UNIVERSITY OF TWENTE.

Faculty of Electrical Engineering,
Mathematics & Computer Science

ORTEC FINANCE

Machine Learning Applications in Financial Advisory

Ruxandra Vulpoiu

M.Sc. Thesis
October 2018

Supervisors:
dr. Stefano Schivo
dr. Mannes Poel

dr. Martin van der Schans
Jelle Verstegen

Summary

This study is done in collaboration with Ortec Finance and in completion of Masters Degree in Computer Science at University of Twente.

The goal of the study is to explore machine learning and recommender systems capabilities in the task of financial advisory. More specifically, we focus the study on prediction of analyst rating assessed by Bloomberg and on construction of diversified portfolios of stocks by means of clustering. The following research questions guide this study:

1.To what extent can the buy, hold, sell recommendations list of financial analysts be automatically generated from easily accessible financial data, using simple machine learning methods?

2.How can we design recommender systems for financial advice by using a data driven approach?

3.To what extent can machine learning and recommender engines methods support financial advisers in the process of portfolio creation, tailored to individuals?

First, a prediction module uses supervised learning to predict analyst *buy/hold/sell* recommendations, also known as analyst rating. In the end system this would be used as input for the second module. We use regression algorithms such as Linear Regression with Lasso Regularization, Random Forests and Gradient Boosting and ensemble classification algorithms such as AdaBoost and Bagging with Decision Trees and Support Vector Machines with One vs Rest multiclass classification. We took a novel approach by clustering the stocks prior to estimation and using the cluster number as feature. The best results are given by this method applied on Support Vector Machines, with a micro-average F1-score of 72%. The Ensemble methods prove their efficiency, but we believe that results could be better predicted with a more accurate target label and more various sources of input, such as news and public statements as financial analysts use in real life. The correlation between the accuracy of prediction models and returns gained by following their prediction has not been sufficiently studied yet.

Second, a portfolio construction module uses clustering to recommend diversified portfolios. Twenty portfolios are constructed using clustering with different settings. The diversification constraint is implemented by design: the clustering

technique groups similar stocks together and by picking one stock from each cluster a mix of diversified (dissimilar) stocks is created. K-means, Agglomerative and Spectral algorithms are used on the distance matrix obtained from the correlation matrix of stocks. K-means and Agglomerative are used also on a two dimensional data set of encoded features Sector and Region. Various methods of stock selection from each cluster are explored. The variance of the portfolios created by the proposed methods is slightly higher than benchmark, meaning that the portfolios created take more risk. Variance fluctuates between 0.1 and 1 in the first 6 months of holding and increases until at most 3.1 by the end of the first year. Returns are consistently higher than benchmark. The values of returns also vary, reaching 10% in the first 6 months and 31% by the end of the first year. Sharpe Ratio puts the quality of the portfolios into perspective and in this case varies between -0.10 and 7. The sector and region clustering resulted in 82.8% to 80.3% similarity of portfolio sector allocation, while correlation clustering gave high similarity score in terms of portfolio region allocation, with values between 79.5% and 73.16%. These similarities are calculated against a benchmark allocation of MSCI World Index. In terms of diversification the clustering technique could not have performed better. Financial performance is too volatile from the perspective of advisers and it is difficult to asses prior to investing what portfolio will perform well. More research could solve this issue. Addition of a fail-safe mechanism is considered in this work as a promising solution.

The study extends the research done so far and opens new perspectives in the task of prediction of analyst rating. It also proved successful in integrating machine learning methods in recommender systems that respect constraints of the domain. The system was highly evaluated by stakeholders in aspects of diversification and speeding up the advisory process. The complete task of portfolio construction includes diversification across types of securities, thus the applications of the study could be extended.

Contents

List of Figures

List of Tables

List of Acronyms

ANR Analyst Rating

MDS Multidimentional Scaling

MPT Modern Portfolio Theory

MSE Mean Squared Error

RSS Residual Sum of Squares

Chapter 1

Introduction

1.1 Motivation

Financial advisers have the complex role of managing multiple investors' wealth. To make proper financial recommendations, they need thorough periodical analysis on the performance of each company and changes in the market, in addition to following the financial situation of each client and the progress of their individual portfolio. It is an elaborate task, thus we propose an automated decision support tool that aids them with financial recommendations and portfolio construction.

Financial analysts perform financial research and periodically rate the future performance of companies. This is also a challenging task, prone to subjectivity and mistakes [1]. As the market expands, so do the information and factors needed to be taken into consideration for prediction, making the analysts' job even more complex. Moreover, there is no standard method of rating companies, meaning that we do not know how to program a system to do specifically this task. Machine learning can deal with these drawbacks and it is specifically designed for data intensive tasks such as ours.

In portfolio construction, advisers use the analyst ratings to asses the future performance of investments. Furthermore, they have to take into consideration investment best practices, such as diversification, and create the right mix of assets for the particular needs of each investor. This is also a data intensive task with no specific methodology. Previous literature has successfully used unsupervised learning methods for construction of diversified portfolios.

The resulting system will recommend a mix of diversified stocks for portfolio construction. As we implement the constraints of the application domain described in section 2.3.2 through the design of the machine learning method we use, the end result will be a constraint-based recommender system integrating machine learning.

Advisers usually take on wealthy clients exclusively, with an high initial investment capital. These practices are justified by the considerable amount of effort and

knowledge needed for managing other people's wealth. As each portfolio has different characteristics and needs to be managed individually, there is a limit in the number of clients a financial advisor can manage. Our proposed system could also contribute in speeding up the process of financial advisory.

1.2 Framework

The project is done in collaboration with Ortec Finance. Ortec Finance provides software for advisers of financial institutions and assists them with financial decision making. The results of this exploration should aid Ortec Finance in their development decisions by assessing the capabilities and limitations of machine learning integration in products for financial advisory.

1.3 Goals

The project explores the learning capabilities of machine learning algorithms in automation of the financial advisory processes. More precisely, we focus on the portfolio construction task.

The first part of the study will explore weather we can easily automate the task of the financial analysts of issuing buy/hold/sell recommendations. In the second part we construct diversified portfolios using the clustering technique. The main objective of the study is to construct a recommender system that integrates data driven technologies, such as machine learning. Two secondary research questions will guide this study:

To what extent can the buy, hold, sell recommendations list of financial analysts be automatically generated from easily accessible financial data, using simple machine learning methods?

How can we design recommender systems for financial advice by using a data driven approach?

Design is evaluated by analyzing whether the portfolios created using the data driven approach have comparable financial performance with the portfolios made by our benchmark advisor. This should answer the main research question of the thesis, specifically:

To what extent can machine learning and recommender engines methods support financial advisers in the process of portfolio creation, tailored to individuals?

The end goal is to build a decision support tool that provides advisers with a diversified mix of stocks. As the system implements the constrains of application

domain described in section 2.3.2, and it can also implement investor preferences as we will see in chapter 4, the end result will be a constraint-based recommender system integrating machine learning.

1.4 Report organization

This is a multidisciplinary study, applying technologies from Computer Science in the consulting area of Financial domain. Thus we organized the presentation of the theoretical background as follows: in Chapter 2 we introduce the financial areas that are of interest for this study; we construct our system from two parts, described in Chapters 3 and 4 respectively. The theory behind the technological techniques used are introduced in the beginning of each chapter. Chapter 3 concerns prediction of analyst rating. Then, in Chapter 4, we use machine learning to create a diversified mix of stocks. Finally, in Chapter 5 conclusions and recommendations are given.

Chapter 2

Background

In this chapter we present the theoretical background of this study.

We start by introducing the domain of trading and financial advisory. Section 2.2 presents the analyst rating and the common assessment methods. This information will help us further in understanding the application domain of Chapter 3. Furthermore, section 2.3 dives into the theory and guidelines of portfolio construction, along with the mathematical framework stock selection is based on. This is the application domain of Chapter 4.

We conclude this chapter by giving an overview of the technological frameworks we use in this study: machine learning and recommender systems.

In Chapters 3 and 4 we apply machine learning methods on the financial domain. The theoretical background behind the applied techniques are presented in a dedicated section in the chapters they are used.

2.1 Introduction

Wealthy investors hire professional financial advisers to manage their wealth in exchange for a part of it. These advisers act as a consultant to their clients. The financial analysts is the one who provides the research and analysis necessary to formulate the investment advice. The results of his analysis are presented in the form of rating for each considered company. We included this analyst rating in this study as it is a commonly used input in the financial advisory process. The first part of the study targets prediction of analyst rating, with the end goal of automation of financial advisory process.

The job of financial advisers is to asses a client's financial situation, risk aversion and investment objectives and make recommendations accordingly [2]. The advice includes construction of investor's portfolio and changes applied to it over time. More specifically, it concerns which securities to pick, how much of each and holding

period. In these terms we say that an advisor manages a portfolio. The second part of this study focuses on selecting the proper set of stocks for the portfolio creation step.

2.2 Analyst Rating

Financial advisers need to asses the future performance of companies before deciding to further recommend them to investors. For this, they need thorough periodical analysis. That is what financial analysts do. They reflect the conclusions of their analysis in a single value, analyst rating.

To produce analyst rating, financial analysts perform research on the economical performance of companies listed on the market. Their main source of information are the publicly released financial statements, such income statement, balance sheets and cash flow statements. They may also collect information by participating in public conference calls and following the trends of markets and industries [3]. This study focuses on a single method of information elicitation by the financial analyst, namely the financial statements.

From their investigation and experience, analysts rate companies and issue investment recommendations. The ratings may be presented as categorical labels, namely *buy*, *hold* or *sell* or as a rating, on a scale from 1 to 5, with values closer to 1 representing a *sell* advice. The list is updated daily and the labels and assessment methods may differ from analyst to analyst. Most commonly, a *buy* label indicates expected excess returns of at least 10% relative to the market, *hold* label indicates expected returns between 0 and 10% and a *sell* label announces expected loss. For others, the labels are interpreted relatively to well known Indices such as S&P500: a *buy* label signifies potential of outperforming the Index by more than 20%, *hold* means that the company is following the Index and *sell* means under-performance relative to Index [4].

An Index is a hypothetical portfolio, representing a sample of the market and a benchmark for investors.

Advisers use the analyst rating as input when assessing which companies should be recommended to investors. In chapter 3, we aim at predicting this value using supervised methods of machine learning.

2.3 Portfolio Construction

A portfolio is a set of financial assets representing investments in stocks, bonds, commodities, currencies and diverse types of funds. These financial assets can

also be found under the name of securities. Our study focuses solely on stocks. These offer the investor ownership rights on profits and assets of a small part of the emitting company. Market capital expresses public financial opinion of the value of the company and can also be used as an indicator of size. Depending on market capital, companies sells a number between 10,000 and 1,000,000 small parts of themselves, also named shares. An investor purchases a number of shares and stores them in one of his portfolios. The percentage of the respective portfolio the shares occupy is called exposure. There is an enormous variation of types of stocks. They are commonly classified per type of economical activity: sectors and industry. Sectors are sections of economy that group companies with similar products and services. Industry is similar, but interpreted as a subclass of Sectors. Recently, groupings per regions were introduced, as to highlight the potential of emerging markets [5].

Investors purchase securities expecting to make a certain amount of return, that is the amount of money that came from the investment made after a period of time. It can be either positive, representing profit, or negative, expressing loss. The chance of that return not ending up at the expected value is named risk. A common way to express risk is as variance of returns.

In determining the risk and return trade-off of financial assets, experts look at company's financial indicators such as volatility and beta, among many others. Volatility measures the variation in prices of an asset over a determined period of time. It is calculated as standard deviation of returns or squared root of variance. The beta indicator describe the volatility of an asset relative to the market. It's value depends on the chosen benchmark. A common practice is to take one of the major indexes as benchmark, such as S&P500 or Dow Jones. A beta value greater than 1 shows a more risky asset compared to market benchmark. For example, a beta of 1.1 means that the respective security is predicted to return 10% more during good market and lose 10% more compared to benchmark when the market is down. A value below 1 is attributed to less volatility that the benchmark. Continuing with our example, if beta would have a value of 0.8, it means that the respective security is 20% less volatile than benchmark, returning 20% less in good markets and 20% more than benchmark in bad markets.

2.3.1 Modern Portfolio Theory

Modern Portfolio Theory is the most widely accepted mathematical framework for portfolio construction. It was pioneered by Harry Markowitz in 1952 in a Nobel winning paper *"Portfolio Selection"* [6]. Before him, investors were only focusing on the prices and returns of securities, chasing undervalued stocks with fundamental

analysis. Markowitz was the first to consider the risk of a portfolio in a trade-off with returns. He assumes that investors are rational and risk averse, meaning that between two portfolios with the same level of returns and different levels of risk, an investor will choose the portfolio with the lower level of risk. Thus, the rational investor will expect higher returns if he is willing to take on more risk. Markowitz defined the returns of portfolio as:

$$R_p = \sum w_i R_i \tag{2.1}$$

where R_p are the returns of the portfolio, R_i is the return on asset i and w_i is respective asset's exposure. Similar to expected value as defined by probability theory [7], expected returns of portfolio are defined as $E(R_p) = \sum w_i E(R_i)$, where $E(R_p)$ is the expected returns of the portfolio, $E(R_i)$ is the return on asset i and w_i is respective asset's exposure. Markowitz was interested in finding the variance of the portfolio's expected returns, thus approximating the risk of the portfolio.

Variance is defined as the average squared deviation of the expected return. As returns are expressed as a weighted sum, the expected value of a weighted sum is the weighted sum of the expected values [7]. To express the variance of a weighted sum, Markowitz uses the concept of covariance.

Covariance is a statistical measure that describes the linear relationship between two variables. In our case, if covariance has a positive value it means the returns of the respective assets vary in the same direction. Conversely, a negative sign shows opposite directions for returns, meaning that when one asset creates profit, the other one creates a loss. To calculate covariance we use the formula:

$$cov(i, j) = \beta_i \beta_j var(benchmark) \tag{2.2}$$

where β is the beta coefficient described in section 2.3 and $var(benchmark)$ is the volatility of the benchmark, may it be a major index or the market. We choose S&P500 as benchmark as it is a popular choice in literature.

The value of covariance is difficult to interpret as it depends on the scale of the variables. For that reason, we use the normalized covariance, also known as Pearson correlation coefficient. To calculate the correlation coefficient we use formula:

$$\rho_{ij} = corr(i, j) = \frac{cov(i, j)}{\sigma_i \sigma_j} \tag{2.3}$$

where σ is the standard deviation of returns for a determined period of time and cov the covariance between respective assets.

Thus, the variance of the overall portfolio as defined by Markowitz is:

$$\sigma_p^2 = \sum_i w_i^2 \sigma_i^2 + \sum_i \sum_{j \neq 1} w_i w_j \sigma_i \sigma_j \rho_{ij} \tag{2.4}$$

where σ_p is the variance of the portfolio, σ_i is the standard deviation of periodic returns of asset i and ρ_{ij} is the correlation coefficient between assets i and j. The first part of the equation expresses the individual contribution of assets i in the total risk of the portfolio. The second part concerns the combined risk contribution of assets i and j. From the formula we see how the value of the correlation influences the variance of the portfolio:

- if ρ_{ij} is 1 then nothing changes
- if ρ_{ij} is close to 0 we discard the second term altogether, though decreasing variance
- if ρ_{ij} has a negative sign we have something to subtract from the first term, thus the variance decreases even more

Thus, an investor can pick a portfolio of diverse stocks that together decrease the risk of not getting expected returns. We will see in the following section how financial advisers put these concepts into practice.

2.3.2 Risk & Diversification

Investors have to consider two types of risk: *systematic risk*, also known as *market risk*, representing the variance of the market returns caused by uncontrollable factors, such as interests rates, recession or natural disasters; and *unsystematic risk*, also referred to as *idiosyncratic risk* or *diversifiable risk*, attributed to uncertainty of a specific asset, sector or region. Although there are ways to mitigate against systematic risk, these methods do not always guarantee results. Mitigating against unsystematic risk is what every investor should do and this can be achieved, according to modern portfolio theory, through diversification.

Diversification means holding a variety of investments to protect the total returns of the portfolio in case one company or economic area drops in value. The variation might be across types of investments such as stocks, bonds, funds and cash, or across sectors, regions or industries in case of stocks. For example between years 2005 and 2015, the Health Care sector brought returns of 192% to the S&P500 index, the Utilities sector had returns of 115%, while the returns of sector Financials only covered 5%. During the stock market crash of the summer of 2011, the S&P500 index dropped 18.3%: the Utilities sector registered a drop of 2.9%, the Health Care sector fell by 13.5% and the Financials dropped by 26.4% [8]. If the index would have been invested only in sector Financials, it would have suffered greatly. From here, financial advisers have created the rule of diversifying a portfolio on types of assets held and on sectors and industries of economy. Diversification across regions

was also introduced recently as countries develop differently and have particular economical potential.

From Modern Portfolio Theory we can deduct that diversification can be achieved by mixing assets that do not have high covariances among them [6].

Financial advisers mitigate investment risk by following guidelines directed by their employer, known as *rulebooks*. Usually, these are kept private. Very common ones, confirmed in consultation with advisers collaborating with Ortec Finance, are:

1. Having an exposure of maximum 5% per security
2. Diversifying assets across sectors and regions

In chapter 4 we implement both views of diversification with the help of machine learning. The aim of the system is to recommend a diversified mix of stocks. As benchmark for a correct sector and region allocation, we refer to the allocation of MSCI World Index[1], detailed in section 4.5.

We use rules from the rulebook as constraints for our system, thus creating a constraint-based recommeder system. These are introduced in the following section. In chapter 4 section 4.4 we explain how machine learning implements the required constraints by design.

2.4 Machine Learning & Recommender Systems

Derived from pattern recognition, computational statistics and mathematical optimization, machine learning aims at solving a problem without being specifically programmed on how to do so. Contrary to traditional approaches where models are explicitly specified, machine learning models aim at estimating a function from data and incrementally adapt it by minimizing errors, hence the term *learning*. This methods efficiency at a given task is dependent on the data it processes and learns from. On the other hand, the availability of data dictates what techniques would give the best results. In this study, we use machine learning for prediction of analyst rating and for creating a diversified mix of stocks.

Recommender systems [9] developed from information retrieval domain and statistics. They provide a means to filter vast amount of data (movies, songs, products) according to the needs and preferences of the user. As a vastly researched area, few techniques are commonly used:

Content-based Filtering (CB) suggests new items that are similar to those preferred in the past by the user. Machine learning algorithms may be used to predict the preference of the user towards an unrated item or to learn the profile of the user

[1] `https://www.msci.com/world`

from the interaction with the system. Popular applications[28] incorporating Content Based Recommenders are Yahoo! News, The Music Genome Project that sustains Pandora radio[29] and Casper subsystem for JobFinder [10].

Collaborative Filtering (CF) calculates the similarity between users and recommends what people alike preferred. Some of the real life applications based on Collaborative Filtering Recommenders include Google News [11], Amazon [12], YouTube [13] and Netflix [14].

Knowledge-based recommender systems have developed more recently. They incorporate domain knowledge and specific user requirements into the resulted recommendations. There are two classes of knowledge-based recommender systems: *case-based* and *constraint-based* engines. In the former one, the engine keeps track of the solutions it made in the past and when a new request is made, it looks for similar cases in its database and aims at adjusting the past solution to the new requirements. In the latter case, user and item properties are defined along with specific compatibility restrictions. These constraints are domain specific and give rise to *knowledge bottleneck* issues as software developers find it difficult to acquire and translate domain specific rules into programmable conditions.

In this study we experiment with integration of investor preferences when constructing a diversified mix of stocks, as it is desirable from such systems to consider the individual desires of the client. We evaluate the methodology with and without integration of user preferences. The constraints of the domain are not explicitly specified. They are implemented by the design of the machine learning method used.

Chapter 3

Prediction of analyst ratings

3.1 Introduction

In this chapter we aim at predicting the analyst rating defined in section 2.2. This is a data intensive task and with no clearly defined methodology. For these reasons, we use machine learning. There has been successful research in this field, summarized in Section 3.3. Our aim is to extend on this work, while keeping the task simple.

Firstly, we use regression to predict the analyst rating from four carefully selected sets of features. We analyze the issues encountered and adjust our approach. In a second step we select the stocks from only one sector and use multi-class classification for prediction of analyst rating. We have a more persistent issue of data imbalance in this case. In a third step we perform more analysis on data.

The section starts by presenting theoretical background of supervised learning. We continue with presenting previous research done in the field. Furthermore, the steps taken in the experimental procedure are explained, followed by results. The section concludes with discussion on obtained results and future recommendations.

3.2 Background: Supervised Learning

Supervised learning is used when the results of the task are known in advance. This expected output is called target variable and is denoted by variable y. The input variables are also known as explanatory variables, features or attributes. Supervised learning algorithms aim at learning the mapping from features to targets where the error is minimal. The mapping has the form of a function:

$$y = f(X) \tag{3.1}$$

where X is the input data, usually a set of vectors, and y is the target variable, known in advance.

Depending on the type of the target variable, a distinction between classification and regression is made:

- In classification, the target variables are categorical, also known as labels. In our case, as described in section 2.2, these labels would be *strong sell*, *sell*, *hold*, *buy* or *strong buy*. Classification algorithms try to find patterns and rules that separate the instances with different labels, also known as decision boundaries. By finding the decision boundary closest to reality, also seen as having the minimum error, each sample point is fitted in one of the target classes. Evaluation of performance is made by comparing the values of prediction for each data sample with the corresponding initially known target output. Common metrics calculate various ratios between correctly and missclassified instances.

- In regression we try to predict a continuous variable. The model will aim at fitting a line as close as possible to the trend the data follows, thus evaluation is assessed by calculating distances between the fitted line and the sample points.

The task of the supervised learning algorithms is divided in two phases: a training phase and a prediction phase.

- In training a subset of data samples is given to the algorithm, along with the respective target variables. The goal of this phase is to create a predictive mathematical model that maps the input to the output. During multiple iterations, the model makes predictions on the given subset of data samples, calculates the error, meaning the difference between it's predictions and previously known target variables, and makes an adjustment to it's mathematical model. The objective of the algorithm is to minimize the error. When the errors stop minimizing from one iteration to another, we say that the algorithm converges. The model of the iteration with least error will be given as solution.

- In the prediction phase we give the model new data, unseen samples, and evaluate the new predictions the trained model makes against the respective target output.

3.2.1 Models

There is a great variation of supervised learning algorithms and each variation of training data, algorithm and pre-set parameters creates a different model. Choosing the best solution involves adaptation to input data, followed by trial of different parameter settings. Analysis of data becomes an important step. Algorithms differ in

the assumptions they make about the underlying structure of the input data, more specifically in the form of the function f and in the form of the error function they minimize.

In the first part of this chapter, we use regression to predict the analyst rating. We select four regression algorithms, covering both linear and non-linear cases: Linear Regression, Lasso, Gradient Boosting and Random Forests. In the second part of this chapter we also experiment with classification for prediction of analyst *buy/hold/sell* labels. For this task, we selected four classification algorithms: Decision Tree, AdaBoost, Bagging and Support Vector Machines.

We shortly present the methods and algorithms used:

Linear Regression is a well know algorithm, heavily used in statistics. Given a data set $x_1, x_2, ..., x_n$ comprising of N observations and their corresponding target variables $y_1, y_2, ..., y_n$, the goal is to find a linear function of the form (3.2) by estimating the coefficients $w_0, w_2, ..., w_n$ (with w_0 being the intercept) so that the constructed model is able to predict the value of $\hat{y}$ for new (unseen) values of x.

$$y = w_0 + w_1 x_1 + ... + w_n x_n \tag{3.2}$$

When approximating the w coefficients, the objective of any machine learning algorithm is to minimize the error, also known as cost function J. In case of Linear Regression, the error is measured by the Mean Squared Error and is defined as:

$$J = \frac{1}{N} \sum_{i=1}^{n} (y_i - \hat{y}_i)^2 \tag{3.3}$$

Lasso is developed from Linear Regression by addition of L1 regularization parameter to the cost function J to improve generalization. The regularization parameter adds a penalty to features for the model and it may shrink some of the coefficients to zero. For this reason, Lasso algorithm can also be used for feature selection. The degree to which coefficients will be penalized can be varied through the parameter α. The objective becomes minimizing the cost function:

$$J = \frac{1}{N} \sum_{i=1}^{n} (y_i - \hat{y}_i)^2 + \alpha \sum_{i=1}^{n} |w_i| \tag{3.4}$$

Decision Trees is can be used for both regression and classification tasks [15]. A decision tree is composed of nodes and edges. The inner nodes correspond to queries on features, while the ending nodes, also called leafs, mark the class label assigned to the respective path of the tree. Each internal node splits in two branches according to the decision of the query.

The algorithm initializes with the assignment of a root node, chosen by iteratively splitting at each feature and calculating the error in the splits made. We use the

Gini Impurity Index as a measure for a good split and the objective of the algorithms is to minimize it. Gini Impurity measures of how often a randomly chosen element from the set would be incorrectly labeled if it was randomly labeled according to the distribution of labels in the subset. It follows from formula:

$$Gini = 1 - \sum_{i=1}^{C} (p_i)^2 \tag{3.5}$$

where C is the number of classes and p_i is the probability of an item with label i to be chosen. The feature that produces the lowest gini index is chosen for the root node. Iteratively, this process happens for each node.

Gradient Boosting is an ensemble method that combines the predictions of a sequence of weak classifiers into one single averaged prediction. In our case, decision trees are used. The first model initializes the weights of the coefficients and each following model corrects the residual errors of the previous one. Adding the results of previous predictions with each iteration will result in a more complex function that fits the non linear data better. The method is sensitive to noisy samples as it will try to fit everything with minimum error, including the noise. Thus, a proper stopping point needs to be found.

AdaBoost (Adaptive Boosting) is the first algorithm to use Boosting. Each of the weak learners is fitted on a modified data set, in which, at each iteration, the previously wrongly classified samples receive higher weights. The predictions of each learner are weighted in proportion to their respective error. The final prediction is obtained by summing up the weighted predictions of the weak learners.

Random Forests uses *Bagging* method to improve accuracy. In bagging, we shuffle the training set and iteratively extract a few random samples. Independent trees aim at fitting the chosen sets with each iteration. The final prediction will be an average prediction of all trees. The method is well known for reducing the variance of the prediction with minimum increase in bias. The bias / variance trade off is discussed in section 3.2.2.

Support Vector Machines is an algorithm that can be used both in regression and classification tasks and for both linear and non-linear solutions; it is more commonly used in classification. In the linear case, the solution is given by the hyperplane $w^T x + b = 0$, where b is the intercept coefficient and w is the vector of model's coefficients. The goal is to maximize the distance between the hyperplane and the closest data points to the hyperplane, chosen as "support vectors". The objective function is simplified to minimizing J of the form:

$$J = \min_{w,b} \frac{1}{2} ||w||^2 \tag{3.6}$$

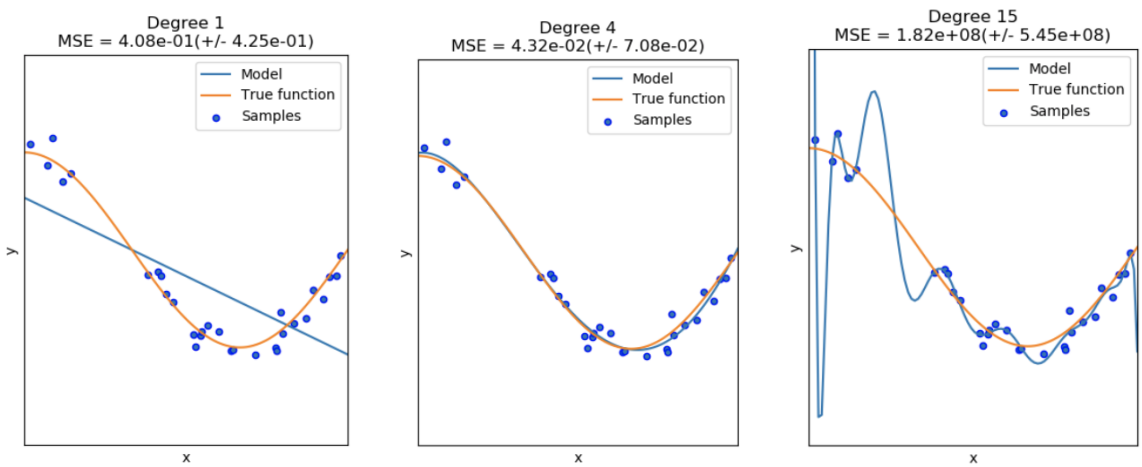
subject to $(\langle x_i, w \rangle + b) \cdot y_i \geq 1$ for every $i = 1, ..., n$, where we have n training samples,

x_i is the ith training sample and y_i is it's target label. For non-linear cases, we use the kernel trick so to fit the model in a transformed feature space.

3.2.2 Generalization of the model

The task of the supervised learning algorithms consists of a training phase and a prediction phase. The challenge is to make correct predictions on a broad spectrum of unseen observations (test set) with a model trained on a specific set of observations (training set). This is what we call generalization of the model.

The issues that can appear in generalization are over-fitting and under-fitting. Over-fitting is the result of high performance when fitting the training data and low performance when fitting the unseen samples in the test data. In this case the model is fitting the training data and the noise too well. We say in this case that the model has high variance. It becomes too specific to generalize over unseen samples. Over-fitting is captured graphically in Figure 3.1(c)[1] At the other end of the spectrum a model could be under-fitting the data, meaning its function is not complex enough to capture the relationships in the data. We may also say that the model is biased, as it fits an unrepresentative data set. In this case the model will badly fit both the training and the test set. The challenge is finding the perfect balance in the complexity of the model and the acceptable compromise for error, as depicted in Figure 3.1(b).



(a) Under-fitting (b) Appropriate fitting (c) Over-fitting

Figure 3.1: Generalization of the model. The case of (a) under-fitting: the model is not complex enough to capture the true function, (b) appropriate fitting: the model is similar to the true function and (c) over-fitting: the model is too complex and introduces more noise.

[1] `http://scikit-learn.org/stable/auto_examples/model_selection/plot_underfitting_overfitting.html`

There are a number of techniques to deal with these issues. In case the model is under-fitting, it has not learned enough. One solution is to increase the number of features, thus giving to the model more variables to learn from. Varying the complexity of the mathematical function that describes the model may allow it to fit data better. An example of the this would be to change an under-fitting linear model to a polynomial form, exemplified by the transition from Figure 3.1(a) to Figure 3.1(b).

In case of over-fitting, the model is too complex to generalize well. As a solution, we may add more correct observations to the training set as to give the model a more representative data set to learn from. Adding more data can be expensive or simply not feasible, our case included. Alternatively, we can reduce model complexity by switching to a more simple mathematical function, as in the transition from Figure 3.1(c) to Figure 3.1(b). Moreover, we can use techniques such as *regularization*, *feature selection* and *hyper-parameter tuning*. Regularization adds a penalization parameter to the objective function of the algorithm, introducing a correction to the miss-classified instances. This penalization can vary in the degree of change it adds, making it suitable for both high bias or high variance cases. Feature selection refers to choosing the ideally sized attribute set for faster computation and minimum information loss. The objective is to eliminate those features that bring noise rather than information to the learning process. Hyper-parameters are the variables that characterize the machine learning models, thus refining their value can improve the fitness of the model with our data.

3.3 Literature review

Advanced computation is already heavily used in trading and more than 50% of the daily average volume of exchanges are done by programmed trading platforms [16]. The competitiveness of trading has motivated many researches to look for more technologically advanced methods such as machine learning to gain advantage over the market and increase investment earnings. Previous research on machine learning applications in performance prediction focus on forecasting the direction of prices, on analysis of market sentiment and risk management, with less research targeting forecasting analyst recommendations.

Milosevic, N. (2016) [17] aims at predicting whether stock prices will go up by 10% in one year time frame. He uses binary classification algorithms, labelling a stock as 'good' if it is estimated to have expected returns of 10% in the next year and 'bad' otherwise. The study uses a balanced data set with 1298 companies and 28 predictors, spanning quarterly from year 2012 to 2015. Out of the classification algorithms used (Decision Trees, SVM, JRip [18], Random Trees, Random Forests, Logistic Regression, Naive Bayes [19], Bayesian Networks [20]) Random Forest per-

forms the best with an F-score of 75.1%. Feature selection is done manually and using trial-and-error method. This is not state of the art in machine learning, but it does slightly improve the F-score of the models, with Random Forest improving by 1.4%. Milosevic's study deduces that using financial information that describes the company's performance in the present in sufficient for estimation of future performance. This hypothesis implies it is not necessary to look at past performance. Milosevic leaves this hypothesis open for more analysis in future research.

Schumaker, R,.& Chen, H. (2010) [21] created an automated trader. Their methodology uses Support Vector Regression to predict the 20-minute discrete price of S&P500 stocks from textual news. They aggregate the news per sector when training and we find this as a good approach as well after the first round of experiments. If the price is predicted to increase by 1% in the 20 minute window after the news release, the system buys short and sells after 20 minutes. The data set for validation consisted of 2,809 news articles and minute prices of S&P500 stocks gathered in 5 weeks time frame in late 2005. They selected features using Proper Nouns method and retention of words that appear multiple times in any one article. Proper Nouns is a method of information retrieval in which we extracts Noun Phrases [22] and Named Entities [23] [24] without a predefined relationship between nouns and categories of named entities. During experimental procedure, the system was given 1000$ to invest with. After a year, it's performance is compared with top ten best quantitative funds and S&P 500 Index. The created system placed itself on the fifth place, with returns of 8.5% after the first year. The authors note that trading on the S&P500 exclusively limits the performance of the system. The better performing traders were buying and selling on a greater market, thus it would be interesting to see the system applied on a greater data set.

Barbosa, R. P. & Belo, O. (2008 & 2010) [25] [26] construct a multi-agent system that is capable of trading autonomously on the Forex market. The Forex market trades currencies only, and for 24 hours per day (compared to the stock market which trades only on week days and from 9:30am until 8pm EST, including after hours). Barbosa's system has three modules: an *Intuition Module* which uses multiple classification and regression algorithms in Ensemble [27] to predict the direction of the price, an *A Posteiori Knowledge Module* which uses Case-Based Reasoning to suggest when and how much to trade and an *A Priori Knowledge Module* that makes the final trading decision by inputs the outputs of the previous modules in a Rule-Based Expert System. The system we are building in this chapter is related to the Intuition Module, as both aim at predicting whether a security is worth investing at a specific time. The Intuition Module includes a variety of classification and regression algorithms, such as Naive Bayes, Decision Trees and Support Vector Machines among others. Features express preferences towards technical financial

analysis, with variables describing time, moving averages, relative strength index, rate of change and different types of returns. The user needs to define himself the algorithms and features to be used when configuring the system, which might present a limitation: the system is meant to trade autonomously, but it is targeted to wealth holders; thus the investor or his advisor need to have machine learning knowledge, which is rarely the case. The class at a previous moment is included in the options set of features. In terms of accuracy of prediction, the Intuition Module is approximately 52-53% successful. The authors argue that even if the accuracy seems low, the profits made from the correct trades exceed the losses made from incorrectly classified trades. Thus, in this specific three module setting, there is an efficient fail-safe mechanism for the miss-classified predictions.

Gerlein, E., McGinnity, M., Belatreche, A.,& Coleman, S.(2006) [28] extend on Barbosa's study and create an autonomous trading system with two agents: a *Trading Agent* that calculates the technical indicators from prices and classifies the direction of the price in the next period, and a *Market Agent* that encodes financial information and keeps track of trades and profits. They analyze the efficiency of simple machine learning algorithms such as K* [29], C4.5 [30], JRip [18], Naive Bayes [19], Logistic Model Tree [31] and OneR [32] for a binary classification of stock price direction (up or down). The returns generated by the algorithm are also considered as an evaluation metric. Accuracy is not as high as desired, with the simples algorithm getting the highest score, 51.6%. In contrast, most of the models were capable of generating profit in stable market conditions. The author observe that machine learning models do not perform well for the period of 2007-2009, a period of high volatility in the market. Periodic retraining does not improve model accuracy, but improves the cumulative returns. The study uses 9 indicators usually used in technical financial analysis. The best performing model uses 5 predictors. In out study we will vary the same small number of features, but the predictors will comprise of fundamental indicators.

To our knowledge, only one study focuses precisely of prediction of analyst recommendations as assessed by Bloomberg and using machine learning. Thakur et al [33] uses supervised learning on time series data spanning quarterly from 2006 to 2015. The data set includes the companies of S&P500 Index. Features describe a preference towards fundamental financial analysis. Macroeconomic indicators are also included, summing up to 100 features used in total. The percentage of change from the previous period for each feature is added as a feature, doubling the feature space. Feature selection is done only with Lasso regularization. The author argues that the label predicted is usually the same as the previous one. For this reason, the benchmark is "the percent of labels perfectly predicted by the previous periods label" [33]. The resulting accuracy is at worst 0.7% below benchmark in case of

Support Vector Machines and at best 4.3% when using Random Forests and Logistic Regression with Lasso Regularization. As in Barbosa's study, the label from previous period is added as feature.

In our study we use forward step-wise regression to select the ideal number of features for each algorithm as opposed to Milosevic's study. Milosevic also leaves an open hypothesis that we will test in our study by choosing data that describes the status of the company in the present, with few attributes that consider short-term past. Schumaker's study uses textual news as input. We consider this idea a good addition to our system for future research as textual news are the first to predict events of higher volatility in prices. As these events do not happen very often and we need a periodic assessment of companies, we will extract our features from data usually used in fundamental financial analysis. Gerlein's study leaves an open question whether the highest accuracy was obtained by the simplest model by "consequence of mere luck" [28]. Our study aims at contributing with more research in this direction by also using simple algorithms such as Linear Regression. Decision Trees and Support Vector Machines also obtained good results in previous studies, thus we will use them to experiment with non-linear cases. Contrary to the studies of Gerlein and Barbosa, our features are chosen from the indicators usually used in fundamental analysis and we will put more focus of tuning the parameters of the models used. We will avoid using the previous label as feature as in the studies of Thakur and Barbosa because we desire an independent prediction for each period.

3.4 Experimental procedure

The section describes steps we took for prediction of financial analyst *but/hold/sell* recommendations (or rating) defined in section 2.2 from cross-sectional data and using basic machine learning models.

First, we use regression to predict analyst rating as taken from Bloomberg, in the form of a continuous variable from 1 to 5, with low values directing towards strong sell and high values towards strong buy. We calculate the ideal number of features for each algorithm. We use various methods for feature selection. In the end, we apply 4 supervised algorithms on 4 subsets of features. We fine tune the chosen algorithms and make the first round of predictions. We analyze the issues encountered and continue with another iteration of experiments.

Secondly, we focus on prediction of analyst rating per single sector. This time we use classification algorithms for the prediction task. We transform the target variable from continuous to nominal. The classification algorithms will now target labels *strong buy, buy, hold, sell, strong sell*. Section 3.4.3 describes the applied transformation. The focus of this iteration of experiments is on dealing better with

the class imbalance problem. We use ensemble methods bagging and boosting with decision tree as the base algorithm. We also use SVM algorithm in ensemble with one-vs-rest classification technique, as described in section 3.4.3.

Thirdly, we do more data analysis by testing the explanatory power of feature used. Assuming that the rating process of the financial analyst is precise, the previously used features are expected to be correlated with returns, as described in section 3.5.

3.4.1 Data & Preprocessing

Data presents the situation of companies at date 22nd of February 2018 and specified past intervals. For example, feature ”volatility 30 days” refers to the value of volatility at the date that is 30 trading days prior to 22nd of February 2018. Trading days exclude weekends. We considered the list of stocks from S&P 1500 as it is a popular choice in literature and it covers 90% of the US market, thus including a diversified selection of differently sized companies.

As Ortec Finance and the advisers they work with adhere to fundamental financial analysis, we also focus the choice of features in accordance to that. Table 3.1 describes each of the chosen variables for the starting data set. These were chosen by exploring the related literature suggestions, by looking at what information real life financial analysts use to issue their recommendations, and by interviewing financial experts at Ortec Finance.

The set of features chosen describe stocks from different points of view. Risk of the stock is measures by volatility and beta. Quick ratio also measures risk, but from perspective of paying short term liabilities. Returns are described at various moments in time to put them into better perspective. Investors realistic gains are covered by various ratios. Assets, gains and market evaluation express the value of the company.

3.4.2 Regression

Data Preprocessing

We scale some of the features to the size of the company they represent. The motivation is to improve relevancy of these features. For example a significant value in gross profit for a start up is more impressive than the same amount for a big company. The features scaled relative to market capitalization are: *inventory turnover*, *revenue*, *gross profit*, *net income*, *operational cash flows* and *total assets*. Next, two new features were created: *Size* and *Price-Sales Ratio (PSR)*. *Size* groups similar companies based on market capitalization and *PSR* refers to the value placed

Table 3.1: List of features used for the prediction of analyst ratings task

Feature	**Description**
Quick ratio	Measures a company's capability of paying short term liabilities from present liquidities
Inventory turnover	How fast a company sells it's inventory items
Revenue	Ratio of the price of a stock and the company's earnings per share
Gross profit	Revenue made from sales after discounting the costs of goods and service the company provides
Net income	The profit of the company in the past period
Operating cash flow	Liquid net income of the company
Earnings per Share	Net income earned per each share in the stock
Price per Earnings	The dollar amount an investor can expect to invest in order to receive one dollar of that company's earnings
Market cap	The total market value of the company expressed in dollars
Total assets	Value of resources and liabilities the company owns
Adjusted beta	Measures the risk of the stock relative to the market. More details in section 2.3.2
Volatility 30 days	Measures the degree of variation of a trading price series over a period of 30 days
Volatility 90 days	Measures the degree of variation of a trading price series over a period of 90 days
Volatility 360 days	Measures the degree of variation of a trading price series over a period of 360 days
Returns last 3 months	Gains or losses for the past 3 months
Returns last 6 months	Gains or losses for the past 6 months
Returns last year	Gains or losses for the past year
Returns last 5 years	Gains or losses for the past 5 years
Size	Market cap binned into sizes and encoded as numbers
PSR	Value placed on each dollar of a company's sales
Analyst rating	Bloomberg average of analyst ratings

on each dollar of company's revenue. The missing values were replaced with zero as they are present regularly in financial data sets and the models need to adapt accordingly.

All features were scaled using Python *Standard Scaler*, which removes the mean of the feature vectors and scales them to unit-variance. 52 samples were removed because they had missing value for the target variable. In addition we removed 136 outliers with a total of 1312 samples remaining.

The data set presents class imbalance. The class imbalance issue is created by high variation in class frequency. To correct this, the training set used in estimation was balanced using over and under sampling. Under sampling is done by randomly removing observations from the more frequent class. Reversely, over sampling refers to randomly replicating minority observations or synthesize a sub-

set of them [2] [34]. The balanced data set did not improve the results, thus this step was discarded. Other approaches on dealing with class imbalance are described in Section 3.4.3 as they will be used in a future step.

Data Analysis & Feature Selection

We explore with four different subsets of features. The first data set illustrates the case of a small number of attributes, 5 respectively. In the other 3 sub sets, the ideal number of features is calculated using Stepwise Forward Selection algorithm [35]: 13 for the Linear Regression model, 10 for Random Forests and 8 for Gradient Boosting.

In the first two subsets, the features are chosen from data analysis. We select the features that show a close to normal distribution. Histograms of selected features after this step are shown in Appendix A, Figure A.1. We then calculate the independent correlation of each feature with the target variable, analyst rating. Table 3.2 presents the first 13 most correlated features, thus these were chose for the second sub set.

Table 3.2: The thirteen highest independent correlations of features with target variable analyst rating

feature	corr with ANR
return last year	0.157800976
quick ratio	0.129498028
PSR	0.115765508
market cap	0.104811958
adjusted beta	0.092137992
returns last 6 months	0.087656697
volatility 360 days	0.082562320
size	0.079194270
volatility 30 days	0.073358218
volatility 90 days	0.055528836
return last 3 month	0.051653948
P/E	0.050482462
EPS	0.025501506

For the third subset we use Lasso feature selection. Figure 3.2 illustrates the selection process. The x-axis contains different values for λ [3] and the y-axis shows the values feature coefficients may take. Each line in the figure represent one of the input features. We can see how much each feature influences the end result by the

[2] Depending on the task, we may also replicate a cluster of the minority observations.

[3] The x-axis shows the values of $-log(\alpha)$ to reverse the direction of the graph and to ease visualization. We actually see which features are the last to leave the model, thus the respective feature is considered important.

value of the coefficient [36]. For example, the first feature to enter the model is volatility_90_days, with a negative influence. The second feature to enter is net_income, with a positive influence. The pink line with the highest negative influence enters late in the model and it's not included in the final sub set. The first eight features to enter the model are chosen for estimation.

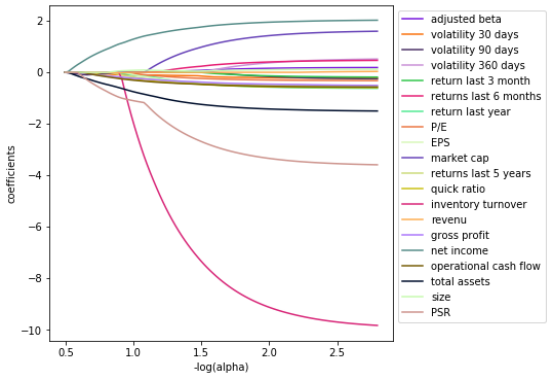


Figure 3.2: Feature selection using Lasso regularization. The features enter the model in order of importance.

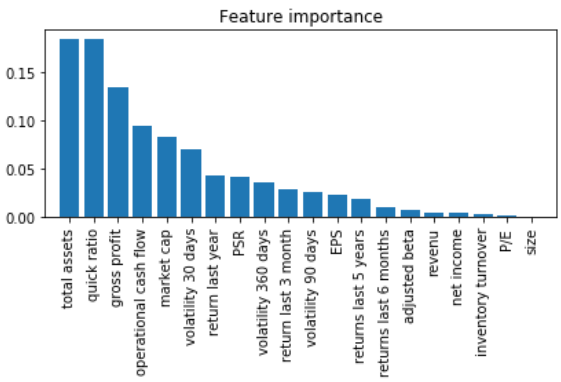


Figure 3.3: Ranking of feature importance using Random Forest

Subset four is chosen using Random Forest feature importance algorithm [37]. From this, top ten features are chosen for analyst rating estimation. Figure 3.3 shows Random Forest feature importance ranking. A summary of the chosen subsets of features is presented in table 3.3.

Table 3.3: The selected subsets of features to be used in prediction

Subset 1	adjusted beta, volatility 360 days, return last year, market cap, net income
Subset 2	return last year, quick ratio, PSR, market cap, adjusted beta, return last 6 months, volatility 360 days, size, volatility 30 days, volatility 90 days, return last 3 months, P/E, EPS
Subset 3	volatility 90 days, net income, total assets, PSR, gross profit, operational cash flow, volatility 30 days, quick ratio
Subset 4	total assets, quick ratio, gross profit, operational cash flow, market cap, volatility 30 days, return last year, PSR, volatility 360 days, returns last 3 months

Hyper-parameter Tuning

We fine tune the parameters for Lasso, Random Forest and Gradient Boosting, individually for each subset of features.

For Lasso model, complexity is chosen by varying the value of the regularization parameter α using K-fold cross validation method with 10 folds. K-fold cross validation is a technique used for out-of-sample testing on the same data set. It divides the data set into K folds and iteratively uses, by rotation, one fold as training set and the rest K-1 folds as test set. Figure 3.4 illustrates this process of choosing α. We fit the Lasso model iteratively with different values for α (x axis) on each fold of the 10-fold cross validation method (y axis). The dotted lines represent the error value for each fold. We see how the error develops with the increase of the regularization parameter α. The black horizontal line marks the average error across folds. The point where the average error is the least is marked by the vertical dotted black line, which marks the chosen value for α. The figure was created on the whole data set. The ideal choice of α differs for each data subset, thus the final estimation is made with different values of α for each subset.

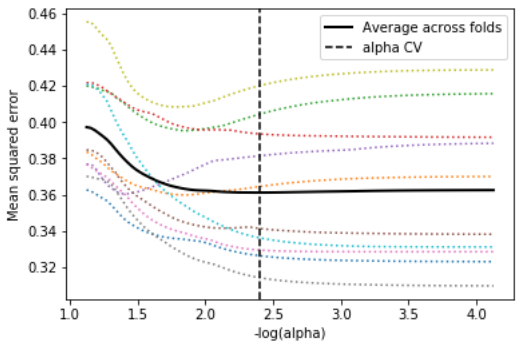


Figure 3.4: The average error across 10 folds of the Lasso model at various values of the regularization parameter α; The vertical line marks the lowest average error and the ideal value to give α

The hyper-parameters of the Random Forest model were tuned with *GridSearch* and 4 folds cross-validation. These parameters are *min_sample_leaf*, representing the minimum number of samples for a node to be become a leaf, *min_sample_split*, representing the minimum number of samples required to split a node and *max_depth*, referring to the maximum depth of the tree.

For Gradient Boosting Regressor *max_depth*, *min_sample_split*, *min_sample_leaf*, *max_features*, *subsample* and *learning_rate* are adjusted. *Max_features* describes the maximum number of features considered when choosing the best split, *subsample* represents the fraction of samples used for fitting individual base learners and *learning_rate* represents the degree of change the model allows when estimating.

3.4.3 Classification

In this section we aim at predicting analyst rating per individual sector using classification algorithms. The sector we choose for our experiments is 'Financials' as it has the highest number of samples, 194, and has samples from all 5 classes.

For this we need to transform the analyst rating from a continuous variable to a categorical one. We create 5 labels: *strong sell, sell, hold, buy, strong buy*. The stocks with ratings between 1 and 1.5 are labeled with *strong sell*, the ones with rating between 1.5 and 2.5 are labeled with *sell*, rating between 2.5 and 3.5 corre-

sponds to *hold* label, rating between 3.5 and 4.5 are labelled with *buy* and rating from 4.5 to 5 become *strong buy*. Our classification task becomes thus a multi-class one.

Using previous data analysis, we selected a small subset of features for estimation: adjusted_beta, return_last_year, PE and market_cap. We clustered the stocks based on selected features and used the cluster numbers as a new feature for prediction. We also create polynomial features of 3nd degree. Polynomial features are generated new features created from polynomial combinations of the existing features.

Class imbalance

Class imbalance occurs when there is a great difference in the number of observations that belong to each class. In our case, after selection of stocks from sector *Financials*, the data set contains 2 samples belonging to class *strong sell* and 1 observation with a *sell* label. For class *hold* we have 62 samples, class *buy* has 145 instances and class *strong buy* has 72 samples.

To deal with class imbalance, one usual approach is to adjust the data set as to have equal number of samples from each class, as described in Section 3.4.2. In our case, the number of minority observations is too small to reduce the majority class to the same size and also too small to synthesize more observations without adding bias to the model. As our minority observations are, in the best case, only two, there is not much variance in the characteristics of the minority observations for the model to truly learn to recognize similar new observations.

Another approach on dealing with class imbalance is to modify the classifier so it can learn to deal with the imbalance. The approach is based on ensemble methods and it improves the prediction by constructing several classifiers that together participate in the end result.

There are two main techniques in constructing ensemble classifiers: Bagging and Boosting. Bagging generates *m* new training sets from the given observations. Some of the observations may be repeated across the generated training sets. Then *m* models are trained on the generated data sets, followed by all *m* fitting the same test set. The output is decided by the majority of classifiers. In Boosting multiple weak base classifiers are trained in series to build a single strong one. We used AdaBoost as it the first boosting algorithm and most popular. AdaBoost works by modifying the distribution of the training set at each iteration, giving more weight to the previously miss-classified observations. The weak learner contributes to the strong final classifier in proportion to it's performance [38].

Models

We use Decision Tree Classifier as base for Boosting and Bagging methods. Support Vector Machines was successfully used in literature [21] [25] [33]. The Python implementation of Support Vector Machines uses One-vs-One approach to deal with multi-class classification. In One-vs-One we create an ensemble of binary classifications for each pair of classes. The method is computationally expensive, but insensitive to data imbalance. We experiment with the addition of a second technique of multi-class classification, One-vs-Rest classification, on top of SVM algorithm. In One-vs-Rest we train one binary classifier per class, with instances of the respective class being labelled as positive and the rest of the instances being the negative class.

3.5 Results

Analyst Rating Prediction

Analyst ratings were estimated with Linear Regression, with and without Lasso regularization, Gradient Boosting Regressor and Random Forest Regressor, on the four subsets of features described in section 3.4.2. The prediction of the average analyst rating is taken as benchmark. We use Mean Squared Error, Residual Sum of Squares and R squared statistics to evaluate the regression results. Mean Squared Error (MSE) measures the average squared difference between predictions and target variable. It expresses the average error in the prediction of the model and it is used to asses the quality of the model. It's value is always positive because of the squares and a value close to 0 is desired. Residual sum of squares (RSS) is a simple metric that sums up the squared errors between target variable and prediction. It's value tells us how much of the variance of the input data is not explained by the model. R squared measures the percentage of the total variation of the dependent variable that is explained by the by the independent variable. It can gave values between 0 and 100, with high values expressing a good fit of the model with the given data.

Table 3.4 presents our estimation results. As benchmark we took the prediction of the average rating for all instances. The benchmark has a MSE of 0.407. All models perform slightly better than the benchmark, meaning that they learn something. As expected, the best performing model is Random Forest Regressor, with a mean squared error of 0.36 on subset 4. Errors do not vary much across models and subsets. Worst performing algorithm predicts with a MSE of 0.38 in case of Linear Regression. The regularization parameter does not improve any of the models.

Low residual sum of squares and R Squared values show poor fit of the data with the model. Comparing RSS and R Squared scores of models with the ones of benchmark, our models learn very little from the data and the data itself is not very representative for the chosen models

Data	Model	MSE test	RSS test	R^2 test
Set 1	LinearRegression	0.388	509.274	0.03
	Lasso	0.388	508.448	0.032
	GradientBoostingReg	0.381	500.057	0.056
	RandomForestReg	0.373	489.027	0.074
Set 2	LinearRegression	0.385	504.919	0.043
	Lasso	0.385	504.782	0.043
	GradientBoostingReg	0.371	486.268	0.079
	RandomForestReg	0.369	484.093	0.08
Set 3	LinearRegression	0.387	508.373	0.032
	Lasso	0.387	508.375	0.032
	GradientBoostingReg	0.367	481.05	0.082
	RandomForestReg	0.369	484.283	0.077
Set 4	LinearRegression	0.378	495.908	0.052
	Lasso	0.378	495.72	0.053
	GradientBoostingReg	0.363	476.658	0.092
	RandomForestReg	0.36	472.386	0.098
	Average (Benchmark)	0.407	533.734	0

Table 3.4: Analyst Ratings prediction results

Prediction per Sector

We validate the results of classification using F1 score. The F1 score is the harmonic mean of precision and recall metrics. Precision refers to the number of correctly classified instances over all instances classified to the respective class. Recall is the ratio between correctly classified instances over all instances in the actual class. The micro-average F1 score takes class imbalance into consideration and gives increased weights to classes with fewer samples. Table 3.5 presents the micro-average F1 score of the classification models previously mentioned. We considered a naive benchmark the average F1-score across prediction of each class. More specifically, given n classes (five in our case), we calculate the benchmark based on formula $Benchmark = \frac{1}{n}\sum_{i=1}^{n} f_1score(\textit{predicting class } i)$.

The majority of the models score above benchmark. Columns 2 and 3 of results table represent the micro-average F1 score of models that use a small set of features from data, comprised of adjusted_beta, return_last_year, PE and market_cap. The last 2 columns of results belong to models that contain additional polynomial features of 3rd degree. In case of decision tree based models, the polynomial features don't improve the end result. SVM model is improved from an F1 score of 0.13 to 0.28.

Table 3.5: Results of classification per individual sector.

Model	without polifeature		with polifeature	
	F1_score without clusters	F1_score with clusters	F1_score without clusters	F1_score with clusters
Decision Tree Classifier	0.46	0.46	0.46	0.46
AdaBoost with base Decision Tree	0.48	0.46	0.46	0.43
Bagging with base Decision Tree	0.46	0.46	0.46	0.46
SVM linear kernel	0.13	0.44	0.28	0.28
OneVSRest with base linear SVM	0.13	0.62	0.32	0.32
SVM poly kernel 9d & 11d	0.02	0.57	0.03	0.03
OneVSRest with base poly SVM	0.03	0.72	0.08	0.48
Average prediction (benchmark)	0.2			

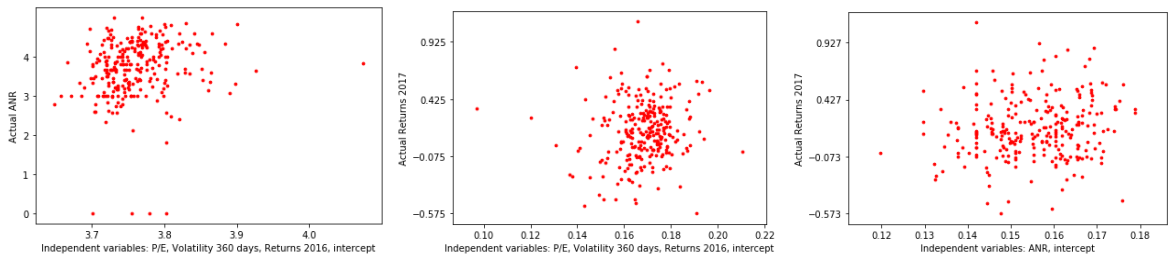
As mentioned in section 3.4.3, we clustered the observations based on financial features and used the cluster number as an additional feature for classification. The

third and last column of Table 3.5 show results of models in which we included the cluster number as feature. This does not affect the results of the decision tree classifiers, nor any of the models containing polynomial features. But in the case of SVM it makes a significant difference, bringing the F1 score from 0.03 to 0.72, the highest value.

Predicting Returns from Analyst Ratings

Further analysis was carried out in order to have a complete overview over the explanatory power of data. As analyst ratings correspond to expected future returns, we want to test whether returns are correlated with the set of features we used: volatility 360 days, P/E, returns_last_year.

We applied Multiple Linear Regression on features volatility 360 days, P/E, returns_2016 to see their cumulative linear influence over analyst ratings. The result of this method is depicted in Figure 3.5(a). Figure 3.5(b) shows the cumulative influence of features volatility, P/E and returns of 2016 over returns of 2017. Lastly, Figure 3.5(c) shows how much the returns of 2017 are explained by analyst ratings emitted at the end of 2016.



(a) Correlation with ANR (b) Correlation with returns of 2016 (c) Correlation with returns of 2017

Figure 3.5: Multiple Linear Regression

In case of correlation we would expect the red data points to form a pattern. Contrary to expectations, these plots show little explanatory value in the independent variables. Figure 3.5(a) and 3.5(b) show that the set of features used do not influence the analyst rating, nor do they explain returns of 2017. Lastly, Figure 3.5(c) shows that analyst ratings emitted at the end of the year 2016 are not correlated with returns of 2017. In theory they should, which suggest that the labels we have for analyst ratings are not precise.

3.6 Conclusions & Discussion

In this chapter we aimed at predicting analyst rating using variables from fundamental analysis and a data driven approach. Our goal was to verify weather we can simplify this task by using easily accessible data and basic machine learning algorithms.

We first experimented with regression algorithms. All the models we created estimate with slightly less error that the benchmark, meaning that they all learn something. The high values in the RSS scores show that data is not representative for the chosen models, as a considerable amount of variance in the explanatory variables is not explained by the models. The low values of R squared strengthen this belief as they show that less than 10% of the variance of analyst rating is explained by the features chosen. This explains also why data imbalance solutions did not improve the results. The regularization parameter does not add improvements, which indicates that the Linear Regression model is not complex enough to be affected.

We believe the lack of correlation between independent and dependent variables may be caused by different characteristics among the economical sectors in which companies in our data set operate in. For instance, the total assets of a company activating in the *Utilities* sector should always be high, no matter the performance of the respective company, as the infrastructure required for such companies is substantially costly. As is difficult to build such infrastructure without a loan, the liabilities of such companies may also be high without a direct correlation with future performance. To test this belief, we conducted a second round of experiments in which we employ classification algorithms in analyst rating prediction per sector. The novelty of our approach lies in clustering the input data set and using the cluster numbers as features for estimation. This significantly improves the F1 score of the SVM model. The results of this estimation is slightly improved, showing that indeed there are characteristics of stocks that are better captured when grouping the data set per sector. The same technique was applied in price prediction from news [21] and we encourage it for future research.

Lastly, we analyze the possibility that the data itself is not representative for the given task. We aim by this to explain the low scores of RSS and R squared. By definition, the analyst rating should be correlated with future returns. Through Multiple Linear Regression we found weak linear correlation between the mix of features we use to predict analyst rating and the actual future returns. Similarly, no correlation was found between analyst rating values at the end of 2016 with the actual returns of 2017. This means that the target variable is not accurate and representative for our task.

A possible explanation for the lack of correlation is that the target variable con-

tains significant noise as it represents an average rating from 46 financial advisory houses. As showed in section 2.2 advisers have different interpretations on the significance of a buy or sell label. This implies different principles of financial analysis and approaches on rating companies, thus it may be difficult for the model to learn a general way of solving this task from an aggregate score. It would be interesting for future research to experiment the methodology of this study on a data set containing the analyst rating of individual advisory houses.

It is important to note also the varied sources of information financial analysts use in real life, as presented in section 3.5. While fundamental analysis describes the long term trend of the performance of companies, the market is affected in the short term by news and press releases about companies. Thus, for an accurate prediction and adaptability to a continuously changing market, future research should incorporate the other forms of information elicitation as well, such as the study of Schumaker [21].

Moreover, the accuracy of the models does not necessarily represent the profit an investor may gain by trading as the system recommends. ”A single losing trade can wipe out the profit of several accurately predicted trades. A low mean squared error is also not a guarantee that a model can produce profitable predictions” [26] [39]. As precision is needed in financial domain, more research is needed on the correlation between model performance and trading profits. As profits are relative to the market and the market is constantly changing, this relationship may be difficult, if not impossible to asses. A fail-safe mechanism for incorrect investment recommendations may also be a solution.

Better future engineering could be a promising direction for future research. This is indicated by the high improvement in F1-score obtained by Support Vector Machines model when adding the cluster number as feature. This finding seems appropriate for the context as the label we are aiming to predict is relative to the opinion of the financial analysts and there may be multiple layers of thinking between the raw data and the final rating.

It would also be interesting to experiment in the future with price prediction per individual stock. This recommendation is motivated by the fact that companies themselves have different characteristics that are not highlighted by financial indicators, such as the underlying value of their service or product. Human financial analysts implicitly asses these from experience. For instance, a system such as ours would predict a sell (or strong sell) label for a company that has decreasing profits over the years and high negative net income. The company we are talking about is Tesla, which invests highly in research and development of autonomous electric cars and it is just now starting to sell products at an affordable price. A human advisor understands that these company’s high liabilities are caused by necessary high invest-

ments in knowledge and infrastructure and foresees the future added value of the product this company is aiming to build. For this reason, he is very likely to give a buy label for Tesla's stock. As the market continuously expands, it is possible that there will be more anomalies like this in the future. Thus, an individual estimation model for each stock may better incorporate its particular characteristics in the prediction.

Chapter 4

Diversified stock allocation through clustering

4.1 Introduction

In this chapter we propose several methods of constructing a diversified portfolio by means of unsupervised learning. We limit the application to one type of assets, namely stocks. With the purpose of exploring, we employ different types of clustering algorithms. Clustering is done based on encoded features *Sector* and *Region* and on the correlation matrix of the whole data set. Twenty clusters are created, one for each slot of the portfolio. We then have the challenge of selecting one stock from each cluster. We experiment with four methods of stock selection. One of them enables integration of investor preferences, a requirement for Constraint-Based Recommender Systems that are part of the goal of this study. We validate our approach against a constructed benchmark that is specifically programmed to create portfolios with respect to the sector and region allocation of a common benchmark in the field, MSCI World.

The chapter starts by presenting the theoretical background of technologies used. Moreover, previous work done in the field is presented in Section 4.3. We then go into methodology of our proposed techniques and experimental settings. Section 4.5 describes validation results of our experimental procedure and Section 4.6 summarizes evaluation results from presenting the system to stakeholders. Finally, Section 4.7 concludes the chapter with discussion of results and final remarks.

4.2 Background: Unsupervised Learning

Unsupervised learning is used when the results of the tasks are not known in advance. It is used in analysis and structuring of data, pattern discovery and knowl-

edge extraction. It can also be used for dimensionality reduction of the input space for other tasks, such as visualization in 2D of higher dimensional data sets and improvement of performance in machine learning algorithms. In our case, we want to group together stocks with similar characteristics. The end goal is to make a selection of dissimilar stocks by picking items from different clusters. What is defined as similar is of crucial importance in this case, thus we dedicate section 4.2.1 for a detailed description of our design choices.

Clustering is an unsupervised learning technique which discovers groups within data. The samples are assigned to one of the clusters on the basis of geometric distances: samples that are close to each other will be grouped together. Closeness of data points is measured in terms of *similarity*, with a value of 0 representing two identically positioned data points [40].

K-means is one of the first clustering algorithms, created separately by Lioyd [41] and Forgy [42]. It is a type of partitional clustering, namely it divides the observation into disjoint sets such that each data point belongs to one single cluster. It takes as input a parameter k that counts for the number of clusters to be formed. It then randomly initializes a number of k vectors that will represent the geometric center of the clusters, also known as centroids. The goal is to assign each data point to the correct centroid as to minimize the function:

$$J = \sum_{n=1}^{N} \sum_{k=1}^{K} r_{nk} ||x_n - \mu_k||^2 \tag{4.1}$$

where x_n is a data point, N is the total number observations and K is the number of clusters. $r_{nk} \in \{0, 1\}$ describes to which cluster data point x_n belongs to, so that if x_n is assigned to cluster k then r_{nk} will be 1 and 0 otherwise. μ_k is a prototype vector associated with cluster k, also known as centroid of the cluster. To achieve this, the algorithm works iteratively in a two step approach:

- Step 1: Each data point is assigned to one of the centroids, the one that is the closest; this is similar to minimizing J from equation (4.1) with respect to r_{nk} and keeping μ_k fixed.

- Step 2: Move the centroids as to meet the average distance between the points assigned to it; here we minimize J in equation (4.1) with respect to μ_k by keeping r_{nk} fixed.

The algorithm will iteratively repeat step 1 and 2 as long as the change in J is significant. Once the change in J is lower that a predefined threshold, the algorithm ends.

In hierarchical clustering nested groups are formed. The method iteratively combines clusters to form fewer groups or iteratively divides clusters to create more

groups [43]. We use Agglomerative algorithm, which follows bottom-up approach. Initially, each data point is considered an individual cluster. With each iteration, the algorithm combines pairs of close clusters together, until the whole data set is contained in one single cluster. The results of this method can be visually described as a dendrogram, similar to the example in Figure 4.1[1]. The desired number of clusters can be extracted by cutting the dendrogram at the corresponding position. For example, we may cut the dendrogram at the height specified by the lower red dashed line in Figure 4.1 and obtain 9 clusters or at the height of the second red dashed line which will result in 6 clusters.

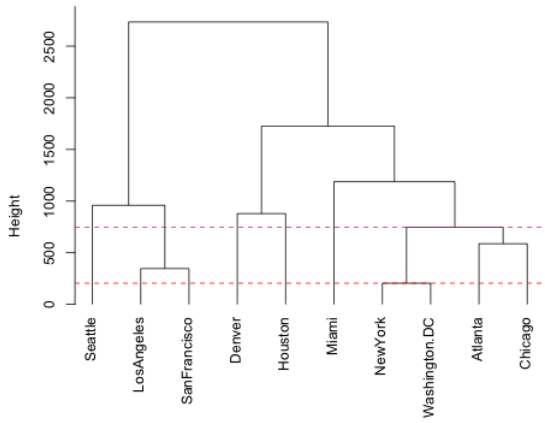


Figure 4.1: Hierarchical clustering dendrogram of 10 US cities based on geographic distance.

Clusters are combined based on similarity measures. We can further characterize the hierarchical clustering algorithms by the similarity measures they use when splitting or merging clusters:

- Single-linkage clustering considers the distance between two clusters equal to the shortest distance from any member of one cluster to any member of the other cluster. [44].
- Complete-linkage clustering defines distance between two clusters as the longest distance between members of either class [45].
- Average-linkage is a compromise between the two above. It calculates the distance between two clusters as the distance between the centroids of the respective clusters [46].

The different types of hierarchical clustering are depicted in figure 4.2[2].

[1] https://research.bowdoin.edu/digital-computational-studies/digital-computational-studies/under-the-hood-hclust/

[2] https://www.researchgate.net/publication/281014334_Inference_of_a_human_brain_fiber_bundle_atlas_from_high_angular_resolution_diffusion_imaging/figures?lo=1

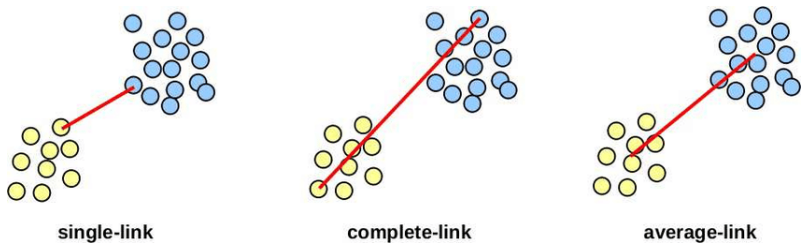


Figure 4.2: The three main types of hierarchical clustering

The choice of method used depends on the dataset to be clustered. If samples are sparse, single-link may create irregular clusters, also known as chaining effect. This happens when clusters that are distant from one other are merged together due to single elements being close. Complete-link tackles this issue by the considering the farthest point when merging, but, for the same reason, it may be sensitive to outliers. Average-link clustering represents a good compromise between previous methods, one that comes at the cost of higher computational complexity. With less than 2000 samples our dataset cannot be considered large, thus we can use average-link clustering.

Spectral clustering focuses on the affinity of the data points rather than their location. The algorithm works by constructing a similarity graph with all data points given, extracting the eigenvectors with the largest values and applying PCA to map the vectors into a new space where it's easier for an algorithm like k-means to define the separate clusters [47].

4.2.1 Distance metrics & similarity

The similarity of two data points can be seen as the inverse of the distance between them: as the distance decreases, the similarity increases [48]. Thus, when the distance is 0, similarity = identical.

The centroids from k-means clustering need to adjust their position to represent the average distance among containing points. The objective of k-means is to minimize the within-cluster variance [49]:

$$J = \sum_{k=1}^{K} \frac{1}{n_k} \sum_{x,x' \in C_k} \sum_{j} d_{x,x',y} \tag{4.2}$$

Where k is the cluster, n_k is the number of points in that cluster, C_k contains the respective instance index. When using an euclidean distance function of the form $d_{x,x,y} = (x_{i,j} - x_{i,j})^2$, the objective above becomes minimizing the residual sum of squares [50]:

$$J = \sum_{j=1}^{k} \sum_{n \in S_j} |x_n - \mu_j|^2 \tag{4.3}$$

From [50] we extract a well known formula:

$$SStotal = SSwithin + SSbetween \tag{4.4}$$

As the points don't move, thus SST remains constant and minimizing SSwithin (tight clusters) implies maximizing SSbetween (cluster well separated) [49]. Hence, the similarity measure must be a distance function valid in euclidean space [51]. Only then the calculation of the average position for the centroid will maintain its significance and the k-means can converge to an optimal solution.

For implementing the diversification strategy we will experiment with similarity based on correlation values and on attributes from fundamental analysis, namely *Sector* and *Region*. For correlation clustering we use a distance metric of the form $\sqrt{2(1-\rho)}$, where ρ is the Pearson correlation coefficient described in section 2.3.1 and defined by formula (2.3). The function is taken from Rosen's paper [52], which also references Mantegna's paper for validity proof of the distance function in euclidean space [53]. We choose this formula for the good results obtained in studies previously using it and because it is mathematically proven. To measure similarity of stocks based on *Sector* and *Region* we need to transform these categorical values into ordinal ones. The applied transformation is detailed in section 4.4.1.

4.3 Literature review

This section presents previous research concerning machine learning integration in portfolio construction and recommender systems. We first look into research related to construction of diversified portfolios through clustering. Then we go into state of the art recommender systems with applications in financial advisory. Finally we dig into previous research that integrates machine learning into recommender systems.

Portfolio diversification implemented through clustering technique has been researched previously with successful results.

Joglekar, S. (2014) [54] clustered stocks by the range of time-weighted correlation between them. In a second step, he applied a genetic algorithm to generate combinations of stocks that form a diversified portfolio. The correlation clustering is done with *k-means++* algorithm [55] and using a distance function of the form $(1-\rho(X,Y))$, with ρ being the pairwise Pearson correlation coefficient detailed in section 2.3.1 and defined by formula (2.3). The number of clusters is user defined and it also represents the number of slots in the final portfolio. The proposed framework has been evaluated on time-series data representing closing prices of stocks from various sectors. Results show a stable portfolio during market fluctuations with positive returns compared to the benchmark index. In terms of risk, the constructed portfolio is approximatively three times more risky than the benchmark index.

Ren, Z. (2005) [56] employed the clustering technique to reduce the estimation error of regular mean-variance portfolio optimization. His method is based on previous research which established that correlations estimates between stocks are more accurate that estimations of expected returns [57]. In the study, highly correlated stocks are manually grouped into clusters. Each cluster is treated as a single stock by averaging intra-cluster expected returns. Threshold of correlation is chosen experimentally as 0.2 and 0.15. The portfolio weights are set by running mean variance optimization over the clusters. Results show that correlation clustering created more robust portfolio, less sensitive to changes in expected returns, thus requiring less rebalancing when the market changes. However, these results may differ when stocks are real instances and not the average of the cluster.

Rosen, F. (2006) [52] proves that clustering stocks based on the co-movement of prices between them creates economically relevant groups. His method implemented hierarchical clustering with average linkage. Similarity between time series of securities is calculated using a distance function of the form $\sqrt{2(1-\rho)}$ [53], where ρ is Pearson correlation coefficient detailed in section 2.3.1 and defined by formula (2.3). Correlation is calculated from daily logarithmic returns of OMX Stockholm 30 Index. The study proves successful, resulting in efficient grouping of stocks with similar sector and sub-industry.

Craighead, S. & Klemesrud, B. (2002) [58] used PAM clustering [59] to filter out stocks with non-uniform series. The algorithm successfully eliminates two stocks that went bankrupt soon after their experiments. He then applied outliers algorithm [60] to decide when to buy and hold stocks in different investing strategies. Both passive and active strategies created in the study exceed in returns the passive strategy of following S&P500 index. The study did not consider transaction costs, but the authors state that the difference in returns would cover transaction costs. Although clustering contributes to one third of the full technique, the authors regard the success of their method to PAM clustering for filtering out weak performing securities from the initial universe of stocks.

Marvin, K. (2015) [61] also experiments with k-means clustering for diversification, but uses an alternative measure of similarity based on fundamental analysis. It especially analyses the performance of the clustering technique before and after moments of high fluctuation in the market, such as the economic crisis of 2008. After splitting the data disjoint clusters, one stock is selected from each group based on highest Sharpe Ratio. The number of clusters in decided dynamically, from the silhouette coefficient and the ratio of cluster sum of squares over the total sum of squares. The results of experiments show a more volatile portfolio compared to the S&P500 benchmark with few larger than benchmark negative peaks and considerable more large positive peaks. Returns are also evaluated in a simulated

portfolio with initial capital of 1000$ and 15 years on investment horizon. Compared to S&P500, Marvin's portfolio earns up to 5.7 times more in periods pre- and post-crisis and similar to benchmark during the crisis.

Korzeniewski, J. (2018) [62] focuses on what would be the ideal number of clusters for this type of problem. To score the fitness of clusters the Calinski-Harabasz index is used. He uses k-means and PAM algorithm [59] for clustering on time series data. From each cluster the stock with the highest Sharpe Ratio in selected to create the final portfolio. Mean return rate is used to evaluate the resulting portfolio against the market. Best results are obtained by k-means clustering, when the number of clusters recommended by Calinski-Harabasz index is modified. The study assumes that the maximum number of positions for a good portfolio is 12. This contradicts our knowledge acquired from interviewing financial advisors, who recommend an exposure of no more that 5% for each position, thus allowing a minimum of 20 positions in a portfolio. The study confirms the findings of [61]

Knowledge based recommender engines have been explored in literature in the context of financial advisory. Felfernig, A. & Kiener, A. (2005) [63] built an interactive application for investment advice incorporating knowledge based recommender system. It includes a model based diagnosis of customer requirements, adapted dialog to customer needs, personalized repairs solution and multi-attribute object ranking. Furthermore, their tool supports automated generation of solution explanation and test cases, all integrated in a friendly graphical user interface. Musto, C., Semeraro, G., Lops, P., De Gemmis, M. & Lekkas, G. (2015) [64] use case-based method to recommend asset allocation strategies. Although the returns yielded by the system during three and six months post evaluation decrease, they are still higher than an adapted kNN baseline and real human advisers. Zanker, M. (2008) [65] integrates collaborative filtering with knowledge based recommendation engine. The former method elicits a personalized constraints set that forms the knowledge base served as input for the latter engine. This hybrid approach solves the knowledge bottleneck issue and yields better results than algorithms with manual encoded knowledge bases in terms of precision and recall.

A well known study is the one conducted by Barbosa, B. & Belo, O. (2010) [25]. They create a multi-agent autonomous trading system for the Forex market, which trades currencies. This system has three modules: an *Intuition Module* which uses multiple classification and regression algorithms in Ensemble [27] to predict the direction of the price, an *A Posteiori Knowledge Module* which uses Case-Based Reasoning to suggest when and how much to trade and an *A Priori Knowledge Module* that makes the final trading decision by analyzing the outputs of the previous modules in a Rule-Based Expert System. We described the Intuition Module of this

system in Section 3.3. With the addition of the case-based reasoning module and the rule based module, Barbosa's study presents a different approach on combining the capabilities of machine learning methods with recommender systems that follow the constraints of the domain. Their multi-agent technique works as a fail safe mechanism for incorrect predictions. The results of the study show the method is successful in gaining good profits with a decreased number of trades, which consequently decreases trading commissions and interest payments. The estimation accuracy is between 52.82% and 56.33%. The authors note that is difficult to establish the correlation between accuracy of prediction models and profits made by investing according to these predictions. The results of their study brings justice to this argument, as we observe an increase of 177% in profits given by 2.74% increase in classification prediction.

Integration of machine learning approaches with Collaborative Filtering and Content Based recommender engines has been explored by few in domains outside finance. To our knowledge, integration of machine learning has not been explored for constraint based recommender systems and not in finance. Sahu, P., Nautiyal, A. & Prasad, M. (2017) [66] offer a comparative analysis between classic recommendation techniques such as content-based, collaborative filtering and hybrid approach, and machine learning algorithms, specifically k-Mean Clustering and Naive Bayes. Experiments were conducted on different sizes of the MovieLens dataset, with Naive Bayes approach giving the best precision followed by k-Mean Clustering in all cases. Thiengburanathum, P., Cang, S. & Yu, H. (2015) [67] constructs a decision tree recommender system for traveling, using C4.5 algorithm. Peska, L. (2016) [68] extract implicit user feedback from the interaction with the recommender system and predict purchasing probability using Linear Regression, Lasso, Decision Trees and AdaBoost for classification and regression.

In this project we wish to extended the work done by Marvin [61] and experiment with creating diversified portfolios by clustering based on sector and region attributes. In Marvin's study, clustering is done on *Revenue* and *Net Income* divided each by *Total Assets*. The experimental setting puts stronger emphasis on hedging against a volatile market. For this, stocks are picked based on Sharpe Ratio. We also employ the same stock selection method and experiment with three others. Moreover, we cluster stocks based on the correlation between them as in the studies of Ren and Rosen [56] [52]. We validate the created portfolios in terms of returns as in studies of Jonkelar [54], Craighead [58] and Marvin [61]. In addition to these studies, we propose a methodology of integrating investor's preferences, going into the field of Recommender Systems and taking a novel approach into combining them with machine learning.

4.4 Implementation

We use the clustering technique to group similar stocks together. Stocks inside a cluster will be similar to each other and, likewise, stocks from different clusters will be dissimilar to each other. The end goal of our diversification proposal is to create a mix of dissimilar stocks.

From the perspective of financial advisers a diversified portfolio is one that contains different securities, meaning they belong to varied groupings, such as sectors and regions. From Modern Portfolio Theory we learned that a diversified portfolio is one that contains securities with low correlations between them. We reflect both views on diversification in our proposed methods. On one hand we cluster based on features sector and region, thus mimicking the approach of the advisor. On the other hand, we cluster based on correlation between stocks, as recommended by MPT.

4.4.1 Data & Preprocessing

For construction and validation of diversified portfolios we use a data set consisting of 1963 stocks from various sectors and countries. We group stocks per regions as specified by MSCI Index[3]. Table 4.1 lists the variables used in the implementation and validation of proposed methods. Identification of stocks is done by ticker and name. Sector and Region are used for one method of clustering. For the second method, based the the correlation matrix, we calculate the Pearson coefficient using volatility and beta values. The rest of the variables are used in the implementation of investor preferences described in Section 4.4.3. We chose them as to put the risk, returns and gains for the investor into perspective. These variables are usually used in fundamental financial analysis.

To cluster using *Sector* and *Region* the distance we calculate between stocks needs to have significance in the application domain. With categorical variables, such as the ones representing *Sector* and *Region* obtained in the raw data set, we can only measure if variables are the same or not. For our system, we need to measure a degree of similarity between stocks with financial relevancy. Thus, the categorical variables are encoded into ordinal ones with the help of financial knowledge.

[3] `https://www.msci.com/emerging-markets`

Table 4.1: List of features used in task of creating diversified portfolios with machine learning.

Feature	**Description**
Stock ticker	Identifier code of security
Stock name	Name of the emitting company
Sector	Economic area of the company
Region	Geographic area of the company
Beta_120d	Beta value calculated on the last 120 trading days of 2016
Volatility_120d	Volatility values calculated on the last 120 trading days of 2016
Sharpe ratio_120d	Sharpe ratio calculated on the last 120 trading days of 2016
Dividents	Divident payment per share in year 2016
Quick ratio	Indicator of company's liquidity in paying short term liabilities
Revenue_5y_avg	Average gross income for the past 5 years
Total liabilities	Short term and long term liabilities
EPS	Portion of profit allocated per outstanding share in the last year
EPS_5y_avg	Average portion of profit allocated per outstanding share in the past 5 years

- Feature *Sector* follows the values of MSCI World allocation as weights and the ranking is given by standard sector recommendations in varied business cycles, such as Early, Mid, Late or Recession[4]. For example, in business cycle *Mid*, which we chose randomly in this implementation, sector *Consumer Staples* is encoded as 24, *Technology* as 19, *Financials* as 17, *Industrials* as 15, *Consumer Discretionary* as 9, *Energy* as 6, *Communications* as 5, *Materials* as 3 and sector *Utilities* as 2. In contrast, in business cycle *Recession*, sector *Consumer Staples* is encoded as 24, *Financials* as 19, *Consumer Discretionary* as 17, *Utilities* as 15, *Communications* as 9, *Energy* as 6, *Technology* as 5, *Industrials* as 3 and sector *Materials* as 2.

- *Region* is manually ranked according to nominal GDP value in 2017[5]. Thus, region *USA* is encoded as 10, *Europe ex UK* as 9, *Asia ex Japan* as 7, *Japan* as 6, *UK* as 5, *Canada* as 4, *Emerging Markets* as 2 and region *Others* as 1.

In clustering based on correlation matrix we use equation (2.3) to calculate the correlation between pairs of stocks and then formula $\sqrt{2(1-\rho)}$ to transform the correlation matrix in a distance matrix suitable for our algorithms. The formula has been used before in literature [52] and is proven to be valid in euclidean space in [53]. At first, the clustering algorithms would group the majority of stocks in one cluster and the very few rest in a second cluster. Looking more closely at the distance values, we notice there are very small differences among them, with changes appearing after

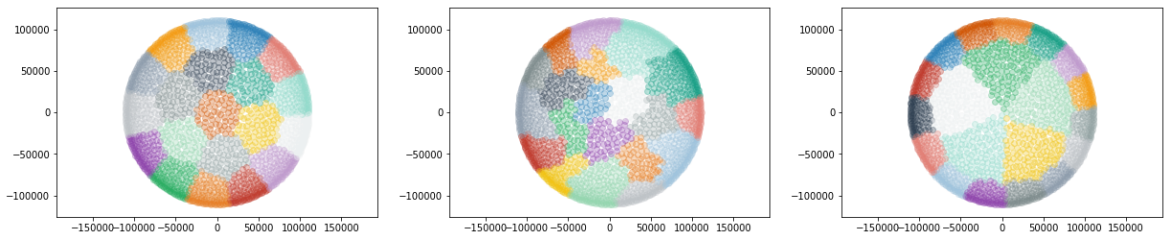
[4] `https://eresearch.fidelity.com/eresearch/markets_sectors/sectors/si_business_cycle.jhtml?tab=sibusiness`

[5] `https://en.wikipedia.org/wiki/List_of_countries_by_GDP_(nominal)`

the 12th digit. We obtained better clustering after applying Multidimensional Scaling (MDS). MDS is a technique from data visualization which uses double centering on the distance matrix and eigenvalue decomposition to extract the 2 dimensional coordinates of the elements of the distance matrix. These coordinates are further used as input in correlation clustering.

4.4.2 Methodology

Correlation clustering is done using k-means, agglomerative and spectral algorithms. We input the distance matrix resulted from the correlation matrix in 3 clustering algorithms: k-means, agglomerative and spectral. The clusters resulted are plotted in Figure 4.3. The colors are mapped to the cluster number assigned to each data point, but the numbering of the clusters is non deterministic by the design of the algorithms. There is no best result at this point, but the differences between methods are clear: k-means clusters are better defined into a round shape; agglomerative clusters are interlaced because of the average distances calculated between groups; the spectral algorithm finds greater concentrations of data points towards the extremities of the plot, meaning stocks with low correlation. MDS allows us to position the correlations between stocks in a 2D space. The visualization takes a rounded shape because Pearson's correlation coefficient can only take values in interval $[-1, 1]$.



(a) Clusters resulting from algorithm k-means

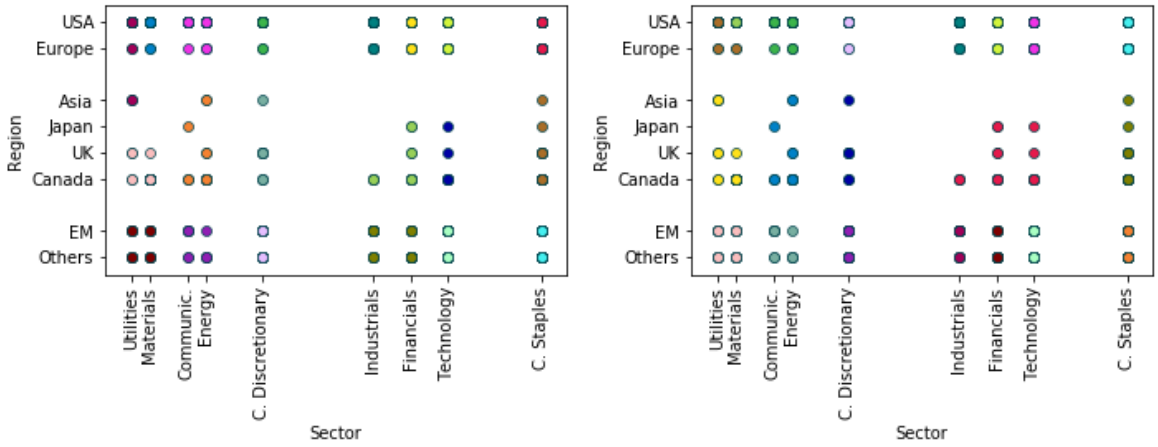
(b) Clusters resulted from algorithm agglomerative

(c) Clusters resulting from algorithm spectral

Figure 4.3: Shape and localization of clusters obtained from the correlation matrix

For clustering based on features sector and region we used k-means and agglomerative algorithms. The resulting clusters are plotted in Figure 4.4. Given the categorical type of the variables describing these features, there is no affinity between data points for the spectral algorithm to capture and perform better.

Applying the rules mentioned in section 2.3.2, we set a maximum of 5% exposure per security. We end up with a number of 20 slots available in the portfolio, thus requiring 20 clusters. We then select 1 stock from each cluster to fill one slot of the final portfolio. We picked 4 methods of selection:



(a) Clusters resulting from algorithm k-means (b) Clusters resulted from algorithm agglomerative

Figure 4.4: Shape and location of clusters obtained by clustering on features sector and region

1. Pick the one with the highest Sharpe ratio, which represents the best trade-off between risk and returns;
2. Pick one similar to a given stock. We chose Amazon for this as it is a well performing investment;
3. Pick the one top ranked by investor's preferences. How these preferences are implemented is described in more detail in section 4.4.3;
4. Pick one at random

Thus, we have 5 methods of clustering and 4 methods of selecting a stock to store in the final portfolio, resulting in the construction of 20 different portfolios. These are summarized in Table 4.2. We will further use the labels from the fourth column of Table 4.2 to refer to individual portfolios created by our system.

Table 4.2: Proposed methods of contracting diversified portfolios using clustering.

Clustering method	Clustering algorithm	Stock selection method	Label
Correlation clustering: clustering based on correlation matrix	K-means	Highest Sharpe ratio	portfolio_ml_corr_kmeans_Sharpe
		Stock similar to Amazon	portfolio_ml_corr_kmeans_similarity
		According to investor preferences	portfolio_ml_corr_kmeans_preferences
		At random	portfolio_ml_corr_kmeans_random
	Agglomerative	Highest Sharpe ratio	portfolio_ml_corr_aggl_Sharpe
		Stock similar to Amazon	portfolio_ml_corr_aggl_similarity
		According to investor preferences	portfolio_ml_corr_aggl_preferences
		At random	portfolio_ml_corr_aggl_random
	Spectral	Highest Sharpe ratio	portfolio_ml_corr_spectral_Sharpe
		Stock similar to Amazon	portfolio_ml_corr_spectral_similarity
		According to investor preferences	portfolio_ml_corr_spectral_preferences
		At random	portfolio_ml_corr_spectral_random
Clustering based on features sector and region	K-means	Highest Sharpe ratio	portfolio_ml_sere_kmeans_Sharpe
		Stock similar to Amazon	portfolio_ml_sere_kmeans_similarity
		According to investor preferences	portfolio_ml_sere_kmeans_preferences
		At random	portfolio_ml_sere_kmeans_random
	Agglomerative	Highest Sharpe ratio	portfolio_ml_sere_aggl_Sharpe
		Stock similar to Amazon	portfolio_ml_sere_aggl_similarity
		According to investor preferences	portfolio_ml_sere_aggl_preferences
		At random	portfolio_ml_sere_aggl_random

4.4.3 Integration of investor preferences

We personalize recommendations by ranking them according to preset possible preferences. In the benchmark advisor, this happens when adding a stock in the portfolio. Before each addition, the system checks which sectors and regions don't have the desired exposure and creates a list of stocks that belong to them. We then have to decide which stock to pick from the list. In the advisor that integrates machine learning, ranking of stocks happens after clustering, when each cluster contains more than 100 stocks. We score these stocks relative to the others in the same cluster, rank them according to sum of scores and pick the top one.

The investor can specify through the user of the system a list of desired features in stocks. This stocks are scored according to how much features meet desired preference. Scoring the features of a stock is made relative to the features of other stocks, meaning the maximum value for each desired feature will always get the maximum score.

The user can set seven options for preferences:

1. Prefer stocks that have given dividends in the past year
2. Prefer stocks with high EPS in the past year
3. Prefer stocks with high average EPS in the past 5 years
4. Prefer stocks with high quick ratio

5. Prefer stocks with high average returns for the past 5 years
6. Prefer stocks with low liability
7. Consider risk profile when ranking stocks

Each setting gives stocks a score from 0 to 100, depending on how much the preference is met. For example, the user of the system (financial advisor) selects preferences 1 and 2. Score 1: Stock A has the highest dividend payment for the past year of all stocks in the group, a value of 12.12%, thus receiving 100 points; stock B paid dividends of only 7.4% in the last year, thus receiving 61.05 points; stock C has missing value for dividends paid last year, resulting in 0 points. Score 2: Stock A has missing value for EPS value and gets 0 points; stock B has the highest EPS value in the group and gets 100 points; finally, stock C has a medium EPS value relative to the others in the group, thus receives 50 points. After each stock is scored, we sum these scores per stock and make a ranking. Continuing with our example, Stock A receives 100 + 0 = 100 points, stock B gets 61.05 + 100 = 161.05 points and stock C scores 0 + 50 = 50 points. We pick the top one, check if the stock is not in the portfolio already, add it if not or add the next unused stock available in the ranking. We then move to another sector and region (in case of benchmark) or cluster (in case of machine learning integration) and repeat.

In case of preference 7, the one that enables personalizing according to risk profile, the procedure for assessing scores is more complex. We focus on the value of volatility in the past 120 days and match that with the risk appetite of the investor. There are 5 types of risk profiles: defensive, moderate defensive, balanced, moderate offensive and offensive. We take the minimum and maximum value of feature volatility from all stocks in the pool list and split that interval in 5 buckets that will match the risk profiles in ascending order: the higher the volatility, the more suited is for offensive investors. For example, if the risk profile would be offensive, the stock with the highest value for volatility, let's say 10, would be assessed with 100 points. If the investor would be balanced, we would be recommended more stocks with value between 4 and 6, meaning the 3rd bucket. The stock in our example, with volatility of 10, would be in the 5th bucket, receiving 0 points. Another stock, with a volatility of either 2 to 4 or 6 to 8, would receive 50 points, because it belongs to a closer bucket, either 2nd or 4th respectively.

4.5 Validation

Using the experimental procedure described in section 4.4 we created the 20 portfolios. These were listed in Table 4.2. In this section we validate weather the portfolios

resulted from our system are diversified and financially performant. We will further use the labels assigned in the 4th columns of Table 4.2 to refer to individual portfolios created by our system. For the case that implements investor preferences we selected preference 1 (stocks that have value for paid dividends last year ordered by the amount paid) and preference 5 (stocks that have high average returns in the past 5 years ordered by the amount of returns).

We assume the creation of portfolios on date 30th of December 2016. We use data from year 2017 to evaluate the performance of the portfolios constructed by our system. We calculate variance, returns, Sharpe Ratio and Information Ratio after a period of 1 month (1st of January - 1st of February 2017), 3 months (1st of January - 1st of April 2017), 6 months (1st of January - 1st of July 2017) and 12 months (1st of January - 29st of December 2017) of holding.

4.5.1 Benchmark

To be able to measure to what extend is our system useful for financial advisers we need to asses it's financial performance and diversification similarity with a recognized benchmarks. We chose MSCI World Index as template sector and region allocation. This standard allocation is implemented by a separate system. We call it *algorithmic advisor* because it is specifically programmed to construct portfolios that match exactly the target allocation. One important advantage of constructing a benchmark in this way is that it enables us to relate the financial performance of portfolios constructed by our proposed methods with random portfolios that respect the diversification standard.

The algorithmic advisor implements the diversification requirement by ensuring the following constraints are met:

1. No individual stock has an exposure of more than 5%
2. Sector allocation must be similar to MSCI World Index[6]
3. Region allocation must be similar to MSCI World Index[7]
4. It must integrate investor preferences

The algorithm starts with the addition of a random stock with 5% exposure, followed by validation of constraints. If constraints are not met, the algorithm takes the desired sector and region allocations as template and modifies the portfolio by adding stocks or adjusting exposures of held ones.

[6] https://www.msci.com/documents/1296102/1362201/MSCI-MIS-World-Brochure-May-2018.pdf/8f4db460-a0cb-d845-226b-fe2472b3dc08

[7] https://www.msci.com/world

The system needs a sufficiently large pool of stocks to ensure a list of choices to rank and apply investor preferences on. Our data set contained too few samples in Asian regions and insufficient sectors diversification in combination with European regions. There are six stocks in region 'Asia ex Japan', divided in sectors Communication, Technology, Financial and Consumer Staples. Japan has a total of seven stocks, in sectors such as Utilities, Energy, Consumer Staples and Consumer Discretionary. The issue appears because the few stocks in Europe and UK occupy the same sectors as the Asian region. Thus, after completing the region allocation with either Asian or European stocks, the other region will not be recommended because the sectors they contain are not needed anymore. The following table shows the number of samples for each region and sector.

Table 4.3: Frequency of data samples containing specific region and sector

Region	Samples
USA	1653
Emerging Markets	85
Others	83
Canada	70
Europe ex UK	43
UK	19
Asia ex Japan	7
Japan	6

Sector	Samples
Consumer Staples	403
Consumer Discretionary	390
Financial	321
Technology	254
Industrials	196
Materials	132
Energy	111
Utilities	96
Communications	63

As a solution, we modified the algorithm to be *greedy* with these regions: the iterations start from the sector with least exposure to be filled and, for each sector, we first try to add stocks from the regions with fewer samples. The algorithm allows a level of permissiveness to be set: the stock added by greed cannot be in the second half of the ranking. With the dataset described in table 4.3 we get a list of 100 to 300 possible choices of stocks at each iteration. Only the last three iterations generate less that 50 stocks to choose from.

4.5.2 Results

The validation methodology is guided by the following three questions:

Q1. How much of the standard sector and region allocation is respected in the resulted portfolios? To answer, we measure the similarity between sector and region allocation vectors of different resulted portfolios with the allocation vectors given by the benchmark advisor.

As mentioned in section 4.5.1 the benchmark advisor is programmed to select a

mix of stocks that meet precisely the region and sectors allocations of MSCI World Index, taken by us as standard for this study. We encode allocation exposures of both the benchmark and portfolios described in Table 4.2 into vectors and use MSE to calculate the sector and region allocation similarity of each created portfolio with the target allocations.

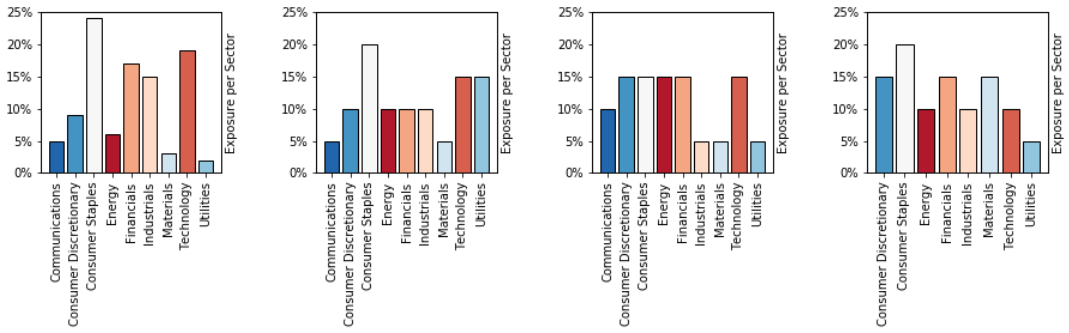
Sector allocation:

Table 4.4 shows the similarity score and number of different sectors for the top 5 closest sector allocations with benchmark and the 5 most distant. The highest similarity score is 82.79%, obtained by portfolio created using clustering by sector and region attributes, agglomerative algorithm and similarity with Amazon for stock selection. The majority of the portfolios have high sector allocation similarity with benchmark. 70% have sector allocation more than 70% similar to benchmark and 85% of portfolios have sector allocation more than 50% similar to benchmark. 15% of the portfolios contain at least 5% exposure for all of the 9 available sectors and 30% of them include 8 sectors out of 9. This is desired from the perspective of diversification. 25% contain 7 sectors out of 9, and the rest of 10% have 6 or less different sectors.

Table 4.4: First and last 5 portfolios ranked according to similarity in sector allocation with benchmark

Order	Portfolio	Sector allocation similarity	No. sectors contained
1	portfolio_ml_sere_aggl_similarity	82.79%	9
2	portfolio_ml_sere_kmeans_random	81.13%	9
3	portfolio_ml_corr_spectral_random	81.13%	8
4	portfolio_ml_sere_aggl_sharpe	80.86%	8
5	portfolio_ml_sere_kmeans_preferences	80.35%	9
...	...	...	...
16	portfolio_ml_corr_spectral_similarity	67.81%	7
17	portfolio_ml_corr_aggl_similarity	62.50%	7
18	portfolio_ml_corr_aggl_sharpe	37.10%	3
19	portfolio_ml_corr_spectral_sharpe	31.32%	5
20	portfolio_ml_corr_kmeans_sharpe	20.84%	3

The target sector allocation is divided as depicted in figure 4.5(a). Figures 4.5(b), 4.5(c) and 4.5(d) show the sector division of portfolios with top 3 closest sector allocation: Figure 4.5(b) has the allocation of *portfolio_ml_sere_aggl_similarity*, which is 82.79% similar to benchmark sector allocation. Figure 4.5(b) shows the allocation



(a) (b) (c) (d)

Figure 4.5: Figure 4.5(a) shows the template sector allocation. The next three figures show the top 3 most similar sector allocations: Figure 4.5(b) is of portfolio_ml_sere_aggl_similarity, Figure 4.5(c) is of portfolio_ml_sere kmeans_random and Figure 4.5(d) is of portfolio_ml_corr_spectral random.

of *portfolio_ml_sere_kmeans_random*, which is 81.13% similar to benchmark sector allocation. Lastly, Figure 4.5(d) shows the allocation of *portfolio_ml_corr_spectral_random*, which is also 81.13% similar to benchmark sector allocation. We notice the proportions among sectors' exposure are more or less similar, with each portfolio allocating high exposure to sector *Consumer Staples* and low exposure to sectors *Communications*, *Materials* and *Utilities* as does the benchmark. The variation of exposure of sectors among portfolio construction methods are between +-5% and +-10%. We notice that the allocation in Figure 4.5(d) is missing sector *Communications*. The next closest allocation that has all 9 sectors is on the 5th place with 80.35% similarity and belongs to portfolio *portfolio_ml_sere_kmeans_preferences*. The allocation of this portfolio is more uniformly distributed, with no sector having an exposure more than 15%, while the portfolio in Figure 4.5(d) follows better the proportions of the template allocation.

Region allocation:

For validation of region allocation we use a similar approach. Table 4.5 shows the similarity score and number of different regions included for the top 5 and last 5 most similar region allocations with benchmark. We notice a shorter range of values compared to sector allocation, with a maximum similarity of 79.5% and a minimum of 47.17%, while the sector similarity ranged from 82.79% to 20.84%. 45% of portfolios have region allocation more than 70% similar to benchmark and 95% have region allocation more than 50% similar to benchmark. In contrast with sector allocation, the majority of created portfolios don't include all available regions, with 45% of portfolios having 4 or less regions out of the 8 available. 15% of portfolios have 5 regions out of 8, 25% of portfolios contain 6 regions, 5% contain 7 regions out of 8 and 10% contain all 8 regions.

Table 4.5: First and last 5 portfolios ranked according to similarity in region allocation with benchmark

Order	Portfolio	Region allocation similarity	No regions contained
1	portfolio_ml_corr_aggl_preferences	79.50%	5
2	portfolio_ml_corr_kmeans_preferences	78.09%	5
3	portfolio_ml_corr_kmeans_random	73.92%	4
4	portfolio_ml_corr_spectral_preferences	73.92%	4
5	portfolio_ml_sere_aggl_preferences	73.16%	6
...	...	...	...
16	portfolio_ml_corr_aggl_similarity	57.45 %	3
17	portfolio_ml_corr_aggl_sharpe	57.45%	3
18	portfolio_ml_corr_kmeans_sharpe	52.98%	2
19	portfolio_ml_corr_spectral_sharpe	52.98%	2
20	portfolio_ml_corr_spectral_similarity	47.17%	1

Figure 4.6(a) shows the desired region allocation. Figures 4.6(b), 4.6(c) and 4.6(d) show the region allocation of portfolios with top 3 closest sector allocation: Figure 4.6(b) contains the allocation of *portfolio_ml_corr_aggl_preferences*, which is 79.5% similar to benchmark region allocation; Figure 4.6(c) shows the allocation of *portfolio_ml_corr_kmeans_preferences*, which has region allocation 78.09% similar to benchmark; lastly, Figure 4.6(d) shows the region division of *portfolio_ml_corr_spectral preferences*, with an allocation similarity score of 73.92%. In all of these portfolio the *USA* region has the most exposure, between 55% and 70% among portfolios, which imitates the benchmark that has 52%. The rest of the regions have low exposure as benchmark, with *Emerging Markets* having the highest exposure among them, which is relatively close to the benchmark.

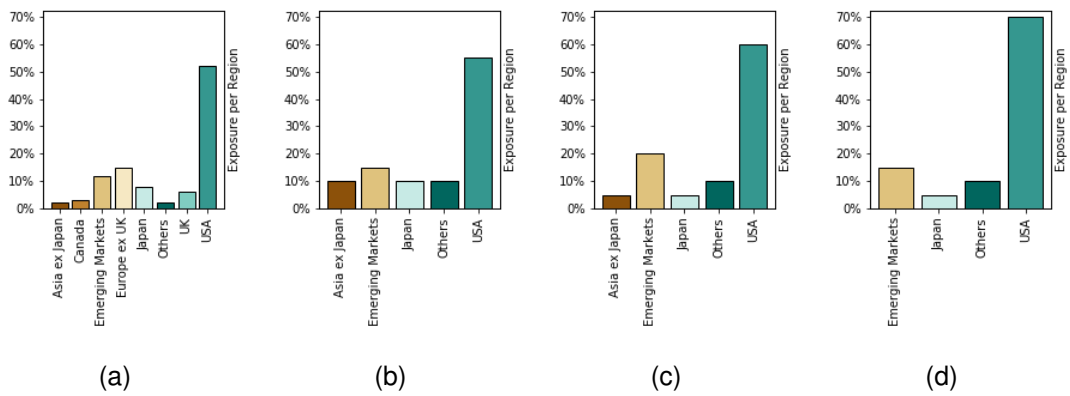


Figure 4.6: Figure 4.6(a) shows the template region allocation. The next three figures show the top 3 most similar region allocations: Figure 4.6(b) is of portfolio_ml_corr_aggl_preferences, Figure 4.6(c) is of portfolio_ml_corr kmeans_preferences and Figure 4.6(d) is of portfolio_ml_corr_kmeans random.

The top 3 most similar allocations do not contain all 8 sectors. Actually, the closest allocations to incorporate all 8 sectors are on the 10th and 12 place, belonging to *portfolio_ml_sere_aggl_sharpe* and *portfolio_ml_sere_kmeans_sharpe*. These allocations are depicted in Figure 4.7(a) and 4.7(b) respectively. We notice lower exposure

thank benchmark for region *USA* and undesirable high exposure for region *Others*. The rest of the regions are close to benchmark, with varying exposures by 5%.

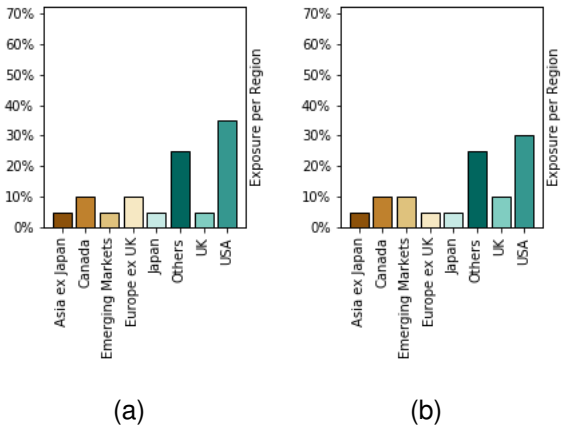


Figure 4.7: Region allocation of (a) portfolio_ml_sere_aggl_sharpe and (b) portfolio_ml_sere_kmeans_sharpe

Q2. What is the level of portfolio variance in the resulted portfolios compared to benchmark advisor?

As described in section 2.3.1, variance is used to measure diversification of portfolios. We will use formula (2.4) to calculate and compare the variance of the benchmark portfolio with the portfolios created by our system.

Figure 4.8 shows the distribution of variance of our 20 portfolios created with machine learning at different moments in time. The red line is the portfolio created by the benchmark advisor, detailed in section 4.5.1. We notice that more than 50% of our portfolios start with higher variance than benchmark in the first month of holding. After 3 months the majority of the portfolios have very low variance, close to the benchmark's variance at 1 month. For the rest of the year, our portfolios have higher variance than the benchmark, with few abnormal values.

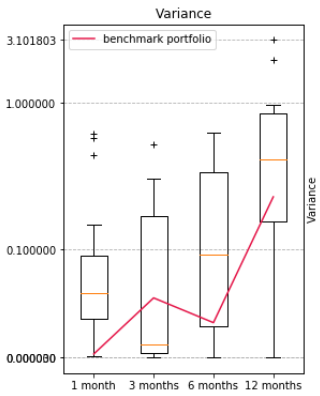


Figure 4.8: Variance of portfolios after 1 month, 3 months, 6 months and 12 months of holding

The following 4 figures show the trend of the variance of our portfolios (each color represents one portfolio):

1. Figure 4.9 groups portfolios that had most similar allocation in **Q1**. The only portfolio with a declining variance over time is *portfolio_ml_sere_aggl_similarity*, the same one that has most similar sector allocation with the benchmark in

Q1. Random selection of stocks from clusters results in variance close to benchmark, but with a more volatile trend. This shows that the clustering itself contributes to desired variance values and that the clustering technique is a good choice as a base method for diversification.

2. Figure 4.10 groups portfolios in which the selection of stocks from clusters was done by taking the top one ranked according to investor preferences. Over time, all lines show an increasing trend in variance. The portfolios clustered using features *sector* and *region* have very low variance, close to 0, in the first 6 months of holding and the variance increases to meet the benchmark in the following 6 months. Among the portfolios in which clustering was done using the correlation matrix, the agglomerative algorithm is the one that results in the closest to benchmark in terms of variance trend.

3. Figure 4.11 groups portfolios in which the stock most similar to Amazon was selected from each cluster. The trends in this group are more or less similar to each other, meaning that they have something in common. Overall variance is higher than benchmark.

4. Figure 4.12 groups portfolios in which the stock with the highest Sharpe Ratio is picked from clusters. The variance of the portfolios in this group show an inverse trend compared to the one of the benchmark. The portfolios clustered using a correlation matrix show almost identical trends. Overall, the trends show increased risk in the portfolios resulted from this method.

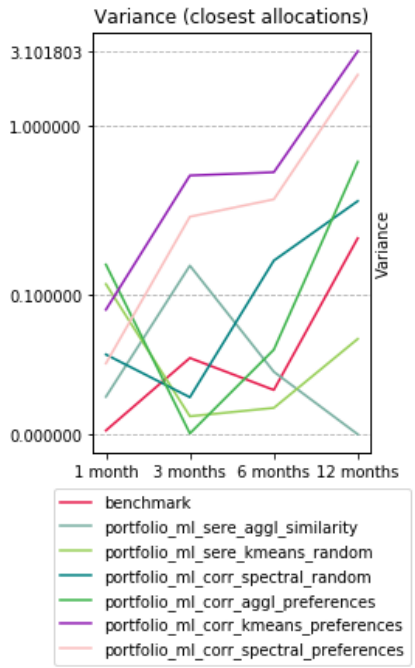


Figure 4.9: Variance of portfolios with closest allocation to benchmark.

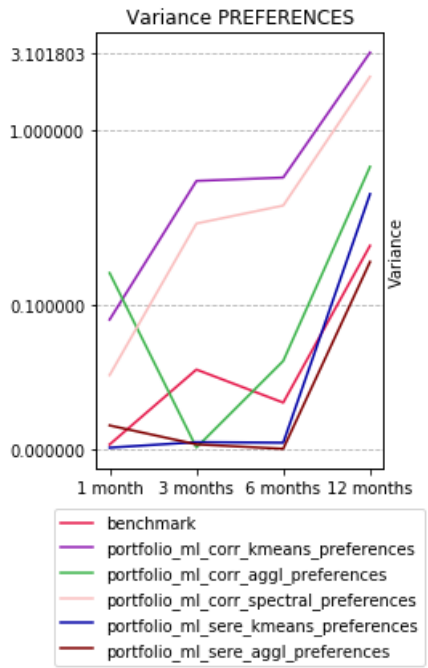


Figure 4.10: Variance of portfolios with stock selection method by preferences

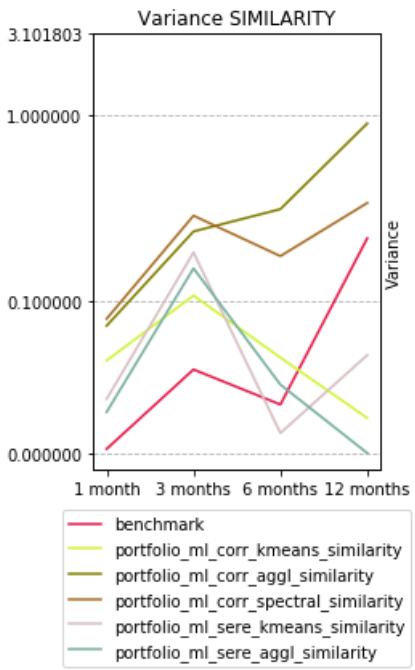


Figure 4.11: Variance of portfolios with stock selection method based on similarity with Amazon

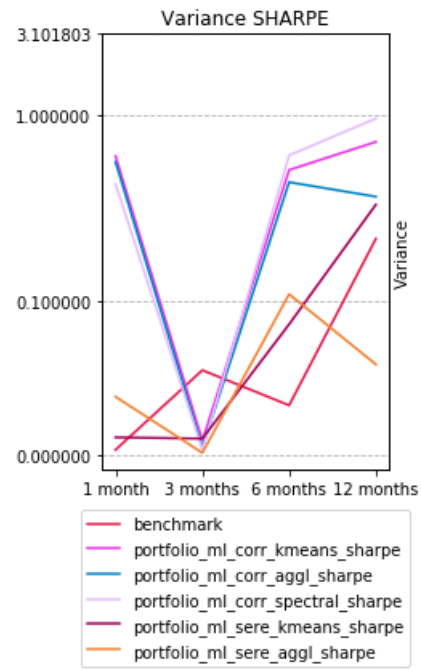


Figure 4.12: Variance of portfolios with stock selection method based on highest Sharpe Ratio

Q3. What is the performance of the resulted portfolios in the market?

We assess performance of portfolios by calculating returns, Sharpe Ratio and Information Ratio on all portfolios after the specified periods of time. Moreover, we compare the returns of our portfolios after 1 month of holding with the average monthly historical return of S&P500 Index in the past 10 years (2008-2018).

In Figure 4.13 we show the distribution of the returns of our portfolios over the same intervals of time as before. More than 75% of portfolios have higher returns than benchmark in the first month, with the highest returns 30.4% offered by portfolio constructed by clustering based on correlation matrix, using spectral algorithm and selecting the stock most similar to Amazon. The majority of the portfolios sustain their good returns over the year: 90% have returns higher than benchmark after 3 and 85% are above benchmark after 6 months and 12 months. The highest return has value of 33.67% and is produced by *portfolio_ml_corr_kmeans_similarity* after 12 months of holding. This portfolio is 68.29% similar with benchmark in terms of sector allocation and 57.46% similar with benchmark in terms of region allocation, but produces returns 18 times higher than benchmark.

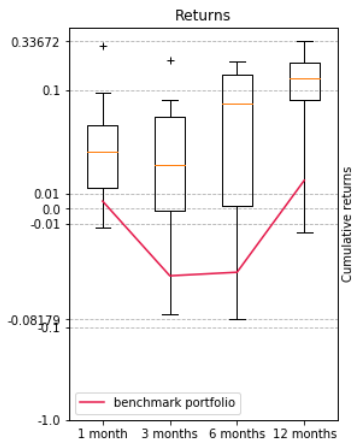


Figure 4.13: Return of portfolios after 1 month, 3 months, 6 months and 12 month of holding

The following 4 figures are grouped as in **Q2**. Figure 4.14 shows the variance trends of the 6 portfolios that were most similar in terms of allocation in Q1: 4 have higher returns than benchmark after the first month of holding, with values between 5% and 5.8%. With one exception, the portfolios have consistently good returns for the rest of the year. In figure 4.15 we see the trends in returns of portfolios created with investor preferences stock selection. The returns in this case show again similar trends, consistently and considerably higher than benchmark. The only portfolio with a decreasing trend in returns is grouped in Figure 4.16. The rest of the portfolios in this group show a similar return trend with the benchmark. Lastly, the portfolios grouped in figure 4.17 show good returns for the higher variance we saw in **Q2**, with a close to stable trend in the first half of the year and an altogether increase in the following 6 months of holding.

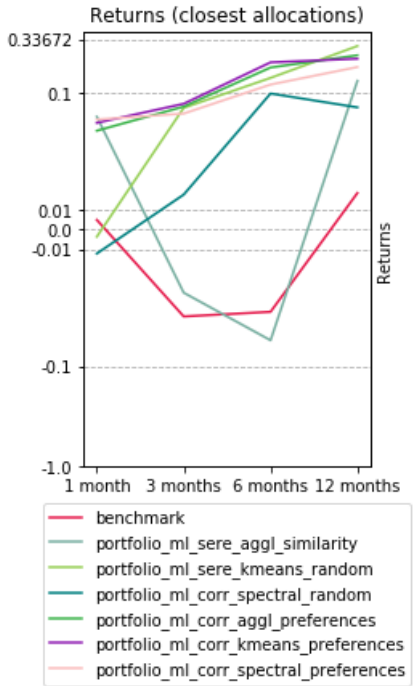


Figure 4.14: Variance of portfolios with closest allocation to benchmark

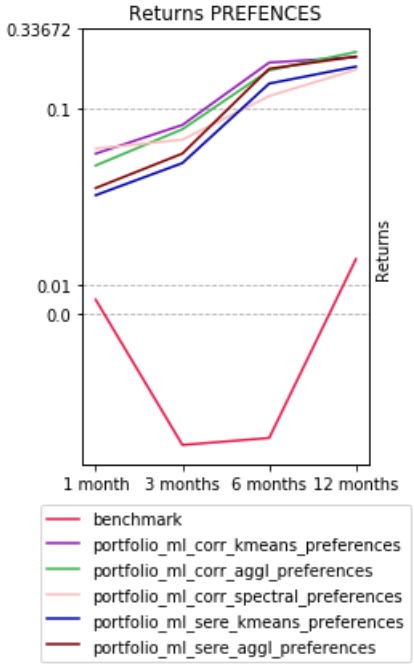


Figure 4.15: Variance of portfolios with stock selection based on investor's preferences

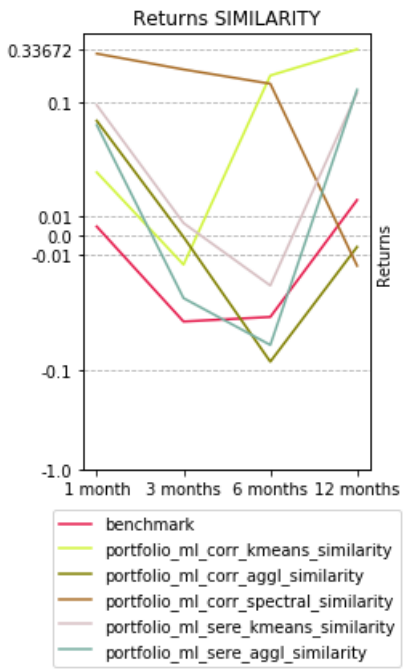


Figure 4.16: Variance of portfolios with stock selection method based on similarity with stock Amazon

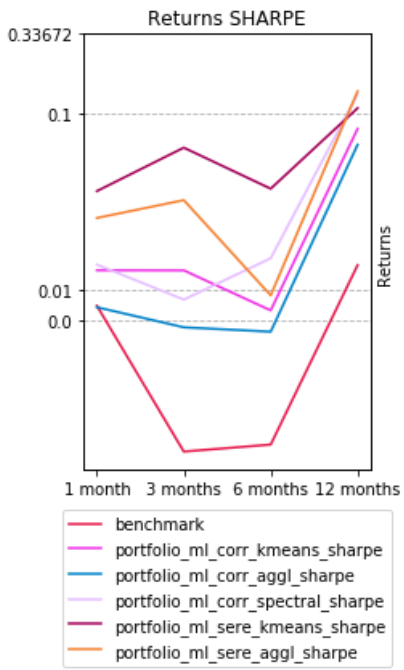


Figure 4.17: Variance of portfolios with stock selection method based in highest Sharpe Ratio

Best returns are obtained by portfolios constructed by clustering based on correlation matrix, with returns ranging from 0.4% to 30.45%. The method that clusters based on features sector and region gets highest returns in case of portfolio clustered using k-means algorithm and similarity selection method, with a value of 9.22%. Two portfolios present a loss after 1 month of holding, both using a random stock selection method. We see a similar trend after 3 months, 6 months and 12 months of holding, but the portfolios that have returns lower than benchmark vary among periods of time. It is important to note that these 'dips' are usually the exception.

A good portfolio not only has good returns, but also does not take unnecessary additional risk. To put both risk and returns into perspective, we calculate Sharpe and Information Ratios. Sharpe Ratio is a very well known method of measuring the excess returns an investor may gain by assuming the extra risk. To calculate it, we use formula $S = \overline{R}_a R_f / \sigma_p$, where $\overline{R}_a$ is the average return of the asset, R_f is the return of a security with zero risk[8] [9] and σ_p is the standard deviation of the portfolio. Zero risk securities have a certain future return. They only exist in theory, as any investment carries a small amount of risk. The returns of such securities are close to US interest rate.

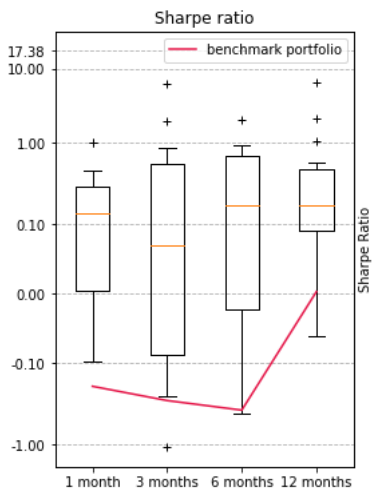


Figure 4.18: The distribution of Sharpe Ratio among created portfolios at different moments of holding.

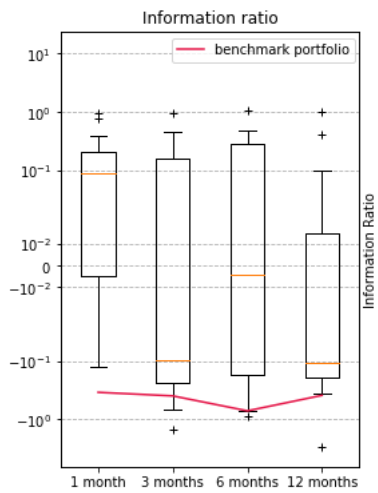


Figure 4.19: The distribution of Information Ratio among created portfolios at diferent moments of holding.

There is no ideal value for the Sharpe Ratio. Usually, a value of 1 is good, a value of 2 is better and a value of 3 is great. Some of our portfolios reach values of 6,53 in case of *portfolio_ml_sere_aggl_similarity* after 12 months and 17,38 in case

[8]= http://www.global-rates.com/interest-rates/libor/american-dollar/2016.aspx
[9]= http://www.global-rates.com/interest-rates/libor/american-dollar/2017.aspx

of *portfolio_ml_sere_aggl_preferences* after 6 months. The ratio is greatly influenced by the value in the variance of the portfolio, which is the denominator in the Sharpe Ratio formula. Thus, even if some of the ratios may look off the charts, they are due to very low portfolio variance, meaning high diversification. All portfolios created using the clustering technique have better Sharpe Ratio than benchmark after 1 month of holding. By the end of the year, 90% of them maintain a higher than benchmark ratio.

Information Ratio is similar, but used for assessing the excess returns against a popular index, such as S&P500. Formula is similar to the one of Sharpe Ratio, with benchmark index returns instead of risk free return. From this values we should asses whether our portfolios outperforms or not the strategy of passively following S&P500 Index.

The Information Ratio of the benchmark is negative, thus it is not hard for all portfolios to perform better. 70% of portfolios have positive information ratio after 1 month of holding, meaning higher returns than passively following the S&P500 index. 35% of them maintain the same good returns by the end of the year.

To put the performance of portfolios into a better perspective, we use Figure 4.20 to describe the distribution of the monthly average return of S&P500 for the past 10 years. The scattered dots represent the 1 month returns of our portfolios. The plot shows that all of our portfolios have had higher returns after 1 month than the average 1 month return of S&P500 (the red horizontal line), calculated historically.

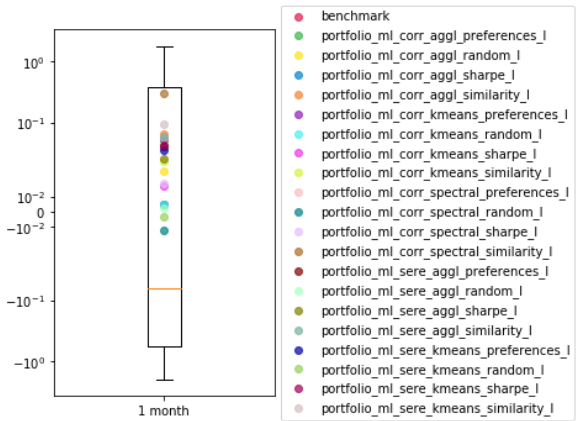


Figure 4.20: The boxplot shows the distribution of historical average S&P500 1 month return. The dots are the 1 month returns of our portfolios.

4.6 Evaluation

To asses the practical utility of the system resulted from this study, we held an evaluation session with five stakeholders employed at Ortec Finance. We presented three use-cases of our system and their financial outcome for a fictional investor. The use-cases presented integrate investor's preferences by recommending according to risk profile. The audience, comprised of three product consultants, one researcher and one programmer, evaluated the system by filling up a questioner, which is presented in Annex B.

Table 4.6 summarizes the criteria graded by participants and the respective scores. The system received a total average score of 3.6 out of 5 which is consid-

Table 4.6: Evaluation criteria and respective average score

Aspect	Average Score [1,5]
Level of diversification in the created portfolios	4
Financial performance of the created portfolios	3.8
Adaptability of the system to different use cases	2.6
Speed up the advisory process	4
Interest in further development of the system	3.6
Total	**3.6**

ered good. The lowest score received was on aspect *use case adaptability* caused by the variability of results between different methods. The methodology needs more research, but this score highlights how important it is for advisor that the results are consistent between methods. The highest scores were obtained on aspects *level of diversification* and *speeding up the advisory process*, followed by *financial performance of created portfolios*. This shows the potential of our system as it is able to incorporate diversification constraints and make good profits, with more tweaking needed on the methodology of giving personalized recommendations. The high score in *speeding up the advisory process* shows the practical utility of this study.

4.7 Conclusions & Discussion

The chapter presented our approach in creating a diversified portfolio recommender system that integrates machine learning. We validated the system against the algorithmic advisor and S&P500 Index and obtained promising results.

When looking at diversification from the perspective of financial advisers, we asses similarity of allocations with MSCI Index chosen as standard and count the number of sectors and regions included in the portfolios recommended by our system. Results vary among portfolio creation methods, with 70% of portfolios having at least 70% similarity with benchmark in sector allocation and 45% of portfolios having more than 70% similarity in region allocation. We notice that the system diversifies consistently better across sectors than regions. We believe this is caused by the chosen unit of minimum exposure of 5%. For instance, the sector allocation template contains values of 2%, 3%, while our system adds stocks to portfolios in incremental units of 5%. This means that when targeting an exposure of 2-3%, our system can match it with either 5% or 0%. Sector allocation contains more values close to numbers divisible by 5 such as 9, 6, 19 or 24. Our system could meet these values with an error of 1%. In case of region allocation, values such as 2, 3, 8, 12 and 52 are more common, allowing higher errors of 2%-3% for each sector. Adding

up these little differences, our methodology could not perform better. This is a limitation of this study. More precision in allocation exposures could improve the results even more.

We analyze the possibility that the consistent high exposure allocated for region *USA* by our system may be caused by the imbalance in our data set. As the stocks from this region are more frequent, there is a higher chance for them to be picked no matter the selection method. Table 4.3 summarizes the frequency of each sector and region in the used data set. We also notice in this table that region *Japan* has the least number of samples and it still gets selected next to region *USA* in Figures 4.6(b), 4.6(c) and 4.6(d). Region *Asia ex Japan* is another similar instance. Thus we cannot conclude that high frequency is the only cause for high *USA* exposure. It may also be the case that stocks from regions *Canada, Europe* and *UK* are considered similar as these countries have similar economic characteristics. The data imbalance issue is persistent in real life as well, with USA being the leading country in stock investment and, of course, having the most numerous number of securities trading.

From the perspective of MPT, diversification is measured mathematically with variance. We compared the variance trend of portfolios created by our system with the variance of the benchmark portfolio. The variance of our portfolios varies between 0,00003 and 3.1, with most of the values below 1 which is a generally accepted margin. Our portfolios are close to benchmark, but the majority of them have higher variance, meaning they assume more risk. In most of the cases variance increases over time, which is normal and expected in practice. Overall, the variance trends are volatile and difficult to predict, but they all are in acceptable boundaries.

From the perspective of possible future users of the system, diversification is incomplete without good returns. We showed that the portfolios created by our system take more risk. The returns in exchange are considerable higher, thus the risk previously mentioned offers a higher reward. This is also confirmed by the high values of Sharpe Ratio. The positive values of Information Ratio signify that our portfolios perform better than S&P500 Index. It is important to note that 2017 was one of the best year for stocks, with consistent positive returns [69].

Chapter 5

Conclusions and recommendations

5.1 Conclusions

In this study we bring our contribution in the exploration of machine learning capabilities in the domain of financial advisory. We investigate two aspects of financial advisory, namely prediction of companies' future performance and recommendation of diversified portfolios of stocks.

The first question of our research was *"To what extent can the buy, hold, sell recommendations list of financial analysts be automatically generated from easily accessible financial data, using supervised learning methods?"*. We use financial data from Bloomberg as this is the main data source for Ortec employees, thus it is considered *easily accessible* in our case. To our knowledge, Thakur's study [33] was the only one to try prediction of Analyst Rating as calculated by Bloomberg. The first part of our study extends on this work. In contrast, we use a smaller number of features and estimate with basic regression algorithms and ensemble classification methods. We used a novel approach of clustering the observations prior to estimation and adding the cluster number as a feature. The regression models and most of the classification models perform better than benchmark, meaning that they learn something, but not sufficient to be useful in the financial domain. Support Vector Machines models perform poorly in the classification task, but adding the cluster number as feature greatly improves the results. This interesting finding suggests that feature engineering could be a proper next step for future research. Data analysis we performed on the explanatory power of Analyst Rating as taken from Bloomberg on the future one year returns shows a very weak linear correlation. Nevertheless, the best results were obtained with polynomials of ninth and eleventh degree. Our study shows that prediction of analyst rating is a complex task, but with more research and implementation of fail-safe mechanisms, such as the multi-agent approach proposed by Barbosa [25], machine learning methods could perform at least as good as humans.

The second question of this study was *"How can we design recommender systems for financial advice by using a data driven approach?"*. In the exploration of previous research, we looked at diversification by means of clustering. The second part of this research extends several studies as described in Section 4.3. In addition to these studies, we integrate investor preferences in the stock selection process for portfolio construction. This enables us to integrate machine learning methods in a constrained-based recommender system. The constraints of portfolio construction domain, described in Section 2.3.2, are implemented by design by the clustering technique. The integration of investor preferences allows the system to make personalized recommendations. The methodology described in details in Section 4.4 answers the second research question.

The main research that guided this study was *"To what extend machine learning and recommender engines methods support financial advisers in the process of portfolio creation, tailored to individuals?"*. To answer this question we validate the system we designed in Section 4.4 with various settings and evaluate it's investor preferences integration functionality with stakeholders, as described Section 4.6. The allocation similarity varies across portfolio construction methods, with an average of 81% similarity with benchmark in the best case and 34% at worst. The simulated financial performance shows a good risk-return trade-off in the created portfolios, with several portfolios having a Sharpe Ratio above 3 and positive Information Ratio against S&P500. Performance results vary across portfolio construction methods, thus more research is needed is improving the methodology and making it more precise. The stakeholders evaluation brought the same conclusion. Moreover, in their opinion the advisory process would be quicker with the support of a system such as ours. Nevertheless, our study proves that the clustering methodology is a solid starting point for future research in automation of financial advisory and that integration of machine learning techniques with constraint based recommender systems is possible and a promising approach.

5.2 Recommendations

A first step towards improving prediction of analyst rating is to use a balanced data set with labels from individual advisory houses. Our findings suggest that feature engineering could bring better data interpretation for the more complex machine learning models. Intuitively, it would seem an appropriate approach, as analyst recommendations tend to be subjective and there may be multiple layers of thinking between the raw financial data and the purchasing recommendation. The results of diversified portfolio construction through clustering should be validated on data sets longer than one year, as suggested by the conclusions of stakeholder's evaluation.

Usage of other clustering algorithms is also encouraged. A more correct approach would be to choose the number of clusters according to desired values in chosen metrics, which in consequence raises more research questions.

It would be interesting to see from future research prediction of analyst rating for individual security. As analysts in real life use multiple sources of information to reach their recommendations, future research should consider integrating these methods in a multi-agent approach. Nevertheless, our study opens perspectives to other applications. For instance, diversification across types of assets is an even more complex task that can be treated similarly.

References

[1] "Wikipedia list of trading losses." [Online]. Available: https://en.wikipedia.org/wiki/List_of_trading_losses

[2] "Financial advisor: Job description." [Online]. Available: https://www.thebalance.com/financial-advisor-career-information-526017

[3] E. A. Helfert, *Techniques of financial analysis: A practical guide to managing and measuring business performance*. Irwin,, 1994, no. 658.15.

[4] "Guide to analyst rating assessment methods." [Online]. Available: https://www.marketwatch.com/tools/guide.asp

[5] J. Driessen and L. Laeven, "International portfolio diversification benefits: Cross-country evidence from a local perspective," *Journal of Banking & Finance*, vol. 31, no. 6, pp. 1693–1712, 2007.

[6] H. Markowitz, "Portfolio selection," *The journal of finance*, vol. 7, no. 1, pp. 77–91, 1952.

[7] J. V. Uspensky, "Introduction to mathematical probability," 1937.

[8] "When the stock market last crashed, these sectors fared best." [Online]. Available: https://www.marketwatch.com/story/when-the-stock-market-last-crashed-these-sectors-fared-best-2015-08-21

[9] B. S. P. B. K. Francesco Ricci, Lior Rokach, *Recommender Systems Handbook*. Springer, 2010.

[10] "Web site of job finder." [Online]. Available: http://www.jobfinder.com/

[11] A. S. Das, M. Datar, A. Garg, and S. Rajaram, "Google news personalization: scalable online collaborative filtering," in *Proceedings of the 16th international conference on World Wide Web*. ACM, 2007, pp. 271–280.

[12] G. Linden, B. Smith, and J. York, "Amazon. com recommendations: Item-to-item collaborative filtering," *IEEE Internet computing*, vol. 7, no. 1, pp. 76–80, 2003.

[13] P. Covington, J. Adams, and E. Sargin, “Deep neural networks for youtube recommendations,” in *Proceedings of the 10th ACM Conference on Recommender Systems*. ACM, 2016, pp. 191–198.

[14] “Article on the netflix challenge.” [Online]. Available: https://mesuvash.github.io/blog/2015/Quora-Machine-Learning-In-Recommender-Systems/

[15] J. R. Quinlan, “Induction of decision trees,” *Machine learning*, vol. 1, no. 1, pp. 81–106, 1986.

[16] M. J. McGowan, “The rise of computerized high frequency trading: use and controversy,” *Duke L. & Tech. Rev.*, p. i, 2010.

[17] N. Milosevic, “Equity forecast: Predicting long term stock price movement using machine learning,” *arXiv preprint arXiv:1603.00751*, 2016.

[18] W. W. Cohen, “Fast effective rule induction,” in *Machine Learning Proceedings 1995*. Elsevier, 1995, pp. 115–123.

[19] K. P. Murphy *et al.*, “Naive bayes classifiers,” *University of British Columbia*, vol. 18, 2006.

[20] D. Heckerman, D. Geiger, and D. M. Chickering, “Learning bayesian networks: The combination of knowledge and statistical data,” *Machine learning*, vol. 20, no. 3, pp. 197–243, 1995.

[21] R. P. Schumaker and H. Chen, “A discrete stock price prediction engine based on financial news,” *Computer*, vol. 43, no. 1, 2010.

[22] S. Liu, F. Liu, C. Yu, and W. Meng, “An effective approach to document retrieval via utilizing wordnet and recognizing phrases,” in *Proceedings of the 27th annual international ACM SIGIR conference on Research and development in information retrieval*. ACM, 2004, pp. 266–272.

[23] D. Nadeau and S. Sekine, “A survey of named entity recognition and classification,” *Lingvisticae Investigationes*, vol. 30, no. 1, pp. 3–26, 2007.

[24] E. Marsh and D. Perzanowski, “Muc-7 evaluation of ie technology: Overview of results,” in *Seventh Message Understanding Conference (MUC-7): Proceedings of a Conference Held in Fairfax, Virginia, April 29-May 1, 1998*, 1998.

[25] R. P. Barbosa and O. Belo, “Autonomous forex trading agents,” in *Industrial Conference on Data Mining*. Springer, 2008, pp. 389–403.

[26] ——, "Multi-agent forex trading system," in *Agent and multi-agent technology for internet and enterprise systems*. Springer, 2010, pp. 91–118.

[27] T. G. Dietterich, "Ensemble methods in machine learning," in *International workshop on multiple classifier systems*. Springer, 2000, pp. 1–15.

[28] E. A. Gerlein, M. McGinnity, A. Belatreche, and S. Coleman, "Evaluating machine learning classification for financial trading: An empirical approach," *Expert Systems with Applications*, vol. 54, pp. 193–207, 2016.

[29] J. G. Cleary and L. E. Trigg, "K*: An instance-based learner using an entropic distance measure," in *Machine Learning Proceedings 1995*. Elsevier, 1995, pp. 108–114.

[30] J. R. Quinlan, *C4. 5: programs for machine learning*. Elsevier, 2014.

[31] N. Landwehr, M. Hall, and E. Frank, "Logistic model trees," *Machine learning*, vol. 59, no. 1-2, pp. 161–205, 2005.

[32] R. C. Holte, "Very simple classification rules perform well on most commonly used datasets," *Machine learning*, vol. 11, no. 1, pp. 63–90, 1993.

[33] S. Thakur, T. Vadpey, and S. Ayyar, "Predicting stock prices and analyst recommendations."

[34] F. S. Hanifah, H. Wijayanto, and A. Kurnia, "Smotebagging algorithm for imbalanced dataset in logistic regression analysis (case: Credit of bank x)," *Applied Mathematical Sciences*, vol. 9, no. 138, pp. 6857–6865, 2015.

[35] R. R. Hocking, "A biometrics invited paper. the analysis and selection of variables in linear regression," *Biometrics*, vol. 32, no. 1, pp. 8–9, 1976.

[36] V. Fonti and E. Belitser, "Feature selection using lasso," *VU Amsterdam Research Paper in Business Analytics*, 2017.

[37] R. Genuer, J.-M. Poggi, and C. Tuleau-Malot, "Variable selection using random forests," *Pattern Recognition Letters*, vol. 31, no. 14, pp. 2225–2236, 2010.

[38] Y. Freund and R. E. Schapire, "A decision-theoretic generalization of on-line learning and an application to boosting," *Journal of computer and system sciences*, vol. 55, no. 1, pp. 119–139, 1997.

[39] K. Swingler, "Financial prediction: Some pointers, pitfalls and common errors," *Neural Computing & Applications*, vol. 4, no. 4, pp. 192–197, 1996.

[40] D. S. Wilks, “Cluster analysis,” in *International geophysics*. Elsevier, 2011, vol. 100, pp. 603–616.

[41] S. Lloyd, “Least squares quantization in pcm,” *IEEE transactions on information theory*, vol. 28, no. 2, pp. 129–137, 1982.

[42] E. W. Forgy, “Cluster analysis of multivariate data: efficiency versus interpretability of classifications,” *biometrics*, vol. 21, pp. 768–769, 1965.

[43] S. Kotsiantis and P. Pintelas, “Recent advances in clustering: A brief survey,” *WSEAS Transactions on Information Science and Applications*, vol. 1, no. 1, pp. 73–81, 2004.

[44] G. N. Lance and W. T. Williams, “A general theory of classificatory sorting strategies: 1. hierarchical systems,” *The computer journal*, vol. 9, no. 4, pp. 373–380, 1967.

[45] B. King, “Step-wise clustering procedures,” *Journal of the American Statistical Association*, vol. 62, no. 317, pp. 86–101, 1967. [Online]. Available: http://www.jstor.org/stable/2282912

[46] F. Murtagh, “A survey of recent advances in hierarchical clustering algorithms,” *The Computer Journal*, vol. 26, no. 4, pp. 354–359, 1983.

[47] U. V. Luxburg, “A tutorial on spectral clustering,” 2007.

[48] L. Rokach and O. Maimon, “Chapter 15 clustering methods.” [Online]. Available: http://citeseerx.ist.psu.edu.ezproxy2.utwente.nl/viewdoc/summary?doi=10.1.1.149.9326

[49] D. M. Witten and R. Tibshirani, “A framework for feature selection in clustering,” *Journal of the American Statistical Association*, vol. 105, no. 490, pp. 713–726, 2010.

[50] N. R. Draper and H. Smith, *Applied regression analysis*. John Wiley & Sons, 2014, vol. 326.

[51] B. Szalkai, “An implementation of the relational k-means algorithm,” *arXiv preprint arXiv:1304.6899*, 2013.

[52] F. Rosén, “Correlation based clustering of the stockholm stock exchange,” 2006.

[53] R. N. Mantegna, “Hierarchical structure in financial markets,” *The European Physical Journal B-Condensed Matter and Complex Systems*, vol. 11, no. 1, pp. 193–197, 1999.

[54] S. R. Joglekar, "Two-stage stock portfolio construction: Correlation clustering and genetic optimization." in *FLAIRS Conference*, 2014.

[55] D. Arthur and S. Vassilvitskii, "k-means++: The advantages of careful seeding," in *Proceedings of the eighteenth annual ACM-SIAM symposium on Discrete algorithms*. Society for Industrial and Applied Mathematics, 2007, pp. 1027–1035.

[56] Z. Ren, "Portfolio construction using clustering methods," Ph.D. dissertation, Worcester Polytechnic Institute, 2005.

[57] B. Scherer, "Portfolio construction and risk budgeting," 2007.

[58] S. Craighead and B. Klemesrud, "Stock selection based on cluster and outlier analysis," in *Fifteenth International Symposium on Mathematical Theory of Networks and Systems University of Notre Dame*, vol. 42, 2002.

[59] L. Kaufman and P. Rousseeuw, *Clustering by means of medoids*. North-Holland, 1987.

[60] P. De Jong and J. Penzer, "Diagnosing shocks in time series," *Journal of the American Statistical Association*, vol. 93, no. 442, pp. 796–806, 1998.

[61] K. Marvin, "Creating diversified portfolios using cluster analysis," *Princeton University*, 2015.

[62] J. Korzeniewski, "Efficient stock portfolio construction by means of clustering," 2018.

[63] A. Felfernig and A. Kiener, "Knowledge-based interactive selling of financial services with fsadvisor," in *Proceedings of the National Conference on Artificial Intelligence*, vol. 20, no. 3. Menlo Park, CA; Cambridge, MA; London; AAAI Press; MIT Press; 1999, 2005, p. 1475.

[64] C. Musto, G. Semeraro, P. Lops, M. De Gemmis, and G. Lekkas, "Personalized finance advisory through case-based recommender systems and diversification strategies," *Decision Support Systems*, vol. 77, pp. 100–111, 2015.

[65] M. Zanker, "A collaborative constraint-based meta-level recommender," in *Proceedings of the 2008 ACM conference on Recommender systems*. ACM, 2008, pp. 139–146.

[66] S. P. Sahu, A. Nautiyal, and M. Prasad, "Machine learning algorithms for recommender system-a comparative analysis."

[67] P. Thiengburanathum, S. Cang, and H. Yu, “A decision tree based recommendation system for tourists,” in *Automation and Computing (ICAC), 2015 21st International Conference on*. IEEE, 2015, pp. 1–7.

[68] L. Peska, “Using the context of user feedback in recommender systems,” *arXiv preprint arXiv:1612.04978*, 2016.

[69] “A record year for the s&p 500.” [Online]. Available: https://www.businessinsider.nl/us-stock-market-higher-every-month-in-calendar-year-2017-12/?international=true&r=US

Appendix A

Histograms of ANR estimation features

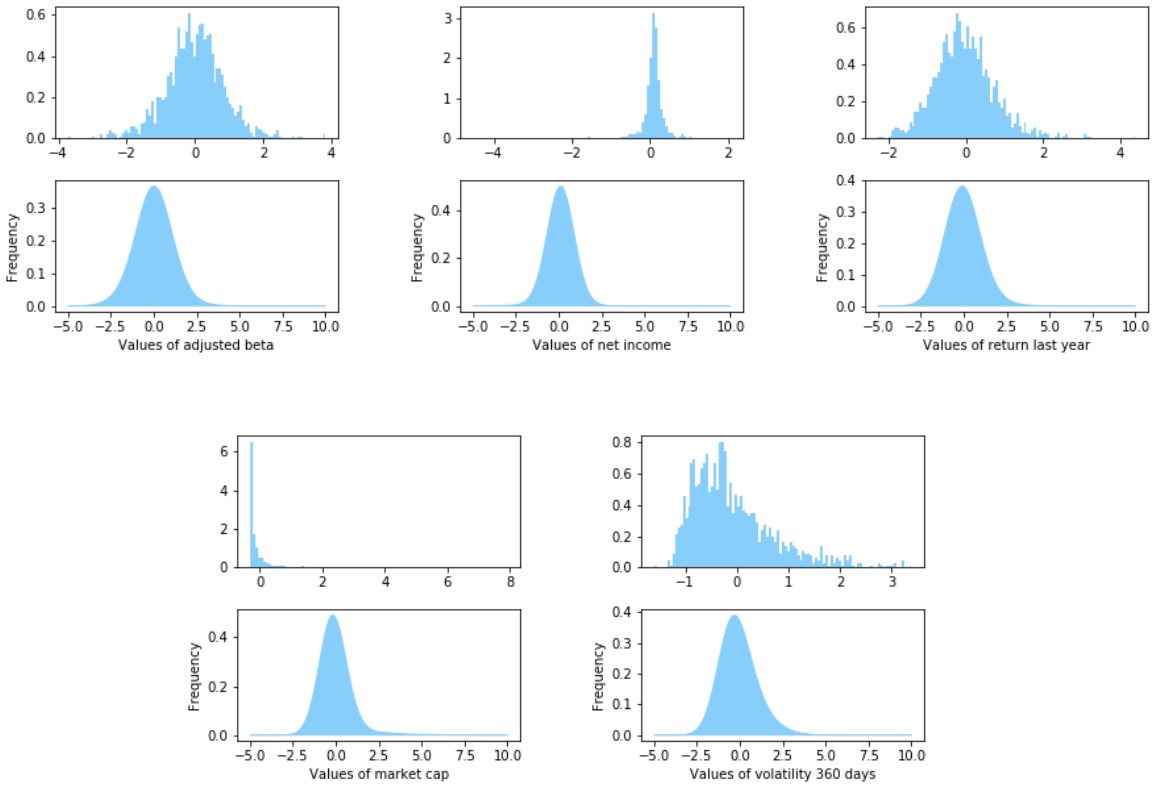


Figure A.1: Histograms of subset 1 of features and Gaussian Kernel distributions

Appendix B

System Evaluation Questionnaire

Q1. How do you find the level of diversification in the created portfolios?
Answer by: linear scale from 1 (insufficient) to 5 (excellent).

Q2. How do you find the performance of the created portfolios?
Answer by: linear scale from 1 (insufficient) to 5 (excellent).

Q3. What is your opinion on the adaptability of the system to different risk profiles??
Answer by: linear scale from 1 (insufficient) to 5 (excellent).

Q4. In your opinion, how much would a system like this speed up the advisory process?
Answer by: linear scale from 1 (not at all) to 5 (very much).

Q4. How interested would you be in further development of the system??
Answer by: linear scale from 1 (not at all) to 5 (very much).