Aalto University
School of Science

MS-E2177 - Seminar on Case Studies in Operations Research

Applying Advanced Analytics in Asset Allocation

Final Report

Client:
Varma

Team:
Leevi Olander (Project Manager)
Julius Lind
Samuli Mäkinen
Rasmus Heikkilä

May 25, 2019

Contents

1 Introduction

1.1 Background

The pension system provides individuals with income after they have retired from working life and no longer earn regular income from employment. In Finland, there are three statutory pensions: the national pension, the guarantee pension and earnings-related pension. The first two provide individuals with basic income security and are administered by the Social Insurance Institution of Finland (Kansaneläkelaitos). The earnings-related pensions are administered by many organizations. The national pension and earnings-related pension complement each other: as the amount of the earnings-related pension that an individual is entitled to increases, the amount of national pension decreases. [1]

The national pension and the guarantee pension are funded from the state budget. In contrast, the earnings-related pensions are funded by insurance contributions from employers and employees. Funding of the earnings-related pensions is based on two principles. The pay-as-you-go principle means that the contributions from currently employed individuals are used to finance the pensions of current retirees. The partial funding principle stands for investing a fraction of the insurance contributions. This decreases the amount of insurance contributions required to finance the pensions, as investment returns can be used to cover a part of the cost. [2]

Recently the pension expenditure has exceeded the total insurance contributions in both private and public sectors [3]. For instance, in the private sector in 2017 insurance contributions totalled 16.5 billion EUR, whereas the sum of pensions paid was 17.2 billion EUR [3]. As the number of pensioners in Finland increases and the working age population shrinks, the deficit will only increase. This emphasizes the importance of pension funds in the Finnish pension system.

Our client is the Varma Mutual Pension Insurance Company, which manages earnings-related pensions for private sector employees and self-employed individuals. Varma is responsible for the pensions of almost 900 000 Finns, making it one of the largest pension insurance companies in Finland. It is a mutual company owned by its clients: private companies, insured employees, self-employed persons and owners of guarantee capital. [4]

The 45 billion EUR investment portfolio of Varma consists mainly of equities (46% of the portfolio value in 2017), fixed-income investments (28%), real estate investments (8%) and hedge funds [4]. Investment returns are realized not only as changes in the value of the investments, but also as cash income in the form of dividends, interest payments and rents. Achieving good return on investments is of great importance to Varma, as the company uses the investment returns to cover the deficit between insurance contributions received and pensions paid [4].

1.2 Objectives

The key decision in managing the investment portfolio of Varma is the allocation of funds between equities and fixed-income investments, as these are the largest asset categories. The objective of our team is to use machine learning methods to make allocation decisions based on a large number of macroeconomic and financial market indicators.

The objective consists of three tasks. First, we study the interactions of the economic indicators by means of clustering and factor analysis. The results of this task could also be important in the next two tasks, which are the development of two models for performing allocation decisions. One model should allocate funds between stocks and model, and the other between short and long duration bonds. Time horizons of 3-6 months should be used in making the allocation decisions. Finally, the allocations performed by the models are compared against a static allocation where initial wealth is evenly divided between two assets.

2 Literature Review

2.1 Clustering

Clustering methods help in finding similar groups among a number of objects [5]. This facilitates analysis of large amounts of data, as more focus can be put in understanding the groups rather than the individual objects. For instance, clustering of stocks allows an analyst to focus on the behavior of groups of stocks instead of trying study each stock in isolation.

Typically the measure of similarity is some kind of distance metric. A very common distance metric is the Euclidean distance, which is straightforward to apply if the samples can be interpreted as independent and identically distributed random variables. However, time series are sequences of samples that are ordered by timestamps. Applying Euclidean distance can be difficult, for example in the case where time series are not of equal length.

Some common measures of distance include Dynamic Time Warping, correlation based distance and model based distance measures. Dynamic Time Warping is a similarity metric that performs alignment of two sequences. While it is great for detecting similar patterns in time series, the alignment procedure does not preserve temporal correlation structures. Correlation based distance does not suffer from this issue, as correlation captures the correspondence of two variables only if they move in the same direction at the same time. Model based distance measures are based on fitting the same model on each time series, and then using Euclidean distance as a measure of similarity in the parameter space. [5]

Two common methods for clustering are k-means and hierarchical clustering. k-means is an iterative algorithm that proceeds by assigning samples to the closest of k clusters, and then recomputing the cluster center as the mean of cluster members. Hierarchical clustering comprises of two variants: agglomerative and divisive clustering. In agglomerative clustering, each object is initially in its own cluster and two closest clusters are merged each round until all objects belong to a single cluster. Divisive hierarchical clustering proceeds the other way round, all objects are initially in one cluster and splits are performed iteratively. In k-means the number of clusters must be chosen a priori, whereas hierarchical clustering produces a dendrogram that can be used to determine a suitable number of clusters a posteriori, after the merges or splits performed at each iteration by the algorithm have been examined. [5]

2.2 Factor analysis

Factor analysis is used to describe observed variables as linear combination of some underlying factors or common factors [10]. Factor analysis also helps us to interpreter these factors and gives us estimate of strength and direction of the common factors on each of the measurement points.

One commonly used factor analysis method is principal component analysis (PCA). It uses an orthogonal linear transformation that transforms the original data into new coordinate system. The new coordinates or principal components are uncorrelated and the greatest possible variance lies on the first coordinate axis [7].

PCA can be used for many applications such as clustering and dimensional reduction. For clustering, we can analyze each observation in the first few axis of the new coordinate system and in this way investigate whether there is some clear clusters in fewer dimensions. We can also analyze the behaviour of different observed variables. For example, if two variables behave very similarly, the angle between the scores of these two variables should be very low in the plane of first two or more principal components.

PCA is most commonly used to reduce the dimensionality of data. This is based on the fact that the principal components are ordered in the way that the most of the variance lies on first component and least on the last component [7]. After this we can ignore the last components that only provide very little explanation on the variance of the original data.

2.3 Asset allocation, market timing and machine learning with financial data

The failure rate in quantitative financial machine learning is very high. Marcos López de Prado summarizes in his paper [11] common errors made by machine learning experts when they apply the methods on financial data. He also provides solutions to avoid these errors. He stated, for example, that one should use purging and embargoing to avoid cross validation leakage, use combinatorial cross validation to avoid limitations of walk forward backtesting, to model side and size of the allocation separately to reduce the complexity of the model and not to research trough backtesting in order to avoid finding any false strategies that lead to good result only in the model testing period. In this project, we followed many of his suggestions to avoid common problems that might lead to poor results.

However, it is certainly possible to generate trading strategies that produce positive returns on one's testing and model evaluation data, as showed by Hull [14]. It should be noted that he allocates between cash and SPDR S&P 500 ETF (SPY) and allows an allocation between -50% and 150% in SPY. This means that first and foremost he can choose to convert all his holdings to a risk free asset and that he allows for shorting and leveraging. We do not have an risk free asset to invest in and neither do we allow for shorting or leveraging, thus Hulls trading strategy is not completely compatible with Varma's case. In another of his papers Hull has described a model that allocates between 0% and 150% in SPY and the rest in US T-bills [15]. This situation closely resembles ours, but still leveraging was allowed. However, both of his strategies seem to work on paper, thus building an argument for the predictability of investment returns.

2.4 Model validation

Model validation is used to test whether the outputs of a model sufficiently correspond to the actual expected outputs. A common method of model validation for predictive models is cross-validation [6]. In general, cross-validation is performed by partitioning the available data into two subsets called the training and validation set. The training set is used to fit the model, and the performance of the model is tested on the validation set.

One of the most commonly used variations of cross-validation is k-fold cross-validation [8, pp. 241 - 249]. In k-fold cross-validation the data is randomly partitioned into k-subsets. The model is then trained using k-1 subsets, and tested on 1 subset. This is done k-times such that each subset is used once for testing. However standard k-fold cross-validation can not be used for time series data. Since time series data is usually correlated along the time-axis, randomly sampling data will cause highly correlated data to exist in the training and validation sets. This defeats the point of cross-validation, as we are in effect testing the model on data that we have trained it on.

This problem can be solved using a variant of k-fold cross validation called purged and embargoed combinatorial cross-validation. In combinatorial cross-validation the data is partitioned into N groups without shuffling. At each train-test iteration, k groups are chosen as the testing set, and N-k groups are used as the training set. This is repeated until all possible combinations of train-test splits have been used.

Each of our data points is associated with a prediction time and an evaluation time. At evaluation time, the results of a given action become known. This corresponds to the situation where if we invest in a given stock, the 20-day returns of that investment is only known after 20 days. Purging means eliminating from the training set all data points where the evaluation time overlaps with prediction times in the testing set. Embargoing solves the problem of correlation along the time-axis. It involves eliminating from the training set all data points that follow a testing set.

Purging and embargoing can thus be used to prevent information from leaking to the validation set. And combinatorial cross-validation provides a way to generate a large number of training-validation splits, to get a more accurate estimate of model performance.

3 Data & methods

3.1 Data validation and processing

Our dataset consists of 63 time series of macroeconomic and financial market indicators. The time series are of different frequencies, with daily, weekly and monthly series. They are also of different lengths, some starting as early as 1998 and some as late as 2011. Three of these time series can be considered our target variables, which depict the performances of global stock markets, global short- and long duration bonds.

During exploratory data analysis the data was visually inspected to identify outliers or changes in data reporting (i.e changes in the scale of survey questions). No outliers or changes were found.

Some time series did not contain data for some time periods, and were dropped from the dataset. Since the time-series were of differing lengths, some beginning as early as 1998 and some as late as 2011, we restricted the data set to series that had data starting from 2002. This left us with a dataset consisting of 59 time series out of the original 63.

To transform all the series to the same length, we mapped the weekly and monthly values to daily values. The dataset was augmented by adding the difference from moving average of 20 trading days for each daily time series. For the weekly and monthly time series, we added the difference between two consecutive values and mapped that to daily frequency.

3.2 Clustering of indicators & factor analysis

3.2.1 Principal component analysis

PCA transforms the original data x in to new observations y:

$$y = \Gamma^T(x - \mu) \tag{1}$$

where μ is the mean vector of x and $\Gamma^T\Sigma\Gamma = \Lambda = diag(\lambda_1\lambda_2, ...\lambda_p)$ where $\Gamma \in \mathbb{R}^{p\times p}$, $\lambda_n \geq \lambda_m$ when $n > m$ and Σ is the covariance matrix of x.

The components of y are called principal components. The variance explained by first m components can be calculated as sum of first m eigenvalues divided by sum of all eigenvalues:

$$\frac{\lambda_1 + \lambda_2 + ... + \lambda_m}{\lambda_1 + \lambda_2 + ... + \lambda_p} \tag{2}$$

We used PCA to have a deeper understanding on the behaviour of different indicators. This can be done by performing PCA for the standardized indicator data and plotting the score vectors of the indicators in the plain of two first principal components. In this way we, can easily see whether two indicators behave similarly in the two dimensional plain that explains as much variation as possible.

We also wanted to reduce the dimensionality of the indicator data to improve the performance of different machine learning models, because we did not have enough data to support very large number of dimensions and we wanted to reduce the noise in the daily indicator data. Also, if we are able to reduce the multicollinearity in the data, we can improve the interpretation of input parameters. This could be done by performing PCA for some cluster of indicators that is found using e.g. hierarchial clustering or selected in some other way. In this way, most information is still given as an input for the models, but we can make sure that there is less correlation between different variables. We choose the appropriate number of principal components to take into account by using Equation 2. We need to explain most of the variance, but we do not want to take new components that increase the explanation of variation for chosen components only by a little.

3.2.2 Hierarchial clustering

We need to select a suitable distance metric to perform cluster analysis, as discussed in the literature review. Model-based distance metrics are unsuitable in our case, since our data contains a range of different economic indicators. If all time series were representing similar data, such as stock prices, a model-based distance metric would be more sensible. Furthermore, Dynamic Time Warping is not ideal for our purposes either, since our primary interest is not to discover patterns that have occurred in different time series over time but to see which variables seem to move in the same direction at the same time. Thus, we use correlation-based distance, which is defined as follows:

$$d(x, y) = \sqrt{2[1 - \rho(x, y)]}, \tag{3}$$

where $\rho(x, y)$ is the Pearson's correlation coefficient between time series x and y. This metric has $d(x, y) = 0$ when $\rho(x, y) = 1$, whereas a negative correlation coefficient results in large distance.

As we do not know a priori the suitable number of clusters, we opt for agglomerative hierarchical clustering. In hierarchical clustering we also need to define a distance metric for two clusters, so that the two clusters that are closest to each other can be selected. We explore the results of two alternatives: average linkage and complete linkage.

Average linkage is the average pairwise distance of the between the objects of two clusters A and B:

$$d_{avg}(A, B) = \frac{1}{|A||B|} \sum_{x \in A} \sum_{y \in B} d(x, y). \tag{4}$$

Complete linkage is the maximum pairwise distance between the objects of two clusters:

$$d_{max}(A, B) = \max\{d(x, y) : x \in A, y \in B\}. \tag{5}$$

3.2.3 Cross correlation

When working with time series, an important component is the relative time displacement. One time series may precede another or be lagged behind it, this is especially the case with financial indicators that deal with the same underlying data. An example of two financial indicators that most likely have a lag based relationship are the number of building projects started and the number of building projects finished. Cross correlation can be thought of as lag based correlation, thus, it will contain the value for standard correlation at lag zero. Cross correlation is then used to determine the most statistically significant lag by choosing it so that the absolute value of the correlation is maximized. As a result, we may be able to extract time displaced features not known upfront, which may then be used in forecasting.

3.3 Predictive models

3.3.1 Logistic regression

Logistic regression is a variant of the generalized linear model for predicting a categorical response variable. Instead of predicting the value of a response variable directly, logistic regression models the probability of a particular response. In binary logistic regression, the response variable Y belongs to one of two categories encoded as 0 and 1. The logistic model defines the conditional probability of Y belonging to class 1 given the observed data:

$$P(Y = 1|\mathbf{x}) = \frac{1}{1 + e^{-w_0 - \mathbf{w}^T\mathbf{x}}} \tag{6}$$

where $\mathbf{x} \in \mathbb{R}^{d\times 1}$ is the vector of observed values of predictor variables, $\mathbf{w} \in \mathbb{R}^{d\times 1}$ is the vector of coefficients and w_0 is the intercept term. [8, p. 119]

The loss function to be minimized in fitting a logistic regression model is the logarithmic loss, given in [8, p. 120]:

$$L(w_0, \mathbf{w}) = -\sum_i (y_i \log p_i + (1 - y_i)\log(1 - p_i)), \tag{7}$$

where y_i is the true category of the i-th observation and p_i is the probability of class 1 given by Equation (6). We augment this error function with the penalty term:

$$R(\lambda, p, \mathbf{w}) = \lambda \sum_{i=1}^{d} |w_i|^p \tag{8}$$

For $p = 1$ the penalty term (8) corresponds to the Lasso penalty, which shrinks the coefficients and can set some of them to zero. For $p = 2$ it corresponds to the Ridge penalty that only shrinks the coefficients. [8, p. 72]

By applying a penalty term to the model coefficients, our goal is to decrease the prediction variance, while allowing the prediction bias to increase. Using cross-validation we can search for values of λ and $p \in \{1, 2\}$ for which this tradeoff increases the overall prediction accuracy.

3.3.2 Random forests

Decision trees are powerful, nonparametric hiearchical models for classification and regression problems. A decision tree maps the feature space into rectangular subspaces using simple rules. A simple model, typically a constant, is then fitted on the subspaces.

Trees can have extremely low bias, but high variance in their predictions. The motivation for random forests is to reduce variance by averaging the predictions of multiple trees that have differing internal structures. This is done by bootstrap aggregation. The data is resampled with replacement and each tree is trained on the resampled data, resulting in an ensemble of different learners. In addition, a subset of variables is randomly selected as candidates at each split of a tree. [8, pp. 587-598]

In the tree-building process, the splitting variable in each node is chosen by Gini index:

$$G_m = \sum_{k=1}^{N} \hat{p}_{mk}(1 - \hat{p}_{mk}), \tag{9}$$

where $\hat{p}_{mk}$ is the proportion of class k observations in node m and N the total number of classes. For two classes, the index is just $2\hat{p}_m(1-\hat{p}_m)$. The variable that has the largest decrease in Gini is chosen as the splitting variable. [8, p. 309]

3.3.3 k-nearest neighbor classifier

The k-nearest neighbor classifier differs from the previous two in the sense that no model is actually estimated. At prediction time, the k data points in the training set that are nearest to the query point are retrieved. The classification is done by majority voting. In this model, the main hyperparameter that needs to be optimized is the number of data points to retrieve, k. [8, pp. 463-465]

We use the Euclidean distance to determine the distance between data points. In addition, we weight the vote of each data point by its distance to the query point.

3.3.4 Rule based models

```
Function CalcReturns(AllocationDecisionFunction, targetVariables,
 dates):
    # CalcReturns is a function that returns a list of returns, which are
    # analyzed at a later stage.

    # AllocationDecisionFunction is a function that takes a date as input
    # and then returns a vector containing an allocation decision. The
    # length of this vector is exactly the length of targetVariables and the
    # sum of all of its elements is exactly 1.

    # targetVariables is a list of variables that is used to simulate an actual
    # investment action. The amount invested in each variable is defined
    # by AllocationDecisionFunction.

    # dates is a list of dates that defines the interval within which
    # the returns are calculated

    var returns = [ ]
    var previousDate = dates[0]
    var allocation = AllocationDecisionFunction(previousDate)
    foreach var date in dates do
        var newValue = 0
        var previousValue = 0
        for var i = 0 to length(targetVariables) - 1 do
            newValue += targetVariables[i][date] * allocation[i]
            previousValue += targetVariables[i][previousDate] * allocation[i]
        end
        returns.append(newValue / previousValue)
        allocation = AllocationDecisionFunction(date)
        previousDate = date
    end
    return returns
```

Algorithm 1: Pseudocode for the algorithm that computes the returns for a rule based model within a specific set of dates.

An interesting area of research is the comparison of machine learning methods with rule based methods. To achieve this we defined Algorithm 1 for the test set. By analyzing the order and the structure of the code one can conclude that the algorithm does not look in to the future, at least if we assume that the function AllocationDecisionFunction does not process data after the date that was passed as an argument to it. The advantages of rule based models compared to ML models is that the underlying logic is explicitly described. Quite often the term "black box" is used to describe complex ML methods that can not be easily inspected [9]. Term may, however, not accurately characterize all ML models, but still rule based models are generally easier to understand.

3.4 Modeling and benchmarking

We chose to model the problem as a binary classification task, where the goal is to predict whether stocks or bonds will have higher expected returns during the forecasting period. As discussed in Section 2.4, each data point is associated with a prediction time, and an evaluation time. The prediction time corresponds to the moment when an investment decision is made, and the evaluation time when the returns of the investment are known. The evaluation time for each data point is therefore n-days after the prediction time, where n is the length of the forecasting period. The models output a 0-1 class label and class probabilities, with 0 corresponding to stocks performing better during the forecast period.

The model used as input a further transformed dataset, which included the log-returns for indicators as well as the absolute values. For some indicators, the absolute value can offer more predictive information, and for trending indicators, the change can be more important, so both were included in the model. To reduce the effects of random variation, dimensionality reduction using PCA was implemented.

To optimize hyperparameters, we used purged and embargoed combinatorial cross validation with a total of 12 splits of which 10 were used to train the models, and 2 used for testing. Logarithmic loss (7) was used as the testing metric.

The performance of different models was tested against the performance of a static 50/50 allocations. The allocation decisions for the models were made using class probabilities. If the probability of an asset having higher returns was greater than 0.6, a 60/40 allocation was used, with 60% being allocated to the asset with predicted higher returns. Otherwise a 50/50 allocation was used.

To simulate the performance of models we used historical backtesting. Models were initially trained using 2000 business days worth of data. Predictions were then made for the next forecasting period, whereafter the model was retrained using more data. This was repeated until predictions were made for all periods following the initial training data. Performance was measured using mean log returns per forecast period, and their standard deviation.

4 Results

The original goal of the team was to develop market timing models for making allocation decision between stocks and bonds, and between bonds of short and long durations. The desired forecast horizon was 3-6 months. However, closer inspection of data revealed that market timing between long and short duration bonds would not be of great interest, since long duration bonds generally have higher returns than short duration bonds. Therefore we decided to focus on market timing between stocks and long duration bonds.

We also changed the forecast horizon, since with a forecast horizon of 60 trading days (around 3 months in calendar time) our models would constantly allocate more funds to stocks as opposed to bonds, essentially yielding a static allocation. This forced us to rethink our goals, and we decided to try performing allocation decisions with forecast horizons of 5 and 20 trading days. These models are presented in sections 4.2 and 4.3.

4.1 Clustering of indicators & factor analysis

4.1.1 Principal component analysis

The result from plotting scores of different indicators for monthly data in the plane of two first principal components can be seen in Figure 2. As stated in methods section, we can interpret the figure by studying the angles between the score vectors of different indicators. For example, from the figure, we can see that V2X and VIX indicators behave very similarly in the plane of first two principal components as they point almost in the same direction. If the angle between two indicators is less than $90°$, we say that they attract each other or behave in similar way.

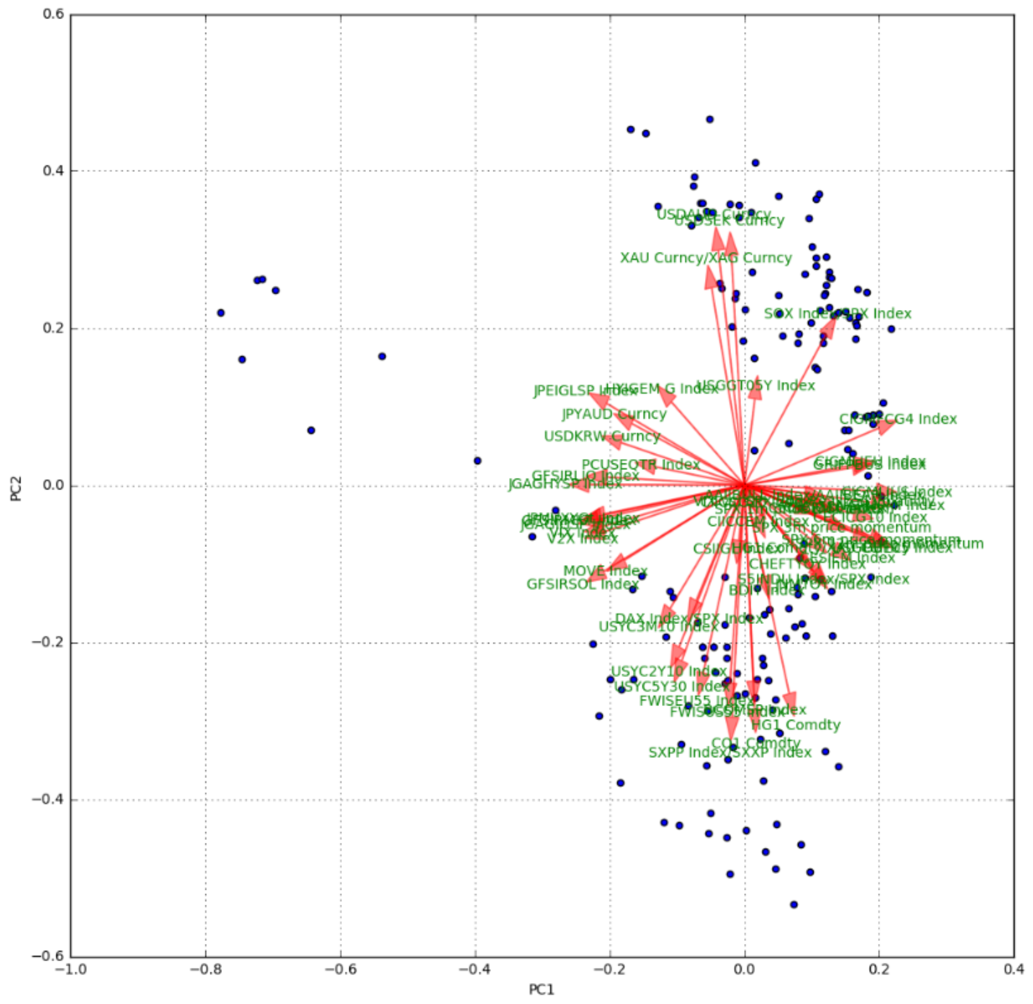

Figure 1: Scores of indicators for monthly data and monthly observations in the plane of two first principal components.

As we want to use both, the value of each indicator and change in value for each indicator, we divided the indicators into six groups to apply PCA for dimensional reduction: growth indicators, inflation indicators, volatility indicators and other three similar groups for change in value for indicators. We applied PCA separately for each of these groups because in this way we would be able to interpret the coefficients of different inputs given for the model better. Now, for example, if we got highest coefficient in some model for first principal component of growth indicators, it would already tell us a lot more than just the first principal component of all the indicators.

When we selected the appropriate number of components for inputs, we had two rules: At least 60% of the variation should be explained and principal components should be taken into account if it explained at least 6% of the total variation in the group of indicators. We also tried different approaches, but this seemed to give the best performance of models in our initial testing. We thus took 3 first principal components of volatility indicators, 4 of changes in values of volatility indicators, 4 of growth indicators and 5 of changes in values of growth indicators. For inflation indicators, we chose not to use PCA as it would have only reduced a few dimensions of the input data. This allowed us to reduce the number of input variables from 59 to 37.

4.1.2 Hierarchial clustering

The results of hierarchical clustering using average linkage are presented below in the form of a dendrogram in Figure 2. The dendrogram can be cut at any level to yield an appropriate number of clusters. However, our focus is mostly on understanding how the groups are formed rather than making a decision on how to partition the variables to a certain number of clusters.

We see that some indicators for market liquidity, volatility and risk sentiment form a cluster early on in the process. This cluster is the green one that is visible on the right. The risk indicator group is eventually merged with the US yield curve and some currencies, and the combined group is eventually merged with the other half of the indicators. In the other half, the large red group is formed by indicators that describe economic growth and investments. The other groups do not have such straightforward interpretation. The purple one consist mainly of inflation indicators, while the green one consists of S&P500 price momenta, the realized volatility of S&P500 against the VIX volatility index and a bull/bear sentiment ratio.

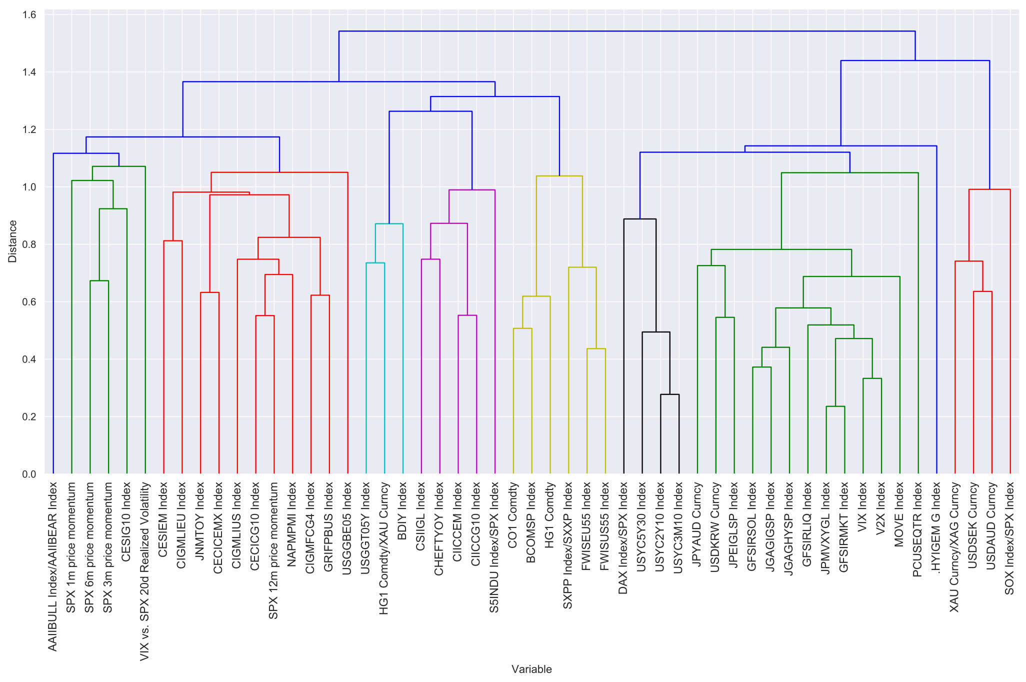

Figure 2: Dendrogram with average linkage.

The dendrogram that results from clustering using complete linkage is presented in Figure 3. For the most part the results are similar to Figure 2. On the left, the risk indicators again form their own group, which is later merged with a group that mostly contains currencies. The S&P500 price momenta form their own group again and so do the economic growth indicators and the inflation indicators. One major change from Figure 2 is that the US yield curve is now in a rather mixed group that is merged with growth indicators, instead of being merged with the risk indicators and currencies.

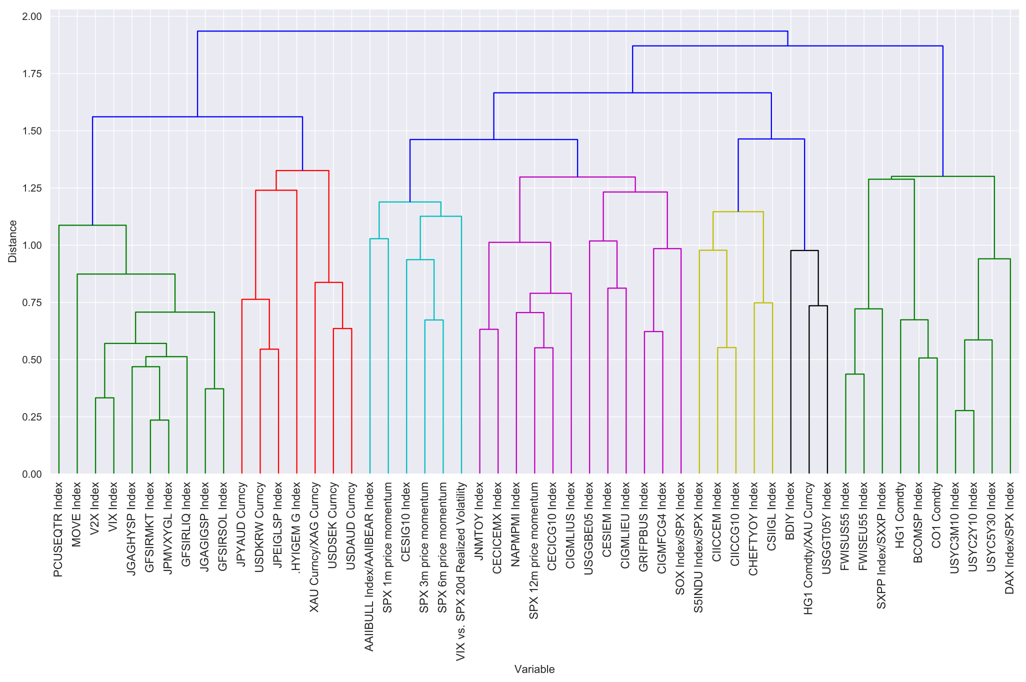

Figure 3: Dendrogram with complete linkage.

The hierarchical clustering results are quite robust to the choice of linkage criterion used. We identify two large groups in the dendrograms. One is formed by the market liquidity, volatility and risk sentiment indicators. The other contains various indicators related to economic growth.

4.1.3 Cross correlation

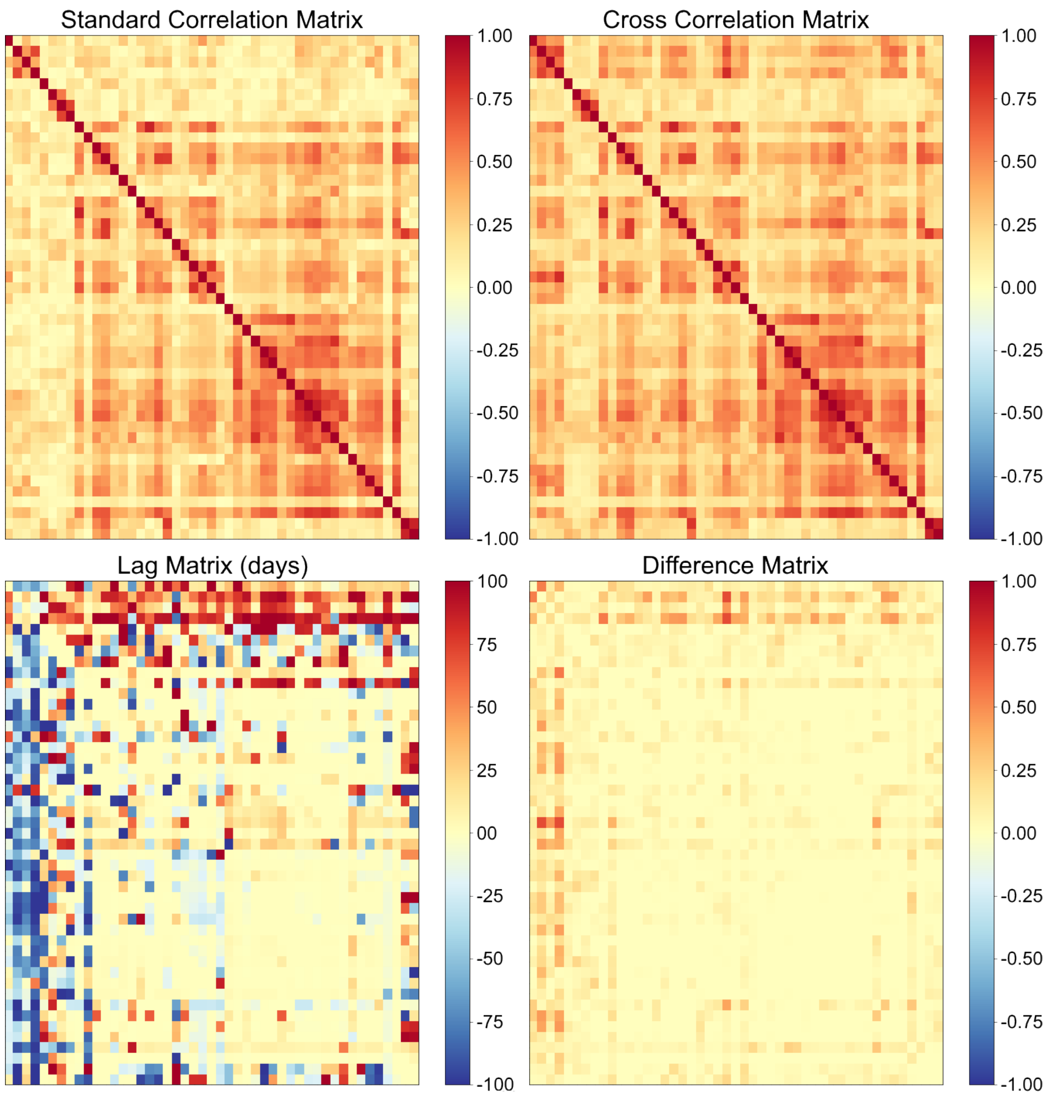

Figure 4: Comparison between standard correlation of returns and the maximum value found by cross correlation, accompanied by a plot of the most significant lag and by a plot that shows how much greater correlation could be found be introducing said lag. The indicator names are omitted from this plot, but can be found in the same order in Appendix A.

Figure 4 shows a comparison between standard correlation of returns and the maximum value found by cross correlation. All correlation values are plotted as the absolute values of the actual correlations, this was done to make the plot easier to understand. Figure 4 also shows the lag at which the most significant correlation was found and how much this differs from the standard correlation. The lag was constrained to an absolute maximum of 100 days, by allowing longer lags higher values for correlations might be found, but too large lags will probably find relationships that do not actually exist. When comparing the Cross Correlation Matrix to the Standard Correlation Matrix, we can clearly see an increase in correlation between variables, for example in the top left corner. This means that some variables are to some extent preceded or lagged by others. However, none of our target variables, the three last entries in each matrix, can accurately be represented by preceding data from other variables. The results are expected, because if this actually were possible, then the markets should quickly react to this relationship and ultimately the information would lose its value.

4.2 Predictive models

We evaluate the predictive accuracy of the models using two metrics. The first metric is classification accuracy, which is defined for a vector of predicted labels $\hat{\mathbf{y}}$ and a vector of true labels $\mathbf{y}$ as

$$A(\mathbf{y}, \hat{\mathbf{y}}) = \frac{1}{n} \sum_{i=1}^{n} I_{y_i = \hat{y}_i}, \tag{10}$$

where n is the number of elements in both vectors and I is the indicator function. While classification accuracy is easy to understand, it gives limited information on the predictive abilities of a classifier. For instance, a false classification is always penalized equally, regardless of how confident the classifier was. Therefore our main metric is the log loss that was defined for logistic regression in Equation (7). Here, we redefine the log loss as a prediction metric:

$$L(\mathbf{y}, \hat{\mathbf{p}}) = -\frac{1}{n} \sum_{i=1}^{n} (y_i \log \hat{p}_i + (1 - y_i) \log(1 - \hat{p}_i)), \tag{11}$$

where $\hat{p}_i$ is the predicted class probability for $P(y_i = 1)$.

To establish a baseline, we fit a naive classifier on the data and compare it against the other models. The naive classifier simply outputs the relative class frequencies so far observed as class probabilities.

The prediction metrics for market timing between stocks and long-duration bonds with forecast horizons of 5 and 20 trading are in Tables 1 and 2. The naive classifier achieves the smallest log loss and highest accuracy in both cases. This means that none of the three models have any predictive ability in determining which asset would have higher returns after 5 or 20 days. In both cases, logistic regression is the second best performing classifier. This because in cross validation best value for the penalty term λ was very large, meaning that the variables do not actually influence the predictions much and the model degenerates towards the naive classifier.

Model	Log loss	Accuracy
Naive classifier	0.6856	56.43%
kNN classifier	0.6956	53.27%
Logistic regression	0.6879	55.28%
Random forest	0.6941	55.68%

Table 1: Model metrics for a forecast horizon of 5 trading days, stock index against long-duration bonds.

Model	Log loss	Accuracy
Naive classifier	0.6716	61.07%
kNN classifier	0.6906	57.04%
Logistic regression	0.6762	59.54%
Random forest	0.6983	57.76%

Table 2: Model metrics for a forecast horizon of 20 trading days, stock index against long-duration bonds.

Overall, we can observe that the accuracies in Table 2 are higher. This is not surprising, as in the long term stocks are expected to have higher returns than the bonds, and all models are reflecting this to some extent. However, the difference in accuracy between the naive classifier and the models is larger for the longer forecast horizon.

4.3 Benchmarking

The average log returns per forecast horizon and their standard deviations for forecast horizons of 5 and 20 trading days can be seen in tables 3 and 4. Figures 5, 6, 7, 8, 9 and 10 show the portfolio performance of the different classifiers with different forecast horizons. The top plot shows the evolution of wealth during the testing period starting from with an initial investment of 1. The middle plot shows the portfolio value relative to the 50/50 allocation, and the bottom plot shows the allocation decisions.

For the 5 day forecast horizon the 50/50 static allocation achieves the best performance, having identical returns and standard deviation to logistic regression. Looking at Figure 7, we see that only 50/50 allocations are chosen. Meaning that with a forecast horizon of 5 days, the predicted class probabilities are all close to 0.5. kNN and random forest both perform worse than the static allocation, with Random forest having the worst returns as it is underperforming the static allocation over the entire test period. In addition, the random forest model has the highest standard deviation of returns.

Figures 5 and 9 suggest that the relative wealth of the portfolios increases when 60% of funds is allocated to stocks for longer periods of time. When more allocations are made for stocks for only a few periods, the results are less predictable, but more often leading to decreases in relative performance.

When allocations other than 50/50 are made only for a few periods, it is likely that the class probabilities exceed our 60/40 allocation threshold of 0.6 by very little since small changes in the inputs change to allocations back to 50/50. Meaning that a higher allocation threshold could lead to better performance.

As in model metrics defined in Tables 1 and 2, the models performed better when the forecast horizon was increased to 20 days. Random forest achieved the highest returns, followed by logistic regression, kNN and the static allocation performed the worst. Random forest had again the highest standard deviation of returns, whereas static allocation was the least risky in this sense.

Figures 6, 8 and 10 show that with a 20 day forecast horizon, all models allocate significantly more funds to stocks during the testing period. This can explain the increase in returns compared to the 50/50 allocation given the low classification accuracy of the models, since in the long term stocks give higher returns than bonds.

Allocation	Daily mean return	Yearly return	Standard deviation
50/50 static allocation	1.364 e-4	3.469 %	4.294 e-3
kNN classifier	1.269 e-4	3.223 %	4.412 e-3
Logistic regression	1.364 e-4	3.469 %	4.294 e-3
Random forest	1.156 e-4	2.932 %	4.460 e-3

Table 3: Performance metrics of portfolio log returns (forecast horizon 5 days).

Allocation	Daily mean return	Yearly return	Standard deviation
50/50 static allocation	1.660 e-4	4.237 %	4.283 e-3
kNN classifier	1.674 e-4	4.273 %	4.273 e-3
Logistic regression	1.688 e-4	4.310 %	4.310 e-3
Random forest	1.845 e-4	4.720 %	4.595 e-3

Table 4: Performance metrics of portfolio log returns (forecast horizon 20 days).

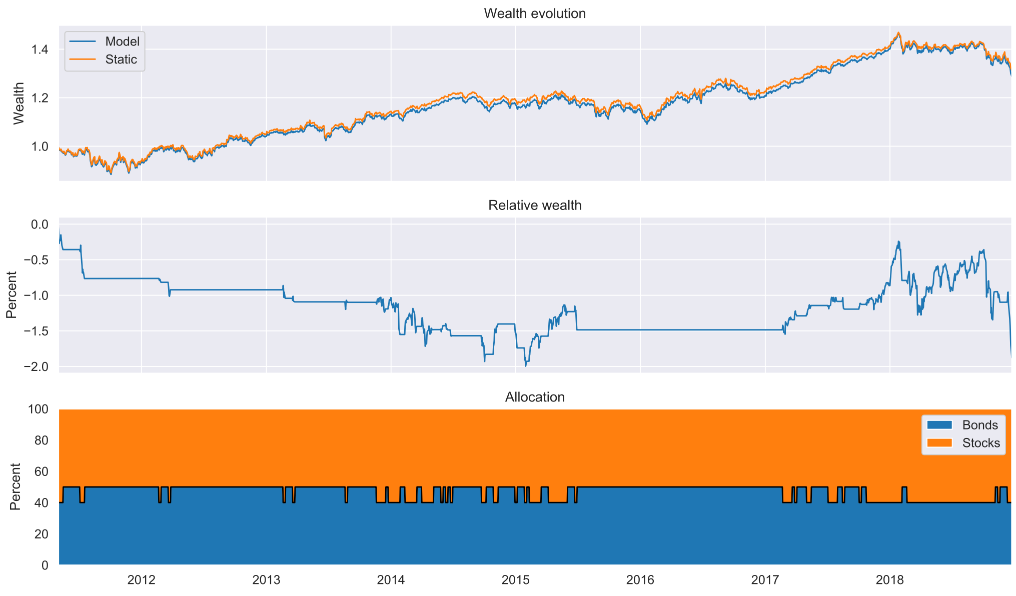

Figure 5: Portfolio performance for kNN classifier with a forecast horizon of 5 days.

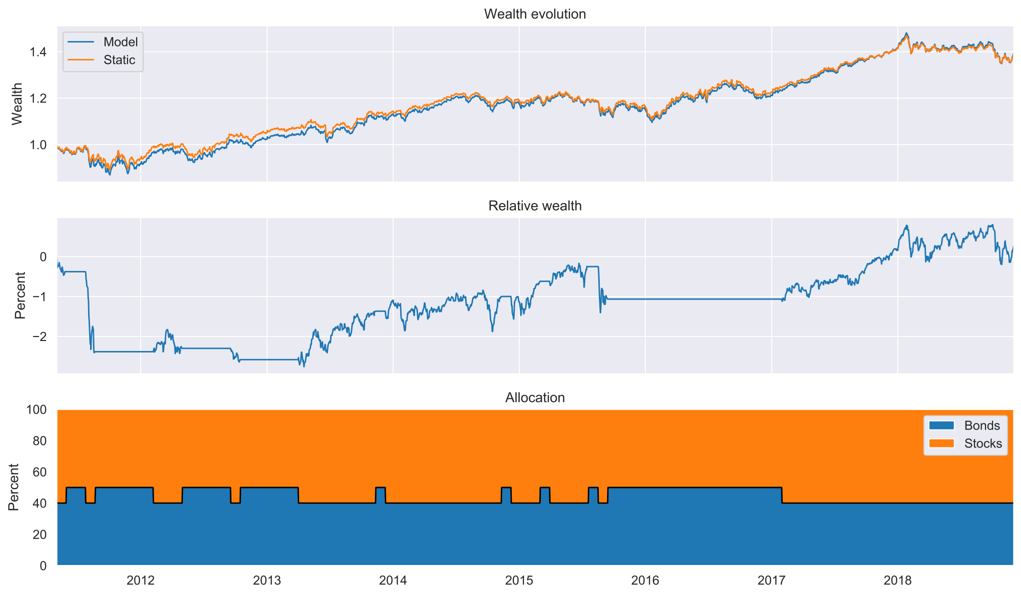

Figure 6: Portfolio performance for kNN classifier with a forecast horizon of 20 days.

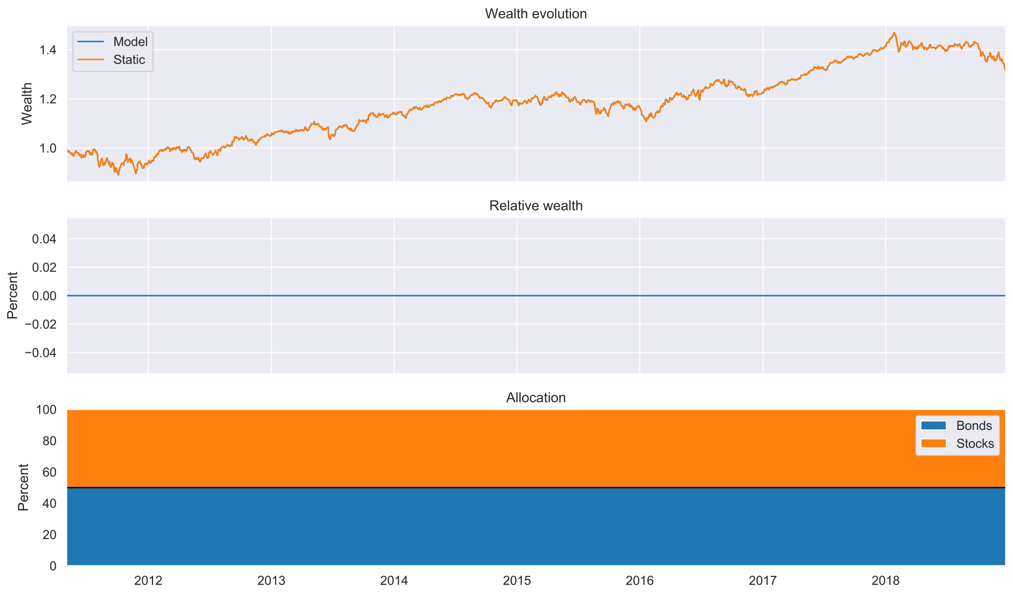

Figure 7: Portfolio performance for logistic regression with a forecast horizon of 5 days.

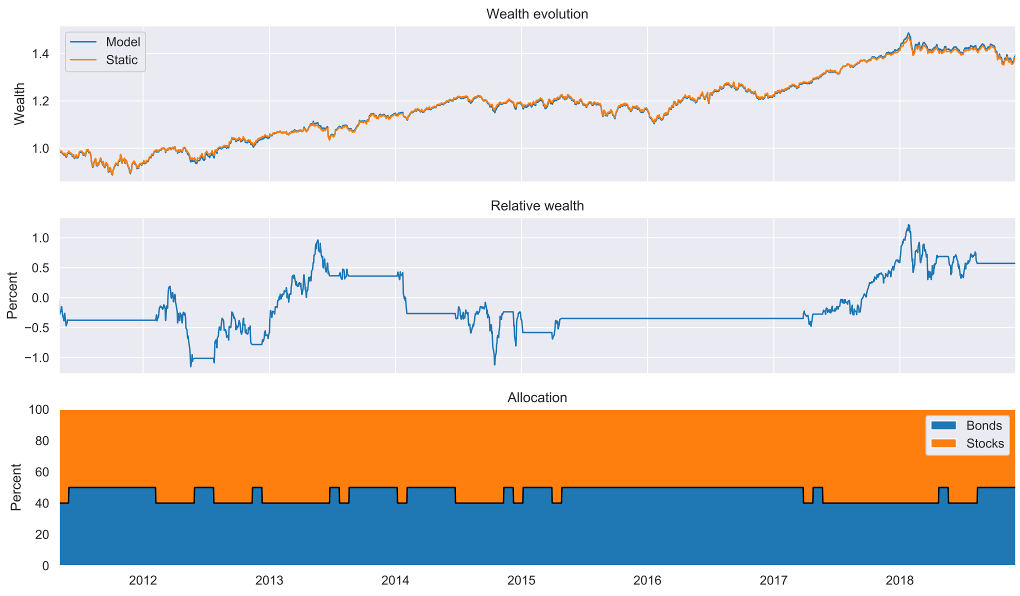

Figure 8: Portfolio performance for logistic regression with a forecast horizon of 20 days.

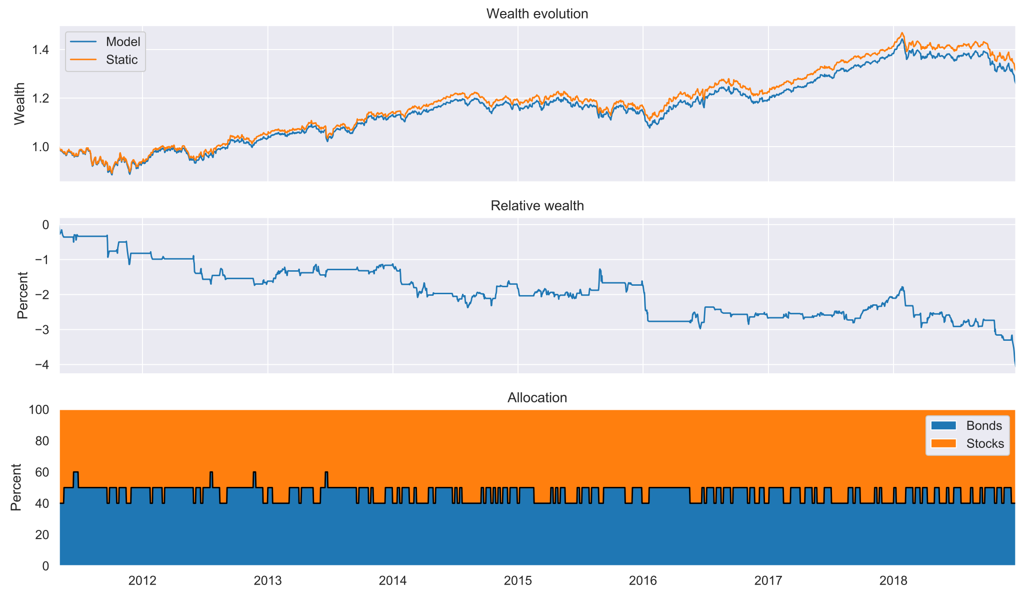

Figure 9: Portfolio performance for Random forest with a forecast horizon of 5 days.

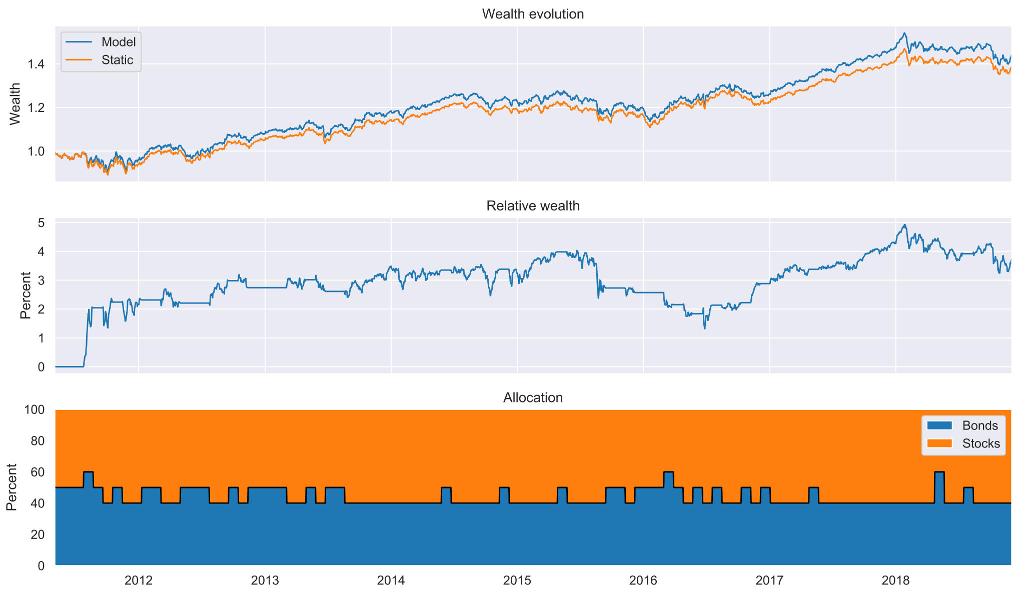

Figure 10: Portfolio performance for Random forest with a forecast horizon of 20 days.

4.3.1 Rule based models

```
**Function** PreviousDirectionStockIndex(date):

    # date is a date
    # StockIndex is a global stock index that can be accessed

    # StockIndex.GetIndex(date) is a function that returns the numerical
    # index for a date

    var dateIndex = StockIndex.GetIndex(date)
    if StockIndex[dateIndex] > StockIndex[dateIndex - 1] then
        # Invest everything in stocks
        return (100%, 0%)
    else
        # Invest everything in bonds
        return (0%, 100%)
```

Algorithm 2: Pseudocode for the algorithm that uses the previous change as an investment decision.

One of the simplest rule based models is to determine the direction of the previous change and make an allocation based on that. This model is described in pseudocode in Algorithm 2, note that the model can not be optimized because it does not contain any optimizable parameters. Thus, it can not possibly overfit, which is a desirable feature, furthermore, the algorithm can also be used to show that Algorithm 1 does not look in to the future. If the algorithm in fact would look in to the future, we would get a result where wealth would accumulate exponentially.

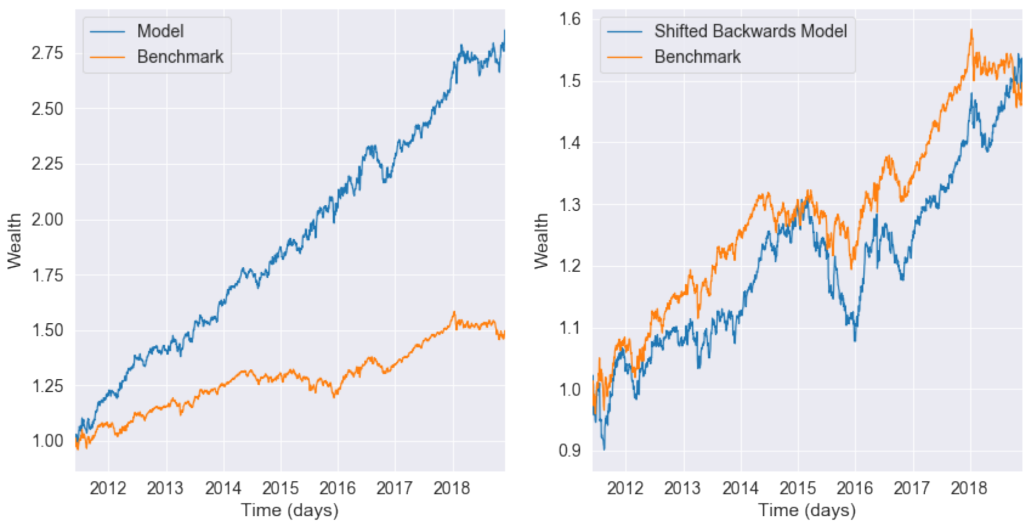

Figure 11: Quantifying the trading strategy defined by Algorithm 2 by running Algorithm 1 over the test dataset with a daily frequency. The "Shifted Backwards Model" uses the same trading strategy, but the decision been shifted one day backwards.

Figure 11 shows how one's wealth could have accumulated if Algorithm 2 had been implemented for trading assets, with a forecast horizon of one day. The figure also shows how the wealth would have grown if the investment decision was made one day earlier. Because the model that knows all information up until now performs significantly better than the one that knows one data point less, we can conclude that the newest information carries significant weight.

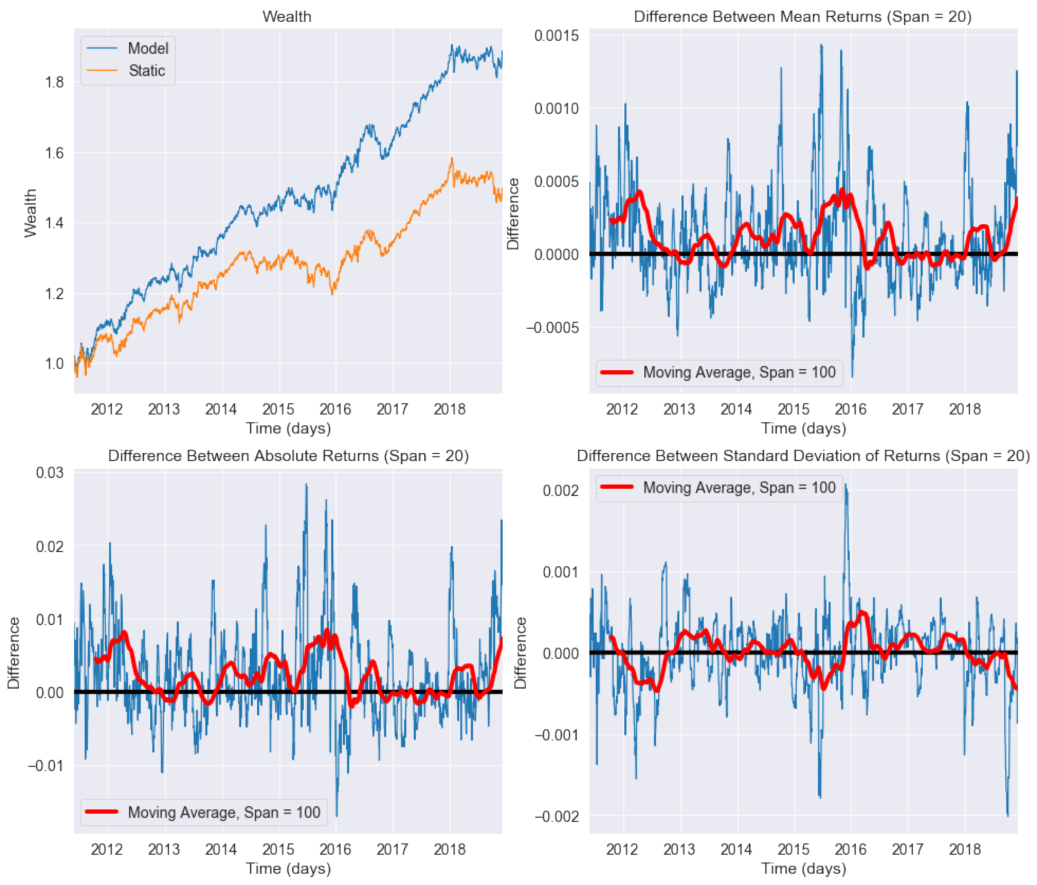

Figure 12: Refinement of Algorithm 2 so that it takes risk in to account in the investment decisions.

Figure 12 shows the performance of a risk constrained verison of Algorithm 2. This version of the model, which can be viewed in Appendix B, has optimizable parameters, which is why the model was first trained on a part of the data, however altering the training data did not in our case alter the optimal found parameters. The optimization had a constraint where the standard deviation of returns of the model had to be less than or equal to the standard deviation of returns of the benchmark. This solution is not re-optimized during the evaluation phase.

Figures with a long time interval can be hard to interpret, so to better evaluate the performance we first divided the data of both the benchmark and the model in to the following index sets: $[0, 1, 2, ..., s-1], [1, 2, 3, ..., s], ..., [n-s, n-s+1, n-s+2, ..., n-1]$, where s is the length of each divided set and n is the length of the entire data. We chose $s = 20$ (20 business days approximately represent one actual month), and thus we can compare how each period of length 20 compares itself across the benchmark and the model. The "Difference Between Mean Returns" and "Difference Between Absolute Returns" show that the model is on average better than the benchmark. On the other hand, the "Difference Between Standard Deviation of Returns" tells us that both have an approximately equal standard deviation. The overall performance metrics are in Table 5.

Allocation	Daily mean return	Yearly return	Standard deviation
50/50 static allocation	2.031 e-4	5.201 %	4.110 e-3
Algorithm 2	5.683 e-4	15.26 %	5.592 e-3
Refined Algorithm 2	3.335 e-4	8.694 %	4.083 e-3

Table 5: Performance metrics of rule based models (forecast horizon one day).

5 Discussion

5.1 Assessment of the results

The results show that with the given modeling choices and dataset, the performance of stocks and bonds can not be predicted. All of the implemented methods were less accurate than the naive classifier. This poor performance means that the results obtained in benchmarking should not be given much weight. During the benchmarking period stocks had 40% higher returns than bonds, meaning that a complete allocation to stocks would have had higher returns than any of the models or the 50/50 allocation. So the better returns can mostly be explained by the models investing more heavily into stocks, rather than the models ability to predict outcomes.

Importantly all of the models rarely performed 60/40 allocations in favor of bonds. Meaning that the models were not able to predict decreases in stock prices. Without the ability to predict decreases in stock prices, the models are of little worth. Since stocks generate higher returns in the long term, we would want the model to keep most of our wealth invested into stocks, and shift the weight into bonds only when we expect stocks to crash.

It is possible that our modeling process was inherently flawed, and we were using too much old data. If market dynamics change quickly, and we are training the models on older data, the models are being trained on data that is no longer relevant and learning outdated dependencies. This could be an interesting area of further research, to determine whether limiting the training data to say, the last n years could improve prediction accuracy.

Another possible reason for our poor results is that the data does not contain sufficient signals to predict the performance of stocks and bonds. Fixing this would require expertise of financial markets, to choose appropriate indicators. And some variables that affect the financial markets are hard to quantify and measure, such as the global political climate.

When considering the performance of Algorithm 2 as seen in Figure 11 it is important note that executing such a strategy could prove impossible in practice. The information required to make the allocation decision is only available at market closing. Thus, the allocation would have to be made during the next market opening, which can lead to much worse performance, as evidenced by when shifting the decisions back one day. The real returns of the model would likely be somewhere between Algorithm 2 trading strategy, and its shifted version. The change in information between market closing and the next days opening is unlikely to be as drastic as between consequent market closings. However, Hull does describe a strategy where orders are submitted 5 minutes before close to the market [14]. Assuming that the value of an asset does not change substantially during the last 5 minutes, Algorithm 2 could actually perform well. Still, we can not assume that Varma would be able to invest a significant amount of funds in a matter of minutes, thus the strategy may only be implemented on a smaller scale.

5.2 Reflection on literature

Our poor results were expected as there is not many publications in the literature about models that are able to overperform the market. Actually there is more literature about markets being efficient and that the prices should contain all the information available. Some say that markets can be inefficient only for short periods of time but our model can not take advantage of these short periods as the allocation horizon is very long in the scale of financial markets [12].

Also if there is a strategy or an indicator that could be utilized to have better expected returns than the market with the same level of risk, the other players in the market would start to mimic the strategy. This would lead to the performance of the strategy decreasing and therefore stable forecasting patterns or strategies are unlikely to persist for long periods of time in the financial markets [13]. This can be one reason for why widely followed indicators did not seem to give any significant predictive information.

Some strategies seemed to work and outperform the market like the one proposed by Hull and Qiao [14]. However, after the inception of ETF based on this or similar strategy, the market has clearly outperformed the ETF. This can be seen from figure 13 as S&P 500 has grown over 40% as meanwhile Hull Tactical US ETF has grown less than 20% starting from the inception of the ETF. This might be due to changing market environment or due to other investors starting to observe similar indicators while making their investment decisions.

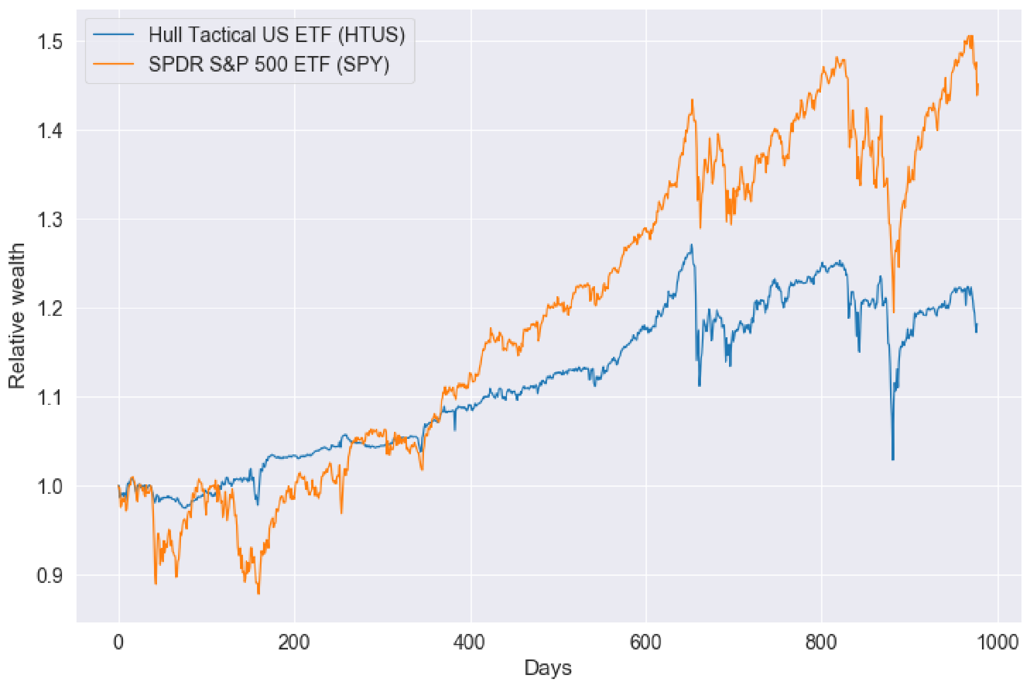

Figure 13: Relative wealth of Hull Tactical US ETF compared to SPDR S&P 500 ETF starting from end of June 2015.

6 Conclusions

During this project we had three objectives: clustering of financial indicators, creation of predictive models that dynamically allocate between assets and benchmarking these models against a static 50/50 allocation. The benchmarking was supposed to be evaluated against global stock and both long and short bond indexes, but the short bond indexes were not extensively analyzed, due to them being generally outperformed by the long term bonds.

To cluster the financial indicators that were available we used hierarchial clustering using a distance metric derived from correlation between two indicators. Additionally, we reduced the dimensions of the data with principal component analysis and found lag based relationships with cross correlation. An interesting avenue of future research could be to evaluate how different distance metrics could change the results of clustering.

When trying to predict the future, we created a modelling philosophy that was informed by previous research within this subject. We tried combinatorial cross validation, introducing moving averages, reduction of dimensions and implemented three machine learning methods: logistic regression, k-nearest neighbors and random forests. All of these models failed to significantly beat the static allocation with forecast horizons 5 and 20 days.

Furthermore, we created and tested a few rule based models, of which some performed exceptionally well. The parameterless model described by Algorithm 2 generates an absolute return of over 12 times greater (for the entire dataset) than that of the static 50/50 allocation, with only a slightly higher volatility. When constraining volatility to the same level as the 50/50 allocation the return was over 2.5 times higher. It must however be noted that the strategy may not be implementable in practice, because the model assumes that one can always execute trades at the closing price of each day, which is generally an idealization.

Another weakness in all of our models is that none of them account for the costs caused by slippage, taxes and transactional fees, thus, the actual performance should be assumed to be even worse. However, to what extent these affect a financial institution like Varma is not known by us.

All in all the project was completed successfully. The experts at Varma were at least interested in the results produced by the clustering of regimes, the hierarchical clustering of time series and the concept of cross correlation to determine lags. We were also able to create seemingly functional rule based models and even though the machine learning techniques did not produces useful results that could be implemented as investment strategies, the concepts that did not work with the given are now known. This is in itself already a significant piece of knowledge and could in the future be used to guide research in the area of financial forecasting and modeling.

7 Appendix

A Indicators in the Cross Correlation Figure (Figure 4)

```
01. CESIG10 Index           |   37. CDX HY CDSI GEN 5Y PRC Corp
02. CECICG10 Index          |   38. ITRX XOVER CDSI GEN 5Y Corp
03. CESIEM Index            |   39. .HYIGEM G Index
04. CECICEMX Index          |   40. SPX 1m price momentum
05. USGGT05Y Index          |   41. SPX 3m price momentum
06. USYC3M10 Index          |   42. SPX 6m price momentum
07. USYC2Y10 Index          |   43. SPX 12m price momentum
08. USYC5Y30 Index          |   44. VIX vs. SPX 20d Realized Volatility
09. HG1 Comdty              |   45. NDUEACWF Index
10. S5INDU Index/SPX Index  |   46. LG13TRUU Index
11. USDKRW Curncy           |   47. LG71TRUU Index
12. USDAUD Curncy           |
13. XAU Curncy/XAG Curncy   |-----------------------------------------
14. SOX Index/SPX Index     |   This list contains only those indicators
15. DAX Index/SPX Index     |   with a daily frequency. The cross
16. SXPP Index/SXXP Index   |   correlation would not have been able to
17. HG1 Comdty/XAU Curncy   |   determine the actual lags if a lower
18. JPYAUD Curncy           |   frequency would have been used.
19. USDSEK Curncy           |
20. FWISUS55 Index          |   The indicators place themselves in
21. FWISEU55 Index          |   the matrices as follows:
22. CO1 Comdty              |       ________________________
23. USGGBE05 Index          |    1. |___|___|___|___|___|___|
24. BCOMSP Index            |    2. |___|___|___|___|___|___|
25. BDIY Index              |    3. |___|___|___|___|___|___|
26. PCUSEQTR Index          |   ... |___|___|___|___|___|___|
27. GFSIRMKT Index          |   46. |___|___|___|___|___|___|
28. GFSIRLIQ Index          |   47. |___|___|___|___|___|___|
30. VIX Index               |         1.  2.  3. ...  46. 47.
31. V2X Index               |
32. MOVE Index              |
33. JPMVXYGL Index          |
34. JGAGIGSP Index          |
35. JGAGHYSP Index          |
36. JPEIGLSP Index          |
```

B Refined Algorithm 2

```
Function PreviousDirectionStockIndexRefined(date,
 maxAllocationSize, riskAdjustment, lookbackWindow):
    # date is a date

    # maxAllocationSize is a number within the range [0, 1.0] and
    # determines the maximum allocation in an asset

    # riskAdjustment is a number within the range [0, 1.0] and
    # determines how much we prefer the bonds over the stocks

    # lookbackWindow defines how many days are looked back
    # when assessing the risk

    # StockIndex is a global stock index that can be accessed

    # StockIndex.GetIndex(date) is a function that returns the numerical
    # index for a date

    # StockIndex.GetRollingStd(lookbackWindow) is a function that
    # returns the rolling standard deviation with a given window

    var distributableAllocation = 1.0
    var dateIndex = StockIndex.GetIndex(date)
    var rollingStockStd = StockIndex.GetRollingStd(lookbackWindow)

    if rollingStockStd(dateIndex) > rollingStockStd(dateIndex - 1) then
        distributableAllocation -= riskAdjustment

    if StockIndex[dateIndex] > StockIndex[dateIndex - 1] then
        var inStocks = distributableAllocation * maxAllocationSize
        var inBonds = (1.0 - distributableAllocation) +
         distributableAllocation * (1.0 - maxAllocationSize)
        return (inStocks, inBonds)
    else
        var inStocks = distributableAllocation * (1.0 - maxAllocationSize)
        var inBonds = (1.0 - distributableAllocation) +
         distributableAllocation * maxAllocationSize
        return (inStocks, inBonds)
```

References

[1] Finnish Centre of Pensions. *Pensions part of social security.* Jul. 3, 2018. [Online]. `https://www.etk.fi/en/the-pension-system/pension-security/pension-as-social-security/`. Accessed Feb. 23, 2019.

[2] Finnish Centre of Pensions. *Principles of pension financing.* Jan, 22, 2019. [Online]. `https://www.etk.fi/en/the-pension-system/pension-financing-and-investments/financing-principals/`. Accessed Feb. 23, 2019.

[3] The Finnish Pension Alliance. *Circulation of pension money.* [Online]. `https://www.tela.fi/en/circulation_of_pension_money`. Accessed Feb. 23, 2019.

[4] Varma Mutual Pension Insurance Company. *Varma annual and CSR report 2017.* 2018. [Online].

`https://www.varma.fi/globalassets/vuosikertomus/varmas-annual-and-csr-report-2017.pdf`. Accessed Feb. 23, 2019.

[5] Aghabozorgi, S., Shirkhorshidi, A. S., and Wah, T. Y. *Time-series clustering - A decade review.* Information Systems 53, 2015, pages 16 - 38.

[6] Kohavi, R. *A study of cross-validation and bootstrap for accuracy estimation and model selection.* Morgan Kaufmann, 1995, pages 1137 - 1143.

[7] Jolliffe, I. *Principal Component Analysis.* Springer Berlin Heidelberg, 2011.

[8] Hastie, T., Tibshirani, R., and Friedman, J. *The Elements of Statistical Learning - Data Mining, Inference, and Prediction* 2nd edition. Springer-Verlag, New York. 2009

[9] Card, D. *The "black box" metaphor in machine learning* Jul 5, 2017. [Online]. `https://towardsdatascience.com/the-black-box-metaphor-in-machine-learning-4e57a3a1d2b0`. Accessed May 8, 2019.

[10] Fabrigar, L. R., Wegener, D. T. *Exploratory Factor Analysis* Oxford University Press, New York, 2012

[11] de Prado, M. L. *The 10 Reasons Most Machine Learning Funds Fail* Journal of Portfolio Management 44, Number 6, 2018, pages 120-133

[12] Malkiel, B. G. *The Efficient Market Hypothesis and Its Critics* Journal of Economic Perspectives 17, Number 1, 2003, pages 59–82

[13] Timmermann, A., Granger, C. W. J. *Efficient market hypothesis and forecasting* International Journal of Forecasting 20, 2004, pages 15–27

[14] Hull, B., Qiao, X. *A Practitioner's Defense of Return Predictability* Available at SSRN 2609814, 2015

[15] Hull, B., Qiao, X., Bakosova, P. *Return Predictability and Market-Timing: A One-Month Model* Journal Of Investment Management, Forthcoming

Self Assessment

Plan vs. Reality

The project was well scoped from the very beginning as Varma had provided us clear tasks to fulfill in order to reach the final result. Because of this the initial schedule and scope did not change during the project for the most part. The scope was appropriate for the length of this project, and we experienced the workload to not be too large even most of our team members were working part or full time during this project.

Small changes in the schedule were due to the initial schedule having too much time allocated for different machine learning methods. Instead, we used a large amount of time to develop an appropriate modelling process. After we had chosen the way to model the market timing, it was easy and fast to try out how different methods performed in the model. In the end we also chose to put more emphasis on making allocation decisions between stocks and bonds as we noticed that during almost all 60 day periods in the past few years, the long term bonds outperformed the short term bonds. Therefore, the model that puts more weight on long term bonds would always beat the static allocation.

We were able to avoid most of the risks listed in the project plan. All of the team members were active during the whole project, communication with the client was active, we got the support we needed from the client team and there were no major data related issues. However, as stated in conclusions the model did not seem to give outstanding results and further research should be done before making any financial decisions using our approach. This was expected as we listed the probability of the model failing in performance benchmark to be high in the beginning of this project.

Project Success

Even though our results were not significant, the project still was very successful in many ways. We were able to provide Varma interesting insights from the clustering part of the project. Also, they got the source code for our modeling part, so they can seek better results using our approach with different set of indicators or with different methods in case they want to investigate our model further.

Our team members learned a lot about different concepts in financial markets and about the operations of mutual pension insurance companies. We were also able to deepen our knowledge on different machine learning methods by applying those to real life problems. This project also taught us many project working skills, such as communication with the client, how to schedule the project and how the possible risks should be taken into account. These skills will help us to work in groups and plan our possible future projects better.

What Could Have Been Done Better

As our final model did not work as well as we would have wanted, we could have done many things better during this project. In the beginning of the project, if we would have allocated even more time for the modeling part, we could have had enough time to try out few different modeling approaches and maybe even include some other models such as neural networks and gradient boosting to model the market timing. We also could have done wider literature survey to find different approaches for market timing problem as we only used few sources to justify our current approach. However, it was very difficult to find well performing models in the literature and even finding the papers we used was not easy.

The teaching staff and the client gave us the support we needed during the course, which is why we think that there is not much that could have been done better by them.