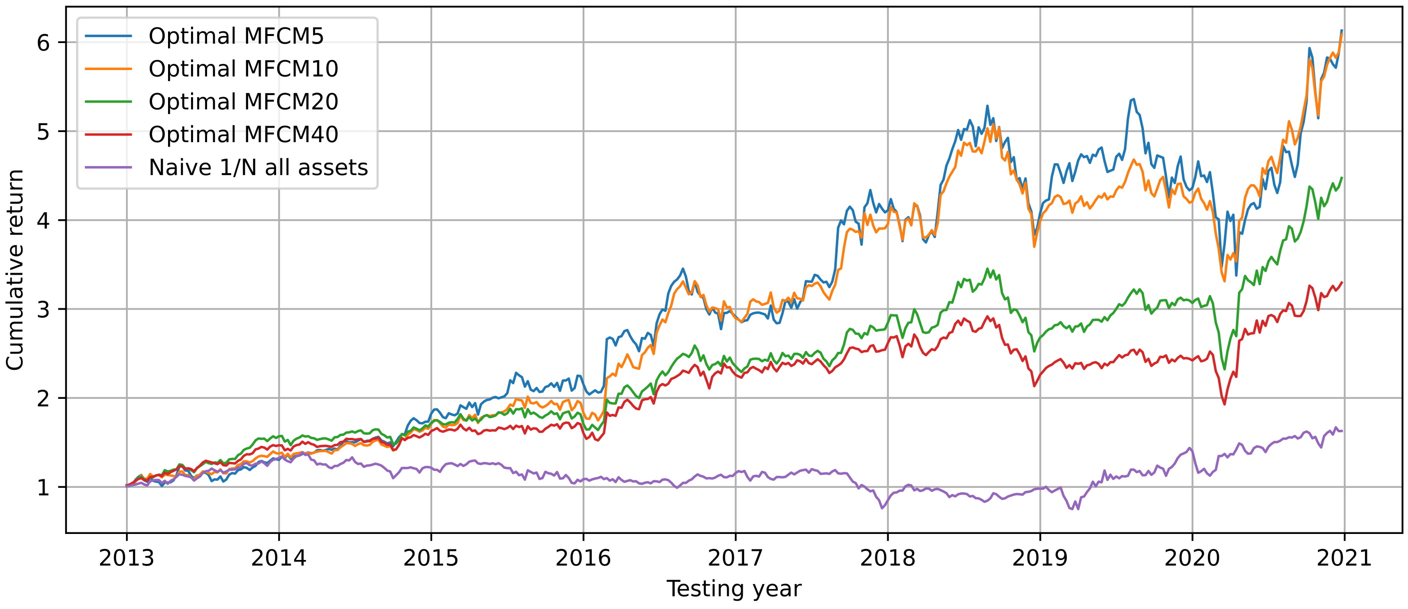


Figure 9: Cumulative return plot for optimal MFCM in the all assets category for the out-of-sample testing set.

6.3 Accuracy Analysis

Table 15 gives the accuracy results of all eight models when trained on the all assets category. It presents a rather striking outcome when considering the portfolio results of section 6.2, such that both linear and optimal MFCM models perform substantially the worst in terms of MSE. The elastic net LR actually performs the best for MSE, and additionally has the lowest standard deviation in predictions.

However, both MFCM models perform substantially the best in terms of classification accuracy, with optimal MFCM being the highest at 51.71%. Even though the portfolio selection is made based on MSE, this advantage in classification accuracy would still be a factor enabling the MFCM models to select a higher ratio of successful assets in their portfolios. Furthermore, the classification accuracy results line up much more closely to the portfolio return results, where MSE is almost the opposite. This shows that a model's classification accuracy is what really influences the success of the portfolio construction method used in this thesis, not MSE.

Additionally, for both MFCM models, all 16 portfolios achieve a higher weekly hit rate than their associated classification accuracy, reaching a maximum of 59.1% as shown in Table 13. This highlights the optimal portfolio construction method used in this thesis, such that the model is only selecting the top 5 to 40 assets out of either 300 or 600 in total. Therefore, this enables selecting only the assets which the model estimates will have by far the highest returns, not just the ones that are predicted to have a slightly above zero return. In doing so, the models focus on selection only the assets which they are most sure about, hence, this is why with a small classification accuracy score, the portfolios can achieve a much larger weekly hit rate.

Additionally, this leads to a further hypothesis that perhaps MFCM is better suited for classification rather than regression. In this case, MFCM's activation function would simply be taken to be a sigmoid function instead of linear, and a further step would be to make the frequency forecasting models also predict probabilities. However, in this thesis, MSE was considered and it is of interest to investigate why MFCM's MSE are so much higher.

The immediate explanation would be that MFCM only produces accurate results when the frequency forecasts, which are taken as inputs in the activation function, are all accurate themselves. If they are not, then two outcomes could happen. The first, is that the larger inaccuracies are on both sides of the true value, evidently cancelling each other out and leading to only a moderate error. The second, is that the larger inaccuracies fall on the same side of the true value, where this convolution of inaccurate forecasts through taking them as input in the activation function may lead some predictions to have very large errors. To analyze this, first the kurtosis, skewness and the standard deviation of the error distributions of each model are presented in Table 16.

Table 15: Accuracy results of all eight models when trained on the all assets category.

Metric	LR	KNN	RF	GBT	FFNN	LSTM	MFCM^L	MFCM^O
Mean squared error	0.006434	0.006523	0.006435	0.006463	0.006439	0.006441	0.008848	0.008848
Classification (%)	49.61	50.05	50.55	50.25	50.44	50.99	51.33	51.71
Precision (%)	49.07	49.56	50.02	49.73	49.92	50.64	50.72	51.39
Recall (%)	48.9	54.66	62.84	53.08	60.74	36.37	56.89	43.76
Standard deviation	0.0014	0.0094	0.0023	0.0054	0.0019	0.003	0.0022	0.0024

Table 16: Error distribution kurtosis, skewness and standard deviations.

Metric	LR	KNN	RF	GBT	FFNN	LSTM	MFCM^L	MFCM^O
Kurtosis	8.27	8.23	8.33	8.32	8.31	8.38	295.07	295.2
Skewness	0.8889	0.9303	0.9143	0.9269	0.8986	0.9385	6.7769	6.7641
Standard Deviation	0.0802	0.0808	0.0802	0.0804	0.0802	0.0802	0.094	0.094

From Table 16 it is seen that the kurtosis is massive for the MFCM models compared to the first six models. This shows that indeed, the MFCM models have very thick tails in their error distribution, with certain outlier predictions far different from their true values. This is further implied by their standard deviations being larger than the first six models. Additionally, the error distributions of all eight models are positively skewed, showing that the models tend to overestimate returns rather than underestimate them.

Having outliers in the MFCM predictions may be why their MSE is much higher than the rest of the models. To analyze if this is the case, the MSE is compared when removing a certain quantile of outliers, as presented in Table 17. After removing the first 10% of outliers in the second row, both MFCM models are no longer the worst performing models, where KNN now has a higher MSE. This trend continues, until getting to the top 10% of predictions in the final row where the MFCM models are exactly center of the pack in terms of MSE. Therefore, certain observations exhibiting large out-of-sample variance in predictions is a key disadvantage of MFCM. However, its ability to classify positive and negative returns with a higher accuracy than the first six models, is what enables it to achieve a substantially higher average weekly return.

Table 17: MSE after removing largest quantiles. Multiplied by 1,000,000.

Quantiles removed	LR	KNN	RF	GBT	FFNN	LSTM	MFCM^{L}	MFCM^{O}
0	6432	6520	6433	6460	6437	6439	8613	8612
0.1	2236	2283	2233	2246	2235	2228	2275	2274
0.5	339.5	351.3	339.1	340.6	339.1	338.7	342.5	342.1
0.9	9.89	11.04	9.76	10.23	9.81	10.16	9.97	10.08

7 Conclusion

This thesis covered a rather straight forward, practical question of the one-sided optimal portfolio potential of asset pricing anomalies. It investigated this by a machine learning, risk premium prediction based portfolio construction method, largely inspire by the likes of Krauss et al. (2017), Fischer and Krauss (2018), and Gu et al. (2020). In doing so, it expanded these methods by introducing a new multi-frequency combination methodology to further enhance forecasting accuracy, and in turn, portfolio performance. The goal of MFCM was not the methodology in itself, but to display that it is possible to combine forecasts over multiple horizons, in similar, parametric ways as the more traditional forecast combination techniques used over different models.

To conclude, there are six key findings discovered in this thesis. First, came from the asset pricing anomaly analysis, which showed that there is a statistically significant decrease, both in terms of intercept and slope, of the size and value anomalies when taking a 12 month holding period over the last 28 years.

Second, the main research question poised in this thesis was answered, stated again as: should an investor, when creating long-only optimal portfolios as subsets of an asset space, strictly consider the assets in the expected higher return side of an anomaly? Or, does disregarding the anomaly, and considering all assets together in both sides of the anomaly, result in the same or higher average returns for the long-only optimal portfolios? The null hypothesis of this question went along with the standard practice in portfolio construction, such that considering all assets together is beneficial when constructing an optimal portfolio, and furthermore, that there is a level of model complexity where the two portfolios would converge in mean weekly returns. This convergence is explained by a complex model's ability to isolate what it considers the optimal assets. Then, since the all assets category includes every asset in the anomaly assets category, and the assets in the anomaly assets category give a higher average return, the models will choose the same subset of assets in each optimal portfolio.

It is found that with the most complex model used, being MFCM with a LSTM network as its activation function, the optimal portfolios from the all assets category actually give a higher or close to identical mean weekly return than the optimal portfolios from anomaly assets category. Additionally, for this model, 88% of the risk metrics for the all assets portfolios show less risk than their respective anomaly assets portfolio. Therefore, the null hypothesis is not rejected, such that the asset pricing anomaly can be ignored, in terms of average return as well as risk, when using a competently complex and accurate machine learning model for risk premia estimation.

Third, MFCM constructs the best performing portfolios by average return, with optimal MFCM having all eight of its portfolios achieve a mean weekly return that is higher than both naive 1/N portfolios for the anomaly assets and the all assets categories.

Fourth, from the accuracy analysis it is clear that classification accuracy is the determining factor in the performance of the optimal portfolios, not MSE. Therefore, when constructing optimal portfolios as a subset of an asset space based on risk premia estimation, it is rather the probability of having a positive risk premia that is more important than the actual predicted value of the risk premia itself.

Fifth, out of the six standard models, elastic net LR had the lowest MSE, but LSTM had the highest classification accuracy. Additionally, out of the six standard models, LSTM is the only one where all eight of its portfolios beat their respective naive 1/N portfolio's mean weekly return.

Sixth, MFCM has by far the highest MSE due to large outlier errors in its predictions, but it also has by far the highest classification accuracy. This leads to MFCM being more suited towards a classification problem, rather than a regression problem.

Overall, the practical side of this thesis shows that it is possible to ignore the use of asset pricing anomalies as a selection device when constructing optimal portfolios as a subset of an asset space. However, this requires extensive work through the use of optimal MFCM, where training and finding the optimal hyperparameters of six separate LSTM networks is required. If using more standard models to construct risk premia predictions, then the asset pricing anomaly should not be ignored, where the optimal portfolios constructed as a subset of the assets in the expected higher return side of the anomaly, consistently produces larger mean weekly returns. Lastly, the theoretical side of this thesis introduces the success of MFCM, where it shows that forecasts can be combined over multiple horizons, in similar, parametric ways as the more traditional forecast combination techniques used over different models.

7.1 Future Research

I believe there are three overarching areas with the potential for future research. The first is with regards to the main question about the optimal portfolio potential of one side of an asset pricing anomaly. In this thesis a focus was made on optimal portfolio construction through only taking a subset of assets from either the one side of the anomaly, or all assets together. However, the same question can be asked for optimal portfolio construction methods which don't form subsets, but instead, build portfolio weights with all assets together, such as Markowitz mean-variance theory. Additionally, the same question can be asked of any other asset pricing anomalies, asset pricing factors, or general conditions that separates a universe of assets into two categories where one is expected to achieve a higher mean return.

The second main area for future research is regards to how to train a model on only the assets in a specific side of the asset pricing anomaly. The path taken in this thesis is the same as previous research, which followed a similar portfolio construction strategy, such that the model is trained on every asset which it can then select from when constructing the optimal portfolio. An alternative training method to this can be done by taking the life of the anomaly into consideration throughout the entire testing set. Specifically, training would take place not just over the assets in the side of the anomaly, but over every known asset. With the variation being only during time periods where that asset falls into the side of the anomaly which is currently being predicted. Hence, every observation in the training data belongs to an asset where at that point in time, resided in the specific side of the asset pricing anomaly.

The third main area for future research is regards to further development of MFCM. To begin, a focus on classification rather than regression, as shown, is where it already performs the best. To start, MFCM's activation function would change to a sigmoid structure, and even each of its frequency forecasting models can as well predict probabilities. Furthermore, and perhaps the most interesting, is the potential to combine forecasts together over both multiple horizons and multiple models. To begin this, a matrix of predictions would be estimated, where each row represents a different model, and each column a different forecasted horizon. From here, the opportunities to create new multi-frequency and multi-model forecast combination techniques seems rather endless.

References

Abadi, M., Barham, P., Chen, J., Chen, Z., Davis, A., Dean, J., ... others (2016). Tensorflow: a system for large-scale machine learning. In *12th {USENIX} symposium on operating systems design and implementation ({OSDI} 16)* (pp. 265–283).

Altman, N. (1992). An introduction to kernel and nearest-neighbor nonparametric regression. *The American Statistician*, *46(3)*, 175–185.

Asness, C. S., Moskowitz, T. J., & Pedersen, L. H. (2013). Value and momentum everywhere. *Journal of Finance*, *68(3)*, 929–985.

Back, K. E. (2017). *Asset pricing and portfolio choice theory* (2nd ed.). Oxford University Press.

Banz, R. W. (1981). The relationship between return and market value of common stocks. *Journal of Financial Economics*, *9(1)*, 3–18.

Bao, W., Yue, J., & Rao, Y. (2017). A deep learning framework for financial time series using stacked autoencoders and long-short term memory. *PLoS ONE*, *12(7)*, 1–24.

Breiman, L. (2001). Random forests. *Machine Learning*, *45(1)*, 5–32.

Chen, T., & Guestrin, C. (2016). XGBoost: a scalable tree boosting system. In *Proceedings of the 22nd acm sigkdd international conference on knowledge discovery and data mining* (pp. 785–794). New York, NY, USA: ACM.

Chollet, F., et al. (2015). *Keras.* GitHub. Retrieved from https://github.com/fchollet/keras

Coulson, N. E., & Robins, R. P. (1993). Forecast combination in a dynamic setting. *Journal of Forecasting*, *12*, 63–67.

Davis, S. J., & Kahn, J. A. (2008). Interpreting the great moderation: changes in the volatility of economic activity at the macro and micro levels. *Journal of Economic Perspectives*, *22(4)*, 155–180.

DeMiguel, V., Garlappi, L., & Uppal, R. (2009). Optimal versus naive diversification: how inefficient is the 1/n portfolio strategy? *The Review of Financial Studies*, *22(5)*, 1915–1953.

Fama, E. F., & French, K. R. (1992). The cross-section of expected stock returns. *Journal of Finance*, *47(2)*, 427–465.

Fama, E. F., & French, K. R. (1993). Common risk factors in the returns on stocks and bonds. *Journal of Financial Economics*, *33(1)*, 3–56.

Fama, E. F., & French, K. R. (2015). A five-factor asset pricing model. *Journal of Financial Economics*, *116*, 1–22.

Fischer, T., & Krauss, C. (2018). Deep learning with long short-term memory networks for financial market predictions. *European Journal of Operational Research*, *270*, 654–669.

Fix, E., & Hodges, J. L. (1951). *Discriminatory analysis. nonparametric discrimination: consistency properties.* USAF School of Aviation Medicine. Retrieved from https://apps.dtic.mil/dtic/tr/fulltext/u2/a800276.pdf

FTSE-Russell. (2021, August 31). *Index factsheet: russell 3000 index.* Retrieved from https://research.ftserussell.com/Analytics/FactSheets/temp/1e256b4f-d8cc-45ce-ba64-c224d9e2fe0a.pdf

Granger, C. W., & Ramanathan, R. (1984). Improved methods of combining forecasts. *Journal of Forecasting*, *3*, 197–204.

Gu, S., Kelly, B., & Xiu, D. (2020). Empirical asset pricing via machine learning. *The Review of Financial Studies*, *33(5)*, 2223–2273.

Hastie, T., Tibshirani, R., & Friedman, J. (2009). *The elements of statistical learning: data mining, inference, and prediction* (2nd ed.). Springer.

Hochreiter, S., & Schmidhuber, J. (1997). Long short-term memory. *Neural Computation*, *9(8)*, 1735–1780.

Hornik, K., Stinchcombe, M., & White, H. (1989). Multilayer feedforward networks are universal approximators. *Neural Networks*, *2(5)*, 359–366.

Huck, N. (2009). Pairs selection and outranking: an application to the s&p 100 index. *European Journal of Operational Research*, *196*, 819–825.

Jegadeesh, N., & Titman, S. (1993). Returns to buying winners and selling losers: implications for stock market efficiency. *Journal of Finance*, *48(1)*, 65–91.

Keskar, N. S., Mudigere, D., Nocedal, J., Smelyanskiy, M., & Tang, P. T. P. (2017, February 9). *On large-batch training for deep learning: generalization gap and sharp minima.* ICLR 2017, Toulon, France.

Kline, D. M. (2004). *Neural networks in business forecasting.* Idea Group Inc.

Kourentzes, N., Petropoulos, F., & Trapero, J. R. (2014). Improving forecasting by estimating time series structural components across multiple frequencies. *International Journal of Forecasting*, *270*, 291–302.

Krauss, C., Do, X. A., & Huck, N. (2017). Deep neural networks, gradient-boosted trees, random forests: statistical arbitrage on the s&p 500. *European Journal of Operational Research*, *259*, 689–702.

Krauss, C., & Stübinger, J. (2017). Non-linear dependence modelling with bivariate copulas: statistical arbitrage pairs trading on the s&p 100. *Applied Economics*, *49(5)*, 5352–5369.

Li, L., Jamieson, K., DeSalvo, G., Rostamizadeh, A., & Talwalker, A. (2018). Hyperband: a novel bandit-based approach to hyperparameter optimization. *Journal of Machine Learning Research*, *18*, 1–52.

Masters, T. (1993). *Practical neural network recipes in c++.* New York: Academic Press.

McCulloch, W. S., & Pitts, W. (1943). A logical calculus of the ideas immanent in nervous activity. *The Bulletin of Mathematical Biophysics*, *5*, 115–133.

McKinney, W. (2010). Data structures for statistical computing in python. In *Proceedings of the 9th python in science conference* (Vol. 445, pp. 51–56).

Medsker, L., & Jain, L. (2000). *Recurrent neural networks: design and applications.* CRC-Press.

Mincer, J. A., & Zarnowitz, V. (1969). The evaluation of economic forecasts. *Economic Forecasts and Expectations: Analysis of Forecasting Behavior and Performance*, 3–46.

Olah, C. (2015). *Understanding lstm networks.* Retrieved from http://colah.github.io/posts/2015-08-Understanding-LSTMs/

Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., ... others (2011). Scikit-learn: machine learning in python. *Journal of machine learning research*, *12*(Oct), 2825–2830.

Petropoulos, F., & Kourentzes, N. (2014). Improving forecasting via multiple temporal aggregation. *Foresight: The International Journal of Applied Forecasting*, *34*, 12–17.

Politis, D. N., & Romano, J. P. (1994). The stationary bootstrap. *Journal of the American Statistical Association*, *89(428)*, 1303–1313.

Sak, H., Senior, A., & Beaufays, F. (2014). Long short-term memory based recurrent neural network architectures for large vocabulary speech recognition. arXiv:1402.1128.

Schapire, R. E. (1990). The strength of weak learnability. *Machine Learning*, *5*, 197–227.

Sharpe, W. F. (1964). Capital asset prices: a theory of market equilibrium under conditions of risk. *Journal of Finance*, *19(3)*, 425–442.

Srivastava, N., Hinton, G., Krizhevsky, A., Sutskever, I., & Salakhutdinov, R. (2014). Dropout: a simple way to prevent neural networks from overfitting. *Journal of Machine Learning Research*, *15(1)*, 1929–1958.

Van Rossum, G., & Drake, F. L. (2020). *Python 3.8 reference manual.* Scotts Valley, CA: CreateSpace. Retrieved from `https://docs.python.org/3.8/`

Veaux, R. D. D., & Ungar, L. H. (1994). Multicollinearity: a tale of two nonparametric regressions. *Selecting Models from Data*, *89*, 393–402.

Walt, S. v. d., Colbert, S. C., & Varoquaux, G. (2011). The numpy array: a structure for efficient numerical computation. *Computing in Science & Engineering*, *13(2)*, 22–30.

White, H. (2000). A reality check for data snooping. *Econometrica*, *68(5)*, 1097–1126.

Zou, H., & Hastie, T. (2005). Regularization and variable selection via the elastic net. *Journal of the Royal Statistical Society*, *67(2)*, 301–320.

A Appendix

A.1 Optimal Hyperparameters and Runtimes

Table 18: Hyperparameter keys of tuple for LR and KNN.

Model	Keys
LR	(alpha regularization, L1 weight)
KNN	(nearest neighbors, prediction weights)

Table 19: Optimal hyperparameters and runtimes for LR and KNN. Since fit times for KNN are instantaneous, runtimes are taken from prediction times. Runtimes are stated in seconds.

Model	Year	Hyperparameters	Validation Runtime	Model runtime
LR - High	2013	(0.25, 0.1)	0.79	0.19
	2014	(0.25, 0.1)	0.8	0.16
	2015	(0.1, 0.1)	0.97	0.22
	2016	(1, 0)	17.27	6.01
	2017	(0.1, 0.1)	0.88	0.21
	2018	(0.1, 0.1)	0.77	0.16
	2019	(0.75, 0.5)	0.94	0.19
	2020	(0.5, 0.75)	0.87	0.13
LR - All	2013	(0.1, 0.1)	0.9	0.4
	2014	(0.1, 0.1)	1.02	0.42
	2015	(0.25, 0.1)	0.94	0.44
	2016	(3, 0)	45.6	14.28
	2017	(0.1, 0.1)	0.87	0.39
	2018	(0.1, 0.1)	0.81	0.4
	2019	(0.1, 0.25)	1.38	0.44
	2020	(0.1, 0.1)	0.87	0.32
KNN - High	2013	(60, uniform)	905	198
	2014	(60, uniform)	810	202
	2015	(60, uniform)	942	212
	2016	(60, distance)	655	194
	2017	(60, uniform)	967	194
	2018	(60, uniform)	721	204
	2019	(60, uniform)	911	221
	2020	(60, uniform)	616	168
KNN - All	2013	(60, uniform)	3524	398
	2014	(60, uniform)	3589	408
	2015	(60, uniform)	3976	427
	2016	(60, uniform)	3003	390
	2017	(60, uniform)	3070	393
	2018	(60, uniform)	3285	410
	2019	(60, uniform)	4161	432
	2020	(60, uniform)	2319	332

Table 20: Hyperparameter keys of tuple for RF and GBT.

Model	Keys
RF	(number of trees, max depth, feature subsample by level, data subsample, L2 regularization)
GBT	(number of trees, max depth, learning rate, feature subsample by level, data subsample, L2 regularization)

Table 21: Optimal hyperparameters and runtimes for LR and KNN. Runtimes are stated in seconds.

Model	Year	Hyperparameters	Validation Runtime	Model runtime
RF - High	2013	(100, 3, 0.2, 0.3, 0.5)	290	6
	2014	(100, 4, 0.2, 0.3, 0.5)	259	7
	2015	(600, 4, 0.2, 0.3, 0.001)	397	43
	2016	(600, 4, 0.2, 0.3, 0.5)	401	34
	2017	(600, 2, 0.5, 0.3, 0.001)	292	32
	2018	(300, 6, 0.2, 0.3, 0.5)	370	26
	2019	(600, 3, 0.2, 0.3, 0.5)	302	39
	2020	(600, 4, 0.2, 0.3, 0.5)	385	34
RF - All	2013	(100, 2, 0.2, 0.3, 0.5)	418	10
	2014	(600, 3, 0.2, 0.3, 0.5)	514	66
	2015	(600, 4, 0.5, 0.3, 0.5)	554	82
	2016	(600, 3, 0.2, 0.3, 0.5)	504	53
	2017	(100, 2, 0.2, 0.3, 0.5)	387	9
	2018	(600, 3, 0.5, 0.3, 0.5)	514	61
	2019	(600, 2, 0.5, 0.3, 0.5)	542	59
	2020	(300, 3, 0.5, 0.3, 0.5)	403	28
GBT - High	2013	(200, 2, 0.07, 0.2, 0.6, 0.5)	354	13
	2014	(200, 2, 0.07, 0.5, 0.6, 0.001)	336	12
	2015	(200, 2, 0.07, 0.5, 0.6, 0.001)	356	12
	2016	(200, 2, 0.07, 0.5, 0.6, 0.001)	401	9
	2017	(200, 2, 0.07, 0.2, 0.6, 0.001)	367	12
	2018	(200, 2, 0.07, 0.2, 0.6, 0.001)	332	10
	2019	(200, 2, 0.07, 0.2, 0.6, 0.5)	352	14
	2020	(200, 2, 0.07, 0.5, 0.6, 0.5)	317	10
GBT - All	2013	(200, 2, 0.07, 0.5, 0.6, 0.001)	501	25
	2014	(200, 2, 0.07, 0.5, 0.6, 0.001)	535	22
	2015	(200, 2, 0.07, 0.5, 0.6, 0.001)	532	23
	2016	(200, 2, 0.07, 0.2, 0.6, 0.001)	523	19
	2017	(200, 2, 0.07, 0.5, 0.6, 0.001)	484	21
	2018	(200, 2, 0.07, 0.5, 0.6, 0.001)	495	21
	2019	(200, 2, 0.07, 0.5, 0.6, 0.001)	547	24
	2020	(200, 2, 0.07, 0.5, 0.6, 0.5)	423	18

Table 22: Hyperparameter keys of tuple for FFNN and LSTM.

Model	Keys
FFNN	(Nodes per hidden layer, activation function, dropout rate, L2 regularization, learning rate)
LSTM	(Nodes per hidden layer, activation function, dropout rate, L2 regularization, learning rate)

Table 23: Optimal hyperparameters and runtimes for LR and KNN. Runtimes are stated in seconds.

Model	Year	Hyperparameters	Validation Runtime	Model runtime
FFNN - High	2013	([4, 4, 4], relu, 0.3, 0.0001, 0.0001)	432	15
	2014	([4, 4, 4], relu, 0.3, 0.0001, 1e-05)	421	10
	2015	([26, 12, 6], relu, 0.3, 0.0001, 0.0001)	468	9
	2016	([4, 4], relu, 0.3, 0.0001, 0.0001)	443	15
	2017	([4, 4], relu, 0.3, 0.0001, 0.0001)	464	6
	2018	([4, 4, 4], relu, 0.3, 0.0001, 0.0001)	411	12
	2019	([4, 4, 4, 4], relu, 0.6, 0.0001, 1e-05)	501	23
	2020	([4, 4, 4, 4], relu, 0.6, 0.0001, 1e-05)	443	20
FFNN - All	2013	([26, 12, 6], relu, 0.3, 0.0001, 0.0001)	794	27
	2014	([26, 12, 6], relu, 0.6, 0.0001, 1e-05)	955	31
	2015	([26], sigmoid, 0.6, 0.0001, 0.0001)	785	15
	2016	([26], relu, 0.6, 0.0001, 0.0001)	621	10
	2017	([4, 4], relu, 0.3, 0.0001, 0.0001)	657	10
	2018	([26], sigmoid, 0.6, 0.0001, 0.0001)	677	14
	2019	([26], sigmoid, 0.6, 0.0001, 0.0001)	757	14
	2020	([4], relu, 0.6, 0.0001, 0.0001)	607	8
LSTM - High	2013	([4], tanh, 0.3, 0.0001, 1e-05)	821	10
	2014	([4, 4, 4], tanh, 0.6, 0.0001, 0.0001)	960	17
	2015	([26, 12, 6], tanh, 0.3, 0.0001, 1e-07)	968	22
	2016	([4, 4, 4], tanh, 0.6, 0.0001, 0.0001)	916	26
	2017	([4, 4, 4], tanh, 0.6, 0.0001, 0.0001)	1089	40
	2018	([4, 4, 4], tanh, 0.3, 0.0001, 0.0001)	1023	23
	2019	([4, 4, 4], tanh, 0.6, 0.0001, 0.0001)	1211	16
	2020	([26, 12, 6], tanh, 0.3, 0.0001, 1e-07)	986	25
LSTM - All	2013	([26, 12, 6], tanh, 0.6, 0.0001, 0.0001)	1780	64
	2014	([4, 4, 4], tanh, 0.3, 0.0001, 0.0001)	1866	53
	2015	([4, 4, 4], tanh, 0.3, 0.0001, 1e-05)	1683	34
	2016	([4, 4, 4], tanh, 0.6, 0.0001, 1e-05)	1373	30
	2017	([4, 4, 4], tanh, 0.3, 0.0001, 1e-07)	1494	45
	2018	([4, 4, 4], tanh, 0.3, 0.0001, 1e-07)	1599	37
	2019	([4, 4, 4], tanh, 0.6, 0.0001, 1e-07)	1612	37
	2020	([4, 4, 4], tanh, 0.3, 0.0001, 1e-07)	1383	18

Table 24: Runtimes for MFCM. Runtimes are stated in seconds.

Model	Year	Validation Runtime	Model runtime
MFCM - High	2013	4224	815
	2014	4391	2174
	2015	3963	1046
	2016	4262	708
	2017	4028	2814
	2018	4129	2675
	2019	4407	1259
	2020	3877	1350
MFCM - All	2013	6499	1634
	2014	6794	1518
	2015	6795	4095
	2016	6567	3292
	2017	6717	1420
	2018	7064	4045
	2019	7116	7184
	2020	6467	3338

A.2 All Portfolio Metrics

Table 25: Elastic net LR out-of-sample portfolio results.

	Anomaly assets				All assets			
Metric	LR5	LR10	LR20	LR40	LR5	LR10	LR20	LR40
Mean weekly return	0.0034	0.0052	0.0034	0.0032	0.0001	-0.0011	-0.0003	0.0006
WRC p-value	0.4337	0.0918	0.0995	0.0566	0.7419	0.8154	0.7233	0.5542
Standard deviation	0.096	0.062	0.044	0.037	0.078	0.059	0.05	0.045
Skewness	11.384	5.803	2.355	0.953	2.53	1.231	0.955	0.821
Kurtosis	189.309	72.008	22.293	8.461	17.648	7.332	5.593	4.475
Sharpe ratio	0.258	0.596	0.555	0.638	0.012	-0.13	-0.049	0.095
VaR 95	-0.088	-0.066	-0.054	-0.056	-0.1	-0.087	-0.071	-0.068
VaR 99	-0.157	-0.101	-0.092	-0.084	-0.16	-0.125	-0.12	-0.099
ES 95	-0.127	-0.1	-0.085	-0.078	-0.134	-0.116	-0.1	-0.091
ES 99	-0.172	-0.153	-0.125	-0.112	-0.183	-0.165	-0.151	-0.126
Annualized return	0.02	0.2	0.136	0.143	-0.127	-0.133	-0.079	-0.021
Maximum drawdown	-0.715	-0.443	-0.434	-0.363	-0.808	-0.737	-0.746	-0.739

Table 26: KNN regression out-of-sample portfolio results.

	Anomaly assets				All assets			
Metric	KNN5	KNN10	KNN20	KNN40	KNN5	KNN10	KNN20	KNN40
Mean weekly return	0.0011	0.0014	0.0025	0.0031	-0.001	-0.0018	-0.0005	0
WRC p-value	0.5093	0.3863	0.1253	0.0345	0.8684	0.8907	0.754	0.6423
Standard deviation	0.05	0.04	0.035	0.03	0.086	0.065	0.05	0.04
Skewness	-0.018	-0.019	0.29	0.389	1.278	1.15	1.125	0.629
Kurtosis	2.164	2.628	3.495	3.053	6.144	4.953	5.858	3.79
Sharpe ratio	0.159	0.242	0.508	0.729	-0.083	-0.204	-0.079	0.004
VaR 95	-0.081	-0.057	-0.051	-0.048	-0.123	-0.096	-0.073	-0.057
VaR 99	-0.143	-0.1	-0.078	-0.068	-0.198	-0.159	-0.109	-0.105
ES 95	-0.118	-0.091	-0.074	-0.062	-0.172	-0.126	-0.099	-0.086
ES 99	-0.161	-0.131	-0.11	-0.083	-0.229	-0.176	-0.133	-0.12
Annualized return	-0.008	0.029	0.101	0.146	-0.21	-0.183	-0.087	-0.04
Maximum drawdown	-0.804	-0.725	-0.495	-0.344	-0.915	-0.914	-0.809	-0.713

Table 27: RF out-of-sample portfolio results.

	Anomaly assets				All assets			
Metric	RF5	RF10	RF20	RF40	RF5	RF10	RF20	RF40
Mean weekly return	0.0002	0.0015	0.0019	0.0029	-0.0045	-0.0039	-0.0004	-0
WRC p-value	0.6556	0.3843	0.1948	0.0427	0.9714	0.9667	0.7121	0.6363
Standard deviation	0.054	0.044	0.036	0.031	0.065	0.051	0.044	0.038
Skewness	0.085	0.333	0.169	0.357	0.635	0.462	0.533	0.189
Kurtosis	2.823	3.61	3.018	4.133	3.827	3.526	3.762	2.992
Sharpe ratio	0.027	0.244	0.39	0.669	-0.495	-0.549	-0.067	-0.005
VaR 95	-0.084	-0.066	-0.056	-0.049	-0.103	-0.082	-0.064	-0.059
VaR 99	-0.136	-0.107	-0.093	-0.073	-0.174	-0.143	-0.115	-0.099
ES 95	-0.121	-0.097	-0.079	-0.068	-0.145	-0.118	-0.098	-0.09
ES 99	-0.173	-0.138	-0.111	-0.094	-0.195	-0.156	-0.131	-0.122
Annualized return	-0.063	0.028	0.07	0.134	-0.292	-0.235	-0.069	-0.038
Maximum drawdown	-0.852	-0.67	-0.537	-0.354	-0.964	-0.928	-0.77	-0.701

Table 28: GBT out-of-sample portfolio results.

	Anomaly assets				All assets			
Metric	GBT5	GBT10	GBT20	GBT40	GBT5	GBT10	GBT20	GBT40
Mean weekly return	0.0014	0.0019	0.0026	0.0029	-0.0038	-0.0027	-0.0015	-0.0003
WRC p-value	0.5011	0.2952	0.1155	0.0546	0.9621	0.9312	0.863	0.6947
Standard deviation	0.056	0.045	0.037	0.033	0.086	0.064	0.051	0.041
Skewness	-0.008	0.093	0.283	0.39	1.569	1.73	0.999	0.619
Kurtosis	2.397	1.633	2.923	4.437	10.214	14.081	6.451	4.238
Sharpe ratio	0.183	0.315	0.509	0.636	-0.313	-0.306	-0.215	-0.05
VaR 95	-0.08	-0.069	-0.057	-0.048	-0.123	-0.097	-0.074	-0.061
VaR 99	-0.143	-0.106	-0.087	-0.073	-0.208	-0.159	-0.135	-0.112
ES 95	-0.119	-0.097	-0.08	-0.069	-0.18	-0.131	-0.108	-0.09
ES 99	-0.187	-0.136	-0.105	-0.095	-0.239	-0.175	-0.15	-0.12
Annualized return	-0.009	0.051	0.107	0.13	-0.317	-0.217	-0.134	-0.057
Maximum drawdown	-0.893	-0.604	-0.452	-0.384	-0.964	-0.921	-0.86	-0.722

Table 29: FFNN out-of-sample portfolio results.

	Anomaly assets				All assets			
Metric	FFNN5	FFNN10	FFNN20	FFNN40	FFNN5	FFNN10	FFNN20	FFNN40
Mean weekly return	-0.0003	0.0015	0.0025	0.0027	0.0007	0.0013	0.0007	0.0016
WRC p-value	0.7218	0.3174	0.1021	0.0693	0.6924	0.5314	0.5354	0.2934
Standard deviation	0.048	0.038	0.034	0.033	0.074	0.057	0.047	0.04
Skewness	-0.014	-0.143	0.194	0.383	1.697	1.048	0.7	0.885
Kurtosis	4.244	4.878	4.448	5.52	11.592	5.376	3.623	5.381
Sharpe ratio	-0.042	0.29	0.528	0.589	0.064	0.161	0.112	0.295
VaR 95	-0.074	-0.062	-0.051	-0.049	-0.105	-0.074	-0.066	-0.057
VaR 99	-0.134	-0.084	-0.083	-0.079	-0.154	-0.133	-0.117	-0.09
ES 95	-0.115	-0.082	-0.075	-0.071	-0.138	-0.109	-0.097	-0.082
ES 99	-0.176	-0.124	-0.111	-0.105	-0.198	-0.167	-0.142	-0.107
Annualized return	-0.073	0.042	0.105	0.118	-0.096	-0.017	-0.019	0.045
Maximum drawdown	-0.799	-0.566	-0.364	-0.324	-0.697	-0.724	-0.636	-0.478

Table 30: LSTM network out-of-sample portfolio results.

	Anomaly assets				All assets			
Metric	LSTM5	LSTM10	LSTM20	LSTM40	LSTM5	LSTM10	LSTM20	LSTM40
Mean weekly return	0.0052	0.0037	0.0033	0.0034	0.0037	0.0031	0.0025	0.0025
WRC p-value	0.0453	0.0519	0.0478	0.0176	0.1093	0.0704	0.0849	0.0542
Standard deviation	0.05	0.04	0.035	0.031	0.047	0.036	0.032	0.029
Skewness	0.385	-0.002	0.213	0.441	0.756	0.373	0.401	0.298
Kurtosis	2.694	1.631	2.548	2.973	3.884	1.731	2.81	2.832
Sharpe ratio	0.744	0.681	0.684	0.786	0.57	0.616	0.561	0.616
VaR 95	-0.076	-0.059	-0.051	-0.046	-0.066	-0.054	-0.047	-0.045
VaR 99	-0.134	-0.112	-0.097	-0.075	-0.106	-0.087	-0.078	-0.076
ES 95	-0.104	-0.086	-0.076	-0.064	-0.089	-0.071	-0.066	-0.063
ES 99	-0.143	-0.123	-0.104	-0.087	-0.12	-0.092	-0.085	-0.084
Annualized return	0.227	0.166	0.151	0.165	0.146	0.135	0.108	0.113
Maximum drawdown	-0.196	-0.346	-0.359	-0.312	-0.59	-0.553	-0.5	-0.382

Table 31: MFCM method with linear activation function out-of-sample portfolio results.

	Anomaly assets				All assets			
Metric	$\mathrm{MFCM^L}5$	$\mathrm{MFCM^L}10$	$\mathrm{MFCM^L}20$	$\mathrm{MFCM^L}40$	$\mathrm{MFCM^L}5$	$\mathrm{MFCM^L}10$	$\mathrm{MFCM^L}20$	$\mathrm{MFCM^L}40$
Mean weekly return	0.008	0.005	0.0031	0.0032	0.0025	0.0033	0.0019	0.0024
WRC p-value	0.0001	0.003	0.0162	0.0096	0.2226	0.0579	0.1575	0.0616
Standard deviation	0.044	0.035	0.028	0.026	0.043	0.036	0.03	0.027
Skewness	0.98	0.43	0.172	0.395	0.524	0.433	-0.182	-0.109
Kurtosis	4.188	2.164	2.494	4.317	2.55	2.548	1.481	2.317
Sharpe ratio	1.316	1.046	0.8	0.866	0.414	0.674	0.454	0.631
VaR 95	-0.062	-0.054	-0.044	-0.037	-0.064	-0.054	-0.047	-0.042
VaR 99	-0.08	-0.074	-0.064	-0.061	-0.098	-0.087	-0.085	-0.071
ES 95	-0.077	-0.068	-0.059	-0.054	-0.085	-0.073	-0.068	-0.06
ES 99	-0.096	-0.09	-0.083	-0.077	-0.115	-0.091	-0.096	-0.088
Annualized return	0.443	0.259	0.153	0.158	0.084	0.151	0.077	0.11
Maximum drawdown	-0.357	-0.287	-0.322	-0.288	-0.543	-0.514	-0.446	-0.409

Table 32: MFCM method with optimal activation function out-of-sample portfolio results.

	Anomaly assets				All assets			
Metric	$\text{MFCM}^{\text{O}}5$	$\text{MFCM}^{\text{O}}10$	$\text{MFCM}^{\text{O}}20$	$\text{MFCM}^{\text{O}}40$	$\text{MFCM}^{\text{O}}5$	$\text{MFCM}^{\text{O}}10$	$\text{MFCM}^{\text{O}}20$	$\text{MFCM}^{\text{O}}40$
Mean weekly return	0.0043	0.004	0.0036	0.0034	0.005	0.0048	0.004	0.0032
WRC p-value	0.1611	0.0448	0.0329	0.0271	0.0049	0.001	0.0025	0.0089
Standard deviation	0.06	0.042	0.035	0.033	0.036	0.029	0.027	0.025
Skewness	3.787	1.746	0.694	0.886	0.713	0.899	0.692	1.036
Kurtosis	42.212	15.502	5.249	8.925	6.158	4.932	6.202	8.201
Sharpe ratio	0.514	0.692	0.722	0.74	0.995	1.172	1.07	0.909
VaR 95	-0.071	-0.058	-0.053	-0.044	-0.052	-0.038	-0.039	-0.037
VaR 99	-0.116	-0.091	-0.09	-0.088	-0.065	-0.059	-0.06	-0.055
ES 95	-0.109	-0.079	-0.074	-0.07	-0.067	-0.053	-0.054	-0.05
ES 99	-0.158	-0.118	-0.109	-0.108	-0.1	-0.071	-0.072	-0.07
Annualized return	0.149	0.179	0.165	0.159	0.254	0.253	0.206	0.161
Maximum drawdown	-0.424	-0.267	-0.277	-0.3	-0.37	-0.348	-0.328	-0.339

A.3 Code Guide

The code for this thesis is separated into six parts:

- Code part 1_Anomaly analysis
- Code part 2_Data
- Code part 3_Functions
- Code part 4_Analysis with six standard models
- Code part 5_Analysis with MFCM
- Code part 6_Results

Additionally, each part contains subsections which are denoted by a hashtag followed by either 5 or 14 hyphens, such as: #———— Subsection ————#. The control find function for "#—–" can be used to scroll through the subsections. The subsections in each part of code are given as follows:

Code part 1_Anomaly analysis

#———— Set up Russell 3000 index membership lists ————#
#———— Get market capitalization data for size anomaly ————#
#———— Get book value of equity data for value anomaly ————#
#———— Get monthly return data for momentum anomaly ————#
#———— Form monthly decile anomaly portfolios ————#
#———— Get pricing data ————#
#———— Clean data ————#
#———— Use delisting codes for returns after delisted ————#
#———— Continue cleaning data ————#
#———— Make asset return dataframes ————#
#———— Analyze asset anomaly returns out-of-sample ————#
#———— 6 month and 1 year holding periods for anomaly returns ————#
#———— Momentum requires new calculations and assets ————#

#————— 6 month and yearly asset anomaly returns out-of-sample —————#
#————— Regression analysis of plots using weekly return data —————#
#————— Intra week returns of 6 and 1 year momentum anomaly —————#
Code part 2_Data
Code part 3_Functions
#————— Neural networks —————#
#————— HYPERBAND —————#
Code part 4_Analysis with six standard models
#————— Anomaly assets —————#
#—– Validation with HYPERBAND —–#
#—– Testing; retrain on both training and validation sets —–#
#—– Implement trading strategy —–#
#————— All assets —————#
#—– Validation with HYPERBAND —–#
#—– Testing; retrain on both training and validation sets —–#
#—– Implement trading strategy —–#
#————— Forecasts for accuracy analysis —————#
Code part 5_Analysis with MFCM
#————— Anomaly assets —————#
#—– Set up data —–#
#—– Validation with HYPERBAND for frequency forecasting models —–#
#—– Construct training data for activation function —–#
#—– Validation with HYPERBAND for activation function —–#
#—– Train final MFCM activation function models before testing —–#
#—– Implement trading strategy —–#
#————— All assets —————#
#—– Set up data —–#
#—– Validation with HYPERBAND for frequency forecasting models —–#
#—– Construct training data for activation function —–#
#—– Validation with HYPERBAND for activation function —–#
#—– Train final MFCM activation function models before testing —–#
#—– Implement trading strategy —–#
#————— Forecasts for accuracy analysis —————#
Code part 6_Results
#————— Portfolio Analysis —————#
#————— Null hypothesis graph —————#
#————— Plotting cumulative return graphs for optimal MFCM with all assets —————#
#————— Risk Analysis —————#
#————— Accuracy Analysis —————#
#————— Hyperparameters and runtime table —————#