CHARLES UNIVERSITY

FACULTY OF SOCIAL SCIENCES

Institute of Economic Studies

Ondřej Karlíček

Application of Machine Learning in Portfolio Construction

Bachelor thesis

Prague 2021

Author: Ondřej Karlíček
Supervisor: Mgr. Jan Šíla MSc.

Academic Year: 2020/2021

Bibliographic note

KARLÍČEK, Ondřej. *Application of Machine Learning in Portfolio Construction*. Prague 2021. 43 pp. Bachelor thesis (Bc.) Charles University, Faculty of Social Sciences, Institute of Economic Studies. Thesis supervisor Mgr. Jan Šíla MSc.

Abstract

The thesis investigates the application of machine learning in portfolio construction. The analysis was conducted on a dataset consisting of 442 American stocks. Initially, we cluster stocks using Principal Component Analysis and K-means algorithms. Then we select stock from each cluster based on return/risk metrics. Where risk was estimated by Value at Risk, and return was predicted using Random Forest and GARCH models. This leaves us with 11 stocks for every monthly period during 2020. The results indicate that the portfolios constructed from the selected stocks were able to outperform the market benchmark. However, the return predictions were not accurate enough. Thus, the portfolio from selected stock using the 1/N approach achieved better results than the portfolio optimized by the Mean-Variance model.

Keywords

portfolio construction, Mean-Variance model, Principal Component Analysis, K-means, Random Forest, GARCH, Machine Learning

Abstrakt

Práce zkoumá využití strojového učení při tvorbě portfolia. Analýza byla provedena na souboru dat, který se skládá ze 442 amerických akcií. Na začátku jsme provedli klastrování akcií pomocí algoritmů analýzy hlavních komponent a K-means. Poté vybíráme akcie z každého klastru na základě metrik výnosnosti/rizikovosti. Kde riziko bylo odhadnuto pomocí Value at Risk a výnos byl předpovězen pomocí modelů Random Forest a GARCH. Takto nám zůstalo 11 akcií pro každé měsíční období v průběhu roku 2020. Výsledky ukazují, že portfolia sestavená z vybraných akcií dokázala překonat tržní benchmark. Predikce výnosů však nebyly dostatečně přesné. Portfolio z vybraných akcií s využitím přístupu 1/N tedy dosáhlo lepších výsledků než portfolio optimalizované pomocí Mean-Variance modelu.

Klíčová slova

konstrukce portfolia, Mean-Variance model, analýza hlavních komponent, K-means, Random Forest, GARCH, strojové učení

Declaration of Authorship

I hereby proclaim that I wrote my bachelor thesis on my own under the leadership of my supervisor and that the references include all resources and literature I have used.

I grant a permission to reproduce and to distribute copies of this thesis document in whole or in part.

Prague, 4 May 2021

Signature

Acknowledgment

I would like to express my deep gratitude to the supervisor Mgr. Jan Šíla, MSc. for his guidance and the teaching he does at IES as his passion for Econometrics and Data Science was always inspirational. Last but not least, I would like to thank to my family for supporting me during these hard times.

Bachelor's Thesis Proposal

Institute of Economic Studies
Faculty of Social Sciences
Charles University in Prague

Author's name and surname: Ondřej Karlíček

E-mail: 96640370@fsv.cuni.cz

Phone: +420 722 959 355

Supervisor's name: Mgr. Jan Šíla MSc.

Supervisor's email: jan.sila@fsv.cuni.cz

Notes: Please enter the information from the proposal to the Student Information System (SIS) and submit the proposal signed by yourself and by the supervisor to the Academic Director ("garant") of the undergraduate program.

Proposed Topic:

Application of Machine Learning in Portfolio Construction

Preliminary scope of work:

The stock market experienced a considerable downfall in the first quarter of 2020 as the S&P 500 index lost 20%. It has been the worst loss since 2008. Portfolio managers should take into the account not just the estimated return but a possible risk too. Finding the right proportion between risk and expected return is a crucial part of portfolio construction, as noted by Pedersen (2015). This can be very challenging during the market shock. Accurate forecasting of assets' returns is challenging, mainly due to high volatility during a market turmoil.

Machine learning algorithms can be instrumental in finding patterns in data samples. Heaton, Polson and Witte (2017) showed that machine learning methods in financial prediction are promising concerning their predictive performance. This result could provide new methods but also improving the classical ones. Hence we might hypothesize whether or not the standard mean-variance model by Markowitz (1952) could benefit from using machine learning.

In fact, Tadlaoui (2017) already showed, that the Markowitz model can achieve better results with the use of random forest algorithms as it improves the returns forecasts. The thesis focuses on further development of this model and together with using other techniques, as Snow (2020) suggests that both supervised and unsupervised learning can be applied in portfolio optimization. My aim is to demonstrate a comparison of these methods of machine learning between themselves and to the Markowitz model in the standard version during market shock based on their performance.

Contribution

Aim of this thesis is a comparison of methods for returns prediction and risk analyses in portfolio construction and the use of machine learning algorithms for better performing portfolios. One of the contributions of the thesis is in using data from and before COVID crisis to discuss the benefit of using

machine learning methods for portfolio optimization.

Methodology

The work shall use data of US Stocks from the latest years where the impact of the recent COVID crisis will be inspected. Then machine learning methods and the Markowitz model are used for portfolio optimization. These models then will be compared to the S&P 500 index and between themselves using the Sharpe ratio.

Outline

1. Introduction
2. Literature review
3. Methodology
4. Data description
5. Results
6. Conclusion

List of academic literature:

Bibliography

Tadlaoui, Ghali. (2018). Intelligent Portfolio Construction: Machine-Learning enabled Mean-Variance Optimization, MSc thesis, Imperial College London

Snow, Derek. (2020). Machine Learning in Asset Management: Part 2: Portfolio Construction — Weight Optimization, The Journal of Financial Data Science March 2020.

Markowitz, Harry (1952). Portfolio selection, The Journal of Finance, Vol. 7, No. 1., pp. 77-91.

Pedersen, Lasse. (2015). Portfolio Construction and Risk Management, Efficiently Inefficient: How Smart Money Invests and Market Prices Are Determined. Princeton University Press, pp. 54-62.

Heaton, J. B., N. G. Polson, and Witte. (2017). Deep Learning for Finance: Deep Portfolios, Applied Stochastic Models in Business and Industry

Contents

List of Figures

List of Tables

1 Introduction

Portfolio construction consists of many tasks, from examining the market to building beliefs about future performance. Following with selection of the assets into the portfolio and lastly weighting each asset in the portfolio to diversify and minimize the risk for the wanted return.

The classic model for diversification was proposed by Markowitz (1952). The model performs well in the sample. But due to errors in estimations, the model is not constantly outperforming the naive 1/N approach (DeMiguel *et al.*, 2009).

Errors in estimation can be lowered by using Machine Learning techniques. Tadlaoui (2017) showed significant improvement in returns from the portfolio created by the Markowitz model extended with use of Random Forest and GARCH model for returns prediction. It is important to note that the research used a universe consisting of 8 stocks that were preselected without any specific method. The preselection without any reasoning is present in many studies, as noted by Fulga & Dedu (2012), and they presented the usage of clustering and Value at Risk in stock selection.

We want to follow it as with clustering, the investor can obtain groups of similar stocks, and by picking the stocks from different groups, can achieve a diversified selection of the stocks. We extended it by adding a prediction of returns.

In the thesis, we conducted analysis on a dataset consisting of 442 American stocks. Firstly we implement PCA and K-means algorithms on our dataset to cluster our stock universe. Following by calculating Value at Risk, which we use for the first selection of stocks. Then we predict the future direction of the asset using Random Forest, which Ballings *et al.* (2015) recommends as the best performing model for predicting the stock direction. Then the magnitude of expected return is estimated using volatility derived from the GARCH model. Therefore, from each cluster, we select one stock based on expected return and Value at Risk. The selected stocks are used to

construct the portfolio using the naive 1/n approach and by the Markowitz model. The resulting portfolios are compared to the market benchmark. The rebalancing will be performed 12-times during 2020. Thus, we will be testing the model during 2020 when the market experienced very fluctuating times because of the COVID pandemics.

The thesis attempts to demonstrate the combinations of the machine learning techniques during the whole portfolio construction process as they are commonly examined separately. The contribution we see in an interesting combination of models, whereby clustering we want to obtain different groups of stocks. By picking stocks from each cluster using return prediction, we want to gain a well-diversified portfolio. The hypothesis is that we will be able to pick stocks from the clustered stock universe, and the portfolio from selected stocks will be able to outperform the market in terms of returns and volatility. Another hypothesis is that the portfolio from selected stocks will perform better with mean-variance optimization than with the 1/N approach.

The thesis is divided into six sections. The second section firstly briefly presents the beginning of the modern portfolio theory and is followed by current literature about the usage of machine learning in the portfolio theory. The third section introduces methods and models used in the thesis. The fourth section provides a description of the data and the manipulation with them. The fifth section presents the empirical results, and everything is summarized in the last section.

2 Literature Review

As the founder of Modern portfolio theory is widely considered Markowitz (1952), his principal claim is that investors should diversify their portfolios. The intercorrelated portfolio is more prone to shock than the well-diversified portfolio. Markowitz derived a theory for optimization of a portfolio called mean-variance optimization. It suggests that expected returns and risk (represented as the variance of the portfolio) should be taken together. The investor should maximize expected returns for a given risk or minimize risk for given expected returns.

Another extension of Modern portfolio theory is the capital asset pricing model (CAPM), independently derived by Treynor (1962), Sharpe (1964), Lintner (1965), and Mossin (1966). CAPM decomposed risk into two components, the systematic and the idiosyncratic. Investors should take into account both types of risk. Nevertheless, diversification can reduce only the idiosyncratic risk.

Although these models are theoretically important, their practical performance is disputable. DeMiguel *et al.* (2009) challenged them in an empirical study, where mean-variance, CAPM, and another 11 modern versions of them were tested using 7 different datasets. They found that these models do not consistently outperform naive 1/N strategy, where N is a number of stocks and each stock in the portfolio is weighted by 1/N. The authors do not imply that naive 1/N portfolio is better than portfolio optimization models as the in-sample mean-variance model outperforms the 1/N benchmark. They claim that gains from optimization of the portfolio are eroded by errors in estimating means and covariances. Thus, more energy should be put into improving asset returns estimation.

Return predictions is where machine learning techniques could help. A variety of types of machine learning can be used, from supervised and unsupervised learning methods to even reinforcement learning (Snow, 2020). The thesis will focus on supervised and unsupervised learning techniques.

Supervised learning algorithms firstly interact with the dataset, where every random vector x is connected to a label or target, which we denote as y (Goodfellow *et al.*, 2016). This phase is called training, we view it as the target vector or value y is given by the instructor. That is why these techniques are called supervised learning. Then comes predicting phase when the algorithm returns y from given x using $p(y \mid x)$ estimated during the first phase. Supervised learning can be divided into two main groups, classification and regression. Classification is about predicting labels, on the other hand, regression is about predicting a quantity.

As the name suggests, unsupervised learning is missing the target value y from the instructor in the training part. The unsupervised learning algorithms interact only with x, from which it is learning the properties of the structure of the dataset (Goodfellow *et al.*, 2016). The most common tasks are estimating the probability distribution which generated data, denoising, dimensionality reduction, or clustering, which divides the dataset into clusters of similar variables.

The usefulness of machine learning in portfolio optimization was demonstrated by Tadlaoui (2017) on the stock universe consisting 8 stocks. Firstly, the Random Forest algorithm was trained on historical data and then used to predict the future direction of the price of the stock. The size of the change of price was estimated by volatility which was predicted using the GARCH model. These two models give predictions of the future return of the stock. Prediction of returns was used to enhance the mean-variance model, this was compared to the mean-variance model using historical returns. The portfolio constructed by extended version mean-variance was able to outperform the classic mean-variance by more than 20% in absolute returns.

The predictive power of the Random Forest method in portfolio construction was also demonstrated by Kaczmarek & Perez (2021). The study is working with companies listed in the S&P500 index. The authors of the study use Random Forest for predicting excess returns. Next, n stocks with the highest prediction values were chosen, where $n = 25, 50, 75, ..., 250$.

These chosen stocks were first diversified by the naive 1/N rule. Secondly, a mean-variance optimizer was used. Thirdly and finally, hierarchical risk parity (HRP) optimizer was implemented. The portfolios were rebalanced monthly during 10 year period, from 01/01/2010 to 31/12/2019. The chosen n stock naively diversified were able to outperform the market benchmark, whole equal-weighted S&P500 index. Finally, a comparison of 1/N with mean-variance and HRP appears to be better for both optimization methods as they were able to outperform naive portfolios from n chosen stocks and market benchmark. The difference in performance between HRP and mean-variance was not significant and was dependent on n.

Chen *et al.* (2021) implemented the eXtreme Gradient Boosting algorithm (XGBoost) for stock price prediction in combination with the firefly algorithm, which serves as an optimizer of the hyperparameters in XGBoost. For practical analysis, 24 stocks were randomly selected from the Shanghai Stock Exchange 50 index. Stock prediction is followed by the mean-variance model. The XGBoost was also compared with other techniques as Long Short-Term Memory (LSTM) Neural Network, Support Vector Regression (SVR), or just random selection of stocks, these methods were combined with the mean-variance model or 1/N diversification, 1/N diversification with machine learning methods was done that only assets with the best prediction were selected and then equally weighted. All portfolios enhanced with XGBoost, LSTM, SVR outperformed simple 1/N or random + 1/N portfolios in terms of Sharpe ratio. The paper also showed that machine learning + mean-variance was able to get better results than machine learning + 1/N. This indicates that the mean-variance model has an essential role in portfolio construction.

Another popular supervised learning technique is the Support Vector Machine (SVM). Yu *et al.* (2014) established a stock selection model using SVM. Researchers also used Principal Component Analysis (PCA) for reducing the dimensionality of the financial data. PCA-SVM model was used on a dataset of companies in A-share of Shanghai Stock Exchange. The model

was trained to determine high return stocks, which were used to construct the equal-weighted portfolio. This portfolio was able to accumulate higher returns than the A-share index of the Shanghai Stock Exchange.

An extensive study of methods for forecasting stock price direction was realized by Ballings *et al.* (2015). Three ensemble methods (Random Forest, AdaBoost, and Kernel Factory) and four single classifiers (Neural Networks, Logistic Regression, SVM, and K-Nearest Neighbor) were compared by cross-validation and AUC as a measure of performance, AUC is the area under probability curve that plots the true-positive rate against the false-positive rate. Analysis was performed on data from 5767 European companies from a variety of industries. For each model was performed five times twofold cross-validation, and the median of AUC was taken as a representation of the model performance. Based on this evidence was Random Forest declared as the best performing model for predicting the stock price direction, followed by SVM, Kernel Factory, and AdaBoost.

As already stated, PCA is mostly used for dimensionality reduction, but it can be used for portfolio diversification as well, this is demonstrated by Pasini (2017). PCA algorithm works that it changes the dataset into the Principal Components, where every Principal Component is a linear combination of the dataset. Principal Components are calculated in the way that the first one captures the most variance in data, the second one captures the second most variance in data, and so on. The paper demonstrates the application of PCA on three subgroups of stock of the American Index DJI. Two groups are homogeneous, but the third is more heterogeneous. PCA is then applied to the correlation matrix of stocks, subsequently, the first two Principal Components are analyzed. The paper shows that portfolios constructed based on the first or second Principal Component follow trends of the market (1/N portfolio). Coefficients of the first Principal Component were also always positive; thus, this component can be seen as a market component. The author of the paper suggests that PCA could be good for diversifying risk, but the drawback is that it cannot say how many stocks

the investor should retain in the portfolio.

The usage of unsupervised machine learning in stock selection was examined by Fulga & Dedu (2012). The research was done on the dataset of 48 financial assets from Bucharest Stock Exchange with seven financial indices, e.g., Price-to-book value, Earnings per share. First, a reduction of the number of variables was achieved with PCA. This reduced number of variables was used for clustering to build classes of similar assets. That was accomplished by the agglomerative hierarchical clustering technique. Agglomerative clustering is a bottom-up approach, which means that every observation is initially in its own cluster, and the algorithm is gradually merging the clusters until the required number of clusters is achieved. In the study, 10 clusters were chosen by authors. Following that, Value at Risk (VaR) was estimated for each asset. Afterward, the asset with minimal VaR was selected from each cluster. Using a mean-risk optimization portfolio was constructed from the selected stocks. The author state that this approach improves the optimization process as only already diversified assets with minimal risk are considered.

Lemieux *et al.* (2014) provided a comparison of clustering algorithms. They applied the following clustering methods on the CRSP US Stock database: K-means, K-medoids, and hierarchical clustering. Each method was used to construct 7 clusters. The paper then demonstrated differences between methods by plotting the variations between techniques. The main claim of the authors is that there are inconsistencies between outputs of different clustering methods.

3 Methodology

The methodology is divided into four sections. The first section describes unsupervised techniques which we used for clustering, more precisely PCA and K-means. The second section will introduce Value at Risk as one of the measures which the thesis use for choosing the best stock from each cluster. In the third section, we are going to present methods for stock return prediction, more specifically we are going to describe Random Forest and GARCH model. The fourth and last sections will introduce the mean-variance optimization problem.

3.1 Clustering Techniques

3.1.1 Principal Components Analysis (PCA)

PCA was developed by Pearson (1901) as the method which takes multivariate data and fits a linear subspace to it by minimizing the chi distances, in simple words how many standard deviations is the given point from the distribution of the data.

PCA is an unsupervised learning algorithm that learns the representation of data in order to present new low-dimensional representations of data by compressing as much information about data in a lower dimension (Goodfellow *et al.*, 2016). The elements of the new representation of data also do not have a linear correlation with each other. PCA can be defined in two ways (Bishop, 2006). Firstly, as maximum variance formulation using orthogonal projection of the data onto a lower-dimensional linear space. Secondly, as the linear projection that minimizes the mean squared distance between the original data points and projection, this approach is called minimization of the average projection cost. Both definitions lead to the same algorithm.

We are going to present a maximum variance formulation. Let us have a set of observations $\{x_n\}$, where $n = 1, ..., N$ and every vector x_n is a D-

dimensional vector.

The sample mean is derived as:

$$\bar{x} = \frac{1}{N} \sum_{n=1}^{N} x_n$$

Using sample mean, we derived sample covariance matrix:

$$S = \frac{1}{N} \sum_{n=1} N(x_n - \bar{x})(x_n - \bar{x})^T$$

Then the goal of PCA is to project every x_n into the vector p_n, where the elements of the vector p_n are defined as:

$$p_{i,n} = w_i^T x_n$$

where $i = 1, ..., D$, but as PCA is used for dimension reduction, we are usually interested in $i = 1, ..., d, d < D$.

Elements of the vectors p are called principal components, we denote vectors of principal components as $(P_1, P_2, ..., P_d)$. The PCA algorithm calculates weights w_i to satisfies two conditions. Firstly, principal components must be orthogonal, that means $E(P_i P_j) = 0, i \neq j$. Secondly, the first principal component explains the largest possible percentage of variability of the original dataset, then the second principal component explains the largest percentage of variability that has not been explained by the first component and so on.

This is achieved by setting w_1 to the eigenvector with the largest eigenvalue of the covariance matrix S, then setting w_2 to the eigenvector with the second largest eigenvalue and so forth (Bishop, 2006). For details, the thesis refers to Bishop (2006) or Goodfellow *et al.* (2016).

We present an example of PCA dimensionality reduction in figure 1. The reader can notice that originally two-dimensionality data were reduced to a single dimension and plotted as in one line on top of the original data. The plot was obtained in Python using an example by VanderPlas (2016).

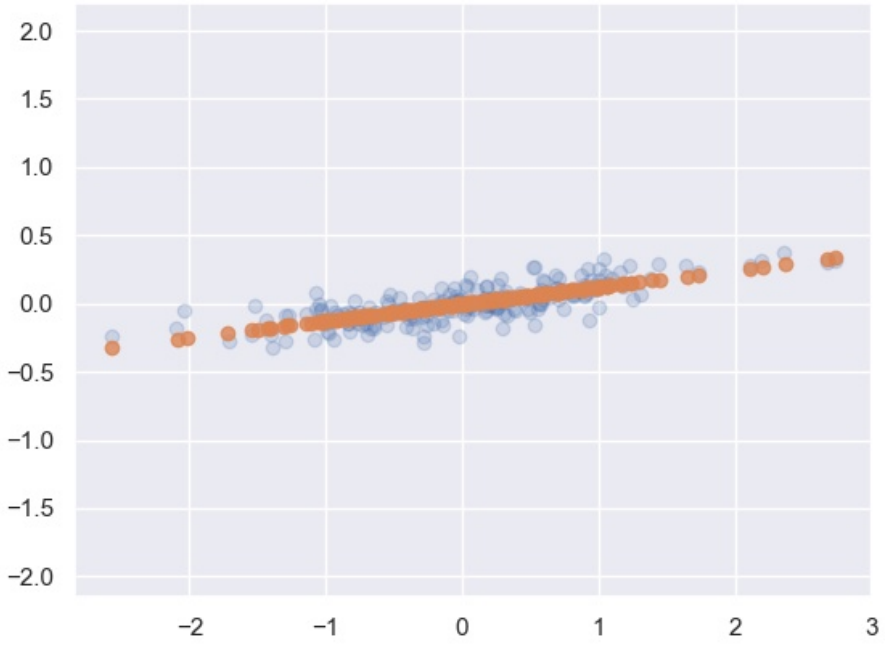


Figure 1: Example of Dimensionality Reduction using PCA

3.1.2 K-means

K-means is an unsupervised classification technique that splits the dataset into a given number of clusters based on the distance between points and the cluster centers. Note that the number of clusters is given by the researcher. The idea of this algorithm was firstly introduced by Steinhaus (1957). Lloyd (1982) is considered to be the first who derived algorithmic solution, he managed to do it in 1957 but note that the paper was not published immediately but in 1982.

Before deriving the K-means algorithm, we need to define squared Euclidean distance which is used as a distance measure in K-means. For two vectors x, y of dimension D, the squared Euclidean distance is defined by

the following equation:

$$\|x - y\|^2 = \sum_{n=1}^{D} (x_n - y_n)^2$$

K-means is the expectation-maximization algorithm as each iteration takes two steps (Bishop, 2006), (Goodfellow *et al.*, 2016). Let us have a set of observations $\{x_n\}$, where $n = 1, ..., N$, and every vector x_n is a D-dimensional vector. We want to split the dataset into the K clusters, where $K < N$ is given.

K-means algorithm approach this by first initializing a set of D-dimensional vectors $\{\mu_k\}, k = 1, ..., K$. We denote μ_k as a center of the kth cluster. These centers can be given or random. Then we define $r_{n,k}$, which is equal to 1 if observation x_n is in the kth cluster, and equal to 0 otherwise. Remark, every observation can be only in one cluster. By this setup, we can define the function J of the K-means optimization problem:

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

We can minimize J by $r_{n,k}$ or μ_k. As we already stated, the algorithm has two main steps. Firstly, we will minimize J with respect to $r_{n,k}$ while holding μ_k fixed. This step is called 'Expectation' (Bishop, 2006). As we can see directly from the equation, in order to minimize J with respect to $r_{n,k}$ we just set $r_{n,k}$ equal to 1 for k with lowest $\|x_n - \mu_k\|^2$, and 0 otherwise. In other words, we assign x_n to the cluster with the nearest center.

The second step, called 'Maximization' (Bishop, 2006), is to minimize J with respect to μ_k. This is a quadratic problem, the first-order condition is following:

$$2 \sum_{n=1}^{N} r_{n,k}(x_n - \mu_k) = 0$$

from that, we easily derive that minimum is for:

$$\mu_k = \frac{\sum_{n=1}^{N} r_{n,k} x_n}{\sum_{n=1}^{N} r_{n,k}}$$

Note that this is the mean of all x_n that belongs to the cluster k. Therefore, the algorithm in every iteration firstly assigns points to their nearest cluster center and then updates the cluster center based on the mean of the points of that cluster. This is repeated until the algorithm converges. The algorithm always converges, but note that it does not have to be a global minimum (MacQueen *et al.*, 1967).

As stated above, the number of clusters is given to the algorithm. Thus, there is the problem of choosing the right number of clusters. The rule of thumb is running the algorithm multiple times for different numbers of clusters and choosing the number for which the value of converged J will stop significantly decreasing (Mirkin, 2011). The J is commonly called Within-Cluster-Sum of Squared Errors.

In figure 2, we present an example of the K-means algorithm used on two-dimensional data for 4 clusters. The plot was generated in Python following an example by VanderPlas (2016).

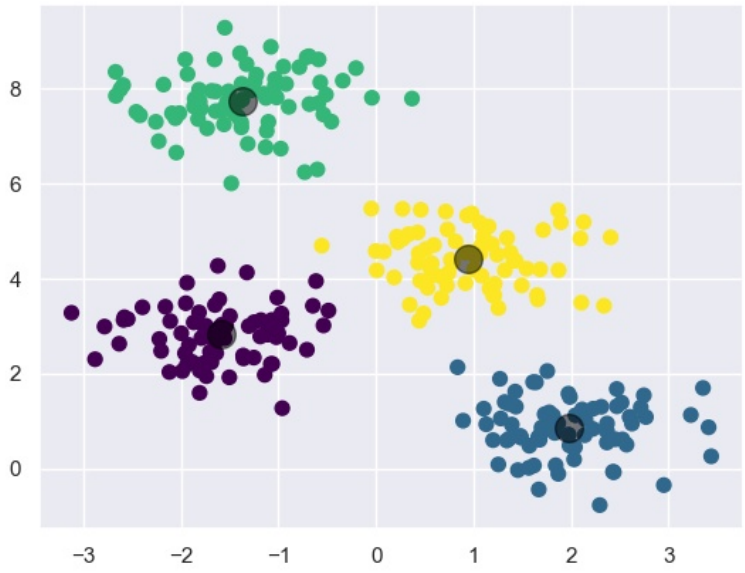


Figure 2: Example of K-means

3.2 Value at Risk (VaR)

Initially, we should say what risk is. Jorion (2006) defines risk as the volatility of unexpected outcomes. Risk is usually seen as the probability of loss. Measurements of the risk are important in the finance world as the economic agents are taking a risk for their actions, and they should take it into account. The future is not certain, and investors should make a decision based on the level of risk they are able to accept (Markowitz, 1952).

Value at Risk (VaR) is a statistical risk measure of potential losses (Jorion, 2006). It was developed at J.P.Morgan in the 1980s. Researchers of this investment bank were looking for simple and understandable risk measures, and so the managers of the bank could take risk into consideration when making decisions. The methodology of VaR was published by J.P.Morgan with their simplified version of the bank's internal model in 1994. VaR was quickly adopted by other financial institutions and also become standardly used in banking regulations.

Hull (2015) is introducing VaR as $100 \cdot \alpha$ percent certainty that in time T we will not lose more than V dollars. Where for the given stock or the portfolio V denotes its VaR with respect to two parameters, confidence level α and time horizon T.

To express that mathematically, let us have profit-loss random variable X, for better intuition, we define flipped profit-loss variable $Y = -X$. Thus, profits are negative and losses are positive. Then we define VaR at confidence level $\alpha \in [0, 1]$ with the following formula:

$$VaR_\alpha(X) = inf\{y : P(Y \leq y) \geq \alpha\}$$

As noted by Jorion (2006), there are 3 main ways of computing VaR in practice. The parametrical approach assumes that the distribution of the returns is from the parametric family, usually normal distribution. The calculation then involves estimating the parameters from the data, in the case of normal distribution, it is the mean and standard deviation. These para-

meters are then used for calculating the quantile of the assumed distribution which gives us VaR, based on the confidence level we want.

Another method is using the concept of the Monte Carlo simulations. The Monte Carlo method is to simulate a random process multiple times, which recreates a new sample of observations. This needs an assumption of the distribution of the data as in the parametrical method, or the distribution of data can be estimated with a different technique. After that, the profits and losses are simulated for a given number of simulations. The number of simulations is commonly very high, which makes the method very computationally intensive. VaR is then calculated as a quantile of the simulated data based on the wanted confidence level.

The last method, the one used in the thesis, is the historical approach. This method does not need any assumption about the distribution of the data. It is simple derived as quantile from the historical data. For VaR at the 0.95 confidence level, we can imagine it as that we sort profit-loss historical observations and $VaR_{0.95}$ is a cut-off value that only 5 percent of losses are above this value.

3.3 Techniques for stock return prediction

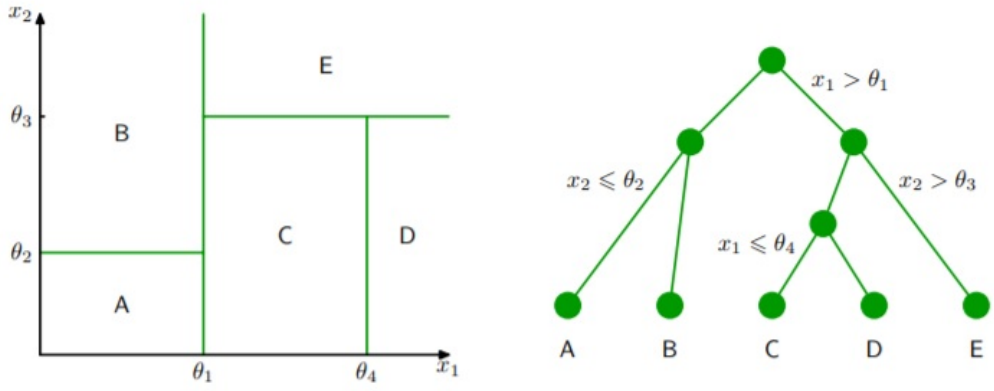
In this part, we are going to describe methods for predicting future returns. This involves Random Forest as the classifier to predict the direction of the stock price and the GARCH model for modeling the magnitude of this change. Before describing the Random Forest, we must introduce the Decision Tree model as an important part of the Random Forest.

3.3.1 Decision Tree

Decision Trees, also called Classification and Regression Trees, is a simple and intuitive tree-like model (Breiman *et al.*, 1984), (Bishop, 2006). Building the tree is done by splitting the nodes into two daughter nodes. The splits are determined by the splitting criterion. The splitting will create non-

overlapping smaller regions in the predictor space. The example of that we can see in figure 3, where is Decision Tree demonstrated on two-dimensional space. On the left side of the figure, we can notice that the space was splitted into smaller regions, and on the right side is an example of the simple tree structure belonging to that space.

Figure 3: Example of Decision Tree



Source: Bishop (2006)

As we can notice that the tree is made in a recursive way. Firstly space is divided by θ_1 then the two new regions are divided independently by different criteria. The first node is called a root node, nodes "in the middle" which are split into the new two nodes are called internal nodes, and the final node is called a leaf node.

The splitting stops (the leaf node is created) when the node contains only one observation or some of the stopping parameters are satisfied. The most common stopping parameters are:

- **Maximal depth** controls the size of the tree, it represents the maximum number of nodes on the way to the leaf node.
- **Minimal samples split** determines the minimal number of observations in the node for continuing in the splitting.
- **Minimal samples leaf** regulates for the minimal number of observations in leaf node (final node).

As noted by Bishop (2006), the determination of the optimal structure of the Decision Tree is computationally very demanding, and for most cases, it is computationally infeasible. Thus, every step is determined by choosing the decision with the best immediate gain in performance, this is tackled by using a greedy algorithm. But with this approach, the convergence to the optimal Decision Tree is not assured, but we can reduce this issue by using Random Forest described in the section 3.3.2.

The performance measure of the split for regression tasks is usually the sum of squares error. In the case of the classification problem, the measure is commonly Giny index or cross-entropy (Bishop, 2006). We are going to define only Gini index, as it is the one used in the thesis:

$$Gini\_index = \sum_{k=1}^{K} p_k(1 - p_k)$$

where K is a number of classes and p_k indicates the proportion of data points in the given subset which belongs to the class k, where $k = 1, ..., K$. In the thesis, we are working with a binary classification problem (K = 2), price direction prediction. Thus, we can further simplify this equation in the following way:

$$Gini\_index = 2p(1 - p)$$

where p denotes the proportion of data points in a given subset that belongs to the first class. It is easy to derive that maximum for the Gini index is at $p = 0.5$ and the index is equal to zero when $p = 0$ or $p = 1$.

The decision tree algorithm finds the best split that for the two new nodes calculates Gini indices. Then the Total Gini index is derived as the weighted mean of these two Gini indices, where weights indicate the proportional size of the nodes. The goal of the split is to achieve the lowest Total Gini index.

3.3.2 Random Forest

As stated above, the algorithm used in the Decision Tree does not have to converge to the optimal model. This instability can be reduced by using en-

semble technique, bagging or extended version of these two, Random Forest (Hastie *et al.*, 2016). Before describing the Random Forest, we are going to briefly introduce ensemble and bagging:

- **Ensemble learning** is based on using a high number of models (Decision Trees in our case), every model is trained, and the prediction is derived from the voting of the models. It is important that predictions between models do not have a very high correlation. The variation between models can reduce the mistakes of a single model. An ensemble model with perfect correlation would act similarly to the single model.
- **Bootstrap aggregating (Bagging)** is part of the ensembling techniques. The important part of bagging is drawing random samples with replacement from the training data. Thus, every model is trained on a different subset of the original data. The Decision Trees are sensitive to even a slight change in the data, and this technique reduces the final variance of the predictions to make them more robust to slight changes. The averaging multiple Decision Trees trained on a different subset of the data give us more stable outcomes (Hastie *et al.*, 2016).

Random Forest is extending these methods by random choice of features during every split. Thus, the procedure is as follows; firstly drawing random subset with replacement as in Bagging, then the Decision Tree is trained on this subset. However, during each split, random m features is selected, for classification the default setting of m is the square root of the number of features.

The idea is that when some feature is very important, which means that the feature is responsible for a very big part of all splits. Then with the Random Forest approach, this feature's presence is reduced, and the importance of others is increased. This is the main difference to Bagging. Although a single tree's predictive power is lower, this decrease is averaged in all trees and is beneficial for the Random Forest method (Hastie *et al.*,

2016).

Performance of the Random Forest will be evaluated based on these metrics:

- **Accuracy** - a fraction of right predictions

$$\frac{TP + TN}{TP + TN + FP + FN}$$

- **Precision** - a fraction of correct positive predictions

$$\frac{TP}{TP + FP}$$

- **Recall** - a fraction of correctly predicted real positive

$$\frac{TP}{TP + FN}$$

  where TP is true positive, TN true negative, FP false positive, and FN false negative

3.3.3 GARCH Model

The generalized autoregressive conditional heteroskedasticity model (GARCH) is a technique for modeling volatility, assuming that variance follows a mean-reverting process. The model was proposed by Bollerslev (1986) as an extension of the ARCH model. The ARCH model is describing volatility using error terms from previous periods. GARCH model is extending ARCH with the use of past volatility as well. The difference is that ARCH is assuming an autoregressive process, but GARCH is assuming a more general autoregressive moving average process.

We are going to present only GARCH(1,1), which is the most simple version of GARCH. GARCH(1,1) is convenient for financial data as Miah *et al.* (2016) showed that GARCH(1,1) achieved a better result in modeling volatility than the more complicated methods.

The "(1,1)" in GARCH(1,1) specify that the current variance is dependent on the most recent observation. GARCH(1,1) is defined in the following

way:

$$R_t = \mu + \epsilon_t$$

$$\epsilon_t = \sigma_t * \zeta$$

$$\sigma_t^2 = \omega + \alpha\epsilon_{t-1}^2 + \beta\sigma_{t-1}^2$$

where R_t denotes asset return with mean μ and error term ϵ_t. ϵ_t is related to the volatility and the white noise ζ, white noise is a random variable with zero mean and no correlation between the values. First-term in the equation ω denotes the weight of long-run average variance rate V_L, $\omega = \gamma V_L$. Interpretation of the next parameters is following, α represents the persistence of the short-term shocks, and $(\alpha + \beta)$ represents the persistence of the long-term shock (Campbell *et al.*, 1996). For the process to be stable, we require $(\alpha + \beta) < 1$ (Hull, 2015). The difference between general GARCH(p,q) and GARCH(1,1) is that the general version of the model for estimating σ_t^2 is using the most recent p observations of ϵ^2 and the most recent q observations of σ^2.

An essential part of the functionality of the model is assuming that ϵ_t are following normal distribution, although the model can be adjusted to t-distribution. As noted by Hansen & Lunde (2005), the returns of stock are commonly more heavy-tailed than the normal distribution. Thus, we are using GARCH(1,1) with the assumption that ϵ_t are following t-distribution.

Another assumption is that ϵ_t needs to be uncorrelated as they are related to the white noise.

3.4 Mean-Variance Optimization (MVO)

The final step in our portfolio construction is the optimization of weights of the given assets. We are going to introduce Mean-Variance Optimization (MVO) by Markowitz (1952). Harry Markowitz with his work is widely considered as the founder of the Modern Portfolio Theory.

Markowitz's principal claim is that investors should not only focus on maximizing expect returns of the portfolio, but they should take the risk into

account as well. In the MVO, the risk is represented as the portfolio variance. This risk-reward approach is a mathematical technique for diversification in investing. It follows the idea that by investing in different types of assets, the investor is taking lower risk than by investing in only one type of asset.

In defining the MVO, we follow Dupačová *et al.* (2006), for N assets, we define the weight vector as $w = (w_1, ..., w_N)^T$, where w_i denotes how big part of the initial wealth was invested into the ith asset. It is clear that $\sum_{i=1}^{N} w_i = 1$. In the thesis, we also add another constraint that $w_i \in [0, 0.2]$, for every i. Thus, we forbade short-selling and, to avoid corner solution, we allow for every asset's weight to be maximally set to 20 percent. This will ensure diversification across assets.

The expected return of the ith asset we denote as r_i. Therefore, the expected return of the portfolio is $r_p = r^T w$, where r is the vector of r_i. Then we define covariance matrix $V = (\sigma_{i,j})$, where $\sigma_{i,j} = cov(r_i, r_j)$; $i, j = 1, ..., N$. Note that for $i = j$, it is the variance of returns for r_i. The variance of the portfolio (risk) is given by $\sigma_p^2 = w^T V w$. Remark, Dupačová *et al.* (2006) are using the square root of σ_p^2 (standard deviation), but in this, we are following the setting from *PyPortfolioOpt*, Python library for portfolio optimization, where the variance is used.

This leaves us with the following optimization problem:

$$r_p = r^T w$$

$$\sigma_p^2 = w^T V w$$

$$\sum_{i=1}^{N} w_i = 1$$

$$w_i \in [0, 0.2], i = 1, ..., N$$

There are 4 main ways how to solve these equations, depending on the goal of the investor. Remark, we will not state the constraints for weights anymore as they are the same for all methods.

- **Maximize return for a given risk**

  Let us consider that the maximal risk which the investor is able to take

is σ^2_{MAX}. Then the setting is following:

Maximize

$$r^T w$$

Constraint

$$w^T V w \leq \sigma^2_{MAX}$$

- **Minimize risk for a given return**

  Let us consider that the investor wants at least a return r_{MIN} from the portfolio. Then he is minimizing the risk for this return:

  Minimize

  $$w^T V w$$

  Constraint

  $$r^T w \geq r_{MIN}$$

- **Global minimum-variance portfolio**

  This method ignores the expected returns and only focuses on risk minimization. Therefore:

  Minimize

  $$w^T V w$$

- **Maximize Sharpe ratio**

  Sharpe ratio is the return-risk ratio derived by Sharpe (1966), and is given by the equation:

  $$S = \frac{r_p - r_f}{\sqrt{\sigma_p^2}}$$

  where r_f denotes the return of a risk-free asset, the default value of r_f in *PyPortfolioOpt* is equal to 0.02. However, the thesis omit the problem of choice between risk-free and risky assets. The thesis focus on optimization between risky assets, and our main goal is a demonstration of machine learning during the risky side of portfolio construction. Thus, we will set risk free rate to 0, which simplify the maximization of the Sharpe ratio into the following equation:

  Maximize

  $$\frac{r^T w}{\sqrt{w^T V w}}$$

For details, the thesis refers to Dupačová *et al.* (2006). The thesis will optimize the portfolio by maximizing the Sharpe ratio as it is related to the way we are selecting stocks from clusters, where we are using return/risk metric.

To summarize this section, we firstly described unsupervised learning techniques, PCA and K-means. Then we introduced VaR. Followed by a description of supervised learning techniques, Random Forest and GARCH. Lastly, we presented MVO.

4 Data Description and Processing

For the analysis, we wanted to use companies from the Standard and Poors 500 (S&P500) index. However, the list of companies in S&P500 is regularly changing. This brings problems with choosing the dataset for clustering companies. The goal is rebalancing the portfolio during the year 2020; thus, the best data for clustering would be from the end of the year 2019. Nevertheless, we could not find the dataset from this date as most of the data on the internet is updated to the current date.

Therefore, the chosen dataset for clustering (more described in the section 4.1) comes from the second half of 2018, one year away from the ideal dataset from the end of 2019. The drawback is that we are going to use the outdated list of companies in the S&P500 index. When we compared this outdated version to the current index (start of the 2021), we found out that 101 companies are not in the current index and were replaced with different companies.

Unfortunately, we have to filter out 63 companies because their time series for returns predictions were not long enough. We are using data from the end of 2016 to the end of 2020, where 3 years of data are for training. Out of 63 companies, 59 companies are not currently in the S&P500 index, and the data are mainly missing because the companies were merged or acquired by another company. The remaining 4 companies were too young to provide enough data for prediction. This leaves us with 442 assets. Remark, there are 505 assets in the clustering dataset because it includes 5 companies with dual-class stock. Because of the filtering, we have survival bias in the dataset, but we assume that the bias is lower than if we use the current list of companies in the index.

In the following sections, we are going to present our datasets and the operations with them. Firstly, we will introduce the dataset for clustering, and then we will move to the dataset for return predictions.

4.1 Data for Clustering

The dataset was obtained from *DataHub*[1]. It is a cross-sectional dataset of companies in the S&P500 from the second half of 2018. The dataset provides 10 features for our analysis. The features and their description is following:

- **Dividend Yield**

  It is expressed in percents, and it represents the amount of dividend paid out relative to the price of the asset. It is calculated by dividing dividend by the price of the asset.

- **Earnings per Share (EPS)**

  Demonstrates profitability of the company. It is simply earnings (profits) divided by the number of available shares.

- **Price to Earnings Ratio (P/E Ratio)**

  Closely related to the EPS, it is derived as the ratio of the price of share to the EPS.

- **EBITDA**

  Proxy of the earning potential. The acronym corresponds to the earnings before interest, taxes, depreciation, and amortization.

- **Book Value**

  Important number in the balance sheet of the company derived as total value of an asset minus depreciation and other expenses.

- **Price to Book Ratio (P/B Ratio)**

  The measure of comparison of the price and book value calculated as the ratio of the market price to the book value.

- **Price to Sales Ratio (P/S Ratio)**

  This ratio shows a comparison between price and the sales of the company, derived as market price divided by the company's revenue.

[1] `https://datahub.io/JohnSnowLabs/standard-and-poors-500-companies-list-with-financial-information`

- **Market Capitalisation**

  Demonstrate the value of the company held by shareholders. It is calculated by multiplicating the price of the company's share with the number of shares held by shareholders, also denoted as shares outstanding.

- **52-Week Low**

  Denotes the lowest price of the stock during the last 52 weeks.

- **52-Week High**

  Opposite of the 52-Week Low. The highest price of the stock for the last 52 weeks.

In the Dividend Yield feature, there were altogether 69 missing values, these missing values denoted that the company is not paying out dividends. Thus, missing values were replaced with 0.

Another feature with the most missing values was the P/E Ratio with 46 missing values, and the next was EBITDA with 28 and the third P/B Ratio with 18. The remaining features had 2 missing values at maximum. To not lose the information, we decided to replace missing values in these features with the mean of the feature.

The last operation with the dataset is norm-standardization because the dataset will be used to fit PCA. As Yu *et al.* (2014) suggest, it is an essential step in using PCA because the algorithm is sensitive to the magnitude of the data. Norm-standardization for variable X is given by:

$$Z = \frac{X - mean(X)}{std(X)}$$

4.2 Data for Return Predictions

Time series for given assets were obtained from *Yahoo Finance* using Python package *pandas_datareader*. For each asset, we loaded adjusted close price and daily volume. Adjusted close price denotes the closing with adjustment of dividends and splits. Volume represents the number of shares traded during the day for a given asset.

Firstly, we define d-days[2] returns from adjusted close price, which are calculated by the following formula:

$$R_{d,t} = \frac{P_t - P_{t-d}}{P_{t-d}}$$

where P_t denotes adjusted closing price at time t. We calculated 21-days returns ($d = 21$) and daily-returns ($d = 1$).

4.2.1 Features for Random Forest

For the rest of the section, we will be deriving features for predicting stock direction by Random Forest. Firstly, we need to denoise raw data of adjusted closing prices. Following Basak *et al.* (2019), we will use exponential smoothing given by:

$$S_0 = P_0$$

$$S_t = \alpha P_t + (1 - \alpha) S_{t-1},\ t > 0$$

where $\alpha \in (0, 1)$ is a smoothing factor. Various levels of smoothing factors were tried out, based on the correctness of the predictions of Random Forest we choose $\alpha = 0.3$.

From the smoothed price we derived 5 features that were inspired by Basak *et al.* (2019) and Tadlaoui (2017). Remark, all features were derived using smoothed adjusted closing price. Features are following:

- **Stochastic Oscillator (%K)**

  Stochastic Oscillator was introduced by Lane (1984). The feature provides a comparison between price and the high-low 14-days range. It is calculated as:

  $$\%K = 100 \frac{P_t - Low_{14}}{High_{14} - Low_{14}}$$

  where Low_{14} and $High_{14}$ are the lowest and highest closing prices over the last 14 days.

- **Relative Strength Index (RSI)**

  RSI is commonly used as an indicator of overbuying or overselling of

[2]In the thesis, by d-days we mean d-trading days.

the given asset, derived by Wilder (1978). The formula is following:

$$RS = \frac{avg\_gain_{14}}{avg\_loss_{14}}$$

$$RSI = 100 - \frac{100}{1 + RS}$$

where avg_gain_{14} and avg_loss_{14} denote respectively average positive and negative change in the price of the asset over the last 14 days.

- **Moving average convergence divergence (MACD)**

  MACD is the indicator of momentum, introduced by Appel (2005). The indicator shows changes in trend and strength of that change. It is calculated as the difference between two moving averages:

$$MACD = EMA_{12} - EMA_{26}$$

  where EMA_t is t-day exponential moving average of the price.

- **Signal MACD**

  It is derived from MACD:

$$Signal\_MACD = EMA_9(MACD)$$

  Signal MACD and MACD are used together. If the MACD gets over the signal, it is indicating that the price is going up and vice versa if the MACD is below the signal.

- **7-Day Difference between On Balance Volume (OBV)**

  OBV is an indicator of trends by analyzing the traded volume (Granville, 1976). If the price goes up, the traded volume is added to the OBV, and if the price goes down, the volume is subtracted. The formula is following:

$$OBV_t = OBV_{t-1} + \begin{cases} Volume_t, & \text{if } P_t > P_{t-1} \\ 0, & \text{if } P_t = P_{t-1} \\ -Volume_t, & \textit{if } P_t < P_{t-1} \end{cases}$$

  Basak *et al.* (2019) used OBV in their analysis as it is. However, when we were examining the importance of variables in Random Forest, the

importance of OBV was not sufficient. Thus, we derived a 7-day difference between OBV, which achieved higher importance in the algorithm. The formula is following:

$$diff_7\_OBV_t = OBV_t - OBV_{t-7}$$

We norm-standardized this variable for each asset to get the same scale across assets.

- **Label Variable**

  Finally, we derived a label variable for the prediction of the stock direction. Remark, the label variable is derived from the original non-smoothed adjusted close price. It is given by:

$$Y_t = \begin{cases} 1, & \text{if } P_{t+21} \geq P_t \\ -1, & \text{if } P_{t+21} < P_t \end{cases}$$

5 Results

In this section, we are going to present empirical results. Firstly, we will cluster our stock universe using PCA and K-means algorithms. This step will provide us groups of similar stocks. Then we will, by VaR and prediction of returns, choose the best stock from each cluster.

Using the selected stocks, we will construct two portfolios. Firstly, using MVO, we denote this portfolio as "selected MVO". Secondly, using the simple 1/N rule, we name this portfolio "selected 1/n". This provides us the comparison of how big benefit provides the optimization with MVO using predicted returns with the selection. As the main benchmark, we will construct a portfolio from all stocks using the 1/N rule, this portfolio represents the market, and we name it the "1/N" portfolio. For interpreting the results, every portfolio will start with initial 100 units.

Rebalancing will be done every 21-days during 2020. The periods follows table 1.

Table 1: Rebalancing Calendar

	Start	**End**
Period 1	02.01.2020	03.02.2020
Period 2	03.02.2020	04.03.2020
Period 3	04.03.2020	02.04.2020
Period 4	02.04.2020	04.05.2020
Period 5	04.05.2020	03.06.2020
Period 6	03.06.2020	02.07.2020
Period 7	02.07.2020	03.08.2020
Period 8	03.08.2020	01.09.2020
Period 9	01.09.2020	01.10.2020
Period 10	01.10.2020	30.10.2020
Period 11	30.10.2020	01.12.2020
Period 12	01.12.2020	31.12.2020

5.1 Clustering: PCA and K-means

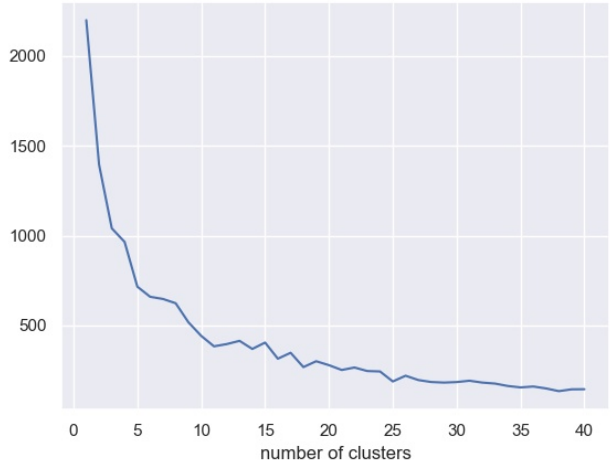
Initially, we will use PCA[3] on the norm-standardized dataset described in the section 4.1. The goal of using PCA is reducing the dimensionality of the data. Therefore, reducing the complexity of clustering, but also eliminating the highly intercorrelated variables. PCA achieved this by transforming variables into principal components as we described in the methodology 3.1.1. The table 2 shows how much variance of the original data is explained by each principal component. First 6 principal components are explaining almost 90 percent of the variance. Therefore, we are choosing 6 principal components for further analysis.

Table 2: Explained Variance by Principal Components

Principal Component	**PC1**	**PC2**	**PC3**	**PC4**	**PC5**	**PC6**	**PC7**	**PC8**	**PC9**	**PC10**
Explained Variance (%)	32.35	17.38	13.40	10.53	8.42	7.60	5.53	3.26	1.42	0.12
Cumulative Variance (%)	32.35	49.73	63.13	73.65	82.07	89.67	95.20	98.46	99.88	100

The next step is clustering the companies using the K-means[4] algorithm. We firstly run K-means for 1 to 40 clusters to obtain Within-Cluster-Sum of Squared Errors (WSS). The WSS was plotted against a number of clusters in the figure 4.

Figure 4: WSS from K-means



[3]We used PCA from Python library *scikit*, *sklearn.decomposition.PCA*
[4]*sklearn.cluster.KMeans*

The good number of clusters seems to be 11 as the WSS stops decreasing so drastically. We tried 11 clusters, but it did not achieve what we wanted from clustering. K-means with 11 clusters generates 1 big cluster and a lot of small ones. It seems that with this setting, the algorithm is not able to break the big group "in the middle".

Therefore, we choose 16 clusters, and it provides a better split of the dataset. Evaluating the quality of clusters derived by the K-means is difficult. We are at least presenting the table 3, where we are comparing our clusters with GICS sectors[5].

Table 3: Comparison of Clusters with GICS Sector

GICS_Sector / Cluster	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**11**	**12**	**13**	**14**	**15**	**16**
Consumer Discretionary	6	11	1	5	0	0	12	0	16	1	15	2	0	1	1	1
Consumer Staples	4	11	0	7	0	1	1	0	5	0	1	0	0	2	0	1
Energy	0	8	0	2	0	1	6	0	2	0	4	0	0	1	0	4
Financials	9	16	0	3	0	0	4	0	18	0	3	2	0	1	0	3
Health Care	11	4	0	8	0	1	0	0	5	0	15	3	1	0	1	3
Industrials	8	14	0	3	2	0	2	0	14	0	14	0	0	2	0	0
Information Technology	1	12	0	5	0	2	5	2	5	0	17	0	2	1	0	8
Materials	2	5	0	0	0	0	2	0	9	0	4	0	0	0	0	0
Real Estate	0	0	0	0	0	0	5	0	0	0	0	0	1	0	0	21
Telecommunications Services	0	0	0	0	0	2	1	0	0	0	0	0	0	0	0	0
Utilities	0	1	0	0	0	0	18	0	8	0	0	0	0	0	0	0
Cardinality	41	82	1	33	2	7	56	2	82	1	73	7	4	8	2	41

Interesting is that the 'Real Estate' sector is mostly in one cluster. Similar is for the 'Utilities' sector, where 18 companies are in the 7th cluster with 12 companies from 'Consumer Discretionary' and few other companies. The remaining sectors are spread across the clusters. There are two big clusters with 82 companies. But there are 6 clusters with less than 5 companies. It is unreasonable to choose the best asset from a cluster with 2 assets. Thus, we decided to merge these 6 clusters into one. We can view the merged cluster as the cluster of outliers. The cardinality of the final clusters is presented in table 4.

[5]The Global Industry Classification Standard (GICS) is used by S&P to separate companies into industrial sectors.

Table 4: Cardinality of Cluster

Cluster	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**11**
Cardinality	41	82	12	33	7	56	82	73	7	8	41

5.2 Predictions and Construction of Portfolio

For every period in the table 1, we are now going to choose one stock from each cluster. We firstly will calculate VaR for each asset. The expected return will be estimated by the prediction of direction and volatility. Then based on the $return/risk$ metric, we will select stocks for the portfolio. All these steps are done in a three-year rolling window presented in the table 5.

Table 5: Rolling windows for prediction

	Start train	**End train RF**	**End train**	**Forecast day**
Period 1	03.01.2017	02.12.2019	02.01.2020	03.02.2020
Period 2	02.02.2017	02.01.2020	03.02.2020	04.03.2020
Period 3	06.03.2017	03.02.2020	04.03.2020	02.04.2020
Period 4	04.04.2017	04.03.2020	02.04.2020	04.05.2020
Period 5	04.05.2017	02.04.2020	04.05.2020	03.06.2020
Period 6	05.06.2017	04.05.2020	03.06.2020	02.07.2020
Period 7	05.07.2017	03.06.2020	02.07.2020	03.08.2020
Period 8	03.08.2017	02.07.2020	03.08.2020	01.09.2020
Period 9	01.09.2017	03.08.2020	01.09.2020	01.10.2020
Period 10	03.10.2017	01.09.2020	01.10.2020	30.10.2020
Period 11	01.11.2017	01.10.2020	30.10.2020	01.12.2020
Period 12	01.12.2017	30.10.2020	01.12.2020	31.12.2020

All methods will follow **Start train - End train** period except for Random Forest. The ending period for training the Random Forest must be shifted back by 21-days, because in these 21-days we would not know the label variable (direction).

5.2.1 VaR

We calculated VaR using the historical approach described in the section 3.2 from a 3-years rolling window of 21-days returns. We tried VaR_{95} and VaR_{99}. The VaR_{99} seems to be too strict for 3-year data. Thus, we choose VaR_{95}.

We then removed from each cluster 50 percent of stocks with the highest VaR. This assures us that the riskiest assets will not be in further analysis. Because the GARCH model was commonly unstable for the highly volatile stocks and overestimated them a lot. Also this step lowers the complexity of the following task.

5.2.2 Random Forest - Direction Prediction

For setting parameters in Random Forest[6] we firstly run GridSearch Cross-Validation[7] which randomly split the dataset, into training and validation datasets. Then the model is trained for multiple settings and GridSearch chooses the best performing setting based on performance on the validation dataset.

After inspecting the recommended setting, we decided to rather use the default setting. Even though the recommended setting had better accuracy, but it failed to predict negative returns in the validation set. Therefore the setting is following: *n_estimators=101*[8]*, max_depth=None, min_samples_split=2, min_samples_leaf=1*

Also, we made a decision if to train the model on the whole dataset or on each asset separately. The model trained for each asset separately gives better results on the validation set. Thus, we choose this approach.

The procedure is following, for each stock we take features derived in the section 4.2 from period **Start train - End train RF** denoted in the table 5. Then the dataset is randomly split into training and validation datasets

[6] *sklearn.ensemble.RandomForestClassifier*

[7] *sklearn.model_selection.GridSearchCV*

[8] Default is 100, but we changed it to 101 to have an odd number of trees.

by a ratio of 80:20. We trained the Random Forest on the training dataset and then obtain performance metrics on the validation dataset. For better predictions, we subsequently trained the model using all data (training + validation dataset). Moreover, using the features from **End train** date, we predicted the direction of the asset's price for the next 21 days. This was done for all stocks (= the better half, which remained after the VaR filtering).

In the table 6, we present average accuracy, precision, and recall from the validation set for every period. The average of metrics is stable over the periods. The accuracy is between 0.76 and 0.78. Precision is a little bit higher but similar to the accuracy and recall is mostly around 0.84.

Table 6: Average performance of Random Forest

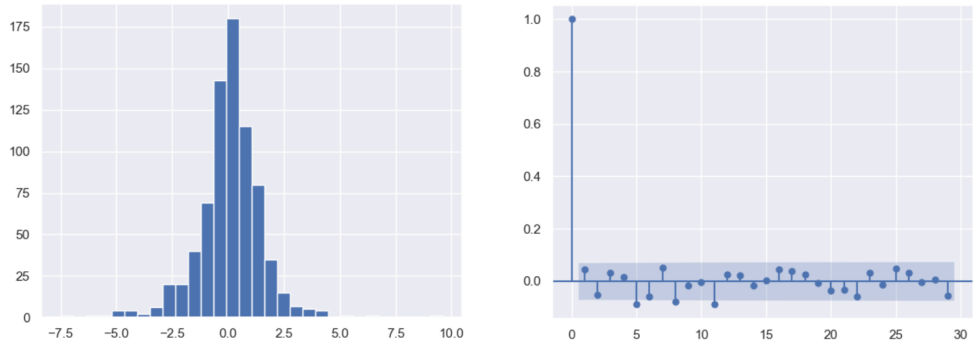
	Accuracy	**Precision**	**Recall**
Period 1	0.765	0.781	0.854
Period 2	0.762	0.768	0.859
Period 3	0.769	0.789	0.854
Period 4	0.763	0.779	0.834
Period 5	0.767	0.788	0.842
Period 6	0.772	0.800	0.834
Period 7	0.764	0.771	0.844
Period 8	0.768	0.787	0.846
Period 9	0.774	0.792	0.852
Period 10	0.763	0.783	0.841
Period 11	0.765	0.776	0.842
Period 12	0.766	0.786	0.843

5.2.3 GARCH - Volatility Prediction

Firstly we checked for assumptions in the GARCH model. In the figure 5, we present a histogram of standardized residuals ($\frac{\epsilon_t}{\sigma_t}$) obtained in the GARCH process and ACF plot for the daily returns of the Google stock. The distribution of standardized residuals seems to be close to the t-distribution. The second plot, the ACF plot, investigates the autocorrelation in the stand-

ardized residuals. On the x-axis, we can see a number of lags and their significance. The shaded area corresponds to the 5% confidence level. Most of the lags are in the area of the 5% confidence level, except the 5th, 8th, and 11th lags that are a little bit out of the shaded area. Nevertheless, there is no pattern in the points that represent autocorrelation. From that, we can conclude that there is no significant autocorrelation in our data.

Figure 5: Histogram of standardized residuals and ACF plot - Google



For the volatility forecast, we fitted the GARCH model[9] for each stock separately using daily returns from **Start train - End train** period (table 5). Then we forecast daily variance for the next 21-days. From these 21 predicted values, we calculated the mean and multiply it by 21 to transform it to 21-days variance. Volatility was then derived as the square root of the 21-day variance.

The predicted volatility was then used to calculate the expected return by multiplicating the volatility with the prediction of the asset's direction from the Random Forest.

[9]Using *arch_model* package in Python (*arch_model.arch*)

5.2.4 Portfolio Construction

Initially, we chose one stock from each cluster. The decision was made using both predicted returns and VaR by the following formula:

$$return/risk = \frac{Predicted\_Return}{VaR_{95}}$$

The stock with the highest $return/risk$ was chosen from each cluster. This leaves us with 11 stocks out of the total 442. The $return/risk$ can be problematic if the VaR is negative. It could theoretically happen, but not likely in reality, as the given stock would need to have almost only positive returns. We did not observe any negative VaR in our data. Thus, we did not need to handle this case.

From the 11 stocks, the "selected 1/n" portfolio was created in the way that every stock was bought to represent 1/11 of the whole portfolio.

The next step is constructing the "selected MVO" portfolio. The MVO takes two arguments, expected returns and covariance matrix. We used the predicted returns derived in the section 5.2.2 and 5.2.3 as the expected returns. The covariance matrix was estimated from historical daily-returns from **Start train - End train** period (table 5).

All calculations were done using the *PyPortfolioOpt* Python package. The covariance matrix[10] was transformed from daily data to the 21-days by setting the frequency to 21; thus, it matches the 21-days expected returns. The weights for the given assets were then calculated with constraint to maximize the Sharpe ratio.

The table 7 presents portfolios for the first period. The right direction was predicted for 8/11 of these stocks. The accuracy of the prediction of the exact returns is worse. As we can notice that the best prediction is for **MA**. Another good prediction was made for **CHTR**, but the remaining were mispredicted by more than 0.015. This confirms the nature of the returns data as they are hard to predict. Although, the "selected 1/n" and "selected MVO" portfolios achieved positive returns during the first period. The

[10] *pypfopt.risk_models.sample_cov*

Table 7: Period 1

Company_Symbol	cluster	forecast_return	weights	real_return
TMO	1	0.05594	0.02412	-0.02460
WM	2	0.04319	0.10650	0.07416
EQIX	3	0.05813	0.07782	0.04186
CSCO	4	0.06701	0.04120	-0.03903
MSFT	5	0.05844	0	0.08567
CNP	6	0.05606	0.2	-0.02263
NEE	7	0.04199	0.18332	0.11814
ZTS	8	0.06117	0.03453	0.01175
CHTR	9	0.07295	0.12240	0.08486
INTU	10	0.10197	0.2	0.06459
MA	11	0.07058	0.01010	0.07107
		Initial Value	**Ending Value**	**Return**
1/N		100	98.19471	-0.01805
Selected 1/n		100	104.23471	0.04235
Selected MVO		100	105.05122	0.05051

"1/N" portfolio, representing the market, finished with a negative return. Therefore, we can conclude that clustering and selecting the stock provide a positive effect in constructing a portfolio.

In the next table 8, we present results for the third period, which take place during March 2020. There we can see a reaction to the start of the COVID pandemic. We failed to predict the right direction for all selected stocks as all the actual returns are negative. It is important to note that only 14 from 442 assets achieved a positive return. However, the return for the "1/N" portfolio was -0.26241. Thus, the fall in the market was by more than one quarter. Nevertheless, the clustering and selection of the stock seem to provide at least a little positive effect. The return for "selected 1/n" was -0.16155, and for "selected MVO" -0.18552. Thus, the selection of stocks achieved a smaller drop in the portfolio value.

The remaining periods are presented only in the Appendix A because we can see their results in the next section, where we examine the overall

performance.

Table 8: Period 3

Company_Symbol	cluster	forecast_return	weights	real_return
TMO	1	0.13186	0	-0.11899
FIS	2	0.14810	0.18001	-0.21684
EQIX	3	0.10659	0	-0.02996
UNH	4	0.20980	0.2	-0.16564
MSFT	5	0.18112	0.07957	-0.08965
D	6	0.14362	0.2	-0.20184
NEE	7	0.11549	0.2	-0.19761
FISV	8	0.12257	0	-0.20886
MTD	9	0.11120	0	-0.15289
SPGI	10	0.16181	0.12534	-0.18483
MA	11	0.17366	0.01508	-0.20995
		Initial Value	**Ending Value**	**Return**
1/N		92.91590	68.53387	-0.26241
Selected 1/n		102.43842	85.88939	-0.16155
Selected MVO		103.44463	84.25386	-0.18552

5.3 Evaluation of Results

Figure 6: Value of Portfolios

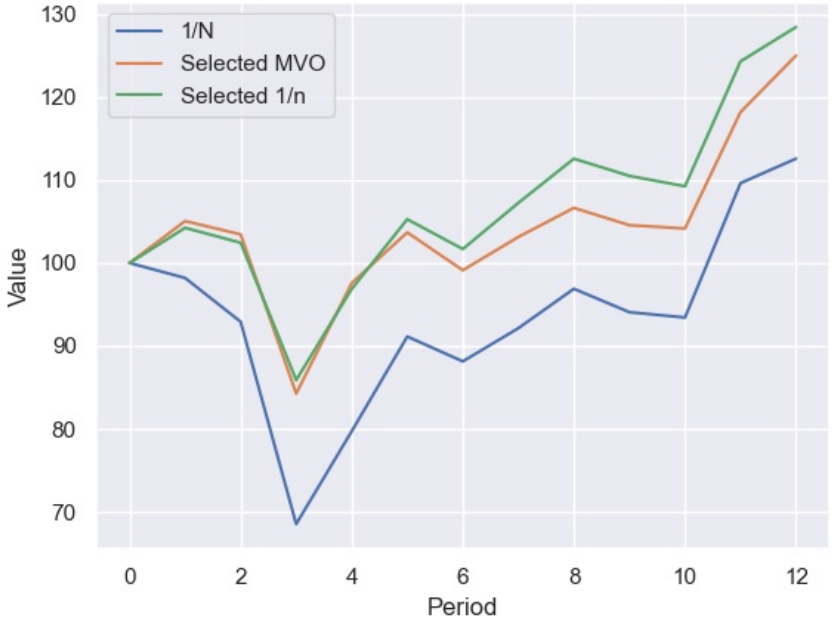
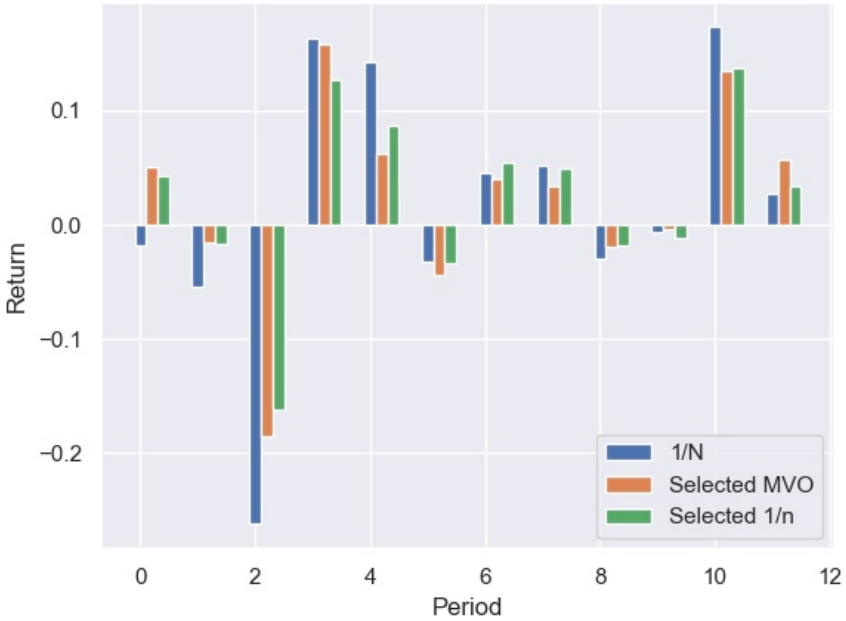


Figure 7: Returns for Portfolios



It is evident that the "selected" portfolios are above the "1/N" (figure 6). That is a good sign of the benefit of the clustering. However, when we compare the "selected" portfolios between themselves, then it seems that the "1/n" strategy achieved better results than the strategy with MVO. Thus, it does not support our hypothesis that MVO will improve performance.

The figure 7 shows returns of portfolios for every period. We can see that the "selected" portfolios were outperforming the "1/N" mainly during the first three periods. Then the differences diminished.

The average returns are higher for "selected" portfolios than for the "1/N" (table 9). The "selected 1/n" has a higher average return by more than 44 percent, and the "selected MVO" has higher average returns by 34 percent than the market benchmark "1/N". In addition, both "selected" portfolios achieved a lower standard deviation of returns (volatility), which indicates that we were able to obtain some level of diversification of the portfolio by using the clustering method. Furthermore, the annualized Sharpe ratio is higher for the portfolios with clustering. Where "selected 1/n" has twice as large Sharpe ratio as "1/N". Thus, it supports our claim that by the selection of stocks from clusters, we will be able to outperform the market.

Table 9: Results of Portfolios

	Average Return	**Standard Deviation**	**Annualized Sharpe Ratio**
1/N	0.01672	0.11304	0.51236
Selected 1/n	0.02415	0.07759	1.07801
Selected MVO	0.02249	0.08497	0.91684

The comparison of "selected MVO" and "selected 1/n" shows that we did not demonstrate the efficiency of MVO enabled with Random Forest and GARCH predictions as showed by Tadlaoui (2017). But our analysis differs significantly in the number of stock as we used a dataset of 442 stock. Furthermore, we were not just focusing on the MVO, but on the stock selection also.

The results indicate that gains, in comparison to the "1/N" portfolio, were mostly due to clustering and selection of stocks. The application of the clustering during the portfolio construction was already shown by Fulga & Dedu (2012). However, we extended it with rebalancing the portfolio. It seems that clustering and selecting the stocks from each cluster can be practical as we were able to construct portfolio using just 11 assets from the 442 in total, and we accomplished higher average returns and lower volatility than the "1/N" market benchmark portfolio during the fluctuating year 2020. A portfolio with 11 stocks seems to be attainable for the individual investor.

However, the better results for the "selected 1/n" than for the "selected MVO" portfolio imply that the predictions of the returns were not accurate enough to obtain better results with MVO.

We also tested various combinations of our model, as return prediction with MVO without clustering, clustering with estimating returns only based on historical data, etc. However, these combinations did not outperform the presented ones. Thus, we decided to present them in the appendices to keep the results readable and straightforward.

Appendix B compares the presented portfolios with MVO without any machine learning techniques. The MVO has volatility similar to our "selected" portfolios, but the MVO has a lower Sharpe ratio than the market benchmark because of the low average return.

Comparison of portfolios with a selection of stocks from clusters but without stock prediction is in Appendix C. The expected returns were estimated from historical 21-days returns, then the process was identical to portfolios presented above. Portfolios were able to outperform the market benchmark but did not achieve the same result as portfolios with returns prediction in terms of the Sharpe ratio. This demonstrates that the machine learning predictions of returns improved our model.

Appendix D compares the 1/N strategy constructed from stocks that

remained after filtering half of them from each cluster based on VaR in the section 5.2.1. It shows that it was not the most influential step in our thesis as the main benefit of this filtering seems to be that we obtain the subset of assets with lower volatility. Although, it is an important step as we are predicting returns using prediction of volatility. Therefore, it is a safety step as very risky stocks were highly overestimated by Random Forrest + GARCH approach.

6 Conclusion

The main goal of the thesis was to use clustering of the stock universe followed by selecting the stocks from each cluster to construct a portfolio that will be able to outperform the market benchmark. Also, we wanted to demonstrate the benefit of using MVO with the stock return prediction.

We conducted an analysis on 442 American stocks. Firstly by clustering them based on information about the companies using PCA and K-means algorithms. This approach splits our dataset into 11 clusters. The goal of clustering was to pre-diversify our stocks universe. Based on these clusters, we were constructing a portfolio that was rebalanced 12-times during 2020.

Every rebalancing started by calculating VaR. We filtered out 50 percent of stocks from each cluster based on VaR. This was followed by returns predictions for the next 21-days using a combination of the Random Forest and the GARCH model. Then from each cluster, we picked 1 stock based on return/risk value. Therefore, for each period, we ended with 11 stocks that were used for portfolio construction by MVO and the naive 1/N approach.

Both portfolios were able to outperform the market benchmark during 2020 with higher average returns and lower volatility. The lower volatility implies that the clustering helps us in diversifying the portfolio. However, results indicate that the gain in returns was mainly achieved by clustering and the selection of the stocks from clusters. As the 1/N portfolio from the selected stocks accomplished better results than the portfolio using MVO. Thus, we failed to demonstrate the benefit of MVO with machine learning.

We consider that it is mainly due to errors in return predictions; thus, this part could be improved in further research. However, our proposed model could still be practical as with clustering and picking the right stocks. We were able to construct a portfolio with 11 stocks that were able to outperform the market benchmark. This low number of stocks in the portfolio we consider obtainable for the individual investor.

References

Appel, G. (2005): *Technical Analysis: Power Tools For The Active Investors*. Upper Saddle River, NJ: Ft Pr.

Ballings, M., D. Van den Poel, N. Hespeels, & R. Gryp (2015): "Evaluating multiple classifiers for stock price direction prediction." *Expert Systems with Applications* **42(20)**: pp. 7046–7056.

Basak, S., S. Kar, S. Saha, L. Khaidem, & S. R. Dey (2019): "Predicting the direction of stock market prices using tree-based classifiers." *The North American Journal of Economics and Finance* **47**: pp. 552–567.

Bishop, C. (2006): *Pattern Recognition and Machine Learning*. Information Science and Statistics. New York: Springer-Verlag.

Bollerslev, T. (1986): "Generalized autoregressive conditional heteroskedasticity." *Journal of Econometrics* **31(3)**: pp. 307–327.

Breiman, L., J. Friedman, C. J. Stone, & R. A. Olshen (1984): *Classification and Regression Trees*. Boca Raton: Chapman and Hall/CRC, 1st edition edition.

Campbell, J. Y., A. W. Lo, A. C. MacKinlay, & A. Y. Lo (1996): *The Econometrics of Financial Markets*. Princeton, N.J: Princeton University Press, 2nd ed. edition edition.

Chen, W., H. Zhang, M. K. Mehlawat, & L. Jia (2021): "Mean–variance portfolio optimization using machine learning-based stock price prediction." *Applied Soft Computing* **100**.

DeMiguel, V., L. Garlappi, & R. Uppal (2009): "Optimal Versus Naive Diversification: How Inefficient is the 1/N Portfolio Strategy?" *The Review of Financial Studies* **22(5)**: pp. 1915–1953.

Dupačová, J., J. Hurt, & J. Štěpán (2006): *Stochastic Modeling in Economics and Finance*. Springer Science & Business Media.

Fulga, C. & S. Dedu (2012): "Mean-Risk portfolio optimization with prior PCA-based stock selection." *Proceedings. Vilnius* pp. 37–42.

Goodfellow, I., Y. Bengio, & A. Courville (2016): *Deep Learning*. The MIT Press.

Granville, J. E. (1976): *Granville's New Strategy of Daily Stock Market Timing for Maximum Profit*. Englewood Cliffs, N.J: Simon & Schuster.

Hansen, P. R. & A. Lunde (2005): "A forecast comparison of volatility models: does anything beat a GARCH(1,1)?" *Journal of Applied Econometrics* **20(7)**: pp. 873–889.

Hastie, T., R. Tibshirani, & J. Friedman (2016): *The Elements of Statistical Learning: Data Mining, Inference, and Prediction, Second Edition*. New York, NY: Springer, 2nd edition edition.

Hull, J. C. (2015): *Risk Management and Financial Institutions, Fourth Edition*. Hoboken: John Wiley &Sons, 4th edition edition.

Jorion, P. (2006): *Value at Risk, 3rd Ed.: The New Benchmark for Managing Financial Risk*. McGraw-Hill Education, 3rd edition edition.

Kaczmarek, T. & K. Perez (2021): "Building portfolios based on machine learning predictions." *Economic Research-Ekonomska Istraživanja* **0(0)**: pp. 1–19.

Lane, G. (1984): "Lanes stochastics." *Technical Analysis of Stocks & Commodities* **2**: pp. 87–90.

Lemieux, V., P. S. Rahmdel, R. Walker, B. L. W. Wong, & M. Flood (2014): "Clustering Techniques And their Effect on Portfolio Formation and Risk Analysis." In "Proceedings of the International Workshop on Data Science for Macro-Modeling," DSMM'14, pp. 1–6. New York, NY, USA: Association for Computing Machinery.

Lintner, J. (1965): "Security Prices, Risk, and Maximal Gains from Diversification*." *The Journal of Finance* **20(4)**: pp. 587–615.

Lloyd, S. (1982): "Least squares quantization in PCM." *IEEE Transactions on Information Theory* **28(2)**: pp. 129–137.

MacQueen, J. *et al.* (1967): "Some methods for classification and analysis of multivariate observations." In "Proceedings of the fifth Berkeley symposium on mathematical statistics and probability," volume 1, pp. 281–297. Oakland, CA, USA.

Markowitz, H. (1952): "Portfolio Selection." *The Journal of Finance* **7(1)**: pp. 77–91.

Miah, M., A. Rahman *et al.* (2016): "Modelling volatility of daily stock returns: Is garch (1, 1) enough?" *American Scientific Research Journal for Engineering, Technology, and Sciences (ASRJETS)* **18(1)**: pp. 29–39.

Mirkin, B. (2011): "Choosing the number of clusters." *Wiley Interdisc. Rew.: Data Mining and Knowledge Discovery* **1**: pp. 252–260.

Mossin, J. (1966): "Equilibrium in a Capital Asset Market." *Econometrica* **34(4)**: pp. 768–783.

Pasini, G. (2017): "Principal Component Analysis for Stock Portfolio Management." *International Journal of Pure and Applied Mathematics* **115(1)**: pp. 153–167.

Pearson, K. (1901): "Liii. on lines and planes of closest fit to systems of points in space." *The London, Edinburgh, and Dublin Philosophical Magazine and Journal of Science* **2(11)**: pp. 559–572.

Sharpe, W. F. (1964): "Capital Asset Prices: A Theory of Market Equilibrium Under Conditions of Risk." *The Journal of Finance* **19(3)**: pp. 425–442.

Sharpe, W. F. (1966): "Mutual Fund Performance." *The Journal of Business* **39(1)**: pp. 119–138.

Snow, D. (2020): "Machine Learning in Asset Management—Part 2: Portfolio Construction—Weight Optimization." *The Journal of Financial Data Science* **2(2)**: pp. 17–24.

Steinhaus, H. (1957): "Sur la division des corps matériels en parties." *Bulletin de l'Académie Polonaise des Sciences, Classe 3* **4**: pp. 801–804.

Tadlaoui, G. (2017): *Intelligent Portfolio Construction: Machine-Learning enabled Mean-Variance Optimization.* Master's thesis, Imperial College London.

Treynor, J. L. (1962): "Toward a Theory of Market Value of Risky Assets." *SSRN Scholarly Paper ID 628187*, Social Science Research Network, Rochester, NY.

VanderPlas, J. (2016): *Python Data Science Handbook: Essential Tools for Working with Data.* O'Reilly Media, Inc., 1st edition.

Wilder, J. W. (1978): *New Concepts in Technical Trading Systems.* Greensboro, N.C: Trend Research.

Yu, H., R. Chen, & G. Zhang (2014): "A SVM Stock Selection Model within PCA." *Procedia Computer Science* **31**: pp. 406–412.

A Appendix 1

Table 10: Period 1

Company_Symbol	cluster	forecast_return	weights	real_return
TMO	1	0.05594	0.02412	-0.02460
WM	2	0.04319	0.10650	0.07416
EQIX	3	0.05813	0.07782	0.04186
CSCO	4	0.06701	0.04120	-0.03903
MSFT	5	0.05844	0	0.08567
CNP	6	0.05606	0.2	-0.02263
NEE	7	0.04199	0.18332	0.11814
ZTS	8	0.06117	0.03453	0.01175
CHTR	9	0.07295	0.12240	0.08486
INTU	10	0.10197	0.2	0.06459
MA	11	0.07058	0.01010	0.07107
		Initial Value	**Ending Value**	**Return**
1/N		100	98.19471	-0.01805
Selected 1/n		100	104.23471	0.04235
Selected MVO		100	105.05122	0.05051

Table 11: Period 2

Company_Symbol	cluster	forecast_return	weights	real_return
TMO	1	0.07846	0.04094	0.02607
FIS	2	0.05522	0	0.02868
AMZN	3	0.12873	0.19189	-0.01416
PFE	4	0.07850	0.2	-0.02959
WMT	5	0.05347	0.06010	0.02188
PEG	6	0.04435	0.14916	-0.06304
NEE	7	0.04161	0.18538	0.06328
FISV	8	0.06527	0.01154	-0.08004
GS	9	0.06834	0.06033	-0.12131
INTU	10	0.08090	0	0.00972
V	11	0.08938	0.10066	-0.03107
		Initial Value	**Ending Value**	**Return**
1/N		98.19471	92.91590	-0.05376
Selected 1/n		104.23471	102.43842	-0.01723
Selected MVO		105.05122	103.44463	-0.01529

Table 12: Period 3

Company_Symbol	cluster	forecast_return	weights	real_return
TMO	1	0.13186	0	-0.11899
FIS	2	0.14810	0.18001	-0.21684
EQIX	3	0.10659	0	-0.02996
UNH	4	0.20980	0.2	-0.16564
MSFT	5	0.18112	0.07957	-0.08965
D	6	0.14362	0.2	-0.20184
NEE	7	0.11549	0.2	-0.19761
FISV	8	0.12257	0	-0.20886
MTD	9	0.11120	0	-0.15289
SPGI	10	0.16181	0.12534	-0.18483
MA	11	0.17366	0.01508	-0.20995
		Initial Value	**Ending Value**	**Return**
1/N		92.91590	68.53387	-0.26241
Selected 1/n		102.43842	85.88939	-0.16155
Selected MVO		103.44463	84.25386	-0.18552

Table 13: Period 4

Company_Symbol	cluster	forecast_return	weights	real_return
HUM	1	0.33247	0.15460	0.25901
FIS	2	0.25014	0	0.09794
ISRG	3	0.29683	0.04610	0.07327
UNH	4	0.30355	0.01994	0.19589
WMT	5	0.08814	0	0.04256
ES	6	0.25634	0.2	0.01904
NEE	7	0.24422	0.15093	0.01607
FISV	8	0.28182	0.12060	0.14411
REGN	9	0.10626	0	0.08636
OKE	10	0.50449	0.2	0.39416
CCI	11	0.24006	0.10783	0.07761
		Initial Value	**Ending Value**	**Return**
1/N		68.53387	79.72347	0.16327
Selected 1/n		85.88939	96.86781	0.12782
Selected MVO		84.25386	97.57789	0.15814

Table 14: Period 5

Company_Symbol	cluster	forecast_return	weights	real_return
HUM	1	0.18486	0.13188	0.04205
FIS	2	0.11837	0	0.08547
GOOGL	3	0.15814	0.15273	0.08795
GILD	4	0.14884	0.2	-0.07290
VZ	5	0.05893	0	0.01049
WELL	6	0.19414	0.11539	0.23900
NEE	7	0.08451	0	0.15344
L	8	0.22966	0.2	0.11934
REGN	9	0.09932	0	0.11528
CLX	10	0.08842	0.2	0.03362
MA	11	0.15471	0	0.14042
		Initial Value	**Ending Value**	**Return**
1/N		79.72347	91.13490	0.14314
Selected 1/n		96.86781	105.27042	0.08674
Selected MVO		97.57789	103.68330	0.06257

Table 15: Period 6

Company_Symbol	cluster	forecast_return	weights	real_return
NOC	1	0.11167	0.2	-0.07736
SYF	2	0.23753	0.2	-0.07725
AMZN	3	0.07897	0.07781	0.16620
JPM	4	0.13960	0.09292	-0.10269
MSFT	5	0.06107	0	0.11275
WELL	6	0.19932	0.2	-0.10751
NEE	7	0.07742	0.00607	-0.05077
ZTS	8	0.08631	0	-0.03417
REGN	9	0.11545	0.2	0.03007
OKE	10	0.13772	0	-0.20774
AIV	11	0.12847	0.02319	-0.02691
		Initial Value	**Ending Value**	**Return**
1/N		91.13490	88.14172	-0.03284
Selected 1/n		105.27042	101.67807	-0.03412
Selected MVO		103.68330	99.12620	-0.04395

Table 16: Period 7

Company_Symbol	cluster	forecast_return	weights	real_return
WLTW	1	0.09520	0.2	0.04095
WM	2	0.07518	0.11373	0.03820
AMZN	3	0.11294	0.2	0.07667
UNH	4	0.09540	0.00276	0.01794
MSFT	5	0.07832	0	0.04984
WELL	6	0.14159	0.16590	0.00456
NEE	7	0.06302	0	0.12658
ZTS	8	0.08936	0.01636	0.12398
CHTR	9	0.08434	0.12045	0.13910
INTU	10	0.08759	0	0.02271
CCI	11	0.09476	0.18080	-0.03878
		Initial Value	**Ending Value**	**Return**
1/N		88.14172	92.13949	0.04536
Selected 1/n		101.67807	107.24024	0.05470
Selected MVO		99.12620	103.13527	0.04044

Table 17: Period 8

Company_Symbol	cluster	forecast_return	weights	real_return
TMO	1	0.06622	0	0.03493
CHD	2	0.10705	0.2	0.02314
AMZN	3	0.11648	0.13879	0.12444
GILD	4	0.08811	0.08432	-0.08225
MSFT	5	0.13521	0	0.05209
FE	6	0.23681	0.2	-0.01864
ECL	7	0.11946	0.14435	0.09354
CTXS	8	0.12194	0.2	0.03847
CHTR	9	0.08466	0.03253	0.04670
INTU	10	0.08103	0	0.11383
V	11	0.08303	0	0.12052
		Initial Value	**Ending Value**	**Return**
1/N		92.13949	96.88986	0.05156
Selected 1/n		107.24024	112.57083	0.04971
Selected MVO		103.13527	106.63703	0.03395

Table 18: Period 9

Company_Symbol	cluster	forecast_return	weights	real_return
WLTW	1	0.07970	0.19601	0.01153
FIS	2	0.07372	0	-0.03369
AMZN	3	0.09175	0.2	-0.07941
KO	4	0.06207	0.08934	0.00933
VZ	5	0.04940	0.18473	0.00490
WELL	6	0.11020	0.11988	-0.02942
APD	7	0.09194	0.2	-0.02217
DHR	8	0.06684	0.01005	0.03110
MTD	9	0.06652	0	-0.00509
INTU	10	0.07396	0	-0.04274
V	11	0.07712	0	-0.04687
		Initial Value	**Ending Value**	**Return**
1/N		96.88986	94.06759	-0.02913
Selected 1/n		112.57083	110.49815	-0.01841
Selected MVO		106.63703	104.55443	-0.01953

Table 19: Period 10

Company_Symbol	cluster	forecast_return	weights	real_return
UHS	1	0.14231	0.17455	0.03154
ABT	2	0.08444	0.12726	-0.02927
GOOG	3	0.09135	0.2	0.08786
DIS	4	0.07850	0	-0.01671
MSFT	5	0.08737	0	-0.04702
DUK	6	0.11340	0.2	0.02288
NEE	7	0.07521	0	0.03565
EW	8	0.10010	0.09819	-0.09230
MTD	9	-0.07222	0	0.02780
INTU	10	0.09904	0	-0.05147
AIV	11	0.13267	0.2	-0.09349
		Initial Value	**Ending Value**	**Return**
1/N		94.06759	93.42858	-0.00679
Selected 1/n		110.49815	109.24722	-0.01132
Selected MVO		104.55443	104.15370	-0.00383

Table 20: Period 11

Company_Symbol	cluster	forecast_return	weights	real_return
ROP	1	0.11528	0.06772	0.14181
FIS	2	0.13117	0.13228	0.20700
AMZN	3	0.15728	0.2	0.06058
UNP	4	0.10977	0	0.14765
MSFT	5	0.11331	0	0.07066
WU	6	0.14685	0.2	0.16255
ADM	7	0.14462	0.2	0.09153
PYPL	8	0.14938	0	0.16338
MTD	9	0.08564	0	0.16252
INTU	10	0.10717	0	0.13391
MA	11	0.17490	0.2	0.17406
		Initial Value	**Ending Value**	**Return**
1/N		93.42858	109.62399	0.17335
Selected 1/n		109.24722	124.29996	0.13779
Selected MVO		104.15370	118.18617	0.13473

Table 21: Period 12

Company_Symbol	cluster	forecast_return	weights	real_return
TMO	1	0.09215	0.2	-0.01083
SYY	2	0.11871	0.04937	0.01130
EQIX	3	0.07913	0.01811	0.00225
JPM	4	0.11696	0.11953	0.06122
MSFT	5	0.06922	0	0.02872
ETR	6	0.06770	0	-0.08277
NEE	7	0.06336	0	0.04046
PYPL	8	0.11395	0.06153	0.08156
CHTR	9	0.08228	0.15146	-0.00106
SPGI	10	0.12171	0.2	-0.01672
AIV	11	0.13186	0.2	0.25198
		Initial Value	**Ending Value**	**Return**
1/N		109.62399	112.57786	0.02695
Selected 1/n		124.29996	128.43686	0.03328
Selected MVO		118.18617	125.00071	0.05766

B Appendix 2

Figure 8: Comparison of Values with MVO

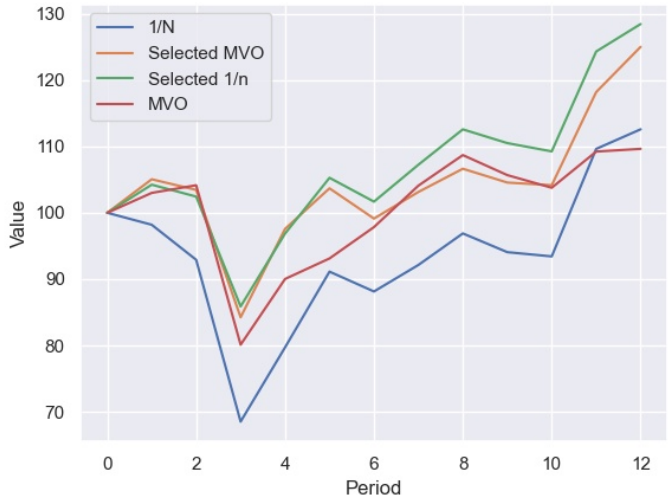


Figure 9: Comparison of Returns with MVO

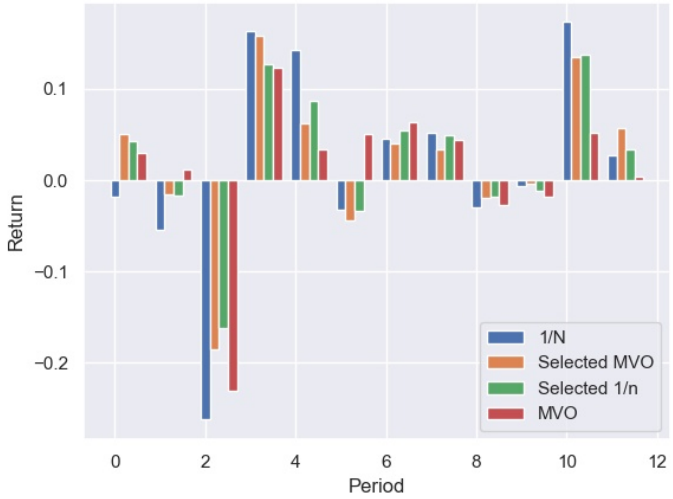


Table 22: Comparison of Results with MVO

	Average Return	**Standard Deviation**	**Annualized Sharpe Ratio**
1/N	0.01672	0.11304	0.51236
Selected 1/n	0.02415	0.07759	1.07801
Selected MVO	0.02249	0.08497	0.91684
MVO	0.01148	0.08246	0.48211

C Appendix 3

Figure 10: Comparison of Values with Portfolios with Historic Returns

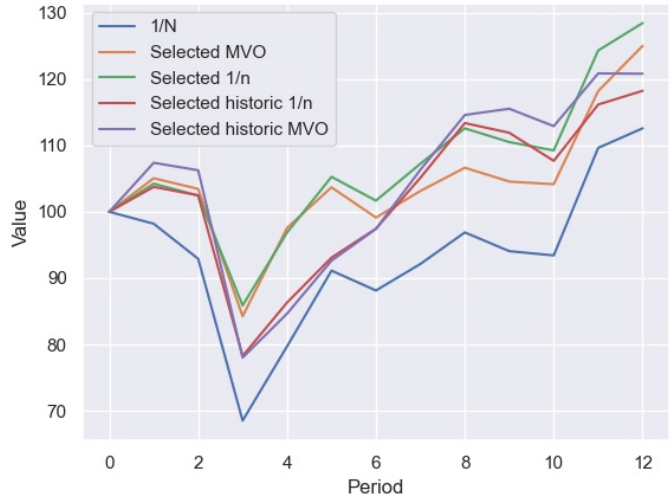


Figure 11: Comparison of Returns with Portfolios with Historic Returns

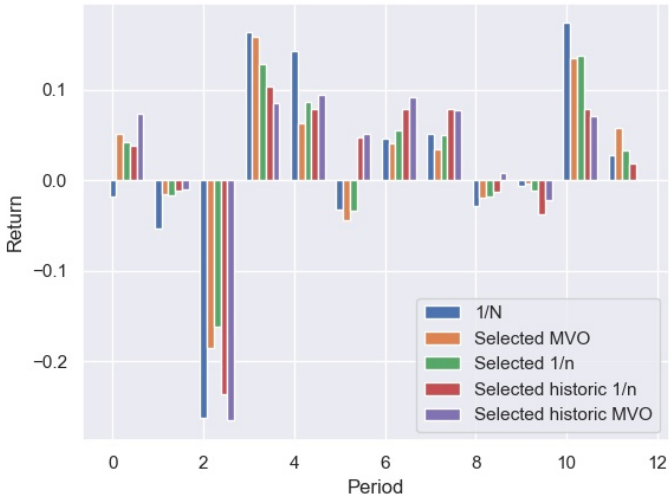


Table 23: Comparison of Results with Portfolios with Historic Returns

	Average Return	**Standard Deviation**	**Annualized Sharpe Ratio**
1/N	0.01672	0.11304	0.51236
Selected 1/n	0.02415	0.07759	1.07801
Selected MVO	0.02249	0.08497	0.91684
Selected historic 1/n	0.01835	0.08789	0.72325
Selected historic MVO	0.02110	0.09555	0.76480

D Appendix 4

Figure 12: Comparison of Values with VaR 1/N

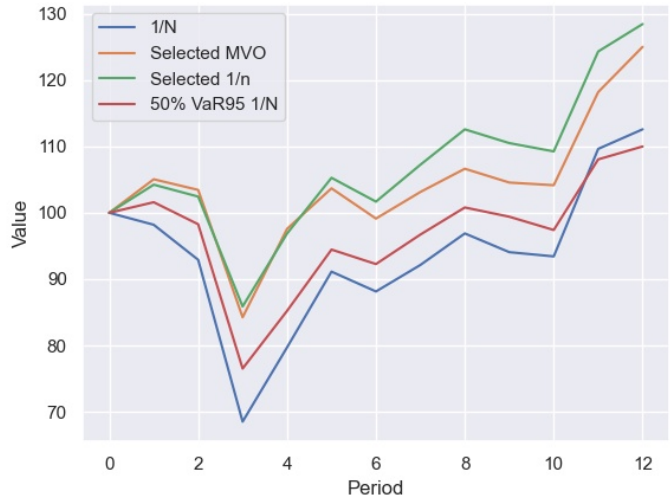


Figure 13: Comparison of Returns with VaR 1/N

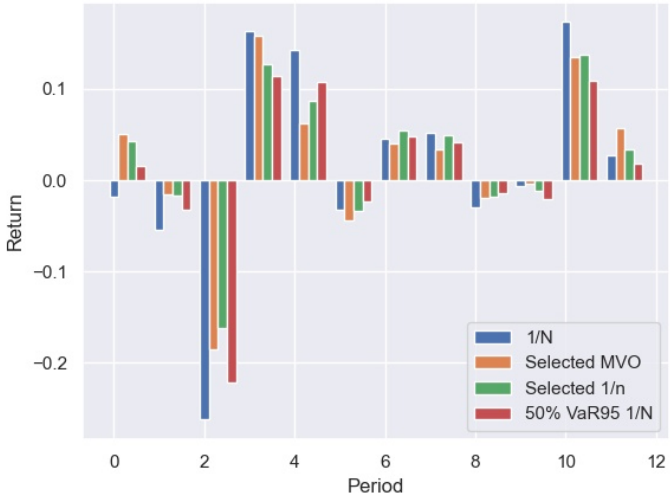


Table 24: Comparison of Results with VaR 1/N

	Average Return	**Standard Deviation**	**Annualized Sharpe Ratio**
1/N	0.01672	0.11304	0.51236
Selected 1/n	0.02415	0.07759	1.07801
Selected MVO	0.02249	0.08497	0.91684
50% VaR95 1/N	0.01204	0.08693	0.47983