ERASMUS UNIVERSITEIT ROTTERDAM

Erasmus School of Economics

Bachelor Thesis Quantitative Finance

Analysing the Predictive Power of various Regression Tree Ensembles for Asset Pricing

Author:
Maud van Lent

Student ID:
483143ml

Supervisor:
Terri van der Zwan

Second Assessor:
Michel van der Wel

July 4, 2021

Abstract

One of the most widely studied problems in financial economics is empirical asset pricing. Recent additions to literature still incorporate traditional methods for stock return forecasts (e.g. Fama and French (2015); Lewellen (2014); Koijen and Van Nieuwerburgh (2011)). It is, however, shown that the use of machine learning methods could resolve problems that arise in these traditional methods (Gu, Kelly, and Xiu, 2020). In this research, a comparative analysis is performed to analyse the predictive power of modifications of the random forest model. We investigate random forest models, extremely randomised forest models, AdaBoosting models and histogram-based gradient boosting models. We find positive forecasting results for the random forest model and the extremely randomised forest model. Additionally, we find that for these two models momentum and return volatility are found to be driving factors in the model estimations.

The views stated in this thesis are those of the author and not necessarily those of the supervisor, second assessor, Erasmus School of Economics or Erasmus University Rotterdam

Contents

1 Introduction

Empirical asset pricing is a widely studied financial problem. This is substantiated by the fact that it forms the basis for trading strategies and is, therefore, a highly relevant subject for financial economics. For this purpose equity risk premium forecasts are to be obtained, for which methods and influential factors are extensively examined in existing literature. In his paper on views of financial economists, Welch (2000) even alleges equity risk premium to be "perhaps the single most important number in financial economics". Two prediction methods come forward in the existing empirical literature for return forecasts. These are the cross-sectional regression (Fama and French (2008); Lewellen (2014)) and time-series regression (Welch and Goyal (2008); Koijen and Van Nieuwerburgh (2011); Rapach and Zhou (2013)). However, due to the inability of these models to capture complex nonlinear relations and other shortcomings, renewed types of prediction models are examined. Since machine learning methods are effectively used in predictions due to high flexibility, dimension reduction possibilities and the application of efficient algorithms allowing for nonlinearity, these methods are expected to resolve the problems that occur in traditional asset pricing. The application of machine learning models for the problem of predicting stock returns has proven to be advantageous (Harvey and Liu (2021); Giglio and Xiu (2021); Kelly, Pruitt, and Su (2019); Moritz and Zimmermann (2016)). In their comparative analysis, Gu, Kelly, and Xiu (2020) find random forest and neural network algorithms to outperform a wide range of machine learning methods.

The objective of this paper is to further extend the research on stock return forecasting using machine learning methods. This is done by combining existing literature on enhancements of regression tree models with widely studied stock return predictor variables, to enhance the comparative analysis commenced by Gu, Kelly, and Xiu (2020). We construct a random forest model, extremely randomised models, AdaBoost models and histogram-based gradient boosting models and subsequently analyse their predictive performance.

We find extremely randomised forests to outperform the random forest model. While these results are insignificant, this instigates the belief that further optimisation of the included parameters produces substantially improved return forecasts.

In this paper, we employ the following structure. In Section 2, an overview of the existing literature on empirical asset pricing is presented, substantiated by improvements that result from the use of machine learning in financial economics. Thereafter we elaborate on the choice of predictive features to incorporate in the construction of the forecasting models in Section 3. Subsequently, in Section 4, the proposed models for the comparative analysis are explained in detail. This is followed by a quantitative and qualitative analysis of the produced forecasts in Section 5. Section 6 concludes the findings. Lastly, a discussion with possible improvements of our research is presented in Section 7.

2 Literature

Asset pricing has long been an important part of financial economics. To this purpose the systematic risk of stock returns is estimated. Equity risk premium is defined as the conditional expected stock returns on top of the risk free rate obtained from government bond prices. This risk premium compensates the risks associated with investments in stocks (Best and Byrne, 2001). Understanding the behaviour of risk premiums is of major importance to asset pricing. Expected returns are often divided into a systematic part, the risk premium, and an idiosyncratic part. The behaviour of expected returns, however, is difficult to predict since the predictable part, the risk premium, is obscured by the idiosyncratic part. The variation in the stock returns is mostly influenced by unpredictable news, overshadowing the risk premium. Affirmatively, Lau, Ng, and Zhang (2012) found risk premium to be highly dependent on the accessibility of information.

A multitude of asset pricing models and theories have been proposed to model the asset pricing (e.g. ICAPM, Merton (1973), LAPM, Holmström and Tirole (2001), X-CAPM, Barberis et al. (2015), APT, Ross (1976)). In alike manner, much research has been done to uncover the predictive variables involved in the cross-sectional variation in expected stock returns.

In traditional methods for stock return prediction linear factor models are employed. Collot and Hemauer (2021) propose two main methods that are used to identify factors and their respective loadings. For the first method, the two-pass regression, firstly, a time-series regression is estimated. Examples of a time-series regression are the three-factor model (Fama and French (2021a), Fama and French (2021b)), the four-factor model (Carhart, 1997) and the five-factor model (Fama and French, 2015). Subsequently, a cross-sectional regression is implemented which regresses the average excess returns of the test assets on the estimated factor loadings. A commonly used variation of this method is the Fama-MacBeth two-stage approach (Fama and MacBeth, 1973).

In the second method, used in recent empirical asset pricing, a stochastic discount factor is modelled, which is expressed as a linear combination of the potential factors. Examples of this method are given by Cochrane (2009) and Jagannathan and Wang (2002).

Recent publications in the field of empirical asset pricing still make use of traditional methods. Fama and French (2008) and Lewellen (2014), for example, make use of cross-sectional regressions of future stock returns on several lagged variables. There are, however, several drawbacks to the use of linear factor models. A prominent issue is the omitted-variable bias. This occurs when an explanatory variable, that is correlated with the dependent and other explanatory variables, is excluded from the model. To avoid this problem of omitted variables, recent surveys have documented large sets of potentially correlated variables. Subrahmanyam (2010) finds 50 earning-based predictor variables, McLean and Pontiff (2016) find another 82 and Harvey, Liu, and Zhu (2016) and Green, Hand, and Zhang (2013) even further increase this number to around 330. However, simply including all potential predictor variables raises the problem of overfitting. Next to that, prediction and determining the functional form can be problematic as is mentioned by Keim and Stambaugh (1986), Pesaran and Timmermann (1995), Torous and Valkanov (2000) and Welch and Goyal (2008). Machine learning can resolve these issues as it is flexible and allows for nonlinear non-predetermined relations.

The use of machine learning methods for financial problems has grown considerably (Hull, 2021). Machine learning could resolve issues in traditional asset pricing (Gu, Kelly, and Xiu, 2020). Firstly, because machine learning methods produce a diverse set of high-dimensional models that are effectively used in predictions due to high flexibility. Secondly, machine learning employs efficient algorithms for selection of model specifications from a vast set of options. Furthermore, compared to traditional methods, it allows for a much larger set of predictive variables. Notably, due to a focus on variable selection, dimension reduction and the mitigation of overfit through regularisation, machine learning techniques are even able to handle predictive variables which are highly correlated. Lastly, the distinguishing flexibility takes into account complex associations and nonlinearities present in functional forms that translate predictive variables to risk premium approximations. Hence, by employing machine learning methods in the field of empirical asset pricing we aim to produce significant and economically relevant advancements to the existing literature.

In his paper, Weigand (2019) specifically elaborates on the use of machine learning in asset pricing. He underlines theoretical insights and empirical results obtained from recent academic studies. Based on which he concludes machine learning applications to be profitable in the field of empirical asset pricing. Findings by Moritz and Zimmermann (2016) affirm this idea. They produce a tradings strategy that has an information ratio which is about three times higher compared to linear models that do not account for nonlinearities, like the Fama-Macbeth two-stage approach.

In their comparative analysis on machine learning methods for asset pricing, Gu, Kelly, and Xiu (2020) recognise regression trees and neural networks to be the best performing methods. In general, neural networks perform well on complex machine learning problems due to their flexibility. This flexibility is caused by the possibility to incorporate many telescoping layers and complex multivariate nonlinear functions, proving this method to be successful for a wide variety of applications. Hornik, Stinchcombe, and White (1989) argue that multilayer feedforward networks are capable of approximating nearly any function, mapping one finite dimensional space to another. The use of neural networks is, however, discouraged by many economists. This originates from the lack of transparency, impeding the interpretation of economic effects and variable relations, which is the main drawback in machine learning for financial problems. The complexity of neural networks impedes the transparency of the model, causes the model to be difficult to interpret and results in a highly parameterised machine learning method. Also Weymaere and Martens (1991) note several drawbacks to the use of neural networks. For this method long training times are required. Also the exact numbers of units in hidden layers and the number of hidden layers should be known in advance to solve the problem without deteriorating the performance of the network. For the optimal performance, several networks need to be trained in order to find the perfect configuration.

The use of regression trees induces advancements in transparency and so in the interpretation of model performance, compared to neural networks. Therefore, in this research we further elaborate on the possibilities in the field of regression trees.

Regression trees are popular machine learning methods (Gu, Kelly, and Xiu, 2020). In these types of models it is possible to take account of multivariate functions for predictors. Especially applications of regression trees in large-scale problems significantly improve the accuracy and usefulness of results. Furthermore, the problem of overfitting is resolved by model-averaging, since only subsets of the data and predictor variables are incorporated in the model (Moritz and Zimmermann, 2016).

In their paper, Gu, Kelly, and Xiu (2020) examine boosted tree and random forest algorithms. Aside from neural networks, these tree selection methods, especially random forest, produce desirable results. In research done by Medeiros et al. (2021) the random forest model is found to produce the most accurate results in forecasting inflation. They also make use of deep neural networks, instigating the believe that making improvements to the random forest model yield results that equate the results found for the neural network models in terms of asset pricing. Therefore, apart from the random forest model, proposed by Breiman (2001), three other forest models are examined. The objective is to provide further insights in the field of regression trees to enhance the research on optimal modelling using machine learning. Subsequently, it aims to obtain better results in the field of machine learning for statistically optimal forecasts and large economic gains in regard to asset pricing.

Gu, Kelly, and Xiu (2020) have found "shallow" learning to outperform "deep" learning when predicting asset prices. In the field of regression forests this means trees with few leaves outperform larger tree depths. This can be explained by a small signal-to-noise ratio in the prediction data, since, compared to nonfinancial implementations, their data exhibits considerably weaker signals. Hence, the strength of machine learning in their appliance lies in averaging over shallow randomised trees. For this reason, the method of extremely randomised forest, introduced by Geurts, Ernst, and Wehenkel (2006), is examined. This extension on a regular random forest algorithm makes use of randomly selected threshold values, in addition to random bootstrap sampling. The use of randomised selection over optimisation drastically lowers the computation time allowing for model estimation using large datasets. As early as 1989, a study by Mingers (1989) has shown randomised trees to preform as well as the classical models. Compared to state-of-the art randomised regression models, Geurts, Ernst, and Wehenkel (2006) have found extremely randomised trees to produce competitive results in regard of accuracy and computational efficiency.

A differentiation from bagging algorithms are boosting algorithms. Drucker (1997) has found boosting to produce the same or improved results in terms of prediction error. The AdaBoost algorithm, short for adaptive boosting algorithm, introduced by Freund and Schapire (1997), makes use of weights to sequentially alter the data each iteration. AdaBoost has a strong adaptability and, hence, is widely used as one of the most popular boosting algorithms (Wang and Feng (2020)). Furthermore, even with a large number of base estimators Wyner (2003) has found the model to rarely be prone to overfitting. This is desirable since a large number of estimators is expected to be required, considering the large size of the data used in this research. Research by Dietterich (2000) has shown that, compared to randomising and bagging, AdaBoosting gives the best results provided that there is minimal noise in the dataset.

Another diversification on the widely used gradient boosted tree algorithm introduced by Friedman (2001), is the Histogram-based gradient boosting algorithm. In this method the computational complexities that boosting algorithms are notorious for are limited. The data is reduced to integer-valued data instances, reducing the complexity and therewith decreasing the computation time needed for implementation. This is advantageous when working with large datasets. Ke et al. (2017) find histogram based boosting to be the fastest compared to other gradient boosting methods like XGBoost (Chen and Guestrin (2016)) and stochastic gradient boosting (Friedman (2002)), while maintaining the same accuracy as the regular general boosting model.

3 Data

For the empirical study we use U.S. equities, based on which the performance of machine learning methods for measuring equity premiums is examined. Similar to the data used by Gu, Kelly, and Xiu (2020), for the firms in the NYSE, AMEX and NASDAQ, monthly equity returns from the Centre for Research in Security Prices (CRSP) are obtained. These are collected from the Wharton Research Data Services. The sample ranges from March 1957, which is the starting date of the S&P 500, to December 2016.[1] The data used contains close to 30,000 stocks, 6,200 per month on average. Details on the stock choices can be found in Appendix A.

Stock portfolios for asset pricing models are composed based on empirically motivated characteristics. Gu, Kelly, and Xiu (2020) use a set of 94 stock-level predictive characteristics based on those used by Green, Hand, and Zhang (2017). We incorporate a selection of these characteristics (P_c), in line with findings by Gu, Kelly, and Xiu (2020), that prove a set of 25 characteristics to be the most important variables for the regression forests and neural network models. We only incorporate this selection of variables in our research so less computational power is required. We believe only including the 25 most important characteristics is justified as Gu, Kelly, and Xiu (2020) find that the most influential characteristics are not affected by the presence of irrelevant characteristics. Details on the characteristics can be found in Appendix B. The 74 dummy variables denoting the last two digits of the SIC number are excluded from our research due to the large computational power required, while having little importance in the final model (Gu, Kelly, and Xiu (2020))

The following set of 8 macroeconomic predictors is included in combination with a constant (P_x), as described in detail by Welch and Goyal (2008): dividend-price ratio, earnings-price ratio, book-to-market ratio, net equity expansion, treasury-bill rate, term spread, default spread, and stock variance.[2]

Regression forests and neural network algorithms are able to capture complex interactions amongst predictors. Therefore, we include 20 interactions between stock-level characteristics and macroeconomic variables in our models. This selection is based of the top 100 interactions that were found to be most important by Gu, Kelly, and Xiu (2020). The specific interactions between stock characteristics and macroeconomic variables are listed in Appendix C.1.

The complete variable vector $z_{i,t}$, containing stock characteristics ($P_c = 25$), macroeconomic variables ($P_x = 9$) and interactions (P_i) is a $P \times 1$ vector $z_{i,t}$, for stock i, i = 1,...,N_t, in month t, t = 1,...,T, with $P = P_c + P_x + P_i$. Here T is the last month of the data sample and N_t denotes the number of stocks included in month t.

3.1 Sample splitting

For the machine learning models it is of importance to determine the optimal number of parameters to use for out of sample equity risk premium prediction. For this purpose, the sample is split up into three samples: a training sample (X_t), a validation sample (X_v) and sample for out-of-sample testing (X_{oos}). The samples are updated annually. This is because the tuning of parameters is computationally intensive, and little improvement in the results is expected compared to monthly

[1] Observations for which the stock return is unknown are excluded

[2] From the website of Amit Goyal the corresponding dataset is obtained

updating of the samples, since most characteristics are updated annually. A recursive strategy is used to train the model. The first training sample ranges from 1957 to 1974 (18 years), the validation sample consists of the following 12 years and ranges from 1975 to 1986. These two samples are used to estimate a model to predict the first out-of-sample values for the year 1987. In order to predict the out-of-sample observations of the following year, the training sample increases by one year and the validation sample rolls forward one year. For a detailed explanation on the sample selection consult Appendix D. The training sample is used to estimate the model using a specific set of parameters. The performance of this model is tested on the validation sample, after which the model is re-estimated using a new set of parameters. The best performing parameter combination is used to predict the out-of-sample stock returns. In the methodology section we further elaborate on parameter selection (hyperparameter tuning).

3.2 Optimising data

To optimise the performance of machine learning algorithms it is beneficial to standardise the data in the training sample (Ian and Eibe, 2005). This is done by subtracting the training sample mean and dividing by the standard deviation per period. Thereafter, these mean and standard deviation are combined in a scaler that is also used to standardise the validation and out-of-sample observations. It is important to only include data of the training sample for the computation of the scaler. Otherwise, information on the validation and out-of-sample data is already included in the estimated parameters, that are trained on the standardised training sample. This results in false prediction accuracy when the estimators are tested on the validation sample.

4 Methodology

We examine a selection of machine learning methods for the purpose of optimal asset pricing. Firstly, results for the random forest model (RF), introduced by Breiman (2001), are computed. Also extremely randomised random forests (ERF) are computed, based on an algorithm introduced by Geurts, Ernst, and Wehenkel (2006). Thereafter, a boosting algorithm AdaBoost (AB), introduced by Freund and Schapire (1997), is implemented. Lastly, histogram-based gradient boosted regression trees (H-GBRT), inspired by Ke et al. (2017) is implemented, as a variation on the gradient boosted random forest used by Gu, Kelly, and Xiu (2020).

For the methods listed above a detailed description is given in the following sections containing the statistical model, used for risk premium predictions, an objective function, for estimating model parameters, and lastly the computational algorithms used to identify the optimal specification for the given method[3].

Excess return of an asset is formulated as an additive prediction error model:

$$r_{i,t+1} = E_t(r_{i,t+1}) + \varepsilon_{i,t+1}, \qquad \text{in which } E_t(r_{i,t+1}) = g^*(z_{i,t}). \tag{1}$$

This is for stock i, i = 1,...,N_t in month t, t = 1,...,T. A function for the conditional expected returns

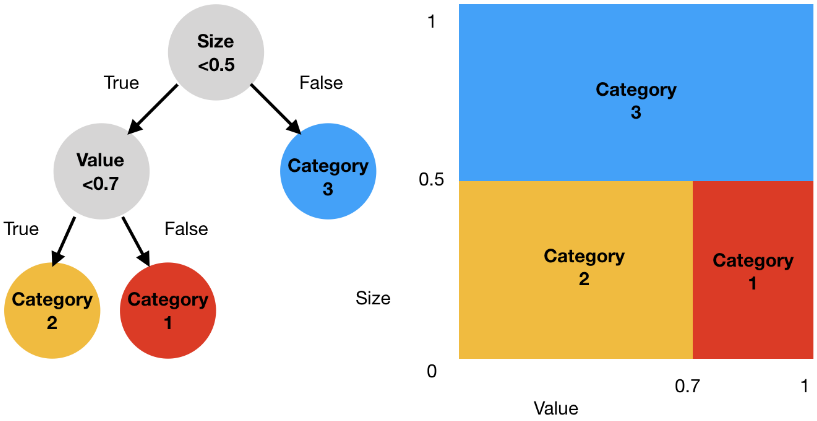
[3] For the assembly of the total dataset, and, thereafter, the selection of specific features and relations MATLAB(2020a) was used. The cross covariates were added in Python. Also the models are programmed in Python, as this is more fit for machine learning in combination with large datasets. We used Google Colab to be able to fit the models on the dataset, which has a considerable size.

$E_t(r_{i,t+1})$ is to be found, given by the model function $g^*(z_{i,t})$, containing predictor variables $z_{i,t+1}$, which maximises the out-of-sample explanatory power. $z_{i,t}$ is a $P \times 1$ vector, in which P denotes the total number of predictor variables consisting of the characteristics (P_c), the macroeconomic predictor variables including a constant (P_x) and the cross covariates for interactions between characteristics and predictor variables (P_i).

4.1 Random forest

The first method we examine in this research is the random forest (RF). The method is formulated following the research from Breiman (2001). RF is a nonparametric machine learning method, able to capture nonlinearities and include multivariate functions of predictors. In general, trees aim to group observations based on similar behaviour, splitting up the sample per branch, therewith growing the tree. As shown in figure 1, adopted from a figure from Gu, Kelly, and Xiu (2020), the 1 by 1 space is partitioned into two smaller rectangles per branch, based on predictor variables such as size and book-to-market ratio. The resulting model function $g^*(.)$ is composed with the average values of the variable outcomes in each partition. The aim is to minimise the forecast error by choosing specific predictor variables and values for the splitting of branches.

Figure 1. Example of a regression tree



The representation of the regression tree (left) is mapped into a [1, 1] space (right), drawn up by two characteristics. The colours blue, yellow and red denote the terminal nodes.

A greedy algorithm from Hastie, Tibshirani, and Friedman (2009) is used to grow the binary regression trees. The standard least square or l_2 objective function denoting the loss or impurity for branch C is:

$$H(\Theta, C) = \frac{1}{|C|} \sum_{z_{i,t} \in C} (r_{i,t+1} - \Theta)^2, \qquad (2)$$

in which $|C|$ is the number of observations in set C. For minimum loss the optimal choice for Θ is: $\Theta = \frac{1}{|C|} \sum_{z_{i,t} \in C} r_{i,t+1}$. On each split $s = (j, \alpha)$, in which $j = 1, 2, ..., P_f$ denotes a feature and α a threshold level, set C can be bisected into set C_{left} and C_{right}, containing observations with features smaller than and larger than the threshold value α, respectively. The trees are split up early based on the same prominent return predictor variables, causing high correlation in the produced forecasts. To minimise this correlation, the set of features P_f, from which the predictor to split the sample on is chosen, is randomly selected per split from the complete feature set P, with a fixed size of F. For every possible split s the impurity function is given by:

$$\mathcal{L}(C, C_{left}, C_{right}) = \frac{|C_{left}|}{|C|} H(\Theta, C_{left}) + \frac{|C_{right}|}{|C|} H(\Theta, C_{right}). \tag{3}$$

By minimising the impurity function the optimal split s^* is chosen, further branching out the tree:

$$s^* \leftarrow \underset{s}{\operatorname{argmin}} \, \mathcal{L}(C(s), C_{left}(s), C_{right}(s)). \tag{4}$$

Branching stops when the prespecified threshold of maximum tree depth (L) is reached.

The forecasts from many different trees are combined into a single prediction. This regularisation is necessary to reduce overfitting with the use of large data sets. For the computation of multiple trees, B bootstrap samples b are generated from the original dataset. A regression tree is fit to each individual sample. Subsequently, the average of the resulting forecasts is calculated. The function for each individual tree becomes:

$$\hat{g}_b(z_{i,t}; \hat{\Theta}_b, L) = \sum_{k=1}^{2^L} \Theta_b^{(k)} 1\{z_{i,t} \in C_k(L)\}, \tag{5}$$

resulting in a final RF function given by the averages of all B trees:

$$\hat{g}(z_{i,t}; L, B) = \frac{1}{B} \sum_{b=1}^{B} \hat{g}_b(z_{i,t}; \hat{\Theta}_b, L), \tag{6}$$

The performance of the model is influenced by the size of the set of features (P_f), the maximum tree depth of the estimated trees (L), and the total number of trees or bootstrap samples included in the RF (B). The optimal combination of these parameters for various out-of-sample forecasts is highly dependent on the size and characteristics of the training and validation data used. The optimal values are found using hyperparameter tuning. Here a model is estimated on the training sample with a specific set of parameters and subsequently its performance is tested on the validation sample. Appendix E further elaborates on this method. The set of parameters from which the optimal parameters are determined is given in figure 5 in Appendix E. Due to time limitations hyperparameter tuning is only performed for RF, the other models are run on several fixed sets of parameters. These fixed parameters are based on the most frequently used parameters in computation of the trees in the random forest model which are $P_f = 20$, $L = 3$ and $B = 300$, for respectively the size of the set of features, the maximum tree depth and the total number of trees. Thereafter we examined the performance, using these values as a starting point, for one-year forecasts, while considering the time required for estimating the models on the training sample. This resulted in multiple models for each method.

4.2 Extremely Randomised Forest

An extension of RF is extremely randomised trees (ERF), introduced by Geurts, Ernst, and Wehenkel (2006). The difference between these two methods is in the selection of splits ($s = (j, \alpha)$). Similar to the RF, the set of candidate features (P_f) is randomly generated. However, in ERF the threshold values α are also drawn at random for each feature, instead of being optimised using the impurity function given by equation (3). This is done by picking a uniform value for α from the range $[\alpha^C_{min}, \alpha^C_{max}]$, in which α^C_{min} and α^C_{max} are, respectively, the minimum and maximum value for a specific feature in branch C. Using the randomly generated thresholds the optimal split is chosen using equation (4), to further branch out the tree. Similar to RF, the tree is terminated when the maximum depth L is reached.

Due to the randomised selection of thresholds this algorithm is much faster than the random forest, allowing for the computation of a larger number of trees B. Furthermore, the performance of the algorithm depends on the size of the set of features (P_f) and the maximum tree depth (L).

The first of three ERF models denoted by "Extremely Random Forest (1)", makes use of $P_f = 25$, $B = 500$ and $L = 3$. Here a larger number of features is chosen compared to the RF model as its computation time is much shorter and a larger subset is expected to lower the bias. A variation on the number of features is given by "Extremely Random Forest (2)", with $P_f = 30$, $B = 500$ and $L = 3$. After 25 features was found to be best performing, to inspect the dependence of the performance on the number of estimators, models using $B = 250$ and $B = 1000$ were computed, of which the later was found to perform best ($DM = 0.437865$, see Appendix F.2). This method uses $P_f = 25$, $B = 1000$ and $L = 3$, and is grouped as "Extremely Random Forest (3)".

4.3 AdaBoost

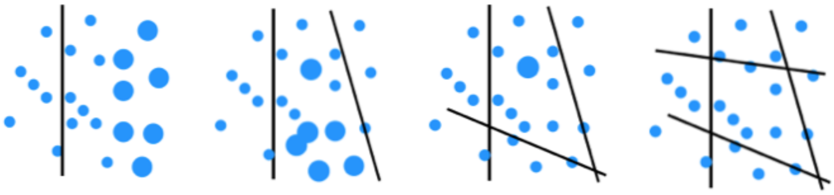
As an extension of the regular RF model, several boosting techniques are examined. In boosting, machines are sequentially trained and the resulting regressors are additive models:

$$\hat{g}(z_{i,t}) = \sum_{w=1}^{W} \hat{g}_w(z_{i,t}), \tag{7}$$

in which $\hat{g}_w(z_{i,t})$ are estimators referred to as weak learners. In boosting new weak learners are computed with the aim to improve the error from the previous model, opposed to bagging, which is used in the RF models, where new trees are fitted to bootstrap samples of the data and generated independently from the other trees.

Firstly, the AdaBoost algorithm (AB), short for adaptive boosting algorithm, introduced by Freund and Schapire (1997), is implemented. In this algorithm a set of weak models is computed and fit to datasets that are constantly altered using various weights. A linear combination of the weak models is constructed to obtain a strong model. Subsequently, a weighted majority vote combines their predictions.

Figure 2. Reweighing of datapoints after model fitting



The blue dots denote the datapoints in the training sample and the black lines represent the weak learners. Difficult to predict datapoints are given a larger weight, therewith increasing the probability of being included in the training sample for the following model.

The regression is based on the model from Drucker (1997). Instead of resampling, which is used for the RF model, the training data is now reweighted after each regression. The process of assigning new weights to the data that is poorly predicted by the previously fitted model is shown in figure 2. After each model fit, the datapoints are reweighted to stimulate the poorly predicted datapoints to be included in the training sample of the following model. The main loop stops when all datapoints in the training sample are correctly specified or when the iteration limit is met.

We use a simple decision tree regressor with $P_f = 20$, $L = 3$, for our weak learner models, from which the boosted ensemble is computed[4]. The weak learners are fit to a training set, which is assembled by picking N_t times a uniform number from the range $[0, \sum_{i=1}^{N_t} \omega_i]$. When weight ω_i is chosen this causes observation i to be included in the training set. Here $i \in 1, ..., N_t$, is an observation in the complete training sample which has size N_t. To initiate the model all dataweights ω_i are set to 1. This causes the probability p_i for observation i to be in the training set to be 1 ($p_i = \omega_i / \sum_{i=1}^{N_t} \omega_i$). Thereafter, the weak learner is fit to the complete training sample and a forecast $\hat{r}_{i,t}$ is obtained. We use the following linear loss function to calculate the loss of each observation i from the training sample:

$$L_i = \frac{|\hat{r}_i - r_i|}{max\{|\hat{r}_i - r_i|, i = 1, ..., N_t\}}, \tag{8}$$

of which the average is taken over the complete training sample resulting in an average loss of:

$$\bar{L} = \sum_{i=1}^{N_t} L_i p_i. \tag{9}$$

Based on these losses the weights are transformed. A factor $\beta = \frac{\bar{L}}{1-\bar{L}}$ denotes the confidence of the prediction. β is lower when the confidence is higher due to a lower value of the average loss function. The original weights (ω_i) are transformed according to the following formulation:

$$\omega_i^{new} = \omega_i^{old} \beta^{[1-L_i]}. \tag{10}$$

This results in an increase in weight for a large loss function, increasing the probability that this observation will be picked for the next training set, which is used to fit the following weak learner.

[4] These values are based on the most frequently occurring options that were found after tuning the hyperparameters for our regular random forest model.

The fitting of new weak learners terminates once the maximum number of estimators (W) is reached.

The resulting model is formulated by:

$$\hat{g}_W(z_{i,t}) = \sum_{w=1}^{W} \alpha_w \hat{g}_w(z_{i,t}), \tag{11}$$

and consists of a linear combination of the fitted weak learners $\hat{g}_w(z_{i,t})$, where α_w is a weight assigned to the weak learner models. Weight α_w is formulated as $\alpha_w = \rho * log(\frac{1-\bar{L}}{\bar{L}})$, in which ρ is the learning rate. The model performance depends on the number of included weak learners (W), and their contribution in the resulting strong regressor through the learning rate (ρ).

We compute two models with different sets of hyperparameters. For "Ada Boosted (1)" and "Ada Boosted (2)", the incorporated parameters are $W = 250$, $\rho = 0.5$ and $W = 200$, $\rho = 1$, respectively. Due to the slow computation of AB, a relatively small number of estimators is used compared to the original RF model. The different learning rates are chosen based on the trade off between the number of estimators included (W) and the learning rate (ρ).

4.4 Histogram-based gradient boosting

Another boosting algorithm we examine is Histogram-based gradient boosting. Histogram-based gradient boosting regression trees (H-GBRT) are inspired by light gradient boosting machines (LightGBM), introduced by Ke et al. (2017). When using large datasets these estimators are found to be faster than regular gradient boosting regression trees (GBRT). Different from the RF algorithm, the input training sample is first split into integer-valued groups, forming a feature histogram. This drastically decreases the number of splits that need to be considered when optimising the impurity function (3). This is because instead of individual datapoints, sets of datapoints are considered.

For the H-GBRT algorithm we use a formulation of Cai et al. (2020). The algorithm is a variation on the standard GBRT algorithm adopted from Friedman (2001). To initialise the algorithm the first weak learner $\hat{g}_0(z_{i,t})$ is set to 0. For gradient boosting the negative function gradient of the loss function $L(z_{i,t}, r_i) = (r_i - \hat{g}_W(z_{i,t}))^2$ is to be defined every iteration for each stock $i = 1, ..., N_t$ and month $t = 1, ..., T$:

$$\varepsilon_{i,t+1} = -\frac{\partial L(r_{i,t+1}, g_W(z_{i,t}))}{\partial g_W(z_{i,t})}\bigg|_{g_W(z_{i,t})=\hat{g}_{W-1}(z_{i,t})}, \tag{12}$$

in which $\hat{g}_{W-1}(z_{i,t})$ is the sum of weak learners from the previous iterations. Thereafter, a new tree is fit to the negative gradient $\varepsilon_{i,t+1}$. This is done by performing the minimisation given by equation (4) on every split, using the computed feature histogram, the use of which minimises the computation time. This tree is multiplied by a shrinkage factor ρ and added to the sum of weak learners:

$$\hat{g}_W(z_{i,t}) = \hat{g}_{W-1}(z_{i,t}) + \rho \hat{g}_w(z_{i,t}). \tag{13}$$

The resulting model is given by:

$$\hat{g}_W(z_{i,t}) = \sum_{w=1}^{W} \rho \hat{g}_w(z_{i,t}). \tag{14}$$

The performance is, similar to regular regression trees, influenced by the maximum number of estimators (W) and the tree depth (L). Specific for boosting algorithms is the influence of the shrinkage factor (ρ) for the effect of weak learners in the final regression model. For the model "Histogram-based GBRT (1)" the following parameters are used: $L = 3$, $\rho = 0.01$ and $W = 1000$. For "Histogram-based GBRT (2)" the parameters $L = 3$, $\rho = 0.01$ and $W = 2000$ are used, for an expected increase in the predictive performance.

4.5 Performance

For examination of the performance of the various model for out of sample forecasting, we examine two test statistics. The performance of stock return forecasts is denoted by the out-of-sample R^2. This test statistic is computed to enable quantitative comparison of the predictive performance of the models. The R^2_{OOS} statistics for the various models are first computed for the entire data sample after which the R^2_{OOS} test statistic is computed for the top-1000 largest and bottom-1000 smallest stocks in terms of market value. The stocks are grouped per month after which the predictions corresponding to the top-1000 and bottom-1000 stocks are isolated for estimation of the R^2_{OOS} statistics. We use the following formulation for the out-of-sample R^2:

$$R^2_{OOS} = 1 - \frac{\sum_{(i,t)\in T_3} (r_{i,t+1} - \hat{r}_{i,t+1})^2}{\sum_{(i,t)\in T_3} r^2_{i,t+1}},$$

here, T_3 denotes the testing sample, ensuring data involved in estimation or tuning is not included.

Following the quantitative comparison of stock return forecasts, the Diebold-Mariano test statistics are evaluated to determine the statistical significance of these differences. The test statistic is given by:

$$DM_{12} = \frac{\bar{d}_{12}}{\hat{\sigma}_{\bar{d}_{12}}},$$

in which $\bar{d}_{12}$ is the average over the observations in the testing sample from the following formulation:

$$d_{12,t+1} = \frac{1}{n_{3,t+1}} \sum_{i=1}^{n_{3,t+1}} \left(\left(\hat{e}^{(1)}_{i,t+1}\right)^2 - \left(\hat{e}^{(2)}_{i,t+1}\right)^2 \right).$$

Details on the formulation and computation for both the R^2_{OOS} and the Diebold-Mariano test statistics can be found in Appendix F.

Apart from testing the predictive performance of the models incorporated in this research, it is of importance for interpretation of the machine learning models to uncover the characteristics, macroeconomic variables and cross covariates that show to be most important in the estimated models. Similar to Gu, Kelly, and Xiu (2020), the feature importance is denoted by the decrease in the R^2_{OOS} when excluding a specific predictor. In order to determine the importance of feature j its values are set to zero in a new adjusted dataset. For each year the model is refit on the adjusted dataset and, subsequently, computes forecasts for the out of sample returns. Using these forecasts, a new R^2_{OOS} statistic is obtained. The difference between the original and the newly computed R^2_{OOS} is the resulting feature importance. We obtain these importances for each out-of-sample year, using a corresponding newly fit model each iteration. To portray the importance of characteristics the feature importance is summed over each out-of-sample year and subsequently normalised. Alternatively, to examine the model contributions of all stock level characteristics, these characteristics are given a rank based on their importance per forecast sample. These ranks are summed over all 30 out of sample years to show the overall importance between features.

5 Results

This section contains the results obtained for the various models in regard to the predictive power of the models discussed. We examine both the quantitative importance and the statistical significance of these differences. Subsequently an analysis is done of the most prominent predictive features in the various models.

5.1 Prediction performance

We start by examining the predictive performance of the various proposed models in predicting stock returns and their underlying economic relations. The performance is measured using the R-squared test statistic. Figure 3a shows the total R^2_{OOS} in combination with the R^2_{OOS} for the top en bottom 1000 stocks in market value each period, averaged over all out of sample years. Here the performance of the AB[5] models is left out as these showed large negative values ranging from 0 to -74. To further investigate the prediction performance, the desirable performing models, RF[6] and ERF[7], are isolated form the H-GBRT models as shown in figure 3.

Table 1

Annual out-of-sample prediction performance (R^2)

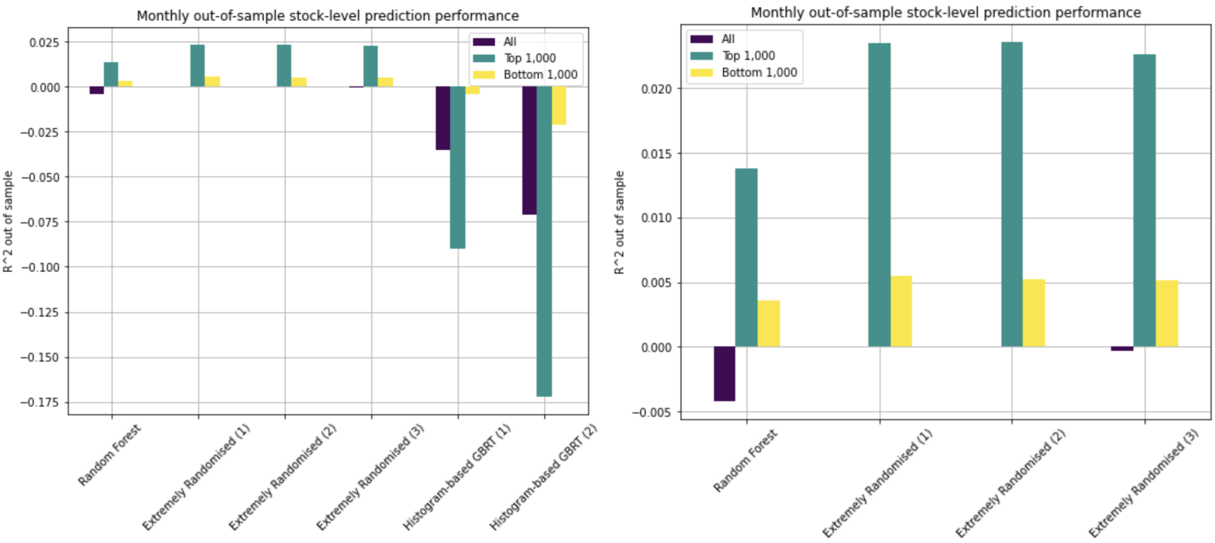
	RF	**ERF (1)**	**ERF (2)**	**ERF (3)**	**AB (1)**	**AB (2)**	**H-GBRT (1)**	**H-GBRT (2)**
All	-0.004	0.000	0.000	0.000	-35.534	-39.060	-0.035	-0.071
Top	0.014	0.023	0.024	0.023	-73.740	-65.053	-0.090	-0.172
Botttom	0.004	0.005	0.005	0.005	-9.516	-10.038	-0.004	-0.021

[5] AdaBoost

[6] Random forest

[7] Extremely randomised forest

Figure 3. Monthly out-of-sample stock-level prediction performance (R^2_{OOS})



(a) Complete set of models

(b) RF compared to ERF

From figures 3a and 3b it follows that the ERF models show overall improvements of the regular RF model. Especially in predicting the top 1000 largest stocks the RF R^2_{OOS} is almost doubled by the ERF R^2_{OOS}, with values of respectively 0.014 and 0.023 to 0.024. From fig 3b it can be seen that for the ERF models the use of different hyperparameters has a minimal influence on the resulting prediction accuracy. Figure 3a shows that the H-GBRT models have a relatively low predictive power. Negative values are found for the R^2_{OOS} test statistic indicating that even a simple forecast of zero for all forecasts would outperform the predictions. This can possibly be explained by the values of the characteristics that are grouped in an integer-valued histogram, causing the distinction between variables whose values lie close together to dissipate. This generalisation detracts from the predictive power of the model.

The preference for bagging models (RF and ERF) over boosting models (AB and H-GBRT) is in line with findings by Dietterich (2000), that show this predilection to hold when working with noisy datasets.

Table 2

Comparison of monthly out-of-sample prediction using Diebold-Mariano tests

	ERF (1)	**ERF (2)**	**ERF (3)**	**AB (1)**	**AB (2)**	**H-GBRT (1)**	**H-GBRT (2)**
Random Forest	1.588	1.476	1.370	**-7.361**	**-8.658**	**-2.746**	**-3.954**
Extremely Randomised (1)		-0.113	-1.045	**-7.362**	**-8.660**	**-2.993**	**-4.069**
Extremely Randomised (2)			-0.677	**-7.362**	**-8.659**	**-3.015**	**-4.079**
Extremely Randomised (3)				**-7.362**	**-8.659**	**-2.945**	**-4.029**
Ada Boosted (1)					0.344	**7.349**	**7.339**
Ada Boosted (2)						**8.644**	**8.631**
Histogram-based GBRT (1)							**-5.347**

After having evaluated the quantitative comparison, the statistical significance is substantiated by Diebold-Mariano test statistics, shown in table 2. Here a positive value indicates a better performance of the column model compared to the row model, due to smaller prediction errors. The bold values show the individual significance at a 5% level. These values remain significant after a Bonferroni correction for 12-way comparison (see Appendix F).

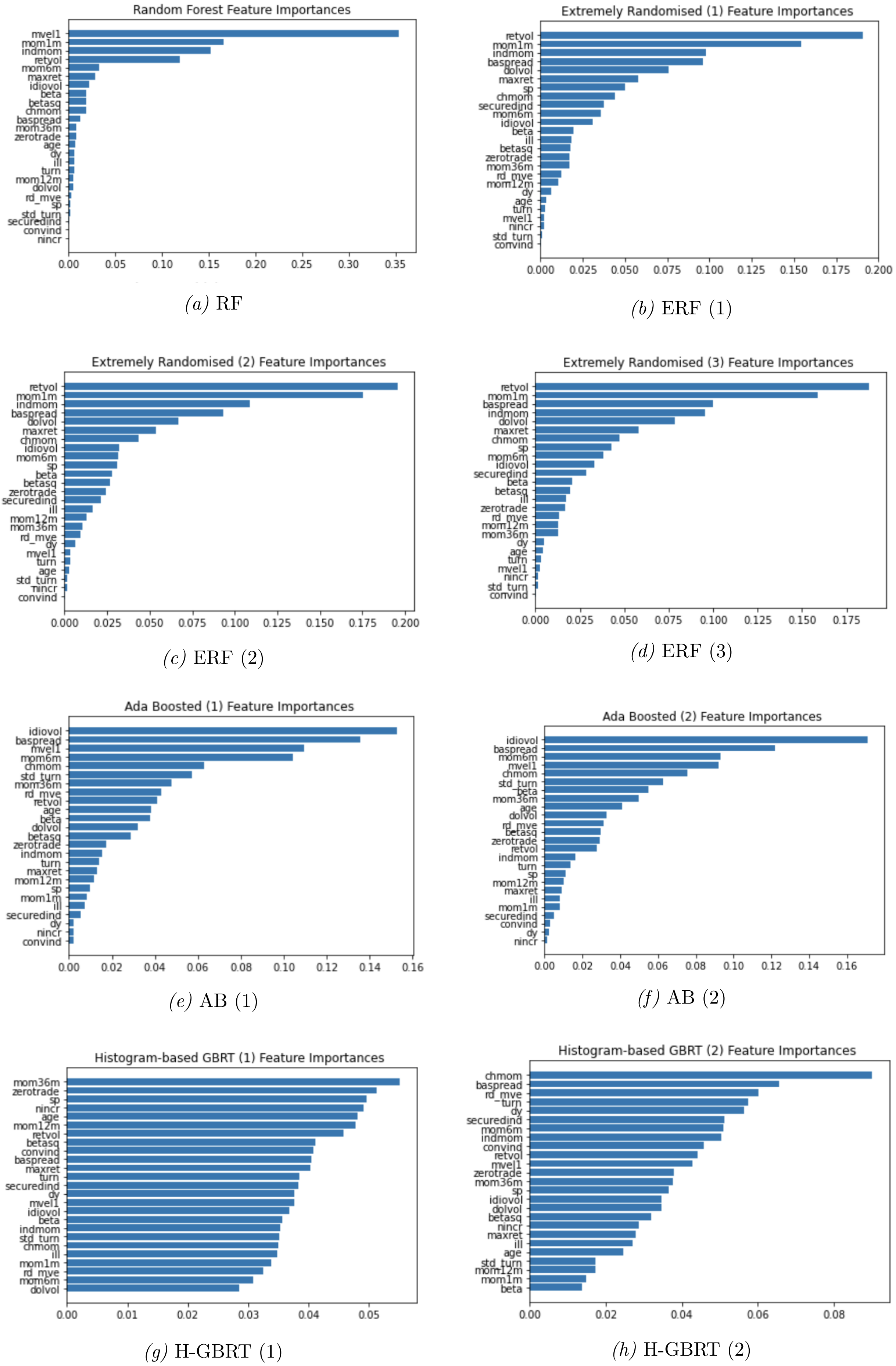
From this table it can be concluded that only the ERF models are found to outperform the standard random forest model. However, the computed DM statistics show this result to be insignificant ($DM = 1.588$). Amongst the three ERF models, the first model is found to be best performing, followed by the second and third model. This instigates the believe that there is an optimal set of estimators, since the use of $B = 1000$ as opposed to $B = 500$ does not further improve the predictive performance. This follows from the DM statistic of -0.677, comparing ERF(2) and ERF(3). This can also be derived from the dataset, that includes observations with a great amount of noise, causing the variance of the predictions to increase when incorporating an increasing amount of estimators. Still satisfactory results are obtained since machine learning methods are known for being able to filter out the underlying correlation especially for large and noisy datasets.

Also, for H-GBRT a larger set of estimators results in less accurate forecasts. Furthermore, the H-GBRT forecasts outperform the forecasts computed by the AB models, and of the two diversifications, H-GBRT (1) shows to perform optimal. The AB models produce the least accurate forecasts. Since the regular RF model is used as a weak learner, this estimator is not expected to be problematic for the accuracy. The poor performance can be explained by the relatively small number of 250 and 200 estimators, which were used due to the slow computation time of this model. These numbers show to be unfit for our dataset, which requires a larger number of estimates, for the machine learning method to capture the the correlations.

5.2 Feature importance

The importance of each feature is estimated each month. These values are added and normalised to show their relative importance. For each model the feature importance is visualised in figure 4, subplot a to h. Another visualisation of the feature importance is given by figure 5. Here the feature importance is ranked each period, hereafter summed, and subsequently plot in. The y-axis denotes the feature order based on the total rank summed over each model. The colours in each column show the specific importance of a feature in the corresponding model.

Figure 4. Variable importance by model



(a) RF

(b) ERF (1)

(c) ERF (2)

(d) ERF (3)

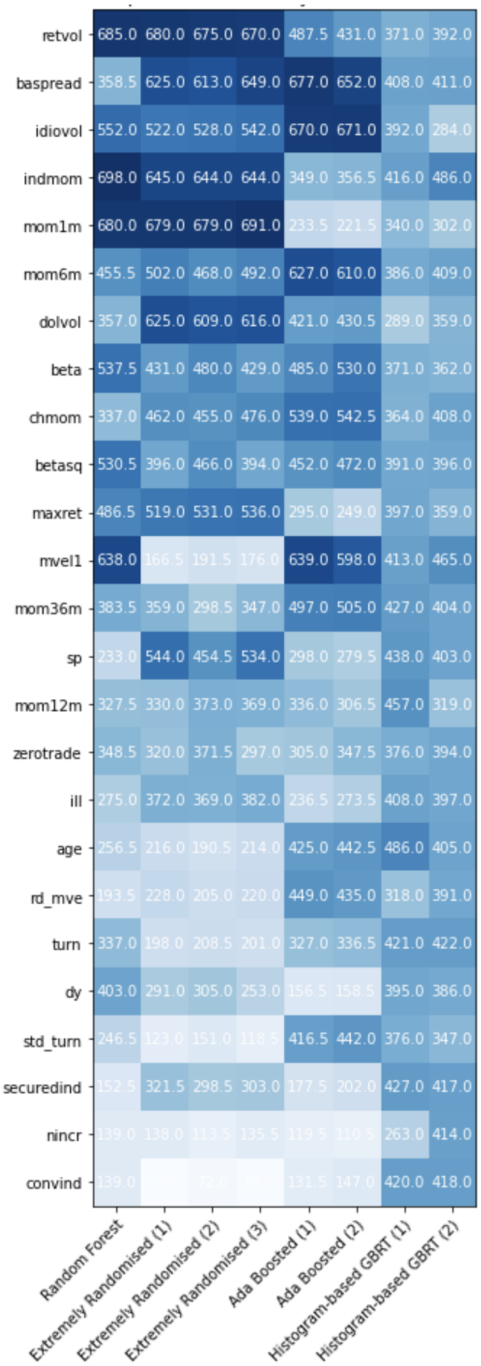
(e) AB (1)

(f) AB (2)

(g) H-GBRT (1)

(h) H-GBRT (2)

Figure 5. Characteristic importance



It is found that the selection of characterisitcs is uncorrelated with the choice of hyperparameters for ERF and AB. The top features are the same for different sets of hyperparameters. For the H-GBRT models this similarity is not found, indicating that the choice of features is highly dependent on the number of estimators incorporated. Moreover, the distribution is likely to deviate for different estimations due to the variation in characteristics on which the regression trees estimators are branched out.

When comparing figures 4b, 4c, 4d and 4e, for the ERF and AB models, a similar feature importance distribution is found. For the ERF models retvol and mom1m are found to be the most influential, taking up a combined importance between 35% and 40% from the total of variable influence. In the AB models idiovol and baspread are most often incorporated in the model estimation, with a combined importance of around 30%. This is followed by a slowly declining importance for the other characteristics. Unlike the ERF and AB, the RF model in figure 4a, is largely influenced by mvel (35%), accompanied by mom1m, retvol and mom6m, after which the variable importance rapidly declines. This indicates a more distinctive and consistent selection of variables for the regression tree estimators compared to the other models. The variable importance of mom1m for the RF model is in line with findings by Gu, Kelly, and Xiu (2020), while, dissimilar, they also find dy to be of great influence and obtain a less skewed distribution. In figure 5, portraying the added ranks, the ERF and RF are more in accordance. For both models, recent prince trend variables indmom and mom1m and risk measure retvol are of importance.

For the well performing methods RF and ERF, a similar set of variables is incorporated in the models, which is evident from heat map in figure 5. Only mvel is of little importance in the ERF models compared to the RF model in which it is a leading variable in the model computation. Inclusion of these variables is thus thought to produce desirable results. It is, however, also possible this similarity originates from the way in which the models are constructed. Apart from the randomly chosen threshold values in ERF, these methods are identical.

Furthermore, the distribution of the histograms in figure 4g and 4h, for the H-GBRT models, is more evened out. This entails that there is a great variation in the features that are chosen for the estimators. This can possibly be explained by the generalisation of features due to the use of a histogram when selecting characteristics for the optimal split. This causes the selection of different characteristics to produce the same optimal value. Consequently, a great variation in estimator trees is found.

6 Conclusion

In this thesis we studied the performance of the random forest model compared to various variations on this model, for the purpose of asset pricing. The objective was to obtain a model formulation that captures complex underlying correlations of stock returns and therewith produces forecasts with a large explanatory power. We investigate the random forest, the extremely randomised regression trees, the adaptive boosting model and the histogram-based gradient boosting model. For each of the models considered, we test and compare the out of sample forecasts. Furthermore, we examine which predictive features are leading in the model computation.

Our research suggests only the extremely randomised regression trees outperform the simple random forest model. The other modifications of this simple model do not equate improvements in the predictive accuracy. Our analysis suggests there exists an optimum for the number of estimators to include in the extremely randomised regression model, after which the predictive performance diminishes again due to the increase in noise. The increased randomisation of the regression tree construction compared to the regular random forest model is found to work well with the large and noisy dataset. This is indicated by the R^2_{OOS} test statistics of 0.023 and 0.005, for respectively the top 1000 and bottom 1000 stocks in market value, which are larger than the values found for the random forest ($R^2_{OOS,\text{ top 1000}} = 0.014$, $R^2_{OOS,\text{ bottom 1000}} = 0.004$).

Our implementation of adaptive boosting algorithms is inadequate for the purpose of return forecasting, as a consequence of the inclusion of a small number of estimates.

Likewise, histogram-based gradient boosting is found to produce undesirable results. By grouping characteristics before branching out, distinctions in predictive characteristics become obscured which detracts from the predictive power of the model. Incorporating more estimators further amplifies this problem.

Overall, we find the best results for forecasting are obtained using the extremely randomised forest model. Even without the use of hyperparameter tuning in the selection of model parameters this model proves to perform better. The Diebold-Mariano test statistic suggests, however, that this difference is insignificant.

Our results show that in the optimal performing models similar features are incorporated in the construction of regression trees. For both random forest and extremely randomised forest the retvol, mom1m and indmom capture the largest part of the model specifications. These variables form the drivers of the model and have the largest influence on the stock returns from all predictive variables incorporated.

These promising findings imply further enhancement of the model specifications is expected to produce optimal results by means of more accurate forecasting. In future research advancements in predictive power of the extremely random forest model can, therefore, be attained by using hyperparameter tuning and targeted feature selection on the basis of the most important predictor variables. To conclude, exploration of the extremely randomised regression trees for predicting stock returns is a valuable addition to existing literature on asset pricing.

7 Discussion

Our research is limited in several respects. Further research is required to obtain a more complete and extensive analysis in the field of research on asset pricing using machine learning models.

Firstly, in respect to the dataset, several improvements are possible through inclusion of independent variables, which are currently excluded due to time limitations. Since machine learning methods are suitable for capturing underlying correlations in big noisy datasets, it is beneficial to use a great number of useful and available predictor variables. We only include 25 of the 94 characteristics used by Welch and Goyal (2008). The dummy variables for the SIC number, for example, are currently left out, even though Gu, Kelly, and Xiu (2020) show this to have a substantial predictive power at annual frequencies. Even so, Collot and Hemauer (2021) underline that hundreds of factors for return prediction have been identified. Including only a small selection of variables is prone to result in an omitted variable bias. This bias originates from the exclusion of variables from the predictive model that correlate with the dependent variable. Since omitted variable bias is one of the main problems in the empirical asset pricing literature, it is desired for future research to incorporate all available stock characteristics. Moreover, only a selection of all possible variable cross covariates is included. Further research that incorporates multiple and more complex variable relations is believed to increase the predictive accuracy of the machine learning models.

Secondly, apart from the predictive features included in the research, the predictive power is highly dependent on the training sample which is used to estimate the model parameters. Currently, the various models are fit to a training sample ranging from 1957 to 1974, including a new year after each out of sample prediction. Employing a different method than this rolling window method elaborated on in Appendix D, might result in forecasts that describe recent stock return behaviour more accurately. This can either be done through rolling the training sample window forward instead of constantly increasing its size. Or by making use of a smaller, more recent sample. Paying more attention to the sample selection is thought to produce better results.

Furthermore, we are able to make instant advancements in the model formulations by employing hyperparameter tuning for the selection of model parameters. Doing so results in time specific optimal parameters and thus more accurate predictions. Moreover, improvements for the individual models might enable the boosting algorithms, that are found to be unfit for the purpose of asset pricing in our research, to equate the performance of random forest models. AdaBoosting, for example, is expected to show significant advancements when the type of weak learner is not fixed. The regression models used as estimators can be obtained through hyperparameter tuning using a set of different weak learnsers. Therefore, the AdaBoosting model becomes more flexible in return predictions. Also experimenting with the type of law loss function, which can also be exponential or squared, might more adequately capture the out of sample data points. Additionally, the histogram-based gradient boosting algorithm can be altered to hinder the generalisation of predictive features. The sizes of the histogram groups could be computed differently to prevent to many datapoints to be grouped together, and similar variables to be seen as the same characteristic. This leaves the benefit of a decreased computation time while maintaining clear distinctions amongst different variables.

Apart from more extensive research into enhancement of the models examined, analysing various other models for comparison could further justify and extend the obtained results. Since Gu, Kelly, and Xiu (2020) found neural networks to perform slightly better than the random forest models, it is advised to incorporate this method using our specific dataset to be able to compare the results. This way we are able to analyse whether the extremely randomised forest algorithm is actually the best performing method in the existing machine learning literature for asset pricing, outperforming the neural network models. Also, further extensions using machine learning methods for forecasting, that we were not able to incorporate in our research, are interesting for further research and essential for the completeness of the existing literature. Especially since the random forest model and extremely random forest model were found to perform best, recently introduced modifications of these models are a promising area for improved results. Examples are the Regression-enhanced random forests, introduced by Zhang, Nettleton, and Zhu (2019), which incorporates the benefits of penalized parametric regression. Another example is the macroeconomic random forest used to model parameters in a linear macro equation (Goulet Coulombe, 2020). And, lastly, exploration of local linear forests, introduced by Friedberg et al. (2020), designed to better capture smoothness of signals, is expected to produce interesting advancements.

The biggest downside to the use of machine learning in financial applications is the lack of transparency (Moritz and Zimmermann, 2016). Even though machine learning produces desirable results, no economic theory, associations or equilibria can be derived or substantiated based on the obtained values. In future research the understanding of economic mechanisms can be better examined by using different methods to portray the machine learning results. This adds to the meaningfulness in economic context of the obtained results from the new model advancements. In this relatively new area, we expect further research into the explanation of the workings of machine learning models to give an indication of the driving economic correlations that are important for optimised asset pricing.

References

Barberis, N., Greenwood, R., Jin, L., & Shleifer, A. (2015). X-capm: An extrapolative capital asset pricing model. *Journal of financial economics*, *115*(1), 1–24.

Best, P., & Byrne, A. (2001). Measuring the equity risk premium. *Journal of Asset Management*, *1*(3), 245–256.

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

Cai, Y., Hang, H., Yang, H., & Lin, Z. (2020). Boosted histogram transform for regression. *International Conference on Machine Learning*, 1251–1261.

Carhart, M. M. (1997). On persistence in mutual fund performance. *The Journal of finance*, *52*(1), 57–82.

Chen, T., & Guestrin, C. (2016). Xgboost: A scalable tree boosting system. *Proceedings of the 22nd acm sigkdd international conference on knowledge discovery and data mining*, 785–794.

Cochrane, J. H. (2009). *Asset pricing: Revised edition*. Princeton university press.

Collot, S., & Hemauer, T. (2021). A literature review of new methods in empirical asset pricing: Omitted-variable and errors-in-variable bias. *Financial Markets and Portfolio Management*, *35*, 77–100.

Diebold, F. X., & Mariano, R. S. (2002). Comparing predictive accuracy. *Journal of Business & economic statistics*, *20*(1), 134–144.

Diebold, F. X., Mariano, R. S. et al. (1991). *Comparing predictive accuracy i: An asymptotic test.* Institute for Empirical Macroeconomics, Federal Reserve Bank of Minneapolis ...

Dietterich, T. G. (2000). An experimental comparison of three methods for constructing ensembles of decision trees: Bagging, boosting, and randomization. *Machine learning*, *40*(2), 139–157.

Drucker, H. (1997). Improving regressors using boosting techniques. *ICML*, *97*, 107–115.

Fama, E. F., & French, K. R. (2008). Dissecting anomalies. *The Journal of Finance*, *63*(4), 1653–1678.

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

Fama, E. F., & French, K. R. (2021a). *Common risk factors in the returns on stocks and bonds.* University of Chicago Press.

Fama, E. F., & French, K. R. (2021b). *Multifactor explanations of asset pricing anomalies.* University of Chicago Press.

Fama, E. F., & MacBeth, J. D. (1973). Risk, return, and equilibrium: Empirical tests. *Journal of political economy*, *81*(3), 607–636.

Freund, Y., & Schapire, R. E. (1997). A decision-theoretic generalization of on-line learning and an application to boosting. *Journal of computer and system sciences*, *55*(1), 119–139.

Friedberg, R., Tibshirani, J., Athey, S., & Wager, S. (2020). Local linear forests. *Journal of Computational and Graphical Statistics*, 1–15.

Friedman, J. H. (2001). Greedy function approximation: A gradient boosting machine. *Annals of statistics*, 1189–1232.

Friedman, J. H. (2002). Stochastic gradient boosting. *Computational statistics & data analysis*, *38*(4), 367–378.

Geurts, P., Ernst, D., & Wehenkel, L. (2006). Extremely randomized trees. *Machine learning*, *63*(1), 3–42.

Giglio, S., & Xiu, D. (2021). Asset pricing with omitted factors. *Journal of Political Economy*, *129*(7), 000–000.

Goulet Coulombe, P. (2020). The macroeconomy as a random forest. *Available at SSRN 3633110.*

Green, J., Hand, J. R., & Zhang, X. F. (2013). The supraview of return predictive signals. *Review of Accounting Studies*, *18*(3), 692–730.

Green, J., Hand, J. R., & Zhang, X. F. (2017). The characteristics that provide independent information about average us monthly stock returns. *The Review of Financial Studies*, *30*(12), 4389–4436.

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

Harvey, C. R., & Liu, Y. (2021). Lucky factors. *Journal of Financial Economics.*

Harvey, C. R., Liu, Y., & Zhu, H. (2016).
and the cross-section of expected returns. *The Review of Financial Studies*, *29*(1), 5–68.

Hastie, T., Tibshirani, R., & Friedman, J. (2009). *The elements of statistical learning: Data mining, inference, and prediction.* Springer Science & Business Media.

Holmström, B., & Tirole, J. (2001). Lapm: A liquidity-based asset pricing model. *the Journal of Finance*, *56*(5), 1837–1867.

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

Hull, I. (2021). Machine learning for economics and finance in tensorflow 2.

Ian, H. W., & Eibe, F. (2005). Data mining: Practical machine learning tools and techniques.

Jagannathan, R., & Wang, Z. (2002). Empirical evaluation of asset-pricing models: A comparison of the sdf and beta methods. *The Journal of Finance*, *57*(5), 2337–2367.

Ke, G., Meng, Q., Finley, T., Wang, T., Chen, W., Ma, W., Ye, Q., & Liu, T.-Y. (2017). Lightgbm: A highly efficient gradient boosting decision tree. *Advances in neural information processing systems*, *30*, 3146–3154.

Keim, D. B., & Stambaugh, R. F. (1986). Predicting returns in the stock and bond markets. *Journal of financial Economics*, *17*(2), 357–390.

Kelly, B. T., Pruitt, S., & Su, Y. (2019). Characteristics are covariances: A unified model of risk and return. *Journal of Financial Economics*, *134*(3), 501–524.

Koijen, R. S., & Van Nieuwerburgh, S. (2011). Predictability of returns and cash flows. *Annu. Rev. Financ. Econ.*, *3*(1), 467–491.

Lau, S. T., Ng, L., & Zhang, B. (2012). Information environment and equity risk premium volatility around the world. *Management Science*, *58*(7), 1322–1340.

Lewellen, J. (2014). The cross section of expected stock returns. *Forthcoming in Critical Finance Review, Tuck School of Business Working Paper*, (2511246).

Lo, A. W., & MacKinlay, A. C. (1990). Data-snooping biases in tests of financial asset pricing models. *The Review of Financial Studies*, *3*(3), 431–467.

McLean, R. D., & Pontiff, J. (2016). Does academic research destroy stock return predictability? *The Journal of Finance*, *71*(1), 5–32.

Medeiros, M. C., Vasconcelos, G. F., Veiga, Á., & Zilberman, E. (2021). Forecasting inflation in a data-rich environment: The benefits of machine learning methods. *Journal of Business & Economic Statistics*, *39*(1), 98–119.

Merton, R. C. (1973). An intertemporal capital asset pricing model. *Econometrica: Journal of the Econometric Society*, 867–887.

Mingers, J. (1989). An empirical comparison of selection measures for decision-tree induction. *Machine learning*, *3*(4), 319–342.

Moritz, B., & Zimmermann, T. (2016). Tree-based conditional portfolio sorts: The relation between past and future stock returns. *Available at SSRN 2740751.*

Pesaran, M. H., & Timmermann, A. (1995). Predictability of stock returns: Robustness and economic significance. *The Journal of Finance*, *50*(4), 1201–1228.

Rapach, D., & Zhou, G. (2013). Forecasting stock returns. *Handbook of economic forecasting* (pp. 328–383). Elsevier.

Ross, S. A. (1976). The arbitrage theory of capital asset pricing. *Journal of Economic Theory*, *13*(3), 341–360. `https://doi.org/https://doi.org/10.1016/0022-0531(76)90046-6`

Subrahmanyam, A. (2010). The cross-section of expected stock returns: What have we learnt from the past twenty-five years of research? *European Financial Management*, *16*(1), 27–42.

Torous, W., & Valkanov, R. (2000). Boundaries of predictability: Noisy predictive regressions.

Wang, Y., & Feng, L. (2020). Improved adaboost algorithm for classification based on noise confidence degree and weighted feature selection. *IEEE Access*, *8*, 153011–153026.

Weigand, A. (2019). Machine learning in empirical asset pricing. *Financial Markets and Portfolio Management*, *33*(1), 93–104.

Welch, I. (2000). Views of financial economists on the equity premium and on professional controversies. *The Journal of Business*, *73*(4), 501–537.

Welch, I., & Goyal, A. (2008). A comprehensive look at the empirical performance of equity premium prediction. *The Review of Financial Studies*, *21*(4), 1455–1508.

Weymaere, N., & Martens, J.-P. (1991). A fast and robust learning algorithm for feedforward neural networks. *Neural Networks*, *4*(3), 361–369.

Wyner, A. J. (2003). On boosting and the exponential loss. *AISTATS.*

Zhang, H., Nettleton, D., & Zhu, Z. (2019). Regression-enhanced random forests. *arXiv preprint arXiv:1904.10416.*

Appendices

A Details of stock choice

The sample contains stocks with prices below \$5, share codes beyond 10 and 11, and financial firms. The largest pool of assets is chosen for multiple reasons. Firstly, it is not desirable to have stocks that are components of the S&P 500 index removed as these are of importance to the examination of asset pricing. Furthermore, since the index is predicted based on return predictions of various individual stocks, these stocks should not be excluded. Secondly, using this large selection of stocks preserves our results from sample selection or data-snooping biases. Lo and MacKinlay (1990), for example, highlight the importance of preventing data-snooping biases, as these are found to induce substantial changes in results for financial asset pricing models. Due to the use of a larger sample, the ratio of observation count is increased to parameter count, which helps prevent overfitting.

B Details of Characteristics

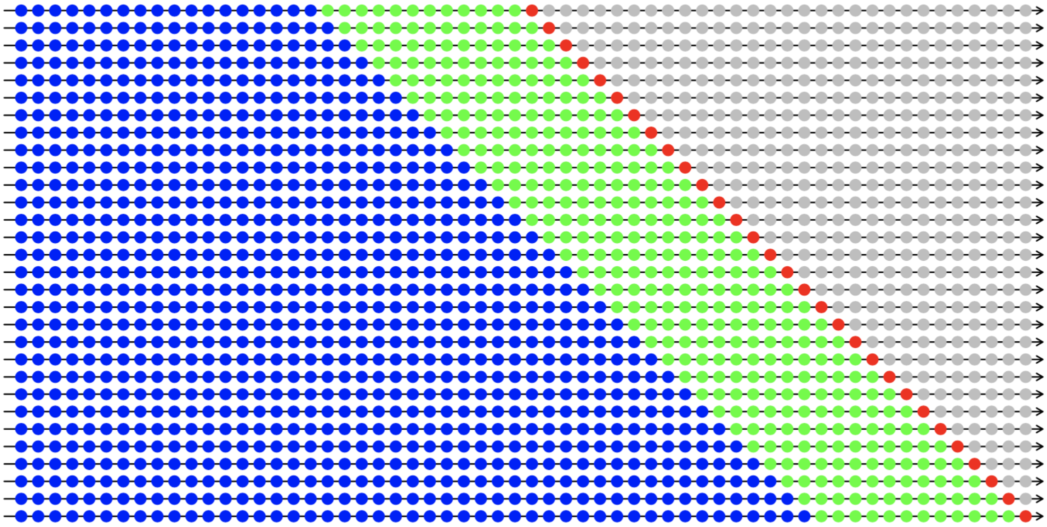
From the original 94 characteristics, 61 are updated annually, 13 are updated quarterly and 20 are updated monthly. The selection of characteristics is based on the cross-section of stock returns literature.

Table 3

Details of the characteristics

No.	Acronym	Firm characteristic	Paper's author(s)	Year,	Journal	Date Source	Frequency
1	age	# years since first Compustat coverage	Jiang, Lee & Zhang	2005	RAS	Compustat	Annual
2	baspread	Bid-ask spread	Amihud & Mendelson	1989	JF	CRSP	Monthly
3	beta	Beta	Fama & MacBeth	1973	JPE	CRSP	Monthly
4	betasq	Beta squared	Fama & MacBeth	1973	JPE	CRSP	Monthly
5	chmom	Change in 6-month momentum	Gettleman & Marks	2006	WP	CRSP	Monthly
6	convind	Convertible debt indicator	Valta	2016	JFQA	Compustat	Annual
7	dolvol	Dollar trading volume	Chordia, Subrahmanyam & Anshuman	2001	JFE	CRSP	Monthly
8	dy	Dividend to price	Litzenberger & Ramaswamy	1982	JF	Compustat	Annual
9	idiovol	Idiosyncratic return volatility	Ali, Hwang & Trombley	2003	JFE	Compustat	Monthly
10	ill	Illiquidity	Amihud	2002	JFM	CRSP	Monthly
11	indmom	Industry momentum	Moskowitz & Grinblatt	1999	JF	CRSP	Monthly
12	maxret	Maximum daily return	Bali, Cakici & Whitelaw	2011	JFE	CRSP	Monthly
13	mom1m	1-month momentum	Jegadeesh & Titman	1993	JF	CRSP	Monthly
14	mom6m	6-month momentum	Jegadeesh & Titman	1993	JF	CRSP	Monthly
15	mom12m	12-month momentum	Jegadeesh	1990	JF	CRSP	Monthly
16	mom36m	36-month momentum	Jegadeesh & Titman	1993	JF	CRSP	Monthly
17	mvel1	Size	Banz	1981	JFE	CRSP	Monthly
18	nincr	Number of earnings increases	Barth, Elliott & Finn	1999	JAR	Compustat	Quarterly
19	rd_mve	R&D to market capitalization	Guo, Lev & Shi	2006	JBFA	Compustat	Annual
20	retvol	Return volatility	Ang, Hodrick, Xing & Zhang	2006	JF	CRSP	Monthly
21	securedind	Secured debt indicator	Valta	2016	JFQA	Compustat	Annual
22	sp	Sales to price	Hong & Kacperczyk	2009	JFE	Compustat	Annual
23	std_turn	Volatility of liquidity (share turnover)	Chordia, Subrahmanyam, &Anshuman	2001	JFE	CRSP	Monthly
24	turn	Share turnover	Datar, Naik & Radcliffe	1998	JFM	CRSP	Monthly
25	zerotrade	Zero trading days	Liu	2006	JFE	CRSP	Monthly

Figure 6. Sample splitting over years ranging from 1957 to 2016 (Gu, Kelly, and Xiu, 2020)



C Details of variables

In table 4, details on the macroeconomic variables, as used by Welch and Goyal (2008) is given.

Table 4

Details of the macroeconomic variables

No.	Acronym	Macroeconomic variable	Date Source	Frequency
1	dp	dividend-price ratio	Robert Shiller's website, S&P Corporation	Monthly
2	ep	earnings-price ratio	Robert Shiller's website, S&P Corporation	Monthly
3	bm	book-to-market ratio	Value Line's website	Monthly
4	ntis	net equity expansion	CRSP	Monthly
5	tbl	treasury-bill rate	FRED	Quarterly
6	tms	term spread	NBER, Ibbotson's yearbook	Monthly
7	dfy	default spread	FRED	Monthly
8	svar	stock variance	G. William Schwert, CRSP	Monthly

C.1 Variable interactions

The following variable interactions are incorporated in the model estimation: mom1m×bm, mom1m×C, mom1mt×bl, mom1m×dp, turn×ntis, maxret×ntis, retvol×tbl, chmom×bm, mvel1×tms, maxret×tbl, retvol×ntis, mom12m×tbl, mom1m×ntis, idiovol×tbl, mom12m×dp, indmom×tms, indmom×tbl, mvel1×C, nincr×bm and sp×tms. The choice is based on the interactions that Gu, Kelly, and Xiu (2020) have found to be the most important interactions of stock characteristics with macroeconomic variables.

D Sample Selection

The sample splitting scheme is adopted from Gu, Kelly, and Xiu (2020). A recursive strategy is used to train the model. For this purpose, the training-, validation- and out of sample samples

Table 5

Hyperparameters for all methods

Random Forest		**Ada Boosted**		**Extremely Randomised**		**Histogram-based GBRT**	
#Trees	300	#Trees	$\in \{200, 250\}$	#Trees	$\in \{250, 500, 1000\}$	Maximum #trees	*in* {1000, 2000}
Depth	1 - 5	Learning rate	$\in \{0.5, 1\}$	Depth	3	Depth	3
#Features per split	$\in \{17, 20, 23, 25\}$			#Features	$\in \{25, 30\}$	Learning rate	0.01

are redistributed for each out of sample year. The sample selection is explained by means of figure 6 from Gu, Kelly, and Xiu (2020), denoting the distribution of years over the three different samples. For the prediction of every out of sample year, the corresponding training en validation sample to which the model is estimated, are updated. The initial samples, used to estimate the model for prediction of observations in out of sample year 1987, denoted by the red dots, consist of: the training sample which ranges from 1957 to 1974 (12) years, denoted by the blue dots. And the validation sample, which ranges from 1975 to 1986, and therefore contains the subsequent 12 years. This sample is denoted by the green dots. After having estimated the model on the training sample, selected the optimal parameters and predicted the returns for the first out of sample year, a year is added to the training sample. Simultaneously, the validation sample rolls forward 1 year and using these samples the next out of sample year can be predicted. This causes the training sample to grow, while the validation sample maintains its size. This is done for all 30 out of sample years, denoted by the red and gray dots.

E Hyperparameter tuning

In order to avoid overfitting, which is a common problem in machine learning, hyperparameter tuning is used for estimation and testing of the models. As mentioned in the previous section. For each iteration the training and validation samples are redistributed. In order to select the best parameters for the out-of-sample prediction an optimisation problem is used over a fixed set of parameter options. The training sample is used to estimate the model based on a set of tuning parameters or hyperparameters. The performance of these parameters is examined using the predictive performance on the validatio sample. The set of parameters that results in the optimal forecasts is selected and used for out-of-sample prediction.

The hyperparameter options for the other models are determined based on the run time for a single year, with the hyperparameters of the random forest as starting point. The resulting set of hyperparameters per option is given by figure 5. For each combination of parameters, the model is estimated using the training sample. Thereafter the estimated model is used to predict the return values given by the validation sample. From this prediction the mean squared errors (MSE) are determined. The estimated model with the combination of parameters that gives the smallest MSE is consequently used to predict the corresponding out of sample year. The same approach is used for the following 30 out of sample years until the year 2016 is reached. Hyperparameter tuning is, therefore, very computationally intensive.

F Test statistics

F.1 R^2_{OOS}

We use the following formulation for the out-of-sample R^2:

$$R^2_{OOS} = 1 - \frac{\sum_{(i,t)\in T_3}(r_{i,t+1} - \hat{r}_{i,t+1})^2}{\sum_{(i,t)\in T_3} r^2_{i,t+1}},$$

here, T_3 denotes the testing sample, ensuring data involved in estimation or tuning is not included. Pairwise comparisons of the various methods is done through the Diebold, Mariano, et al. (1991) test. Here the historical mean returns that are normally present in the denominator are set to zero. This is done to avoid lowering the R^2_{OOS} statistic for a 'good' forecast as a consequence of a noisy historical mean stock return.

F.2 Diebold-Mariano

Comparing the statistical significance of the differences between the forecasts of the problem is done using the Diebold and Mariano (2002) (DM) test statistic. The statistic denotes the difference in square or absolute forecast error between the forecasts of two models. Forecast errors $\hat{e}^{(j)}_{i,t+1}$ for stock $i = 1...N$ at time $t = 1...T$ using model j, are given by the following formulation:

$$\hat{e}^{(j)}_{i,t} = \hat{r}_{i,t} - r_t,$$

in which $\hat{r}_{i,t}$ and r_t denote the return forecast and actual returns respectively. We use the adjusted Diebold-Mariano test statistic as is given by Gu, Kelly, and Xiu (2020). The test statistic is given by:

$$DM_{12} = \frac{\bar{d_{12}}}{\hat{\sigma}_{\bar{d}_{12}}},$$

in which $\bar{d_{12}}$ is the average over the observations in the testing sample from the following formulation:

$$d_{12,t+1} = \frac{1}{n_{3,t+1}} \sum_{i=1}^{n_{3,t+1}} \left(\left(\hat{e}^{(1)}_{i,t+1}\right)^2 - \left(\hat{e}^{(2)}_{i,t+1}\right)^2 \right).$$

Here $n_{3,t+1}$ is the number of stocks in the testing sample. $\hat{\sigma}_{\bar{d}_{12}}$ in the DM is the Newey-West standard error of $d_{12,t}$ over the testing sample.

The DM test statistic follows a standard normal distribution. With a 5% significance level the hypotheses of the models performing equally well is, therefore, rejected for values of DM $<$ -1.96 and DM $>$ 1.96.

F.3 Bonferroni correction

For each model comparison test there is a chance of making a type 1 error. A type 1 error means rejection the null hypotheses while it is true. The corresponding probability of making this error for a family of tests, depends on the number of comparison tests executed. The formula to obtain the family wise error rate ($FWER$) that follows from the family of comparisons is formulated as:

$$FWER = 1 - (1 - \alpha)^n. \tag{15}$$

Here α is the significance level and n the number of comparisons. This $FWER$ denotes the chance of discovering a false rejection of the null hypotheses. To control for this error that arises when performing multiple comparisons, a Bonferroni correction can be used. A Bonferroni-corrected α is used for the comparison tests given by:

$$\alpha_{corrected} = \frac{\alpha}{n}. \tag{16}$$

In our case the corrected alpha becomes $\frac{0.05}{7} = 0.00714$. This corresponds to a newly found z-values of 2.44999 to test our DM statistic to.