Erasmus University Rotterdam
Erasmus School of Economics
Bachelor Thesis BSc Econometrie & Operationele Research

Machine Learning in Asset Pricing: Predicting Equity Risk Premia using Neural Networks, Random Forests and Traditional Approaches

Max Dijkman (509960md)

Supervisor:	T. van der Zwan, MSc
Second assessor:	Dr. A. Quaini
Date final version:	July 2, 2023

Abstract

Historically, methods used to evaluate asset risk premia offer limited out-of-sample predictive power. Since the prediction of equity risk premia is a central component of empirical asset pricing, this thesis aims to adress this problem by performing a comparative analysis of methods to improve risk premia forecast quality. We focus on non-linear machine learning methods including neural networks and random forests, and compare these to traditional linear models including the Fama & French three factor model. Our results demonstrate the potential of machine learning for the field of empirical asset pricing. Neural networks are identified as the best performing methods, followed by a simple linear model with Huber loss function. Random forest does not meet expectations and the Fama & French model loses its explanatory power in the out-of-sample setting. We identify stock-level characteristics related to dividends and R&D ratios as most important for random forest model performance. Overall, harnessing neural networks for risk premia prediction shows considerable potential for improving financial decision-making.

Contents

1 Introduction

Evaluating equity risk premia is a fundamental component of asset pricing and therefore crucial for anyone looking to make informed investment decisions. Equity risk premia capture the excess return of an asset over the risk-free rate. These excess returns can be interpreted as the compensation investors receive for tolerating the risk they are exposed to when investing in a certain asset. Prediction of these risk premia has proven to be a considerable challenge due to several factors, including the complexity of financial markets and the occurrence of unexpected events influencing investor sentiment. Accurate risk premia forecasting allows investors, portfolio managers and financial institutions alike to improve financial decision making and is imperative to empirical asset pricing.

Despite its importance, methods historically used to predict these risk premia have drawbacks limiting their effectiveness. Traditional models like the Fama and French Three factor model - which expands on the Capital Asset Pricing Model (CAPM) - have many underlying assumptions and only use a fraction of the financial data we have available in modern society (Fama & French, 1992). Nevertheless, these traditional models have numerous advantages which is why they continue to be widely used in the financial industry. Models like the Fama & French three factor model (FF3) are relatively easy to implement and interpret due to their simple linear structure, while still offering a certain degree of explanatory power (Chiah et al., 2016).

Considering the importance, the question arises whether alternative methods exist that can enhance the prediction of asset risk premia. In the machine learning literature, many methods exist that allow for predictor interactions or non-linear relationships between predictors and asset returns. Employing such methods in the context of empirical asset pricing could potentially improve out-of-sample explanatory power by capturing more complex relationships in the data that are missed by traditional methods. A drawback of these machine learning methods is that their complex structure results in computationally expensive estimation. However, the rapid increase in computational power over the last few decades has enabled us to explore whether using more sophisticated machine learning models combined with the vast amounts of data we have available on stock returns can increase the out-of-sample explanatory power for excess stock returns.

This paper examines two prominent machine learning methods and compares them to traditional approaches for the problem of predicting asset risk premia. Our primary goal is to assess whether the machine learning methods we consider offer increased out-of-sample predictive power based on forecasts of monthly excess stock returns. We are particularly interested in the relative predictive performance of the methods, which we evaluate by means of the out-of-sample R^2 (R^2_{OOS}). By doing so, this paper aims to find whether machine learning can improve financial decision making by offering more accurate predictions of asset risk premia. This leads to our first research question:

> **RQ1** How do machine learning methods compare to traditional methods for forecasting asset risk premia in terms of out-of-sample predictive performance?

In addition, the secondary objective is to identify the driving factors of risk premia by ranking the predictor variables on their relative importance. Hence, our second research question is:

RQ2 Which predictor variables are most important for model performance?

In order to adress the research questions, we focus on random forests and feedforward neural networks and compare these sophisticated methods to the well-known Fama and French three factor model. Additionally we employ a simple linear model using the same set of predictor variables as the machine learning methods. Machine learning offers several advantages over the traditional methods used in this thesis. Firstly, the random forest and neural network allow for non-linear relationships to be captured which may improve forecast accuracy. Secondly, they are able to detect interactions between the predictor variables, while the simple linear models assume independence between predictors. Thirdly, machine learning methods are more robust to outliers, which is especially relevant in the application of return prediction since the distribution of stock returns is known to have fat tails (Officer, 1972).

This thesis builds on the work of Gu et al. (2020) by partly replicating their study and subsequently extending their work by incorporating the Fama & French three factor method as a traditional method to compare against. We perform a large-scale comparative analysis based on monthly data of nearly 6500 stocks over the years 1977 up to 2021, a total of 45 years. Our predictor set consists of 92 stock-level characteristics, 8 macroeconomic variables and 97 Standard Industrial Classification (SIC) code dummies. For the FF3 model, only the three Fama & French factors are used. Our results show great promise for applying machine learning to risk premia prediction. Specifically, neural networks perform well with an R^2_{OOS} of 3.35% for the prediction of monthly returns, a substantial improvement over the 1.23% R^2_{OOS} of the simple linear model containing the same predictor set. Random forest does not meet expectations, providing respectable in-sample accuracy but failing to generalize to the out-of-sample setting. With an out-of-sample R^2_{OOS} of -1.23%, RF underperforms compared to a benchmark prediction of zero for all stocks. Based on our empirical study, the FF3 model seems incapable of producing reliable forecasts and is thus better suited for explaining returns using non-lagged data. We use feature importance on the random forest models and identify the most influential stock-level characteristics to be related to dividends and R&D ratios. Additionally, the convertible debt indicator - and sin stock variable rank among the most important predictors.

Our findings show the potential of machine learning for predicting excess stock returns, which may provide a new set of methods to help improve our understanding of risk premia behaviour.

2 Literature

2.1 Forecasting stock returns

There are two main approaches used in the empirical literature to predict stock returns. The first focuses on estimating the relationship between expected returns and stock-level characteristics, as exemplified by Lewellen (2014). This is usually done by running cross-sectional regressions of returns on a select number of lagged stock-level characteristics (Gu et al., 2020).

The second approach for predicting stock returns views the returns as a time series and typically uses a select set of macroeconomic variables to predict portfolio returns. D. Rapach & Zhou (2013) provide a survey that compares methods on their out-of-sample predictive performance.

This paper focuses on the first approach by cross-sectionally predicting stock returns using several machine learning models and traditional methods, which we compare based on their out-of-sample forecasting ability. There is limited evidence of out-of-sample predictability of returns (Phan et al., 2015). For instance, Welch & Goyal (2008) use a range of macroeconomic predictors and financial ratios to evaluate numerous regression models for the out-of-sample prediction of the equity premium. They find that all forecasts fail to beat the historical average benchmark.

2.2 Machine Learning

In recent years, there has been a surge in the application of machine learning methods because of the increased robustness and predictive accuracy that these methods offer compared to traditional regression-based models (Goldstein et al., 2017). The attractive features that machine learning methods possess combined with significant advancements in computational power in recent years, make it possible to explore the potential of machine learning for complex predictive problems.

The predictive power of machine learning has been proven in many fields. For instance, Kourou et al. (2015) discusses how machine learning may improve cancer prognosis, and Fujiyoshi et al. (2019) explores how image recognition using deep learning can be used to realize autonomous driving.

Since the problem of empirical asset pricing is fundamentally a matter of prediction, we explore whether harnessing machine learning for evaluating asset risk premia can provide an improvement in out-of-sample forecasts.

Machine learning has been applied before in the context of empirical asset pricing. D. E. Rapach et al. (2013) use lasso to predict global equity market returns using lagged returns per country. Moreover, Krollner et al. (2010) conduct a survey of machine learning methods used for stock index forecasting and identify Artificial Neural Networks as the dominant technique.

While machine learning may potentially provide more accurate forecasts compared to linear methods, equity risk premia are notoriously hard to predict. Although the explainable component in stock returns may be minimal, even models that can offer an R^2_{OOS} of around 1% can hold economic relevance (D. Rapach & Zhou, 2013).

2.3 Traditional Methods

Traditional models that are more common in the cross-sectional stock prediction literature include the capital asset pricing model and the fama & french factor models including three or five factors. Parsimonious models like the FF3 model are widely used in practice, with applications ranging from analyzing hedge fund performance (Capocci & Hübner, 2004), to portfolio construction (Eraslan, 2013). The FF3 model, containing three factors, expands on the CAPM which only includes a market risk factor. By including two additional factors, Fama & French (1992) aim to improve the predictive accuracy while maintaining model sparsity. More recently, Fama & French (2015) further expand on the FF3 model by considering a total of five factors, which results in improved performance compared to the previous three factor model.

3 Data and Over-arching Model

We obtain monthly individual stock returns from the CRSP over the period starting from January 1977 up to December 2021, a total of 45 years. The number of stocks we consider in our sample is almost 6500, with the average number of stocks per month being nearly 1600. In total, our data consists of over 830.000 observations. To calculate individual excess returns, we use the 10 year Treasury-bill rate to proxy for the risk-free rate.

In contrast to Gu et al. (2020), this paper considers 92 stock-level predictive characteristics instead of 94 to limit the number of missing characteristics in the data. Any missing characteristics are imputed by means of the local B-XS model detailed in Bryzgalova et al. (2022). The stock characteristics are updated either monthly, quarterly or annually. Since most of the characteristics are available to the public with a delay, we aim to prevent a forward-looking bias by assuming that monthly characteristics are delayed by at most one month, quarterly with at least four months lag, and annual with at least six months lag.

Before letting these stock-level characteristics enter the model as predictor variables, we cross-sectionally rank each characteristic and map the resulting ranks to the [-1,1] interval (Freyberger et al., 2020). We rank the characteristics because we are more interested in the relative rank of a characteristic in the cross-section instead of the absolute value. For example, firm size may increase over time, and since our models are not time dependent, its more important what the relative size of the firm is at a certain point of time in the cross-section. Ranking the characteristics in this manner provides a more straightforward interpretation of the variable observations. In addition to these ranked characteristics, we have 97 industry dummies represented by the first two digits of Standard Industrial Classification (SIC) codes.

In addition to stock-level characteristics, we construct eight macroeconomic predictor variables according to the definitions and data from Welch & Goyal (2008). The macroeconomic predictors we include are: dividend-price ratio, earnings-price ratio, Treasury-bill rate, term spread, default spread, stock variance, net equity expansion and book-to-market ratio.

Finally, we obtain the monthly Fama and French factors for the same time period from the website of Kenneth French. We obtain the market risk (Mkt-RF), size (SMB) and value (HML) factors. Since these are monthly factors we follow the same approach as for the other predictors and lag them by one month.

We denote the vector of predictor variables for our methods by $z_{i,t}$. For the Fama and French three factor model, $z_{i,t}$ consists of the Mkt-RF, SMB and HML factors. For the other three methods we focus on, $z_{i,t}$ consists of 92 stock-level characteristics, 8 macroeconomic predictors and 97 SIC industry dummies.

Next to evaluating the methods based on their ability to predict monthly stock returns, we also perform an analysis at the annual level. We construct annual returns each month which we then use as the target variable for our methods to predict, instead of the monthly returns from the CRSP. Annual returns are calculated each month by computing the compounded return of a stock over the next 12 months, essentially corresponding to a holding period of 12 months. If a stock at time t is missing one or more observations somewhere over the next 12 months, we exclude that stock from the annual analysis at time t.

The total dataset of 45 years is divided into three samples with the following initial sizes:

18 years of training (1977 - 1994), 12 years of validation (1995 - 2006) and 1 year of testing out-of-sample (2007). Every year, we increase the training sample by one year and roll forward the validation and test samples by a year after which we refit and re-evaluate the models. Thus, we use a total of 15 years for out-of-sample testing. A visual representation of the training/validation/test split is provided as appendix figure 7.

4 Methodology

We focus on a total of four methods consisting of two machine learning methods and two traditional methods. For the machine learning methods, we use a tree-based ensemble method (Random Forest) and an artificial neural network. We compare these against a simple linear model including the same predictor set, and the Fama & French three factor model (Fama & French, 1992).

This section discusses all methods in terms of model definition, objective function and tuning the hyperparameters. Each subsection corresponding to a method first defines the general statistical model, then describes the objective function and finally briefly discusses how this paper aims to determine the optimal specification among the considered options.

Similarly to Gu et al. (2020), all estimates share the same objective of minimizing the Mean Squared Prediction Error (MSE). Variations on the MSE are employed to introduce regularization with the goal of improving the model's out-of-sample performance by reducing risk of overfitting and being more robust to outliers.

In its most general form, the model is specified as an additive prediction error model (Gu et al., 2020):

$$r_{i,t+1} = E_t(r_{i,t+1}) + \epsilon_{i,t+1}, \tag{1}$$

where

$$E_t(r_{i,t+1}) = g^*(z_{i,t}). \tag{2}$$

Here, stocks are indexed by $i = 1, ..., N_t$ months are indexed as $t = 1, ...T$. We aim to estimate a function $g^*(\cdot)$ which represents the conditional expected excess return function, that takes a set of predictor values at time t as input to make a prediction for $r_{i,t+1}$ and minimizes the out-of-sample errors, $\epsilon_{i,t+1}$. The function $g^*(\cdot)$ is independent of the individual stock and time. This way, we are able to leverage information from the entire panel instead of restricting the model to only use information specific to a single stock or point in time.

Each of the methods we consider has the goal of approximating the over-arching model $E_t(r_{i,t+1}) = g^*(z_{i,t})$ described in equation (2).

4.1 Training, Tuning and Testing

The predictive advantages associated with machine learning methods also come with the challenge of having to make more decisions about how to tune all of the hyperparameters corresponding to the model. Hyperparameters in machine learning are crucial to the performance of the model since they control regularization of the estimators which is supposed to prevent

overfitting and improve out-of-sample performance (Feurer & Hutter, 2019).

This paper follows the most common approach in the literature for similar applications (Gu et al., 2020) and splits the data in three disjoint sets maintaining the temporal order of the data. The sets will be referred to as the training, validation and testing set. The training set is used to estimate the initial model parameters for a model subject to a specific set of hyperparameter values.

The second sample, or validation set, is used for tuning the hyperparameter values to find the optimal settings for this particular application. Predictions are made for the data points in the validation sample for all considered hyperparameter combinations using the estimated model obtained from the training sample. Next, we evaluate the performance of the models with all hyperparameter settings by computing the MSE. For each model, we pick the hyperparameter combination that has the lowest MSE as the tuned hyperparameters. The hyperparameter values that we set or tune are provided in Table 4 in the appendix.

After obtaining the tuned hyperparameters, we use these settings to re-estimate the model parameters based on the training and validation set combined. We then have the estimated model with the tuned hyperparameters, which we use to predict on the test set. The predictions on the test set are truly out-of-sample, since it is not used for estimation nor tuning. After one iteration, we increase the size of the training set and roll forward the validation and test sets by the most recent twelve months. More details about the motivation behind this approach are highlighted in Gu et al. (2020).

Since Gu et al. (2020) find that shallow learning outperforms deep learning for the prediction of equity risk premia, we focus on shallow model structures when tuning the hyperparameters.

4.2 Simple Linear

For this comparative study we are interested in evaluating the potential of employing sophisticated machine learning models in the context of asset pricing. To be able to accurately assess this potential, we use a simple linear method as a benchmark. While it is expected that the model will perform badly in a high-dimensional setting compared to the machine learning methods that we study, simple linear methods have been used historically to approach these kinds of modelling challenges. The Capital Asset Pricing Model (CAPM), for example, is essentially a simple linear model with a single factor (Fama & French, 1996).

The most commonly used method to estimate the parameters in a simple linear model is Ordinary Least Squares (OLS). Since OLS assumes normally distributed errors with constant variance and gives equal weights to all observations, we expect it to perform poorly in our setting where we predict returns that are known to be fat-tailed. We address this issue by considering an alternative to OLS that is known to be more robust to outliers, the Huber loss function. Huber assigns lower weights to large residuals to negate the influence of extreme values.

4.2.1 Model

The simple linear model fits a linear function of the predictor variables $z_{i,t}$ to the returns with the goal of optimizing the specified objective function. Thus, with θ being the parameter vector, the model becomes:

$$g^*(z_{i,t};\theta) = z'_{i,t}\theta. \tag{3}$$

This model does not allow for any nonlinear effects or predictor interactions, making it straightforward to interpret. The predictor variable vector for this model, $z_{i,t}$, contains all stock-level characteristics, macroeconomic predictors and SIC industry dummies.

4.2.2 Huber Robust Objective Function

As mentioned, the most common objective function for a simple linear model is the OLS function. This function has the advantage of not having to tune any parameters and is thus easily implemented. However, there are also some downsides to ordinary least squares, especially when dealing with noisy data containing many outliers. Stock returns are known to have a low signal-to-noise ratio and fat tails, indicating the presence of outliers (Officer, 1972). In order to limit these harmful effects, we consider the Huber robust objective function (Huber, 1992) presented below as an alternative to the standard least squares objective function.

$$\mathcal{L}_H(\theta) = \frac{1}{NT}\sum_{i=1}^{N}\sum_{t=1}^{T} H(r_{i,t+1} - g(z_{i,t};\theta), \varepsilon), \tag{4}$$

where

$$H(x;\varepsilon) = \begin{cases} x^2, & \text{if } |x| \leq \varepsilon; \\ 2\varepsilon|x| - \varepsilon^2, & \text{if } |x| > \varepsilon. \end{cases} \tag{5}$$

For relatively small errors, the Huber loss function is the regular squared loss. However, when an error is larger than the tuning parameter ε, we consider it relatively large and use an absolute loss. The tuning parameter ε is optimized using the validation sample and determines the degree of regularization.

4.3 Fama and French Three Factor Model

The Fama and French three factor (FF3) model is identical to the simple linear method in terms of model structure and estimation. This means that the FF3 model has the same linear structure and is estimated by applying the Huber loss function.

The difference is that $z_{i,t}$ only contains three predictor variables - or factors - for the FF3 model. These factors are designed to capture various sources of risk and explain excess returns using a simple linear structure.

Since the FF3 is mainly intended for explaining the stock return and not necessarily forecasting, we explore whether the model can also be useful for prediction by lagging the predictors by one month.

The FF3 factors are market risk (Mkt-RF), size (SMB) and value (HML). The market risk factor represents the excess return of the market, and is computed by taking the overall market (S&P500) return, and subtracting the risk-free rate. To proxy for the risk-free rate, the one month T-bill return is used. The factor loading for the Mkt-RF factor tends to be high for stocks that are more sensitive to market movements.

The size factor measures the difference in stock returns between small and large companies based on their market capitalization. Historically, the SMB factor is likely to be positive, indicating that small companies tend to produce higher returns compared to large companies.

Lastly, the value factor measures the return difference for value and growth stocks based on price-to-book ratios. Historically, value stocks (low price-to-book ratio) tend to outperform growth stocks (high price-to-book ratio). When value stocks outperform growth stocks, the HML coefficient has a positive sign.

Reducing the dimensionality of the predictor set to these three variables has several potential advantages. Firstly, it makes the model highly interpretable. Moreover, a large predictor set may increase the chance of overfitting on the training sample and negatively impact out-of-sample performance.

4.4 Random Forest

The random forest (RF) is a tree-based, ensemble method used for classification and regression that combines the outputs of multiple decision trees to reach a single result (Breiman, 2001). Random forest is a variation on 'bagging' (Breiman, 2001), a procedure introduced to stabilize predictive performance by averaging over multiple predictions that are based on different bootstrap samples. Random forests are based on a variation of bagging specifically designed to reduce correlation among trees in different bootstrap samples. This method which reduces correlation is called 'dropout'.

An advantage of RF is a low chance of overfitting compared to a single decision tree due to its averaging component. Furthermore, RF is not sensitive to outliers in the training data compared to a single tree. Lastly, RF is known to be suitable for large high-dimensional datasets, which is promising given the dimensions of our dataset for this application.

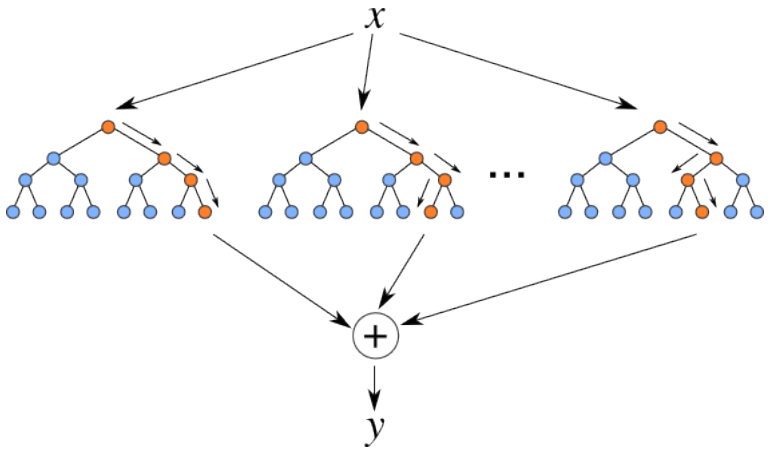


Figure 1: Random Forest Regression is an ensemble method combining the predictions of multiple decision trees into a single output by taking the average (y) of the predicted values

Figure 1 provides a visualisation of the process behind a random forest, where x represents the vector of predictor variables that we feed into the model and y is the combined averaged output from all the trees.

4.4.1 Model

As mentioned, the random forest averages predictions of multiple trees to reach a single result. A more detailed description of regression trees can be found in Gu et al. (2020).

For a single decision tree with K terminal nodes and depth L, the model can be formally written as

$$g(z_{i,t};\theta,K,L)=\sum_{k=1}^{K}\theta_k\mathbf{1}_{\{z_{i,t}\in C_k(L)\}}, \tag{6}$$

where $C_k(L)$ is one of K partitions of the data. A more detailed model description can be found in Breiman (2001).

4.4.2 Objective Function

In each RF iteration, we use bootstrapping to sample (with replacement) various training samples from the training set. The training set is split into a bootstrap sample and an Out-Of-Bag (OOB) sample. For every bootstrap sample, a tree is constructed and trained independently where every tree can contain a different subset of features. Features are randomly selected for each split in the tree, which improves efficiency and prediction power due to lowered correlation among trees. For every OOB sample, the model's prediction is made using all decision trees which do not contain that particular OOB sample in their bootstrap sample. The resulting predictions are compared to the actual target values and the OOB error is computed. This is unbiased since every tree in the RF is tested on data that has not been used in the training phase. All predictions are combined and the average is taken as the final prediction result. There are several hyperparameters that have to be set before starting the training procedure. Given the computational time we fix the number of trees in the ensemble to 100 and focus on tuning the depth of the trees and the number of features randomly sampled at each split. We optimize the amount of features randomly sampled at each split to minimize the Out-Of-Bag estimate of the error rate.

Details of the hyperparameter tuning values can be found in the appendix. We implement the Random Forest regressor in python through SKLearn.

4.5 Feedforward Neural Network

The second machine learning method we use is an Artificial Neural Network (ANN). The ability of ANNs to establish relationships amongst highly non-linear anomalous variables and produce accurate results to complex problems through learning makes them one of the most promising methods currently available in machine learning Jain et al. (1996). One of the most basic ANNs is the Feedforward Neural Network (FFNN), which this paper focusses on.

4.5.1 Model

The FFNN consists of multiple layers, where each layer contains a number of neurons that are connected with the layers before and after them (Svozil et al., 1997). ANNs owe their name to their similarity to the human brain. The name 'Feedforward' comes from the trait that the

data flows through the network from the input layer to the output layer without feedback loops. A simplified visualisation of such a Feedforward Neural Network with a single hidden layer is provided below.

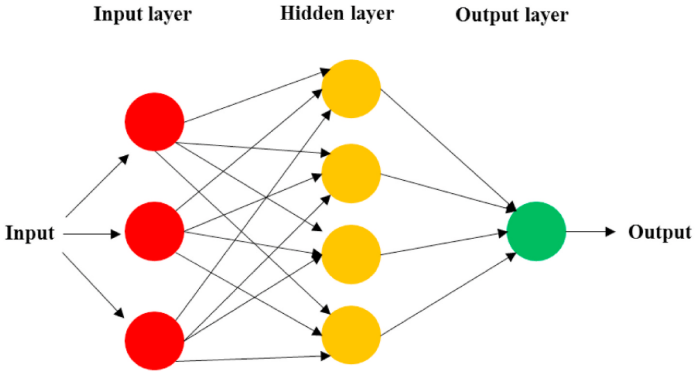


Figure 2: A Visualisation of a simple Feedforward Neural Network

The input layer consists of the raw predictors, where the number of units in the input layer equals the number of predictors. The hidden layers interact with each other in a forward flow, and non-linearly transform the data. Hidden layers consist of groups of neurons, illustrated by the circles in Figure 2.

Selecting the optimal architecture of a neural network is a challenging task. In this case we focus on a network with three hidden layers, as was found optimal in a similar application (Gu et al., 2020). The number of neurons in each layer are 32, 16 and 8 respectively, and are chosen according to the geometric pyramid rule (Masters, 1993).

We use the same activation function at all nodes, the rectified linear unit (ReLU) function, defined as

$$ReLU(x) = \begin{cases} 0, & \text{if } x < 0 \\ x, & \text{otherwise.} \end{cases} . \tag{7}$$

To describe the general statistical model, we first define $K^{(l)}$ as the number of neurons in each layer, l. Next, the output of neuron k in layer l is defined as $x_k^{(l)}$. The output vector of this layer is then defined as $x^{(l)} = (1, x_1^{(l)}, ..., x_{K^{(l)}}^{(l)})'$. The input layer of raw predictors, which is needed to initialize the neural network, is defined as $x^{(0)} = (1, z_1, ..., z_N)'$. It then holds for layer $l > 0$ that the output formula for each neural in that layer is

$$x_k^{(l)} = ReLU(x^{(l-1)'}\theta_k^{(l-1)}), \tag{8}$$

recursively resulting in the final output

$$g(z;\theta) = x^{(L-1)'}\theta^{(L-1)} \tag{9}$$

4.5.2 Objective Function

The FFNN is estimated using the Adam optimizer with the aim of minimizing the MSE. We tune the learning rate and the l1 penalty, which limits the size of the coefficients as a form of regularization. Another regularization method we use is early stopping. Early stopping is implemented using the mean squared error for monitoring. Lastly, we employ an ensemble method using an ensemble of 2, where we average the predictions within the ensemble to reduce prediction variance. Other parameter settings were fixed and can be found in the appendix. The FFNN is implemented in python using Keras from Tensorflow.

4.6 Performance Evaluation

4.6.1 R-Squared out-of-sample

This paper evaluates model performance by calculating the out-of-sample R^2 based on the test set for each model with tuned hyperparameters. The advantage of this performance measure is that we pool all prediction errors from different stocks over time into one number which is easy to interpret. We compute the out-of-sample R^2 as

$$R^2_{OOS} = 1 - \frac{\sum_{(i,t)\in T_3}(r_{i,t+1} - \hat{r}_{i,t+1})^2}{\sum_{(i,t)\in T_3} r^2_{i,t+1}} \tag{10}$$

where $\hat{r}_{i,t+1}$ denotes the predicted value for stock i at time $t+1$ and $r_{i,t+1}$ is the actual excess return. It is important that T_3 is defined as the test set, such that our R^2_{OOS} does not consider any data points that have been used to train or validate the models.

Similarly to Gu et al. (2020), this paper considers the R^2_{OOS} without demeaning in the denominator (sum of squared excess returns). The reason for this being that the historical mean stock return is generally too noisy, causing it to assign a higher R^2_{OOS} value to each of the methods and giving a flawed representation of model performance. Since a naive forecast of zero is typically superior to the historical average excess return when predicting excess stock returns, we benchmark our R^2_{OOS} against a prediction value of zero.

When comparing models, the model with the highest R^2_{OOS} is preferred. Since we fit the models based on the training and validation data, negative R^2_{OOS} values can exist as we are predicting on a distinct test set. A negative R^2_{OOS} can be interpreted as underperformance compared to predictions of zero.

4.6.2 Diebold-Mariano Tests

On top of comparing methods based on their R^2, we perform pairwise comparisons using the Diebold and Mariano (DM) test for comparing tests on their predictive accuracy Diebold & Mariano (2002). The DM test evaluates the difference in prediction errors between two sets of forecasted values and returns a test statistic that tells us which forecast performs best between the two. Additionally, we compute the p-value to assess whether the performance difference is significant. Considering that excess stock returns in the cross-section are likely to have strong error dependence, this thesis considers an alternative implementation of the DM test that meets the requirement of weak error dependence. Where the regular DM test compares errors among

individual returns, we compare the cross-sectional average of prediction errors for each model. Thus, to test whether method (1) produces better out-of-sample forecasts than method (2), we define the DM test statistic as

$$DM_{12} = \bar{d}_{12}/\hat{\sigma}_{\bar{d}_{12}}, \tag{11}$$

where

$$d_{12,t+1} = \frac{1}{n_{3,t+1}} \sum_{i=1}^{n_3} ((\hat{e}_{i,t+1}^{(1)})^2 - (\hat{e}_{i,t+1}^{(2)})^2). \tag{12}$$

The prediction error for stock i at time t for model (1) and (2) are denoted by $\hat{e}_{i,t+1}^{(1)}$ and $\hat{e}_{i,t+1}^{(2)}$ respectively. $n_{3,t+1}$ is the number of stocks in the test set at year $t+1$. Thus, $\bar{d}_{12}$ is defined as the mean of $d_{12,t}$ from the testing sample, and $\hat{\sigma}_{\bar{d}_{12}}$ as the Newey-West standard error. This adjusted Newey-West standard error is computed by taking the average of the cross-sectional Newey-West standard errors.

The null hypothesis is that of no difference in predictive accuracy between method (1) and (2) and we test at a significance level of five percent.

4.7 Variable Importance

The primary objective of this thesis is to evaluate the potential of machine learning methods for predicting asset risk premia based on their predictive performance. While interpretation is not a primary focus, we will identify the most influential predictor variables in the cross-section as a secondary objective to gain some more insights on the drivers of stock returns.

We do so by means of feature importance for the random forest. We measure feature importance by the Mean Decrease in Impurity (MDI). The MDI calculates the average reduction in impurity across all trees in the ensemble when a specific feature is considered for splitting Scornet (2020). In general, features with a higher MDI are deemed more important. Specifically, we compute feature importances each time we fit the model on the training and validation set combined. Afterwards, we derive the average feature importance to get a better understanding of which predictors contribute most to our random forest model.

5 Empirical Results

5.1 Performance Comparison

Table 1 shows the predictive performance of all methods in terms of percentage R^2_{OOS}. We compare a total of four methods, including two linear models and two machine learning models. The linear models we use are the Fama & French three factor model (FF3+H) and a simple linear model (OLS+H) using the same high-dimensional predictor variable set as the machine learning methods. Both linear models are estimated using the Huber loss function to provide more robust out-of-sample predictions. The machine learning methods we forecast with include a feedforward neural network (FFNN) containing three hidden layers and random forest (RF).

The first row of table 1 presents R^2_{OOS} for the entire pooled 15 years that make up the testing sample. In the second and third rows the same R^2_{OOS} is presented for a sub-sample only

containing the 500 top and bottom stocks, respectively, in terms of size. Here size is defined as market capitalization, represented by the stock-level characteristic *mvel*1.

Table 1: Monthly Out-of-Sample Predictive Performance (Percentage R^2_{OOS}) per method for the entire pooled sample (All), high - and low market capitalization companies (Top & Bottom).

	OLS +H	FF3 +H	RF	FFNN
All	1.23	-4.24	-1.14	3.35
Top 500	1.23	-4.25	-1.14	3.34
Bottom 500	-0.41	-1.87	-1.15	1.36

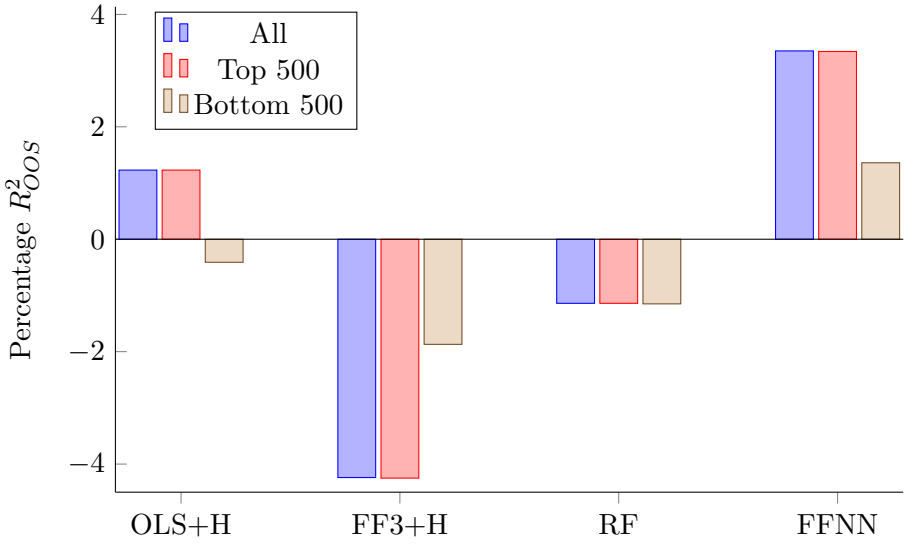


Figure 3: Percentage R^2_{OOS} per method based on the prediction of monthly returns.

Overall we find that the FFNN performs best, producing an out-of-sample R^2 of 3.35% for the entire pooled sample. In second place comes OLS+H, yielding an R^2_{OOS} of 1.23% and therefore beating expectations by outperforming the random forest method. RF underperforms relative to a naive forecast of zero with an out-of-sample R^2 of -1.14%. The Fama & French three factor model adapted to our setting with the Huber loss, is not able to compete with the other models, generating an R^2_{OOS} of -4.24%.

The FFNN is the best performing method overall, which could be partially attributed to the ability of the neural network to incorporate complex interactions between predictors, which are missed by our linear models. The FFNN is not the only method with this ability since this is embedded in tree-based models like RF as well. However, in contrast to FFNN, the random forest model does not produce robust results.

The linear Huber model (OLS+H) containing all 197 predictor variables performs surprisingly well. OLS+H produces a positive R^2_{OOS}, indicating it dominates a naive forecast of zero. Applying the Huber loss function appears to be the crucial factor that causes this performance, since using the standard least squares objective produces an R^2_{OOS} far into negative territory.

As previously discussed, RF fails to generalize for our data, producing a negative out-of-sample R^2. We find that RF performs well for the first two years of testing, producing R^2_{OOS} values of around 8%. After these two years we enter a turbulent period in the testing data which the RF model is unable to appropriately adapt to, producing an R^2_{OOS} of approximately -10% in the third testing year. This turbulent data enters the model as part of the training and validation set in the following years, which could help improve the model in the following years in case of a structural break corresponding to the turbulence. However, one year of data is still only a small proportion of the entire training - and validation set. The inability of random forest to generalize when faced with changing data structures is potentially the bottleneck of the model in this study.

The poor performance of the Fama & French model can be caused by several factors. Firstly, the FF3+H model only considers three monthly factors which do not vary per stock. This means the model is very static compared to the other three methods we consider, since predictions for a given month are the same for all stocks.

Secondly, the Fama & French model is designed to explain stock returns using on common risk factors, which is not what we focus on in this study. We forecast excess returns using the three factors which are lagged by one month, since the actual values of the factors are not available at time t to predict the excess return at time t. When we do not lag the factors, the FF3+H model produces a respectable R^2 of around 8%, which supports the hypothesis that these three factors are able to explain a substantial part of returns. However, our results show that forecasting returns based on the lagged factors is not economically informative.

In addition to analyzing the entire pooled sample, we compute the out-of-sample R^2 for subsamples only containing the 500 top and the 500 bottom stocks in terms of market capitalization. Comparing these per method, we find that the top 500 stocks present nearly identical R^2_{OOS} values as the entire sample. The bottom 500 stocks deviate more from the entire sample, producing R^2_{OOS} values that are generally closer to zero. The contrast between large and small cap stocks could be due to higher market efficiency of large market cap stocks. Higher market efficiency implies that prices reflect all available information quicker which could help a model - when specified correctly - better predict excess returns.

Table 2: Annual Out-of-Sample Prediction Performance (Percentage R^2_{OOS})

	OLS +H	FF3 +H	RF	FFNN
All	1.20	-15.26	-1.31	3.66
Top 500	1.18	-15.23	-1.34	3.64
Bottom 500	1.28	-1.74	1.14	4.03

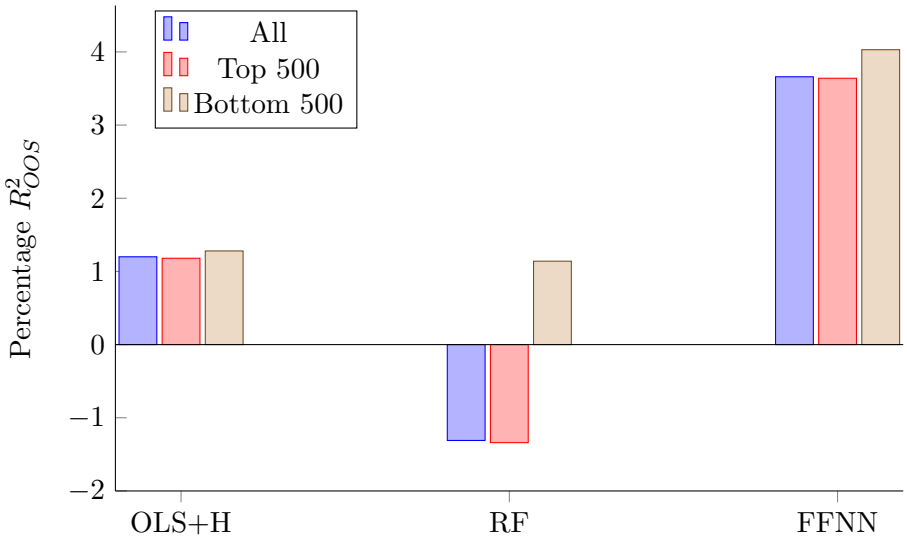


Figure 4: Percentage R^2_{OOS} per method for annual returns. FF3+H is left out due to its large negative values as observed in table 2.

Table 2 shows the results for each of the methods at the annual horizon. The relative performance of the models does not change compared to the monthly setting. FFNN still performs best overall, followed by OLS+H, RF and FF3+H.

Although relative model performance remains the same, there are some noticeable differences when considering annual returns. Firstly, the FF3+H model worsens substantially, dropping down to an out-of-sample R^2 of -15.26%. As discussed, the model is not specifically designed to predict future returns but rather aims explain stock returns based on a select number of driving factors. Using the Fama & French model to predict annual returns fails, from which we can conclude that the model is not suited for individual stock return prediction.

The predictive power of FFNN slightly increases for annual returns, with an R^2_{OOS} of 3.66% for the overall sample, which reaffirms our findings from the monthly setting that the FFNN is able to capture a respectable part of excess returns in the model.

In contrast to a previous study by Gu et al. (2020), our annual R^2_{OOS} results are not an order of magnitude larger than our monthly results. Gu et al. (2020) observe relatively small R^2_{OOS} values for the monthly setting, peaking at 0.40% for a feedforward neural network with three hidden layers. In the annual setting, the results of Gu et al. (2020) improve drastically, yielding a similar magnitude of annual R^2_{OOS} values to our results. Equivalently to Gu et al. (2020), our models that produced negative R^2_{OOS} in the monthly setting (FF3+H and RF) do even worse for predicting annual returns. However, for RF this difference is only 0.17%.

At the annual horizon, all models show a substantial increase in performance when predicting for small (bottom) market capitalization stocks. In the monthly setting, the two models producing positive R^2_{OOS} values both perform substantially worse for the bottom 500 stocks. When predicting for a holding period of one year, all four models produce the best results for small cap stocks. This supports our earlier hypothesis that it takes longer for small cap stocks to reflect all publicly available information in the stock price, which is in line with the literature on financial market efficiency (Hung et al., 2009). Similarly to the monthly setting, performance

of the top 500 stocks is nearly identical to the entire pooled sample R^2_{OOS} when predicting for annual risk premia.

Overall, our annual results show that FFNN and OLS+H are not only able to capture short-term changes in the risk premium, but also successfully isolate excess returns that persist over time. FFNN is the best performing method, producing R^2_{OOS} values of 3.35% and 3.66% for the monthly and annual setting, respectively.

Table 3: Pairwise Model Comparison on Monthly Out-of-Sample Prediction Performance using Diebold-Mariano Tests. Positive numbers indicate the column model produces better forecasts than the row model. None of the differences in performance are significant at the 5% level.

	FF3 +H	RF	FFNN
OLS +H	-0.35	-0.28	0.15
FF3 +H		0.26	0.41
RF			0.28

In addition to evaluating model performance based on the R^2_{OOS}, we perform pairwise model comparisons using Diebold-Mariano tests to assess whether a difference in predictive performance is significant. Table 3 shows our results for each for each pair of models predicting for the monthly horizon.

At the monthly level, our results show that none of the differences in forecasting errors are significant for any of the model pairs at the 5% level.

A potential reason for the insignificant results is high variance of forecast errors. Due to the low signal-to-noise ratio of returns, we observe relatively high variability of forecast errors for all of the models.

Another potential cause for insignificance of a DM test result is a small sample size, since this decreases the power of the test. We adapt our DM test to compare the cross-sectional average of prediction errors instead of comparing errors for all individual returns. Only considering the cross-sectional average means we decrease the sample size for the test to 180 observations, or 15 years worth of monthly average prediction errors. However, a sample size of 180 should still be large enough to have sufficient power, so this is unlikely to cause insignificance.

When applying the unadapted DM test, all of the differences in predictive power are significant, however, these p-values are not reliable due to the expected violated assumption of weak error dependence.

5.2 Feature Importance

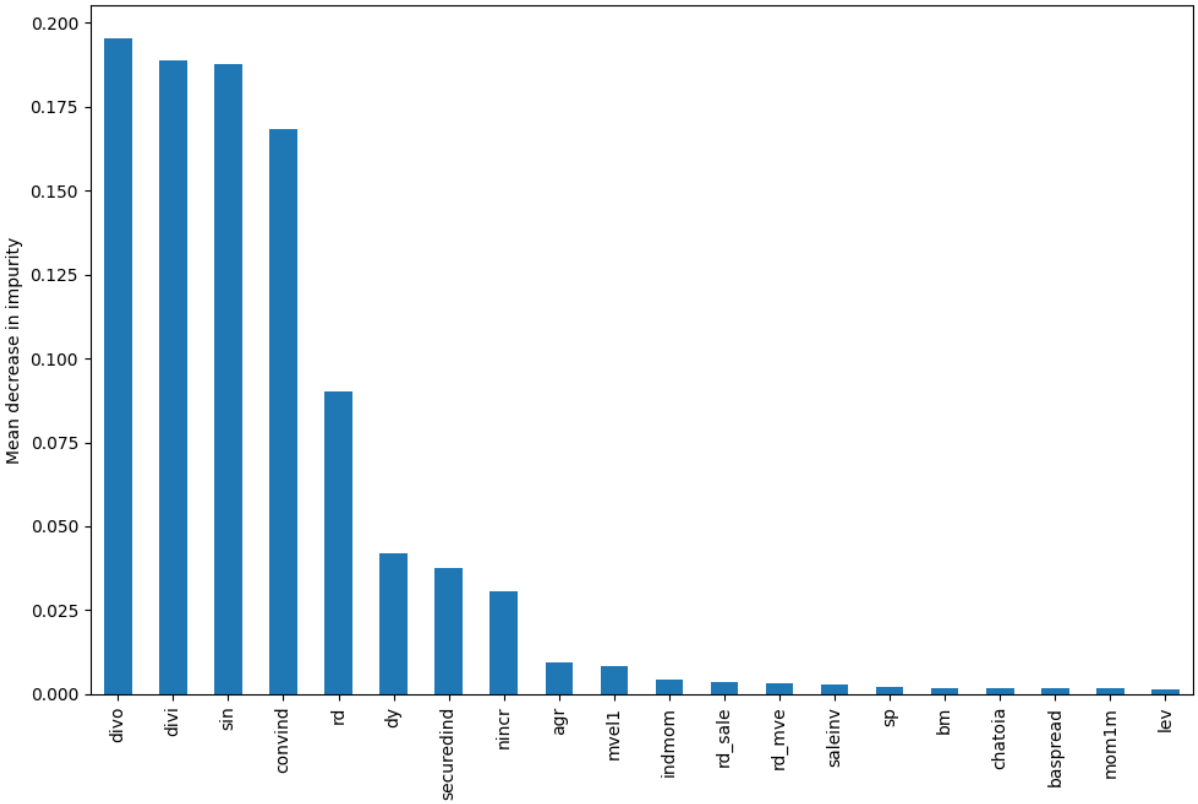
To gain insights on the relative importance of predictors, we compute feature importances from our random forest model based on the monthly setting. We obtain feature importances using the mean decrease in impurity for two different RF models. The first model only uses stock-level characteristics as predictors to construct monthly return forecasts, whereas the second model uses the entire predictor set which additionally contains macroeconomic predictors and SIC

industry dummies.

We compare these two settings because of the difference in data preparation for these different variables. In the data section, we explain the procedure used to cross-sectionally rank the stock-level characteristics and map these to the [-1,1] interval. For the SIC industry variable, we construct dummies indicating whether a stock corresponds to a certain industry. On the contrary, the macroeconomic predictors are not scaled. Not scaling some of the predictors could bias the feature importance calculations to assign higher relative importance to large-scale variables. Additionally, unscaled predictors may influence the tree-growing process of the random forest as splits could favor variables with larger scales, which could potentially impact model performance.

Feature importances are not global but specific to the model and dataset. Since our RF models fail to capture a positive R^2_{OOS}, the features we find to be most important can only be interpretated as most important for these specific models with poor out-of-sample performance.

Figure 5: Random Forest Feature Importances for the top 20 most influential variables in the model when exclusively using the 92 stock-level characteristics as predictors.



Figures 5 and 6 contain feature importances for the RF model containing only stock-level predictors and the model containing the entire predictor set, respectively. Here we only include the top 20 most influential variables. Importances for all variables of the model containing only stock-level characteristics can be found in appendix figure 8.

We find that including the unscaled macroeconomic predictors and SIC dummies produces

a slightly higher R^2_{OOS}, indicating that performance is not negatively affected by incorporating the unscaled variables.

Our results in figure 5 show that the most influential variables for our RF model limited to stock-level predictors are related to dividends. These variables include dividend omission (divo), dividend initiation (divi) and dividend to price ratio (dy). Next, we find that sin stocks (sin) and convertible debt indicator (convind) contribute most. Multiple R&D related variables also enter the top 20, including R&D increase (rd), R&D to sales ratio (rd_sale) and R&D to market capitalization (rd_mve). Our findings deviate from the results of Gu et al. (2020), who find divo, divi and sin to be among the least influential variables for the RF model. They find that variables related to momentum and size (mvel1) contribute most to the model. These variables also appear in our top 20, but contribute substantially less. Overall, only 9 of our top 20 variables also appear in the top 20 of Gu et al. (2020) for their random forest.

Figure 6: Feature Importances for the top 20 most influential variables when including the 92 stock-level characteristics, 8 macroeconomic predictors and 97 SIC dummies as predictors.

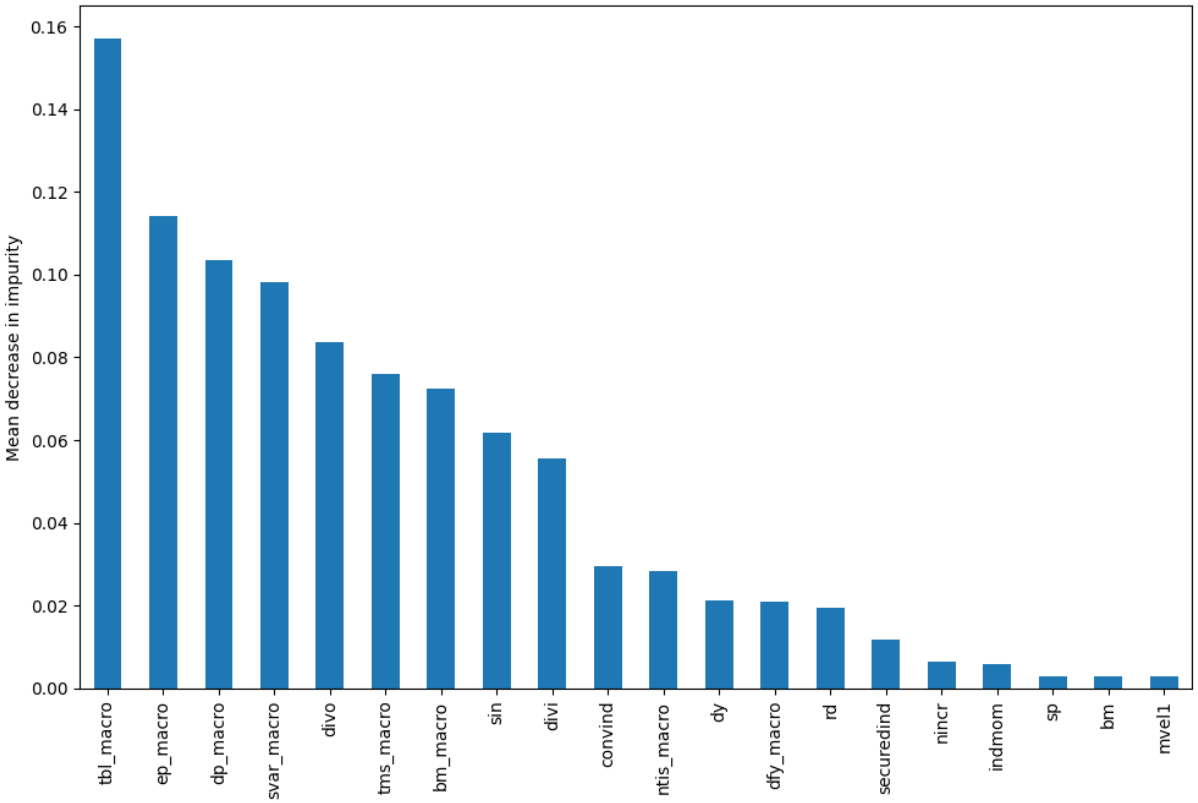


Figure 6 shows the results when we include the macroeconomic predictors and SIC dummies in the model. It is likely that not scaling the variables biases the feature importance calculations towards assigning higher relative importance to these unscaled predictors, since all eight macroeconomic variables appear in the top 20. In fact, out of the seven variables with the highest assigned importance, six of them are macroeconomic predictors.

Since these macroeconomic predictors seem to bias the feature importance results in figure

6, it is unclear which of the variables in the complete RF model actually contribute most to model performance.

Overall, we conclude based on figure 5 including only the stock-level characteristics, that variables related to dividend and R&D are most influential for our RF model. Additionally, the convertible debt indicator and sin stock variables rank among the most important. This contradicts expectations based on Gu et al. (2020), who find momentum predictors, liquidity variables and risk measures to be most important based on the reduction in R^2 when setting a variable to zero.

6 Conclusion

This thesis focuses on replicating and extending the work of Gu et al. (2020), who conduct a large-scale analysis on the effectiveness of machine learning methods for the prediction of equity risk premia. We forecast monthly and annual excess returns each month using a total of four methods. Three of our methods are also used in Gu et al. (2020), including random forest, simple linear with a Huber loss function and feedforward neural network. We extend this set of methods by further incorporating the Fama & French three factor model.

First, we examine how machine learning methods compare to traditional methods for out-of-sample prediction of asset risk premia. Our feedforward neural network outperforms all other methods based on the R^2_{OOS} in both the monthly and annual setting, which is in line with the results of Gu et al. (2020). In contrast to expectations based on Gu et al. (2020), random forest fails to produce a positive R^2_{OOS} and is outperformed by the linear model with all predictors and Huber loss. The Fama & French model is not able to compete with the other methods and seems unsuitable for the prediction of excess returns when its factors are lagged. Based on adjusted Diebold-Mariano tests, where we perform a pairwise comparison of the cross-sectional average forecast errors, we conclude that none of the differences in predictive performance are significant at the 5% level.

We show that risk premia of stocks with a small market capitalization are harder to predict in the monthly setting compared to big cap stocks. When performing our annual analysis, predictive performance for small cap stocks improves considerably, supporting the theory that small cap stocks take more time to reflect all information in their stock prices.

Next, we identify the driving factors of risk premia by evaluating their relative performance using feature importance on our random forest models. The results show that the most important stock-level predictors for our random forest are variables associated with dividends and R&D. Other influential variables are sin stocks, convertible debt indicator, size and momentum related predictors.

Overall, random forest failed to improve upon the out-of-sample performance of traditional methods in our setting. On the other hand, neural networks show promising results for empirical asset pricing applications based on out-of-sample R^2 values of 3.35% and 3.66% for monthly and annual equity risk premia, respectively.

Finally, harnessing machine learning methods like neural networks shows promise to improve prediction of excess returns. Since risk premia are a central component in empirical asset pricing, machine learning could help increase our understanding of the behaviour of asset prices.

7 Discussion

This section discusses several points of improvement, suggestions for future research and any remaining considerations.

Firstly, we observe poor performance of our random forest models in both the monthly and annual setting in terms of overall R^2_{OOS}. We separately evaluate the performance for every year in the testing sample and find that RF R^2_{OOS} is generally stable and positive. In years three and four of the testing set, corresponding to 2008 and 2009, RF produces large negative R^2_{OOS} values. This is around the time of the great recession, taking place from late 2007 to 2009, which explains the change in the data structure. The poor performance of the random forest model implies that RF is potentially less capable of handling turbulent periods in the data relative to neural networks.

Compared to Gu et al. (2020), we obtain fairly high R^2_{OOS} results for the monthly setting. It would be interesting to explore whether this has to do with modelling decisions or the difference in data. Furthermore, our OLS+H model exceeds expectations, producing positive R^2_{OOS} values in both the monthly and annual setting. Since this contradicts the results of Gu et al. (2020), we suggest to expand the dataset to examine whether our result holds in the face of different data.

Moreover, we discussed that including the unscaled macroeconomic predictors did not negatively impact RF performance. However, a suggestion for future research would be to rank and scale the macroeconomic variables over the entire sample and examining whether this further improves performance.

Another possibility is to examine the differences in performance when using a rolling window compared to an expanding window for the training set. It is likely that relationships between predictors and the target variable are not static but change over time, which could result in improved performance when applying a rolling window.

Our results show support for the theory that small cap stocks take more time to reflect all information in the stock price. Hence, it would be interesting to explore whether this holds for a different dataset or when using different predictive models.

Lastly, hyperparameter tuning is of crucial importance for the performance of machine learning models. Due to time restrictions we focus on tuning a select number of hyperparameters per method. Expanding the set of hyperparameters we tune could result in improved out-of-sample performance and more insights on the implications of deep learning compared to shallow learning for risk premia prediction.

References

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

Bryzgalova, S., Lerner, S., Lettau, M. & Pelger, M. (2022). Missing financial data. *Available at SSRN 4106794*.

Capocci, D. & Hübner, G. (2004). Analysis of hedge fund performance. *Journal of Empirical Finance*, *11*(1), 55–89.

Chiah, M., Chai, D., Zhong, A. & Li, S. (2016). A better model? an empirical investigation of the fama–french five-factor model in australia. *International Review of Finance*, *16*(4), 595–638.

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

Eraslan, V. (2013). Fama and french three-factor model: Evidence from istanbul stock exchange. *Business and Economics Research Journal*, *4*(2), 11.

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

Fama, E. F. & French, K. R. (1996). The capm is wanted, dead or alive. *The Journal of Finance*, *51*(5), 1947–1958.

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

Feurer, M. & Hutter, F. (2019). Hyperparameter optimization. *Automated machine learning: Methods, systems, challenges*, 3–33.

Freyberger, J., Neuhierl, A. & Weber, M. (2020). Dissecting characteristics nonparametrically. *The Review of Financial Studies*, *33*(5), 2326–2377.

Fujiyoshi, H., Hirakawa, T. & Yamashita, T. (2019). Deep learning-based image recognition for autonomous driving. *IATSS research*, *43*(4), 244–252.

Goldstein, B. A., Navar, A. M. & Carter, R. E. (2017). Moving beyond regression techniques in cardiovascular risk prediction: applying machine learning to address analytic challenges. *European heart journal*, *38*(23), 1805–1814.

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

Huber, P. J. (1992). Robust estimation of a location parameter. *Breakthroughs in statistics: Methodology and distribution*, 492–518.

Hung, J.-C., Lee, Y.-H. & Pai, T.-Y. (2009). Examining market efficiency for large-and small-capitalization of topix and ftse stock indices. *Applied Financial Economics*, *19*(9), 735–744.

Jain, A. K., Mao, J. & Mohiuddin, K. M. (1996). Artificial neural networks: A tutorial. *Computer*, *29*(3), 31–44.

Kourou, K., Exarchos, T. P., Exarchos, K. P., Karamouzis, M. V. & Fotiadis, D. I. (2015). Machine learning applications in cancer prognosis and prediction. *Computational and structural biotechnology journal*, *13*, 8–17.

Krollner, B., Vanstone, B. J., Finnie, G. R. et al. (2010). Financial time series forecasting with machine learning techniques: a survey. In *Esann.*

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

Masters, T. (1993). *Practical neural network recipes in c++*. Academic Press Professional, Inc.

Officer, R. R. (1972). The distribution of stock returns. *Journal of the american statistical association*, *67*(340), 807–812.

Phan, D. H. B., Sharma, S. S. & Narayan, P. K. (2015). Stock return forecasting: Some new evidence. *International Review of Financial Analysis*, *40*, 38–51.

Rapach, D. & Zhou, G. (2013). Forecasting stock returns. In *Handbook of economic forecasting* (Vol. 2, pp. 328–383). Elsevier.

Rapach, D. E., Strauss, J. K. & Zhou, G. (2013). International stock return predictability: what is the role of the united states? *The Journal of Finance*, *68*(4), 1633–1662.

Scornet, E. (2020). Trees, forests, and impurity-based variable importance. *arXiv preprint arXiv:2001.04295*.

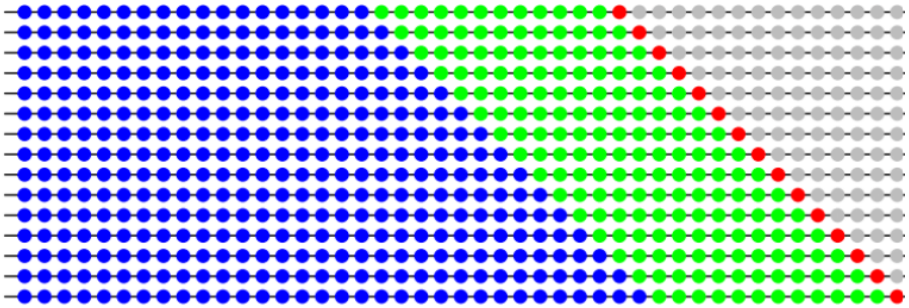
Svozil, D., Kvasnicka, V. & Pospichal, J. (1997). Introduction to multi-layer feed-forward neural networks. *Chemometrics and intelligent laboratory systems*, *39*(1), 43–62.

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.

A Hyperparameter tuning

A.1 Sample Splitting

Figure 7: Visual representation of the training, validation and testing split, where each horizontal line represents one iteration. Blue dots correspond to the training set which expands each year. Green dots correspond to the rolling validation set and red dots represent the moving test set. Each dot represents a year of data. This figure is obtained from the online appendix of Gu et al. (2020), who use a similar sample split scheme.



A.2 Tuning Scheme

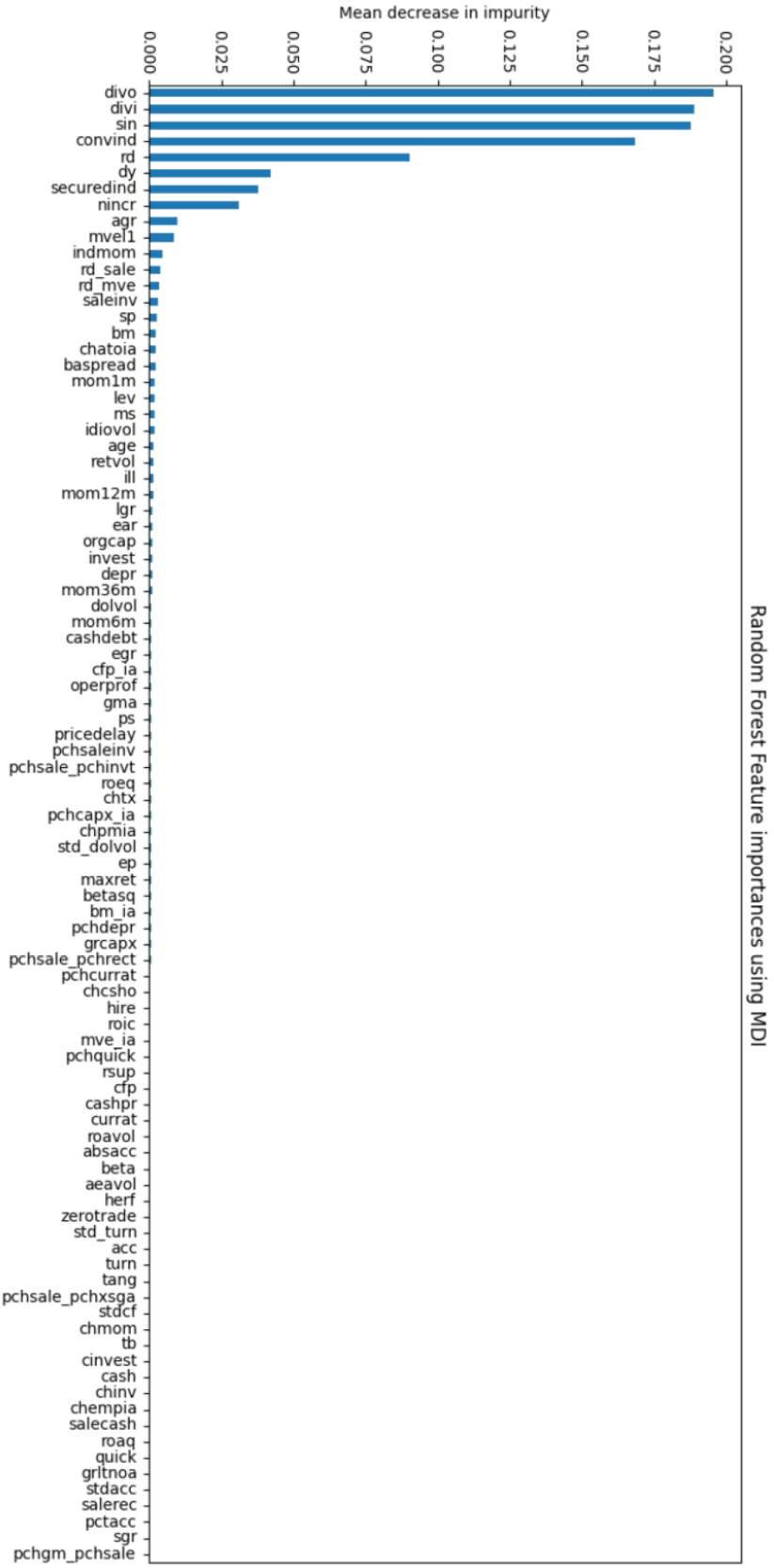
Table 4: Hyperparameters that are set or tuned per method. This table does not contain all hyperparameters. Hyperparameters that are set identically to Gu et al. (2020) are left out. Epsilon refers to the only hyperparameter of the Huber loss function that controls the amount of regularization. LR refers to the learning rate of the FFNN. #Features refers to the number of features considered per split for the random forest.

OLS +H	FF3 +H	RF	FFNN

epsilon
$\in \{2.0, 2.5, 3.0\}$
	epsilon
	$\in \{2.0, 2.5, 3.0\}$
		#Trees = 100
		Depth = $\in \{1, 2, 3, 4\}$
		#Features = $\in \{8, 10, 20, 30\}$
			# hidden layers = 3
			ensemble = 2
			LR $\in \{0.001, 0.01\}$
			L1 penalty $\in \{10^{-5}, 10^{-3}\}$

B Feature Importances

Figure 8: Random Forest Feature Importances when only including the 92 stock-level characteristics as predictors.



C Predictor Variables

Table 5: Details of the stock-level characteristics

No.	Acronym	Description	Frequency
1	mvel1	Size	Monthly
2	beta	Beta	Monthly
3	betasq	Beta squared	Monthly
4	chmom	Change in 6-month momentum	Monthly
5	dolvol	Dollar trading volume	Monthly
6	idiovol	Idiosyncratic return volatility	Monthly
7	indmom	Industry momentum	Monthly
8	mom1m	1-month momentum	Monthly
9	mom6m	6-month momentum	Monthly
10	mom12m	12-month momentum	Monthly
11	mom36m	36-month momentum	Monthly
12	pricedelay	Price delay	Monthly
13	turn	Share turnover	Monthly
14	absacc	Absolute accruals	Annual
15	acc	Working capital accruals	Annual
16	age	# years since first Compustat coverage	Annual
17	agr	Asset growth	Annual
18	bm	Book-to-market	Annual
19	bm_ia	Industry-adjusted book-to-market	Annual
20	cashdebt	Cash flow to debt	Annual
21	cashpr	Cash productivity	Annual
22	cfp	Cash flow to price ratio	Annual
23	cfp_ia	Industry-adjusted cash flow to price ratio	Annual
24	chatoia	Industry-adjusted change in asset turnover	Annual
25	chcsho	Change in shares outstanding	Annual
26	chempia	Industry-adjusted change in employees	Annual
27	chinv	Change in inventory	Annual
28	chpmia	Industry-adjusted change in profit margin	Annual
29	convind	Convertible debt indicator	Annual
30	currat	Current ratio	Annual
31	depr	Depreciation / PP&E	Annual
32	divi	Dividend initiation	Annual
33	divo	Dividend omission	Annual
34	dy	Dividend to price	Annual
35	egr	Growth in common shareholder equity	Annual
36	ep	Earnings to price	Annual
37	gma	Gross profitability	Annual
38	grcapx	Growth in capital expenditures	Annual
39	grltnoa	Growth in long term net operating assets	Annual
40	herf	Industry sales concentration	Annual
41	hire	Employee growth rate	Annual
42	invest	Capital expenditures and inventory	Annual
43	lev	Leverage	Annual
44	lgr	Growth in long-term debt	Annual
45	mve_ia	Industry-adjusted size	Annual
46	operprof	Operating profitability	Annual

Table 6: Details of the stock-level characteristics (continued)

No.	Acronym	Description	Frequency
47	orgcap	Organizational capital	Annual
48	pchcapx_ia	Industry adjusted % change in capital expenditures	Annual
49	pchcurrat	% change in current ratio	Annual
50	pchdepr	% change in depreciation	Annual
51	pchgm_pchsale	% change in gross margin - % change in sales	Annual
52	pchquick	% change in quick ratio	Annual
53	pchsale_pchinvt	% change in sales - % change in inventory	Annual
54	pchsale_pchrect	% change in sales - % change in A/R	Annual
55	pchsale_pchxsga	% change in sales - % change in SG&A	Annual
56	pchsaleinv	% change sales-to-inventory	Annual
57	pctacc	Percent accruals	Annual
58	ps	Financial statements score	Annual
59	quick	Quick ratio	Annual
60	rd	R&D increase	Annual
61	rd_mve	R&D to market capitalization	Annual
62	rd_sale	R&D to sales	Annual
63	roic	Return on invested capital	Annual
64	salecash	Sales to cash	Annual
65	saleinv	Sales to inventory	Annual
66	salerec	Sales to receivables	Annual
67	securedind	Secured debt indicator	Annual
68	sgr	Sales growth	Annual
69	sin	Sin stocks	Annual
70	sp	Sales to price	Annual
71	tang	Debt capacity/firm tangibility	Annual
72	tb	Tax income to book income	Annual
73	aeavol	Abnormal earnings announcement volume	Quarterly
74	cash	Cash holdings	Quarterly
75	chtx	Change in tax expense	Quarterly
76	cinvest	Corporate investment	Quarterly
77	ear	Earnings announcement return	Quarterly
78	nincr	Number of earnings increases	Quarterly
79	roaq	Return on assets	Quarterly
80	roavol	Earnings volatility	Quarterly
81	roeq	Return on equity	Quarterly
82	rsup	Revenue surprise	Annual
83	stdacc	Accrual volatility	Quarterly
84	stdcf	Cash flow volatility	Quarterly
85	ms	Financial statement score	Quarterly
86	baspread	Bid-ask spread	Monthly
87	ill	Illiquidity	Monthly
88	maxret	Maximum daily return	Monthly
89	retvol	Return volatility	Monthly
90	std_dolvol	Volatility of liquidity (dollar trading volume)	Monthly
91	std_turn	Volatility of liquidity (share turnover)	Monthly
92	zerotrade	Zero trading days	Monthly

D Code Description

All methods have been implemented using python in Jupyter Notebook. Before running any code, we construct one excel file where all provided CRSP data is included. Additionally, we compute excess returns in a separate column and obtain the corresponding macroeconomic predictors. We use a separate data file for the Fama & French three factor model, only containing the dates, companies, excess returns, FF3 factors and mvel1 (size) to be able to sort the stocks for the top 500 and bottom 500 calculations. Lastly, construct annual returns in python based on the original df, and write these results to an excel file called 'annual'.

We use Keras to implement our feedforward neural network, randomforestregressor for our random forest, and huberregressor for our simple linear models.

For data preparation, we cross-sectionally rank the stock-level characteristics and map to [-1,1]. Moreover, we construct SIC dummies.

After doing so, we run the methods separately for the monthly and annual setting, updating the training/validation/test sets each iteration. We obtain the out-of-sample R^2 and DM test results from functions we coded based on our adjused settings.

We provide all code for the replication of the monthly and annual results, including DM test, feature importances and R^2 function.