universität
wien

MASTERARBEIT / MASTER’S THESIS

Titel der Masterarbeit / Title of the Master‘s Thesis

„A Comparative Analysis of Equity Premium Prediction
via Machine Learning“

verfasst von / submitted by

David Wögerer, BSc (WU)

angestrebter akademischer Grad / in partial fulfilment of the requirements for the degree of

Master of Science (MSc)

Wien, 2022 / Vienna 2022

Studienkennzahl lt. Studienblatt /
degree programme code as it appears on
the student record sheet: UA 066 974

Studienrichtung lt. Studienblatt /
degree programme as it appears on
the student record sheet: Masterstudium Banking and Finance

Betreut von / Supervisor: ao. Univ.-Prof. Dipl.-Ing. Dr. Erhard Reschenhofer

Mitbetreut von / Co-Supervisor:

Abstract

In this thesis, we extensively reinvestigate the performance of variables that are considered good predictors of the equity premium in the academic literature. The methods used are extended by so-called machine learning models. For the most part, neither the conventional models nor the extended models can improve the predictions. They give poor out-of-sample results and do not appear to be stable either. Attempting to use these predictions as signals for market timing also largely results in poor outcomes.

Keywords: Machine Learning, Equity Premium, Equity Risk Premium Prediction, Forecasts, Out-of-Sample Prediction

Abstract (German)

In dieser Arbeit wird die Leistung von Variablen, die in der akademischen Literatur als gute Prädiktoren für die Aktienprämie gelten, eingehend untersucht. Die verwendeten Methoden werden durch sogenannte maschinelle Lernmodelle erweitert. In den meisten Fällen können weder die herkömmlichen Modelle noch die erweiterten Modelle die Vorhersagen verbessern. Sie liefern schlechte Out-of-Sample-Ergebnisse und scheinen auch nicht stabil zu sein. Der Versuch, diese Vorhersagen als Signale für das Market-Timing zu verwenden, führt ebenfalls größtenteils zu schlechten Ergebnissen.

Schlagwörter: Maschinelles Lernen, Marktprämie, Vorhersage der Marktrisikoprämie, Prognosen, Vorhersagen außerhalb der Stichprobe

Contents

List of Figures

List of Tables

1 Introduction

Stock market forecasting is perhaps as old as the stock market itself. So why not apply *advanced technology* to this old topic - in this case: machine learning. Equity risk premiums are not only one of the most discussed topics in finance, but also fundamental to many areas such as asset pricing and portfolio theory. There have been numerous attempts to predict market movements, and although some linear models can explain much of the variance in equity risk premia in the sample, many *robust* econometric models fail to predict movements at the simple historical average benchmark, as Welch & Goyal (2008) have shown in their paper. In particular, predicting the out-of-sample equity risk premium remains the biggest problem. Cochrane (2008), on the other hand, presented arguments for why predictability of returns must be possible if there is no predictability of dividends. Some of the more recent papers like Gu et al. (2020) claim to have found a way to predict equity risk premia more accurately using nonlinear models. However these machine learning algorithms depend on parameters that are in turn susceptible to data snooping, since, as Lo & MacKinlay (1990) already mentioned, given enough time and trials, almost any pattern can be found in the available data. Pesaran & Timmermann (1995) also point out that scientists have been relying on snooped data because they all necessarily use the same data set when predicting the U.S. equity (risk) premium.

The objective of this paper is to investigate whether new technologies can outperform this historical average benchmark and how different models compare. To compare these models in their predictive power, performance is measured by statistical accuracy. This accuracy is defined as the difference between the mean square error of the model and the benchmark. If the value is greater than zero, the model has a higher predictive accuracy compared to the historical average. The variables and the methodology of the annual predictions are based on the work of Welch & Goyal (2008). The model of this aforementioned paper will then be extended with new data (until 2020) and it will be examined whether the results have changed in the last 15 years and whether there are any new findings. The next step was to test the models for their monthly predictive power. Here, linear and non-linear models (i.e. machine-learning applications) were used to compare them with each other and to investigate whether they can outperform the linear models from the paper by Welch & Goyal (2008).

From an investor's perspective, it would be interesting to gain an advantage with these predictions by using them for market timing. One can divide the predictions into two categories, long for positive predictions and short for negative predictions, and invest one's portfolio accordingly. Therefore, long-short portfolios are created and compared to apply these academic findings to a practical environment.

The hypothesis of this paper is therefore: can linear or nonlinear models improve the predictive accuracy compared to the historical average. And how well do these models perform for different types of parameters and methods. The difficulty in this and similar papers is not to be too susceptible to snooping biases and at the same time to achieve robust results.

The paper proceeds as follows. Section 2 gives a brief overview of the current state of the academic literature. In Section 3 and 4, the data model, which is used for this analysis, and the methodology of this work is described. Section 5 describes the main results of this work and Section 6 concludes the findings of this paper.

2 Literature Review

The beginnings of market forecasting in the literature are very sobering. As early as 1900, the French mathematician Louis Bachelier began mathematically modeling stock price movements in his dissertation "Théorie de la spéculation", seeking a probabilistic approach to stock price movements. In his dissertation, he came to the rather disappointing conclusion that the calculation of probabilities can never be applied to stock market developments and that stock market dynamics will never be an exact science. Only in a static state the stock market can be described mathematically. His findings later formed the basis for the Black-Scholes option pricing model.

And already Dow (1920) has asked the question, if there is such thing as "scientific speculation". He also mentions the problem of market timing - being able to "buy low and sell dear". But to know, when the prices are low and when they are "dear" is the real problem. At that time, the rule of some traders in the commission offices was, to go against whatever the outside opinion was. However, this could also prove to be a disadvantage if prices have a long-lasting upswing and they may never fall back below the former prices. As the economist John Maynard Keynes said in the 1930s: “Markets can stay irrational longer than you can stay solvent.” Dow then goes on in his article to explain some methods that can be used to predict the performance of stocks. Most of them come from the field of technical analysis - he also gives enough arguments on why these methods do not work - and he further explains why dividends provide information about the value of a stock. Dividends, so to speak, already have a long history in the field of forecasting.

As an honorable mention, I would like to briefly introduce the work of Harry Markowitz from 1952, in which he introduced mathematical concepts to the world of finance with his seminal dissertation "Portfolio Selection". He significantly coined the term "diversification" by using the mean return and covariances of stocks to create a portfolio with the most favorable risk-return profile. This allowed him to quantify the concept of diversification in a market. This approach paved the way for further theories in the field of financial quantitative analysis, such as stock or index return forecasting.

Dividends, or more specifically dividend yields, are likely to be one of the first and most studied predictor variables used to predict equity premiums. For example Campbell (1987) and Fama & French (1988) discussed this variable and their predictive power. Over the next two decades, many more predictor variables have been discussed and described in the scientific literature. Some of these variables include, in addition to dividends, stock variance, book values, corporate issuing activity, different interest rates and spreads, inflation, the capital investment ratio, the consumption, wealth and income ratios, and net or equity issuance activity. These and more variables are explained in more detail in the third section. In 2001, Lettau and Ludvigson concluded that it is now recognized in the academia that excess returns are predictable by various financial indicators, among others also dividends.

In 2008, Ivo Welch and Amit Goyal re-examined all the predictor variables that had been scientifically discussed up to that point to determine whether the prevailing consensus on the predictability of the equity risk premium holds. They investigated whether the forecasts are significant over a very long period of time and which years influence the forecasts and thus tried to draw a conclusion. They conclude that the predictive power of these variables was poor over the entire out-of-sample period. Some did well at the beginning of the period, but have declined over the past 30 years. In addition, the models were not reliable as they were also highly unstable. However, some models provide descriptive information when based on historical data and used in-sample. Therefore, no useful signals can be generated for an investor to use to gain an advantage.

In the same year, even in the same journal, Cochrane (2008) defended the hypothesis that stock returns must be predictable in his work "The Dog That Did Not Bark: A Defense of Return Predictability". He attempts to refute the view that stock returns are not simply a random walk and can therefore be predicted. In his paper he sets up the thesis, that the variation in dividend-price ratios must be forecastable, either trough the predicted dividend growth or the stock return. He further argues that if the work of Fama & French (1988) can be statistically rejected, the results of Shiller's (1981) volatility tests still raise unresolved questions. In this work, Shiller finds that stock price volatility measures are far too high to provide information about future dividends. Thus, the impression remains that only stock price movements must be predictable. In Cochrane's (2008) paper, he demonstrates that he could not predict future dividends from stock prices, confirming his assumption that stock returns should be predictable. But he did not seem entirely satisfied with this conclusion, arguing that predicting excess returns is a much more difficult task than predicting dividends using stock prices, since it would be much easier predicting dividend growth via stock prices.

Gu et al. (2020) attempt to extend the forecasts with nonlinear models and a new set of variables. These variables include 94 stock-level characteristics, which are based on Green et al. (2013), and 74 dummy variables consisting of Standard Industrial Classification (SIC) codes. On top of that, eight macroeconomic predictors from the variables from Welch & Goyal (2008) are included. These are the dividend-price ratio, earnings-price ratio, book-to-market ratio, net equity expansion, Treasury-bill rate, term spread, default spread, and stock variance. Thus, this set of variables is much more comprehensive than the data set used in this paper. They draw the conclusion that machine learning methods can improve the predictive power of prediction models and investors can generate higher Sharpe-ratios using the predictions for market timing, compared to a buy-and-hold strategy. The best methods according to their results are tress and neural networks, which are also included in this work. However, all methods in this paper restrict the variables in the variable selection process to the variations of momentum, liquidity and volatility. Thus, the previously mentioned variables, which were defined by the academic literature, are only of minor significance in this work.

It is difficult to draw a clear conclusion from the currently prevailing literature. Some articles claim that prediction works, others claim the opposite. As Welch & Goyal (2008; p. 1456) mentioned ”...a healthy skepticism is appropriate when it comes to predicting the equity premium”, I wanted to maintain this skeptical view in my approach to the topic of machine learning and forecasting. Using the same in-sample and out-of-sample tests, combined with more recent data and different linear and non-linear models, the results should show an unbiased view of the added value of machine learning models in the field of equity premium prediction.

3 Data

For this analysis, data from the website of Ivo Welch[1] is gathered and imported via the pandas DataReader into the Python project. To be able to comprehensibly compare the findings, the same data, updated until 2020, is used for this paper. There are some minor differences between the new data and the data used in the 2008 paper, possibly due to updated values from the relevant websites. The methodology of how the variables are created from the data set was exactly adopted and afterwards compared. In the following subsections, the dependent and independent variables used in the models are explained in more detail. More information on where the data was collected can be found in the appendix.

3.1 Dependent Variables

The dependent variable is in our case always the equity risk premium $r_{erp,t}$ of the S&P 500 index,

$$r_{erp,t} = r_{m,t} - r_{rf,t} \tag{1}$$

where $r_{m,t}$ is the index return including dividends and $r_{rf,t}$ is the risk-free rate. For the period 1920 to 2005, the Treasury-bill rate is used as the risk-free rate. Prior to that, no rate was available. Therefore, Welch & Goyal (2008) estimated a risk-free via regression using the Commercial paper rates for New York City. The index return is calculated from the continuously compounded monthly and annual returns including dividends. The time window for the available index data ranges from 1926 to 2020 for monthly returns and

[1] Data available at https://www.ivo-welch.info/professional/goyal-welch/

from 1871 to 2020 for annual returns. The data stems from the Center for Research in Security Press (CRSP) and the official website from Robert Shiller.

3.2 Independent Variables

There is a wide range of variables for which there would be a rational explanation as to why they would have predictive power for market risk premia. Exploring which variables might be best suited for this task is not part of this paper, so I will solely rely on the ones that have already been studied by the scientific literature. Recent papers (e.g. Gu et al. 2020) showed that non-linear models produce good results with momentum, moving averages or related variables. However, these will not be part of the models in this paper as the focus is on economic and business metrics and their predictive powers. A brief summary[2] of the variables follows:

- Dividend Price Ratio (d/p): Moving sum of the trailing 12-month difference between the logarithm of dividends and the logarithm of stock prices.
- Dividend Yield (d/p): 12-month moving sum of the difference between the logarithm of the dividends and the logarithm of the *lagged* stock prices.
- Earnings Price Ratio (e/p): Logarithm of the 12-month moving sum of earnings on the S&P 500 index minus the logarithm of stock prices.
- Dividend Payout Ratio (d/e): Is the difference between the 12-month moving sum of the logarithm of dividends and the logarithm of earnings.
- Stock Variance (svar): The annual or monthly sum of squared daily returns of the S&P 500 index.
- Cross-Sectional Premium (csp): Measures the relative valuation of high and low beta stocks, where this variables were imported directly from the data set.
- Book-to-Market Ratio (b/m): The book-to-market value ratio for the Dow Jones Industrial Average.
- Net Equity Expansion (ntis): Ratio of a 12-month moving sum of net equity issues by NYSE-listed stocks to the total market capitalization of the NYSE stocks. The formula to calculate the net equity issues is: IPOs + SEOs + stock repurchases - dividends.
- Percent Equity Issuing (eqis): Similar to (ntis), this is the is the ratio of net equity issuing activity as a *fraction* of total issuing activity.
- Treasury Bills (tbl): For the period from 1920 to 1933, the yields on short-term U.S. securities (Three-Six Month Treasury Notes) and for the subsequent period the three-month Treasury-bill rate (secondary market) are used.
- Long Term Yield (lty): Yields on long-term U.S. Bonds collected from the NBER Macrohistory data base.
- Long Term Rate of Returns (ltr): Using the return on long-term government bonds.

[2] For a more detailed explanation of the variables, see Welch & Goyal (2008), pp. 1457-1461

- Term Spread (tms): Is calculated as the difference between the long-term yield on government bonds and the Treasury-bill.
- Default Yield Spread (dfy): BAA-rated corporate bond yields minus AAA-rated corporate bond yields.
- Default Return Spread (dfr): Is the difference between long-term government bond and long-term corporate bond returns.
- Inflation (infl): The inflation rate is generated via the Consumer Price Index from the Bureau of Labor Statistics. The exact category is the "All Urban Consumers".
- Investment to Capital Ratio (i/k): Is the aggregate investment to aggregate capital of the whole economy, as introduced by Cochrane (1991).
- "Kitchen Sink" (all): Here, all previously mentioned variables are combined. The following variable (cay) is not included because of limited data availability.
- Consumption, Wealth, Income Ratio (cay): Variable introduced by Lettau & Ludvigson (2001). Using a regression to estimate coefficients and calculate (cay).

4 Methodology

This section explains the methodology used in this thesis. The models are presented only superficially, as the technical part of this thesis is not the main focus. The results of these models are then evaluated in the following chapter.

Predicting the future equity risk premium or value of a stock (or stocks) can be done in three ways: Fundamental, Technical or Quantitative Analysis. Another possibility would be to categorize the operations into Quantitative and Qualitative Analysis. Although some methods can be categorized to more than one category. Fundamental analysis tries to determine the true value of the stock based on economic and business variables. The most famous representative of this method is probably Warren Buffet. A term coined by Buffet is the "Cigar-Butt" strategy, whereby companies are sought that are trading below their liquidation value. This probably won't be the best approach for a professional investor, but it does point to the focus of fundamental analysis. With the help of the balance sheet, the income statement or macroeconomic data, an attempt is made to derive a value for the company. But the dependence on reports prepared by the company is a shortcoming of this method. Technical analysis, on the other hand, uses only data derived from the stock market - one could thus say it is the exact opposite of Fundamental Analysis when it comes to data. The difference is not only the data, but also the investment horizon. While fundamental analysis usually focuses on the medium to long term, technical analysis is more of a short term strategy. Data such as historical stock prices, trading volumes, and chart patterns of stock price movements are used in an attempt to predict the stock price. These include, for example, reverse, flag and head-and-shoulders patterns. But also data like moving averages and momentum are used. Quantitative analysis could be described as a combination of the two aforementioned methods. It combines the economic and business variables with historical prices and tries to find patterns which could be exploited. For example, Welch & Goyal (2008) used simple linear regression to try to fit the data to historical prices and used the coefficients, to predict the future price movements.

The range of available models from which to choose is extensive. Similar to Gu et al. (2020), this paper introduces supervised learning methods like OLS, Decision Tree and Multi-Layer Perceptron, to try predict the equity risk premia of the S&P 500 index. This equity risk premium can also be described as an excess return of the index, also called the equity premium, over the risk-free rate.

In its simplest form, the excess return $r_{i,t+1}$ of an asset i can be described as follows:

$$r_{i,t+1} = E_t(r_{i,t+1}) + \epsilon_{i,t+1} \tag{2}$$

where $\epsilon_{i,t+1}$ is the error term of the excess return and $E_t(r_{i,t+1})$ is described as:

$$E_t(r_{i,t+1}) = g^{\star}(z_{i,t}) \tag{3}$$

The algorithms used for these models are represented as $g^{\star}(\cdot)$ and try to maximise the out-of-sample predictive power for $r_{i,t+1}$ with a given set of parameters $z_{i,t}$. These parameters are described in Chapter 3.2. The time-lag between the two periods in t and t+1 is one year for the annual forecasts and one month for the monthly forecasts. In the nonlinear models, parameter tuning is an important factor that strongly influences the result. These tuning parameters are roughly based on similar papers (e.g. Gu et al. (2020)) and are explained in detail in the appendix to ensure comparability. Additionally, as mentioned earlier, this part of the work is prone to data snooping. Cross-validation, for example, can be used to find the perfect parameters for each model after several runs, which are adjusted exactly for the selected sample.

Therefore, for this work it was decided to make only a small adjustment of the parameters compared to the basic settings of the algorithms. These adjustments take into account the existing scientific literature and also the problem of overfitting. The problem of overfitting, for example, can be reduced by preventing the algorithm from adapting too precisely to the input data. I try to optimize only those parameters that can be logically explained. Cross-validation (CV) could also be used to find the perfect set of hyperparameters, but it too has its pitfalls. If CV is used, it will be discussed in more detail in the following section. Methods and fitting will also be explained in more detail in the following section.

4.1 Forecast Methods

This subsection attempts to provide a brief introduction to the linear and nonlinear models used in this work as well as a rationale for the choice of hyperparameters. As already explained, the focus in this paper is not on the technical aspect and the reader is not expected to be fully versed in this subject. Therefore, only a brief overview of this topic is given here. If necessary, the theory behind some hyperparameters used will be explained in more detail.

4.1.1 Linear Regression

The predominant empirical tool to find a relation between two data sets is probably linear regression. One of the most famous linear regression model in finance is the capital asset pricing model. In linear regression, a linear relationship is calculated between the scalar β and the dependent and independent variables. If only one explanatory variable is used, it is a simple linear regression. If there are several explanatory variables, as in our "Kitchen Sink" model, it is called a multiple linear regression. In our case, the dependent variable is the equity risk premium r_t and the independent variables x_i are the predictor variables. This linear approach is estimated using ordinary least squares ("OLS"), attempting to minimize the sum of squared residuals.

The simple linear regression can be expressed as follows:

$$r_t = \beta_0 + \beta_1 \cdot x_{t-1} + \epsilon_t \tag{4}$$

where the scalar β_1 can be interpreted as the coefficient of significance of the lagged predictor variable x_{t-1}. Lagged because we want to use data from the past period to calculate a forecast for the subsequent period. These variables can be one or a combination of the variables discussed in section 3.2. If multiple predictor variables are used, the calculations are performed by means of a multiple linear regression.

Linear regression works best in cases with a large number of data points and a relatively small number of predictors.

4.1.2 Elastic Net

Another linear regression model is Elastic Net. It is a regularized regression method, combining the penalties ℓ_1 from the Lasso regression and the ℓ_2 from the Ridge regression. Regularization is applied to counteract overfitting of a model. A comparison of models which are over- or underfitted can be seen in Figure 1. The model with $degree = 1$ can be classified as underfitted and the model with $degree = 15$ as overfitted. The true function is a cosine function, which is approximated by a linear function using polynomials of different degrees.

The combination of elastic net, using lasso and ridge, allows for learning a sparse model where few of the weights are non-zero like Lasso, while still maintaining the regularization properties of Ridge. The convex combination of ℓ_1 and ℓ_2 can be adjusted using the $l1 - ratio$ parameter. One of the advantages of Lasso is the reduction of model complexity and multicollinearity by reducing the weights of some coefficients towards zero, thus also performing variable selection. According to Zou & Hastie (2005), one shortcoming is that in the case of "$p > n$", lasso selects only n variables before the model is saturated. Another

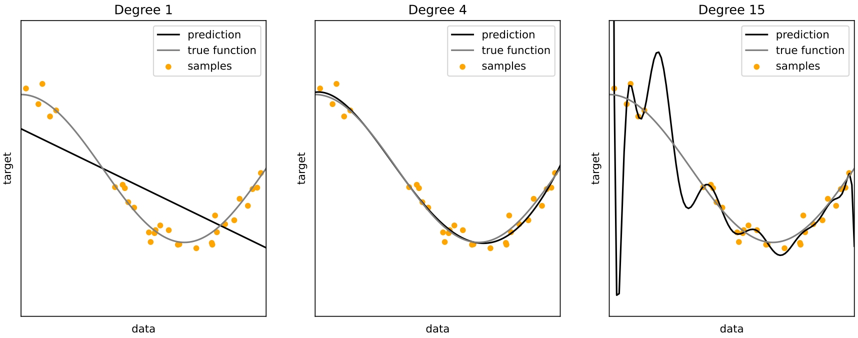


Figure 1: Prediction models with polynomial features of different degrees

shortcoming is that in the presence of multicollinear variables, lasso tends to select only one variable and ignore the rest of the variables. Ridge regression was introduced by Hoerl & Kennard (1970) as solution to linear regression models with multicollinear independent variables. The combination of these two methods in elastic net regression improves the shortcomings of lasso. This results in a sparse model with good predictive accuracy while encouraging the grouping effect [Zou & Hastie (2005)].

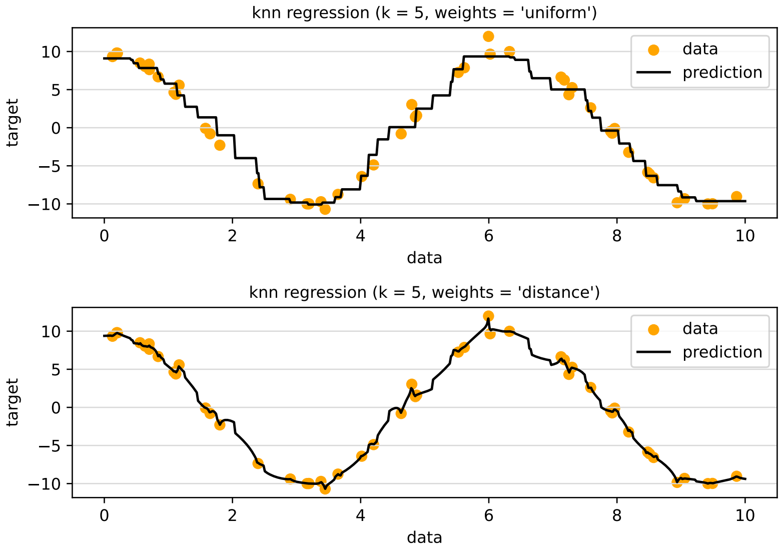
Zhou et al. (2014) proved that elastic net regression can be reduced to the linear Support Vector Machine, using squared hinge loss classification.

4.1.3 *k*-Nearest Neighbors

k-Nearest Neighbors, first introduced by Fix & Hodges (1951), is a non-parametric supervised learning method. In simple terms, this method is used to find the closest samples of a given data point. This can be achieved in two distinct ways. The samples can be found either as a function of a given number of neighbors (k-nearest neighbor method) or based on a distance to the data point (radius-based neighbor method). Since this method remembers all training data, nearest neighbors are categorized as a non-generalized machine learning method. This means that the model tends to overfit and is unstable to new data.

Nearest neighbors can also be used in cases where the data points are continuous rather than discrete variables; this is called nearest neighbor regression. In this case, the label of the data points is calculated as the mean of the k-nearest neighbors (or samples). Whereas in a classification application of this method, the data point is assigned the label of the majority of the k-nearest samples.

The results depend heavily on the hyperparameters used in this regression algorithm. Two of the most important parameters that I would like to explain briefly are the weights and the number of the nearest neighbors. First, however, it must be clarified how the Nearest Neighbors in a regression model are selected. The selection process starts with a vertical line trough the data point. This line is the starting point from which the nearest neighbors are calculated horizontally. Depending on the hyperparameter, the neighbors contribute different weights to the value depending on their distance from the data point. Two of the most common methods for selection are compared in Figure 2.

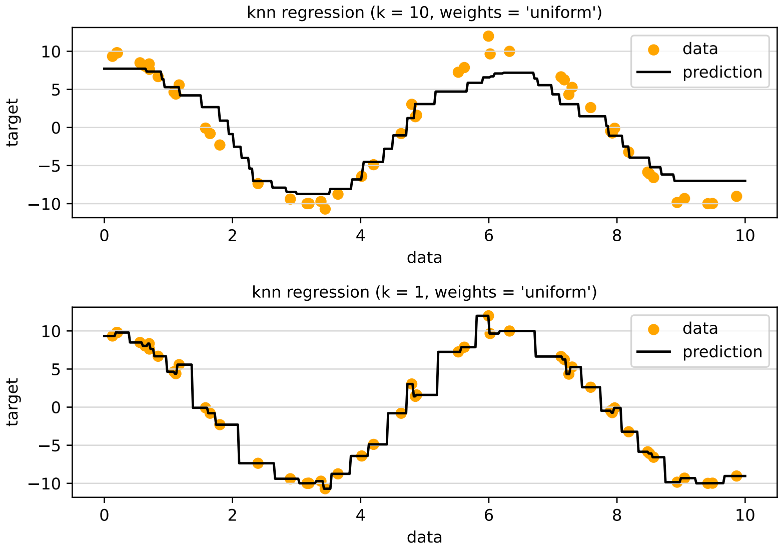

knn regression (k = 5, weights = 'uniform')

knn regression (k = 5, weights = 'distance')

Figure 2: Comparison of different weights for knn

As one can see in Figure 2, the prediction made with uniform weights is more stable than the prediction made with distance weights. When uniform weights are used, the mean value of the selected neighbors is used. If distance weights are used, the closer the neighbors are to the data point, the more they contribute.

Another important hyperparameter is compared in Figure 3. Using a higher number (k) of nearest neighbors leads to a more stable model, as each prediction is not overly influenced by a single data point.


knn regression (k = 10, weights = 'uniform')

knn regression (k = 1, weights = 'uniform')

Figure 3: Comparison of different numbers (k) of nearest neighbors

The conclusion is that using uniform weights in combination with a larger number of neighbors should prevent overfitting of the model and therefore improve the predictive power of the model. Cross-validation could be used to find the perfect set of hyperparameters for each data set. This was intentionally not done, as it would result in a myriad of combinations that are also difficult to justify in terms to data snooping.

4.1.4 Decision Tree

Decision trees are a non-parametric supervised learning method which can be used for classification and regression tasks. In this paper it is used as a regression method. The goal of this method is to build trees based on decision rules which are derived from the data features. This is achieved via if-then-else decision rules. One main hyperparameter which largely influences on how the model works is how *deep* the tree should be. A deeper tree produces more branches and is therefore more complex. The disadvantage is that such a deep tree is more prone to overfitting the model.

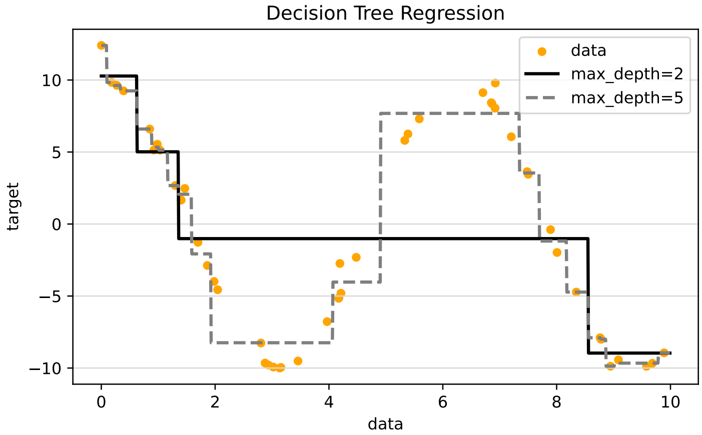


Figure 4: Comparison of different depths of decision trees

As we can see in Figure 4, the model with a depth of 2 is underfitted. Using a deeper tree, it will generate a more complex model with more decision nodes. The decision model of a decision tree with a depth of 2 from previous example can be seen in Figure 5.
Decision trees are prone to overfit on data with a large number of features. Typically, the individual decision trees have high variance and tend to overfit. Random forests attempt to mitigate these problems, so it will be interesting to compare the results of these two methods.

4.1.5 Random Forest

The following two methods are ensemble methods, meaning they combine several estimators to improve the robustness compared to a single estimator. This can be done either trough boosting or bagging. Random forest is a bagging method, also called an averaging method.

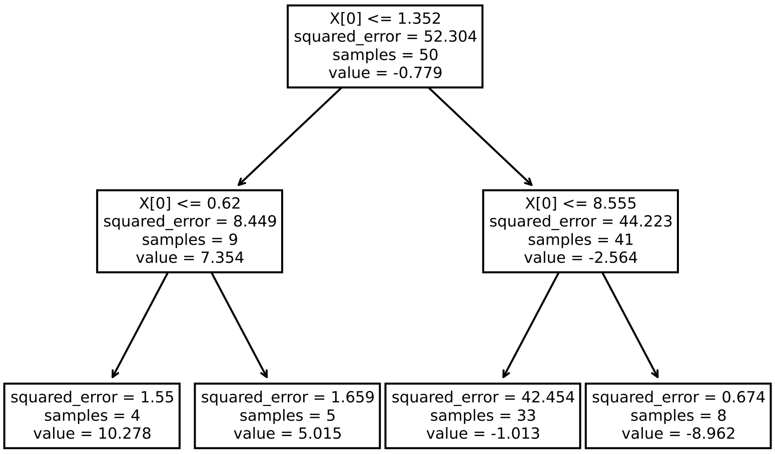


Figure 5: Decision tree with two layers (depth=2)

In a random forest regression, the sub-estimators are several decision trees - meaning it fits decision trees on sub samples of the data set. A similar technique is bootstrapping, in which subsets of a data set are randomly selected over a specified number of iterations and a specified number of variables. In a classification task, a majority vote of all sub-models is performed, and the label is the class with the most votes. For regression tasks, either the mean or the average prediction of the sub-trees is returned. A more stable outcome can be expected compared to the decision tree regression.

The decision making process can be seen in Figure 6. Several trees compute a prediction and this result is then averaged to determine the equity risk premium $r_{erp,t}$ of period t. In this case, the hyperparameters affect each individual decision tree. For example, in Figure 6 the maximum depth of each tree is defined as $max\_depth = 2$.

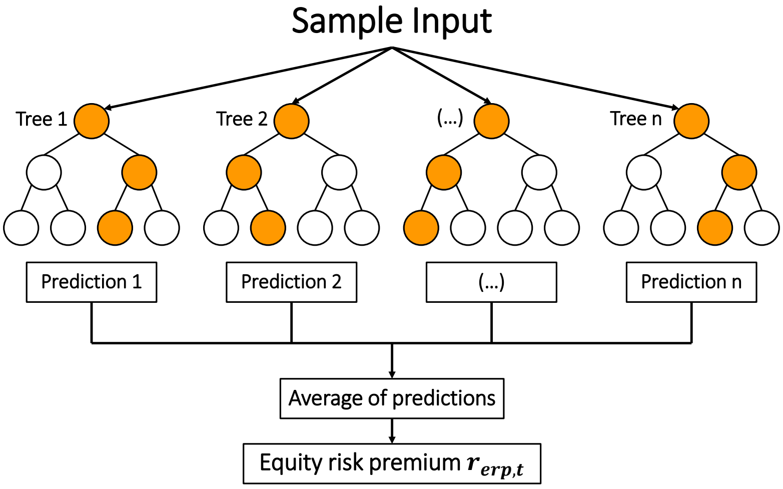


Figure 6: Bagging process of the random forest decision tree regression

4.1.6 Gradient Tree Boosting

The next ensemble method is the Gradient Boosted Decision Tree (GBDT). In contrast to random forest, GBDT is a boosting method that uses a sequence of estimators to improve the combined result relative to the individual estimators. These estimators are typically weak learners that perform only slightly better than chance, such as shallow decision trees.

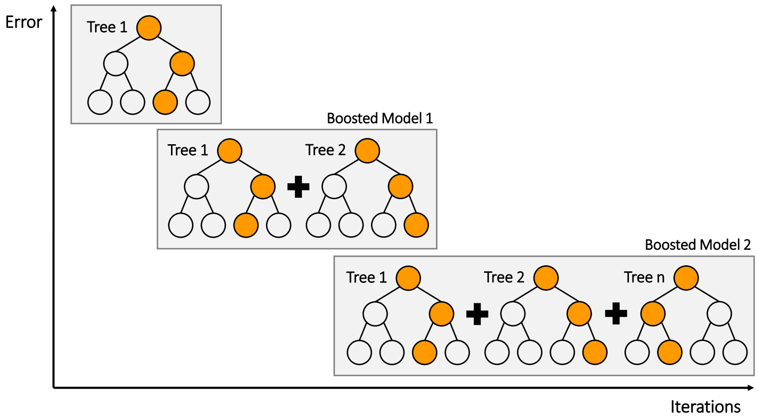


Figure 7: Schematic overview of the iterations of gradient boosted decision trees

As can be seen in Figure 7, this method starts with one decision tree and adds another decision tree in each iteration. The existing trees are not changed, but the boosted model is *extended* to include the new decision tree. This step is repeated until it no longer improves, in terms of mean squared error, or is bounded by a hyperparameter. Depending on the task, GBDT usually outperforms random forests, so it is interesting to compare these two models.

A variant of this method is the Histogram-Based Gradient Boosting (LGBM), which uses integer data structures (histograms) instead of relying on sorted continuous values to build the trees. This enhances the speed of the algorithm if the sample size $>$ 10,000.

4.1.7 Multi-Layer Perceptron

These artificial neurons were first invented in 1958 by psychologist Frank Rosenblatt based on biological neurons. These artificial neurons receive one or more inputs, process them, and produce an output. Combining multiple neurons in multiple layers creates a neural network, also known as a multi-layer perceptron (MLP). A multi-layer perceptron typically consists of three types of layers, an input layer, output layer and some hidden layers.

Figure 8 shows the structure of a multi-layer perceptron with one hidden layer. The input layer consists of all the variables which were discussed in section 3.2. These variables, also called input features, are represented by a set of neurons $\{x_i|x_1, x_2, ..., x_n\}$. In the hidden layer, the neurons transform the input with a weighted linear summation function followed by a non-linear activation function. If there are multiple hidden layers, the process is repeated in each subsequent layer with the data from the previous hidden layer. The last layer, the output layer, converts the data from the previous hidden layer into an output value, in our case the equity risk premium $r_{erp,t}$.

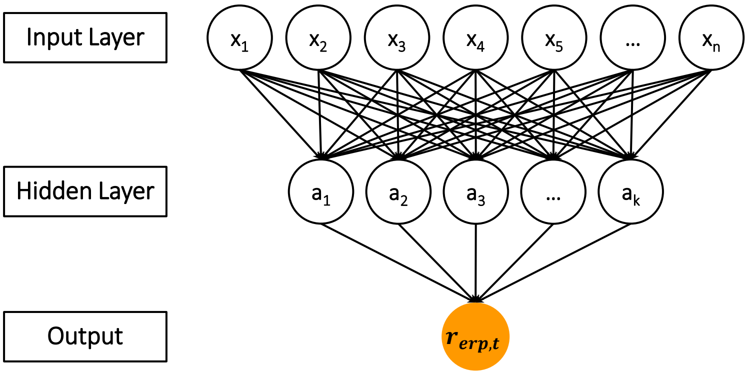


Figure 8: Multi-layer perceptron with one hidden layer

Hyperparameter-tuning greatly influences the outcome as well as the running time of code, as MLP is generally one of the most computationally demanding methods. One of these parameters is the amount of neurons and hidden layers. According to Gu et al. (2020), a neural network with three hidden layers leads to the best predictions when comparing the annual R^2_{OOS} and Sharpe ratio. The hidden layers have 32, 16, and 8 neurons, respectively.

Neural networks found applications in many fields such as data mining, medical diagnosis, pattern recognition and so on. So it will be interesting to see how the MLP will perform with the financial data used in this work.

4.1.8 Data Preprocessing

In addition to handling missing data, data preprocessing is also about standardizing features. Many machine learning algorithms require that these values look like normally distributed data. This means that they have a mean of zero and a unit variance. If a feature has a mean or variance which are a lot bigger than others, it might dominate the other features. Some learning algorithms, such as ℓ_1 and ℓ_2 regularizers of linear models, assume that all features are centered around zero or have variance of the same order of magnitude. This is the case when the elastic net algorithm is used. Other distance-based models can be improved by using a range of scaling techniques. There is not one solution that solves everything but rather trial and error. If any type of feature scaling is used, it is described in Section 5. Tree-based algorithms, on the other hand, are generally insensitive to the size of the variables. Therefore, no standardization is applied to the features when these models were used.

The standardized features can also be interpreted as the Z-score, i.e. the number of standard deviations the feature is above or below the population mean. Equation 5 is the formula for the Z-score Z, where μ is the population mean and σ is the population standard deviation. In Equation 6, we adjust Equation 5 with the variables used in this work, where $x_{t,s}$ is the standardized value for the feature x in period t and X corresponds to the vector of features from the period 0 up until t.

$$Z = \frac{x - \mu}{\sigma}, \tag{5}$$

$$x_{t,s} = \frac{x_t - \bar{X}}{\sigma_X}, \tag{6}$$

To avoid data snooping, the feature scaling is performed only on the training data. The scaling-factors of this period are then used to scale the features used for forecasting. If, on the other hand, the entire period of each feature were scaled, then even a sample period would contain future information. This can be explained by the fact that the scaling-factors are determined over the entire period and are then applied to each individual factor. If, for example, a factor was always very low in the past and increases in the future, it will be assigned a value of < 0 after standardization. From this it can be concluded that the factor will increase in the future because it must reach a value of > 0 at some point.

4.2 Forecast Evaluation

In terms of forecast evaluation measures this paper investigates the in-sample and out-of-sample R^2 and adjusted $\bar{R}^2$, the mean-squared error $MSFE$ as well as the $\Delta RMSE$. R^2_{OOS} is used to evaluate the forecast accuracy in terms of $MSFE$ and defined as suggested in Campbell and Thompson (2008).

$$R^2 = 1 - \frac{MSFE^M}{MSFE^{bmk}} \tag{7}$$

$$\bar{R}^2 = 1 - (1 - R^2) \times \frac{N - 1}{N - p - 1} \tag{8}$$

where N is the sample size and p is the number of independent variables.

$$\Delta RMSE = \sqrt{MSFE_{bmk}} - \sqrt{MSFE_M} \tag{9}$$

where $MSFE^M$ corresponds to the mean-squared error of the respective model and $MSFE^{bmk}$ corresponds to the mean-squared error of the benchmark:

$$MSFE^M = \frac{1}{p} \sum_{t=T+1}^{T+p} (r_t - \hat{r_t}^M)^2 \tag{10}$$

where $[T+1; T+p]$ is the out-of-sample evaluation period and $\hat{r_t}^M$ is the predicted excess return generated by the methods we compare. This $\hat{r_t}^M$ is compared to the actually occurred excess return r_t from period t.

4.2.1 Benchmark

The choice of benchmark is already debated in the scientific literature. It can be either the historical average of monthly or annual returns, respectively, or a zero mean. Both views are represented in the literature and there are also enough arguments for and against them. The following quote from Gu et al. (2020, p.2246) comprehensively summarizes the advantages of the zero mean over the historical mean:

> ”In many out-of-sample forecasting applications, predictions are compared against historical mean returns. Although this approach is sensible for the aggregate index or long-short portfolios, for example, it is flawed when it comes to analyzing individual stock returns. Predicting future excess stock returns with historical averages typically under-performs a naive forecast of zero by a large margin.”

However, in this paper the historical average is used, as it makes, in my opinion, more relevance from an investor's point of view. After all, an investor would hardly invest in a market where he expects a market risk premium of zero percent. Because then he might as well just invest his wealth in the risk-free asset. With this rationale, I think the historical average has greater significance in this paper.

4.3 Directional Prediction

After determining the statistical significance of each method, the predicted values are converted into signals representing the predicted directional movement of the S&P 500 index. It should be mentioned that the methods are not ”designed” to predict discrete variables, so it is to be expected that they will not perform as well as they could. These signals can be either *”long”*, for positive forecasts, or *”short”*, for negative signals. The strategy is then to open the same position in the market as the model predicts. One could also take a different approach and only invest if the signal is *”long”* and stay out of the market if the signal is *”short”*. To compare the different methods, the annualized Sharpe Ratio[3] (”SR”) is calculated. Annualized because we make the directional forecast only on the basis of the monthly forecasts. The annualized SR is defined as:

$$SR_{Annualized} = \frac{T\bar{r}}{\sqrt{T}\sigma_r} = \sqrt{T}\frac{\bar{r}}{\sigma_r} \tag{11}$$

where T is the frequency, in our case monthly an therefore equal to 12 (periods). The average monthly excess returns are described as $\bar{r}$ and the monthly standard deviation of

[3] Formula for this metric stems from Sharpe (1994)

the excess return is σ_r. This metric should allow a rough comparison between the different methods.

5 Empirical Study

In this section, the results of the equity premium predictions are presented. First, the annual predictions are compared with the results of Welch & Goyal (2008) to see if the new data from 2005 until 2020 provide new insights. Only the OLS method was used for this purpose. Next, we look at the monthly forecasts and compare them with different methods as presented in Section 4.1. In the last subsection, directional forecasts are used to create long and short signals. These signals are then used to create and compare long-short portfolios.

5.1 Annual Prediction

First we look at the annual predictions of different variables. For each variable, the parameters are calculated by linear regression, more precisely by ordinary least squares.

In Figure 9 we can see the in-sample (IS) and out-of-sample (OOS) performance as measured by the cumulative mean squared error (MSE). The lines in each graphs plot the sum of the null minus the alternative, where the dotted line depicts the IS results and the solid line the OOS results. The null is in our case the mean equity risk premium over the period, and the alternative is the model with the specified variable. The IS periods start with the start date specified in Table 1 and end in 2020. An expanding window is applied for the OOS predictions. A minimum of 20 years is used as training data, and thereafter the window is expanded by one year for each forecast. The prediction period is one year, which means that one prediction is made at each iteration. This procedure is in line with that of Welch & Goyal (2008). If the line increases, the alternative predicts better than the null. This means that the predictions of the models have a lower error compared to the historical mean. But as we can see in the results, no model outperforms consistently the null. There are some periods where this is the case, especially during the Oil Shock in 1974. Without this, no model would outperform the null significantly. Section 5.3 presents the results when these predictions are used for market timing.

Other approaches could have been taken for the window and the forecast period. The statistics for a rolling window is attached in the appendix. The window for this approach is also 20 years, but in this case it is set to the last 20 years for each forecast iteration. In most cases, the results are worse compared to the expanding window. The forecast period for this is again one year.

We can see in Figure 9 that no model outperforms the null consistently over a longer period. The models using long-term yields (**lty**) and the Treasury bill rate (**tbl**) show a sharp increase around the oil shock in 1974, but fall back to previous levels almost immediately thereafter. If we would exclude this period from the results, most models

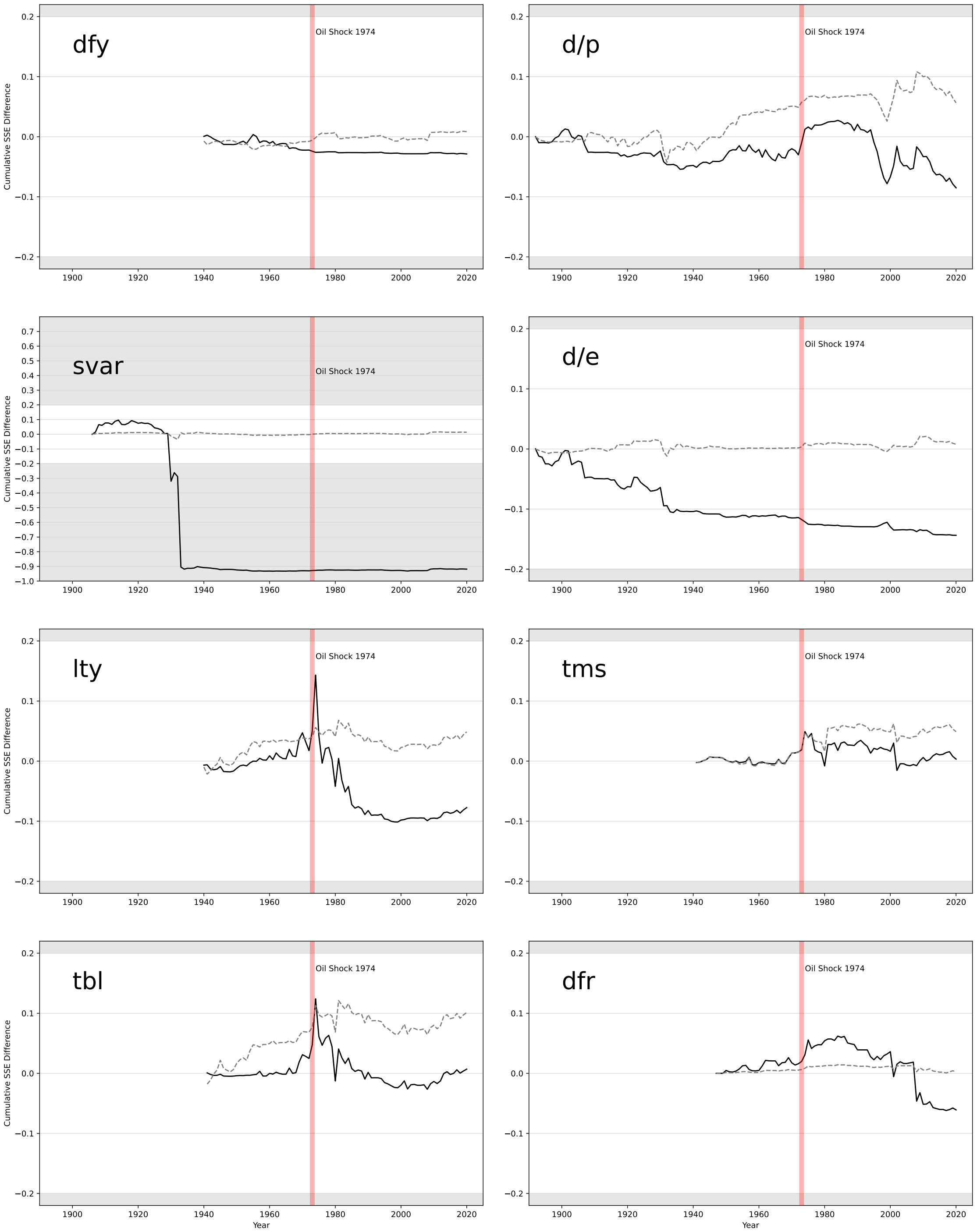


Figure 9: Annual predictive performance using linear regression. Explanation: These figures show the in-sample (gray line) and out-of-sample (black line) performance of the predictions. Performance is measured by the cumulative squared prediction error. This is calculated by subtracting the squared prediction error of the model shown from the squared prediction error of the historical average model. Where the line increases, the model using the depicted variable performs better than the model using the historical mean. The mean is calculated either over the entire period (in-sample) or over the entire period up to the prediction (out-of-sample).

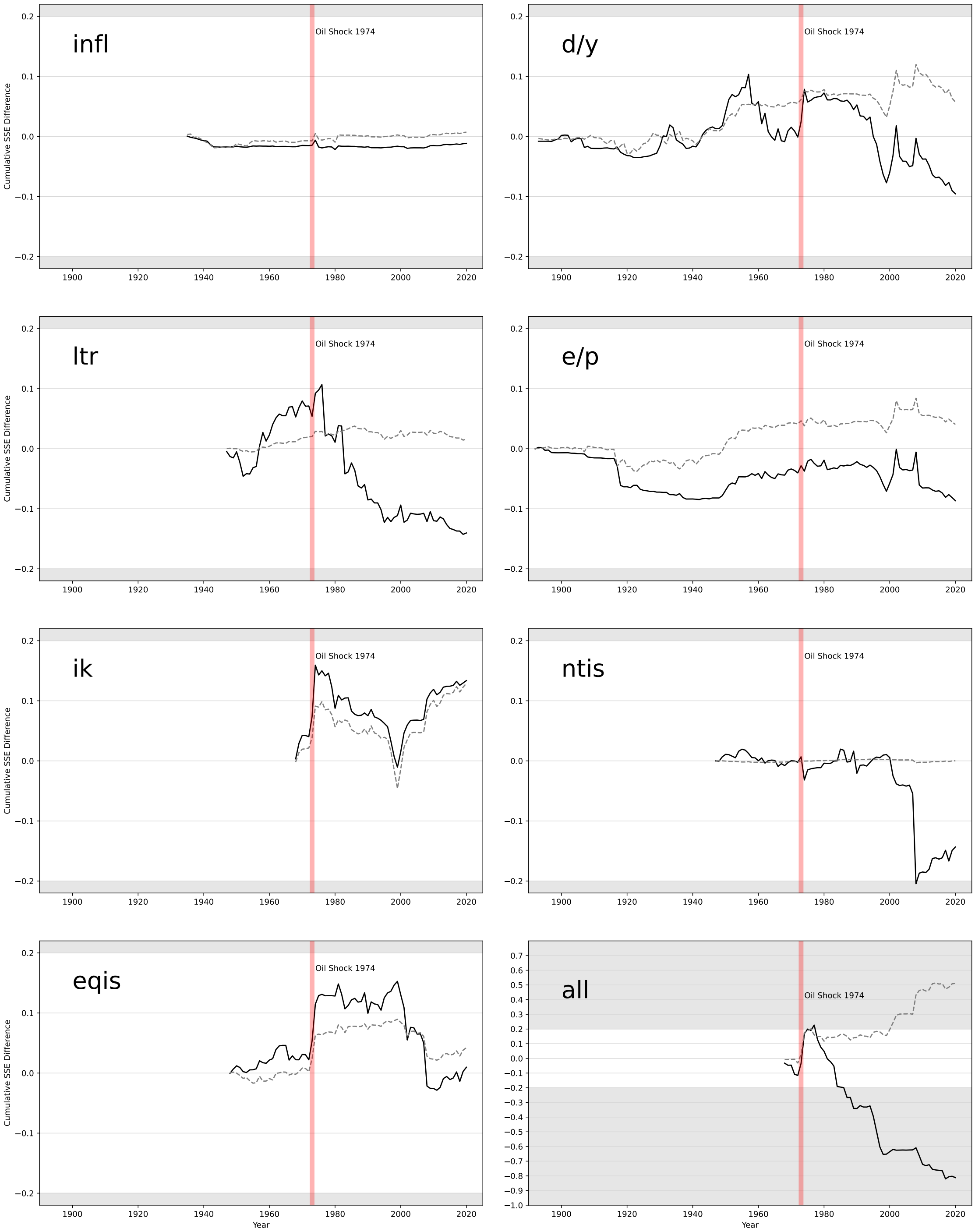


Figure 9 (cont.): Annual predictive performance using linear regression

			IS		**OOS**		
	Variable	**Start**	$\mathbf{R^2}$	$\bar{\mathbf{R}}^2$	$\mathbf{R^2}$	$\bar{\mathbf{R}}^2$	$\mathbf{\Delta}$**RMSE**
dfy	Default yield spread	1919	0.41	−0.85	−1.34	−2.62	−0.11
infl	Inflation	1914	0.28	−0.90	−0.45	−1.64	−0.04
svar	Stock variance	1885	0.33	−0.55	−22.10	−23.18	−2.00
d/e	Dividend payout ratio	1871	0.18	−0.60	−3.27	−4.08	−0.30
lty	Long term yield	1919	2.33	1.09	−3.62	−4.93	−0.29
tms	Term spread	1920	2.38	1.13	0.16	−1.12	0.01
tbl	Treasury-bill rate	1920	4.91	3.69	0.33	−0.95	0.03
dfr	Default return spread	1926	0.13	−1.25	−3.13	−4.56	−0.25
d/p	Dividend price ratio	1871	1.31	0.53	−1.93	−2.73	−0.18
d/y	Dividend yield	1872	1.33	0.55	−2.17	−2.98	−0.20
ltr	Long term return	1926	0.83	−0.55	−7.22	−8.71	−0.57
e/p	Earning price ratio	1871	0.94	0.16	−1.97	−2.77	−0.18
ik	Investment capital ratio	1947	9.16	7.38	8.94	7.16	0.77
ntis	Net equity expansion	1926	0.01	−1.38	−7.37	−8.86	−0.59
eqis	Pct equity issuing	1927	2.22	0.84	0.50	−0.91	0.04
all	Kitchen sink	1947	36.30	7.99	−54.33	−122.92	−4.07

Table 1: Statistics of the annual predictions. Explanation: This table shows the in-sample (IS) and out-of-sample (OOS) statistics of the predictions. All values are in percent. The variables are explained in Section 3.2 and the calculations in Section 4.2. The Start column shows the date when the data availability of the respective variable starts. The first forecast is then made after 20 years.

would perform even worse. Or in the case of Treasury bill rate (**tbl**), Term spread (**tms**) and Percent equity issuing (**eqis**), the positive $\Delta RMSE$ would become negative. The Stock variance (**svar**) and Kitchen sink (**all**) models perform poorly in annual predictions compared to the other models. The poor OOS performance of the Kitchen sink model can be explained by overfitting. This is the disadvantage of using too many independent variables to achieve good IS performance. These models usually don't perform very good OOS. Only four models have a positive $\Delta RMSE$ as can be seen in Table 1. But only one model, the Investment capital ratio **ik**, has a positive adjusted R^2. However, as can be seen in Figure 9, much of this result is also due to the 1974 oil shock. Another point is that this model does not consistently outperform the benchmark. In the period from about 1975 to 2000, the historical mean performs better as a predictor. Thus, it can be concluded that no model consistently performs better than the historical mean when conducting annual forecasts using linear regression. This conclusion is consistent with the Welch & Goyal (2008) paper, and this has not changed over the past 15 years.

In Section 5.3 these models are used to generate signals for market timing. These portfolios are then compared to a buy-and-hold strategy to see if they can outperform it.

5.2 Monthly Prediction

In this section we look at the performance of different models using different algorithms as discussed in Section 4.1. For the linear regression, elastic net, k-nearest neighbors and decision tree we use the same methodology as in the annual predictions, meaning that every variables is tested for their performance. With the random forest, GBM, LGBM an neural networks only the the Kitchen sink model (**all**) is used. The forecast period for the monthly forecasts, as in the annual section, is one period, which corresponds to one month.

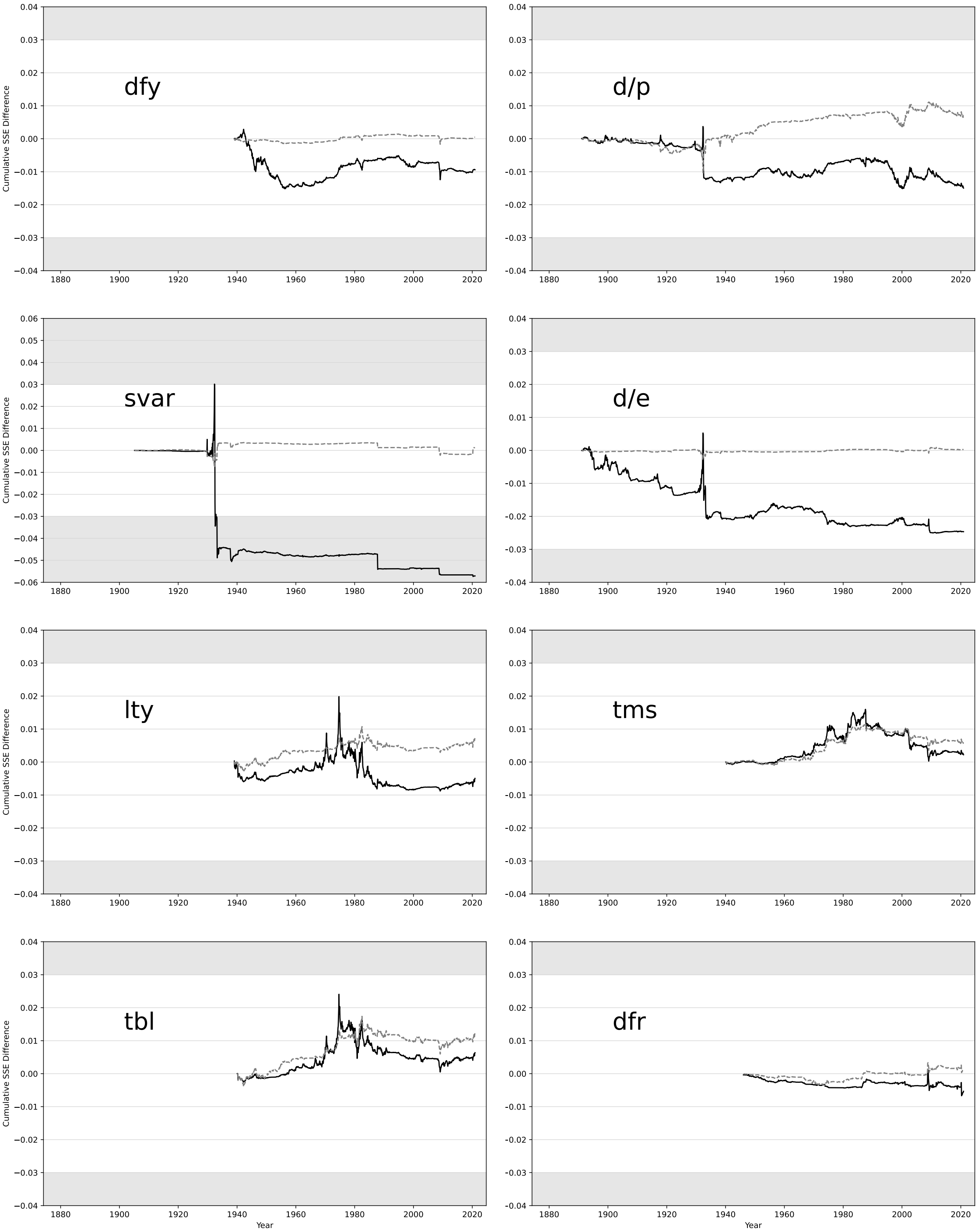


Figure 10: Monthly predictive performance using linear regression. For explanation, please refer to Figure 9.

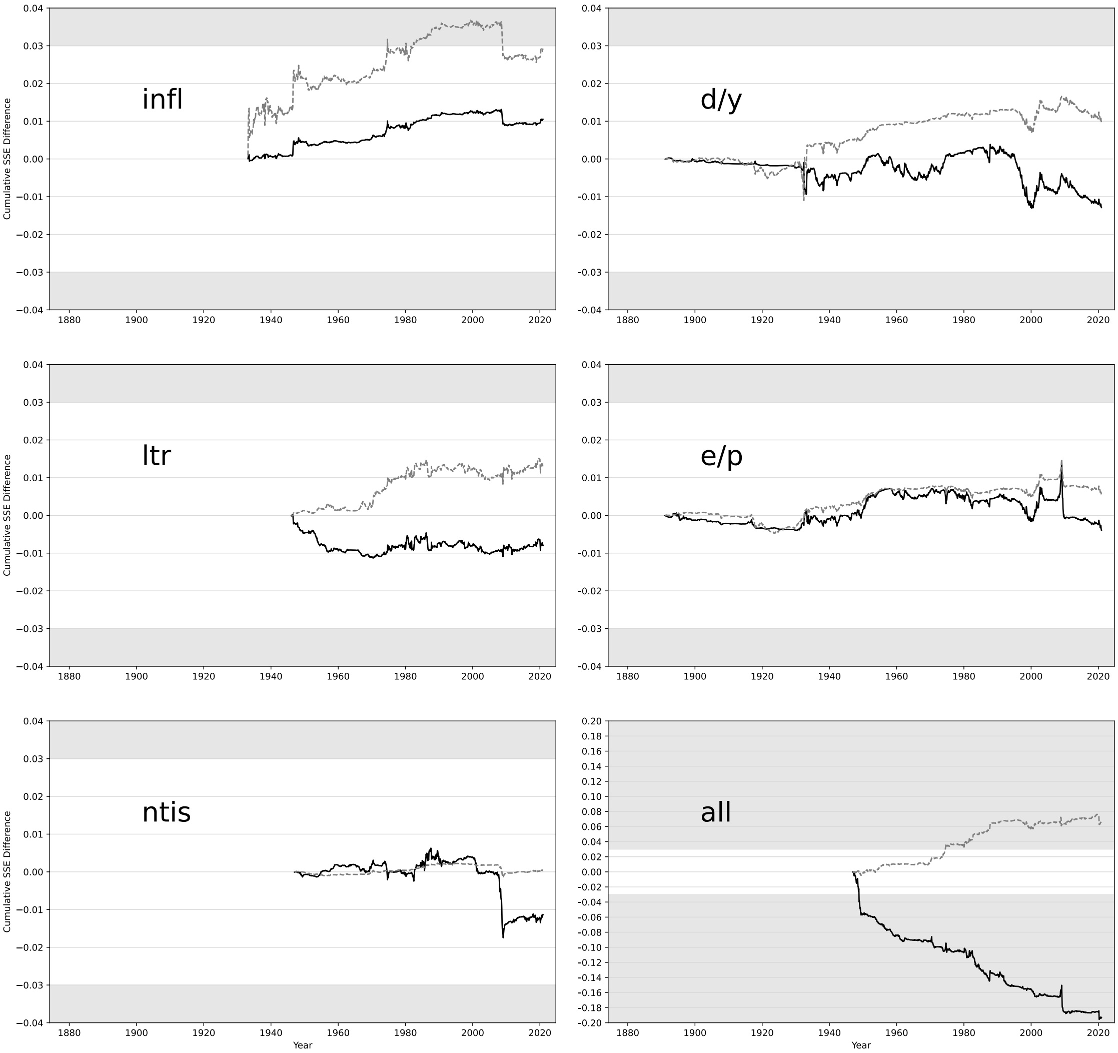


Figure 10 (cont.): Monthly predictive performance using linear regression

			IS		OOS		
	Variable	**Start**	$\mathbf{R^2}$	$\mathbf{\bar{R}^2}$	$\mathbf{R^2}$	$\mathbf{\bar{R}^2}$	**ΔRMSE**
dfy	Default yield spread	1919-01	0.03	−0.07	−0.52	−0.62	−0.01
infl	Inflation	1913-02	1.20	1.11	0.43	0.34	0.01
svar	Stock variance	1885-02	0.03	−0.04	−1.58	−1.65	−0.04
d/e	Dividend payout ratio	1871-01	0.01	−0.06	−0.65	−0.71	−0.02
lty	Long term yield	1919-01	0.40	0.30	−0.27	−0.37	−0.01
tms	Term spread	1920-01	0.32	0.22	0.13	0.02	0.00
tbl	Treasury-bill rate	1920-01	0.70	0.60	0.36	0.26	0.01
dfr	Default return spread	1926-01	0.09	−0.02	−0.34	−0.45	−0.01
d/p	Dividend price ratio	1871-01	0.16	0.10	−0.39	−0.46	−0.01
d/y	Dividend yield	1871-02	0.26	0.19	−0.34	−0.40	−0.01
ltr	Long term return	1926-01	0.84	0.73	−0.50	−0.61	−0.01
e/p	Earning price ratio	1871-01	0.15	0.08	−0.10	−0.17	0.00
ntis	Net equity expansion	1926-12	0.03	−0.08	−0.73	−0.85	−0.02
all	Kitchen sink	1926-12	4.22	2.68	−12.48	−14.28	−0.25

Table 2: Statistics of the monthly predictions using linear regression. For explanation, please refer to Table 1.

As can be seen in Figure 10 (please note the change of scale compared to the annual predictions), again no variable seems to produce a stable prediction model. The kitchen sink (**all**) model again overfits the training data and performs poorly in the forecasts. The Treasury bill rate (**tbl**) has an adjusted OOS $\bar{R}^2$ of 0.26, but does not yield a stable model. There is a peak around the 1970-1980 period, followed by a sharp decline. The only other model which produces a notable adjusted OOS $\bar{R}^2$ of 0.34 is the Inflation (**infl**). This model also appears to be stable over the entire period, with a small drop around the financial crisis in 2008. To see if this variable can be used to generate signals that provide positive excess returns, see Section 5.3. The Term spread (**tms**) also generates a positive adjusted OOS $\bar{R}^2$, but this is only 0.02. Moreover, the model also does not appear to be stable, similar to the Treasury bill rate (**tbl**), as seen in Figure 10.

In Table 3 we can see the statistics of the monthly predictions using elastic net regression. Again, the Inflation (**infl**) and the Treasury bill rate (**tbl**) models are able to produce a positive OOS $\bar{R}^2$. Many models yield an IS R^2 of 0, which can be explained by variable selection. If the model decides that no variable is suitable, it uses the historical mean, which obviously has no advantage over the benchmark, as it uses the historical mean as well. As a reminder, the IS statistics are calculated using the whole period from the start date until 2020. Nevertheless, the OOS R^2 can be different from the IS R^2, as at each period the model decides if each variable is suitable. The results of the models are similar to those of the linear regression models in that the main difference is the variable selection. One important difference is that the Kitchen sink (**all**) model performs better using the elastic net regression, as it prevents the overfitting of the model on too many useless variables. For this method, hyperparameter-tuning was performed using cross-validation, choosing the best parameters for each period and each forecast.

			IS		**OOS**		
	Variable	**Start**	$\mathbf{R^2}$	$\mathbf{\bar{R}^2}$	$\mathbf{R^2}$	$\mathbf{\bar{R}^2}$	$\mathbf{\Delta}$**RMSE**
dfy	Default yield spread	1919-01	0.00	−0.10	−0.17	−0.27	0.00
infl	Inflation	1913-02	1.20	1.11	0.37	0.27	0.01
svar	Stock variance	1885-02	0.00	−0.07	0.11	0.03	0.00
d/e	Dividend payout ratio	1871-01	0.00	−0.06	−0.50	−0.57	−0.01
lty	Long term yield	1919-01	0.38	0.28	−0.10	−0.20	0.00
tms	Term spread	1920-01	0.00	−0.10	−0.08	−0.18	0.00
tbl	Treasury-bill rate	1920-01	0.70	0.60	0.23	0.12	0.00
dfr	Default return spread	1926-01	0.00	−0.11	−0.43	−0.54	−0.01
d/p	Dividend price ratio	1871-01	0.00	−0.06	0.02	−0.04	0.00
d/y	Dividend yield	1871-02	0.00	−0.06	−0.03	−0.09	0.00
ltr	Long term return	1926-01	0.83	0.72	−0.56	−0.67	−0.01
e/p	Earning price ratio	1871-01	0.00	−0.06	−0.14	−0.21	0.00
ntis	Net equity expansion	1926-12	0.00	−0.11	−0.82	−0.93	−0.02
all	Kitchen sink	1926-12	2.58	1.02	−0.31	−1.92	−0.01

Table 3: Statistics of the monthly predictions using elastic net regression. For explanation, please refer to Table 1.

For k-nearest neighbors, the independent variables were standardized as described in the 4.1.8 sections. In this way, the performance could be improved. Nevertheless, the models using k-nearest neighbors were not able to produce forecasts which outperformed

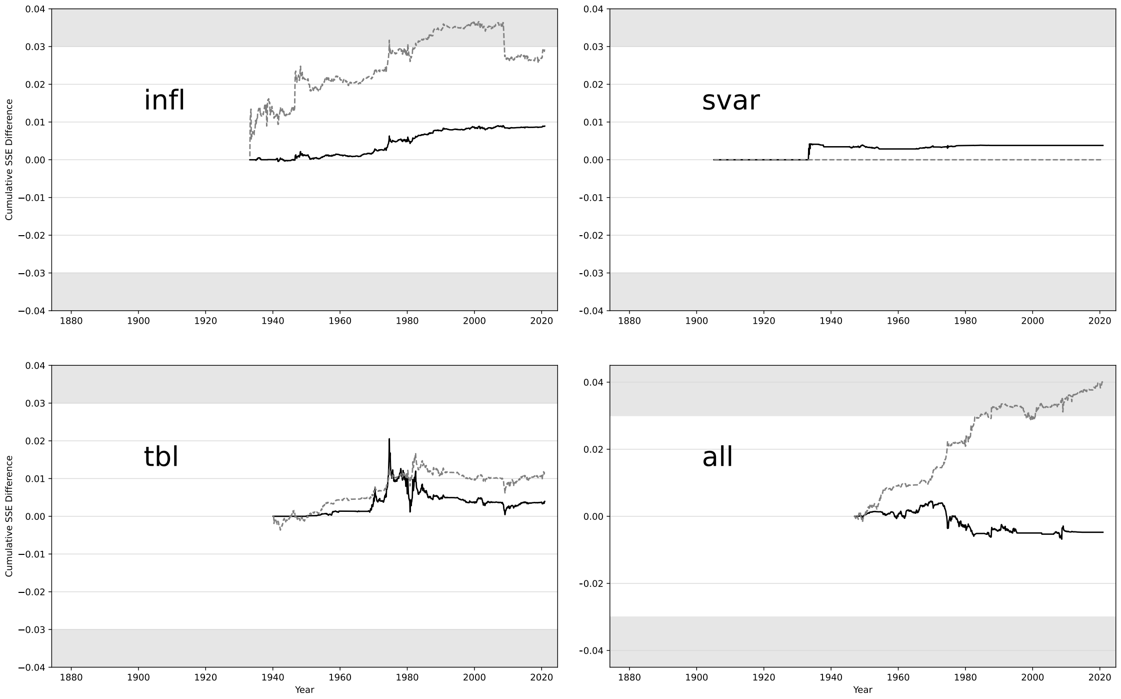


Figure 11: Monthly predictive performance using elastic net regression (positive $\bar{R}^2_{OOS}$). For explanation, please refer to Figure 9.

the benchmark. We can see in Table 4, that no model achieved a positive adjusted $\bar{R}^2_{OOS}$. A detailed overview of the prediction accuracy can be found in the Appendix (Figure 15). One point that can be elaborated on is that the Kitchen sink model (**all**) performs better with k-nearest neighbors than with linear regression. Unlike the previous methods, the Kitchen sink model is also one of the better performing models. The conclusion can be drawn, that the prediction accuracy of this method can be increased by using more independent variables, in contrast to the linear regression. Overall, the k-nearest neighbor algorithm does not increase the prediction performance compared to the previous methods. And as seen in Figure 15, apart from a few brief outliers, there was no extended period in which the benchmark was consistently outperformed.

The following methods all use decision trees as the basis for their algorithm. First, we consider the basic decision tree method. Then we compare all decision tree methods using the Kitchen Sink model. Most of the models do not seem to perform very well, only the Dividend yield (**dy**) gives a positive adjusted $\bar{R}^2_{OOS}$. The Kitchen sink model (**all**) does not perform particularly well or poorly in comparison with the other models. If we compare the performances of all decision tree based methods, as seen in Table 6, we can see that each method improves on the previous method. This is not a surprise, as the advanced models try to prevent the models from overfitting too much. This can be seen in the adjusted $\bar{R}^2_{IS}$. This leads to better predictive performance, as shown by the adjusted $\bar{R}^2_{OOS}$. Nevertheless, no model is able to outperform the benchmark. In Figure 12 we can

			IS		OOS		
	Variable	**Start**	$\mathbf{R^2}$	$\mathbf{\bar{R}^2}$	$\mathbf{R^2}$	$\mathbf{\bar{R}^2}$	$\mathbf{\Delta}$**RMSE**
dfy	Default yield spread	1919-01	3.49	3.39	−3.84	−3.95	−0.08
infl	Inflation	1913-02	0.72	0.63	−3.18	−3.28	−0.08
svar	Stock variance	1885-02	2.56	2.48	−3.61	−3.68	−0.09
d/e	Dividend payout ratio	1871-01	4.07	4.00	−3.45	−3.51	−0.08
lty	Long term yield	1919-01	3.47	3.37	−5.40	−5.51	−0.11
tms	Term spread	1920-01	4.29	4.19	−4.00	−4.10	−0.08
tbl	Treasury-bill rate	1920-01	4.04	3.95	−4.19	−4.30	−0.09
dfr	Default return spread	1926-01	2.76	2.66	−6.49	−6.61	−0.13
d/p	Dividend price ratio	1871-01	2.75	2.68	−3.97	−4.03	−0.10
d/y	Dividend yield	1871-02	4.46	4.40	−3.47	−3.54	−0.09
ltr	Long term return	1926-01	5.86	5.75	−5.78	−5.90	−0.12
e/p	Earning price ratio	1871-01	3.87	3.80	−4.82	−4.89	−0.12
ntis	Net equity expansion	1926-12	2.84	2.73	−10.45	−10.57	−0.21
all	Kitchen sink	1926-12	6.75	5.26	−1.70	−3.34	−0.04

Table 4: Statistics of the monthly predictions using k-nearest neighbors. For explanation, please refer to Table 1.

also observe that there was almost no prolonged period in which the algorithm produced predictions that were better than the simple historical mean.

			IS		OOS		
	Variable	**Start**	$\mathbf{R^2}$	$\mathbf{\bar{R}^2}$	$\mathbf{R^2}$	$\mathbf{\bar{R}^2}$	$\mathbf{\Delta}$**RMSE**
dfy	Default yield spread	1919-01	0.74	0.64	−0.04	−0.14	0.00
infl	Inflation	1913-02	1.60	1.51	−0.34	−0.43	−0.01
svar	Stock variance	1885-02	0.86	0.79	−0.45	−0.52	−0.01
d/e	Dividend payout ratio	1871-01	0.44	0.38	−1.51	−1.57	−0.04
lty	Long term yield	1919-01	1.00	0.89	−8.55	−8.66	−0.18
tms	Term spread	1920-01	1.54	1.44	−0.84	−0.95	−0.02
tbl	Treasury-bill rate	1920-01	0.99	0.89	−0.27	−0.37	−0.01
dfr	Default return spread	1926-01	0.95	0.84	−7.84	−7.96	−0.16
d/p	Dividend price ratio	1871-01	1.89	1.83	−3.27	−3.33	−0.08
d/y	Dividend yield	1871-02	7.59	7.53	3.57	3.50	0.09
ltr	Long term return	1926-01	1.62	1.51	−7.74	−7.86	−0.16
e/p	Earning price ratio	1871-01	0.44	0.38	−0.55	−0.62	−0.01
ntis	Net equity expansion	1926-12	1.51	1.40	−47.66	−47.82	−0.90
all	Kitchen sink	1926-12	1.72	0.14	−12.30	−14.10	−0.25

Table 5: Statistics of the monthly predictions using decision trees. For explanation, please refer to Table 1.

				IS		OOS		
Method		**Variable**	**Start**	$\mathbf{R^2}$	$\mathbf{\bar{R}^2}$	$\mathbf{R^2}$	$\mathbf{\bar{R}^2}$	$\mathbf{\Delta}$**RMSE**
Trees	**all**	Kitchen sink	1926-12	1.72	0.14	−12.30	−14.10	−0.25
Forest	**all**	Kitchen sink	1926-12	3.22	1.67	−9.42	−11.17	−0.19
GBDT	**all**	Kitchen sink	1926-12	2.62	1.06	−4.78	−6.46	−0.10
LGBM	**all**	Kitchen sink	1926-12	2.41	0.85	−0.63	−2.24	−0.01

Table 6: Statistics of the monthly predictions of different algorithms based on decision trees. For explanation, please refer to Table 1.

The last method we discuss is the multi-layer perceptron. In previous work, e.g., Gu et al. (2020), this was the method that provided the best predictions, but using other independent variables. However, this was not the case for this set of variables. Measured by the adjusted $\bar{R}^2_{OOS}$, this model is one of the worst performing compared to other methods and models. In general, the performance of all methods is very sensitive to

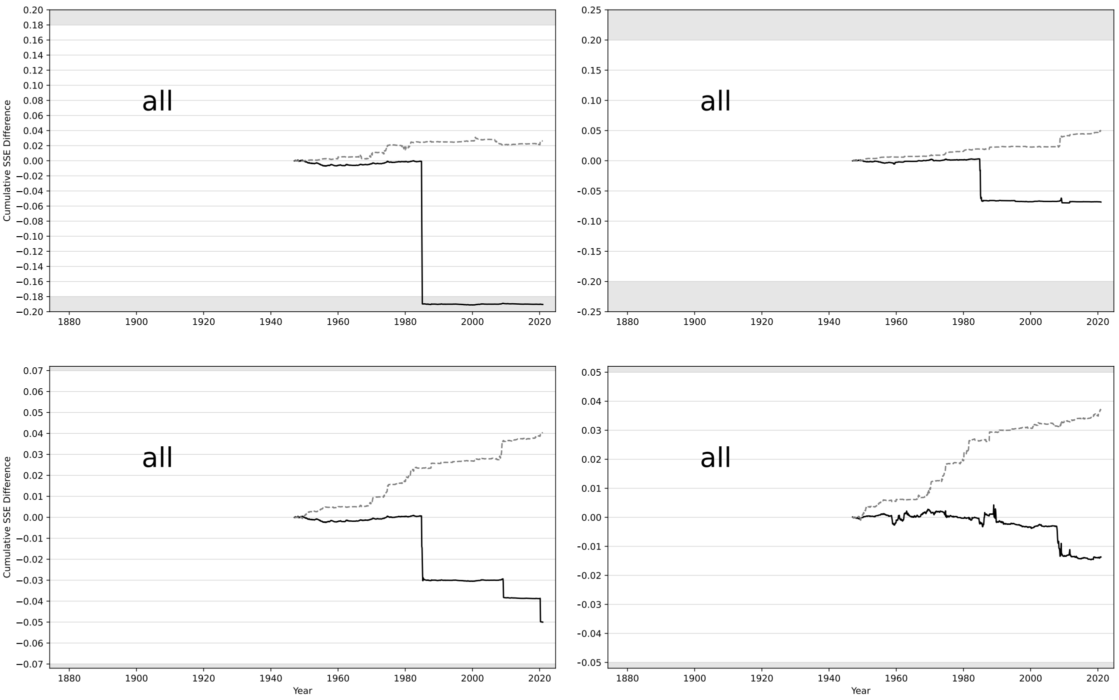


Figure 12: Monthly predictive performance using different algorithms based on decision trees (top left: Decision trees, top right: Random forest, bottom left: GBDT, bottom right: LGBM. For explanation, please refer to Figure 9.

the hyperparameters used, but MLP was comparatively the most sensitive. This fact, of course, complicates the search for the most appropriate hyperparameters. To improve the predictive power somewhat, the samples were normalized. This means that the samples are converted so that all values lie between -1 and +1. Nevertheless, this method does not seem to yield good results when using this set of predictors.

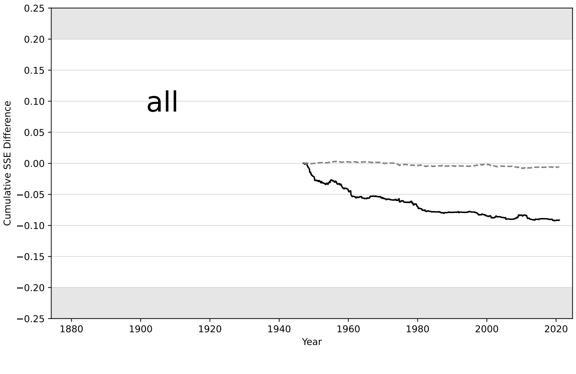


Figure 13: Monthly predictive performance using the multi-layer perceptron. For explanation, please refer to Figure 9.

			IS		**OOS**		
	Variable	**Start**	$\mathbf{R^2}$	$\bar{\mathbf{R}}^{\mathbf{2}}$	$\mathbf{R^2}$	$\bar{\mathbf{R}}^{\mathbf{2}}$	**ΔRMSE**
all	Kitchen sink	1926-12	−0.36	−1.97	−5.91	−7.61	−0.12

Table 7: Statistics of the monthly predictions using the multi-layer perceptron. For explanation, please refer to Table 1.

5.3 Long-Short Portfolios

In this section, several portfolios are created and compared. The structure of the variables displayed for each method is consistent with the previous chapters. The calculated statistics such as the average monthly return $\bar{r}$, the standard deviation σ and the annual Sharpe ratio SR are calculated over the entire period for which the variables are available. This is also indicated by the *Start* column, which indicates the date of the first month for which these variables are available. The delta column ΔSR shows the difference between the Sharpe ratios of the portfolio and the benchmark. If this measure is positive, the portfolio using the signals achieves a higher Sharpe ratio than the buy-and-hold strategy. When comparing the monthly returns of the different models, the same start date must be chosen, otherwise the comparison would not be possible. Therefore, the start date is January 1947, which is also the date of the first prediction of the kitchen sink model (**all**), since 20 years are used as the minimum for the training data. Logically, the portfolio can only outperform the benchmark if it correctly predicts a negative return and creates a short position, since the portfolio is long only.

The first statistics we compare are the portfolios using the linear regression, as can be seen in Table 8. Let us recall that in Table 2 the Inflation (**infl**) and the Treasury bill rate (**tbl**) produced the highest adjusted $\bar{R}^2$, but this did not translate into a positive ΔSR. This means that numerical prediction accuracy should not be the only metric for selecting variables for signal generation. Only the Term spread (**tms**) produced a positive adjusted $\bar{R}^2$ as well as a positive ΔSR. However, this difference in the Sharpe ratio can be described as only marginal. Nevertheless, if one invested into the market according to these signals, he would have outperformed the market over the period of 1920 until 2020.

Next we look at the performance of the elastic net. Using the elastic net algorithms improves the SR of some portfolios, such as the Stock variance (**svar**), the Default yield spread (**dfy**), and the Kitchen sink model (**all**). Overall, all models perform either better or equally well compared to linear regression, which is not a surprising result since the elastic net also performs linear regression, but with coefficient regularization.

Looking at the results of the portfolios using the k-nearest neighbor algorithm in Table 10, we can see that no portfolio is able to outperform the benchmark. This is not a surprise, as we have already discussed in the previous section that this method does not yield positive adjusted $\bar{R}^2_{OOS}$ results. Another point is that all models except the Kitchen sink (**all**) model perform far worse when compared to the benchmark. Thus, we could conclude that the results of all portfolios using this method are consistent with the results of their predictive performance.

			Portfolio			**Benchmark**			
	Variable	**Start**	$\bar{\mathbf{r}}$	σ	**SR**	$\bar{\mathbf{r}}$	σ	**SR**	**ΔSR**
dfy	Default yield spread	1919-01	0.60	4.31	0.48	0.68	4.30	0.54	−0.06
infl	Inflation	1913-02	0.74	4.81	0.53	0.76	4.80	0.55	−0.02
svar	Stock variance	1885-02	0.57	5.10	0.39	0.63	5.09	0.43	−0.04
d/e	Dividend payout ratio	1871-01	0.53	4.95	0.37	0.61	4.94	0.43	−0.05
lty	Long term yield	1919-01	0.64	4.31	0.51	0.68	4.30	0.54	−0.03
tms	Term spread	1920-01	0.72	4.23	0.59	0.68	4.24	0.55	0.04
tbl	Treasury-bill rate	1920-01	0.65	4.24	0.53	0.68	4.24	0.55	−0.03
dfr	Default return spread	1926-01	0.65	4.19	0.53	0.64	4.19	0.53	0.01
d/p	Dividend price ratio	1871-01	0.54	4.95	0.38	0.61	4.94	0.43	−0.05
d/y	Dividend yield	1871-02	0.37	4.97	0.26	0.61	4.94	0.43	−0.17
ltr	Long term return	1926-01	0.65	4.19	0.54	0.64	4.19	0.53	0.01
e/p	Earning price ratio	1871-01	0.55	4.95	0.39	0.61	4.94	0.43	−0.04
ntis	Net equity expansion	1926-12	0.66	4.17	0.55	0.66	4.17	0.55	0.00
all	Kitchen sink	1926-12	0.29	4.22	0.23	0.66	4.17	0.55	−0.31

Table 8: Statistics of the monthly portfolios using linear regression. Explanation: This table shows the statistics of the portfolio using the signals for market-timing and the benchmark portfolio which is long-only. The $\bar{r}$ and σ are calculated using the monthly returns. The annual SR is then derived using the method explained in section 4.3. Subtracting the $SR_{Portfolio}$ from the $SR_{Benchmark}$ yields the ΔSR. A positive value indicates better performance of the portfolio using the specified variable. The column Start shows the first month the data availability for the variable start. The first prediction is then made after 20 years.

			Portfolio			**Benchmark**			
	Variable	**Start**	$\bar{\mathbf{r}}$	σ	**SR**	$\bar{\mathbf{r}}$	σ	**SR**	**ΔSR**
dfy	Default yield spread	1919-01	0.68	4.30	0.54	0.68	4.30	0.54	0.00
infl	Inflation	1913-02	0.74	4.81	0.54	0.76	4.80	0.55	−0.01
svar	Stock variance	1885-02	0.63	5.09	0.43	0.63	5.09	0.43	0.00
d/e	Dividend payout ratio	1871-01	0.57	4.95	0.40	0.61	4.94	0.43	−0.03
lty	Long term yield	1919-01	0.65	4.31	0.52	0.68	4.30	0.54	−0.03
tms	Term spread	1920-01	0.73	4.23	0.60	0.68	4.24	0.55	0.04
tbl	Treasury-bill rate	1920-01	0.65	4.24	0.53	0.68	4.24	0.55	−0.02
dfr	Default return spread	1926-01	0.61	4.19	0.50	0.64	4.19	0.53	−0.03
d/p	Dividend price ratio	1871-01	0.57	4.95	0.40	0.61	4.94	0.43	−0.03
d/y	Dividend yield	1871-02	0.58	4.95	0.41	0.61	4.94	0.43	−0.02
ltr	Long term return	1926-01	0.62	4.19	0.51	0.64	4.19	0.53	−0.02
e/p	Earning price ratio	1871-01	0.58	4.95	0.40	0.61	4.94	0.43	−0.03
ntis	Net equity expansion	1926-12	0.66	4.17	0.55	0.66	4.17	0.55	0.00
all	Kitchen sink	1926-12	0.68	4.17	0.56	0.66	4.17	0.55	0.01

Table 9: Statistics of the monthly portfolios using elastic net. For explanation, please refer to Table 8.

			Portfolio			**Benchmark**			
	Variable	**Start**	$\bar{\mathbf{r}}$	σ	**SR**	$\bar{\mathbf{r}}$	σ	**SR**	**ΔSR**
dfy	Default yield spread	1919-01	0.50	4.33	0.40	0.68	4.30	0.54	−0.14
infl	Inflation	1913-02	0.69	5.67	0.42	0.76	4.80	0.55	−0.12
svar	Stock variance	1885-02	0.29	5.12	0.19	0.63	5.09	0.43	−0.23
d/e	Dividend payout ratio	1871-01	0.27	4.98	0.19	0.61	4.94	0.43	−0.24
lty	Long term yield	1919-01	0.21	4.34	0.17	0.68	4.30	0.54	−0.37
tms	Term spread	1920-01	0.34	4.28	0.27	0.68	4.24	0.55	−0.28
tbl	Treasury-bill rate	1920-01	0.52	4.28	0.42	0.68	4.24	0.55	−0.14
dfr	Default return spread	1926-01	0.10	4.24	0.08	0.64	4.19	0.53	−0.45
d/p	Dividend price ratio	1871-01	0.23	4.88	0.16	0.61	4.94	0.43	−0.27
d/y	Dividend yield	1871-02	0.12	5.08	0.08	0.61	4.94	0.43	−0.35
ltr	Long term return	1926-01	0.40	4.21	0.33	0.64	4.19	0.53	−0.20
e/p	Earning price ratio	1871-01	0.17	4.98	0.12	0.61	4.94	0.43	−0.31
ntis	Net equity expansion	1926-12	0.16	4.22	0.13	0.66	4.17	0.55	−0.41
all	Kitchen sink	1926-12	0.65	4.18	0.54	0.66	4.17	0.55	−0.01

Table 10: Statistics of the monthly portfolios using k-nearest neighbors. For explanation, please refer to Table 8.

When using signals generated via decision trees, the results look mostly positive, as can be seen in Table 11. The model using the Default yield spread (**dfy**) generates only positive signals, which is why it generates the same return and Sharpe ratio as the benchmark. The Kitchen sink model (**all**), on the other hand, deviates only briefly and, with rounding to two decimal places, produces a ΔSR of zero. Some models deviate only slightly positively from zero, but two models appear to achieve a Sharpe ratio that outperforms the benchmark by a wider margin. The models are the Stock variance (**svar**) and the Long term return (**lty**). However, neither model was able to achieve a positive adjusted $\bar{R}^2_{OOS}$ value as seen in Table 5. The Dividend yield model (**dy**), on the other hand, generated a positive adjusted $\bar{R}^2_{OOS}$, but was unable to outperform the benchmark in terms of Sharpe ratio. Again, predictive performance does not seem to be the appropriate measure for selecting models and methods, nor for testing whether a method is reliable when it comes to market timing.

			Portfolio			**Benchmark**			
	Variable	**Start**	**r̄**	σ	**SR**	**r̄**	σ	**SR**	**ΔSR**
dfy	Default yield spread	1919-01	0.68	4.30	0.54	0.68	4.30	0.54	0.00
infl	Inflation	1913-02	0.74	4.81	0.53	0.76	4.80	0.55	−0.01
svar	Stock variance	1885-02	0.79	5.07	0.54	0.63	5.09	0.43	0.11
d/e	Dividend payout ratio	1871-01	0.63	4.94	0.44	0.61	4.94	0.43	0.01
lty	Long term yield	1919-01	0.55	4.32	0.44	0.68	4.30	0.54	−0.10
tms	Term spread	1920-01	0.70	4.23	0.57	0.68	4.24	0.55	0.02
tbl	Treasury-bill rate	1920-01	0.69	4.24	0.57	0.68	4.24	0.55	0.01
dfr	Default return spread	1926-01	0.62	4.19	0.52	0.64	4.19	0.53	−0.01
d/p	Dividend price ratio	1871-01	0.55	4.95	0.39	0.61	4.94	0.43	−0.04
d/y	Dividend yield	1871-02	0.58	4.96	0.40	0.61	4.94	0.43	−0.02
ltr	Long term return	1926-01	0.72	4.17	0.60	0.64	4.19	0.53	0.07
e/p	Earning price ratio	1871-01	0.52	4.95	0.36	0.61	4.94	0.43	−0.06
ntis	Net equity expansion	1926-12	0.65	4.17	0.54	0.66	4.17	0.55	−0.01
all	Kitchen sink	1926-12	0.65	4.17	0.54	0.66	4.17	0.55	0.00

Table 11: Statistics of the monthly portfolios using decision trees. For explanation, please refer to Table 8.

Comparing the results of the different methods which are based on decision trees, we see that there is no consistent pattern compared to Table 6. Moreover, no model is able to outperform the benchmark.

				Portfolio			**Benchmark**			
Method	**Variable**		**Start**	**r̄**	σ	**SR**	**r̄**	σ	**SR**	**ΔSR**
Trees	**all**	Kitchen sink	1926-12	0.65	4.17	0.54	0.66	4.17	0.55	0.00
Forest	**all**	Kitchen sink	1926-12	0.66	4.17	0.55	0.66	4.17	0.55	0.00
GBDT	**all**	Kitchen sink	1926-12	0.59	4.18	0.49	0.66	4.17	0.55	−0.06
LGBM	**all**	Kitchen sink	1926-12	0.64	4.18	0.53	0.66	4.17	0.55	−0.02

Table 12: Statistics of the monthly portfolios using different algorithms based on decision trees. For explanation, please refer to Table 8.

Similar to the predictions, the model using the multi-layer perceptron was not able to outperform the benchmark. Not only the average return $\bar{r}$ and the σ, but also the Sharpe ratio are worse compared to the benchmark. Thus, neither the predictions nor the portfolio can be improved by the MLP.

			Portfolio			**Benchmark**			
	Variable	**Start**	$\bar{\mathbf{r}}$	σ	**SR**	$\bar{\mathbf{r}}$	σ	**SR**	**ΔSR**
all	Kitchen sink	1926-12	0.49	4.20	0.41	0.66	4.17	0.55	−0.14

Table 13: Statistics of the monthly portfolios using a multi-layer perceptron. For explanation, please refer to Table 8.

6 Conclusion

It appears that there is no algorithm that consistently outperforms the benchmark using this specific set of independent variables. Other papers such as Gu et al. (2020) may have found methods, and especially hyperparameters, that yield a stable model for predicting the equity risk premium. However, they typically use an extended set of independent variables. With the inclusion of more than 100 variables in conjunction with variable selection, it seems logical to find a combination that works for a given data set. In addition, a shorter time period is usually used, such as 20 years, and the variables are trained on a fixed set of years. In this way, it is much easier to find patterns in this fixed time period. In contrast to this work, a data set of over 100 years is used, and predictions are made every year. This was tested with both an expanding window and a rolling window, with the expanding window performing better than the rolling window in almost all cases. This procedure should circumvent any snooping bias that may occur when using a single fixed date range. However, if one tweaks the models long enough, it may still be possible to find a hidden gem in terms of method and hyperparameters. However, this is difficult to justify empirically.

As mentioned earlier, the papers that get good results with MLP, for example, use a different set of independent variables. These include mean reversion, moving averages, and momentum variables. In this work, we have examined specific economic variables that are already widely discussed in the academic literature. So either these methods don't work with this kind of predictors or the correct set of hyperparameters have not been found. Another explanation could be that the positive results so far are due to data snooping. One solution would be, that a specific method for a specific set of predictor variables are defined academically. If there was a longer period of economic data, it would also be much easier to look for and find patterns. Logically, it is also very likely that there are still patterns and crises which are unique and have not yet occurred. So if there was a perfect application of an algorithm and the data, it would probably look completely different in a few years.

Another question is the actual application in the financial industry. When using these predictions as signals for market timing, there seems to be very rarely a match between the prediction performance and the resulting portfolio using the signals. However, this could be improved if the method predicts categories rather than numerical values, in this case "long and short" or "up and down", using classification rather than regression methods. The cases where only the portfolio is positive but their respective adjusted $\bar{R}^2_{OOS}$

is negative seem to be a result of lucky coincidences where a bad numerical prediction leads to the correct directional prediction. Or rather, there is no empirical explanation for it.

Thus, this paper might answer some questions about the prediction of equity premia, but it also raises some further open questions. For example, exactly what steps need to be taken to prevent data-snooping or what method can be used for what type of data. It could also be that some of the variables used in this paper are not suitable for predictive purposes. So, again, the task is to find the best variables that have the greatest predictive power. Hopefully, this work has contributed a helpful part in finding some of these answers or raising new questions.

7 References

Bachelier, L.; 1900; ”Théorie de la spéculation”; Annales Scientifiques de l’École Normale Supérieure, Vol. 3, No. 17, pp. 21–86; http://archive.numdam.org/item/ASENS_1900_3_17__21_0/

Campbell, John Y.; 1987; ”Stock returns and the term structure”; Journal of Financial Economics; Vol. 18, No. 2; pp. 373-399; https://www.sciencedirect.com/science/article/abs/pii/0304405X87900456

Campbell, J., Thompson, S.; 2008; ”Predicting Excess Stock Returns out of Sample: Can Anything Beat the Historical Average?”; The Review of Financial Studies, Vol. 21, No. 4, pp. 1509-1531; https://www.jstor.org/stable/40056860

Cochrane, John H.; 1991; ”Production-Based Asset Pricing and the Link Between Stock Returns and Economic Fluctuations.” The Journal of Finance, Vol. 46, No. 1, pp. 209–37.; https://www.jstor.org/stable/2328694

Cochrane, John H.; 2008; “The Dog That Did Not Bark: A Defense of Return Predictability.”; The Review of Financial Studies; Vol 21; No. 4; pp. 1533–1575; http://www.jstor.org/stable/40056861

Dow C. H.; 1920; ”Scientific Stock Speculation”; The Magazine of Wall Street; https://openlibrary.org/books/OL14818632M/Scientific_stock_speculation

Fama E., French K.; 1988; ”Dividend yields and expected stock returns”; Journal of Financial Economics; Vol. 22; Issue 1; pp. 3-25; https://www.sciencedirect.com/science/article/abs/pii/0304405X88900207

Fama E., French K.; 2014; ”A five-factor asset pricing model”; Social Science research Network; http://papers.ssrn.com/sol3/papers.cfm?abstract_id=2287202

Fix E., Hodges J. L.; 1989 (first appeared in 1951); ”Discriminatory Analysis. Nonparametric Discrimination: Consistency Properties.” International Statistical Review/Revue Internationale de Statistique; 57(3); 238–247; https://www.jstor.org/stable/1403797

Green, J., J. R. M. Hand, and X. F. Zhang; 2013; ”The supraview of return predictive signals”; Review of Accounting Studies; Vol. 18; pp. 692–730; https://papers.ssrn.com/sol3/papers.cfm?abstract_id=2062464

Gu, S., Kelly, B., Xiu, D.; 2020; ”Empirical Asset Pricing via Machine Learning”; The Review of Financial Studies, Vol. 33, Issue 5, pp. 2223–2273; https://academic.oup.com/rfs/article/33/5/2223/5758276?login=true

Hoerl, A. E., Kennard, R. W.; 1970; ”Ridge Regression: Biased Estimation for Nonorthogonal Problems”; Technometrics, Vol. 12; No. 1, pp. 55–67; https://www.jstor.org/stable/1267351

Hoerl, A. E., Kennard, R. W.; 1970; "Ridge Regression: Applications to Nonorthogonal Problems"; Technometrics, Vol. 12; No. 1, pp. 69–82; https://www.jstor.org/stable/1267352

Jacobsen, B., Jiang, F., Zhang, H.; 2020; "Equity Premium Prediction with Bagged Machine Learning"; https://papers.ssrn.com/sol3/papers.cfm?abstract_id=3310289

Lettau, M., Ludvigson, S.; 2001; "Consumption, Aggregate Wealth, and Expected Stock Returns"; The Journal of Finance, Vol. 56, No. 3, pp. 815–849. http://www.jstor.org/stable/222534

Lo, A.W., MacKinlay, A.C.; 1990; "Data-Snooping Biases in Tests of Financial Asset Pricing Models"; The Review of Financial Studies, Vol. 3, No. 3, pp. 431-467; https://www.jstor.org/stable/2962077

Markowitz, Harry; 1952; "Portfolio Selection." The Journal of Finance; Vol 7, No. 1 (1952); pp. 77–91; https://www.jstor.org/stable/2975974

Pesaran, M.H.; Timmermann, A.; 1995; "Predictability of Stock Returns: Robustness and Economic Significance"; The Journal of Finance; Vol. 50, No. 4 , pp. 1201-1228; https://www.jstor.org/stable/2329349

Sharpe, William F.; 1994; "The Sharpe Ratio"; Journal of Portfolio Management; Vol. 21; pp. 49-58.; https://jpm.pm-research.com/content/21/1/49

Shiller, R. J.; 1981; "Do Stock Prices Move Too Much to Be Justified by Subsequent Changes in Dividends?"; American Economic Review; Vol. 71; pp. 421-36; https://www.jstor.org/stable/1802789

Welch, I., Goyal, A.; 2008; "A Comprehensive Look at the Empirical Performance of Equity Premium Prediction"; The Review of Financial Studies, Vol. 21, No. 4, pp. 1455-1508; https://www.jstor.org/stable/40056859

Zou, H., Hastie, T.; 2005; "Regularization and Variable Selection via the Elastic Net"; Journal of the Royal Statistical Society, Series B (Statistical Methodology); Vol. 67; No. 2; pp. 301–320.; https://www.jstor.org/stable/3647580

Data Sources from Websites:

Bureau of Labor Statistics Webpage - CPI; http://www.bls.gov/cpi/

FRED Economic Indicators - Data Library; https://fred.stlouisfed.org/

Ivo Welch - Data for Welch & Goyal (2008) updated up until 2020; https://www.ivo-welch.info/professional/goyal-welch/

NBER Macrohistory: Interest Rates; http://www.nber.org/databases/macrohistory/contents/chapter13.html

Robert Shiller's Data; http://www.econ.yale.edu/~shiller/data.htm

8 Appendix

```python
def rolling(regr, x, y, window):

    # Variables
    p        = len(list(x))
    date     = x.index
    title    = x.columns if (p == 1) else ['all']
    y        = y[x.index][1:].to_numpy()
    x        = x[:-1].to_numpy()
    n        = len(x[window:])
    y_act    = y[window:]
    avg_rol  = pd.DataFrame(y).rolling(window=window).mean().dropna()[0].to_numpy()[:-1]

    # IS Calculations
    regr.fit(x[window:], y[window:])
    err_is_a = (y_act - regr.predict(x[window:]))**2
    err_is_n = (y_act - y_act.mean())**2
    sse_delta_is = np.cumsum(err_is_n - err_is_a)
    r_2_is   = regr.score(x[window:], y[window:])
    ar_2_is  = 1-(1-r_2_is)*(len(x[window:])-1)/(len(x[window:])-p-1)

    # OOS Rolling calculations
    y_pred = []
    for i in range(n):
        x_train_ = x[i:i+window]
        y_train_ = y[i:i+window]
        x_test_  = x[i+window]
        regr.fit(x_train_, y_train_)
        y_pred_  = regr.predict(np.array(x_test_).reshape(-1,p))
        y_pred.append(y_pred_[0])

    # Stats
    err_oos_a     = (y_act - y_pred)**2
    err_oos_n     = (y_act - avg_rol)**2
    sse_delta_oos = np.cumsum(err_oos_n - err_oos_a)
    mse_a         = err_oos_a.mean()
    mse_n         = err_oos_n.mean()
    r_2_oos       = (1-mse_a/mse_n)
    ar_2_oos      = 1-(1-r_2_oos)*(n-1)/(n-p-1)
    drmse_oos     = math.sqrt(mse_n)-math.sqrt(mse_a)

    # Long-short portfolio
    ls_ret   = []
    for yi in y_pred:
        if yi > 0:
            ls_ret.append( 1 * y_act[np.where(y_pred == yi)][0] + 1)
        elif yi == 0:
            ls_ret.append(0)
        elif yi < 0:
            ls_ret.append(-1 * y_act[np.where(y_pred == yi)][0] + 1)
    ls_cumsum  = np.cumsum(np.log(ls_ret))
    erp_cumsum = np.cumsum(np.log(y_act + 1))

    # Data preparation
    df_output     = pd.DataFrame({'y_pred': y_pred, 'y_act': y_act, 'err_oos_a': err_oos_a, 'err_oos_n': err_oos_n, 'sse_delta_oos': sse_delta_oos, 'sse_delta_is': sse_delta_is, 'avg_rol': avg_rol, 'ls_ret': ls_ret, 'ls_cumsum': ls_cumsum}, index = date[window + 1:])

    df_statistics = pd.DataFrame({'R^2_IS': r_2_is, 'adj_R^2_IS': ar_2_is, 'R^2_OOS': r_2_oos, 'adj_R^2_OOS': ar_2_oos, 'dRMSE_OOS': drmse_oos}, index = title)\
    .multiply(100)

    df_portfolio  = pd.DataFrame({'ls_cumsum': ls_cumsum, 'erp_cumsum': erp_cumsum}, index = date[window + 1:])

    df_plot       = pd.DataFrame({'sse_delta_oos': sse_delta_oos, 'sse_delta_is': sse_delta_is}, index = date[window + 1:])

    dict_output = {
    'Output'     : df_output,
    'Statistics' : df_statistics,
    'Portfolio'  : df_portfolio,
    'Plot'       : df_plot
    }

    return(dict_output)
```

Listing 1: Rolling window - Python code

```python
def expanding(regr, x, y, window):

    # Variables
    erp_wg    = y
    p         = len(list(x))
    date      = x.index
    title     = x.columns if (p == 1) else ['all']
    y         = y[x.index][1:].to_numpy()
    x         = x[:-1].to_numpy()
    n         = len(x[window:])
    y_act     = y[window:]
    avg_rol   = pd.DataFrame(y).expanding(min_periods=window).mean().dropna()[0].to_numpy()[:-1]

    # IS Calculations
    regr.fit(x[window:], y[window:])
    err_is_a = (y_act - regr.predict(x[window:]))**2
    err_is_n = (y_act - y_act.mean())**2
    sse_delta_is = np.cumsum(err_is_n - err_is_a)
    r_2_is    = regr.score(x[window:], y[window:])
    ar_2_is  = 1-(1-r_2_is)*(len(x[window:])-1)/(len(x[window:])-p-1)

    # OOS Rolling calculations
    y_pred = []
    for i in range(n):
        x_train_ = x[:i+window]
        y_train_ = y[:i+window]
        x_test_  = x[i+window]
        regr.fit(x_train_, y_train_)
        y_pred_  = regr.predict(np.array(x_test_).reshape(-1,p))
        y_pred.append(y_pred_[0])

    # Stats
    err_oos_a     = (y_act - y_pred)**2
    err_oos_n     = (y_act - avg_rol)**2
    sse_delta_oos = np.cumsum(err_oos_n - err_oos_a)
    mse_a         = err_oos_a.mean()
    mse_n         = err_oos_n.mean()
    r_2_oos       = (1-mse_a/mse_n)
    ar_2_oos      = 1-(1-r_2_oos)*(n-1)/(n-p-1)
    drmse_oos     = math.sqrt(mse_n)-math.sqrt(mse_a)

    # Long-short portfolio
    ls_ret   = []
    for yi in y_pred:
        if yi > 0:
            ls_ret.append( 1 * y_act[np.where(y_pred == yi)][0])
        elif yi == 0:
            ls_ret.append(0)
        elif yi < 0:
            ls_ret.append(-1 * y_act[np.where(y_pred == yi)][0])
    ls_cumsum  = np.cumsum(np.log(np.asarray(ls_ret)+1), axis = 0)
    erp_cumsum = np.cumsum(np.log(y_act + 1), axis = 0)
    bmk_ret_47 = pd.DataFrame(erp_wg["1947":])
    ls_ret_47  = pd.DataFrame({'ls_ret': ls_ret[-len(bmk_ret_47):]}, index = bmk_ret_47.index)
    df_ls_47   = bmk_ret_47.merge(ls_ret_47, left_on='Date', right_on='Date')

    # Long-short statistics
    ls_avg     = pd.DataFrame(ls_ret).mean()[0]*100
    ls_std     = pd.DataFrame(ls_ret).std()[0]*100
    ls_sr      = math.sqrt(12) * ls_avg/ls_std
    y_avg      = pd.DataFrame(y_act).mean()[0]*100
    y_std      = pd.DataFrame(y_act).std()[0]*100
    y_sr       = math.sqrt(12) * y_avg/y_std
    delta      = ls_sr-y_sr

    # Data preparation
    df_output       = pd.DataFrame({'y_pred': y_pred, 'y_act': y_act, 'err_oos_a': err_oos_a,'err_oos_n': err_oos_n, 'sse_delta_oos': sse_delta_oos, 'sse_delta_is': sse_delta_is, 'avg_rol': avg_rol, 'ls_ret': ls_ret, 'ls_cumsum': ls_cumsum}, index = date[window + 1:])

    df_statistics   = pd.DataFrame({'R^2_IS': r_2_is, 'adj_R^2_IS': ar_2_is, 'R^2_OOS': r_2_oos, 'adj_R^2_OOS': ar_2_oos, 'dRMSE_OOS': drmse_oos}, index = title)\
    .multiply(100)

    df_portfolio    = pd.DataFrame({'ls_cumsum': ls_cumsum, 'erp_cumsum': erp_cumsum}, index = date[window + 1:])

    df_portfolio_47 = pd.DataFrame({'ls_cumsum': np.cumsum(np.log(ls_ret_47+1))['ls_ret'], 'erp_cumsum': np.cumsum(np.log(bmk_ret_47+1))['erp_wg']}, index = date[window + 1:])

    df_p_stats      = pd.DataFrame({'ls_avg': ls_avg, 'ls_std': ls_std, 'ls_sr': ls_sr, 'y_avg': y_avg, 'y_std': y_std, 'y_sr': y_sr, 'delta': delta}, index = title)

    df_plot         = pd.DataFrame({'sse_delta_oos': sse_delta_oos, 'sse_delta_is': sse_delta_is}, index = date[window + 1:])

    dict_output = {
    'Output'       : df_output,
    'Statistics'   : df_statistics,
    'Portfolio'    : df_portfolio,
    'Portfolio_47' : df_portfolio_47,
    'P_Stats'      : df_p_stats,
    'Plot'         : df_plot
    }

    return(dict_output)
```

Listing 2: Expanding window - Python code

```python
import numpy as np
import pandas as pd
from sklearn import tree
from sklearn.neighbors import KNeighborsRegressor
from sklearn.preprocessing import StandardScaler, Normalizer,
    PowerTransformer, MinMaxScaler
from sklearn.pipeline import make_pipeline
from sklearn.linear_model import ElasticNet, ElasticNetCV, LinearRegression
from sklearn.neural_network import MLPRegressor
from sklearn.ensemble import AdaBoostRegressor, RandomForestRegressor,
    VotingRegressor, GradientBoostingRegressor,
    HistGradientBoostingRegressor
from sklearn.tree import DecisionTreeRegressor

# Linear
lin_reg     = LinearRegression()

# Elastic Net
eln         = ElasticNet(alpha=0.001, l1_ratio=0.5)
eln_cv      = ElasticNetCV(cv=5, random_state=0)

# Nearest Neighbors
knn         = KNeighborsRegressor(n_neighbors=25, weights='uniform',
    algorithm='auto') ## many n_n to avoid overfitting
knn_ss      = make_pipeline(StandardScaler(), knn)

# Trees
tree        = DecisionTreeRegressor(max_depth=1, max_leaf_nodes=4)

# Forest
forest      = RandomForestRegressor(n_estimators=10, max_depth=1,
    max_leaf_nodes=4)

# Boosted Trees
gbm         = GradientBoostingRegressor(n_estimators=10, max_depth=1,
    max_leaf_nodes=4)
lgbm        = HistGradientBoostingRegressor(max_iter=10, max_depth=1,
    max_leaf_nodes=4)

# Neural Networks
mlp         = MLPRegressor(hidden_layer_sizes=(14,56,56,28,14))
mlp_nn      = make_pipeline(Normalizer(), mlp)
```

Listing 3: Methods - Python code

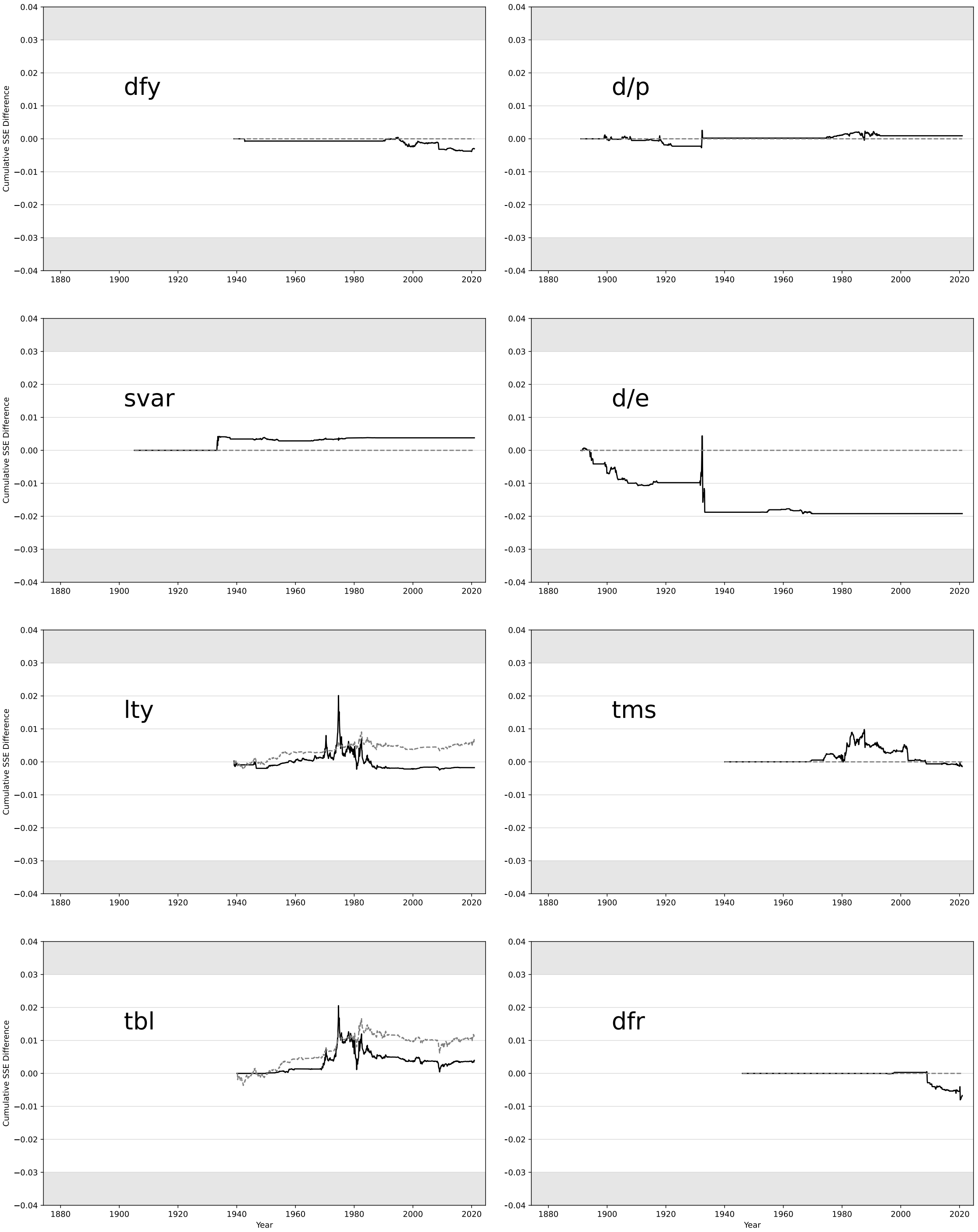


Figure 14: Monthly predictive performance using elastic net regression

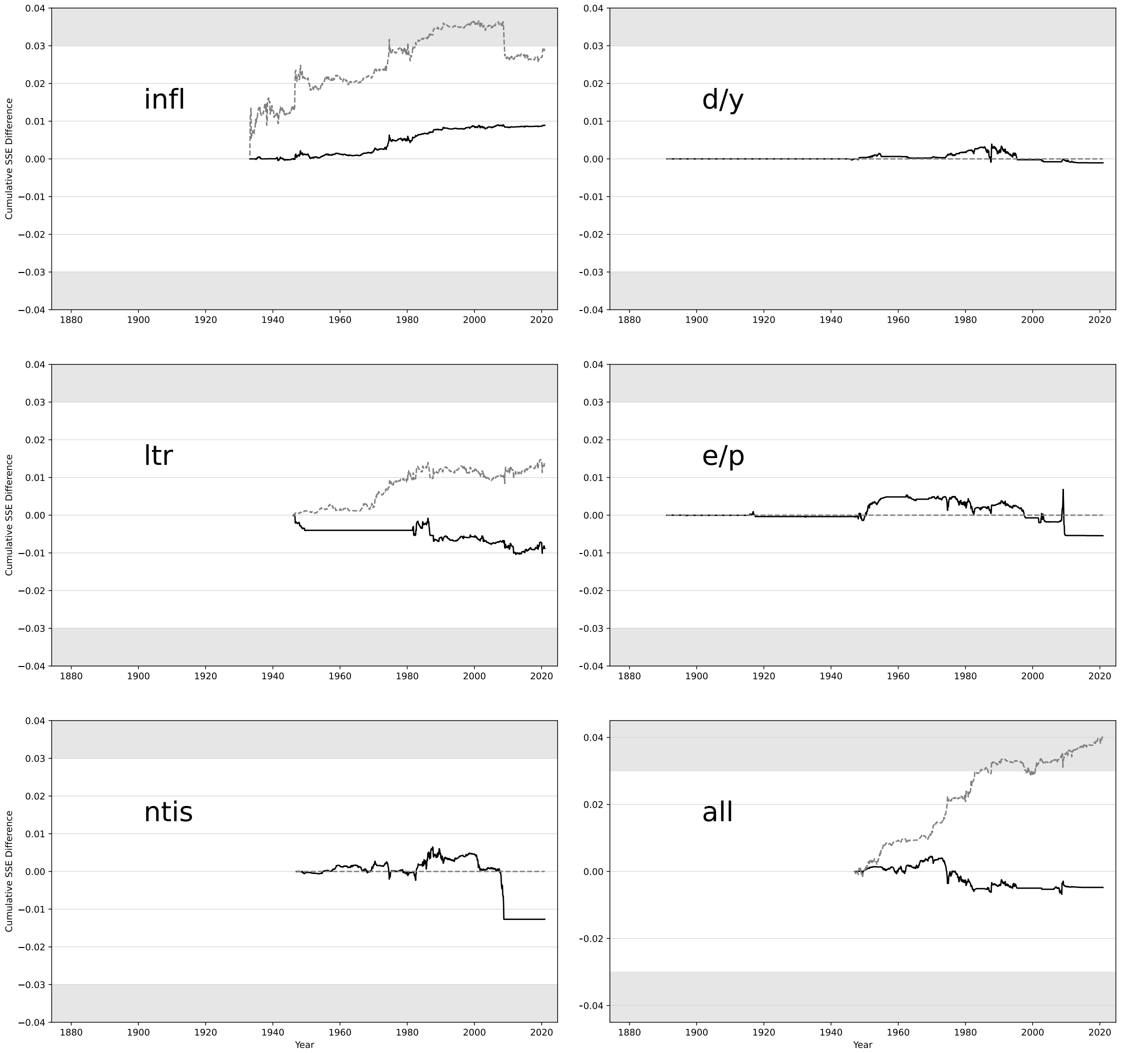


Figure 14 (cont.): Monthly predictive performance using elastic net regression

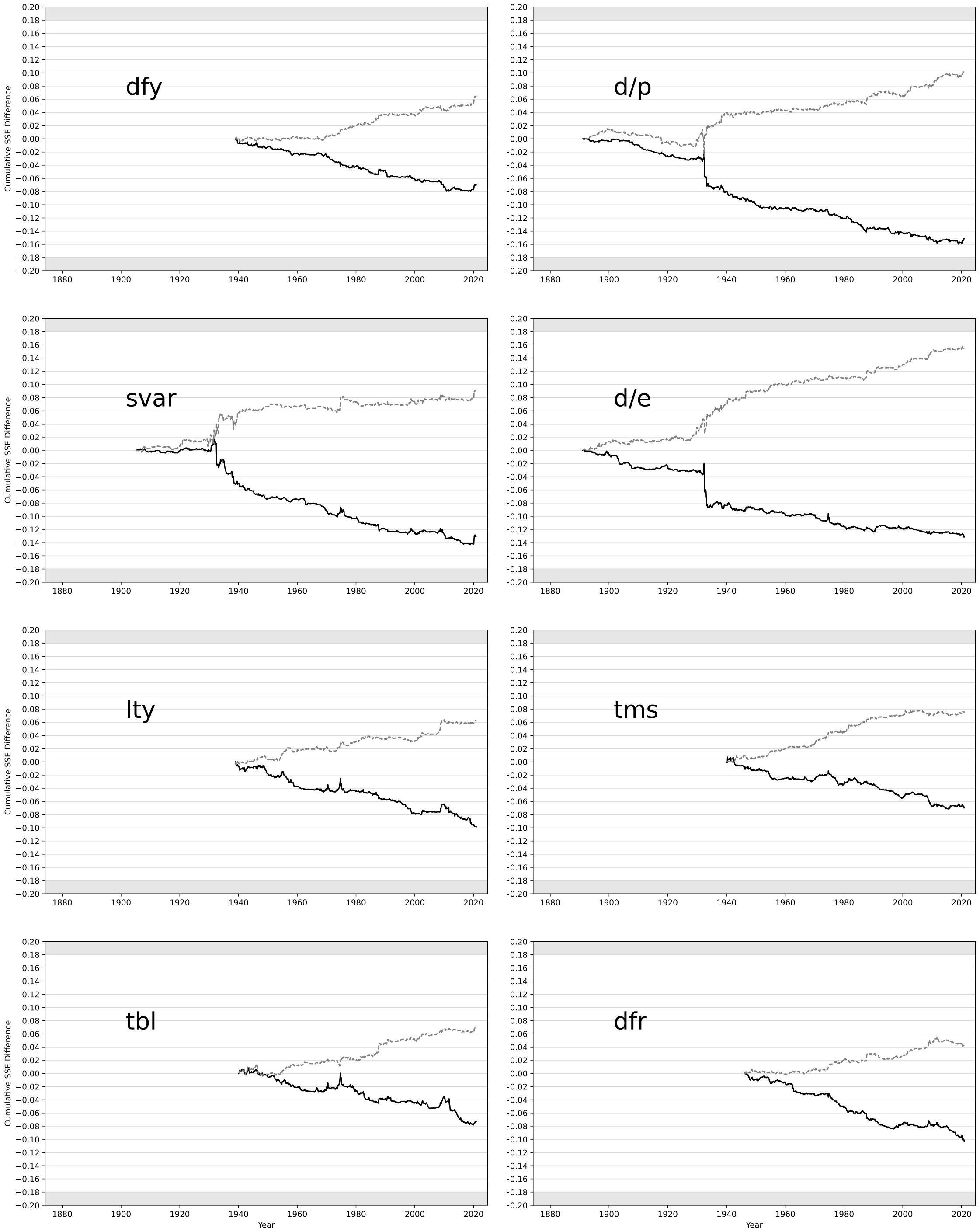


Figure 15: Monthly predictive performance using k-nearest neighbors

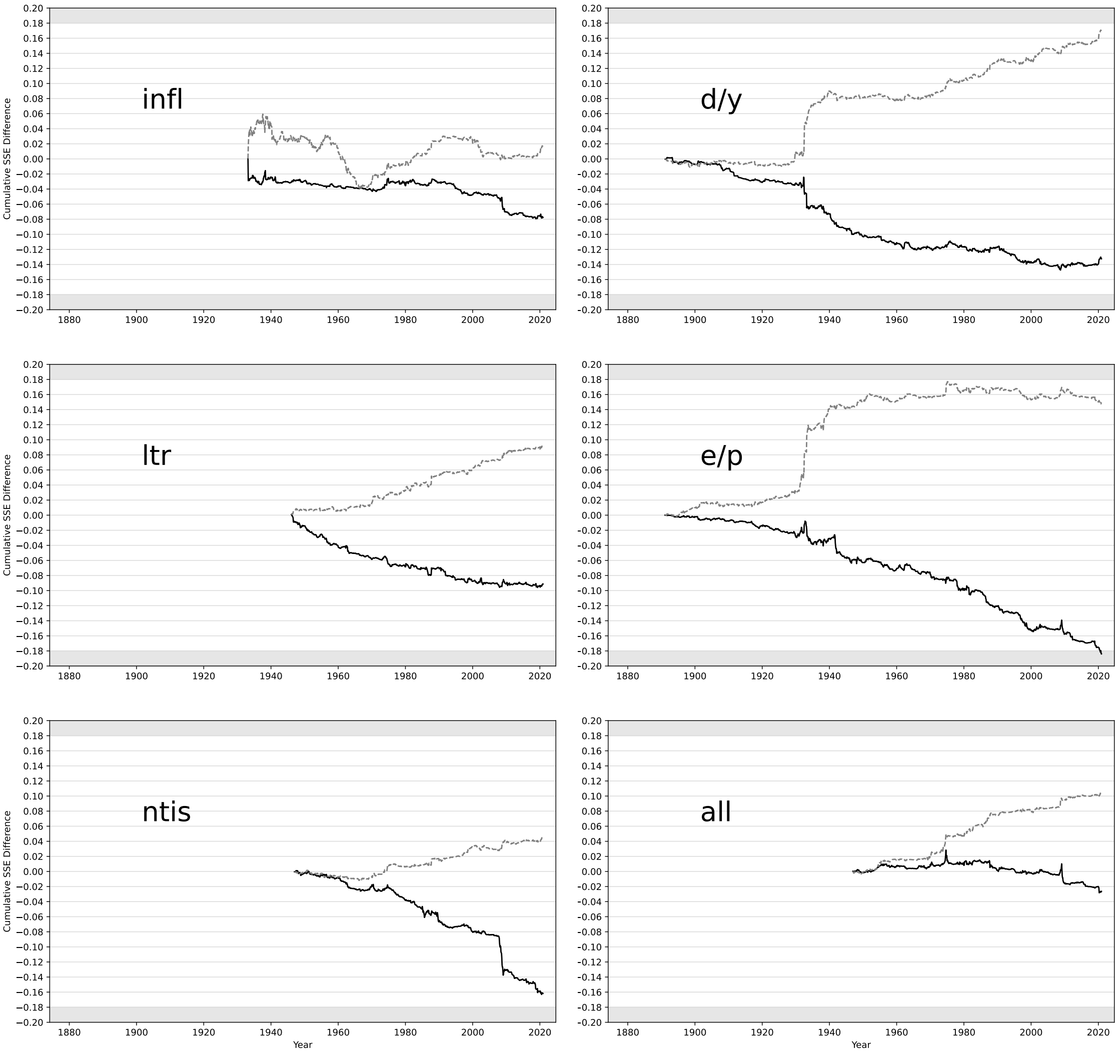


Figure 15 (cont.): Monthly predictive performance using k-nearest neighbors

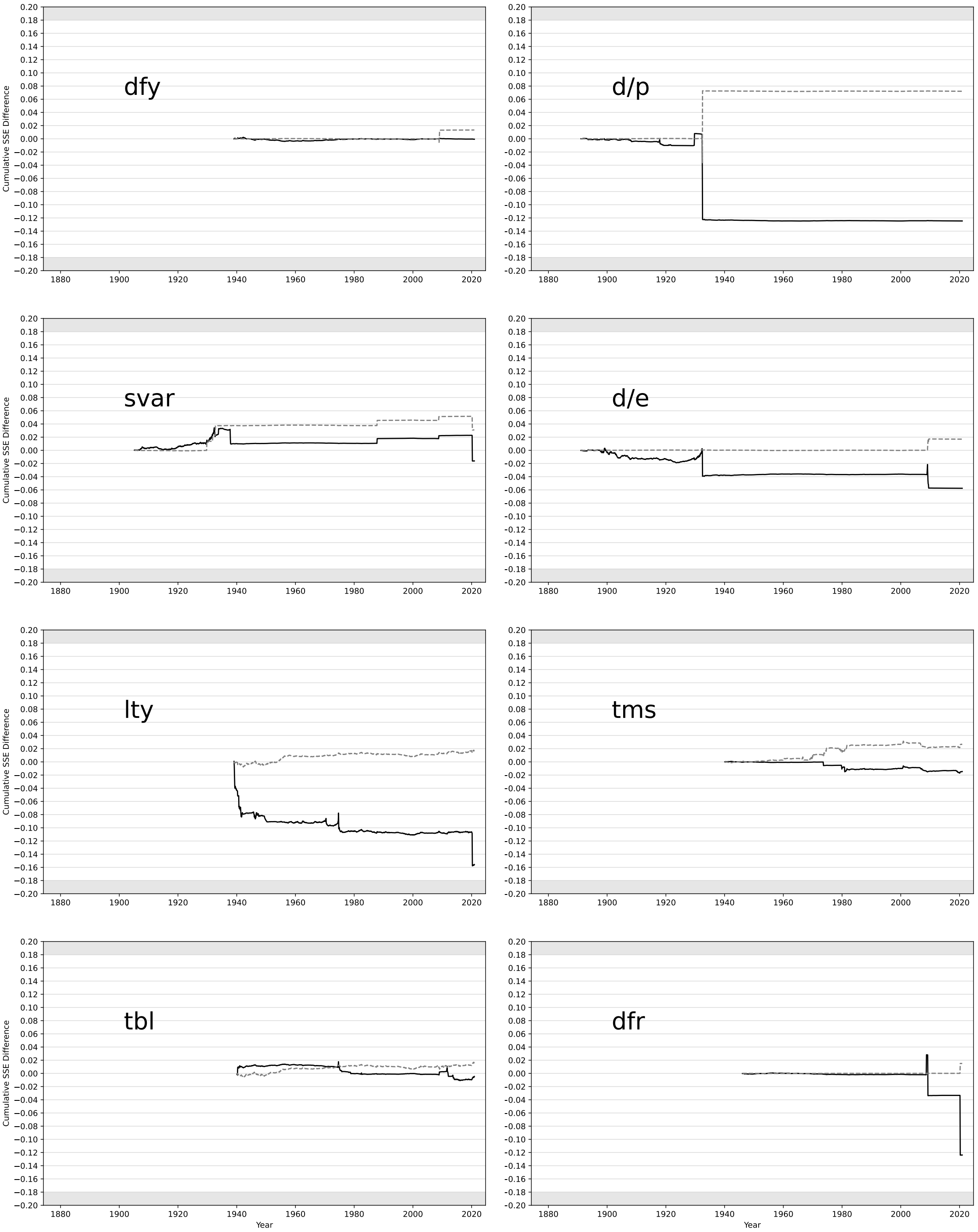


Figure 16: Monthly predictive performance using decision trees

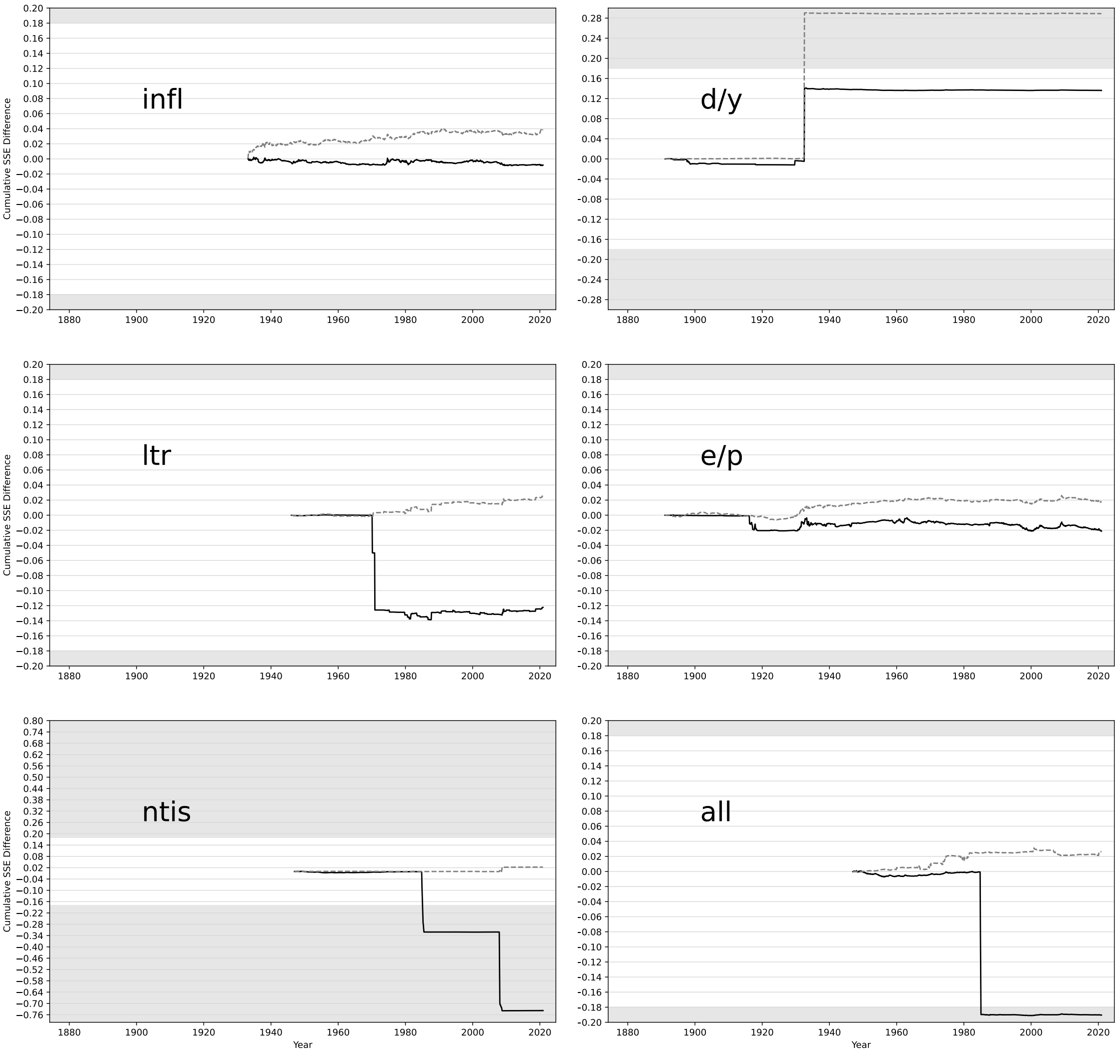


Figure 16 (cont.): Monthly predictive performance using decision trees