UNIVERSITY OF TURKU

ESSAYS ON ECONOMIC FORECASTING USING MACHINE LEARNING

Lauri Nevasalmi

TURUN YLIOPISTON JULKAISUJA – ANNALES UNIVERSITATIS TURKUENSIS
SARJA – SER. E OSA – TOM. 67 | OECONOMICA | TURKU 2020

UNIVERSITY OF TURKU

ESSAYS ON ECONOMIC FORECASTING USING MACHINE LEARNING

Lauri Nevasalmi

TURUN YLIOPISTON JULKAISUJA – ANNALES UNIVERSITATIS TURKUENSIS
SARJA – SER. E OSA – TOM. 67 | OECONOMICA | TURKU 2020

University of Turku

Turku School of Economics
Department of Economics
Economics
Doctoral programme of Turku School of Economics

Supervised by

Professor Heikki Kauppi
University of Turku
Turku, Finland

Professor Matti Viren
University of Turku
Turku, Finland

Reviewed by

Professor Charlotte Christiansen
Aarhus University
Aarhus, Denmark

Professor Seppo Pynnönen
University of Vaasa
Vaasa, Finland

Opponent

Professor Charlotte Christiansen
Aarhus University
Aarhus, Denmark

ISBN 978-951-29-8222-6 (PRINT)
ISBN 978-951-29-8223-3 (PDF)
ISSN 2343-3159 (Painettu/Print)
ISSN 2343-3167 (Verkkojulkaisu/Online)
Painosalama, Turku, Finland 2020

UNIVERSITY OF TURKU
Turku School of Economics
Department of Economics
Economics
LAURI NEVASALMI: Essays on economic forecasting using machine learning
Doctoral Dissertation, 163 pp.
Doctoral Programme of Turku School of Economics
November 2020

ABSTRACT

This thesis studies the additional value introduced by different machine learning methods to economic forecasting. Flexible machine learning methods can discover various complex relationships in data and are well-suited for analysing so called big data and potential problems therein. Several new extensions to existing machine learning methods are proposed from the viewpoint of economic forecasting.

In Chapter 2, the main objective is to predict U.S. economic recession periods with a high-dimensional dataset. A cost-sensitive extension to the gradient boosting machine learning algorithm is proposed, which takes into account the scarcity of recession periods. The results show how the cost-sensitive extension outperforms the traditional gradient boosting model and leads to more accurate recession forecasts.

Chapter 3 considers a variety of different machine learning methods when predicting daily returns of the S&P 500 stock market index. A new multinomial approach is suggested, which allows us to focus on predicting the large absolute returns instead of the noisy variation around zero return. In terms of both the statistical and economic evaluation criteria gradient boosting turns out to be the best-performing machine learning method.

In Chapter 4, the asset allocation decisions between risky and risk-free assets are determined using a flexible utility maximization based approach. Instead of the merely considered two-step approach where portfolio weights are based on the excess return predictions obtained with statistical predictive regressions, here the optimal weights are found directly by incorporating a custom objective function to the gradient boosting algorithm. The empirical results using monthly U.S. market returns show that the utility-based approach leads to substantial and quantitatively meaningful economic value over the past approaches.

TURUN YLIOPISTO
Turun Kauppakorkeakoulu
Taloustieteen laitos
Taloustiede
LAURI NEVASALMI: Essays on economic forecasting using machine learning
Väitöskirja, 163 s.
Turun Kauppakorkeakoulun Tohtoriohjelma
Marraskuu 2020

TIIVISTELMÄ

Tässä väitöskirjassa tarkastellaan millaista lisäarvoa koneoppimismenetelmät voivat tuoda taloudellisiin ennustesovelluksiin. Joustavat koneoppimismenetelmät kykenevät mallintamaan monimutkaisia funktiomuotoja ja soveltuvat hyvin big datan eli suurten aineistojen analysointiin. Väitöskirjassa laajennetaan koneoppimismenetelmiä erityisesti taloudellisten ennustesovellusten lähtökohdista katsoen.

Luvussa 2 ennustetaan Yhdysvaltojen talouden taantumajaksoja käyttäen hyvin suurta selittäjäjoukkoa. Gradient boosting -koneoppimismenetelmää laajennetaan huomioimaan aineiston merkittävä tunnuspiirre eli se, että taantumajaksoja esiintyy melko harvoin talouden ollessa suurimman osan ajasta noususuhdanteessa. Tulokset osoittavat, että laajennettu gradient boosting -menetelmä kykenee ennustamaan tulevia taantumakuukausia huomattavasti perinteisiä menetelmiä tarkemmin.

Luvussa 3 hyödynnetään useampaa erilaista koneoppimismenetelmää S&P 500 -osakemarkkinaindeksin päivätuottojen ennustamisessa. Aiemmista lähestymistavoista poiketen tässä tutkimuksessa kategorisoidaan tuotot kolmeen eri luokkaan pyrkimyksenä keskittyä informatiivisempien suurten positiivisten ja negatiivisten tuottojen ennustamiseen. Tulosten perusteella gradient boosting osoittautuu parhaaksi menetelmäksi niin tilastollisten kuin taloudellistenkin ennustekriteerien mukaan.

Luvussa 4 tarkastellaan, kuinka perinteisen tuottoennusteisiin nojautuvan kaksivaiheisen lähestymistavan sijaan allokaatiopäätös riskisen ja riskittömän sijoituskohteen välillä voidaan muodostaa suoraan sijoittajan kokeman hyödyn pohjalta. Hyödyn maksimoinnissa käytetään gradient boosting -menetelmää ja sen mahdollistamaa itsemäärättyä tavoitefunktiota. Yhdysvaltojen aineistoon perustuvat empiiriset tulokset osoittavat kuinka sijoittajan hyötyyn pohjautuva salkkuallokaatio johtaa perinteistä kaksivaiheista lähestymistapaa tuottavampiin allokaatiopäätöksiin.

Acknowledgements

After completing my Master's Thesis the idea of learning more about binary dependent variable models grew inside me. In Finland there are only a few people with an in-depth knowledge on this matter. Soon after starting my career as a doctoral student I found myself working with both of them.

First of all, I would like to thank my thesis supervisor Professor Heikki Kauppi. Thank you for introducing me to the Turku School of Economics and providing me the chance to pursue my goals. Also thank you for the endless amount of flexibility when it comes to meeting arrangements or adapting to new situations in life. We have had many extremely fruitful discussions from which I have learned so much. Who would have thought that your vague idea to give a closer look at one particular paper on machine learning resulted in an entire Doctoral Thesis and an additional Master's degree on the matter?

Next my deepest gratitude goes towards Associate Professor Henri Nyberg. You have always found the time to carefully read and comment my ongoing work. Without your help throughout the years I would not be writing this. Also thank you for constantly challenging me as a young scientist. Whether it was about first public appearance or first submission you always pushed me forward but also provided invaluable help when needed. I am extremely grateful for all the things you have done for me. I can not thank you enough so I sincerely hope the color of Manchester will be red in the near future.

Thank you Susanne and Salla for getting the best out of the hectic first year as a doctoral student. The unforgettable year provided many great memories and hopefully life-long friendships. I wish to thank the pre-examiners of my thesis, Professor Charlotte Christiansen and Professor Seppo Pynnönen, for their insightful comments and suggestions. The long list of people who have given helpful comments on different parts of this thesis receive my utmost appreciation at the beginning of each corresponding chapter. The financial support from the Emil Aaltonen foundation (personal grant and project funding) and the Academy of Finland (grant 321968) is also gratefully acknowledged. Similarly, I am greatly indebted to my parents for the help

during the times when the financial support have felt insufficient.

Finally, I would like to thank my wife Mari. Thank you for always believing in me and being there for me when things seemed helpless. I am forever grateful for being able to reach towards my dreams knowing that our children are having the best of times with you. And no matter what life brings ahead of us after graduation, I could not feel more confident as long as I am with you. Thank you Leevi, Milli and Masi for reminding me how the little things in life matter and always putting a smile on my face. This thesis is dedicated to all of you.

Espoo, October 2020

Lauri Nevasalmi

Contents

Chapter 1

Introduction

1.1 Background

The amount of data has grown exponentially in the previous decade. The ever-increasing flow of information opens up a wide range of completely new possibilities for economic research. Various studies show how web searches or satellite images, for example, can be used to predict economic activity (see e.g., Ettredge, Gerdes and Karuga, 2005; Choi and Varian, 2012; Henderson, Storeygard and Weil, 2012). New data-related issues such as missing data, large amount of potential predictors and mixed data types arise with such 'modern' economic data. Mullainathan and Spiess (2017) emphasize how the term big data reflects the change in both the scale and nature of data. The demand for flexible methods that can handle such datasets and the potential problems therein grows rapidly both in the industry as well as academia.

The popularity of flexible machine learning methods has been increasing in the previous two decades. Machine learning methods are typically introduced to economic forecasting as a tool to exploit the potential non-linearities in the dataset (see e.g., Stock and Watson, 1999; Kuan and Liu, 1995). Other interesting features of machine learning, such as the model selection capabilities of certain methods, have received attention only recently (see e.g., Bühlmann, 2006; Wohlrabe and Buchen, 2014). This thesis puts emphasis on both approaches and studies the additional value introduced by different machine learning methods to economic forecasting. Not just by allowing for more flexible functional forms in the data but also in terms of the ability to deal

with various issues that can be encoutered in economic forecasting problems with big data. These include studying the class imbalance problem with a high-dimensional dataset or even customizing the entire objective function to better meet the needs of a particular forecasting problem.

The class imbalance and the effects on classification is well covered in the machine learning literature (see e.g., Galar et al., 2012). Suprisingly the class imbalance problem has not been properly taken into consideration in previous economic research. That is the case despite the numerous potential economic applications, such as recession forecasting or fraud detection, where the binary response can be quite severely imbalanced. Moreover, most of the standard machine learning algorithms assume balanced class distributions and hence the forecasting performance can be deteriorated in the presence of class imbalance (see He and Garcia, 2009). On the other hand, customizing the entire objective function is connected to the recent discussion by Elliott and Timmermann (2016, Chapter 2) on what is the appropriate objective function in econometric inference. In this thesis we introduce an innovative synthesis between financial economics and machine learning. This is done by utilizing the gradient boosting machine learning algorithm as a tool to optimize a custom objective function motivated by economics and finance.

In an ideal situation the economic forecasting model is based on economic theory and the main analysis concerns specifying the exact functional form for the relationship between the dependent variable and the predictors. Majority of the time however the final model composition (and the functional form) is purely empirical. But how to identify the optimal predictor set? Several potential predictor candidates have been discovered in the economic literature starting from the pioneering work of Mitchell and Burns (1938). Different financial variables have shown predictive power beyond the benchmark autoregressive specifications when forecasting GDP growth, whereas the yield curve is the single best predictor of economic recession periods for example (see e.g., Stock and Watson, 2003; Wheelock and Wohar, 2009). In such a data-rich environment induced by the recent growth in the availability and accessibility of data the amount of potential predictive variables can be very large (see e.g., Bühlmann, 2006). Traditional econometric methods can only handle very limited predictor sets at a time and some sort of model selection procedure is required. Such procedure can easily become computationally

very demanding or even infeasible as the number of predictors grow.

The ability to handle large predictor sets also varies between different machine learning methods. The risk of overlearning the data with large predictor sets is a serious issue for many commonly used machine learning methods. Methods such as the support vector machines or nearest neighbor are known to suffer from the curse of dimensionality (see e.g., Hastie, Tibshirani and Friedman, 2009). Ensemble methods random forest and gradient boosting machine on the other hand contain an attractive feature as model selection is conducted simultaneously with model estimation. These methods require no prior knowledge on the relevance of different predictors and can provide important details about the most informative predictive variables. For a more detailed description of the internal variable selection associated with these two methods, see Breiman (2001) and Friedman (2001).

Unlike in various business-oriented applications the computational efficiency of the forecasting method does not play a significant role as the amount of observations in economic datasets is typically quite limited. Economic forecasting however places a different type of restriction on the selected method. Although accurate final predictions possibly exploiting non-linearities in the data is the ultimate goal, the econometrician favours interpretable models (Einav and Levin, 2014). Some machine learning methods, such as neural networks, fall in the category of so called 'black-box' models (Hastie et al., 2009). Such models provide no exact information on the linkages between the response and the predictive variables or this information is hard to visualize. Thus the interpretability of the model, in addition to the ability to handle potentially large predictor sets, needs to be taken into account when selecting the appropriate forecasting method with modern economic data.

The rest of this introductory chapter is organized as follows. Section 1.2 presents the econometric framework with different machine learning methods by allowing for various functional forms and loss functions. In addition, short summaries of the three essays are presented in Section 1.3.

1.2 Econometric framework

This section considers the underlying functional forms and objective functions of the five machine learning methods used in this thesis. These are the gradient

boosting machine, random forest, neural networks, support vector machine and k-nearest neighbor method.

1.2.1 More flexible functional forms using machine learning

The goal of statistical modelling and prediction is to specify the functional relationship between the dependent variable and the predictors. In economic forecasting we typically assume the response at time $t + h$, where h is the forecast horizon, to be a function of the predictive variables and forecast error

$$y_{t+h} = F(\mathbf{x}_t) + e_{t+h}, \qquad t = 1, \ldots, N, \tag{1.1}$$

where y_{t+h} is the response, $F(\mathbf{x}_t)$ is a potentially non-linear function of the predictors $\mathbf{x}_t$ and e_{t+h} is the forecast error. In order to make the function approximation strategy feasible $F(\cdot)$ is typically restricted to certain parameterized class of functions.

Linear regression model has been the cornerstone of statistical analysis for several decades (Hastie et al., 2009). In the linear regression model we assume the underlying functional form to be a linear function of the predictor variables

$$F(\mathbf{x}_t) = \beta_0 + \mathbf{x}_t'\boldsymbol{\beta}, \tag{1.2}$$

where β_0 is the constant term and $\mathbf{x}_t$ is a $p \times 1$ vector of predictors. For many real-world applications the assumption of linearity is very restrictive. Machine learning methods allow for more flexible functional forms and can discover all sorts of complex relationships in data. It should however be noted that the linearity assumption could easily be incorporated to majority of these methods as well.

As an extension to the linear model in (1.2) let us consider a single hidden-layer feed-forward neural network with linear output. In this case the network consists of three layers: the input layer, a single hidden layer with M hidden nodes and the output layer. The subsequent layers in the network are connected with each other through weights. The network architecture induced by the assumptions of a single hidden-layer and feed-forward weights is the most commonly used one in practice (Bishop, 2006). The final model in such a network is a linear combination of the original predictors going through a

non-linear activation function

$$F(\mathbf{x}_t) = \sum_{m=1}^{M} \gamma_m \sigma(b_m + \mathbf{x}_t' \boldsymbol{\alpha}_m), \tag{1.3}$$

where b_m is a bias term and $\boldsymbol{\alpha}_m$ is a $p \times 1$ vector of weights connecting the predictors to the hidden node m. Similarly γ_m is the weight connecting the hidden node m to the final output node. One can note the resemblance of the term inside the brackets to the linear model provided in (1.2). Without the non-linear activation function $\sigma(\cdot)$, which usually is a sigmoid function or hyperbolic tangent, the final output would be a linear combination of linear combinations and hence also linear. The total amount of hidden nodes M limits the range of functions that a single hidden layer neural network model can approximate. With a sufficient amount for M the model in (1.3) can approximate any continuous function to arbitrary accuracy (see e.g., Hornik, Stinchcombe and White, 1989; Cybenko, 1989).

In the support vector machine by Vapnik (1995) non-linearity is introduced to the model by transforming the original input feature space into an enlarged space using non-linear functions. In a classification context the data could be linearly separable in this higher dimensional feature space. By considering a d-dimensional set of transformation functions we have $h(\mathbf{x}_t) = (h_1(\mathbf{x}_t), \ldots, h_d(\mathbf{x}_t))'$ and the final model can be written as

$$F(\mathbf{x}_t) = \beta_0 + h(\mathbf{x}_t)' \boldsymbol{\beta}. \tag{1.4}$$

Note that although the classification problem might be linearly separable in the higher dimensional feature space, when transformed back to the original feature space it results in a non-linear decision boundary. Instead of the exact transformation $h(\mathbf{x}_t)$ a kernel function, which computes inner products in the transformed space, is sufficient. A radial basis function and a dth-degree polynomial are typical choices for the kernel function (Hastie et al., 2009). With a linear kernel the final output is also a linear function of the predictors.

Linear regression model in (1.2) assumes a global linear function whereas in the k-nearest neighbor method originally presented by Fix and Hodges (1951) the underlying function is assumed to be well approximated by a locally constant function. Let $\kappa(\mathbf{x}_t)$ denote the indices of the k observations closest to

$\mathbf{x}_t$ based on some distance metric. With a continuous response the final model in k-nearest neighbor method is simply an average of the k responses

$$F(\mathbf{x}_t) = \frac{1}{k} \sum_{i \in \kappa(\mathbf{x}_t)} y_{i+h}, \tag{1.5}$$

where y_{i+h} is the response attached to index i (taking into account the forecasting horizon h). In the classification approach the final model is based on a majority vote between the k datapoints and ties are broken at random. Parameter k controls for the complexity of the model, where smaller values for k result in more flexible models (Bishop, 2006).

In the ensemble methods random forest and gradient boosting the final model, also called an ensemble, has an additive form

$$F(\mathbf{x}_t) = \sum_{m=1}^{M} f_m(\mathbf{x}_t), \tag{1.6}$$

where $f_m(\mathbf{x}_t)$ is the base learner function at iteration m. Note that the single hidden-layer feed-forward network in (1.3) and the support vector machine in (1.4) can also be seen as additive models with unique base learner functions (Friedman, 2001). In random forest the base learner in (1.6) is a classification or regression tree, while in gradient boosting one can consider for example linear or spline based functions as well.

The power of random forest is based on creating a collection of independent trees that are minimally correlated with each other (Breiman, 2001). Depending on the learning problem the final prediction is either a majority vote or an average of the predictions induced by each individual tree. Gradient boosting takes a slightly different approach as the final model is built in a forward stagewise manner by adding new base learner functions to the ensemble that best fit the negative gradient of the loss function. Provided with sufficient amount of data and a flexible base learner, such as regression trees or smoothing splines, boosting can basically approximate any kind of functional form.

1.2.2 Loss function

In the general estimation problem the goal is to find the function $F(\mathbf{x}_t)$ that minimizes the expected loss of some predefined loss function

$$\widehat{F}(\mathbf{x}_t) = \underset{F(\mathbf{x}_t)}{\arg\min}\, E\left[\mathcal{L}\left(y_{t+h}, F(\mathbf{x}_t)\right)\right], \tag{1.7}$$

where y_{t+h} is the response and $\mathbf{x}_t$ is a vector of predictor variables. As was considered in the previous section the optimal function with each method is assumed to belong to some predefined class of functions. The actual functional form of the loss function $\mathcal{L}(\cdot)$ in (1.7) depends on both the response and the considered machine learning method. In economic forecasting problems the response is typically either continuous or discrete with two or more classes.

With a continuous response both the traditional linear regression model and majority of the machine learning methods aim to minimize the (sample) sum of squared error (SSE)

$$L_{SSE} = \sum_{t=1}^{N} \big(y_{t+h} - F(\mathbf{x}_t)\big)^2. \tag{1.8}$$

In the binary two-class classification problem the typically considered loss function is the binomial deviance. Using more familiar terminology the binomial deviance can simply be expressed as the negative log-likelihood (Hastie et al., 2009). Assuming a logistic transformation function the deviance can be written as

$$L_{Dev} = -\sum_{t=1}^{N} \big(y_{t+h}F(\mathbf{x}_t) - \log\left(1 + \exp(F(\mathbf{x}_t))\right)\big). \tag{1.9}$$

In the multinomial extension there are L classes and a separate functional estimate $F_l(\mathbf{x}_t)$ for each class. By denoting each class l of the multinomial response with a separate binary variable $y_{t+h,l}$ the multinomial deviance can be written as

$$L_{MDev} = -\sum_{t=1}^{N}\sum_{l=1}^{L} y_{t+h,l}\log(p_{t,l}(\mathbf{x}_t)), \tag{1.10}$$

where $p_{t,l}(\mathbf{x}_t)$ is the symmetric multiple logistic transform for class l.

The support vector machine takes a slightly different approach. The original idea of support vector machine is presented in a classification context, where the final goal is to produce a margin maximizing classifier (see Vapnik, 1995). Evgeniou, Pontil and Poggio (2000) however show that the optimization problem in support vector machine can also be presented in terms of minimizing a regularized loss function

$$L_{SVM} = \sum_{t=1}^{N} V(y_{t+h}, F(\mathbf{x}_t)) + \lambda J(F). \tag{1.11}$$

In support vector machines we have a specific form for both the general error measure $V(\cdot)$ and the penalty function $J(\cdot)$. The general error measure also depends on whether we are dealing with a regression or classification problem.

Random forest and k-nearest neighbor algorithm can not be conceived as direct optimization procedures. The k-nearest neighbor algorithm by Fix and Hodges (1951) is a model-free method since the classification or prediction of a new observation is based purely on the data points of the training set. The search for the k nearest neighbors to each new datapoint is based on some distance measure say the euclidean distance, but this distance metric can not be considered as a conventional loss function. Similarly for random forest although each independent tree in the final ensemble is optimized based on the usual criterions with classification and regression trees, such as the least squares criterion or the gini index, the entire ensemble is not directly optimizing any loss function (Wyner et al., 2017).

1.3 Summary of the essays

In this section, we review three empirical applications on economic forecasting using machine learning methods. In Section 1.3.1, we study the predictability of binary economic recession periods. Section 1.3.2 considers predicting multinomial stock returns, where the continuous daily stock returns are discretized into three classes. In the last section we take a step further from the classical stock return predictability studies and focus on optimizing asset allocation decisions directly.

1.3.1 Recession forecasting with big data

In this chapter we study the ability of the gradient boosting machine to forecast economic recession periods in the U.S. using high-dimensional data. Recessions are shorter events compared to expansion periods leading to quite heavily imbalanced binary class labels. The class imbalance with large amount of predictive variables creates a risk of forecasting only the non-recessionary periods well. We propose a new cost-sensitive extension to the gradient boosting model using binary class weights. In this approach the sample counterpart of (1.7) is a weighted average instead of the arithmetic mean. We use the data-based approach by Zhou (2012) and choose the binary weights according to the class imbalance observed in the dataset. To the best of our knowledge cost-sensitive gradient boosting model using class weights has not been utilized in previous economic research.

The results confirm the finding of Blagus and Lusa (2017) who note that the performance of a gradient boosting model can be rather poor with high class imbalance, especially when a high-dimensional dataset is used. The cost-sensitive extension to the gradient boosting model using class weights can take the class imbalance problem into account and produces strong warning signals for the U.S. recessions with different forecasting horizons. Different types of interest rate spreads are the most important predictors with each forecasting horizon.

1.3.2 Forecasting multinomial stock returns using machine learning methods

The five different machine learning methods presented in Section 1.2 are compared in their ability to predict the daily returns of the S&P 500 stock market index. Majority of the previous research focus on predicting the actual level and then partly the direction of stock returns (i.e. the signs of stock returns, indicating that we are interested in a binary-dependent time series generated from returns). In this chapter the returns are categorized into three classes based on the upper and lower quartiles of the return series. The less informative and noisy fluctuation associated with small absolute returns is isolated in one class and the other two focus on predicting the large absolute returns. To the best of our knowledge such multinomial approach has not

been taken into consideration in previous economic research.

All the machine learning methods examined produce multinomial classification results which are significant from both the statistical and economic point of view. Among the machine learning methods considered the gradient boosting machine turns out to be the top-performer. The results also show how the predictability of large absolute returns tend to cluster around certain periods of time. These periods are typically associated with high stock market volatility. The conclusion of increased predictability during market turmoil is in line with the findings of Krauss, Do and Huck (2017) and Fiévet and Sornette (2018).

1.3.3 Moving forward from predictive regressions: Boosting asset allocation decisions

As pointed by Leitch and Tanner (1991), among many others, the statistically significant predictability of stock returns does not necessarily imply economic profitability and vice versa. For an individual investor or portfolio manager the economic value of the predictions is the ultimate goal instead of the merely considered statistical predictability. In this chapter we estimate the portfolio weights directly using gradient boosting by incorporating a custom objective function in the context of (1.7). Our contribution is strongly connected to a more general issue in (financial) econometrics on what is the appropriate objective (loss) function in econometric inference (see e.g., Elliott and Timmermann, 2016, chapter 2).

Our empirical results on the monthly U.S. market returns show that substantial and quantitatively meaningful economic value can be obtained with our utility boosting method. Technical indicators yield as a group the largest benefits in out-of-sample forecasting experiments. This is generally in line with the conclusions of Neely et al. (2014) and now confirmed with very different methodology.

References

Bishop, C. M. (2006). *Pattern Recognition and Machine Learning (Information Science and Statistics)*. Springer-Verlag, Berlin, Heidelberg.

Blagus, R. and Lusa, L. (2017). Gradient boosting for high-dimensional prediction of rare events. *Computational Statistics & Data Analysis*, 113:19 – 37.

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

Bühlmann, P. (2006). Boosting for high-dimensional linear models. *Annals of Statistics*, 34(2):559–583.

Choi, H. and Varian, H. (2012). Predicting the present with google trends. *Economic Record*, 88(s1):2–9.

Cybenko, G. (1989). Approximation by superpositions of a sigmoidal function. *Mathematics of Control, Signals, and Systems*, 2(4):303–314.

Einav, L. and Levin, J. (2014). Economics in the age of big data. *Science*, 346(6210).

Elliott, G. and Timmermann, A. (2016). *Economic Forecasting*. Princeton University Press, 1 edition.

Ettredge, M., Gerdes, J., and Karuga, G. (2005). Using web-based search data to predict macroeconomic statistics. *Commun. ACM*, 48(11):87–92.

Evgeniou, T., Pontil, M., and Poggio, T. A. (2000). Regularization networks and support vector machines. *Advances in Computational Mathematics*, 13:1–50.

Fiévet, L. and Sornette, D. (2018). Decision trees unearth return sign predictability in the s&p 500. *Quantitative Finance*, pages 1–18.

Fix, E. and Hodges, J. (1951). *Discriminatory Analysis: Nonparametric Discrimination: Consistency Properties*. USAF School of Aviation Medicine, Randolph Field, TX.

Friedman, J. H. (2001). Greedy function approximation: A gradient boosting machine. *Annals of Statistics*, 29(5):1189–1232.

Galar, M., Fernández, A., Tartas, E. B., Bustince, H., and Herrera, F. (2012). A review on ensembles for the class imbalance problem: Bagging-, boosting-, and hybrid-based approaches. *IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews)*, 42:463–484.

Hastie, T., Tibshirani, R., and Friedman, J. (2009). *The elements of statistical learning: data mining, inference and prediction*. Springer, 2 edition.

He, H. and Garcia, E. A. (2009). Learning from imbalanced data. *IEEE Transactions on Knowledge and Data Engineering*, 21(9):1263–1284.

Henderson, J. V., Storeygard, A., and Weil, D. N. (2012). Measuring economic growth from outer space. *American Economic Review*, 102(2):994–1028.

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

Krauss, C., Do, X. A., and Huck, N. (2017). Deep neural networks, gradient-boosted trees, random forests: Statistical arbitrage on the s&p 500. *European Journal of Operational Research*, 259(2):689 – 702.

Kuan, C.-M. and Liu, T. (1995). Forecasting exchange rates using feedforward and recurrent neural networks. *Journal of Applied Econometrics*, 10(4):347–364.

Leitch, G. and Tanner, J. E. (1991). Economic forecast evaluation: Profits versus the conventional error measures. *The American Economic Review*, 81(3):580–590.

Mitchell, W. C. and Burns, A. F. (1938). *Statistical Indicators of Cyclical Revivals*. NBER.

Mullainathan, S. and Spiess, J. (2017). Machine learning: An applied econometric approach. *Journal of Economic Perspectives*, 31(2):87–106.

Neely, C. J., Rapach, D. E., Tu, J., and Zhou, G. (2014). Forecasting the equity risk premium: The role of technical indicators. *Management Science*, 60(7):1772–1791.

Stock, J. H. and Watson, M. W. (1999). *A Comparison of Linear and Nonlinear Univariate Models for Forecasting Macroeconomic Time Series*, pages 1–44. Oxford University Press, Oxford.

Stock, J. H. and Watson, M. W. (2003). Forecasting output and inflation: The role of asset prices. *Journal of Economic Literature*, 41(3):788–829.

Vapnik, V. N. (1995). *The Nature of Statistical Learning Theory*. Springer-Verlag, Berlin, Heidelberg.

Wheelock, D. C. and Wohar, M. E. (2009). Can the term spread predict output growth and recessions? a survey of the literature. *Federal Reserve Bank of St. Louis Review*, 91:419–440.

Wohlrabe, K. and Buchen, T. (2014). Assessing the macroeconomic forecasting performance of boosting: Evidence for the united states, the euro area and germany. *Journal of Forecasting*, 33(4):231–242.

Wyner, A. J., Olson, M., Bleich, J., and Mease, D. (2017). Explaining the success of adaboost and random forests as interpolating classifiers. *Journal of Machine Learning Research*, 18(48):1–33.

Zhou, Z.-H. (2012). *Ensemble Methods: Foundations and Algorithms*. Chapman & Hall/CRC, 1st edition.

Chapter 2

Recession forecasting with big data

Abstract * †

In this chapter, a large amount of different financial and macroeconomic variables are used to predict the U.S. recession periods. We propose a new cost-sensitive extension to the gradient boosting model which can take into account the class imbalance problem of the binary response variable. The class imbalance, caused by the scarcity of recession periods in our application, is a problem that is emphasized with high-dimensional datasets. Our empirical results show that the introduced cost-sensitive extension outperforms the traditional gradient boosting model in both in-sample and out-of-sample forecasting. Among the large set of candidate predictors, different types of interest rate spreads turn out to be the most important predictors when forecasting U.S. recession periods.

* The author would like to thank Heikki Kauppi, Henri Nyberg, Simone Maxand and seminar participants at the Annual Meeting of the Finnish Economic Association and FDPE Econometrics workshop for constructive comments. The financial support from the Emil Aaltonen Foundation is gratefully acknowledged.

† A paper based on this chapter is available in SSRN working paper series (id3630146).

2.1 Introduction

Recessions are painful periods with a significant and widespread decline in economic activity. Early warning signals of recessions would be important for different kinds of economic agents. Households, firms, policymakers and central bankers could all utilize the information concerning upcoming economic activity in their decision making. The probability of a recession is fairly straightforward to interpret and can be easily taken into consideration in all kinds of economic decision making.

But what are the indicators that consistently lead recessions? Since the early work of Estrella and Mishkin (1998) there has been a large amount of empirical research concerning the predictive content of different economic and financial variables (see e.g., Nyberg, 2010; Liu and Moench, 2016). The amount of potential recession indicators is growing rapidly as the constraints related to data-availability and computational power keep diminishing. Traditionally used binary logit and probit models can only handle small predictor sets at a time, which makes the search for the best predictors quite difficult.

Recent developments in the machine learning literature provide a solution to this problem. State of the art supervised learning algorithm called gradient boosting is able to do variable selection and model estimation simultaneously. Non-parametric boosting can handle huge predictor sets and the estimated conditional probability function can take basically any kind of form. The main objective of this research is to explore how we can exploit high-dimensional datasets when making recession forecasts with the gradient boosting model.

The business cycle consists of positive and negative fluctuations around the long-run growth rate of the economy. These fluctuations are also known as expansions and recessions. The official business cycle chronology for the U.S. is published by the National Bureau of Economic Research (NBER). Recessions are shorter events compared to expansion periods leading to quite heavily imbalanced binary class labels. In our dataset less than 14 percent of the monthly observations are classified as recessions. This class imbalance and the effects on classification is well covered in the machine learning literature (see e.g., Galar et al., 2012). Suprisingly the scarcity of recession periods has not been properly taken into consideration in previous economic research.

Two approaches are usually considered when dealing with imbalanced

classes: resampling techniques and cost-sensitive learning methods (see e.g., He and Garcia, 2009). Resampling is the easiest and most commonly used alternative. The dataset could be balanced by drawing a random sample without replacement from the majority class, which is called undersampling. In the recession forecasting setup the size of the dataset is already very limited so this could create problems when estimating the model, especially with high-dimensional data. In the oversampling approach the idea is to sample with replacement from the minority class. He and Garcia (2009) argue that the duplicate observations from the minority class can lead to overfitting.

Instead of replicating existing observations from the minority class one could learn the characteristics in this class and create synthetic samples based on feature space similarities. This synthetic minority oversampling technique also known as SMOTE is a popular alternative when dealing with imbalanced data. Blagus and Lusa (2013) however find that variable selection is needed before running SMOTE on high-dimensional datasets.

Cost-sensitive learning methods can take the class imbalance into account without artificially manipulating the dataset. In a variety of real-life classification problems, such as recession forecasting or fraud detection, misclassifying the minority class can be considered very costly. The cost-sensitivity can be incorporated into the model by attaching a higher penalty for misclassifying the minority class. Several modified versions of the adaboost algorithm by Freund and Schapire (1996) exist, where the weight updating rule of the original algorithm is modified to better account for the class imbalance (see e.g., Sun et al., 2007; Fan et al., 1999; Ting, 2000).

This is natural since weight updating is a crucial part of the adaboost algorithm designed purely for classification problems. However this is not the case with the more general gradient boosting algorithm presented by Friedman (2001) that can handle variety of problems beyond classification and the cost-sensitivity have to be incorporated otherwise. We propose a cost-sensitive extension to the gradient boosting model by introducing a binary class weight to each observation in the dataset that reflect the asymmetric misclassification costs. To the best of our knowledge cost-sensitive gradient boosting model using class weights has not been utilized in previous economic research.

The traditional gradient boosting model has been utilized in previous

economic research with mixed results. Ng (2014) uses the gradient boosting model with stump regression trees to predict recession periods in the U.S. The dataset used by Ng (2014) has a fairly large predictor set and is from the same source as the dataset used in this paper. With this model setup Ng (2014) concludes that the gradient boosting model is far from perfect in forecasting recessions.

Berge (2015) uses a smaller predictor set to forecast U.S. recessions with the gradient boosting model. The results show how boosting outperforms other model selection techniques such as Bayesian model averaging. Moreover, the results highlight the importance of non-linearity in recession forecasting as boosting with non-linear smoothing splines outperforms boosting with a linear final model. Döpke, Fritsche and Pierdzioch (2017) succesfully forecast German recession periods with the gradient boosting model using regression trees. Unlike Ng (2014) they build larger trees which allow for potential interaction terms between predictors. This approach is used in this study as well.

Our results confirm the finding of Blagus and Lusa (2017) who note that the performance of a gradient boosting model can be rather poor with high class imbalance, especially when a high-dimensional dataset is used. The out-of-sample forecasting ability of the traditional gradient boosting model is quite heavily deteriorated compared to the in-sample results. The cost-sensitive extension to the gradient boosting model using class weights can take the class imbalance problem into account and produces strong warning signals for the U.S. recessions with different forecasting horizons.

The cost-sensitive gradient boosting models estimated using huge predictor sets rely heavily on different kinds of interest rate spreads. This is also the case with the short and medium term forecasting horizons although different variables related to the real economy are also available in the dataset. The internal model selection capability of gradient boosting confirms that predictors with predictive power beyond the term spread are quite hard to find (see e.g., Estrella and Mishkin, 1998; Liu and Moench, 2016).

The results also show how the chosen lag length for a predictor can vary substantially from the forecasting horizon considered. A similar observation has been made by Kauppi and Saikkonen (2008) in the conventional probit model. The term spread is the dominant predictor when forecasting recessions

one year ahead, which is a common finding in the previous literature (see e.g., Dueker, 1997; Estrella and Mishkin, 1998).

The rest of the paper is organized as follows. The gradient boosting framework and the cost-sensitive extension to the gradient boosting model are introduced in Section 2.2. The dataset and the empirical analysis are presented in Section 2.3. Section 2.4 concludes.

2.2 Methodology

The following theoretical framework for the gradient boosting model follows closely the original work of Friedman (2001).

2.2.1 Gradient boosting

Considering two stochastic processes y_t and $\mathbf{x}_{t-k}$ of which y_t is a binary dependent variable of form

$$y_t = \begin{cases} 1, & \text{if economy in recession at time t} \\ 0, & \text{if economy in expansion at time t} \end{cases} \tag{2.1}$$

and $\mathbf{x}_{t-k}$ is a $p \times 1$ vector of predictive variables. The lag length k of each predictor must satisfy the condition $k \geq h$, where h is the forecasting horizon. If $E_{t-k}(\cdot)$ and $P_{t-k}(\cdot)$ denote conditional expectation and conditional probability given the information set available at time $t-k$ and by assuming the logistic transform $\Lambda(\cdot)$ the conditional probability can be written as

$$E_{t-k}(y_t) = P_{t-k}(y_t = 1) = p_t = \Lambda(F(\mathbf{x}_{t-k})). \tag{2.2}$$

We can model this conditional probability by estimating the function $F(\mathbf{x}_{t-k})$ with the gradient boosting model. Exponential loss and binomial deviance are popular alternatives for the loss function to be minimized with binary classification problems. These are second order equivalent (Friedman, Hastie and Tibshirani, 2000). In this research the conditional probability is estimated with the gradient boosting model by minimizing the binomial deviance loss function.

In the general estimation problem the goal is to find the function $F(\mathbf{x}_{t-k})$

that minimizes the expected loss of some predefined loss function

$$\widehat{F}(\mathbf{x}_{t-k}) = \underset{F(\mathbf{x}_{t-k})}{\arg\min}\, E\left[\mathcal{L}\left(y_t, F(\mathbf{x}_{t-k})\right)\right]. \tag{2.3}$$

Even for a simple parametric model, where $F(\mathbf{x}_{t-k})$ is assumed to be a linear function of the covariates, numerical optimization techniques are usually needed for solving the parameter vector that minimizes the expected loss in equation (2.3). Steepest descent optimization technique is a simple alternative. The parameter search using steepest descent can be summarized with the following equation

$$\boldsymbol{\beta}^* = \sum_{m=0}^{M} \boldsymbol{\beta}_m = \sum_{m=0}^{M} -\delta_m \mathbf{g}_m, \tag{2.4}$$

where β_0 is the initial guess and $\{\beta_m\}_{m=1}^{M}$ are steps towards the optimal solution. The negative gradient vector $-\mathbf{g}_m$ determines the direction of each step and δ_m is the stepsize obtained by a line search.

With gradient boosting the optimization takes place in the function space instead of the conventional parameter space. Similarly as in the parametric case numerical optimization methods are needed when searching for the optimal function. Some further assumptions are required in order to make the numerical optimization in the function space feasible with finite datasets. By restricting the function search to some parameterized class of functions the solution to numerical optimization can be written as

$$F^*(\mathbf{x}_{t-k}) = \sum_{m=0}^{M} f_m(\mathbf{x}_{t-k}) = \sum_{m=0}^{M} \delta_m b(\mathbf{x}_{t-k}; \boldsymbol{\gamma}_m), \tag{2.5}$$

where δ_m is the stepsize obtained by line search as in equation (2.4). Now the step "direction" is given by the function $b(\mathbf{x}_{t-k}; \boldsymbol{\gamma}_m)$ also known as the base learner function. This can be a simple linear function or highly non-linear such as splines or regression trees. In this paper regression trees are used and the parameter vector $\boldsymbol{\gamma}_m$ consists of the splitting variables and splitpoints of the regression tree. Equation (2.5) also incorporates the original idea of boosting. The possibly very complex final ensemble $F(\mathbf{x}_{t-k})$ with strong predictive ability is a sum of the fairly simple base learner functions $f_m(\mathbf{x}_{t-k})$.

Using the sample counterpart of the loss function in equation (2.3) and

by plugging in the additive form introduced in equation (2.5) the estimation problem can be written as

$$\min_{\{\delta_m,\boldsymbol{\gamma}_m\}_{m=1}^{M}} \frac{1}{N}\sum_{t=1}^{N} L\Big(y_t, \sum_{m=0}^{M}\delta_m b(\mathbf{x}_{t-k};\boldsymbol{\gamma}_m)\Big). \tag{2.6}$$

This minimization problem can be approximated using forward stagewise additive modeling technique. This is done by adding new base learner functions to the expansion without altering the functions already included in the ensemble. At each step m the base learner function $b(\mathbf{x}_{t-k};\boldsymbol{\gamma}_m)$ which best fits the negative gradient of the loss function is selected and added to the ensemble. Using least squares as the fitting criterion while searching for the optimal base learner function leads to the general gradient boosting algorithm by Friedman (2001):

Algorithm 2.1 *Gradient boosting*

$F_0(\mathbf{x}_{t-k}) = \arg\min_{\rho} \frac{1}{N}\sum_{t=1}^{N} L\left(y_t, \rho\right)$

for $m \leftarrow 1$ *to* M ***do:***

$\tilde{y}_t = -\frac{\partial L(y_t, F(\mathbf{x}_{t-k}))}{\partial F(\mathbf{x}_{t-k})}\Big|_{F(\mathbf{x}_{t-k})=F_{m-1}(\mathbf{x}_{t-k})}, t = 1,\ldots,N$

$\boldsymbol{\gamma}_m = \arg\min_{\boldsymbol{\gamma},\delta} \sum_{t=1}^{N} \left[\tilde{y}_t - \delta b(\mathbf{x}_{t-k};\boldsymbol{\gamma})\right]^2$

$\rho_m = \arg\min_{\rho} \sum_{t=1}^{N} L\left(y_t, F_{m-1}(\mathbf{x}_{t-k}) + \rho b(\mathbf{x}_{t-k};\boldsymbol{\gamma}_m)\right)$

$F_m(\mathbf{x}_{t-k}) = F_{m-1}(\mathbf{x}_{t-k}) + \rho_m b(\mathbf{x}_{t-k};\boldsymbol{\gamma}_m)$

end for

Friedman (2001) suggests a slight modification to Algorithm 2.1 when regression trees are used as the base learner function. Regression trees are a simple yet powerful tool that partition the feature space into a set of J non-overlapping rectangles and attach a simple constant to each one. The base learner function of a J-terminal node regression tree can be written as

$$b(\mathbf{x}_{t-k};\{c_j, R_j\}_{j=1}^{J}) = \sum_{j=1}^{J} c_j I(\mathbf{x}_{t-k} \in R_j), \tag{2.7}$$

where the functional estimate is a constant c_j in region R_j. According to Friedman (2001), the additive J-terminal node regression tree in equation (2.7) can be seen as a combination of J separate base learner functions. One base learner for each terminal node of the regression tree. Therefore after estimating the terminal node regions $\{R_{jm}\}_{j=1}^{J}$ at the mth iteration with least squares on line 4 of the Algorithm 2.1 the line search step on line 5 should produce separate estimates for each terminal node of the regression tree. This minimization problem can be written as

$$\{\hat{c}_{jm}\}_{j=1}^{J} = \underset{\{c_j\}_{j=1}^{J}}{\arg\min} \sum_{t=1}^{N} L\Big(y_t, F_{m-1}(\mathbf{x}_{t-k}) + \sum_{j=1}^{J} c_j I(\mathbf{x}_{t-k} \in R_{jm})\Big). \tag{2.8}$$

The ensemble update on the last line of Algorithm 2.1 is then a sum of these J terminal node estimates obtained in equation (2.8)

$$F_m(\mathbf{x}_{t-k}) = F_{m-1}(\mathbf{x}_{t-k}) + \sum_{j=1}^{J} \hat{c}_{jm} I(\mathbf{x}_{t-k} \in R_{jm}).$$

2.2.2 Cost-sensitive gradient boosting with class weights

With a high class imbalance there is a risk that the estimated binary classifier is skewed towards predicting the majority class well (He and Garcia, 2009). An algorithm can be made cost-sensitive by weighting the dataspace according to the misclassification costs (Branco, Torgo and Ribeiro, 2016). This weighting approach is sometimes referred to as rescaling in the previous literature (see e.g., Zhou and Liu, 2010). The asymmetric misclassification costs, which are the building block of cost-sensitive learning, are incorporated to the gradient boosting model by introducing a binary class weight for each observation in the data. In the traditional gradient boosting model the sample counterpart of the loss function is the sample mean and the minimization problem can be written as in equation (2.6). By introducing a vector of class weights we end up minimizing the weighted average of the sample loss function

$$\min_{\{\delta_m, \boldsymbol{\gamma}_m\}_{m=1}^{M}} \frac{1}{\sum_{t=1}^{N} w_t} \sum_{t=1}^{N} w_t L\Big(y_t, \sum_{m=1}^{M} \delta_m b(\mathbf{x}_{t-k}; \boldsymbol{\gamma}_m)\Big). \tag{2.9}$$

If the weights w_t are equal for each observation the weighted average in equation (2.9) reduces to the sample mean.

Elkan (2001) suggests weighting the minority class observations according to the ratio in misclassification costs. Suppose c_{10} denote the cost when we fail to predict a recession and c_{01} when we give a false alarm of recession. The optimal weight for the minority class observations is then

$$w^* = \frac{c_{10}}{c_{01}}. \tag{2.10}$$

In many cases the exact misclassification costs are unknown and we must rely on rules such that misclassifying the minority class is more costly (Maloof, 2003). The class weights are basically arbitrary as they depend on the unknown preferences how harmful different types of misclassification is considered to be. In this paper we use the data-based approach by Zhou (2012) and choose the weights according to the class imbalance observed in the dataset

$$w_t = \begin{cases} \frac{\sum_{t=1}^{N}(1-y_t)}{\sum_{t=1}^{N} y_t}, & \text{if } y_t = 1 \\ 1, & \text{if } y_t = 0 \end{cases}. \tag{2.11}$$

As can be seen from equation (2.11) the weights depend on the ratio of the number of datapoints in both classes. These binary weights ensure that the sum of weights are equal in both classes. The aim of choosing these weights is to force the algorithm to provide a balanced degree of predictive accuracy between the two classes.

The cost-sensitive gradient boosting algorithm with class weights follows the steps described in Algorithm 2.1 but the binary class weights can have an effect on each step of the algorithm. Table 2.1 illustrates how the class weights alter different parts of the gradient boosting algorithm, when J-terminal node regression trees are used as the base learner functions and the loss function to be minimized is the binomial deviance.

Table 2.1: The effect of class weights on the gradient boosting algorithm

Step	Value
Loss function	$\frac{-2\sum_{t=1}^{N} w_t[y_t F(\mathbf{x}_{t-k}) - \log(1+e^{F(\mathbf{x}_{t-k})})]}{\sum_{t=1}^{N} w_t}$
Initial value	$F_0(\mathbf{x}_{t-k}) = \log(\frac{\sum_{t=1}^{N} w_t y_t}{\sum_{t=1}^{N} w_t(1-y_t)})$
Gradient	$\tilde{y}_{tm} = y_t - p_t, \quad where \quad p_t = \frac{1}{1+e^{-F_{m-1}(\mathbf{x}_{t-k})}}$
Split criterion	$i^2(R_l, R_r) = \frac{W_l W_r}{W_l + W_r}(\bar{g}_l - \bar{g}_r)^2, \quad W_l = \sum_{x_{t-k} \in R_l} w_t, \quad \bar{g}_l = \frac{1}{W_l} \sum_{x_{t-k} \in R_l} w_t \tilde{y}_{tm}$
Terminal node estimate	$\hat{c}_{jm} = \frac{\sum_{x_{t-k} \in R_j} w_t(y_t - p_t)}{\sum_{x_{t-k} \in R_j} w_t p_t(1-p_t)}$

Note that the values for each step of the ordinary gradient boosting model can be obtained from Table 2.1 by setting all the weights equal to one. The cost-sensitive and the traditional gradient boosting algorithms differ starting from the initial values. As the first gradient vector is based on the initial value the gradients are also different. The biggest differences between these two algorithms however are related to the estimation of the regression tree base learners at each iteration m of the algorithm. Blagus and Lusa (2017) argue that the class imbalance problem of the gradient boosting model with high-dimensional data is related to the inappropriately defined terminal regions R_j.

Next we will consider how class weights can have an effect on both the estimated terminal node regions and the terminal node estimates of the regression tree base learner. When J-terminal node regression tree is used as the base learner function, the $J-1$ recursive binary splits into regions R_l and R_r dividing the predictor space into J non-overlapping terminal node regions $\{R_j\}_{j=1}^{J}$ are obtained by maximizing the least-squares improvement criterion. These splits are based on a slightly different criterion if class weights are used. For this reason the estimated terminal node regions and the terminal node estimates can be different between the two algorithms.

From Table 2.1 we can see how the split criterion is based on two parts.

The first part $\frac{W_l W_r}{W_l + W_r}$ illustrates how each split into regions R_l and R_r in cost-sensitive gradient boosting is based on the sum of weights in these two categories instead of the number of observations. The latter part of the split criterion $(\bar{g}_l - \bar{g}_r)^2$ shows that instead of the average gradient we compare the weighted average of the gradient in the regions, when searching for the optimal split point. From the last row in Table 2.1 one can note how the terminal node estimates are functions of both the terminal node regions and the class weights itself and hence the final estimates can be different between the two algorithms.

2.2.3 Regularization parameters in gradient boosting

Friedman (2001, 2002) introduces several add-on requlariza tion techniques to reduce the risk of overfitting or to improve the overall performance of the gradient boosting algorithm. The parameters related to these techniques are often called tuning parameters since it is up to the user to finetune the parameter values for the particular problem at hand. Tuning parameters with the gradient boosting technique can be divided into two categories: parameters related to the overall algorithm and parameters related to the chosen base learner function.

Friedman (2001) incorporates a simple shrinkage strategy to slow down the learning process. In this strategy each update of the algorithm is scaled down by a constant called learning rate. The ensemble update on the last line of Algorithm 2.1 can then be written as

$$F_m(\mathbf{x}_{t-k}) = F_{m-1}(\mathbf{x}_{t-k}) + \upsilon \rho_m b(\mathbf{x}_{t-k}; \boldsymbol{\gamma}_m),$$

where $0 < \upsilon \leq 1$ is the learning rate. Learning rate is a crucial part of the gradient boosting algorithm as it controls the speed of the learning process by shrinking each gradient descent step towards zero. Friedman (2001) suggests to set the learning rate small enough for better generalization ability. Bühlmann and Yu (2010) reach a similar conclusion.

Breiman (1996) notes that introducing randomness when building each tree in an ensemble can lead to substantial gains in prediction accuracy. Based on these findings Friedman (2002) develops stochastic gradient boosting in which subsampling is used to enhance the generalization ability of the gradi-

ent boosting model. At each round of the algorithm a random subsample of datapoints is drawn without replacement and the new base learner function is fitted using this random subsample. Simulation studies show that subsampling fraction around one half seems to work best in most cases (Friedman, 2002).

The total amount of iterations M needed however moves in the opposite direction to learning rate and subsampling. Gradient boosting is a flexible technique which can approximate basically any kind of functional form with sufficient amount of data. This flexibility can also come with a cost. Overfitting the training data is a risk that must be taken into consideration as it can lead to decreased generalization ability of the model. The optimal amount of iterations is usually chosen with early stopping methods such as using an independent test set or cross-validation.

When the amount of observations is scarce K-fold cross-validation is often the only alternative since we can not afford to set aside an independent test set. K-fold cross-validation is based on splitting the data into K non-overlapping folds. Each of these folds is used as a test set once while the model is estimated using the remaining $K-1$ folds. To reduce the effect of randomness the K-fold cross-validation process can be repeated R times (Kim, 2009). In the repeated K-fold cross-validation approach the estimate for the optimal stopping point is based on the average validation error produced by the K folds at each of these R repeats.

Instead of the traditional repeated K-fold we use a more conservative cross-validation approach since the risk of overfitting the data in the high-dimensional setup is fairly high. In this conservative approach only the validation error produced by the fold, which first reaches its minimum and therefore first starts to show signs of overfitting, is selected out of the K folds at each repetition. By denoting the found "weakest" fold in repetition r as k_r^*, the number of observations in this fold as $N_{k_r^*}$ and the model estimated without this fold as $\hat{F}^{-k_r^*}(\mathbf{x}_{t-k})$ the conservative cross-validation estimate for the prediction error can be written as

$$CV = \frac{1}{R}\sum_{r=1}^{R}\frac{1}{N_{k_r^*}}\sum_{t=1}^{N_{k_r^*}} L\big(y_t, \hat{F}^{-k_r^*}(\mathbf{x}_{t-k})\big), \tag{2.12}$$

where binomial deviance is used as the loss function $L(\cdot)$. The final estimate for the amount of iterations is the point where the estimated prediction error in (2.12) reaches its minimum. To the best of our knowledge this simple conservative approach has not been used in the previous academic research.

The complexity of the regression tree base learners is controlled by the number of terminal nodes J in each regression tree. The amount of inner nodes $(J-1)$ in the regression tree limit the potential amount of interaction between predictors as shown with the ANOVA expansion of a function

$$F(\mathbf{x}_{t-k}) = \sum_j f_j(x_j) + \sum_{j,k} f_{jk}(x_j, x_k) + \sum_{j,k,l} f_{jkl}(x_j, x_k, x_l) + \cdots . \quad (2.13)$$

The simplest regression tree with just two terminal nodes can only capture the first term in equation (2.13). Higher order interactions are needed to be able to capture the latter terms, which are functions of more than one variable. These higher-order interactions require deeper trees. Hastie, Tibshirani and Friedman (2009) argue that trees with more than ten terminal nodes are seldom needed with boosting.

2.3 Results

2.3.1 Data and model setup

The dataset used in the empirical analysis is the FRED-MD monthly dataset. The selected timespan covers the period from January 1962 to June 2017. After dropping out variables that are not available for the full period the FRED-MD dataset consists of 130 different economic and financial variables related to different parts of the economy.[1] Three different forecasting horizons h are studied in the empirical analysis: short ($h = 3$), medium ($h = 6$) and long ($h = 12$).

All the available lag lengths k of the predictors up to 24 months are considered as potential predictors (assuming $k \geq h$). The total amount of predictors in the dataset take the value of 2860, 2470 or 1690 depending on the length of

[1] All ISM-series (The Institute for Supply management) have been removed from the FRED-MD dataset starting from 2016/6. These series have been re-obtained using Macrobond. For more general information about the dataset see https://research.stlouisfed.org/econ/mccracken/fred-databases/

the forecasting horizon. For example, the total amount of predictors with the shortest forecasting horizon is 2860, which includes 22 different lags of these 130 variables. See Christiansen, Eriksen and Møller (2014) for a similar study where each lag is considered as a separate predictor.

The term spread has been noted as the best single predictor of recessions and economic growth in general in the U.S. (see e.g., Dueker, 1997; Estrella and Mishkin, 1998; Wohar and Wheelock, 2009). To see if it is actually worthwhile to go through these huge predictor sets with the gradient boosting models, we use a simple logit model with the term spread as a benchmark model. Kauppi and Saikkonen (2008) note that setting the lag length k equal to the forecasting horizon h may not be optimal in all cases. To take this into account we introduce the six nearest lag lengths of the term spread as additional predictors. The term spread is measured as the interest rate spread between the 10-year government bonds and the effective federal funds rate as this is included in the FRED-MD dataset.

The estimated conditional probabilities for different models are evaluated using the receiver operating characteristic curve (ROC). The area under the ROC-curve (AUC) measures the overall classification ability of the model without restricting to a certain probability threshold. AUC-values closer to one indicate better classification ability whereas values close to one half are no better than a simple coin toss. For a more comprehensive review of the AUC-measure in economics context see e.g., Berge and Jordà (2011) and Nyberg and Pönkä (2016).

The gradient boosting model involves internal model selection as the regression trees selected at each step of the algorithm may be functions of different predictors. Some predictors are chosen more often than others and can be considered more important. Breiman et al. (1984) introduce a measure for the relevance of a predictor x_p in a single J-terminal node regression tree T

$$\hat{I}_p^2(T) = \sum_{j=1}^{J-1} \hat{\imath}_j^2 I(v_j = p), \tag{2.14}$$

where v_j is the splitting variable of inner node j and $\hat{\imath}_j^2$ is the empirical improvement in squared error as a result of this split. The least squares improvement criterion was introduced in Table 2.1.

The measure in equation (2.14) is based on a single tree, but it can be generalized to additive tree expansions as well (Friedman, 2001). The relative influence of a variable x_p for the entire gradient boosting ensemble is simply an average over all the trees $\{T_m\}_{m=1}^{M}$ in the ensemble

$$\hat{I}_p^2 = \frac{1}{M} \sum_{m=1}^{M} \hat{I}_p^2(T_m). \tag{2.15}$$

The relative influence measure in equation (2.15) is used to illustrate the most important recession indicators with the gradient boosting model. The relevance of a predictor x_p in the recursive out-of-sample forecasting is the average $\hat{I}_p^2$ of the estimated models.

The following results are obtained using the R programming environment for statistical computing (R Core Team, 2017). The GBM-package (Ridgeway, 2017) with bernoulli loss function is used to estimate the gradient boosting models. With such huge predictor sets it is likely that there are interaction between some predictors. For this reason the maximum tree depth is set to 8 leading to regression trees with nine terminal nodes. Döpke et al. (2017) use 6-terminal node regression trees while predicting recessions in Germany with a much smaller predictor set.

The minimum number of observations required in each terminal node of a regression tree is set to one allowing the tree building process to be as flexible as possible. Similar results are obtained when setting the minimum number of observations to five as is used by Döpke et al. (2017).[2] Learning rate is set to a low value of 0.005 and the default value of 0.5 is used as the subsampling fraction. The conservative cross-validation approach presented in equation (2.12) is conducted using 5 folds and 5 repeats throughout this research to find the optimal amount of iterations. In order to keep the computational time feasible the maximum amount of iterations is set to 800.

2.3.2 In-sample results

Three different models are compared in the in-sample analysis using the full dataset. The benchmark model (bm) is a simple logit model with seven lags of the term spread as predictors. GBM is the ordinary gradient boosting model

[2] Results upon request.

and wGBM stands for the cost-sensitive gradient boosting model with class weights. The class weights are formed according to equation (2.11). The binary response variable for each model is the business cycle chronology provided by the NBER.

Table 2.2 summarizes the in-sample performance as measured with the area under the ROC-curve (AUC) of these three models for all the different forecasting horizons. The rows of the table present the different models and the columns stand for the forecasting horizons considered. The validation AUCs from the 5-fold cross-validation repeated five times are reported in parenthesis.

Table 2.2: In-sample AUC (1962/01 - 2017/06)

	Forecast horizon, Months		
Model specification	3	6	12
Benchmark	0.890 (0.881)	0.910 (0.902)	0.914 (0.897)
GBM	1.000 (0.985)	1.000 (0.980)	1.000 (0.956)
wGBM	1.000 (0.987)	1.000 (0.981)	1.000 (0.961)

As expected, the non-linear gradient boosting models do a better job forecasting recessions in-sample. The larger information set and the more flexible functional form of the GBM-models allow for a more detailed in-sample fit. The perfect in-sample AUCs for the GBM-models can raise questions of overfitting. As a result of using these moderate sized regression trees as base learner functions the GBM-models achieve nearly perfect classification ability after only a few iterations. This can be confirmed by training a shallow single decision tree to the full dataset. The single decision tree alone is sufficient to produce very high in-sample AUCs, even after restricting the predictor space to consider only the eight different interest rate spreads (and their lag lengths).[3] Thereby it is not completely surprising that an ensemble of trees

[3] The in-sample AUCs with a single decision tree are close to or well above 0.95 depending on the forecasting horizon. On the other hand, restricting the GBM-models by considering only the simplest stump regression trees and / or only the interest rate spreads as predictors are not sufficient as models produce in-sample AUCs of one or really close to it. Results upon request.

yield a perfect in-sample fit as measured with AUC. For example, the cost sensitive GBM-model with the shortest forecasting horizon reaches an AUC value of 0.997 after just five iterations. However it should be noted that the estimated conditional probabilites at this point range between 0.488 and 0.512, values that are only slightly different from the initial value of one half because of the shrinkage strategy described in Section 2.2.3. It could be argued that the AUC may not be the most suitable criterion when evaluating the in-sample performance in this setup. But since the main emphasis is on the out-of-sample performance of the models the AUCs are reported here for comparison.

The validation AUCs reported in Table 2.2 provide additional insight into the potential overfitting problem since large deviations between the in-sample and validation performance is typically seen as a sign of overfitting. The validation AUCs for the GBM-models are of similar magnitude as the in-sample AUCs and therefore do not indicate overfitting. Döpke et al. (2017) also report validation AUCs close to one when forecasting recessions in Germany with the gradient boosting model. The validity of the traditional random sampling techniques used in cross-validation with such a highly autocorrelated binary response variable should be further examined. This however is beyond the scope of this research.

Table 2.2 shows how the cost sensitive GBM-model outperforms the other two models as measured with the validation AUC, although the difference between the two GBM-models is small. The gap in validation AUCs between the benchmark and GBM-models decreases slightly as the forecasting horizon grows. Graphical illustrations are an important part of recession forecasting since these can give a better picture of the false alarms and other potential problems related to the models. The estimated conditional probabilities that the economy is in recession h-months from now are calculated according to equation (2.2). These in-sample estimated conditional probabilites are illustrated in Figure 2.1 for each of the three models and forecasting horizons.

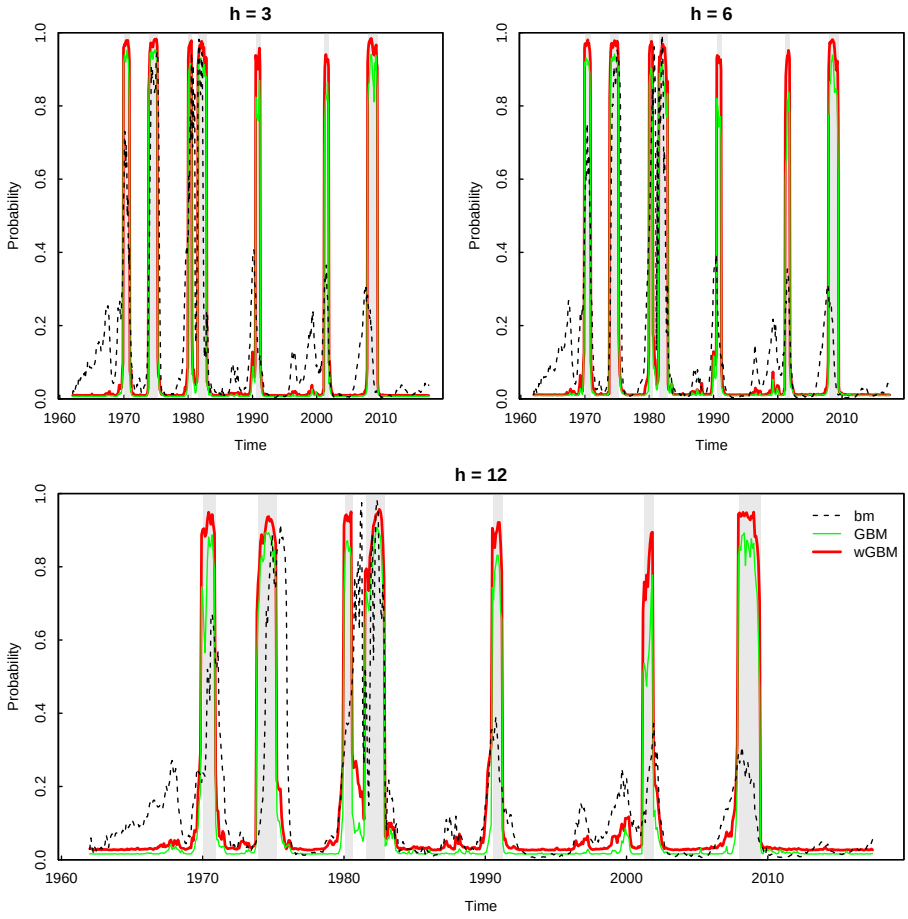

Figure 2.1: In-sample estimated conditional probabilities

The conditional probabilities for both GBM-models can be seen to mimic the shaded recession periods quite nicely. The in-sample fits for the two GBM-models have a rather similar shape without any major differences, which is in line with the results in Table 2.2. However, the recession signals produced by the cost-sensitive GBM-model are constantly stronger compared to the other two models with all the forecasting horizons. It is also noteworthy how the benchmark logit model produces a lot weaker signals for the last three recessions compared to the GBM-models. Figure 2.1 also shows how the estimated conditional probabilities for the GBM-models are not exactly zero or one and the in-sample fit is not perfect in probability terms. Using forecast performance evaluation criterion other than AUC, such as the binomial de-

viance or the quadratic probability score, would not indicate perfect in-sample fit.

2.3.3 Out-of-sample results

Good in-sample results may not always reflect the out-of-sample predictive ability of the model. An expanding window forecasting procedure is used to examine the true predictive ability of the models. Both Berge (2015) and Ng (2014) use rolling window when forecasting U.S. recessions. To ensure the maximum sample size for the estimation of each model an expanding window approach is used in this study.

The out-of-sample evaluation period covers the period starting from December 1988 to June 2017. Because of high computational cost the GBM-models are re-estimated only once a year in December. The class weights are updated according to equation (2.11) as the proportion of zeros and ones change for the binary response. The business cycle recession and expansion periods are not available in real time. The publication lag of the NBER business cycle chronology is thus assumed to be 12 months.

The results from the recursive out-of-sample forecasting procedure are reported in Table 2.3. The out-of-sample performance as measured with the area under the ROC-curve is illustrated for the different models at each of the three forecasting horizons.

Table 2.3: Out-of-sample AUC (1988/12 - 2017/06)

	Forecast horizon, Months		
Model specification	3	6	12
Benchmark	0.748	0.811	0.919
GBM	0.841	0.816	0.867
wGBM	0.915	0.861	0.928

The out-of-sample AUCs show that the cost-sensitive GBM-model outperforms the other two models with all the forecasting horizons. The difference in AUCs between the traditional and cost-sensitive gradient boosting models are quite similar with all the forecasting horizons. The average difference of

the AUCs between the two GBM-models is 0.06.

The out-of-sample performance for the traditional GBM-model is quite heavily deteriorated when compared to the in-sample AUCs reported in Table 2.2. The standard GBM-model can outperform the benchmark model only at the shortest forecasting horizon. This diminished out-of-sample forecasting ability of the traditional GBM-model could indicate problems related to the class imbalance of the response. Blagus and Lusa (2017) note that the traditional GBM-model can perform poorly on high-dimensional data with class imbalance. Figure 2.2 illustrates the out-of-sample estimated conditional probabilities calculated according to equation (2.2) for all the different forecasting horizons and models.

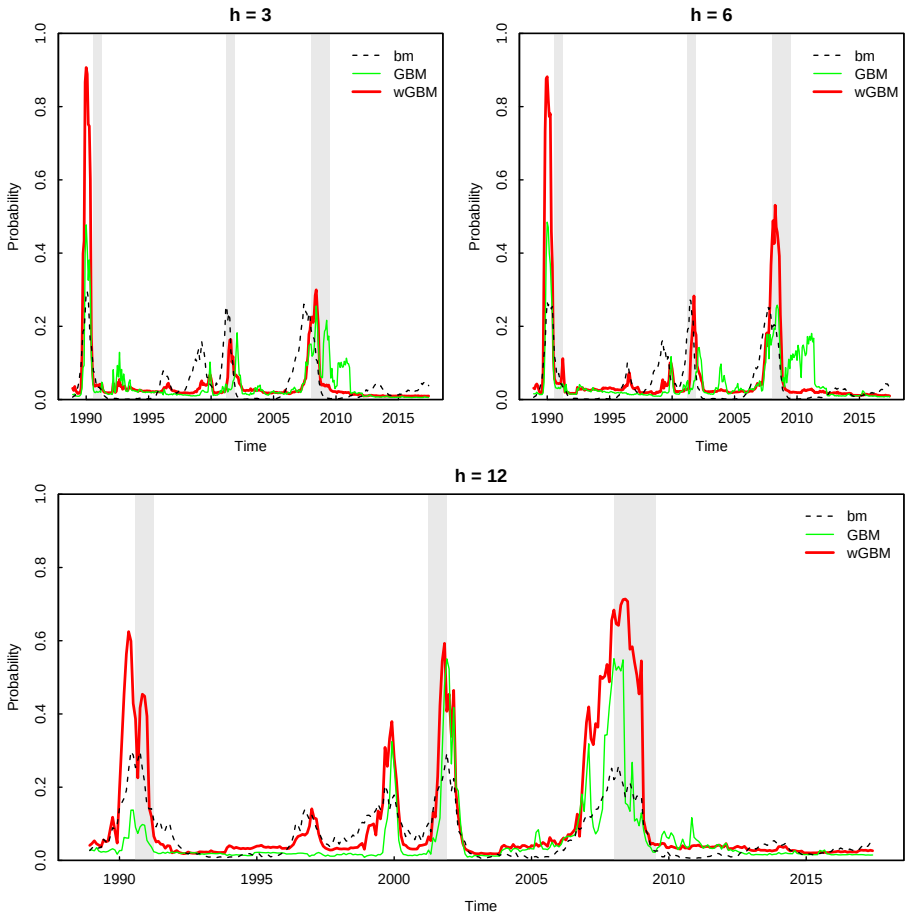

Figure 2.2: Out-of-sample estimated conditional probabilities

Figure 2.2 shows how the recession probabilities for each of the models with the short and medium term forecasting horizons spike just before the actual recession period in the early ninetees. Although these spikes are considered as false alarms and decrease the out-of-sample performance of the models, this hightened risk of an upcoming recession could have considerable practical importance.

Figure 2.2 also illustrates the problems related to the diminished out-of-sample performance of the traditional GBM-model. The traditional GBM-model provides several false alarms, especially at the short and medium term forecasting horizons. With the longest forecasting horizon the traditional GBM-model give a rather weak signal of the upcoming recession period in the early ninetees when compared to the other two models.

The cost-sensitive GBM-model on the other hand provides clear warnings of the upcoming recession periods in the short and medium term without any major false alarms. Although the recession signal for the second recession period with the shortest forecasting horizon is quite modest. It should be noted that the magnitude of the recession signals are diminished for each of the three models when compared to the in-sample probabilities in Figure 2.1.

With the 12-month forecasting horizon the cost-sensitive GBM-model provides strong warning signals for each of the three recessions. The estimated recession probabilities of the cost-sensitive GBM-model bears a close resemblance to the benchmark model. This also includes the two false alarms that are typical when predicting recessions with the term spread (see e.g., Kauppi and Saikkonen, 2008; Nyberg, 2010).

To further consider the composition of the estimated cost-sensitive GBM-models Table 2.4 presents the ten most important out-of-sample predictors according to the relative influence measure presented in equation (2.15).

Table 2.4: Top-10 out-of-sample predictors for wGBM

$h = 3$		$h = 6$		$h = 12$	
Variable	Rel.inf	Variable	Rel.inf	Variable	Rel.inf
6mth - FFrate_4	5.464	6mth - FFrate_6	8.722	10yr - FFrate_12	18.051
10yr - FFrate_9	4.920	10yr - FFrate_9	4.979	5yr - FFrate_15	6.347
6mth - FFrate_6	4.744	5yr - FFrate_15	4.026	5yr - FFrate_14	3.634
6mth - FFrate_5	4.581	1yr - FFrate_6	3.941	10yr - FFrate_13	2.778
6mth - FFrate_7	3.103	6mth - FFrate_7	3.739	5yr - FFrate_16	2.466
5yr - FFrate_15	2.988	3mth - FFrate_6	3.570	10yr - FFrate_14	2.400
10yr - FFrate_8	2.757	10yr - FFrate_8	3.111	5yr - FFrate_13	1.808
3mth - FFrate_6	2.337	1yr - FFrate_7	2.512	AAA - FFrate_12	1.787
1yr - FFrate_6	2.310	6mth - FFrate_8	2.175	5yr - FFrate_12	1.688
1yr - FFrate_7	2.254	10yr - FFrate_11	2.048	PERMITS_15	1.496

The cost-sensitive GBM-models rely heavily on different kinds of interest rate spreads as can be seen in Table 2.4. The only non-interest rate based predictor is the fifteenth lag of the new private housing permits variable (PERMITS_15) with the longest forecasting horizon. This is a bit surprising at the short and medium term forecasting horizons since variables describing the real economy are often found useful when predicting recessions with these forecasting horizons (see e.g., Berge, 2015). The heavy usage of interest rate spreads confirms that predictors with forecasting ability beyond the term spread are quite hard to find (see e.g., Estrella and Mishkin, 1998; Liu and Moench, 2016).

Models based on different kinds of interest rate spreads can be affected by the problems related to the predictive power of the term spread noted in the previous literature. Several studies show how the term spread forecast U.S. output growth less accurately after the mid 1980s (see e.g., Estrella, Rodrigues and Schich, 2003; Stock and Watson, 2003). The slightly lower out-of-sample AUCs reported in Table 2.3 for each of the three models, including the benchmark model, are in line with this finding.

Table 2.4 shows how the interest rate spread between the 6-month treasury bill and the effective federal funds rate with the fourth lag (6mth - FFrate_4) is

the most important predictor when predicting recessions three months ahead. The same predictor with the sixth lag is the most important predictor with the medium term forecasting horizon. The composition of the top-10 out-of-sample predictors are quite similar between the short and medium term horizons.

The chosen lag lengths of the predictors with the short and medium term horizons can deviate quite substantially from the length of the forecasting horizon. For example, the spread between the 5-year treasury bond and the effective federal funds rate with the fifteenth lag (5yr - FFrate_15) is an important predictor with both of these horizons. Similar observation can be made with the spread between the 10-year treasury bond and the effective federal funds rate with the ninth lag (10yr - FFrate_9). With the longest forecasting horizon the term spread with lag length equal to twelve (10yr - FFrate_12) has a very strong impact on the models as measured with the relative influence. Such dominance of a single predictor is not found with the short and medium term horizons.

2.4 Conclusions

This paper introduces a new cost-sensitive gradient boosting model which can take into account the class imbalance of the binary response variable. The cost-sensitive gradient boosting model is applied to predicting binary U.S. recession periods with a high-dimensional dataset of financial and macroeconomic variables. The internal model selection of the cost-sensitive gradient boosting algorithm provides important information about the most useful recession indicators and chosen lag lengths with different forecasting horizons.

The empirical results show how the cost-sensitive extension to the gradient boosting model produces stronger and more stable recession forecasts for the U.S. with each forecasting horizon compared to the traditional gradient boosting model. A logit model based on the term spread is used as a benchmark model to see if the more complex gradient boosting models provide predictive power beyond the best known simple model. The cost-sensitive model outperforms the benchmark model with each forecasting horizon whereas the traditional gradient boosting model is able to outperform the benchmark only at the shortest forecasting horizon. Different kinds of interest rate spreads are

the most important predictors, even with the short and medium term forecasting horizons. The term spread is the dominant predictor when forecasting recessions one year ahead.

The current research can be extended in several ways. First of all, the binary values for the class weights were chosen so that both the minority and the majority class receive similar attention in the learning process. Different choices for the class weights could be further examined. Especially in cases where the class imbalance is even more radical. The cost-sensitive approach could also be extended to multinomial classification problems, where different types of class imbalance problems can emerge. There could be for example more than one minority class with a multinomial response variable. Introducing model dynamics is another potential area for future research. This would allow iterative forecasts to be used instead of the forecast horizon-specific forecasts as in this study.

References

Berge, T. J. (2015). Predicting recessions with leading indicators: Model averaging and selection over the business cycle. *Journal of Forecasting*, 34(6):455–471.

Berge, T. J. and Jordà, Ò. (2011). Evaluating the classification of economic activity into recessions and expansions. *American Economic Journal: Macroeconomics*, 3(2):246–77.

Blagus, R. and Lusa, L. (2013). Smote for high-dimensional class-imbalanced data. *BMC Bioinformatics*, 14(1):106.

Blagus, R. and Lusa, L. (2017). Gradient boosting for high-dimensional prediction of rare events. *Computational Statistics & Data Analysis*, 113:19 – 37.

Branco, P., Torgo, L., and Ribeiro, R. (2016). A survey of predictive modeling on imbalanced domains. *ACM Computing Surveys*, 49.

Breiman, L. (1996). Bagging predictors. *Machine Learning*, 24(2):123–140.

Breiman, L., Friedman, J., Olshen, R., and Stone, C. J. (1984). *Classification and Regression Trees*. Wadsworth, New York.

Bühlmann, P. and Yu, B. (2010). Boosting. *Wiley Interdisciplinary Reviews: Computational Statistics*, 2(1):69–74.

Christiansen, C., Eriksen, J. N., and Møller, S. V. (2014). Forecasting us recessions: The role of sentiment. *Journal of Banking & Finance*, 49:459 – 468.

Döpke, J., Fritsche, U., and Pierdzioch, C. (2017). Predicting recessions with boosted regression trees. *International Journal of Forecasting*, 33(4):745–759.

Dueker, M. J. (1997). Strengthening the case for the yield curve as a predictor of u.s. recessions. *Federal Reserve Bank of St. Louis Economic Review*, 79:41–51.

Elkan, C. (2001). The foundations of cost-sensitive learning. In *In Proceedings of the Seventeenth International Joint Conference on Artificial Intelligence*, pages 973–978.

Estrella, A. and Mishkin, F. (1998). Predicting u.s. recessions: Financial variables as leading indicators. *The Review of Economics and Statistics*, 80(1):45–61.

Estrella, A., Rodrigues, A. R., and Schich, S. (2003). How stable is the predictive power of the yield curve? evidence from germany and the united states. *The Review of Economics and Statistics*, 85(3):629–644.

Fan, W., Stolfo, S. J., Zhang, J., and Chan, P. K. (1999). Adacost: Misclassification cost-sensitive boosting. In *Proceedings of the Sixteenth International Conference on Machine Learning*, ICML '99, pages 97–105, San Francisco, CA, USA. Morgan Kaufmann Publishers Inc.

Freund, Y. and Schapire, R. E. (1996). Experiments with a new boosting algorithm. In *Proceedings of the Thirteenth International Conference on International Conference on Machine Learning*, ICML'96, pages 148–156, San Francisco, CA, USA. Morgan Kaufmann Publishers Inc.

Friedman, J., Hastie, T., and Tibshirani, R. (2000). Additive logistic regression: A statistical view of boosting. *The Annals of Statistics*, 28:337–407.

Friedman, J. H. (2001). Greedy function approximation: A gradient boosting machine. *Annals of Statistics*, 29(5):1189–1232.

Friedman, J. H. (2002). Stochastic gradient boosting. *Computational Statistics & Data Analysis*, 38(4):367 – 378.

Galar, M., Fernández, A., Tartas, E. B., Bustince, H., and Herrera, F. (2012). A review on ensembles for the class imbalance problem: Bagging-, boosting-, and hybrid-based approaches. *IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews)*, 42:463–484.

Hastie, T., Tibshirani, R., and Friedman, J. (2009). *The elements of statistical learning: data mining, inference and prediction*. Springer, 2 edition.

He, H. and Garcia, E. A. (2009). Learning from imbalanced data. *IEEE Transactions on Knowledge and Data Engineering*, 21(9):1263–1284.

Kauppi, H. and Saikkonen, P. (2008). Predicting u.s. recessions with dynamic binary response models. *The Review of Economics and Statistics*, 90(4):777–791.

Kim, J.-H. (2009). Estimating classification error rate: Repeated cross-validation, repeated hold-out and bootstrap. *Computational Statistics & Data Analysis*, 53(11):3735 – 3745.

Liu, W. and Moench, E. (2016). What predicts us recessions? *International Journal of Forecasting*, 32(4):1138 – 1150.

Maloof, M. A. (2003). Learning when data sets are imbalanced and when costs are unequal and unknown. In *ICML-2003 Workshop on Learning from Imbalanced Data Sets II*.

Ng, S. (2014). Viewpoint: Boosting recessions. *Canadian Journal of Economics*, 47(1):1–34.

Nyberg, H. (2010). Dynamic probit models and financial variables in recession forecasting. *Journal of Forecasting*, 29(1-2):215–230.

Nyberg, H. and Pönkä, H. (2016). International sign predictability of stock returns: The role of the United States. *Economic Modelling*, 58(C):323–338.

R Core Team (2017). *R: A Language and Environment for Statistical Computing*. R Foundation for Statistical Computing, Vienna, Austria.

Ridgeway, G. (2017). *gbm: Generalized Boosted Regression Models*. R package version 2.1.3.

Stock, J. H. and Watson, M. W. (2003). Forecasting output and inflation: The role of asset prices. *Journal of Economic Literature*, 41(3):788–829.

Sun, Y., Kamel, M. S., Wong, A. K., and Wang, Y. (2007). Cost-sensitive boosting for classification of imbalanced data. *Pattern Recognition*, 40(12):3358 – 3378.

Ting, K. M. (2000). A comparative study of cost-sensitive boosting algorithms. In *Proceedings of the Seventeenth International Conference on Machine Learning*, ICML '00, pages 983–990, San Francisco, CA, USA. Morgan Kaufmann Publishers Inc.

Wheelock, D. C. and Wohar, M. E. (2009). Can the term spread predict output growth and recessions? a survey of the literature. *Federal Reserve Bank of St. Louis Review*, Part 1(Sep/Oct):419–440.

Zhou, Z.-H. (2012). *Ensemble Methods: Foundations and Algorithms*. Chapman & Hall/CRC, 1st edition.

Zhou, Z.-H. and Liu, X.-Y. (2010). On multi-class cost-sensitive learning. *Computational Intelligence*, 26(3):232–257.

Chapter 3

Forecasting multinomial stock returns using machine learning methods

Abstract * †

In this chapter, the daily returns of the S&P 500 stock market index are predicted using a variety of different machine learning methods. We propose a new multinomial classification approach to forecasting stock returns. The multinomial approach can isolate the noisy fluctuation around zero return and allows us to focus on predicting the more informative large absolute returns. Our in-sample and out-of-sample forecasting results indicate significant return predictability from a statistical point of view. Moreover, all the machine learning methods considered outperform the benchmark buy-and-hold strategy in a real-life trading simulation. The gradient boosting machine is the top-performer in terms of both the statistical and economic evaluation criteria.

* The author would like to thank Henri Nyberg, Heikki Kauppi, Jaakko Peltonen, Martti Juhola, Matthijs Lof, Luis Alvarez Esteban and the seminar participants at Turku Finance Workshop, GSE Econometrics workshop and CFE-CMStatistics 2019 for excellent comments. This work was supported by the Emil Aaltonen Foundation and the Academy of Finland.

† A paper based on this chapter has been accepted for publication in the *Journal of Finance and Data Science* (forthcoming).

3.1 Introduction

Forecasting stock returns has attracted a tremendous amount of interest ever since the introduction of computers to economic forecasting. Kendall (1953) was among the first to reach the conclusion of no predictability in stock prices. Later Fama (1970) stated in the famous efficient market hypothesis that abnormal returns should not be possible to make by using historical data. But has the ever-growing amount of information and computational power in recent decades changed this relationship? State of the art machine learning methods, which can handle large amounts of information and discover complex relationships in data, provide further insight if profitable trading strategies can be discovered using past information.

The predictability of stock returns is a controversial subject. In a comprehensive study, Welch and Goyal (2008) argue that the predictability found for the level of stock returns in the previous literature is time inconsistent and does not hold when new data is introduced. More recent work by Neely et al. (2014), among others, challenge the view of Welch and Goyal (2008) by reporting statistically significant predictability using more sophisticated forecasting methods.

Instead of the actual level of return another strand of literature focuses on predicting the binary sign of stock returns (i.e. directional predictability of stock returns). Leung, Daouk and Chen (2000) provide early evidence in favor of using the binary response variable instead of the actual level. Other studies reporting statistically significant predictability using monthly stock returns are, for example, Nyberg (2011) and Nyberg and Pönkä (2016). Christoffersen and Diebold (2006) show theoretically that sign predictability may exist even without the assumption of mean-predictability.

Although the majority of the previous literature concerns predicting monthly returns, some more recent studies have reported sign predictability using daily returns as well (see e.g., Skabar, 2013; Zhong and Enke, 2017; Fiévet and Sornette, 2018; Karhunen, 2019). The main objective in this research is to predict daily stock returns of the U.S. stock market (more specifically S&P 500 index returns) using different machine learning methods.

Directional prediction of stock returns is based on forecasting whether returns are greater than some pre-specified threshold. Previous research mainly

focuses on sign prediction, where this threshold is equal to zero (i.e. whether the return is positive or negative), but some other alternatives have also been considered. Linton and Whang (2007) use the estimated unconditional quantile of the return as a threshold. Chung and Hong (2007) express the threshold as multiples of the estimated standard deviation when forecasting the direction of exchange rates. Both studies find evidence of directional predictability in asset returns using different statistical testing procedures.

Directional prediction of stock returns has a close connection to the market timing models considered by Merton (1981) and Pesaran and Timmermann (1995). Directional prediction of stock returns leads to simple binary trading strategies which can be used to assess the economic significance of the forecasting ability. Predicting the sign of stock returns involves a large amount of asset allocation decisions and the costs related to these transactions can be problematic when compared to the benchmark buy-and-hold strategy. This problem is even more alleviated with daily data (Becker and Leschinski, 2018).

By considering two different thresholds instead of just one the directional prediction problem becomes multinomial. The signal-to-noise ratio in stock returns is fairly low, especially with daily data (Becker and Leschinski, 2018). Chung and Hong (2007) argue that the informational content of large absolute returns may be more valuable whereas small returns in absolute value are merely noise. It is also noted that the co-movement of individual stocks with the market portfolio is stronger with large absolute returns (see e.g., Longin and Solnik, 2001; Ang and Chen, 2002; Hong, Tu and Zhou, 2007). The multinomial response allows us to isolate some of the noise and put more emphasis on predicting the large absolute returns. The multinomial directional prediction also enables a richer set of possible trading strategies. For example one could choose between buying, holding and selling stocks instead of merely so far considered binary buy and sell decisions. To the best of our knowledge multinomial stock returns have not been utilized in previous economic research.

Our results confirm the previous findings of sign predictability in stock returns. All the machine learning methods considered in this research produce multinomial classification results which are significant from both the statistical and economical point of view. Each method is able to outperform the benchmark buy-and-hold strategy in a real-life trading simulation when

trading costs are taken into account. Among the machine learning methods considered an ensemble method called gradient boosting is the top-performer in terms of both the classification accuracy and the profits from a real-life trading simulation.

The results also show how the predictability of large absolute returns tend to cluster around certain periods of time. This is in line with the findings of Krauss, Do and Huck (2017) and Fiévet and Sornette (2018) who notice increased predictability during high market turmoil. A closely related conclusion often reported in the financial literature is the higher return predictability during business cycle recession periods (see e.g., Henkel, Martin and Nardari, 2011; Cujean and Hasler, 2017). Events such as the financial crisis or the European debt crisis involve high volatility in the stock markets but also highly profitable trading opportunities. Our results show that volatility in the stock market as measured by the VIX-index is the single most influential predictor of next days' stock returns.

The remainder of this paper is organized as follows. The prediction problem and the different machine learning methods are presented in Section 3.2. The dataset and the model selection process for different machine learning methods are described in Section 3.3. The empirical analysis and the results are covered in Section 3.4. Section 3.5 concludes.

3.2 Methodology

3.2.1 Multinomial stock returns

Financial literature usually focuses on the reward of holding a risky asset such as stocks compared to the risk-free investment. This excess return is denoted as

$$Z_t = r_t - rf_t, \tag{3.1}$$

where r_t is the logarithmic daily return of the S&P 500 stock market index at time t and rf_t is the 3-month Treasury bill yield[1]. In directional prediction the binary dependent variable is created from the return series in equation (3.1)

[1] The Federal Reserve reports annualized yields using a 360-day year also known as the bank discount method. The daily yield is therefore calculated as $rf_t = tbill_t \cdot \frac{1}{360}$.

using an indicator function

$$B_t(c) = I(Z_t > c), \tag{3.2}$$

where c is a given threshold. The multinomial response variable with three classes can be derived from the continuous stock returns using two thresholds c_1 and c_2

$$R_t(c_1, c_2) = \begin{cases} 1, & \text{if } Z_t < c_1 \\ 2, & \text{if } c_1 \leq Z_t \leq c_2 \\ 3, & \text{if } Z_t > c_2 \end{cases}. \tag{3.3}$$

A natural question is how to choose the two thresholds that are basically arbitrary. To the best of our knowledge the multinomial approach with two thresholds as in equation (3.3) has not been considered in the previous literature regarding directional prediction of stock returns. Majority of the previous literature with single threshold as in equation (3.2) focus on binary sign prediction, where $c = 0$ (see e.g., Leung et al., 2000; Karhunen, 2019). Although previous literature on directional prediction of stock returns with a single non-zero threshold is quite scarce some alternatives have been considered.

Chung and Hong (2007) argue that the choice of c can be based on the observed data or alternatively held fixed using the magnitude of transaction costs for example. In their data based approach Chung and Hong (2007) use multiples of the estimated standard deviation as a threshold when forecasting the direction of exchange rates. Linton and Whang (2007) consider different unconditional quantiles of the return series when testing for directional prediction in stock returns. Linton and Whang (2007) report statistically significant predictability in daily returns for all but the most extreme quantiles where the amount of data is insufficient.

Maheu and McCurdy (2004) show that large price changes of individual stocks are driven by important news and these large changes tend to be clustered together. It is also noted using market level data that large absolute stock returns contain stronger positive autocorrelation than small absolute returns do and they are therefore more predictable (see e.g., Granger and Ding, 1996). Setting the thresholds c_1 and c_2 in equation (3.3) further apart from zero may result in more predictability but also in more imbalanced classes.

Since the main objective of this research is to compare the predictive ability

of several different machine learning methods we have chosen to use the upper and lower quartiles of the return series as thresholds. This data based approach yields nicely balanced classes as one half of the observations are coming from the middle class in equation (3.3) and the other half from the "abnormal" classes. Well balanced classes also allow for similar rules to be used with each method in the classification process, where the probability estimates are transformed into classification.

Consider the stochastic processes R_t and $\mathbf{x}_{t-1}$, where R_t is the multinomial response variable described in equation (3.3) and $\mathbf{x}_{t-1}$ is a $p \times 1$ vector of predictors at time $t-1$. Conditional on the information set we assume the response variable to follow a categorical distribution

$$R_t|\Omega_{t-1} \sim Cat(\mathbf{p}_t),$$

where Ω_{t-1} is the information set available at time $t-1$ and $\mathbf{p}_t$ is a $k \times 1$ vector of conditional probabilities. Each element of $\mathbf{p}_t$ is the conditional probability of class k being the observed class at time t. More formally the conditional probability for each class k can be written as

$$p_{tk}(\mathbf{x}_{t-1}) = P(R_t = k|\Omega_{t-1}), \quad k = 1, \ldots, K, \tag{3.4}$$

where K will concretely be $K = 3$ in this study. The conditional probabilities in equation (3.4) must satisfy $0 \leq p_{tk} \leq 1$ and $\sum_{k=1}^{K} p_{tk} = 1$. These conditions are met by the symmetric multiple logistic transform. The conditional probabilities for each class k can be constructed using the functional estimates $F_k(\mathbf{x}_{t-1})$ as

$$p_{tk}(\mathbf{x}_{t-1}) = \frac{e^{F_k(\mathbf{x}_{t-1})}}{\sum_{l=1}^{K} e^{F_l(\mathbf{x}_{t-1})}}. \tag{3.5}$$

In the general K-class classification problem the goal is to find the function that minimizes the expected loss of some predefined loss function for each class k

$$\{\widehat{F}_k(\mathbf{x}_{t-1})\}_{k=1}^{K} = \underset{\{F_k(\mathbf{x}_{t-1})\}_{k=1}^{K}}{\arg\min} \; E\left[\mathcal{L}\left(R_t, \{F_k(\mathbf{x}_{t-1})\}_{k=1}^{K}\right)\right].$$

For the majority of methods used in this research the loss function considered

is the multinomial deviance

$$\mathcal{L}\left(R_t, \{F_k(\mathbf{x}_{t-1})\}_{k=1}^{K}\right) = -\sum_{k=1}^{K} I(R_t = k) \log p_{tk}(\mathbf{x}_{t-1}), \tag{3.6}$$

where $p_{tk}(\mathbf{x}_{t-1})$ is the logistic transform presented in equation (3.5).

Accuracy is used as the evaluation metric to compare the classification performance of different machine learning methods to be introduced in the next section (Section 3.2.2). Accuracy is calculated as the proportion of correctly classified data points in the considered sample

$$Acc = \frac{1}{N} \sum_{t=1}^{N} I(\hat{R}_t = R_t),$$

where $\hat{R}_t$ is the predicted class and R_t the true class label at time t.

3.2.2 Machine learning methods

k-Nearest neighbor classifier

The k-nearest neighbor originally presented by Fix and Hodges (1951) can be considered a model-free classification method since the classification of a new observation is based purely on the data points of the training set. Our training set consists of N pairs $\{(\mathbf{x}_{t-1}, R_t)\}_{t=1}^{N}$, where $\mathbf{x}_{t-1}$ is the vector of feature values and R_t is the multinomial response variable given in equation (3.3). In order to classify a new data point $\mathbf{x}_N$ we need to find the k data points in the training data closest to the new data point based on some distance measure. The Euclidean distance is the most commonly used alternative. These k data points are called the nearest neighbors of $\mathbf{x}_N$. The final classification is based on a majority vote of the response values of these k nearest neighbors. Ties are broken at random. This process is repeated for each data point in the test set.

Figure 3.1 illustrates how the k-nearest neighbor classification works with a small artificial dataset containing three classes. The left-hand side of Figure 3.1 illustrates the nearest neighbors of two new data points based on Euclidean distance. The amount of nearest neighbors k is assumed to be either 1 or 3. For two example locations marked with X, the solid gray circle shows the one nearest neighbor and dashed circle illustrates the neighbors when k equals

three. The right-hand side of Figure 3.1 shows the decision boundary for this artificial data when k equals one.

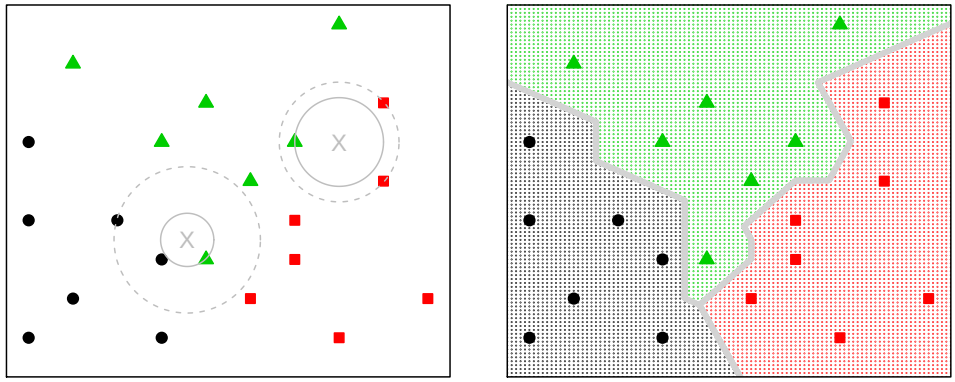

Figure 3.1: k-Nearest neighbor classification

The k-nearest neighbor classification is used as a benchmark method in this research because it is fairly easy to finetune. The only tuning parameter of the method is the amount of neighbors k. Larger values of k lead to smoother and less detailed decision boundaries. Despite its simplicity k-nearest neighbor has shown success in different kinds of financial applications such as forecasting foreign exchange rates or stock market volatility (see e.g., Meade, 2002; Arroyo and Maté, 2009; Andrada-Félix, Fernández-Rodríguez and Fuertes, 2016). Since the features in the dataset could have a variety of different scales each feature is typically re-scaled to have mean zero and variance equal to one.

Gradient boosting

The classification algorithm called adaboost was first introduced by Freund and Schapire (1996). For a long time the classification ability of the adaboost algorithm remained controversial. This was until Friedman, Hastie and Tibshirani (2000) created a statistical framework for the boosting procedure and showed how the adaboost algorithm fits an additive logistic regression model. The more general gradient boosting algorithm was discovered as Friedman (2001) introduced the connection to numerical optimization in function space. The more general gradient boosting algorithm can be used for both classification and regression problems.

In gradient boosting the goal is to find the function minimizing the expected loss of some predetermined loss function

$$\widehat{F}(\mathbf{x}_{t-1}) = \underset{F(\mathbf{x}_{t-1})}{\arg\min}\, E\left[\mathcal{L}\left(y_t, F(\mathbf{x}_{t-1})\right)\right].$$

In order to keep the notation fairly simple let us consider the binary classification problem, where $y_t \in \{0, 1\}$ and $\mathcal{L}(y_t, F(\mathbf{x}_{t-1}))$ is the binomial deviance. With the multinomial response variable presented in equation (3.3) a separate function is estimated for each class, which complicates the notation.

Gradient boosting is an ensemble method, where the possibly very complex final model is a combination of simple models called base learners

$$F_M(\mathbf{x}_{t-1}) = \sum_{m=1}^{M} f_m(\mathbf{x}_{t-1}). \tag{3.7}$$

The base learners $f(\mathbf{x}_{t-1})$ are assumed to belong to some parameterized class of functions. These could be for example simple linear models, spline functions or regression trees. The base learner used in this study is the J-terminal node regression tree, which splits the predictor space into J disjoint regions and attaches a constant to each region. Mathematically the J-terminal node regression tree base learner can be written as

$$f(\mathbf{x}_{t-1}; \{c_j, R_j\}_{j=1}^{J}) = \sum_{j=1}^{J} c_j I(\mathbf{x}_{t-1} \in R_j), \tag{3.8}$$

where $c_j \in \mathbb{R}$ is the functional estimate in region R_j.

Figure 3.2 illustrates J-terminal node regression trees graphically and plots a 4-terminal node regression tree and the terminal node regions created by this tree. The left-hand side depicts the classical tree shape. Each of the three split points is a function of the splitting variables and split locations. The right-hand side of Figure 3.2 shows the terminal node regions $\{R_j\}_{j=1}^{4}$ and the split locations $\{t_l\}_{l=1}^{3}$in a 2-dimensional space.

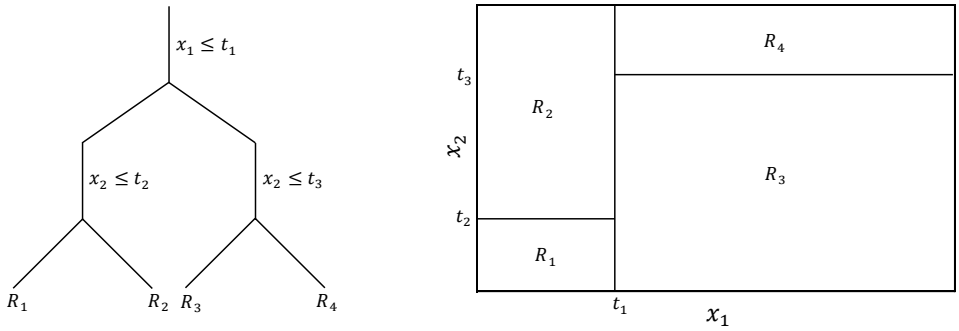

Figure 3.2: 4-terminal node regression tree

The final ensemble in equation (3.7) is estimated in a greedy stagewise fashion using a method called forward stagewise additive modeling. The estimation of a gradient boosting model is described in Algorithm 3.1. The algorithm starts with an initial value, which is a simple constant based on the considered loss function. At each iteration m of the gradient boosting algorithm a new base learner function which best fits the negative gradient of the loss function is selected and added to the current ensemble F_{m-1}. With the J-terminal node regression tree in equation (3.8) as the base learner this corresponds to finding the J non-overlapping terminal node regions $\{R_{jm}\}_{j=1}^{J}$ using a least squares criterion. After finding the terminal node regions the functional estimates $\hat{c}_{jm}$ are obtained in a simple minimization problem. The current ensemble F_{m-1} is then updated with the functional estimates before calculating the pseudo responses $\tilde{y}_t$ for the next round of the algorithm.

Algorithm 3.1 *Gradient boosting using J-terminal node regression trees*

$F_0(\mathbf{x}_{t-1}) = \arg\min_{\rho} \frac{1}{N} \sum_{t=1}^{N} L\left(y_t, \rho\right)$

for $m \leftarrow 1$ *to* M ***do:***

$\tilde{y}_t = - \frac{\partial L(y_t, F(\mathbf{x}_{t-1}))}{\partial F(\mathbf{x}_{t-1})}_{F(\mathbf{x}_{t-1}) = F_{m-1}(\mathbf{x}_{t-1})}, \qquad t = 1, \ldots, N$

estimate $\{R_{jm}\}_{j=1}^{J}$ *using the least squares criterion*

$\hat{c}_{jm} = \arg\min_{c_{jm}} \sum_{\mathbf{x}_{t-1} \in R_{jm}} L\left(y_t, F_{m-1}(\mathbf{x}_{t-1}) + c_{jm}\right), \qquad j = 1, \ldots, J$

$F_m(\mathbf{x}_{t-1}) = F_{m-1}(\mathbf{x}_{t-1}) + \upsilon \sum_{j=1}^{J} \hat{c}_{jm} I(\mathbf{x}_{t-1} \in R_{jm})$

end for

Algorithm 3.1 also illustrates the tuning parameters related to the gradient boosting method. The amount of iterations M and the learning rate $\upsilon \in]0, 1]$ control the learning process. Setting M too low can result in underfitting whereas too many repeats can lead to overfitting. Setting the learning rate smaller than one can be seen as a shrinkage strategy as the parameter υ shrinks each functional estimate towards zero and thereby controls the speed of the learning process. These two parameters are inversely related to each other. A smaller learning rate usually requires more trees to be built (Friedman, 2001).

The amount of complexity related to the J-terminal node regression tree base learner function can be controlled by the amount of terminal nodes J and the amount of observations required at each terminal node region. Requiring more observations in each terminal node region narrows down the amount of potential split points and therefore controls the complexity of each tree. Building larger trees with more terminal nodes results in more complex models but the risk of overfitting also grows.

Note from the graphical illustration in Figure 3.2 that in order to build a J-terminal node regression tree $J - 1$ split points are needed and the size of the regression tree also controls the amount of interactions allowed between different predictors. Instead of requiring the exact amount of terminal nodes J some software implementations use the depth of the tree D as a tuning parameter. The depth of the regression tree is the maximum amount of inner nodes between the root and leaf nodes. The depth of the regression tree in Figure 3.2 for example is two since there are two split points between the root and each leaf node.

Different subsampling strategies can also be used for regularization with the gradient boosting model. The subsampling is usually done row-wise, where only a certain fraction η_{row} of training samples are used when estimating the parameters of the base learner function at each round of the algorithm. By using row-wise subsampling the regression trees at each round tend to be less similar. Additionally column-wise subsampling is also available, where only a certain fraction η_{col} of the available predictors are used at each round of the gradient boosting algorithm. The exact amount of subsampling used both row-wise and column-wise are finetuned using cross-validation.

Random forest

The random forest algorithm of Breiman (2001) has a close connection to both bagging and the adaboost classification algorithm. The final model with each of these three methods is an ensemble of simple models. The original idea of random forest is to improve the classification ability of bagging by reducing the correlation between each component in the final ensemble. This is done by injecting additional randomness when building each component of the final model.

Similarly as with boosting the base learner function used at each step of the random forest algorithm is a tree-based model. Unlike the regression tree presented in equation (3.8) the base learner with random forest classification algorithm is a classification tree. The graphical illustration given in Figure 3.2 holds for the classification tree as well, but now the functional estimate in each terminal node region of the J-terminal node classification tree is the predicted class

$$f(\mathbf{x}_{t-1}; \{C_j, R_j\}_{j=1}^{J}) = \sum_{j=1}^{J} C_j I(\mathbf{x}_{t-1} \in R_j), \tag{3.9}$$

where $\mathbf{x}_{t-1}$ is a vector of inputs at time $t-1$ and C_j is the predicted class in region R_j.

Instead of fixing the number of terminal nodes J as with gradient boosting the complexity of each tree in the random forest is typically controlled by requiring a certain number of observations at each terminal node. In the random forest algorithm the depth of each tree is increased by adding additional split points for as long as the number of observations in the terminal node is greater than a prespecified constant $n_{\min}$. This constant is a tuning parameter related to the random forest algorithm as all the terminal nodes must hold at least $n_{\min}$ data points. Especially with classification problems the trees in the random forest are often grown to the full size requiring only one observation in each terminal node region. Hastie, Tibshirani and Friedman (2009, p. 596) argue that letting the trees in the random forest to grow to the maximum size seldom costs much and results in one less tuning parameter.

The power of random forests comes from combining the predictions of many accurate individual trees that are as diverse as possible. In order to

make the trees in the random forest ensemble less correlated only a subset of features are considered when new split points are added to the classification tree. Suppose the number of features in the dataset is p then only m $(m \leq p)$ randomly chosen features are considered as candidates when selecting a new split point. The exact amount m depends on the problem at hand and is treated as a tuning parameter. Especially in problems where the proportion of relevant features in the whole feature set is small, setting m too low may result in poor performance (Hastie et al., 2009, p. 596).

Similarly as in bagging, a bootstrap sample Z^* of size N is drawn from the training data at each round $b \in \{1, ..., B\}$ of the algorithm. A new decision tree $f_b(\mathbf{x}_{t-1}, \mathbf{\Theta})$ is fit using this bootstrap sample, where the parameter vector $\mathbf{\Theta}$ holds the parameters of the decision tree presented in equation (3.9). The split points of this decision tree are found recursively by considering only the m randomly chosen features at each step. The decision tree is grown to the maximum possible size controlled by the parameter $n_{\min}$, which sets the minimum number of observations needed in each node of the tree. This process is summarized in Algorithm 3.2.

Algorithm 3.2 *Random forest classification*

for $b \leftarrow 1$ *to* B ***do:***
  draw a bootstrap sample $\mathbf{Z}^*$ *of size N from the training data*
  create an empty decision tree $f_b(\mathbf{x}_{t-1} \in \mathbf{Z}^*, \mathbf{\Theta} = \emptyset)$
  while *(the number of observations in some node* $> n_{\min}$*)* ***do****:*
    randomly select m variables
    pick the best split point among these
    split the current node into two daughter nodes
  end while
  include $f_b(\mathbf{x}_{t-1,}\mathbf{\Theta})$ *in the ensemble* F_b
end for

The final ensemble in the random forest algorithm is a combination of the individual trees found in each round b of the algorithm

$$F(\mathbf{x}_{t-1}) = \{f_b(\mathbf{x}_{t-1}, \mathbf{\Theta})\}_{b=1}^{B}.$$

The classification of a new data point $\mathbf{x}_N$ is based on a majority vote between the classifications induced by each individual tree

$$\hat{C}_{rf}(\mathbf{x}_N) = majority\ vote\{\hat{C}_b(\mathbf{x}_N)\}_{b=1}^{B},$$

where $\hat{C}_b(\mathbf{x}_N)$ is the predicted class given by the b_{th} decision tree in the random forest ensemble.

Neural networks

Neural networks were originally designed as a tool to model the information processing capabilities of the human brain and the earliest attempts go as far as the 1940s (Rojas, 1996). There are a vast amount of neural network models with different assumptions regarding the structure of the network and how information flows through the network. The model used in this research is one of the most commonly used neural network models called a single hidden layer feed-forward neural network (Bishop, 2006, p. 229).

The network consists of three layers which are typically named as the input layer, hidden layer and the output layer. Each layer in a feed-forward network is connected with the subsequent layer through weights as is visualized in Figure 3.3. The directed edges represent the weights and the direction of information flow in the network.

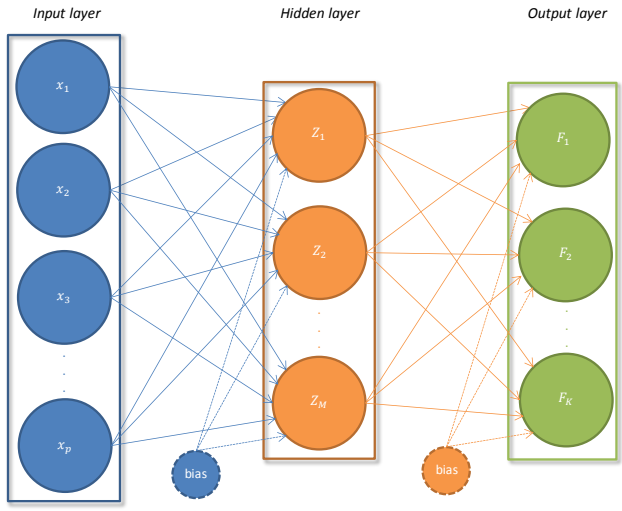

Figure 3.3: Artificial neural network

In general there could be several hidden layers creating a deeper and more complex network. Each unit in the hidden layer of Figure 3.3 is called a hidden unit since these are typically unobserved. These hidden units are linear combinations of the input variables $\mathbf{x} = (x_1, x_2, \ldots, x_p)^{'}$ followed by a non-linear activation function:

$$Z_m = h(\alpha_{0m} + \boldsymbol{\alpha}_m^{'}\mathbf{x}), \quad m = 1, \ldots, M, \tag{3.10}$$

where α_{0m} is the weight from the bias unit, $\boldsymbol{\alpha}_m$ is a $p \times 1$ vector of weights coming into hidden unit Z_m and $h(\cdot)$ is the activation function. The sigmoid function is typically used to transform the linear combinations of inputs into a non-linear form. Another common choice for the activation function is the hyperbolic tangent function. Note that the total amount of weights connecting the units in the input layer and the hidden layer is $M \times (p + 1)$, where M is the number of units in the hidden layer. The exact amount for M is treated as a tuning parameter of the model.

The final output for each class k is formed as a linear combination of the hidden units $\mathbf{Z} = (Z_1, Z_2, \ldots, Z_M)^{'}$, which is transformed in the interval $[0, 1]$ using the softmax function:

$$F_k = g(\beta_{0k} + \boldsymbol{\beta}_k^{'}\mathbf{Z}) = \frac{e^{\beta_{0k} + \boldsymbol{\beta}_k^{'}\mathbf{Z}}}{\sum_{l=1}^{K} e^{\beta_{0l} + \boldsymbol{\beta}_l^{'}\mathbf{Z}}}, \quad k = 1, \ldots, K.$$

Similarly as in equation (3.10) β_{0k} is the weight from the bias unit and $\boldsymbol{\beta}_k$ is a $M \times 1$ vector of weights connecting the units in the hidden layer to the output unit F_k.

The optimal weights in the network minimize the considered loss function. In the multinomial classification problem the loss function to be minimized is the sample counterpart of the multinomial deviance shown in equation (3.6). By denoting the complete set of weights in the network by a weight vector $\boldsymbol{\theta}$ the loss function can be written as

$$L(\boldsymbol{\theta}, F_k) = -\sum_{t=1}^{N}\sum_{k=1}^{K} I(R_t = k) \log F_k, \tag{3.11}$$

where F_k is the output for class k. The set of weights in the network can

be searched using a gradient descent based method called backpropagation. In backpropagation the gradient of the loss function in equation (3.11) is constructed at each iteration. The weights in the network are then updated according to the direction given by the negative gradient. For a more detailed description of the backpropagation algorithm see e.g. Rojas (1996).

A simple regularization strategy called weight decay has been suggested to avoid overfitting while estimating the optimal weights in the network. In weight decay an additional penalty term, which penalizes large weights, is added to the loss function presented in equation (3.11)

$$\tilde{L}(\boldsymbol{\theta},F_k) = L(\boldsymbol{\theta},F_k) + \lambda J(\boldsymbol{\theta}),$$

where λ is the weight decay parameter. The penalization function $J(\boldsymbol{\theta})$ can take various forms. A common choice is to impose quadratic penalization, where $J(\boldsymbol{\theta}) = \boldsymbol{\theta}^{'}\boldsymbol{\theta}$ (Bishop, 2006, p. 256). Larger values for λ thereby shrink the weights towards zero unless traditional backpropagation reinforces the weights. The exact amount of penalization needed is finetuned using cross-validation.

Support vector machines

The support vector machines originally presented by Vapnik (1995) can be used for both classification and regression problems. The basic idea and the terminology of support vector machines can be illustrated using a two-class classification problem with a linear decision boundary. The left-hand side of Figure 3.4 illustrates the case with perfect separability. The right panel in Figure 3.4 shows the nonseparable case, where some data points are misclassified by the linear decision boundary.

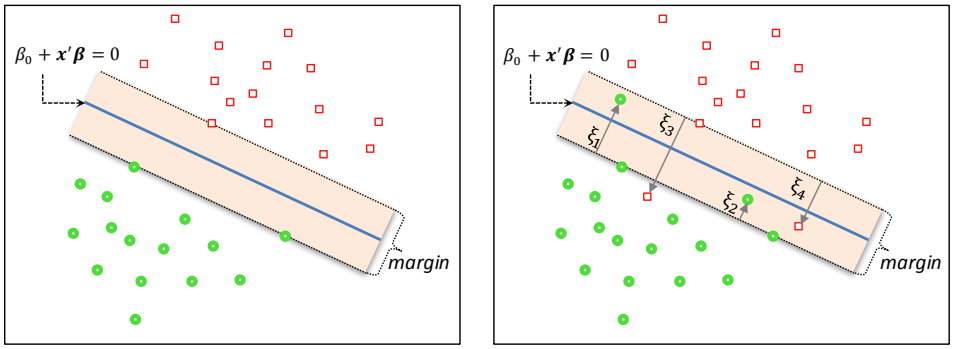

Figure 3.4: Support vector machine

The solid blue line in Figure 3.4 is the decision boundary separating the two classes

$$F(\mathbf{x}) = \beta_0 + \mathbf{x}^{'}\boldsymbol{\beta} = 0,$$

where β_0 is a constant term, $\boldsymbol{\beta}$ is a unit vector and $\mathbf{x}$ is a $p \times 1$ vector of input variables. For notational reasons let us focus on the binary classification case and denote the binary response as $y_i \in \{-1, 1\}, i = 1, ..., N$, where i can be associated to time t. The one-against-one method used in this research for K-class classification with support vector machines is a direct extension to the binary case as the final classification is based on a voting scheme between the $K(K-1)/2$ binary classifiers constructed for each class pair. For more information on the multiclass classification with support vector machines see e.g. Hsu and Lin (2002).

The goal with support vector machines is to find the decision boundary with maximum area on both sides of the boundary also known as the margin $M = \frac{2}{\|\boldsymbol{\beta}\|}$. In Figure 3.4 the dashed lines illustrate the margin and the points located at the dashed lines are known as support vectors. Hastie et al. (2009, p. 132) show that instead of maximizing the margin the optimization problem can be written in terms of minimizing $\|\boldsymbol{\beta}\|$

$$\min_{\beta_0, \boldsymbol{\beta}} \|\boldsymbol{\beta}\| \quad s.t. \;\; y_i(\beta_0 + \mathbf{x}_i^{'}\boldsymbol{\beta}) \geq 1, \quad i = 1, \ldots, N. \tag{3.12}$$

The constraint in equation (3.12) requires each observation to be on the right side of the margin. This constraint does not hold for the nonseparable case shown on the right panel of Figure 3.4. For this reason we need to define

a vector of slack variables $\boldsymbol{\xi} = (\xi_1, ..., \xi_N)$ and the minimization problem becomes

$$\min_{\beta_0, \boldsymbol{\beta}} \|\boldsymbol{\beta}\| \quad s.t. \begin{cases} y_i(\beta_0 + \mathbf{x}_i^{'}\boldsymbol{\beta}) \geq 1 - \xi_i, \quad i = 1, \ldots, N, \\ \xi_i \geq 0, \quad \sum_{i=1}^{N} \xi_i \leq \text{constant}. \end{cases} \tag{3.13}$$

The convex optimization problem with quadratic objective and linear inequality constraints in equation (3.13) can be solved using quadratic programming. The Lagrange primal function can be written as

$$L_P = \frac{1}{2} \|\boldsymbol{\beta}\|^2 + C \sum_{i=1}^{N} \xi_i - \sum_{i=1}^{N} \alpha_i [y_i(\beta_0 + \mathbf{x}_i^{'}\boldsymbol{\beta}) - (1 - \xi_i)] - \sum_{i=1}^{N} \nu_i \xi_i, \tag{3.14}$$

where C is now inplace of the predetermined constant in equation (3.13). The parameters α_i and ν_i are the Lagrange multipliers. The cost parameter C is a tuning parameter of the procedure and controls how wide the margin is. A larger value of C puts more emphasis on the points near the decision boundary and requires a tighter margin.

To consider non-linear decision boundaries the original input feature space is typically transformed into an enlarged space using e.g. polynomials or splines since the data could be linearly separable in this higher dimensional feature space. Without specifying the exact transformation the Lagrange dual objective function can be written using these transformed feature vectors $h(\mathbf{x}_i)$

$$L_D = \sum_{i=1}^{N} \alpha_i - \frac{1}{2} \sum_{i=1}^{N} \sum_{j=1}^{N} \alpha_i \alpha_j y_i y_j \langle h(\mathbf{x}_i), h(\mathbf{x}_j) \rangle, \tag{3.15}$$

where alphas are the Lagrange multipliers and $\langle h(\mathbf{x}_i), h(\mathbf{x}_j) \rangle$ is the inner product of the transformed input vectors i and j. The solution to the dual Lagrangian in equation (3.15) depends on the transformed higher dimensional data only through inner products. Instead of the exact transformation $h(\cdot)$ a kernel function, which computes inner products in the transformed space, is sufficient. A radial basis function and a d_{th}-degree polynomial are typical choices for the kernel function. The radial basis function can be written as

$$K(\mathbf{x}, \mathbf{x}_i) = \langle h(\mathbf{x}), h(\mathbf{x}_i) \rangle = \exp(-\gamma ||\mathbf{x} - \mathbf{x}_i||^2), \tag{3.16}$$

where γ is a tuning parameter related to the radial basis function kernel. The d_{th}-degree polynomial kernel function involves one extra tuning parameter compared to the radial basis kernel presented in equation (3.16). The degree of the polynomial d needs to be finetuned in addition to a scale parameter s

$$K(\mathbf{x}, \mathbf{x}_i) = (1 + s\langle \boldsymbol{x}, \boldsymbol{x}_i \rangle)^d. \tag{3.17}$$

The final classification in the support vector machine is produced by the following equation

$$\hat{G}(\mathbf{x}) = sign\big(\hat{F}(\mathbf{x})\big) = sign\big(\sum_{i=1}^{N} \hat{\alpha}_i y_i K(\mathbf{x}, \mathbf{x}_i) + \hat{\beta}_0\big),$$

where $K(\mathbf{x}, \mathbf{x}_i)$ is one of the kernel functions presented in equations (3.16) and (3.17). $\hat{\alpha}_i$ and $\hat{\beta}_0$ are the solved coefficients from the optimization problem. The coefficients $\hat{\alpha}_i$ are non-zero only for the data points marked as support vectors.

3.3 Data and model setup

3.3.1 Data

The dataset used in the empirical section of this paper covers daily returns of the S&P 500 stock market index from the beginning of the 1990s to the end of 2018.[2] The goal of the empirical analysis is to study a wide spectrum of different variables that could be used for prediction using the maximum amount of daily data available. With such a high frequency as daily returns the potential predictor variables are mostly based on different types of financial market data.

Technical analysis indicators are a common choice for the input variables of different machine learning methods (see e.g., Kim, 2003; Basak et al., 2019). In technical analysis different types of indicators are constructed using the historical price or return information from the stock market. Benchmark yields and different interest rate spreads from the corporate and government bond markets have also been extensively studied with both daily and monthly

[2] After constructing the predictor variables the exact time period is 12.2.1990 - 5.10.2018.

data (see e.g., Zhong and Enke, 2017; Nyberg and Pönkä, 2016). Interest rates express the tightness of the monetary policy set by the Federal Reserve. Different types of interest rate spreads reflect market expectations regarding the upcoming economic activity for example.

Lagged stock returns and returns from other stock markets are another commonly used alternative (see e.g., Zhong and Enke, 2017). A less studied predictor group is the volatility in different markets. A recent study by Becker and Leschinski (2018) shows that the VIX-index, which is often called the fear factor of stock markets, can also be a viable alternative. As with Zhong and Enke (2017) different exchange rates and commodity indices are also considered as predictive features (predictors) in this study. The appreciation (or depreciation) of the dollar relative to other currencies affects the foreign trade and international flow of funds to the U.S. Variables related to the state of the macroeconomy are found to be important predictors when predicting monthly stock returns (Nyberg, 2011). Unfortunately the majority of the macroeconomic information is not available with daily frequency.

Table 3.1 summarizes the input variables using seven different categories. A short description and an illustrative example are shown from each category. The full predictor set and the exact transformations for each predictor can be found in Appendix A.

Table 3.1: Predictor groups

Group	Description	Example
Stock market	S&P 500 price information, Returns from other stock markets	Lagged returns, Returns from DAX or FTSE
Interest rates	Government and corporate benchmark yields	3-month T-bill, Term spread
Exchange rates	The appreciation of dollar relative to other currencies	Dollar/British Pound, major currencies index
Commodities	Information from the commodities market	Copper, Oil, Gold, Silver
Volatility	Volatility in the stock and bond markets	VIX-index, MOVE-index
Technical analysis	Indicators derived from price or return information	Relative strength index
Macro	Information regarding the macroeconomy	ADS-index

The total amount of different predictors studied in this research is 37. Following the approach of Krauss et al. (2017) various lag lengths of the predictors are also considered. Lag lengths beyond ten trading days are found to be uninformative in the preliminary analysis using the model selection capability of the gradient boosting machine.[3] By considering the lagged predictors from the previous ten trading days the full predictor set consists of 370 different inputs (37×10). Only data points (days) with information available for each predictor are considered. For this reason predictor variables utilizing market data outside the U.S. is kept minimal because of the individual holiday periods in each country. After leaving out the data points with missing values the final dataset includes 6686 daily observations.

From the machine learning methods considered in this study only the

[3] Results available upon request.

tree-based classifiers gradient boosting machine (GBM) and random forest (RF) are capable of handling such a large predictor set. Both of these methods are able to perform model selection simultaneously with estimation as new split points are introduced for the tree-based base learners. Support vector machines (SVM), neural networks (ANN) and k-nearest neighbor (k-NN) end up easily overfitting the data with a large predictor set and therefore a reduced dataset is needed. A smaller predictor set is built using a combination of prior knowledge and the results from the tree-based methods.

Both GBM and random forest select the VIX-index as the single most influential predictor.[4] For the random forest model each of the ten most influential inputs are different lag lengths of the VIX-index, whereas GBM includes six lags of VIX in the top-10. Different technical analysis indicators are also considered as important predictors by both models. The best performing technical analysis indicators are the stochastic oscillator (StochK), moving average convergence divergence (MACD) and Williams %R (Rperc)[5]. The ranking of these indicators are slightly different for the two methods.

In addition to these the spread between the daily high and low stock prices is selected by both models. GBM also ranks the corporate interest rate spread among the ten most influential predictors. The results from the principal component analysis by Zhong and Enke (2017) are in favor of using the lagged stock returns and international stock returns. Lagged stock returns from the S&P 500 and the returns from the German stock index are thereby also included in the reduced dataset.

The reduced dataset includes six inputs (predictors) which are the VIX-index, MACD-indicator, spread between daily high and low prices, corporate interest rate spread, lagged stock returns and returns from the DAX-index. The data from the previous three trading days are used for each predictor. The total amount of inputs in the reduced dataset is thereby 18 (6×3). Several other choices for both the composition of predictors and lag lengths were also considered.

[4] More detailed model selection results can be found in Appendix B.
[5] See Appendix A for further details about the indicators.

3.3.2 Tuning parameter optimization

Each of the machine learning methods considered involve free parameters that affect the final output of the model. Often the parameters can affect the results quite dramatically as is the case with neural networks for example (Zhang, Patuwo and Hu, 1998). These parameters are usually called tuning parameters since it is up to the end-user to finetune the optimal parameters for the particular learning task.

Table 3.2 summarizes the tuning parameters of the machine learning methods studied in this research. A brief description and the notation used for the tuning parameter in the methodological part of this paper is shown in Table 3.2. The last column illustrates the considered parameter values. It should be noted that for each method a wider grid search has been conducted in order to find a suitable range for each parameter. Only this smaller interval is depicted in Table 3.2.

Table 3.2: Tuning parameters for each method

Method	Description	Notation	Values
k-NN	Number of neighbors	k	$1, 11, 21, ..., 461$
GBM	Number of iterations	M	$1, ..., 1000$
	Tree depth	D	$1, 2, 3$
	Fraction of training points	η_{row}	$0.5, 0.7, 0.9$
	Fraction of predictors	η_{col}	$0.7, 0.9$
RF	Number of trees	B	$100, 300, 500$
	Number of predictors	m	$10, 50, 100, 200, 300$
	Observations in each node	$n_{\min}$	$10, 50, 100, 200, ..., 600$
ANN	Number of hidden units	M	$3, 5, 8, 10, 12, 15, 20$
	Weight decay	λ	$0.1, 0.2, 0.4, 0.6, 0.8, 1$
SVM	Cost parameter	C	$0.01, 0.1, 0.2, ...9$
	Radial kernel parameter	γ	$0.005, 0.01$
	Polynomial kernel, scale	s	$0.005, 0.01$
	Polynomial kernel, degree	d	$2, 3$

Because the number of different machine learning methods considered in this paper is quite large some simplifications have been done in order to

keep the parameter search feasible. Additional finetuning would be available for several methods. There are for example alternative distance measures for the k-NN method and different learning algorithms for the ANN model. Following the approach of Friedman (2001) the learning rate in the gradient boosting machine algorithm is set as small as possible. The learning rate is held fixed at a value of 0.001. For computational reasons the parameter search is restricted to the parameter values presented in Table 3.2.

The performance of each model specification is evaluated using the validation accuracy produced by the L-fold cross-validation procedure. In L-fold cross-validation the training sample is split into L independent folds. Each of these L folds is used as a hold-out test set once, while the remaining $L-1$ folds are used for estimating the model. This process is repeated for each fold and the validation accuracy is the average accuracy produced by the L independent folds

$$CV_{Acc} = \frac{1}{L}\sum_{l=1}^{L} Acc_l,$$

where L is the amount of folds and Acc_l is the obtained accuracy when the data points of fold l are used as an independent test set.

10-fold cross-validation is used to estimate the optimal tuning parameters for each method. Table 3.3 shows the parameter optimization results and depicts the optimal tuning parameter combination for each method.

Table 3.3: Final model specifications

Method	**Parameters**
k-NN	$k = 331$
GBM	$M = 931, D = 2, \eta_{\text{row}} = 0.9, \eta_{\text{col}} = 0.7$
RF	$B = 500, m = 50, n_{\min} = 300$
ANN	$M = 5, \lambda = 0.8$
SVM	$d = 3, s = 0.005, C = 6.9$

Relatively restricted models are chosen in the cross-validation procedure as can be seen in Table 3.3. This is not very surprising as the noisy stock market data combined with a complex model can easily lead to overlearning

the training data. The limitations in allowed flexibility can be seen for each method. For example, quite a large number of nearest neighbors are used while classifying each data point and the tree-based methods GBM and RF seem to favor quite shallow trees. A fairly small amount of neurons are used in the hidden layer of an ANN model combined with quite a heavy penalization through the weight decay parameter. The polynomial kernel of degree 3 is used for the support vector machine and the cost parameter is rather small, which results in a wider margin and also supports the finding of restricted models.

The low amount of hidden neurons reported in Table 3.3 for the ANN is of similar magnitude as the parameter value selected using trial-and-error in Zhong and Enke (2017). The results from a tuning parameter optimization procedure in Kara, Boyacioglu and Baykan (2011) also favor the use of a polynomial kernel over the radial basis function for the SVM. The polynomial kernel of degree three is the optimal choice when forecasting the direction of the Turkish stock market as well (Kara et al., 2011). Based on prior knowledge Krauss et al. (2017) end up with the same tree depth for the GBM model as reported in Table 3.3.

3.4 Empirical results

3.4.1 Statistical predictive performance

The complete data sample covering the time period from 12.2.1990 to 5.10.2018 is split into two parts. The training set contains data before the year 2007 and is used for training and validating the models. The test set covering the rest of the data is used as an independent test set to evaluate how well each method performs on a completely unseen dataset. Therefore roughly 58 percent of the complete dataset is used for training and the remaining 42 percent for testing. The test set thus contains 2787 daily observations.

Figure 3.5 visualizes the daily returns of the S&P 500 index. The horizontal dashed lines show the upper and lower quartiles of returns, which are used as the fixed thresholds in equation (3.3) to create the multinomial response variable. The vertical gray dashed line illustrates the split into training and testing datasets.

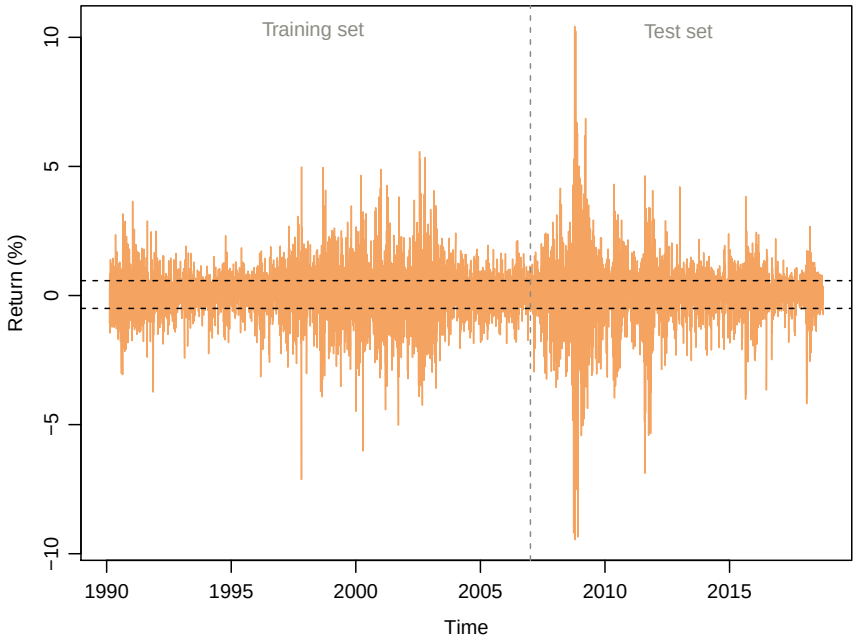

Figure 3.5: Daily returns of the S&P 500 index

In order to keep the test set completely independent, the upper and lower quartiles of returns are calculated using only the training data.[6] The benchmark accuracy for the training and validation is one half as the majority class contains fifty percent of the observations. In the test set there are slightly more observations coming from the majority class and the strategy of always predicting the majority class yields the accuracy of 0.526. This should be used as the benchmark accuracy when evaluating the prediction results for the test set.

The prediction results for each machine learning method are presented in Table 3.4. The first column shows the considered method while the next three columns give the classification accuracies for the training, validation and test sets.

[6] The upper and lower quartiles for the training set are $\{-0.4989\%, 0.5755\%\}$ whereas the quartiles for the entire dataset are $\{-0.4656\%, 0.5723\%\}$.

Table 3.4: Prediction results

Method	Train	Validation	Test
k-NN	0.5204	0.5194	0.5457
GBM	0.5560	0.5294	0.5597
RF	0.6194	0.5304	0.5536
ANN	0.5247	0.5224	0.5558
SVM	0.5463	0.5224	0.5504

The results indicate significant return forecastability as the classification accuracies for the training and validation sets are well above the benchmark of one half for each method. In terms of the training and validation accuracies the results are in favor of using the tree-based methods gradient boosting and random forest, which can utilize the full predictor set. The relatively high training accuracy for the random forest model stands out from the rest, however the validation accuracy is only slightly higher than for GBM. ANN and SVM have a similar validation performance. The simplicity of the nearest neighbor algorithm has led to the lowest training and validation accuracies.

The classification results for the test set indicate how well the models generalize to new data. The conclusion of return predictability seems to hold even when testing on an unseen dataset as the accuracies for each method are well above the test set benchmark performance. Overall, the level of predictability reported in Table 3.4 is in line with recent literature on sign predictability of daily returns (see e.g., Krauss et al., 2017; Zhong and Enke, 2017).

The ranking between the machine learning methods based on the classification accuracies for the test set is slightly different from the ranking obtained using the accuracies for the training set. While GBM still outperforms the other machine learning methods random forest reaches only the third highest test accuracy as neural networks show better generalization ability. The fairly large deviance between the training and testing results for the random forest raises questions of potential overfitting. A similar concluding remark can be made when comparing the generalization capabilities of ANN and SVM. The SVM model has better training accuracy but results in slightly lower

generalization performance.

3.4.2 Economic predictive performance

Leitch and Tanner (1991) argue that a model performing well from a statistical point of view does not necessarily imply economic profitability, especially when trading costs are taken into account. In order to evaluate the ability to gain economic profits, in this section a real-life trading simulation is conducted. Our trading simulation is similar to those in Pesaran and Timmermann (1995) and Leung et al. (2000) for example. The classification patterns are turned into a trading strategy, which depends on the current $(\hat{R}_{t+1})$ and previous forecasted class $(\hat{R}_t)$, as can be seen in Table 3.5.

Table 3.5: Trading strategy

		$\hat{R}_t$		
		1	2	3
$\hat{R}_{t+1}$	1	Stay out	Sell	Sell
	2	Buy	Hold	Hold
	3	Buy	Hold	Hold

Table 3.5 shows how the multinomial response variable presented in equation (3.3) enables a richer set of possible trading strategies compared to the more commonly studied binary response case. Here the possibility to keep the position unchanged (Hold) is used to reduce excessive trading activity. In a traditional market timing setup as presented in Pesaran and Timmermann (1995) an asset allocation decision is made between investing in stocks or in bonds. With the daily trading frequency the returns from investing in the bond market are fairly low and we only consider the options of staying fully invested in stocks or not.

The asset allocation decision between stocks and bonds could easily be incorporated to the trading strategy in Table 3.5. The complexity of the asset allocation strategy based on the multinomial response could be increased even further. Depending on the predicted class one could allocate 0,100 or say 70 percent of the wealth in stocks and the remaining in the bond market. These more complex strategies are left for further research.

In the benchmark buy-and-hold strategy the entire initial wealth is invested in stocks at the beginning of the period and sold at the end of the period. To get a fair comparison with the buy-and-hold strategy additional wealth can not be invested in stocks. This limits the ability to benefit from large positive returns. In reality the investor would certainly like to increase the amount of wealth invested in stocks if large positive stock returns are expected.

Instead of just staying out from the market it is also possible to profit from the large negative price changes through shortselling. Allowing for shortselling as in Becker and Leschinski (2018) can be seen problematic for several reasons. The risks involved in shortselling are large as the possible losses are basically unlimited. The potential restrictions placed on shortselling by the Securities and Exchange comission (SEC) during high market turmoil can also be seen problematic. One such event took place during the financial crisis as shortselling restrictions were imposed on financial companies (Becker and Leschinski, 2018). Although such restrictions are less of an issue when considering investments in a major stock index the main analysis on economic profitability is conducted without shortselling.

The trading cost is assumed to be a fixed percentage rate, which has been a common choice in the previous literature (see e.g., Pesaran and Timmermann, 1995; Fiévet and Sornette, 2018). A trading cost of 0.1 percent is used in this study. This could be regarded as relatively high since an active individual investor can achieve much lower costs when trading the stocks of a private company instead of the index. Becker and Leschinski (2018) show that the average bid-ask spread on individual stocks in the U.S during the period of 2004-2017 is around 0.05 percent. This is also the level of transaction costs used by Fiévet and Sornette (2018). Naturally the average bid-ask spreads are much lower for the more liquid exchange traded funds (ETF) following the S&P 500 (Hsu, Hsu and Kuan, 2010). Over the past year the bid-ask spread of the worlds largest ETF has ranged between 0.003 and 0.005 percent.[7]

The results from the trading simulation are presented in Table 3.6. The first column shows the considered machine learning method. The second column illustrates the final wealth level in dollars after following the trading strategy presented in Table 3.5. The initial wealth of 100 is grown to a final wealth of

[7] See https://www.etf.com/SPY for more information on the oldest ETF following the S&P 500.

195.86 using the benchmark buy-and-hold strategy.

Table 3.6: Results from a trading simulation

Method	**Trading**
k-NN	202.95
GBM	351.54
RF	225.15
ANN	244.51
SVM	210.22

All the considered methods beat the buy-and-hold benchmark. The ranking between the machine learning methods remain the same as based on the test set accuracies. The worst performing nearest neighbor algorithm produces a final wealth that is only slightly higher than the benchmark. The wealth levels of SVM and random forest are also fairly modest compared to the benchmark. The best performing methods based on the test set accuracy outperform the benchmark quite substantially. The final wealth produced by the neural network model is 25 percent higher than the benchmark. The final wealth based on the predictions of the gradient boosting machine yields a final wealth 80 percent higher than the benchmark.

If we allow for shortselling to be used as a tool to profit from the correctly predicted negative returns the final wealth gained using the predictions from a GBM model is 181 percent higher than the benchmark. In shortselling the assets sold short are borrowed from a broker and then returned when closing the short position. It should however be noted that because of these borrowing costs the actual trading costs involved with shortselling can be higher than the considered level of 0.1 percent. The results based on shortselling can therefore be overestimated. We take the approach of Becker and Leschinski (2018) and ignore these additional costs related to shortselling.

In order to see how the wealth is accumulated throughout the test period Figure 3.6 shows the wealth patterns for the benchmark buy-and-hold strategy and for each of the machine learning methods considered. Note how each method starts with the initial wealth of 100 dollars and ends up with the final

wealth level reported in Table 3.6.

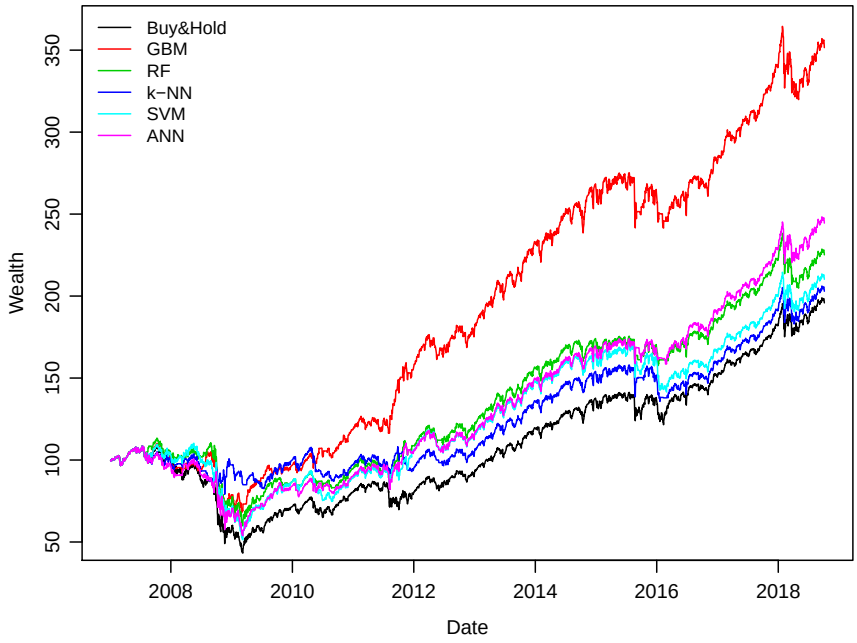

Figure 3.6: Trading simulation results for each method

There are certain periods of time when the wealth produced by the machine learning methods deviate from the benchmark. All the machine learning methods are able to avoid some of the heavy losses involved in the financial crisis. Some do so better than others as k-NN can avoid majority of the losses while SVM is quite close to the benchmark. Another such period is the debt crisis in the Euro zone, which took place between the years 2010 and 2012. In the year 2010 the wealth level of the GBM model starts to deviate from the rest of the field. A similar observation can be made in the end of the year 2011.

To have a closer look at the daily returns during these events Figure 3.7 illustrates the correctly predicted large positive and negative returns for the GBM model.

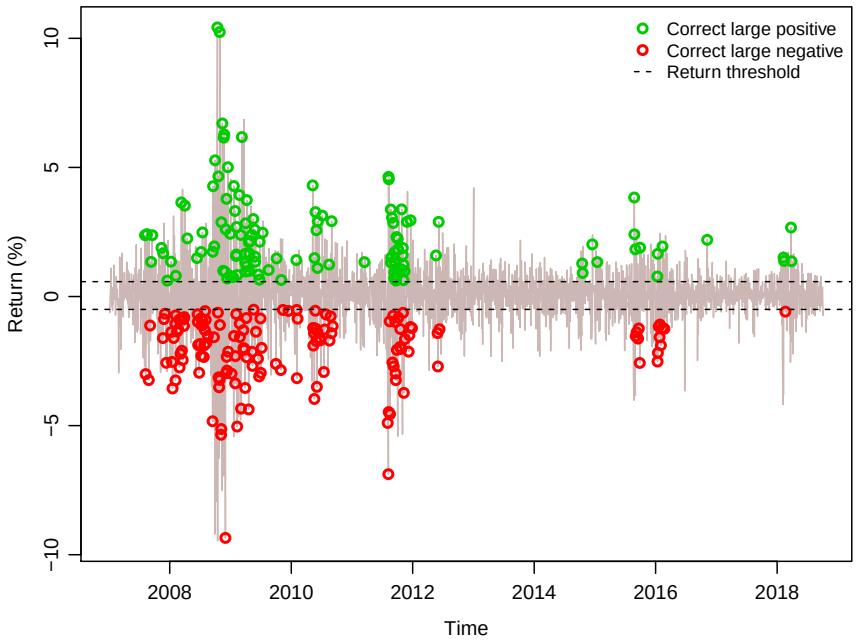

Figure 3.7: Correctly predicted large absolute returns for the GBM model

Several interesting observations can be noted from Figure 3.7. First of all the predictability of large positive and negative stock returns seem to cluster together. The financial crisis and the European debt crisis are periods of higher predictability. This is in line with the results of Krauss et al. (2017) and Fiévet and Sornette (2018). These were also the periods, where the wealth levels of the machine learning methods started to deviate from the benchmark, as was seen in Figure 3.6. Some predictability is also observed at the beginning of year 2016 when the low oil prices caused worries in the market. All of these events involve high volatility.

Figure 3.7 also illustrates how the individual correct predictions for the large positive and negative returns can be of very different magnitude. There are correct predictions with return level right above or below the fixed thresholds depicted using the black dashed lines in Figure 3.7. On the other hand there are positive and negative daily returns close to ten percent. Naturally in terms of the profits obtained from the trading simulation the further apart the correctly predicted observation is from the fixed threshold levels the better. And vice versa for the incorrectly classified observations.

Regarding the classification accuracy all the correctly predicted observa-

tions are considered equally important and thus receive the same weight. This could be a possible explanation why the final wealth levels reported in Table 3.6 and further illustrated in Figure 3.6 deviate quite substantially between the different methods. The trading strategy and the return levels could be more closely incorporated to the actual model estimation process. The level of returns or the preferences regarding different correctly and incorrectly predicted classes could be taken into account using caseweights for example. These alternative approaches are left for further research.

It is also interesting to see that despite the daily trading frequency the trading strategy presented in Table 3.5 could be considered relatively passive. Higher trading activity is observed only during short periods of time involving high volatility. Becker and Leschinski (2018) argue that the assumed fixed trading cost can overestimate the gained profits as the actual bid-ask spreads tend to rise during high market turmoil. The highest bid-ask spread observed by Becker and Leschinski (2018) is around 0.2 percent during a short period of time in the financial crisis. As a sanity check the final wealth gained using the predictions from the GBM model and the trading cost of 0.2 percent throughout the whole period is still 40 percent higher than the final wealth with the benchmark strategy. The maximum cost level yielding the same final wealth as with the buy-and-hold strategy is 0.33 percent. This is significantly higher compared to the 0.21 percent break even cost reported by Fiévet and Sornette (2018).

3.5 Conclusions

This paper introduces a new multinomial classification approach to predict daily stock returns of the S&P 500 stock market index. The multinomial approach puts more emphasis on predicting large absolute stock returns instead of the noisy variation around zero. The multinomial approach also provides a larger set of possible trading strategies compared to the more commonly used binary response variable. The classification ability of five different machine learning methods are compared both from the viewpoints of classification accuracy and from the ability to generate economic profits in a real-life trading simulation.

The empirical results show how the gradient boosting model is the top-

performer among the machine learning methods based on the classification accuracies for both the validation and test sets. The model selection capability of the gradient boosting model also provides important information about the useful predictor variables. The volatility in the stock market as measured by the VIX-index turns out to be the best single predictor. Several technical analysis indicators are also useful when predicting multinomial stock returns.

The validity of the efficient market hypothesis (EMH) is typically tested in a real-life trading simulation. The ability to generate economic profits beyond the passive buy-and-hold strategy when transaction costs are taken into account is seen as a violation of the EMH. All the machine learning methods considered in this research are able to beat the benchmark buy-and-hold strategy after accounting for the transaction cost of 0.1 percent. The best performing gradient boosting model produces returns 80 percent higher than the buy-and-hold strategy. The predictability is highest during the market turmoil of the financial crisis and the European debt crisis, which is in line with recent literature.

The current research can be extended in several directions. Now the two fixed return thresholds used to create the multinomial response variable were based on the upper and lower quartiles of the return series. This choice was based on creating well balanced classes, which simplify the comparison between different machine learning methods. Several other choices are also possible and are left for further research. Setting the return thresholds further away from zero may result in increased statistical predictability but the amount of predictions for the large absolute returns decrease. Thereby the trading strategy would become increasingly passive and hence there might be no additional economic value despite statistically superior predictions over the ones presented in this research.

There are also various alternative trading strategies that could be used to assess the economic profits generated by different methods. One could for example benefit more from the predictions indicating large positive or negative returns by shortselling or by increasing the wealth invested. Alternatively modern financial products such as the bull and bear certificates could be used to exploit the correctly predicted large absolute returns. Furthermore, the linkage between forecastability based on statistical evaluation criteria and the economic profitability of the trading strategy should be more closely examined.

The trading strategy could even be incorporated to the actual model estimation process.

References

Andrada-Félix, J., Fernández-Rodríguez, F., and Fuertes, A.-M. (2016). Combining nearest neighbor predictions and model-based predictions of realized variance: Does it pay? *International Journal of Forecasting*, 32(3):695 – 715.

Ang, A. and Chen, J. (2002). Asymmetric correlations of equity portfolios. *Journal of Financial Economics*, 63(3):443–494.

Arroyo, J. and Maté, C. (2009). Forecasting histogram time series with k-nearest neighbours methods. *International Journal of Forecasting*, 25(1):192 – 207.

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

Becker, J. and Leschinski, C. (2018). Directional Predictability of Daily Stock Returns. Hannover Economic Papers (HEP) dp-624, Leibniz Universität Hannover, Wirtschaftswissenschaftliche Fakultät.

Bishop, C. M. (2006). *Pattern Recognition and Machine Learning (Information Science and Statistics)*. Springer-Verlag, Berlin, Heidelberg.

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

Christoffersen, P. F. and Diebold, F. X. (2006). Financial asset returns, direction-of-change forecasting, and volatility dynamics. *Management Science*, 52(8):1273–1287.

Chung, J. and Hong, Y. (2007). Model-free evaluation of directional predictability in foreign exchange markets. *Journal of Applied Econometrics*, 22(5):855–889.

Cujean, J. and Hasler, M. (2017). Why does return predictability concentrate in bad times? *The Journal of Finance*, 72(6):2717–2758.

Fama, E. F. (1970). Efficient capital markets: A review of theory and empirical work. *The Journal of Finance*, 25(2):383–417.

Fiévet, L. and Sornette, D. (2018). Decision trees unearth return sign predictability in the s&p 500. *Quantitative Finance*, pages 1–18.

Fix, E. and Hodges, J. (1951). *Discriminatory Analysis: Nonparametric Discrimination: Consistency Properties*. USAF School of Aviation Medicine, Randolph Field, TX.

Freund, Y. and Schapire, R. E. (1996). Experiments with a new boosting algorithm. In *Proceedings of the Thirteenth International Conference on International Conference on Machine Learning*, ICML'96, pages 148–156, San Francisco, CA, USA. Morgan Kaufmann Publishers Inc.

Friedman, J., Hastie, T., and Tibshirani, R. (2000). Additive logistic regression: A statistical view of boosting. *The Annals of Statistics*, 28:337–407.

Friedman, J. H. (2001). Greedy function approximation: A gradient boosting machine. *Annals of Statistics*, 29(5):1189–1232.

Granger, C. W. and Ding, Z. (1996). Varieties of long memory models. *Journal of Econometrics*, 73(1):61 – 77.

Hastie, T., Tibshirani, R., and Friedman, J. (2009). *The elements of statistical learning: data mining, inference and prediction*. Springer, 2 edition.

Henkel, S., Martin, J. S., and Nardari, F. (2011). Time-varying short-horizon predictability. *Journal of Financial Economics*, 99(3):560–580.

Hong, Y., Tu, J., and Zhou, G. (2007). Asymmetries in stock returns: Statistical tests and economic evaluation. *The Review of Financial Studies*, 20(5):1547–1581.

Hsu, C.-W. and Lin, C.-J. (2002). A comparison of methods for multiclass support vector machines. *IEEE Transactions on Neural Networks*, 13(2):415–425.

Hsu, P.-H., Hsu, Y.-C., and Kuan, C.-M. (2010). Testing the predictive ability of technical analysis using a new stepwise test without data snooping bias. *Journal of Empirical Finance*, 17(3):471 – 484.

Kara, Y., Boyacioglu, M. A., and Baykan, Ö. K. (2011). Predicting direction of stock price index movement using artificial neural networks and support vector machines: The sample of the istanbul stock exchange. *Expert Systems with Applications*, 38(5):5311 – 5319.

Karhunen, M. (2019). Algorithmic sign prediction and covariate selection across eleven international stock markets. *Expert Systems with Applications*, 115:256 – 263.

Kendall, M. G. (1953). The analysis of economic time series, part I: Prices. *Journal of the Royal Statistical Society*, 96.

Kim, K.-j. (2003). Financial time series forecasting using support vector machines. *Neurocomputing*, 55(1):307 – 319.

Krauss, C., Do, X. A., and Huck, N. (2017). Deep neural networks, gradient-boosted trees, random forests: Statistical arbitrage on the s&p 500. *European Journal of Operational Research*, 259(2):689 – 702.

Leitch, G. and Tanner, J. E. (1991). Economic forecast evaluation: Profits versus the conventional error measures. *The American Economic Review*, 81(3):580–590.

Leung, M. T., Daouk, H., and Chen, A.-S. (2000). Forecasting stock indices: a comparison of classification and level estimation models. *International Journal of Forecasting*, 16(2):173 – 190.

Linton, O. and Whang, Y.-J. (2007). The quantilogram: With an application to evaluating directional predictability. *Journal of Econometrics*, 141(1):250–282.

Longin, F. and Solnik, B. (2001). Extreme correlation of international equity markets. *The Journal of Finance*, 56(2):649–676.

Maheu, J. M. and McCurdy, T. H. (2004). News arrival, jump dynamics, and volatility components for individual stock returns. *The Journal of Finance*, 59(2):755–793.

Meade, N. (2002). A comparison of the accuracy of short term foreign exchange forecasting methods. *International Journal of Forecasting*, 18(1):67 – 83.

Merton, R. (1981). On market timing and investment performance. i. an equilibrium theory of value for market forecasts. *The Journal of Business*, 54:363–406.

Neely, C. J., Rapach, D. E., Tu, J., and Zhou, G. (2014). Forecasting the equity risk premium: The role of technical indicators. *Management Science*, 60(7):1772–1791.

Nyberg, H. (2011). Forecasting the direction of the us stock market with dynamic binary probit models. *International Journal of Forecasting*, 27:561–578.

Nyberg, H. and Pönkä, H. (2016). International sign predictability of stock returns: The role of the United States. *Economic Modelling*, 58(C):323–338.

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

Rojas, R. (1996). *Neural Networks: A Systematic Introduction*. Springer-Verlag New York, Inc., New York, NY, USA.

Skabar, A. (2013). Direction-of-change financial time series forecasting using a similarity-based classification model. *Journal of Forecasting*, 32(5):409–422.

Vapnik, V. N. (1995). *The Nature of Statistical Learning Theory*. Springer-Verlag, Berlin, Heidelberg.

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

Zhang, G., Patuwo, B. E., and Hu, M. Y. (1998). Forecasting with artificial neural networks:: The state of the art. *International Journal of Forecasting*, 14(1):35 – 62.

Zhong, X. and Enke, D. (2017). Forecasting daily stock market return using dimensionality reduction. *Expert Systems with Applications*, 67:126 – 139.

Appendix A: Full predictor set

Table 3.7: Full predictor set

Category	Predictor	Transformation
Stock market	Lagged return	$r_{sp500,t} = \log(\frac{P_{sp500,t}}{P_{sp500,t-1}})$
	DAX	$r_{dax,t} = \log(\frac{P_{dax,t}}{P_{dax,t-1}})$
	FTSE	$r_{ftse,t} = \log(\frac{P_{ftse,t}}{P_{ftse,t-1}})$
	High-Low	$hilow_t = P_{high,t} - P_{low,t}$
	Trade volume	$tvol_t = \log(\frac{Vol_t}{Vol_{t-1}})$
	Market capitalization	$mcap_t = \log(\frac{Cap_t}{Cap_{t-1}})$
	Lagged response	-
	Squared return	$r_t^2 = r_{sp500,t}^2$
	Skew	$skew_t = r_{sp500,t}^3$
	Kurtosis	$kurt_t = r_{sp500,t}^4$
Interest rates	Fed funds rate	$ff_t = fedf_t - fedf_{t-1}$
	3-mth Tbill	$3mth_t = 3m_t - 3m_{t-1}$
	10-yr bond	$10yr_t = 10y_t - 10y_{t-1}$
	30-yr bond	$30yr_t = 30y_t - 30y_{t-1}$
	Moody's Aaa	$Aaa_t = yAaa_t - yAaa_{t-1}$
	Moody's Baa	$Baa_t = yBaa_t - yBaa_{t-1}$
	Term spread	$ts_t = 10y_t - 3m_t$
	Long spread	$ls_t = 30y_t - 10y_t$
	Moody's spread	$corp_t = yBaa_t - yAaa_t$
	Corporate vs Government	$Aaa10y_t = yAaa_t - 10y_t$
	TED-spread	-

*Table is continued on the next page.

Table 3.7: Full predictor set (continued)

Category	Predictor	Transformation
Exchange rates	Major currencies index	$curr_t = \log(\frac{Major_t}{Major_{t-1}})$
	USD / GBP	$usdgbp_t = \log(\frac{\$to£_t}{\$to£_{t-1}})$
Commodities	Copper	$copper_t = \log(\frac{P_{cop,t}}{P_{cop,t-1}})$
	Oil	$oil_t = \log(\frac{P_{oil,t}}{P_{oil,t-1}})$
	Gold	$gold_t = \log(\frac{P_{gold,t}}{P_{gold,t-1}})$
	Silver	$silver_t = \log(\frac{P_{slver,t}}{P_{slver,t-1}})$
Volatility	VIX-index	-
	MOVE-index	-
Macro	ADS-index	-
Technical analysis	Moving average	$ma_t = n^{-1}\sum_{i=0}^{n-1} P_{sp500,t-i}$
	Momentum	$mom_t = P_{sp500,t} - P_{sp500,t-(n-1)}$
	Stochastic %K	$k_t = \frac{P_t - L_n}{H_n - L_n} \times 100$
	Stochastic %D	$d_t = n^{-1}\sum_{i=0}^{n-1} k_{t-1}$
	Relative strength index	$rsi_t = \frac{up}{up+down} \times 100$
	LW %R	$rperc_t = \frac{H_n - P_t}{H_n - L_n} \times 100$
	MACD	see Kara et al. (2011)

*For simplicity n is assumed to be 10 for all the indicators. $H_n$is the highest high in the previous n-1 days whereas L_n is the lowest low. up is the sum of positive price changes and low the sum of negative price changes in the previous n-1 days.

Appendix B: Model selection results of the tree-based methods

Table 3.8: Top-15 most important predictors for RF and GBM

	RF		GBM	
	Variable	Relative influence	Variable	Relative influence
1	VIX_1	100.000	VIX_1	100.000
2	VIX_2	83.542	VIX_2	64.060
3	VIX_5	63.202	VIX_7	21.815
4	VIX_3	59.166	VIX_3	18.444
5	VIX_4	53.447	AAA10y_1	13.928
6	VIX_6	53.221	StochK_1	13.351
7	VIX_8	37.275	VIX_5	13.039
8	VIX_7	33.654	VIX_4	12.232
9	VIX_10	32.415	HiLow_4	11.204
10	VIX_9	27.538	MACD_3	4.368
11	StochK_1	26.352	StochK_2	4.152
12	RPerc_1	23.745	RPerc_1	3.740
13	HiLow_4	23.323	MCap_5	3.085
14	HiLow_5	17.188	HiLow_10	2.635
15	HiLow_6	12.935	SP500_5	2.217

*For more information about the relative influence measure, see Breiman (2001) and Friedman (2001).

Chapter 4

Moving forward from predictive regressions: Boosting asset allocation decisions

Abstract * † ‡

We introduce a flexible utility-based empirical approach to directly determine asset allocation decisions between risky and risk-free assets. This is in contrast to the commonly used two-step approach where least squares optimal statistical equity premium predictions are first constructed to form portfolio weights before economic criteria are used to evaluate resulting portfolio performance. Our single-step customized gradient boosting method is specifically designed to find optimal portfolio weights in a direct utility maximization. Empirical results of the monthly U.S. data show the superiority of boosted portfolio weights over several benchmarks, generating interpretable results and profitable asset allocation decisions.

* This chapter is based on a manuscript written jointly with Henri Nyberg.

† We thank Mika Hannula, Heikki Kauppi, Matthijs Lof, Mika Vaihekoski and seminar participants at the University of Turku for useful comments. We gratefully acknowledge the financial support from the Emil Aaltonen Foundation and the Academy of Finland (grant 321968).

‡ A paper based on this chapter is available in SSRN working paper series (id3623956).

4.1 Introduction

Asset allocation decisions have always been at the heart of finance and asset pricing research. The focus in the existing empirical and econometric implementations has been to predict stock returns to generate profitable portfolio decisions (see, e.g., the survey of Brandt, 2010). These have fundamentally been two-step approaches where especially linear predictive regressions, optimized via ordinary least squares, are first used to predict stock returns and resulting predictions are subsequently utilized to generate portfolio weights and portfolio returns evaluated with economic goodness-of-fit criteria. However, Leitch and Tanner (1991), Kandel and Stambaugh (1996), Xu (2004), Guidolin and Timmermann (2007) and Cenesizoglu and Timmermann (2012), among others, have shown from various perspectives that statistically optimal return predictions do not automatically imply economic gains and that even weak return predictability can have important economic value in terms of utility- and profit-based metrics. These views challenge the foundations of conventional two-step modelling as eventually investors are interested in the received economic gains on their investments, not arguably convenient statistical criteria such as least squares.

In this study, we consider a classic and simple asset allocation problem where an investor trades between risky market return and risk-free rate. Our contribution is to introduce a flexible single-step empirical approach built upon direct utility maximization in finding portfolio weights. That is, we set our objective function in accordance with theoretical utility maximization premises also in empirical implementation instead of statistical criteria such as least squares used in linear predictive regressions. We show that this advanced direct portfolio weight determination can interestingly be combined with modern machine learning, specifically a customized gradient boosting algorithm, enabling various advantages over the past approaches. Our results can help investors with different risk aversion preferences to determine which predictive variables and their combinations are useful in optimizing their asset allocation decisions.

This study follows the footsteps of the seminal works by Welch and Goyal (2008) and Campbell and Thompson (2008), but importantly it is not the next attempt in already extensive literature modifying (linear) predictive re-

gressions to find statistical (out-of-sample) predictability in stock returns. Instead, our single-step approach sets portfolio weights directly by maximizing the underlying empirical utility function. In addition to the fundamental difference in the underlying objective function, we incorporate typically predetermined lower and upper bounds of the portfolio weights as a part of the method, whereas the weights obtained with predictive regressions are typically truncated subsequently to lie between these bounds (see, e.g., Campbell and Thompson, 2008; Rapach and Zhou, 2013; Neely et al., 2014). This 'post-truncation' step has been found successful in out-of-sample stock return forecasting, but it does not give much economic intuition why certain state variables are important predictors to portfolio weight decisions. Our single-step approach circumvents these complications and produces easily interpretable results.

Our contribution is strongly connected to a more general issue in (financial) econometrics on what is the appropriate objective (loss) function in econometric inference: See general discussion and arguments for much more detailed examination in this respect in Elliott and Timmermann (2016, Chapter 2). Moreover, as argued by Brandt (2010) in his literature review (see also Brandt, 1999; Aït-Sahalia and Brandt, 2001; Brandt and Santa-Clara, 2006; Brandt, Santa-Clara and Valkanov, 2009), besides the obvious intuitive fact that asset allocation is the ultimate object of interest, there are several benefits when focusing portfolio weights directly with available predictive information. Ultimately, in contrast to direct utility maximization, the usual two-step statistical approach requires first to learn the notoriously complicated data generating process of stock returns, with various misspecification possibilities, before obtaining necessary input predictions to be plugged in (ad-hoc) trading rules. To evaluate the resulting asset allocation decisions with several economic criteria does not remove the fact that the underlying inference is not directed to optimize economic performance. To address this challenge, we develop a customized gradient boosting algorithm to empirically optimize asset allocation decisions in a flexible and single-step manner.

In economics and finance, machine learning-based methods have so far been (rightly) somewhat criticized about lack of intuition and interpretability. Due to its flexibility to allow custom objective functions, among 'textbook' machine learning algorithms often binded with their specific objectives, gradi-

ent boosting is an excellent workhorse for our purposes. The usual 'statistical' gradient boosting algorithm with regression trees has generally been one of the most successful machine learning algorithms so far in different applications: See Rossi and Timmermann (2015) and Rossi (2018) as recent examples in finance to predict stock returns.

Our customized version diverges from the usual gradient boosting by utilizing the (negative) utility function as the objective function. We are hence able to establish an innovative synthesis between financial economics and machine learning practices instead of following everlasting least squares (mean square error) and hypothesis testing-based inference. That is, we get interpretable results on which predictive variables are truly important for asset allocation decisions when optimizing portfolio performance. Intuitively, customized gradient boosting is iteratively learning from training sample asset allocation mistakes, measured by the gradient of the objective function (i.e. now the selected utility function), before finally resulting in superior portfolio weight forecasts for the next period. This is intuitively in line with investors' attempts in practice to continuously update their trading strategy before taking future positions.

In empirical analysis, following the past (out-of-sample) return predictability studies for a comparison sake, we use a general form of the utility function that resembles to connect quadratic preferences to investors' decision making. This is the evaluation context typically considered in the past studies with an emphasis on out-of-sample forecasting performance, providing thus a natural building block for our approach. With the large updated dataset of macroeconomic and technical indicator predictive variables, originally compiled by Welch and Goyal (2008) and Neely et al. (2014), we are able to examine which predictors are truly important to asset allocation decisions rather than concentrating on the usual statistical predictability of stock returns.

Our empirical results on the monthly U.S. market returns and predictors show that substantial and quantitatively meaningful economic value can be obtained with our utility boosting method. This is the case even despite the fact that monthly stock returns are from the statistical perspective at most only weakly predictable. Technical indicators yield as a group the largest benefits in out-of-sample forecasting experiments. This is generally in line with the conclusions of Neely et al. (2014) and now confirmed with very different

methodology. In the full sample estimation and model selection results, the gains obtained with the utility boosting are broad, also containing some specific macroeconomic variables. Interestingly, theoretically well-motivated inflation and partly also the dividend-price ratio stand out both in- and out-of-sample predictions for which the past empirical findings on their usefulness, obtained with statistical predictive regressions, have been quite inconclusive.

The rest of the paper is organized as follows. In Section 4.2, we frame and present the starting point of our contribution, determined by the related past out-of-sample predictive regression studies, before setting up our utility boosting approach. Empirical results are reported in Section 4.3, before discussion on the main general findings and final conclusions in Sections 4.4–4.5. Various additional and robustness analyses are compiled into the attached Appendix A.

4.2 Methodology

4.2.1 Starting point and two-step statistical approach

Consider a classic and commonly examined simple asset allocation decision problem for an investor with a single-period horizon aiming to optimally compose portfolio value between risky asset $r_{m,t}$ (market return) and risk-free asset return r_{ft}.[1] For a given level of (initial) wealth, the classic asset pricing perspective to the investor's optimization problem is to find portfolio weights maximizing the underlying utility function. Let w_t denote the proportion of the portfolio value allocated to the risky asset at time t, which is based on the predictive information available at time $t-1$ and contained in the vector $\boldsymbol{x}_{t-1}$. The resulting portfolio return (at time t) is hence

$$r_{p,t} = r_{f,t} + w_t r_{e,t}, \tag{4.1}$$

where $r_{e,t}$ is the excess return on the (broad) stock market index in excess of the risk-free rate $r_{f,t}$ from the period $t-1$ to t.

Our aim is to determine (empirical) portfolio weights w_t in (4.1) directly

[1] An extension to multiple asset case requires a separate treatment to extend our methods to multiple-equation case (see details in Discussion in Section 4.4). This is yet out of scope of our advancement and left for the future research.

using one or multiple state variables (predictors) contained in $\boldsymbol{x}_{t-1}$ at time $t-1$ (i.e. $w_t \equiv w(\boldsymbol{x}_{t-1})$). By focusing directly on the weights w_t, we aim to capture predictable and possibly time-varying patterns in weights as opposed to just relying predictive regressions on expected returns. Despite this objective and the fact that we are not explicitly interested in stock return predictions, we connect and frame our approach to the context of (out-of-sample) return predictability examination originating from the seminal contributions by Welch and Goyal (2008) and Campbell and Thompson (2008).

Throughout this study and as a necessary selection to concretely setting up our approach, we consider a general utility function which resembles to attach quadratic (mean-variance) preferences to investor decision making. Following (4.1) and the formulation of Marquering and Verbeek (2004) and Fleming, Kirby, and Ostdiek (2001), among others, we consider to maximize the ex-ante (quadratic) expected utility of form

$$\begin{aligned} &\max_{w_t} \left\{ E_{t-1}(r_{p,t}) - \frac{1}{2}\gamma \operatorname{Var}_{t-1}(r_{p,t}) \right\} \\ =&\max_{w_t} \left\{ E_{t-1}(r_{p,t}) - \frac{1}{2}\gamma\, w_t^2 \operatorname{Var}_{t-1}(r_{e,t}) \right\}, \end{aligned} \tag{4.2}$$

where $\gamma > 0$ is the investor's risk aversion coefficient, representing the degree of risk aversion, and $E_{t-1}(\cdot)$ and $\operatorname{Var}_{t-1}(\cdot)$ denote the conditional expectation and conditional variance, given the information set at time $t-1$. The aim is to maximize (4.2) by determining w_t, using the information included in $\boldsymbol{x}_{t-1}$, resulting to portfolio returns $r_{p,t}$.

The utility scheme (4.2) is directly built upon the large majority of past (out-of-sample) return predictability studies, relevant to our attempt, and portfolio performance evaluation therein (see, e.g., Campbell and Thompson (2008), Rapach et al. (2010), Neely et al. (2014), Rossi (2018), among others, including the survey of Rapach and Zhou (2013)). They, likewise we, are not explicitly claiming that quadratic (mean-variance) utility function is necessarily exactly correct utility configuration in their two-step approaches. Our goal is simply to use an empirical counterpart of (4.2) as the selected underlying objective function in a single-step approach, to be developed in Sections 4.2.2–4.2.3, to determine asset allocation decisions.

Solving the maximization problem (4.2) leads to the solution of the optimal

weights

$$w_t^* = \frac{E_{t-1}(r_{e,t})}{\gamma \operatorname{Var}_{t-1}(r_{e,t})} = \frac{E_{t-1}(r_{m,t}) - r_{f,t}}{\gamma \operatorname{Var}_{t-1}(r_{e,t})}, \tag{4.3}$$

which is here written in terms of the expected excess stock return $E_{t-1}(r_{e,t})$. If the expected return on the risky asset increases (ceteris paribus), an investor increases his/her weight on the risky asset, whereas increasing risk (conditional variance) involved is negatively related to the optimal weights. However and importantly, the optimal theoretical solution (4.3) does not yet tell how to obtain weights empirically.

As the weights w_t are functions of predictors $\boldsymbol{x}_{t-1}$, our goal is to determine empirical weights and asset allocation decisions in direct utility maximization using the 'training' (estimation) data $\{\boldsymbol{x}_{t-1}, r_{e,t}\}_{t=1}^{T}$ where T denotes the sample size. Our utility boosting approach in Sections 4.2.2–4.2.3 follows the steps opened by Brandt (1999), Aït-Sahalia and Brandt (2001) and Brandt and Santa-Clara (2006) also looking beyond predicting conditional moments of excess stock returns when setting portfolio weights. Our utility boosting, however, emphasizes somewhat different final goals than their contributions by integrating recent advancement in machine learning to genuine prediction (forecasting) purposes in asset allocation decisions.

We develop our empirical approach as a complement to the dominant two-step empirical practice followed in the past (in and out-of-sample) return predictability research. It is built upon a simple linear predictive regression

$$r_{e,t} = \boldsymbol{z}_{t-1}'\boldsymbol{\beta} + \varepsilon_t, \qquad t = 1, \ldots, T, \tag{4.4}$$

where $\boldsymbol{z}_t = [1 \quad \boldsymbol{x}_t']'$, $E(\boldsymbol{z}_{t-1}\varepsilon_t) = \mathbf{0}$ and $\boldsymbol{\beta}$ is the vector of unknown parameters, containing also a constant term. The parameters $\boldsymbol{\beta}$ will be estimated by the method of ordinary least squares (OLS) where the underlying objective function is statistical minimizing the least squares criterion, given the estimation data $\{\boldsymbol{x}_{t-1}, r_{e,t}\}_{t=1}^{T}$ (including the (known) initial value $\boldsymbol{x}_0$):

$$\widehat{\boldsymbol{\beta}}_{OLS} = \arg\min_{\boldsymbol{\beta}} \sum_{t=1}^{T} \left(r_{e,t} - \boldsymbol{z}_{t-1}'\boldsymbol{\beta}\right)^2. \tag{4.5}$$

The resulting expected (predicted) excess returns

$$\widehat{r}_{e,t} = \boldsymbol{z}'_{t-1}\widehat{\boldsymbol{\beta}}_{OLS} \tag{4.6}$$

are then used as the empirical proxy for $E_{t-1}(r_{e,t})$ needed in (4.3).

In a similar fashion, an empirical proxy for the conditional variance of excess returns (hereafter denoted by $\sigma_t^2 \equiv \mathrm{Var}_{t-1}(r_{e,t})$) is required given the information available at time $t-1$. To allow for well-documented volatility clustering, we follow Campbell and Thompson (2008) and the subsequent studies (see, e.g., Rapach et al., 2010; Neely et al., 2014) by using a five-year rolling window variance of historical excess returns. This simplification can be relaxed by using, for example, the GARCH model or another realized volatility-based proxy (see the Appendix A.4). Therefore, given the (pre-determined) risk aversion coefficient γ and the result (4.3), an investor allocates the following share of the portfolio value to risky equity for the period t:

$$\widehat{w}_t = \frac{\widehat{r}_{e,t}}{\gamma\,\widehat{\sigma}_t^2} = \frac{\boldsymbol{z}'_{t-1}\widehat{\boldsymbol{\beta}}_{OLS}}{\gamma\,\widehat{\sigma}_t^2}. \tag{4.7}$$

Expression (4.7) summarizes how the portfolio weights can empirically be obtained as a function of predictors $\boldsymbol{x}_{t-1}$ in two steps relying on the commonly considered linear predictive regressions (4.4) (i.e. first constructing $\widehat{r}_{e,t}$ (and $\widehat{\sigma}_t^2$) before plugging them into (4.7) in the second stage).[2] This approach has, however, several critical issues and complications from the final and arguable the most important portfolio performance perspective that we aim to tackle in this study:

(i) The resulting estimated weights $\widehat{w}_t$ in (4.7) are not necessarily nowhere near between typically pre-determined (assumed) bounds

$$w_t \in [w^{\min}, w^{\max}]. \tag{4.8}$$

The common practice (see the above-mentioned return predictability studies)

[2] Several (statistical) nonlinear predictive regressions and systems have also been considered to predict stock returns with subsequent asset allocation decision objectives. See, e.g., the regime switching models surveyed by Guidolin (2011), including Guidolin and Timmermann (2007) who find that asset allocations guided by the forecasts of stock and bond returns from the Markov switching models yield utility gains relative to constant expected excess return predictions as described in equations (4.4)–(4.6).

has been to set the lower bound to zero ($w^{\min} = 0$), i.e. no short selling is allowed, and to consider mainly two different selections for the upper bound ($w^{\max}$): In Campbell and Thompson (2008), Neely et al. (2014) and Zhu (2015), w_t lies between 0 and 1.5 (i.e. $w^{\max} = 1.5$). They argue that these impose realistic portfolio constraints by precluding short sales and preventing more than 50% leverage. On the contrary, Aït-Sahalia and Brandt (2001), Marquering and Verbeek (2004) and Rossi (2018), among others, set the upper bound to $w^{\max} = 1$ (with $w^{\min} = 0$). This selection is also essentially the same as $w^{\max} = 0.99$ in, e.g., Kandel and Stambaugh (1996) and Cenesizoglu and Timmermann (2012). All in all, the empirical weights constructed as in (4.7) by no means guarantee the bounds (4.8) without strict additional and complicated restrictions on the predictive regression (4.4).

(ii) The empirical findings of Campbell and Thompson (2008), Rapach et al. (2010), Neely et al. (2014) and Pettenuzzo et al. (2014) emphasize the importance of restrictions (4.8) on the weights (4.7) to improve both statistical and economic out-of-sample predictive performance. To impose these subsequent constraints (4.8) turn out to modify the initial weights (4.7) substantially. It is important to realize that this post-truncation is not part of the predictive regression (4.4) by any means, and hence all the usual conventional statistical interpretations, such as t-test statistics, goodness-of-fit measures and model selection conclusions on useful predictors $\boldsymbol{x}_t$ are lost for further interpretations. In other words, even if a predictive variable is deemed statistically useful (i.e. 'statistically significant'), this does not necessarily mean that it really has important predictive information on asset allocation decisions when respecting the bounds (4.8) in the end. This logic works also other way round so that statistically poor predictors can be useful in direct portfolio weight determination.

(iii) Consider a commonly used one predictor ($x_{i,t}$) special case of (4.4)

$$r_{e,t} = \beta_{i,0} + x_{i,t-1}\beta_i + \varepsilon_{i,t}. \tag{4.9}$$

In the vast (in-sample) return predictability research, this is the common specification where testing the null hypothesis $\beta_i = 0$ of no (conditional) mean predictability in stock returns with a t-test statistic has been of particular interest. This is important because if (correctly) rejecting the null $\beta_i = 0$, the

resulting statistical predictability is widely interpreted to imply also useful predictive power for systematic asset allocation decisions via equations (4.3)–(4.7). This 'significance testing' setting is, however, commonly reported to suffer 'Stambaugh bias', i.e. substantial size distortions when $x_{i,t}$ is highly persistent and correlated with return innovations $\varepsilon_{i,t}$: See Stambaugh (1999), Rapach and Zhou (2013, Section 3.1) and the recent contributions of Demetrescu et al. (2020) as representative references of voluminous in-sample return predictability results and techniques aiming to improve statistical inference in model (4.9).

(iv) Finally, and importantly as also argued by Brandt (2010) and Elliott and Timmermann (2016, Section 4.2) in their surveys, the two-step predictive regressions-based approach to set the weights (4.7) belongs to the 'plug-in' methods: The statistical least squares criterion (4.5) in econometric modelling does not (most likely) line up with investors' true preferences and the final objective which is in asset allocation decisions. Even though the two-step approach is arguably a simple way to find required predictions to be plugged in (4.7), it generally leads to a discrepancy between the original goal (utility maximization) and the objective function in econometric inference.[3]

4.2.2 Objective function

To acknowledge all the difficulties (i)–(iv) reviewed in Section 4.2.1 and connected to the two-step approach around (linear) predictive regressions, we introduce a flexible non-parametric and nonlinear approach which is strictly building upon utility maximization also in empirical implementation to obtain weights w_t. We are thus integrating asset allocation decision making and machine learning via a customized gradient boosting algorithm with specific and important modifications over the mechanical use of existing machine learning algorithms. Section 4.2.3 presents details of the 'utility boosting' algorithm, with various favourable properties.

Our utility boosting approach follows the general lines and arguments made by Brandt (1999), Aït-Sahalia and Brandt (2001), Brandt and Santa-Clara (2006) and Brandt et al. (2009) in the early literature, arguing the importance of

[3] Sentana (2005) considers formal conditions and assumptions under which least squares-based predictions and mean-variance analyses are connected in associated market timing strategies.

direct portfolio decisions instead of the two-step 'plug-in' statistical approach. Along their studies, we are not explicitly relying on specific assumptions on the excess stock return data generating mechanism, such as the arbitrage pricing theory (APT) or the capital asset pricing model (CAPM). Instead, our (empirical) view is that the portfolio weight w_t is a direct, potentially highly nonlinear, function of state variables $\boldsymbol{x}_{t-1}$ maximizing the investor's utility. This linkage incorporates all the predictive information contained in $\boldsymbol{x}_{t-1}$ to determine the weights, irrespective of the conclusions on statistical mean return predictability as discussed in (ii) and (iii) in Section 4.2.1, including also the possible impact of higher conditional moments than just mean and variance.

On these past utility-based approaches, the closest to our approach seems Brandt and Santa-Clara (2006) where they parametrize the portfolio weight as a linear function of predictors (state variables) $\boldsymbol{x}_{t-1}$ and solve the optimal values of the present parameters maximizing expected quadratic utility function similar to (4.2). Their approach is empirically, however, much more restrictive than ours (see details in the Appendix B) and designed more closely on portfolio choice problems with a genuine cross-sectional dimension as well (i.e. multiple risky assets). Moreover, their resulting portfolio weights can be interpreted as being proportional to the standard OLS regression of a vector of ones on the excess returns and, importantly, additional subsequent constraints to maintain the bounds (4.8) are required to address the point (i) in Section 4.2.1.[4]

To set a tractable empirical counterpart of (4.2), we first convert the maximization problem to a minimization problem. We will train our boosting algorithm with the same training (estimation) data $\{\boldsymbol{x}_{t-1}, r_{e,t}\}_{t=1}^{T}$ (including also the initial values for the volatility proxy) as in the least squares-optimal statistical two-step approach in equations (4.4)–(4.7) where $r_{e,t}$ implicitly contains required information on both $r_{m,t}$ and $r_{f,t}$. Our utility-based empirical

[4] In addition to the OLS interpretation of the portfolio weights as obtained in Brandt and Santa-Clara (2006), Brandt (1999) and Brandt et al. (2009) build upon on the (statistical) method of moments and 'maximum utility estimator', respectively. At the end, they also emphasize statistical hypothesis testing with the aim to obtain 'statistically significant' results in the evaluation stage, whereas our perspective is different and specifically in prediction (forecasting) performance after direct utility-based modelling.

objective function (cf. (4.2)) is

$$\underset{w_t}{\arg\min}\left\{\frac{1}{T}\sum_{t=1}^{T}-u_t\right\}=\underset{w_t}{\arg\min}\left\{\frac{1}{T}\sum_{t=1}^{T}-\left(r_{p,t}-\frac{1}{2}\gamma\, w_t^2\,\sigma_t^2\right)\right\}, \tag{4.10}$$

where $r_{p,t}$ denotes the resulting portfolio returns (see (4.1)). The (negative) utility contribution of the tth observation, $-u_t$, is crucially dependent on the weights w_t constructed with the information contained in $\boldsymbol{x}_{t-1}$. The selected volatility proxy σ_t^2 is already introduced in connection to (4.7): It is the same as in (4.10) and in the two-step approaches throughout this study for comparison reasons (cf. a different formulation in this respect in Brandt and Santa-Clara (2006)). The form (4.10) is the one examined in various return predictability studies as an evaluation diagnostic tool measuring portfolio performance (see Marquering and Verbeek, 2004; Campbell and Thompson, 2008; Rapach et al., 2010; Rossi, 2018) and hence it acts as a natural choice for our advancement.

For simplicity and in accordance with past closely related studies, throughout this study, we set the lower bound in (4.8) as $w^{\min}=0$ and hence excluding short selling.[5] Moreover, to respect the pre-determined maximum weight $w^{\max}$ explicitly as a part of our procedure (cf. the subsequent weight truncation needed in (4.7)), we set

$$w_t = w^{\max}\lambda_t, \tag{4.11}$$

where the portfolio weight is essentially specified by the logistic function

$$\lambda_t=\frac{1}{1+\exp\left(-\frac{1}{\gamma\,\sigma_t^2}F(\boldsymbol{x}_{t-1})\right)}. \tag{4.12}$$

Combined with (4.11), the logistic growth curve form (4.12) guarantees that the weights (4.11) are all the time inside the interval $w_t\in[0,w^{\max}]$. In (4.12), the essential ingredient to determine portfolio weights (cf. (4.4)) is the component $F(\boldsymbol{x}_{t-1})$, which is possibly a complex function of the predictive information $\boldsymbol{x}_{t-1}$ that we aim to teach with our training data. To this end, in Section 4.2.3, we specifically develop a customized version of the gradient boosting, which is in principle only one but seemingly highly relevant empirical algorithm over

[5] An extension allowing also for short selling (i.e. negative weights) is possible but means a different and slightly more complicated parametrization than the one in (4.11)–(4.12).

alternatives to determine $F(\boldsymbol{x}_{t-1})$ when aiming to maximize the empirical utility.

In specification (4.12), the impact of $F(\boldsymbol{x}_{t-1})$ is adjusted by the risk proxy σ_t^2 and the risk aversion coefficient γ along the expression (4.3).[6] We can also strengthen the linkage to (4.7) by the following 'payoff to stake' representation

$$\log\left(\frac{\lambda_t}{1-\lambda_t}\right) = \log\left(\frac{w^{\max}\lambda_t}{w^{\max}(1-\lambda_t)}\right) = \frac{F(\boldsymbol{x}_{t-1})}{\gamma\,\sigma_t^2}.$$

This is the log odds ratio of λ_t where higher λ_t reflects the likelihood of high portfolio weight should take place. In addition to this representation, the inclusion of the volatility proxy σ_t^2 in (4.12) is strongly motivated by the early empirical evidence of Fleming et al. (2001) and Marquering and Verbeek (2004) on the importance of an explicit volatility component to determine portfolio weights.

In practice, the utility boosting algorithm, to be presented more detail in Section 4.2.3, provides the practical method to determine the weights. It can intuitively be interpreted so that an investor, with given risk aversion preferences and constraints (4.8), aim to continuously optimize and update his or her asset allocation decision mechanism with the available past predictive information targeting to find optimal portfolio weight for the next period. This emphasizes the training step of our algorithm before predicting the weight for the next period out of sample. This thinking again somewhat diverges from the goals of Brandt and Santa-Clara (2006), and a few closely related studies, concentrating on full sample portfolio policies.

4.2.3 Customized gradient boosting

To maximize the empirical utility (i.e. to minimize (4.10)), we determine the optimal weights directly by using a customized gradient boosting algorithm, acknowledging all the complications (i)–(iv) of the current two-step statistical approach presented in Section 4.2.1. Specifically, this means to specify the ingredient $F(\boldsymbol{x}_{t-1})$ in (4.12), leading to the 'boosted' portfolio weights max-

[6] The identity $E_{t-1}(r_{e,t}) = w_t^* \gamma \sigma_t^2$ in (4.3) shows the linkage between the estimated weights and implied expected excess returns. Even though it is not of our main interest, the extracted weights (4.11), obtained with (4.12), can thus also be interpreted to imply a proxy for expected excess stock returns.

imizing the empirical utility as the objective function instead of aiming to predict excess stock returns with predictive regressions.

Boosting is a powerful technique originating from the machine learning community. The general idea is that simple weak models, also known as 'base learners', are combined in a stagewise manner to form a 'boosting ensemble' with strong predictive performance, leading to superior portfolio weights in our context. Friedman, Hastie and Tibshirani (2000) introduced the statistical framework for boosting which enables additional theoretical insights into the success of boosting and has led to a variety of new boosting algorithms.

Provided with sufficient amount of data and flexible base learners, such as regression trees or smoothing splines, boosting can basically approximate any kind of functional form to determine weights w_t (cf. the linearity assumption in (4.4) before the weight truncation (4.8)). Boosting also performs model selection simultaneously with estimation as each new optimal base learner function is found by conducting an extensive search involving all the predictor variables. This makes boosting a viable algorithm for (relatively) large predictor sets, such as the one of interest in Section 4.3. Another major advantage is the interpretability of the final model: Unlike the mechanical use of existing machine learning algorithms, as we are now building the method explicitly on the financial economics and asset pricing bases, the final outcome provides important insights on the most relevant predictors (state variables) specifically for portfolio weight determination.

Boosting has shown considerable success in different fields including robotics, medical statistics and economics. Past financial applications range from stock return predictions (Rossi and Timmermann, 2015; Rossi, 2018), volatility forecasting (Mittnik, Robinzonov and Spindler, 2015), yield curve modelling (Audrino and Trojani, 2007) and failures in banking sector (Carmona, Climent and Momparler, 2019). Our context is, however, different than allowed by the basic setup of gradient boosting. That is, here the boosting algorithm contains the customized objective function, the (negative) utility function (4.10), while the usual boosting algorithms are fully statistical containing mean square error (MSE) and likelihood-based ingredients. This shows that instead of the mechanical use of gradient boosting, our goal is to specifically integrate a flexible modern machine learning algorithm with the asset allocation objective.

Following Bühlmann and Hothorn (2007) and the bulk of the boosting estimators, our customized algorithm can be described as follows:

Input: The training data $\{(\boldsymbol{x}_{t-1}, r_{e,t})\}_{t=1}^{T}$ is the same as in linear predictive regressions (4.4)–(4.7). The essential part is to determine $F(\boldsymbol{x}_{t-1})$ in (4.12) using a differentiable objective function (4.10), given the fixed risk aversion coefficient γ, selected volatility proxy σ_t^2 and the number of iterations M (see below).

Algorithm[7]:

1. Initialize the algorithm by a constant value: We select, for simplicity, $F_0(\boldsymbol{x}_{t-1}) = 0$.

2. For $m = 1, \ldots, M$:

   (a) Compute 'pseudo-residuals':

   $$\widetilde{y}_{t,m} = -\left[\frac{\partial(-u_t)}{\partial F(\boldsymbol{x}_{t-1})}\right]_{F(\boldsymbol{x}_{t-1})=F_{m-1}(\boldsymbol{x}_{t-1})} \qquad \text{for } t = 1, \ldots, T.$$

   The gradient is obtained by the chain rule. Using the expression (4.12), given the selected volatility proxy σ_t^2 and noticing the negative sign in (4.10), we get:

   $$\begin{aligned} \frac{\partial(-u_t)}{\partial F(\boldsymbol{x}_{t-1})} &= \frac{\partial(-u_t)}{\partial w_t}\frac{\partial w_t}{\partial \lambda_t}\frac{\partial \lambda_t}{\partial F(\boldsymbol{x}_{t-1})} \\ &= -\Big(r_{e,t} - \gamma\, w_t\, \sigma_t^2\Big)\frac{w^{\max}\,\lambda_t(1-\lambda_t)}{\gamma\,\sigma_t^2}. \end{aligned} \tag{4.13}$$

   (b) Fit a base learner $h_m(\boldsymbol{x}_{t-1})$ to pseudo-residuals, i.e. train it using the training set $\{(\boldsymbol{x}_{t-1}, \widetilde{y}_{t,m})\}_{t=1}^{T}$.

   (c) Update the prediction: $F_m(\boldsymbol{x}_{t-1}) = F_{m-1}(\boldsymbol{x}_{t-1}) + \upsilon h_m(\boldsymbol{x}_{t-1})$.

3. Output $F_M(\boldsymbol{x}_{t-1})$, leading to the solution of (4.10), given (4.11) and (4.12).

As a whole, mathematically the above algorithm can be understood to minimize (4.10) by iterative steepest descent in function space. Bühlman

[7] A slightly modified version will be considered in the Appendix A.3. to obtain evidence for the robustness of our main findings.

and Hothorn (2007) call it the functional gradient descent algorithm. After initializing the algorithm in step 1 with a simple scalar value the pseudo-residuals in step 2.(a) can be constructed. These are the negative gradients of the objective function evaluated with the boosting ensemble learnt so far with $F_{m-1}(\boldsymbol{x}_{t-1})$. The base learner model that best fits the negative gradient is then selected and added to the ensemble. The impact of each update is shrunk towards zero using a step-length factor $\upsilon \in \{0, 1\}$, which we set to 0.001 according to the past boosting studies. The whole process (step 2) is repeated 500 times (i.e. $M = 500$). For the performance of the boosting estimator, the base procedure and the stopping rule in step 2 are the most important ones. These and other tuning parts of the estimator will be presented and discussed in more detail in the Appendix C.[8]

With the above gradient boosting, we can, in theory, reach the ultimate equilibrium with excessive boosting iterations M where the algorithm is overfitted to the training (in-sample) data. However, it should be emphasized that especially in out-of-sample predictions, we only use the data that we have at hand.[9] We will control the boosting iterations and the underlying speed of learning in estimation (learning) stage to terminate the iterative fitting (steps 1–3) at the right time, to get portfolio weights based on their genuinely predictable patterns of interest rather than ending up to overfitted weights. This means that the resulting weights and asset allocation decisions are obtained directly with the aim to maximize (4.2) but at the same time circumventing overfitting of the training data. These selections enable also meaningful full sample analyses, as generally examined in empirical finance research so far, on the importance of different predictors (state variables) in portfolio weight determination.

[8] Following various utility configurations (see, e.g., by Aït-Sahalia and Brandt (2001)), also examined within the numerical 'full-scale optimization' search algorithms (see Adler and Kritzman, 2007), our method can also, in principle, be extended to various other (differentiable) utility functions than the one in (4.2) and (4.10). The quadratic form (4.2) is expected to provide at least a good approximation for most variations of power utility (see also, e.g., Campbell and Thompson, 2008, footnote 11) and is, in particular, in line with the past return predictability studies that we are mainly linking our approach and empirical analysis in Section 4.3.

[9] To clarify our notation, as our estimation (training) data contains the observations $\{(\boldsymbol{x}_{t-1}, r_{e,t})\}_{t=1}^{T}$, in forecasting situation portfolio weight forecasts are made for the time $T+1$, i.e. w_{T+1}, using the information up to time T. This explains the difference in 'forecasting' and more general 'prediction' goals including full sample (in-sample) results.

4.3 Empirical results

In empirical analysis below, we compare our utility boosting method to the existing return predictability studies, with out-of-sample forecasting emphasis, using the same dataset (Section 4.3.1) and evaluation criteria (Section 4.3.2) as in Neely et al. (2014), Welch and Goyal (2008), Campbell and Thompson (2008), Rapach et al. (2010) and Rapach and Zhou (2013). In Sections 4.3.3–4.3.4, we report the full sample (in-sample) estimation results, with the special interest on comparing the best predictors for direct portfolio weight determination. Section 4.3.5 reports the out-of-sample asset allocation results. In Sections 4.3.3–4.3.5, we concentrate on reporting the results, whereas in Section 4.4 we summarize and discuss the obtained empirical results for more general conclusions. Various additional results and robustness checks for our main findings are compiled into the attached Appendix A.

4.3.1 Dataset

Following the closely related return predictability studies, we consider an updated monthly dataset compiled by Welch and Goyal (2008) and extended by Neely et al. (2014), containing various predictive variables to determine portfolio weights directly in our single-step and indirectly in the conventional statistical two-step plug-in methods. As in Neely et al. (2014), our sample period starts from December 1950 and is now updated until December 2018.[10]

Excess stock returns (i.e. the equity risk premium) is obtained as the difference between the return on the S&P 500 index (including dividends) and the risk-free interest rate. The set of macroeconomic predictive variables contains 14 variables as originally in Welch and Goyal (2008). Detailed descriptions of all predictors are presented in Table 4.1. Table 4.2 reports the summary statistics for monthly excess stock returns (Panel A) and macroeconomic variables (Panel B). The average monthly equity risk premium is 0.60% with monthly standard deviation of 4.14%. This produces a monthly Sharpe ratio of 0.144.

[10] The Welch-Goyal dataset, containing macroeconomic variables and the S&P 500 index (returns), is from Amit Goyal's website at http:/www.hec.unil.ch/agoyal. Technical indicators are obtained with the S&P 500 data. For a general and extensive summary of the past return prediction findings obtained with different predictors and existing methodological approaches (see Section 4.2.1), see the survey of Rapach and Zhou (2013).

As the Panel B shows, most of the macroeconomic variables are rather persistent (cf. the challenge (iii) in Section 4.2.1) including, as expected, the valuation ratios, nominal interest rates and partly also interest rate spreads.

Together with the macroeconomic variables, Table 4.1 also lists 14 binary-valued technical indicators as predictors. These are based on the same three representative trend-following technical indicators as in Neely et al. (2014). As they summarize it (see also, e.g., Rapach and Zhou, 2013), instead of macroeconomic fundamentals, technical indicators rely on past price and volume patterns with the idea that they will identify future price trends. The theoretical models explaining the predictive power of technical indicators are based on how investors can be heterogeneous regarding the availability of new information, their response to this information and how they view the overall investor sentiment (see Neely et al., 2014, and the references therein).

The first indicators are moving average (MA) buy and sell signal rules

$$\mathrm{MA}(s,l) = I(\mathrm{MA}_{s,t} \geq \mathrm{MA}_{l,t}), \tag{4.14}$$

where $I(\cdot)$ is an indicator function and

$$\mathrm{MA}_{j,t} = \frac{1}{j}\sum_{i=0}^{j-1} P_{t-i}, \quad j = \{s,l\}, \quad s = \{1,2,3\}, \quad l = \{9,12\},$$

where P_t is the level of the S&P 500 index. These MA rules reflect the short- and long-run trends in stock price movements. Moreover, the second set of technical indicators are based on the momentum signals

$$\mathrm{MOM}(m) = I(P_t \geq P_{t-m}), \quad m = \{9,12\}, \tag{4.15}$$

where a positive (negative) momentum signal means that the current value of the index is higher (smaller) than m periods ago. Finally, the third set of technical indicators incorporates also the volume data to identify market trends. Define

$$\mathrm{OBV}_t = \sum_{k=1}^{t} \mathrm{VOL}_k D_k, \qquad D_k = 2\,I(P_k - P_{k-1}) - 1,$$

where VOL_k is the trading volume during the period k. The trading signal is

then

$$\mathrm{VOL}(s,l) = I(\mathrm{MA}_{s,t}^{\mathrm{OBV}} \geq \mathrm{MA}_{l,t}^{\mathrm{OBV}}), \tag{4.16}$$

where the volume data of the S&P 500 index is obtained from Finance Yahoo and

$$\mathrm{MA}_{j,t}^{\mathrm{OBV}} = \frac{1}{j}\sum_{i=0}^{j-1} \mathrm{OBV}_{t-i}, \quad j = \{s,l\}, \quad s = \{1,2,3\}, \quad l = \{9,12\}.$$

Intuitively, say, relatively high recent trading volume combined with recent price increases indicates a strong positive market trend. Like with macroeconomic variables, the predictive information of the technical indicators (4.14)–(4.16) at time $t-1$ is used in $\boldsymbol{x}_{t-1}$ and in the weight determination for the period t. Due to the binary nature of (4.14)–(4.16), we do not report the descriptive statistics of technical indicators in Table 4.2.

4.3.2 Evaluation and benchmarks

Before reporting our in- and out-of-sample prediction results in Sections 4.3.3–4.3.5, we introduce our main evaluation criteria and benchmark approaches. In the past research, economic evaluation criteria, such as resulting utilities and portfolio returns after fitting the linear predictive regression (4.4) as in (4.5)–(4.6), have typically been treated as secondary over the statistical ones and return predictability considerations. As briefly reviewed in Section 4.2.1, model (4.4), and (4.9) as a typically considered special case, has potentially difficult econometric issues related to the evaluation of statistical significance of persistent predictors in $\boldsymbol{x}_t$. Importantly from our point of view, we are stressing that the usefulness of a certain model, with one or more predictors, is not determined by the conventional statistical significance of individual coefficients. Instead, it is directly its usefulness in asset allocation decisions what matters.[11]

The first economic and profit-based evaluation criteria is naturally the resulting average utility (cf. the objective function (4.10), now percentages per

[11] As in the footnote 9, it is important to point out that below the notation $t = 1, \ldots, T$ refers to the in-sample (training) performance. However, the same criteria will be used to evaluate out-of-sample portfolio performance with necessary changes in notation. In other words, portfolio weight forecasts for the period $T+1$ are constructed using the information available at time T.

month)

$$\bar{u}(\widehat{\boldsymbol{w}}) \quad = \quad 100 \times \frac{1}{T}\sum_{t=1}^{T}\widehat{u}_t = \frac{100}{T}\sum_{t=1}^{T}\Big\{\widehat{r}_{p,t} - \frac{1}{2}\gamma\,\widehat{w}_t^2\,\widehat{\sigma}_t^2\Big\}, \tag{4.17}$$

where $\widehat{\boldsymbol{w}} = (\widehat{w}_1, \ldots, \widehat{w}_T)$ is the vector of estimated weights, $\widehat{u}_t$ is the utility contribution of tth observation and $\widehat{r}_{p,t}$ is the resulting portfolio return (i.e. expression (4.1) evaluated with $\widehat{\boldsymbol{w}}$). To enable explicit comparison between different methods, throughout this study, we utilize the same conditional variance estimate $\widehat{\sigma}_t^2$ (i.e. the 60-month rolling window volatility) constructed using the information at time $t-1$. Robustness checks with the GARCH model and realized variance (RVOL, see Table 4.1) based volatility proxies, reported in the Appendix A.4, lead to largely the same main empirical conclusions. The risk aversion coefficient γ is fixed to the typical and commonly used value $\gamma = 5$ (see, e.g., Neely et al., 2014; Cenesizoglu and Timmermann, 2012; Brandt and Santa-Clara, 2006) or $\gamma = 3$ (see, e.g., Campbell and Thompson, 2008; Rapach et al. 2010; Zhu, 2015).[12]

An important and commonly used benchmark to our utility boosting method, and also for the predictive regressions with the subsequent restrictions (4.8), is that predictors $\boldsymbol{x}_{t-1}$ do not have useful predictive power to determine portfolio weights. For the linear predictive regressions, this implies that the expected excess return (4.6) is constant over time and (4.4) contains only the constant term. In the two-step approach, this leads to the use of historical average ('HA') return, $\bar{r}_{e,t}$, and weights

$$\bar{w}_t = \frac{\bar{r}_{e,t}}{\gamma\,\widehat{\sigma}_t^2}. \tag{4.18}$$

This is the simple benchmark commonly employed in the past (out-of-sample) return predictability literature and subsequent portfolio performance evaluation. As argued by Welch and Goyal (2008), and various subsequent studies, it

[12] Selections $\gamma = 4$ and $\gamma = 6$ have also commonly been used (see, e.g., Marquering and Verbeek, 2004; Rossi, 2018) and hence $\gamma = 5$ turns up a suitable compromise between them. Moreover, following the reasoning of Rossi (2018), moving from $\gamma = 3$ to $\gamma = 5$ provides guard and a reasonable check against the impact of estimation uncertainty by increasing the value of γ. Additional arguments favouring the importance of incorporating estimation uncertainty in asset allocation decisions can be found, e.g., in Kandel and Stambaugh (1996), Barberis (2000) and Kan and Zhou (2007).

is a highly adequate statistical description for excess stock returns. Notice that despite the constant expected return, the weights (4.18) are time-varying and governed by the estimated volatility proxy $\widehat{\sigma}_t^2$. Building upon the thinking of (4.18), we consider a restricted utility boosting approach where $\boldsymbol{x}_{t-1}$ contains only the constant (i.e. $\boldsymbol{x}_{t-1} = 1$) for all t. That is 'Const' approach (cf. (4.12)) hereafter, where $F(\boldsymbol{x}_{t-1})$ in the customized gradient boosting algorithm does not include any predictive information over the conditional volatility σ_t^2, which is in line with the idea of volatility timing (see the general arguments favouring such approach in Fleming et al. (2001)).

A closely related evaluation criterion to the average realized utility (4.17) is the certainty equivalent return gain over the historical average (HA) weights (4.18):

$$\text{CER gain} = 1200 \times (\text{CER} - \text{CER}_{\text{HA}}), \tag{4.19}$$

where

$$\text{CER} = \bar{r}_{p,t} - \frac{\gamma}{2}\,\text{Var}(\widehat{r}_{p,t}), \qquad \bar{r}_{p,t} = \frac{1}{T}\sum_{t=1}^{T}\widehat{r}_{p,t}, \tag{4.20}$$

and CER_{HA} is obtained when replacing $\widehat{w}_t$ by $\bar{w}_t$ in $\widehat{r}_{p,t}$ construction. The difference to (4.17) is that in (4.20) the volatility estimate is the resulting portfolio variance. This change does not turn out to change our main empirical results at all, providing important robustness for our findings. The CER gain (4.19), likewise differences in the estimated average utilities (4.17), can be interpreted as the received additional economic value of the utility boosting method over the historical average benchmark (4.18). We report the CER gain (4.19) multiplied by 1200 so that it can be interpreted as the annual percentage portfolio management fee that an investor would be willing to pay to have an access to the utility boosting-based asset allocations instead of (4.18).

In addition to the above criteria, we also consider the traditional (monthly) Sharpe ratio

$$\text{Sharpe} = \frac{\frac{1}{T}\sum_{t=1}^{T}\widehat{r}_{ep,t}}{\sqrt{\text{Var}(\widehat{r}_{ep,t})}}, \tag{4.21}$$

where $\widehat{r}_{ep,t} = \widehat{r}_{p,t} - r_{f,t}$ denotes the resulting portfolio return in excess of the risk-free rate. The Sharpe ratio (4.21) is hence the mean portfolio return in excess of the risk-free rate divided by the standard deviation of excess

portfolio returns. As a reward-to-variability ratio, the Sharpe ratio measures the additional amount of excess return that an investor receives per unit of increase in risk. Asset allocation decisions with a high Sharpe ratio are preferable to those with a low Sharpe ratio.

4.3.3 In-sample (full sample) results

Following the common practice in empirical finance and specifically in stock return predictability studies, before examining out-of-sample forecasting results in Section 4.3.5, we consider the full sample results for the sample period 1951:1–2018:12 (816 observations). The main interest in this and Section 4.3.4 is to examine differences between the utility boosting method and various benchmarks to explore which predictive variables are the most useful ones for portfolio weight determination when the maximum amount of information (the longest data availability) is employed in the analysis.

As explained in Section 4.2.3, it should be emphasized that our utility boosting approach contains shrinkage type of elements and hence provides guard against potential overfitting concerns, even for in-sample analysis. This means, together with the fact that the utility boosting and the two-stage predictive regression-based weighting are not generally nested approaches (i.e. either one is not obtained as a special case of the other one), utilities resulting from the former are not automatically higher than in the latter in and out of sample. This should happen especially if the predictive power of a certain variable $x_{i,t}$ for portfolio weight determination is indeed non-existent or negligible. Moreover, in line with the arguments of Inoue and Kilian (2005), it should be kept in mind that there are also potential complications in out-of-sample forecasting evaluation, such as the selection of the evaluation period and other potentially random (outlier-type) events, which may obscure forecasting results as well.

Table 4.3 presents the in-sample (full sample) results for the single-predictor models, and the benchmarks described in Section 4.3.2, for the investor with the risk aversion coefficient of $\gamma = 5$. We report the average utilities (4.17) and Sharpe ratios (4.21) for the single-step utility boosting and the conventional two-step (statistical, 'linear') approaches. The resulting portfolio weights fulfil the bounds (4.8), which are $w^{\min} = 0$ and $w^{\max} = 1$ in this table. For illustra-

tive reasons, we present both the average utilities (4.17) (util%) and CER gains (4.19) for the boosting method, leading to essentially the same conclusions here and also in other analysis settings. Moreover, even though the statistical significance of the individual predictors is indeed secondary in this study, to establish comparisons to the past linear predictive regressions studies, we also report the typical heteroskedasticity-autocorrelation consistent (HAC) t-statistics[13] and adjusted-R^2s ('adj-R^2') to measure the degree of statistical return predictability. As pointed out in Section 4.2.1, these statistics provide only partial evidence when the final objective is in asset allocation decisions and economic value of predictions.

Starting with the conventional statistical criteria, given the inherently substantial unpredictable component in monthly stock returns, Campbell and Thompson (2008) and Neely et al. (2014) concluded that a monthly adjusted-R^2 near 0.5% might represent economically significant degree of equity risk premium predictability. Together with using, e.g., the t-value close to 2 in absolute value as an indicative threshold for statistically significant predictors (at the 5% significance level), Table 4.3 shows as a whole that technical indicators seem to express useful but small statistical predictive power. There are also a few macroeconomic predictors, mainly TBL, LTR and RVOL, with statistically significant predictive content. When moving to realized utilities (util% and CER gain), we can see that utility boosting yields consistently higher average utility levels over the two-step 'linear' approach and benchmarks (Const and HA).[14] It is not surprising that in the full sample results our flexible non-parametric approach can find higher economic predictive power in the best (mostly real-valued) macroeconomic variables with potentially larger amount of information than the binary-valued technical

[13] Throughout this study, we systematically report the HAC t-statistics by means of the Newey-West (1987) estimator with lags determined by the rule (integer) $\text{floor}(4 \times (T/100)^{(2/9)})$. In other words, we do not take any specific standpoint on the statistical significance of the estimated β_i coefficients in (4.9), or the appropriateness of the HAC standard errors for all the predictors, due to the potential Stambaugh bias as discussed in the point (iii) in Section 4.2.1.

[14] There is one exception, MA(2,12), with a very marginal difference. As pointed out above, this is possible due to the elements controlling overfitting of the boosting algorithm. Similarly, for DE and DFY the utility boosting does not find substantial additional value in this setting ($\gamma = 5$ and $w^{\max} = 1$). These cases importantly show that the flexible utility boosting do not automatically outperform the linear two-step approach in sample and hence some overfitting concerns can already be eased.

indicators. This fact already shows, together with the analysis of Neely et al. (2014), that it is reasonable to treat these two categories of predictors partly separately at least in this section.

The best macroeconomic predictors in terms of average utility for the investor (when $\gamma = 5$ and $w^{\max} = 1$) are DY, LTR and RVOL. In contrast to other best predictions in terms of statistical criteria, DFR (default return spread) and inflation (INFL) have in relative sense much more useful information content for the direct portfolio weight determination than obtained with the statistical predictive regressions and statistical goodness-of-fit measures. Moving to the results of the technical indicators, the shortest MA rules (MA(1,9)–MA(3,9)), together with VOL(3,12), have the highest predictive power. These are largely the best ones also in terms of statistical criteria, even though some minor differences occur.

In Table 4.4, we provide the first of many alternative specifications to the one considered in Table 4.3. The risk aversion parameter is now lower ($\gamma = 3$) but still a common selection in the past related studies (see Section 4.3.2), implying less risk aversive investor profile. The main results are, however, largely the same as in Table 4.3 where $\gamma = 5$. The biggest differences are the rise of TMS and TBL in terms of received utilities whereas specifically LTR performs well in predictive regressions but does not stand out as a strong predictor in the utility boosting relative to other best predictors. As in Table 4.3, DFR and INFL are again examples of the best performing state variables in which the utility boosting finds much more useful predictive content than in the past two-step statistical approach. As the level of risk aversion, and hence the impact of conditional variance in (4.17), decreases due to the lower value of γ, the realized utility levels are throughout somewhat higher than in Table 4.3.

One of the main empirical results of this study is already evident in Tables 4.3–4.4: In terms of the Sharpe ratio (4.21), the utility boosting method strongly outperforms the conventional two-step statistical approach. That is the received risk-adjusted portfolio returns are substantially and throughout higher in the boosting method. The exact additional value varies between the predictors, but the Sharpe ratios of the boosted weights and resulting portfolio returns are mostly about 10–40% higher than in the two-step approach, but still within the realistic values in sample. This superior risk-return compromise

is coming largely from the smaller portfolio return variance: The mitigated volatility in the utility boosting applies also to the average utility and portfolio weights over the sample period ($t = 1, \ldots, T$).[15]

Another interesting predictor-specific empirical finding, together with the rise of inflation (INFL) and default return spread (DFR), is related to the dividend-price ratio (DP and the dividend yield (DY)). This is the variable which has especially been examined a lot in the past return predictability research due to its tight linkage to asset pricing theory and the relationship with the present value model. It is also probably the most commonly considered predictive variable in connection to the potential Stambaugh bias (point (iii) in Section 4.2.1), with ambiguous empirical conclusions. Goyal and Welch (2003, 2008) specifically argue that despite of their wide attempts, they could not find robust statistical predictive ability in the dividend-price ratio (dividend yield). The utility boosting approach clearly supports the usefulness of DY and DP much more when the objective is in the asset allocation decisions instead of the statistical performance.

As an illustrative example, in Figure 4.1 we depict the estimated portfolio weights $\widehat{w}_t$ in the utility boosting and the linear predictive regression-based two-step approaches using the dividend-price ratio (DP) as a predictor. There are several periods with clearly different weights between the methods. The deviation in the estimated weights is especially large during the 1960s and 1970s and from the year 2000 onwards. The impact of weight truncation to fulfil bounds (4.8) is also evident in the two-step approach, while the utility boosting keeps the weights automatically inside the selected interval. Weight truncation in the predictive regression-based approach is especially needed for the upper limit ($w^{\max} = 1$) and can be seen as multiple flat sections in Figure 4.1.

Similarly as in various return predictability studies, instead of single-predictor analyses, next we utilize multiple predictive variables in $\boldsymbol{x}_{t-1}$ simultaneously. As in Neely et al. (2014), we first consider the macro variables (MACRO) and technical indicators (TECH) separately and then finally jointly (ALL, i.e. combining MACRO and TECH) to determine asset allocation de-

[15] The excess portfolio returns are also higher in the utility boosting approach and the best performing models are basically exactly the same as obtained with presented criteria. Moreover, the returns are in line with the conventional levels (about between 5-8% in annualized terms).

cisions. One of the advantages of the utility boosting method is that no pre-selection is required as the algorithm performs model selection internally. In contrast, following Neely et al. (2014), among others, in the two-step predictive regression approach we first extract the principal components of the candidate set of predictors (MACRO, TECH and ALL). Notice that the exact multicollinearity between some of the macro variables (see Table 4.1) implies that the direct use of OLS, as in (4.5), is not even possible without these additional steps.

Table 4.5 reports the results of multivariate predictor models in otherwise the same setting as above in predictor-specific analyses in Tables 4.3–4.4. In the Panels A and B, the risk aversion coefficients are $\gamma = 5$ and $\gamma = 3$, respectively. In the linear predictive regressions, we utilize the same principal component approach as in Neely et al. (2014).[16] This comparison strengthens the superior performance of the utility boosting: All the combinations outperform the benchmark cases reported in Tables 4.3–4.4 as well as the two-step 'linear' alternatives.

Neely et al. (2014) conclude that the macroeconomic variables and technical indicators provide almost completely complementary predictor sets to predict equity risk premium. When moving to direct asset allocation decisions, we can confirm that this seems to be the case when evaluating linear predictive regressions with economic goodness-of-fit measures while in the utility boosting macroeconomic variables dominate technical indicators in sample. That is, in Table 4.5, the case ALL is driven by fluctuations in the macroeconomic variables with only minor role for technical indicators. This is not surprising at all: As discussed above, this is largely due to the nature of macroeconomic variables almost necessarily containing more information in sample than binary-valued technical indicators. Tree-based models are often reported to favor continuous-type predictors (see e.g., Loh and Shih, 1997). It is hence also important to consider out-of-sample forecasting results before further conclusions.

In accordance with the above views, Table 4.6 presents the top-10 predictors chosen by the internal model selection capability of the customized

[16] This means that the maximum amount of principal components is set to 3 for predictor group MACRO, 1 for TECH and 4 for ALL, and the final selection is made with the adjusted-R^2 (adj-R^2).

gradient boosting. The relative influence criterion gives the normalized empirical improvement as a result of including a particular predictor in the final model. The most contributing macroeconomic and technical analysis predictors are in line with the univariate results in Tables 4.3–4.4. As in Table 4.5, it is noteworthy how the final model is essentially based on different macroeconomic predictors.

4.3.4 In-sample extensions

As described in Section 4.2, our utility boosting method builds upon empirical utility maximization with the explicit linkage to the investor's preferences implemented via the sample objective function (4.10) and bounds (4.8). This leads to an important general point already validated in Section 4.3.3: Utility boosting is more relevant over the past two-step approach when the final interest is in asset allocation decisions. Empirically, this all of course indicate that the exact numbers, as already seen in Section 4.3.3 over the selections $\gamma = 5$ and $\gamma = 3$, might naturally be somewhat dependent on the risk aversion level, the maximum weight bound $w^{\max}$ and the volatility proxy σ_t^2. Therefore, in this section we still briefly present various additional and robustness analyses with detailed results compiled to the Appendix A.

In Section 4.3.3, we considered the setting where taking a leveraged position $w^{\max} > 1$ was not possible. Therefore, it is meaningful to also consider empirical results when the maximum weight is $w^{\max} = 1.5$. This has also been a rather common selection in the past predictive regression studies (see (ii) in Section 4.2.1). It turns out that the main findings on the best single predictors and differences between the utility boosting and the two-step approaches are essentially intact (see the Appendix A.1). Mainly DY, LTR and TMS seem to perform in relative sense even somewhat better when an investor has an access to 50% leverage (vs. the case $w^{\max} = 1$).

One important extension to the analysis presented in Section 4.3.3 is to impose transaction costs as a part of the analysis. It is, however, largely an open issue how much emphasis we should put on this view. In addition to the fact that transaction costs are seemingly becoming smaller and smaller all the time, from the methodological point of view imposing transaction costs means that otherwise optimal portfolio weight determination might be severely disrupted

by effectively unnecessary (continuous) portfolio rebalancing activity. As an example, is it optimal to rebalance the portfolio at all when moving from, say, $w_t = 0.75$ to $w_{t+1} = 0.7$, given the loss of portfolio value due to transaction costs? Following this lead, in the optimal case transaction costs should be part of the econometric procedure and utility maximization. This is not, however, clear cut to implement and requires additional complicated steps extending the utility boosting introduced in Sections 4.2.2–4.2.3. Therefore, in this study we content ourselves to the same view as in past return prediction studies: We evaluate the received portfolio returns when transaction costs are imposed on the evaluation stage, after constructing first the same 'optimal' asset allocation decisions as above.

Following Marquering and Verbeek (2004), in the Appendix A.2 we present the corresponding results as in Tables 4.3–4.4 but now also incorporating low and high transaction costs scenarios given as percentage points (low (0.1%) and high (0.5%)) of the value traded (see also Rossi, 2018, and the references therein). As in the various previous studies, utility gains naturally decrease due to transaction costs, but especially in the low transaction costs scenario the empirical results are still favourable for the utility boosting over the benchmarks. All in all, the results in Tables 4.3–4.4 can be interpreted as upper bound estimates for the received economic gains.

To get additional robustness for our findings, we also consider alternative volatility proxies to the well-established 5-year rolling window estimate. In the Appendix A.4., we consider the natural alternatives where the conditional mean of excess stock returns is constant and the GARCH(1,1) model equation is assumed for the conditional variance providing the volatility proxy σ_t^2. Another check is performed with the realized volatility series RVOL as described in Table 4.1 and Mele (2007). Again the exact numbers in the economic goodness-of-fit measures naturally differ and certain variables (mainly the term spread (TMS) and earnings-price ratio (EP)) perform somewhat better than in our main analysis, but the main conclusions are still intact.

Finally, the results in Tables 4.3–4.4, and more generally the coming Section 4.3.5, show that less persistent macroeconomic predictors perform relatively well in the utility boosting method. This suggests still to consider whether taking the first differences of the (highly) persistent predictors (i.e. all the variables except DFR, LTR and INFL) changes the big picture. It turns out that (see

the Appendix A.5), when concentrating on out-of-sample forecasting results as in the next section, especially the lagged changes in the dividend price ratio contain even higher predictive power than the level of the DP, emphasizing the conclusion that this theoretically well-motivated state variable contains useful information in direct portfolio weight determination.

4.3.5 Out-of-sample forecasting results

In this section, we report out-of-sample asset allocation results based on portfolio weights obtained with the (single-step) utility boosting and (two-step) linear predictive regression-based methods using the same macroeconomic and technical indicator predictors as in Sections 4.3.3–4.3.4. The portfolio weight for the month t is thus constructed using the information contained in $\boldsymbol{x}_{t-1}$ and the volatility proxy σ_t^2. This applies to both methods where, after training the respective weighting algorithms, the portfolio weight forecasts are based on the information available at time $t-1$ only.

Due to additional flexibility allowed by our utility boosting method over the simple linear predictive regressions, we believe that the initial and expanding estimation sample in out-of-sample forecasting should be slightly longer than in various past studies. Therefore, we use January 1951 to December 1989 as the first initial estimation training sample window to generate portfolio weights for January 1990. When moving the forecast origin ahead the initial estimation sample is expanding by one new observation when constructing portfolio weights one month ahead. The forecasting evaluation period hence contains the predicted weights from January 1990 to December 2018 (348 observations).

We analyze portfolio (asset allocation) performance in terms of received average utility, CER gain and the Sharpe ratio (see equations (4.17), (4.19) and (4.21)), all now defined for out-of-sample forecasting evaluation. These are the economic evaluation criteria of interest, enabling a comparison between different approaches to determine portfolio weights. In line with the past out-of-sample return predictability studies, we do not claim that a representative investor would have ended up exactly to the best performing model at each step. Instead and completely in line with the past studies, we are interested in general findings on the usefulness of our new method over the past ones in a

relatively large set of predictors.

The main interest and views to be considered in this section are at least the following ones, together with a comparison to the full sample results:

(a) To which extent our utility boosting method can outperform the benchmarks ('Const' and 'HA'), essentially claiming that there is no useful predictive information in the state variable (variables) $\boldsymbol{x}_{t-1}$? Specifically the historical average ('HA'), i.e. the constant expected equity premium forecast (see (4.18)), has been a popular and very stringent out-of-sample benchmark (see, e.g., Welch and Goyal, 2003, 2008; Campbell and Thompson, 2008; Neely et al., 2014). Linear predictive regressions containing individual (macroeconomic) variables typically fail to outperform this historical average statistically. Moreover, there is also a quite unanimous stylized fact in empirical finance that the time period since 1990 (as examined here) is the most challenging among the past decades in terms of out-of-sample predictability of stock returns (see, e.g., Campbell and Thompson, 2008; Lettau and Van Nieuwerburgh, 2008), presumably making it difficult to any new method to find meaningful additional predictive value.

(b) The post-truncation of the weights obtained with linear predictive regressions (4.4) to fulfil the bounds (4.8) has been found very useful in the past out-of-sample forecasting studies (see (ii) in Section 4.2.1). Therefore, this past 'truncated linear' approach is expected to be a tough and well-trained competitor to beat when the out-of-sample economic forecasting performance is of interest.

(c) Following the main findings of Neely et al. (2014), obtained with the past two-step method, technical indicators turned out to outperform macroeconomic predictors in terms of both statistical and economic out-of-sample evaluation criteria. Whether this is the case in our approach is also of particular interest.

Tables 4.7–4.8 report the out-of-sample forecasting results for the benchmarks and single-predictor models. In these results, we have assumed that the relative risk aversion coefficient γ is fixed to either $\gamma = 5$ (Table 4.7) or $\gamma = 3$ (Table 4.8) with the upper bound weight constraint set as $w^{\max} = 1$. As expected, the utility levels and Sharpe ratios are generally somewhat lower than in the in-sample results and that the historical average (HA) and the two-step linear approaches (with weight truncation to fulfil (4.8)) are indeed,

in accordance with the past studies, performing well and tough competitors to the utility boosting. There are also several candidate predictors that either the utility boosting or the two-step approach cannot find useful out-of-sample predictive power over the historical average weights (4.18).

As a whole, the results confirm that when a certain predictor has (some) reasonable out-of-sample asset allocation predictive power in this sample period, the Sharpe ratios are almost uniformly higher in the boosting approach over the alternative methods. That is, basing the econometric method on economic criteria produces higher risk-adjusted returns than the conventional statistical approaches. The results are also largely intact to the change of risk aversion level ($\gamma = 5$ and $\gamma = 3$).

In accordance with Neely et al. (2014), technical indicators perform as a group better than macroeconomic variables in out-of-sample forecasting purposes and the best (single) indicators are the same ones (i.e. MA(1,9), MA(2,12), VOL(1,12) and VOL(3,12)) as in the in-sample analysis, showing their robustness as predictors. In these cases, economic gains over the benchmarks are clear. For the MA(2,12) and VOL(3,12), the linear approach performs equally well as the utility boosting, which again points out the successfulness of the past truncated two-step approach. The simple binary nature of technical indicators is likely an important contributing factor for seemingly better out-of-sample performance over macroeconomic variables.[17] The latter ones are much more vulnerable to possible, and also arguably happened, time-varying instability issues in this period of interest, including, among others, the IT-bubble around the millennium and low interest rates at the end of the evaluation sample period.

Figure 4.2 illustrates the estimated out-of-sample portfolio weights for both methods using the well-performing technical indicator MA(1,9) (see Table 4.1). Like in the in-sample results (cf. Figure 4.1), there are periods where the weight truncation is indeed strongly taking place in the two-step statistical approach. It is noteworthy how these truncation periods can last even for several years.

As mentioned above and in line with past return predictability findings,

[17] Due to the binary nature of the variables, the superiority of the utility boosting is coming solely from the different objective function as non-linearities behind the construction of $F(\boldsymbol{x}_{t-1})$ in the boosting estimator play no role for binary-valued predictors.

there are only a few macroeconomic variables which seem to have reasonable out-of-sample predictive power in this sample period. From the utility boosting perspective, in line with the full sample results, inflation (INFL) and default return spread (DFR) are the best ones and perform relatively much better than in the 'linear' statistical approach. On the other hand, supposedly the well-documented erratic behaviour of the dividend-price ratio (DP) around the end of the 1990s and the beginning 2000s costs us in its performance when compared with the simple benchmarks. However, the utility boosting clearly outperforms the linear approach also with these predictors, as concluded already in the in-sample results. Moreover and importantly, when considering the monthly changes of the dividend-price ratio (ΔDP), instead of the levels, the utility boosting is able to find substantial out-of-sample predictive power (see details in the Appendix A.5).

Tables 4.7–4.8 present and concentrate on univariate (single-predictor) out-of-sample forecasting results. As mentioned, we do not claim that the best-performing predictors were known in advance. Therefore, multivariate out-of-sample analysis provides important information on the internal model selection capability of the utility boosting to find useful predictors at a time and without any prior knowledge. Table 4.9 reports the multivariate results with both cases of $\gamma = 5$ and $\gamma = 3$. The 'linear' multivariate benchmark is built upon the principal component analysis (PCA) as in Neely et al. (2014). The multivariate out-of-sample results are very much in line with the single-predictor as well as the in-sample results: Each of the utility boosting-based predictions from different groups of predictors outperform the benchmarks. Models based on macroeconomic variables (MACRO) yield only slightly higher utilities compared to the benchmarks, whereas technical indicators (TECH) is the best performing group of predictors also in this multivariate setting.

When comparing the multivariate utility boosting to the two-step predictive regressions (combined with the PCA), we can see that the former outperforms the latter for both MACRO and ALL (all the predictors combined). For the technical indicators (TECH), the performances of the two methods are essentially the same. This largely the same performance is also evident in Figure 4.3 where the estimated portfolio weights using TECH are graphically illustrated for both methods (given $\gamma = 5$). Despite the slightly higher utility

levels in the two-step approach, the risk-adjusted portfolio returns (and the Sharpe ratios) are higher in the utility boosting.

4.4 Discussion

The current stance in empirical finance appears that stock returns are time-varying and at least at times somewhat predictable. Whether these predictable statistical patterns translate to useful asset allocation decisions is arguably of the main interest for investors in practice instead of commonly used statistical criteria. However, the past empirical findings have been quite ambiguous in this respect.

Most academic interest and professional investment advice is so far directed to two-step plug-in approaches to find first useful (macroeconomic) predictive variables such as the dividend-price ratio and information contained in the term structure of interest rates, to predict excess stock returns with linear predictive regressions before constructing portfolio weights. In contrast to these premises, the utility boosting approach developed in this study establishes a direct relationship between the predictive variables and portfolio weights. This linkage is econometrically feasible when combined with a specifically designed customized gradient boosting algorithm to this objective, resulting in superior and less noisy portfolio performance than obtained with the conventional two-step statistical approach relying on attempts to predict weakly predictable stock returns. Albeit only experience on even more extensive empirical examinations will determine the extent to which utility boosting, and related advanced econometric approaches, actually improve investment decisions, we believe it offers a valuable prospect for dealing with a number of complicated aspects in asset allocation decisions in an integrated and single-step manner.

Brandt (1999), Aït-Sahalia and Brandt (2001), Brandt and Santa-Clara (2006), and Brandt et al. (2009) have already emphasized the advantages of focusing portfolio weights directly. Our findings, obtained with a very different econometric method and emphasizing prediction (forecasting) aspects, point out the advantages of this type of general thinking as well. The customized gradient boosting algorithm learns, likewise investors in their practical decisions, from portfolio weighting mistakes in the training stage

before selecting the portfolio weight for the next period. This importantly bypasses the intermediate construction of expected stock returns, which is the fundamental part of the two-step approach, involving hence greater risk for misspecification from the final asset allocation objective due to likely difficulties related to statistical determination of expected returns at most weakly predictable monthly excess stock returns of interest in this study.

The fact that the promising utility boosting approach also internally respects the pre-determined lower and upper bounds of the portfolio weights is continuum of the past out-of-sample return predictability studies pointing out the usefulness of theoretically motivated subsequent restrictions on the linear predictive regressions (see Campbell and Thompson, 2008; Rapach et al., 2010; Pettenuzzo et al., 2014, and the survey of Rapach and Zhou 2013). Imposing such post restrictions turn out to modify the equity premium and portfolio weight predictions substantially: See Figures 4.1–4.3 on both in- and out-of-sample evidence in this respect. This all implies that, even successful from the prediction perspective, the conventional statistical interpretations of predictive regressions are lost, while the utility boosting produces interpretable results on the importance of different predictors and their combinations. It is also noteworthy that our out-of-sample forecasting evaluation sample, starting from the beginning of 1990s, is the one that has been found very challenging from the prediction perspective (see, e.g., Campbell and Thompson, 2008; Lettau and Van Nieuwerburgh, 2008). This makes it particularly notable that the utility boosting produces substantial additional value over the past methods and benchmarks.

Our empirical results generally suggest that there is some but nowhere near one-to-one connection between conventional statistical criteria, such as t-values or adjusted-R^2, and the realized utility levels and economic goodness-of-fit criteria. This is in line with the arguments of Elliott and Timmermann (2016, chapter 2, and the references therein) and their call for a closer look to strengthen the linkage of the appropriate objective function and the employed econometric method. Our general findings coincide also with the conclusions of Leitch and Tanner (1991), Kandel and Stambaugh (1996), Xu (2004) and Cenesizoglu and Timmermann (2012), among others, that some poor return prediction models, producing even worse statistical (out-of-sample) forecasts than simple benchmarks, may add economic value when used to guide portfo-

lio decisions. In accordance with this view, our findings generally emphasize that the constant statistical mean predictability of equity premium over time is not necessary that some advanced econometric methods can be useful for investors' decision making. Similar argumentation is made when predicting the direction (sign) of stock returns (see, e.g., Pesaran and Timmermann, 1995; Christoffersen and Diebold, 2006) or in the recent findings of only episodic ('local' and time-varying) return predictability ('pockets of predictability') in stock returns and its relationship to asset pricing (see Farmer, Schmidt and Timmermann, 2018; Demetrescu et al., 2020).

We obtain several robust empirical conclusions in different empirical settings specified by the investor risk aversion levels, portfolio weight upper bounds and volatility proxies. First, as in Neely et al. (2014), technical indicators perform the best in out-of-sample forecasting and provide more stable performance in different settings than macroeconomic variables. Second, utility boosting mitigates excessive volatility in portfolio weights and resulting portfolio performance, leading to generally superior return-risk compromise over the past two-step statistical approach. Third, utility boosting backs certain macroeconomic variables in their relative performance versus the evidence in the predictive regression models. In particular, theoretically motivated inflation and the dividend-price ratio (cf. Welch and Goyal (2008) and the subsequent studies, surveyed by Rapach and Zhou (2013)), benefit substantially from the direct portfolio weight determination. Notably in the latter case, we find that specifically the changes, rather than the level, of the dividend-price ratio provides important information for portfolio weights.

As emphasized by, e.g., Ban, El Karoui and Lim (2018), and the references therein, several modern academic portfolio optimization models, with large cross-sections of assets, are intractable when applied to real data due to difficulties in estimation although they present solid theoretical properties. Ban et al. (2018) address this by adapting regularization and cross-validation approaches for portfolio optimization. We don't concentrate on large cross-sections of assets in this study: We are instead integrating ongoing advancement in machine learning and financial economics practice to the customized gradient boosting approach that can be extended to handle multiple risky assets, after some modifications and requiring a separate attempt. This advancement is also partly dependent on the development of boosting-based methods to

multiple-equation cases which are still largely non-existent in the machine learning literature.

Finally, to enable the utility boosting method, we need to assume some form for the underlying utility function. We rely on the quadratic form commonly used in forecast evaluation of relevant past (out-of-sample) return predictability studies. We are well aware of its limitations and we do not claim that the conditions behind the mean-variance analysis are strictly satisfied in practice. In the future research, not just to explore alternative utility schemes, another important connected point is to consider whether it is optimal to rebalance the portfolio at all in certain time points. This view is linked with the impact of transaction costs, larger cross-sections of assets and longer than one period investment horizon. These all extensions require additional steps to be taken with the utility boosting approach opened in this study.

4.5 Conclusions

In contrast to commonly used linear predictive regressions, we introduce a flexible utility-based empirical approach to directly determine asset allocation decisions in a simple setting between risky and risk-free assets. From our standpoint and diverging substantially from the usual financial economics perspective, whether stock returns are statistically predictable is not of main interest. Instead, we focus directly on the portfolio weights and their dependence on the predictive variables by maximizing a sample analogy of the utility function characterizing investors' preferences in their asset allocation decisions.

Our utility boosting approach arises from the synthesis between practices in financial economics and the recent advancements in machine learning. It builds upon a customized gradient boosting introduced in this study, selecting and combining predictive variables to form optimal portfolio weights designed specifically to that objective instead of using general textbook machine learning algorithms. Methodologically our approach contains built-in mechanisms circumventing overfitting, keeping the portfolio weights inside pre-specified bounds and not basing the method on the usual statistical significance testing framework to determine the usefulness of certain predictive variables in asset allocation decisions.

When applied to monthly U.S. excess stock returns, the utility boosting method generates superior and economically meaningful utility gains over the typical and commonly used benchmarks. These gains apply both full sample and out-of-sample predictions, providing systematically higher risk-adjusted portfolio returns than the past two-step statistical approach based on linear predictive regressions. We find that especially various technical indicators and some specific macroeconomic variables perform better in terms of portfolio performance than the benchmarks based on historical average stock returns when the objective is in the economic gains of asset allocations decisions.

References

Adler, T., and M. Kritzman (2007). Mean-variance versus full-scale optimisation: In and out of sample. *Journal of Asset Management* 7, 302–311.

Aït-Sahalia, Y., and M.W. Brandt (2001). Variable selection for portfolio choice. *Journal of Finance* 56(4), 1297–1351.

Audrino, F., and F. Trojani (2007). Accurate short-term yield curve forecasting using functional gradient descent. *Journal of Financial Econometrics* 5, 591–623

Ban, G-Y., El Karoui, N., and A.E.B. Lim (2018). Machine learning and portfolio optimization. *Management Science* 64(3), 1136–1154.

Barberis, N. (2000). Investing for the long run when returns are predictable. *Journal of Finance* 55, 225–264.

Brandt, M.W. (1999). Estimating portfolio and consumption choice: A conditional Euler equations approach. *Journal of Finance* 54(5), 1609–1645.

Brandt, M.W. (2010). Portfolio Choice Problems. Handbook of Financial Econometrics, Yacine Aït-Sahalia and Lars Hansen (eds), Vol. 1, Chapter 5.

Brandt, M.W., and P. Santa-Clara (2006). Dynamic portfolio selection by augmenting the asset space. *Journal of Finance* 61, 2187–2217.

Brandt, M.W., Santa-Clara, P., and R. Valkanov (2009). Parametric portfolio policies: Exploiting characteristics in the cross-section of equity returns. *Review of Financial Studies* 22(9), 3411–3447.

Bühlmann, P., and T. Hothorn (2007). Boosting algorithms: Regularization, prediction and model fitting. *Statistical Science* 22, 477–505.

Campbell J.Y., and S.B. Thompson (2008). Predicting excess stock returns out of sample: Can anything beat the historical average? *Review of Financial Studies* 21, 1509–1531.

Carmona, P., Climent, F., and A. Momparler (2019). Predicting failure in the U.S. banking sector: An extreme gradient boosting approach. *International Review of Economics & Finance* 61, 304–323.

Cenesizoglu T., and A. Timmermann (2012). Do return prediction models add economic value? *Journal of Banking & Finance* 36, 2974–2987.

Christoffersen, P.F., and F.X. Diebold (2006). Financial asset returns, direction-of-change forecasting, and volatility dynamics. *Management Science* 52, 1273–1287.

Demetrescu, M., Georgiev, I., Rodrigues, P.M.M., and A.M.R. Taylor (2020). Testing for episodic predictability in stock returns. *Journal of Econometrics*, in press.

Elliott, G., and A. Timmermann (2016). *Economic Forecasting*. Princeton University Press, Princeton and Oxford.

Farmer, L., Schmidt, L., and A. Timmermann (2018). Pockets of predictability. CEPR Discussion Paper 12885.

Fleming, J., Kirby, C., and B. Ostdiek (2001): The economic value of volatility timing. *Journal of Finance* 56, 329–352.

Friedman, J.H., Hastie, T., and R. Tibshirani (2000). Additive logistic regression: a statistical view of boosting (With discussion and a rejoinder by the authors). *Annals of Statistics* 28(2), 337–407.

Friedman, J.H. (2001). Greedy function approximation: A gradient boosting machine. *Annals of Statistics* 29(5), 1189–1232.

Friedman, J.H. (2002). Stochastic gradient boosting. *Computational Statistics & Data Analysis* 38(4), 367–378.

Goyal, A., and I. Welch (2003). Predicting the equity premium with dividend ratios. *Management Science* 49(5), 639–654.

Guidolin, M. (2011). Markov switching in portfolio choice and asset pricing models: A survey. *Advances in Econometrics* 27B, 87–178.

Guidolin, M., and A. Timmermann (2007). Asset allocation under multivariate regime switching. *Journal of Economic Dynamics and Control* 31, 3503–3544.

Inoue, A., and L. Kilian (2005). In-sample or out-of-sample tests of predictability: Which one should we use? *Econometric Reviews* 23, 371–402.

Kan, R., and G. Zhou (2007). Optimal portfolio choice with parameter uncertainty. *Journal of Financial and Quantitative Analysis* 42, 621–656.

Kandel, S., and R.F. Stambaugh (1996). On the predictability of stock returns: An asset-allocation perspective. *Journal of Finance* 51, 385–424.

Leitch, G., and J. Tanner (1991). Economic forecast evaluation: Profits versus the conventional error measures. *American Economic Review* 81, 580–590.

Lettau, M., and S. Van Nieuwerburgh (2008). Reconciling the return predictability evidence. *Review of Financial Studies* 21(4), 1607–1652.

Loh, W-Y., and Y-S. Shih (1997). Split selection methods for classification trees. *Statistica Sinica* 7(4), 815–840.

Marquering, W., and M. Verbeek (2004). The economic value of predicting stock index returns and volatility. *Journal of Financial and Quantitative Analysis* 39, 407–429.

Mele, A. (2007). Asymmetric stock market volatility and the cyclical behavior of expected returns. *Journal of Financial Economics* 86, 446–478.

Mittnik, S., Robinzonov, N., and M. Spindler (2015). Stock market volatility: Identifying major drivers and the nature of their impact. *Journal of Banking & Finance* 58, 1–14.

Neely, C.J., Rapach, D.E., Tu, J., and G. Zhou (2014). Forecasting the equity risk premium: The role of technical indicators. *Management Science* 60, 1772–1791.

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

Pettenuzzo D., Timmermann, A., and R. Valkanov (2014). Forecasting stock returns under economic constraints. *Journal of Financial Economics* 114, 517–553.

Rapach D., and G. Zhou (2013). Forecasting stock returns. In Handbook of Economic Forecasting, G. Elliott and A. Timmermann (eds), vol. 2A, North-Holland.

Rapach, D.E., Strauss, J.K., and G. Zhou (2010). Out-of-sample equity premium prediction: Combination forecasts and links to the real economy. *Review of Financial Studies* 23, 821–862.

Rossi, A. (2018). Predicting stock market returns with machine learning. Manuscript, University of Maryland (August 2018).

Rossi, A.G., and A. Timmermann (2015). Modeling covariance risk in Merton's ICAPM. *Review of Financial Studies* 28, 1429–1461.

Sentana, E. (2005). Least squares predictions and mean-variance analysis. *Journal of Financial Econometrics* 5, 56–78.

Stambaugh, R.F. (1999). Predictive regressions. *Journal of Financial Economics* 54, 375–421.

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

Xu, Y. (2004). Small levels of predictability and large economic gains. *Journal of Empirical Finance* 11, 247–275.

Zhu, X. (2015). Tug-of-war: Time-varying predictability of stock returns and dividend growth. *Review of Finance* 19, 2317–2358.

Tables and Figures

Table 4.1: Predictive variables.

	Panel A: Macroeconomic variables
DP	Log dividend-price ratio $\log(D/P)$
DY	Log dividend yield $\log(D/Y)$, where Y is the lagged P
EP	Log earnings-price ratio $\log(E/P)$
DE	Log dividend-payout ratio $\log(D/E)$
RVOL	Equity risk premium volatility, 12-month moving standard deviation (Mele, 2007)
BM	Book-to-market value ratio for the DJIA (Dow Jones Industrial Average)
NTIS	Net equity expansion
TBL	Treasury bill rate (three-month Treasury bill, secondary market)
LTY	Long-term government bond yield
LTR	Return on long-term government bonds
TMS	Term spread: LTY-TBL
DFY	Default yield spread
DFR	Default return spread
INFL	Inflation (CPI inflation), lagged by one period due to the delay in CPI releases.
	Panel B: Technical indicators
MA(1,9)	Moving average indicator (4.14) with $s = 1$ and $l = 9$
MA(1,12)	Moving average indicator (4.14) with $s = 1$ and $l = 12$
MA(2,9)	Moving average indicator (4.14) with $s = 2$ and $l = 9$
MA(2,12)	Moving average indicator (4.14) with $s = 2$ and $l = 12$
MA(3,9)	Moving average indicator (4.14) with $s = 3$ and $l = 9$
MA(3,12)	Moving average indicator (4.14) with $s = 3$ and $l = 12$
MOM(9)	Momentum indicator (4.15) with $m = 9$
MOM(12)	Momentum indicator (4.15) with $m = 12$
VOL(1,9)	Volume indicator (4.16) with $s = 1$ and $l = 9$
VOL(1,12)	Volume indicator (4.16) with $s = 1$ and $l = 12$
VOL(2,9)	Volume indicator (4.16) with $s = 2$ and $l = 9$
VOL(2,12)	Volume indicator (4.16) with $s = 2$ and $l = 12$
VOL(3,9)	Volume indicator (4.16) with $s = 3$ and $l = 9$
VOL(3,12).	Volume indicator (4.16) with $s = 3$ and $l = 12$

Notes: D and E refer to log of a 12-month moving sum of dividends paid (D) and earnings (E) on the S&P 500 index (P). Net equity expansion (NTIS) is a ratio of a 12-month moving sum of net equity issues by NYSE-listed stocks to the total end-of-year market capitalization of NYSE stocks. Default yield spread (DFY) is the difference between BAA and AAA-rated corporate bond yields, whereas the default spread (DFR) is defined as the difference between corporate bond return minus LTR. Following Welch and Goyal (2008), and subsequent follow-up studies, inflation is the Consumer Price Index (All Urban Consumers) and we consider its first lag as inflation information is released only in the following month. Binary-valued technical indicators are defined in (4.14)–(4.16) with different price window lengths.

Table 4.2: Descriptive statistics of excess stock returns ($r_{e,t}$) and macroeconomic predictive variables.

Variable	Mean	Std	Min	Max	$\widehat{\rho}_1$	$\widehat{\rho}_2$	$\widehat{\rho}_3$
Panel A: Excess stock returns							
$r_{e,t}$	0.60	4.14	-22.11	16.14	0.04	-0.03	0.04
Panel B: Macroeconomic variables							
DP	-3.53	0.42	-4.52	-2.60	0.99	0.98	0.97
DY	-3.53	0.42	-4.53	-2.59	0.99	0.98	0.97
EP	-2.80	0.42	-4.84	-1.90	0.99	0.97	0.94
DE	-0.73	0.29	-1.24	1.38	0.99	0.95	0.90
RVOL	0.002	0.001	0.0002	0.008	0.96	0.91	0.86
BM	0.51	0.25	0.12	1.21	0.99	0.99	0.98
NTIS	0.01	0.02	-0.06	0.05	0.98	0.95	0.93
TBL	4.25	3.08	0.01	16.30	0.99	0.98	0.96
LTY	5.95	2.76	1.75	14.82	0.99	0.99	0.98
LTR	0.52	2.75	-11.24	15.23	0.04	-0.07	-0.02
TMS	1.70	1.38	-3.65	4.55	0.96	0.91	0.86
DFY	0.96	0.44	0.32	3.38	0.97	0.92	0.88
DFR	0.02	1.40	-9.75	7.37	-0.08	-0.09	-0.02
INFL	0.29	0.36	-1.92	1.81	0.55	0.39	0.27

Notes: This table presents descriptive statistics for the excess stock returns (Panel A, i.e. simple equity risk premium) in percent (i.e. returns multiplied by 100) and 14 macroeconomic variables (Panel B) where LTR, DFR, and INFL (TBL, LTY, TMS, and DFY) are measured in percent (annual percent). The sample period is December 1950–December 2018. The first three sample autocorrelation coefficients of each variable are denoted by $\widehat{\rho}_1$, $\widehat{\rho}_2$ and $\widehat{\rho}_3$.

Table 4.3: In-sample results for different predictors with $\gamma = 5$ and $w^{\max} = 1$.

	Utility boosting			Linear, constrained weights			
Variable	Util(%)	CER(%)	Sharpe	Util(%)	Sharpe	t-val	adj-R^2
Panel A: Macroeconomic variables							
DP	0.684	1.706	0.185	0.581	0.137	1.76	0.28
DY	0.731	2.296	0.201	0.581	0.138	1.87	0.33
EP	0.656	1.573	0.181	0.569	0.135	0.77	0.04
DE	0.593	0.727	0.159	0.587	0.139	0.69	-0.02
RVOL	0.730	2.355	0.206	0.601	0.137	2.80	0.63
BM	0.718	2.101	0.202	0.568	0.132	0.58	-0.07
NTIS	0.661	1.546	0.180	0.583	0.138	0.27	-0.10
TBL	0.668	1.617	0.185	0.634	0.164	2.26	0.55
LTY	0.634	1.268	0.181	0.606	0.150	1.49	0.18
LTR	0.727	2.249	0.197	0.659	0.164	2.68	0.69
TMS	0.695	1.914	0.192	0.657	0.169	1.90	0.41
DFY	0.603	0.816	0.163	0.587	0.138	0.38	-0.08
DFR	0.715	2.455	0.208	0.602	0.152	0.96	0.08
INFL	0.687	1.945	0.192	0.578	0.139	0.35	-0.09
Panel B: Technical indicators							
MA(1,9)	0.634	1.557	0.181	0.613	0.166	1.63	0.29
MA(1,12)	0.656	1.895	0.190	0.653	0.184	1.97	0.55
MA(2,9)	0.642	1.631	0.184	0.628	0.172	1.83	0.39
MA(2,12)	0.676	2.149	0.197	0.677	0.193	2.34	0.76
MA(3,9)	0.651	1.745	0.185	0.642	0.176	1.78	0.41
MA(3,12)	0.613	1.217	0.170	0.594	0.153	1.02	0.08
MOM(9)	0.618	1.294	0.173	0.598	0.156	1.11	0.11
MOM(12)	0.618	1.273	0.171	0.596	0.156	1.10	0.12
VOL(1,9)	0.624	1.358	0.175	0.593	0.154	1.30	0.16
VOL(1,12)	0.640	1.593	0.181	0.623	0.170	1.73	0.39
VOL(2,9)	0.613	1.236	0.171	0.598	0.157	1.31	0.18
VOL(2,12)	0.619	1.331	0.174	0.610	0.164	1.49	0.29
VOL(3,9)	0.606	1.112	0.167	0.585	0.149	0.95	0.04
VOL(3,12)	0.651	1.692	0.183	0.643	0.175	1.93	0.52
Panel C: Benchmarks							
Const	0.570	0.264	0.141				
HA	0.570		0.135				

Notes: This table reports portfolio performance measures for an investor, with the utility function (4.2) and the relative risk-aversion coefficient of five ($\gamma = 5$), who allocates portfolio value between risky market portfolio and risk-free rate in each month. The bounds for portfolio weights are $w^{\min} = 0$ and $w^{\max} = 1$. The reported predictions are from the customized gradient boosting (Section 4.2.3) with different single macroeconomic (Panel A) or technical indicators (Panel B) in $\boldsymbol{x}_{t-1}$ at a time and two benchmarks ('Const' refers to the case, where $\boldsymbol{x}_{t-1} = 1$ and 'HA' to the historical average weights (4.18)). On the right, we report the results of the linear predictive regressions ('Linear') with the subsequent weight constraints (4.8). 'Util(%)' refers to the realized utility (4.17), CER(%) is the CER gain (4.19). Finally, adj-R^2 is the adjusted-R^2 and 't-val' is the heteroskedasticity-autocorrelation consistent (robust) t-statistic for the null hypothesis $\beta_i = 0$ in the predictive regression (4.9).

Table 4.4: In-sample results for different predictors with $\gamma = 3$ and $w^{\max} = 1$.

	Utility boosting			Linear, constrained weights			
Variable	Util(%)	CER(%)	Sharpe	Util(%)	Sharpe	t-val	adj-R^2
Panel A: Macroeconomic variables							
DP	0.814	1.526	0.187	0.711	0.147	1.76	0.28
DY	0.858	2.084	0.202	0.705	0.146	1.87	0.33
EP	0.739	0.760	0.173	0.698	0.145	0.77	0.04
DE	0.701	0.209	0.159	0.687	0.139	0.69	-0.02
RVOL	0.829	1.801	0.201	0.698	0.139	2.80	0.63
BM	0.823	1.628	0.183	0.702	0.143	0.58	-0.07
NTIS	0.779	1.209	0.178	0.702	0.143	0.27	-0.10
TBL	0.897	2.694	0.212	0.728	0.160	2.26	0.55
LTY	0.789	1.340	0.192	0.698	0.148	1.49	0.18
LTR	0.870	2.290	0.205	0.771	0.167	2.68	0.69
TMS	0.921	2.958	0.223	0.777	0.171	1.90	0.41
DFY	0.706	0.255	0.163	0.705	0.144	0.38	-0.08
DFR	0.879	2.654	0.221	0.721	0.155	0.96	0.08
INFL	0.881	2.577	0.210	0.693	0.142	0.35	-0.09
Panel B: Technical indicators							
MA(1,9)	0.739	1.061	0.181	0.700	0.161	1.63	0.29
MA(1,12)	0.766	1.405	0.190	0.751	0.185	1.97	0.55
MA(2,9)	0.747	1.110	0.183	0.717	0.169	1.83	0.39
MA(2,12)	0.786	1.650	0.197	0.780	0.195	2.34	0.76
MA(3,9)	0.756	1.211	0.184	0.722	0.169	1.78	0.41
MA(3,12)	0.716	0.680	0.169	0.685	0.147	1.02	0.08
MOM(9)	0.721	0.795	0.172	0.683	0.148	1.11	0.11
MOM(12)	0.723	0.778	0.171	0.687	0.149	1.10	0.12
VOL(1,9)	0.724	0.788	0.174	0.679	0.148	1.30	0.16
VOL(1,12)	0.746	1.106	0.181	0.719	0.169	1.73	0.39
VOL(2,9)	0.712	0.590	0.168	0.699	0.154	1.31	0.18
VOL(2,12)	0.729	0.892	0.175	0.717	0.165	1.49	0.29
VOL(3,9)	0.707	0.581	0.167	0.684	0.145	0.95	0.04
VOL(3,12)	0.759	1.230	0.184	0.745	0.178	1.93	0.52
Panel C: Benchmarks							
Const	0.698	0.182	0.144				
HA	0.695		0.141				

Notes: See the notes to Table 4.3.

Table 4.5: In-sample results with different predictor groups.

	Utility boosting			Linear (PCA), constrained weights		
Variables	Util(%)	CER(%)	Sharpe	Util(%)	Sharpe	adj-R^2
Panel A: Selections $\gamma = 5$ and $w^{\max} = 1$						
MACRO	0.975	5.480	0.303	0.648	0.166	0.54
TECH	0.680	2.144	0.199	0.640	0.178	0.45
ALL	0.978	5.541	0.306	0.671	0.184	0.92
Panel B: Selections $\gamma = 3$ and $w^{\max} = 1$						
MACRO	1.066	4.882	0.293	0.742	0.160	0.54
TECH	0.835	2.215	0.211	0.768	0.187	0.45
ALL	1.070	4.895	0.293	0.782	0.187	0.92

Notes: In the utility boosting, all the macroeconomic variables (MACRO) and technical indicators (TECH) or both of them (ALL) are simultaneously included in $\boldsymbol{x}_{t-1}$ and the customized algorithm includes only the best of them in the final predictive models (see also Table (4.6)). In the linear predictive regressions, the principal components of the predictor groups are first constructed (e.g., due to the perfect multicollinearity between the variables), combined with weight restrictions (4.8). The amount of principal components is chosen according to adjusted R^2 as in Neely et al. (2014).

Table 4.6: Top-10 in-sample predictors in the multivariate utility boosting models with $\gamma = 5$ and $w^{\max} = 1$.

MACRO		TECH		ALL	
Variable	Rel.inf(%)	Variable	Rel.inf(%)	Variable	Rel.inf(%)
LTR	0.158	MA(2,12)	0.432	LTR	0.159
TMS	0.148	VOL(1,12)	0.120	TMS	0.134
NTIS	0.130	VOL(3,12)	0.108	NTIS	0.111
DFR	0.101	VOL(1,9)	0.051	DFR	0.093
RVOL	0.090	MA(2,9)	0.050	RVOL	0.089
EP	0.074	MA(3,12)	0.042	EP	0.068
DY	0.060	MA(1,9)	0.041	DY	0.065
BM	0.051	MA(3,9)	0.036	BM	0.054
TBL	0.046	VOL(3,9)	0.035	TBL	0.048
DE	0.034	VOL(2,12)	0.026	DE	0.031

Notes: For more information on the relative influence measure, see Friedman (2001).

Table 4.7: Out of-sample predictive results with $\gamma = 5$ and $w^{\max} = 1$.

	Utility boosting			Linear, constrained weights		
Variable	Util(%)	CER(%)	Sharpe	Util(%)	CER(%)	Sharpe
Panel A: Macroeconomic variables						
DP	0.427	0.058	0.133	0.245	-2.325	0.032
DY	0.420	-0.017	0.131	0.259	-2.032	0.041
EP	0.412	0.358	0.150	0.499	0.948	0.167
DE	0.371	-0.236	0.124	0.299	-1.600	0.085
RVOL	0.444	0.302	0.141	0.382	-1.123	0.104
BM	0.376	-0.644	0.110	0.412	-0.562	0.115
NTIS	0.451	0.656	0.152	0.372	-0.751	0.114
TBL	0.423	0.082	0.135	0.472	0.746	0.155
LTY	0.433	0.507	0.148	0.463	0.298	0.142
LTR	0.421	0.019	0.133	0.413	-0.453	0.122
TMS	0.420	0.193	0.138	0.430	0.268	0.144
DFY	0.405	-0.191	0.125	0.417	-0.399	0.122
DFR	0.472	1.095	0.172	0.462	0.553	0.148
INFL	0.450	0.445	0.145	0.391	-0.647	0.117
Panel B: Technical indicators						
MA(1,9)	0.511	1.805	0.193	0.474	1.056	0.164
MA(1,12)	0.496	1.769	0.187	0.528	2.065	0.197
MA(2,9)	0.494	1.745	0.189	0.478	1.349	0.174
MA(2,12)	0.542	2.437	0.209	0.565	2.580	0.214
MA(3,9)	0.470	1.319	0.171	0.446	0.869	0.158
MA(3,12)	0.470	1.184	0.172	0.453	0.765	0.155
MOM(9)	0.484	1.402	0.178	0.487	1.211	0.169
MOM(12)	0.485	1.392	0.178	0.486	1.130	0.166
VOL(1,9)	0.492	1.513	0.181	0.442	0.654	0.152
VOL(1,12)	0.527	2.004	0.195	0.484	1.372	0.175
VOL(2,9)	0.461	1.137	0.169	0.446	0.759	0.155
VOL(2,12)	0.495	1.507	0.181	0.504	1.417	0.175
VOL(3,9)	0.474	1.177	0.171	0.448	0.500	0.147
VOL(3,12)	0.535	2.056	0.198	0.547	2.043	0.195
Panel C: Benchmarks						
Const	0.435	0.222	0.139			
HA	0.445		0.133			

Notes: See the notes to Table 4.3.

Table 4.8: Out of-sample predictive results with $\gamma = 3$ and $w^{\max} = 1$.

	Utility boosting			Linear, constrained weights		
Variable	Util(%)	CER(%)	Sharpe	Util(%)	CER(%)	Sharpe
Panel A: Macroeconomic variables						
DP	0.525	-0.408	0.137	0.296	-3.324	0.051
DY	0.478	-0.973	0.122	0.289	-3.389	0.048
EP	0.526	-0.118	0.155	0.589	0.514	0.165
DE	0.426	-1.394	0.112	0.438	-1.423	0.109
RVOL	0.509	-0.641	0.129	0.523	-0.595	0.127
BM	0.507	-0.681	0.134	0.508	-0.704	0.127
NTIS	0.545	0.073	0.150	0.511	-0.428	0.131
TBL	0.561	0.222	0.147	0.590	0.689	0.153
LTY	0.519	-0.326	0.140	0.581	0.484	0.150
LTR	0.533	-0.157	0.140	0.505	-0.603	0.128
TMS	0.528	-0.108	0.141	0.608	0.869	0.158
DFY	0.456	-1.275	0.114	0.541	-0.122	0.138
DFR	0.594	0.847	0.179	0.562	0.441	0.154
INFL	0.597	0.640	0.163	0.527	-0.253	0.135
Panel B: Technical indicators						
MA(1,9)	0.609	1.190	0.186	0.557	0.548	0.159
MA(1,12)	0.609	1.352	0.192	0.621	1.649	0.196
MA(2,9)	0.595	1.132	0.184	0.551	0.625	0.163
MA(2,12)	0.654	1.959	0.212	0.660	2.151	0.212
MA(3,9)	0.575	0.809	0.174	0.508	0.100	0.149
MA(3,12)	0.555	0.439	0.163	0.522	0.094	0.147
MOM(9)	0.596	1.023	0.180	0.544	0.425	0.157
MOM(12)	0.601	1.039	0.178	0.559	0.516	0.158
VOL(1,9)	0.583	0.887	0.179	0.523	0.130	0.149
VOL(1,12)	0.616	1.349	0.191	0.566	0.839	0.171
VOL(2,9)	0.565	0.653	0.171	0.540	0.374	0.156
VOL(2,12)	0.603	1.093	0.179	0.594	1.011	0.171
VOL(3,9)	0.543	0.255	0.156	0.528	0.020	0.144
VOL(3,12)	0.650	1.663	0.200	0.638	1.650	0.192
Panel C: Benchmarks						
Const	0.563	0.320	0.146			
HA	0.551		0.141			

Notes: See the notes to Table 4.3.

Table 4.9: Out of-sample predictive results for predictor groups with $\gamma = 5$ or $\gamma = 3$ and $w^{\max} = 1$.

	Utility boosting			Linear (PCA), constrained weights		
Variables	Util(%)	CER(%)	Sharpe	Util(%)	CER(%)	Sharpe
Panel A: Selections $\gamma = 5$ and $w^{\max} = 1$						
MACRO	0.480	1.075	0.164	0.269	-1.658	0.075
TECH	0.513	1.840	0.191	0.522	1.885	0.191
ALL	0.487	1.273	0.171	0.348	-0.136	0.132
Panel B: Selections $\gamma = 3$ and $w^{\max} = 1$						
MACRO	0.566	0.322	0.159	0.377	-1.932	0.097
TECH	0.628	1.561	0.196	0.631	1.717	0.195
ALL	0.571	0.441	0.164	0.441	-0.729	0.148

Notes: See the notes to Table 4.5.

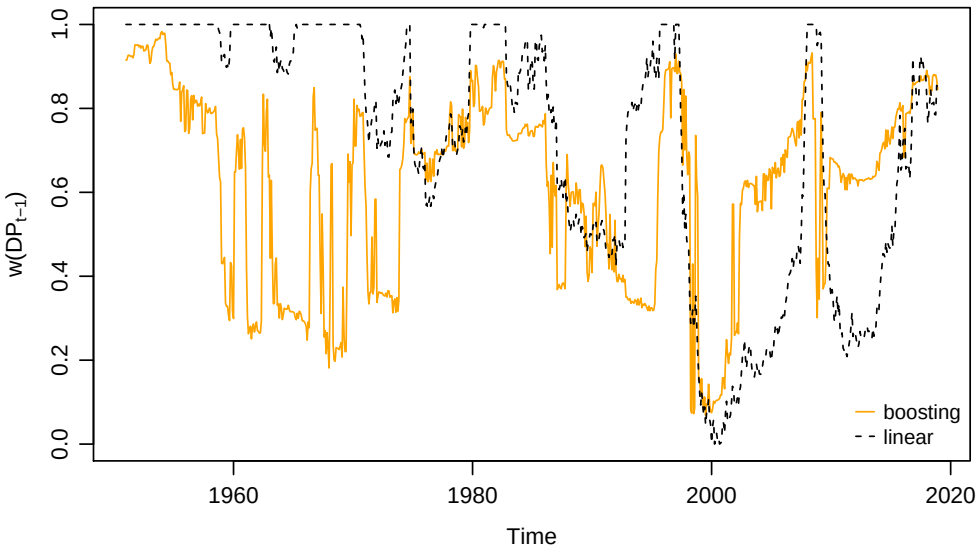

Figure 4.1: In-sample portfolio weights using the predictor DP (dividend-price ratio).

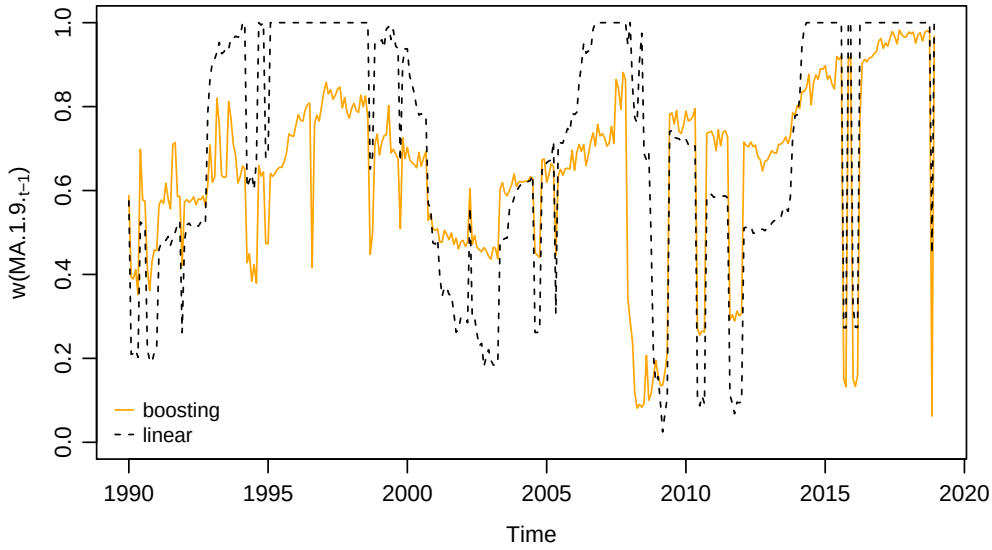

Figure 4.2: Out of-sample portfolio weights using the predictor MA(1,9).

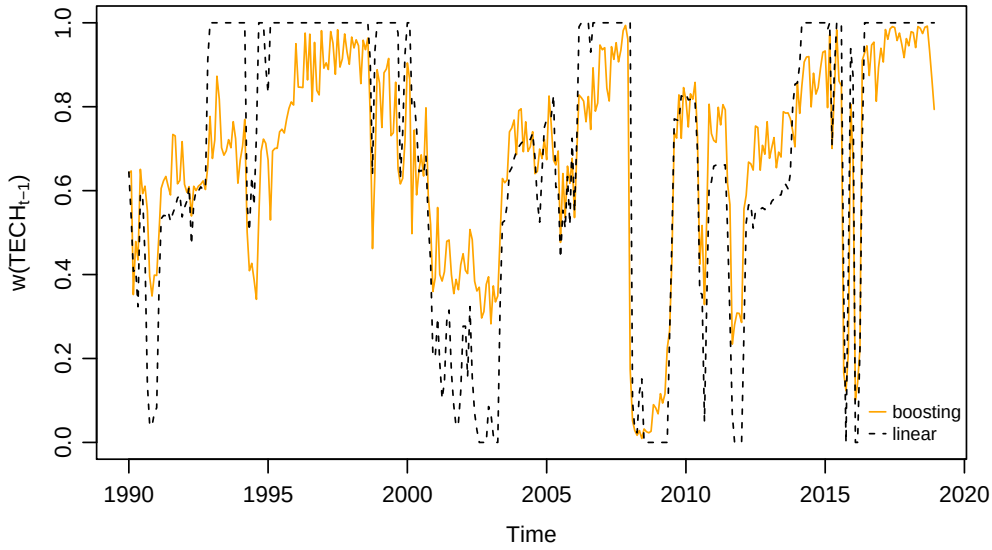

Figure 4.3: Out of-sample portfolio weights using the predictor group TECH.

Appendix A: Additional empirical results

This Appendix presents several additional analyses and robustness checks for the main empirical results reported in Section 4.3 of the main paper. If not otherwise mentioned, we consider the same monthly sample period and constructed volatility proxies as in Section 4.3. In Section A.1, we explore the impact of allowing for leveraged positions while Section A.2 concentrates on the results when low and high transaction costs scenarios are also imposed on the evaluation stage. Section A.3 contains alternative setting of the customized gradient boosting algorithm where the regression tree-based base learner function is replaced by a spline-based learner. Section A.4 examines the robustness of our findings on the selected volatility proxy by utilizing the GARCH model and a shorter-horizon realized variance as volatility proxies. Finally, Section A.5 considers the predictive power of monthly changes in the highly persistent macroeconomic predictor variables, such as the dividend-price ratio, instead of their levels.

A.1 Leveraged weights

In Tables A.1.1 and A.1.2, we report the results when the upper bound for the portfolio weights is set to $w^{\max} = 1.5$. This allows an investor to take leverage in his or her positions. When comparing to the results of Tables 4.3-4.4 and 4.7-4.8 in the main analysis, the essential conclusions are intact (with some minor predictor-specific differences).

Both in- and out-of-sample results reflect the fact that the utility boosting approach takes into account the risk awareness in estimation. Naturally the utility boosting benefits less than the two-step 'linear' approach from the ability to take leverage as risky leveraged positions are less frequently taken when $\gamma = 5$ (in the latter the upper bound $w^{\max} = 1.5$, and even levels above that, are often reached without restrictions on the maximum portfolio weight). However, the Sharpe ratios are consistently higher in the utility boosting. With the higher tolerance to risk ($\gamma = 3$), riskier positions are allowed and hence resulting utilities are then naturally also somewhat higher.

Table A.1.1: In-sample results of predictor-specific models with $\gamma = 5$ and $\gamma = 3$ when $w^{\max} = 1.5$.

	$\gamma = 5$				$\gamma = 3$			
	Utility boosting		Linear, constr. weights		Utility boosting		Linear, constr. weights	
Variable	Util(%)	Sharpe	Util(%)	Sharpe	Util(%)	Sharpe	Util(%)	Sharpe
Panel A: Macroeconomic variables								
DP	0.677	0.177	0.587	0.136	0.929	0.191	0.738	0.141
DY	0.831	0.223	0.595	0.141	0.999	0.208	0.734	0.140
EP	0.716	0.193	0.593	0.138	0.848	0.181	0.717	0.138
DE	0.636	0.170	0.573	0.127	0.806	0.172	0.740	0.141
RVOL	0.695	0.185	0.616	0.136	1.114	0.243	0.758	0.138
BM	0.719	0.186	0.566	0.128	0.890	0.183	0.712	0.135
NTIS	0.668	0.178	0.576	0.129	0.869	0.182	0.739	0.141
TBL	0.782	0.212	0.642	0.155	0.877	0.187	0.808	0.165
LTY	0.694	0.193	0.600	0.140	0.866	0.195	0.766	0.151
LTR	0.803	0.215	0.688	0.161	1.059	0.220	0.841	0.164
TMS	0.768	0.206	0.642	0.155	1.139	0.245	0.846	0.170
DFY	0.650	0.170	0.565	0.124	0.823	0.175	0.745	0.141
DFR	0.652	0.180	0.593	0.144	1.069	0.237	0.766	0.155
INFL	0.690	0.187	0.571	0.128	0.931	0.197	0.733	0.142
Panel B: Technical indicators								
MA(1,9)	0.643	0.182	0.633	0.164	0.810	0.181	0.770	0.164
MA(1,12)	0.667	0.190	0.672	0.179	0.845	0.190	0.837	0.184
MA(2,9)	0.651	0.183	0.649	0.171	0.824	0.184	0.797	0.172
MA(2,12)	0.691	0.198	0.703	0.190	0.876	0.197	0.874	0.194
MA(3,9)	0.662	0.185	0.664	0.174	0.836	0.185	0.816	0.175
MA(3,12)	0.616	0.171	0.605	0.150	0.777	0.170	0.741	0.152
MOM(9)	0.622	0.173	0.612	0.153	0.784	0.172	0.745	0.153
MOM(12)	0.622	0.172	0.612	0.153	0.786	0.171	0.745	0.154
VOL(1,9)	0.633	0.176	0.617	0.154	0.793	0.174	0.737	0.152
VOL(1,12)	0.648	0.181	0.643	0.167	0.820	0.181	0.788	0.169
VOL(2,9)	0.617	0.172	0.606	0.153	0.780	0.172	0.752	0.156
VOL(2,12)	0.622	0.174	0.611	0.156	0.792	0.176	0.773	0.164
VOL(3,9)	0.609	0.168	0.597	0.146	0.766	0.167	0.730	0.148
VOL(3,12)	0.659	0.182	0.654	0.169	0.838	0.183	0.824	0.176
Panel C: Benchmarks								
Const	0.564	0.144			0.715	0.142		
HA	0.564	0.127			0.714	0.137		

Notes: See the notes to Tables 4.3 and 4.4 in the body text.

Table A.1.2: Out of-sample results of predictor-specific models with $\gamma = 5$ and $\gamma = 3$ when $w^{\max} = 1.5$.

	$\gamma = 5$				$\gamma = 3$			
	Utility boosting		Linear, constr. weights		Utility boosting		Linear, constr. weights	
Variable	Util(%)	Sharpe	Util(%)	Sharpe	Util(%)	Sharpe	Util(%)	Sharpe
Panel A: Macroeconomic variables								
DP	0.403	0.125	0.245	0.032	0.574	0.137	0.255	0.031
DY	0.383	0.119	0.259	0.041	0.563	0.135	0.280	0.041
EP	0.404	0.145	0.510	0.162	0.547	0.150	0.665	0.167
DE	0.339	0.126	0.228	0.067	0.393	0.103	0.383	0.092
RVOL	0.414	0.135	0.381	0.097	0.578	0.139	0.502	0.110
BM	0.421	0.124	0.398	0.102	0.487	0.116	0.536	0.118
NTIS	0.441	0.152	0.364	0.109	0.588	0.148	0.489	0.119
TBL	0.363	0.131	0.412	0.134	0.620	0.152	0.627	0.155
LTY	0.436	0.156	0.441	0.126	0.520	0.135	0.615	0.144
LTR	0.427	0.139	0.439	0.122	0.547	0.134	0.525	0.122
TMS	0.425	0.146	0.345	0.125	0.566	0.143	0.590	0.148
DFY	0.432	0.136	0.351	0.092	0.436	0.103	0.556	0.128
DFR	0.471	0.167	0.460	0.136	0.659	0.179	0.618	0.152
INFL	0.417	0.137	0.368	0.105	0.622	0.153	0.514	0.122
Panel B: Technical indicators								
MA(1,9)	0.514	0.188	0.509	0.160	0.709	0.196	0.627	0.164
MA(1,12)	0.531	0.197	0.560	0.187	0.680	0.193	0.706	0.197
MA(2,9)	0.506	0.189	0.515	0.172	0.675	0.190	0.628	0.173
MA(2,12)	0.561	0.207	0.609	0.206	0.748	0.213	0.763	0.214
MA(3,9)	0.455	0.166	0.470	0.154	0.636	0.176	0.572	0.156
MA(3,12)	0.466	0.169	0.480	0.149	0.645	0.176	0.586	0.154
MOM(9)	0.482	0.175	0.516	0.161	0.685	0.186	0.635	0.167
MOM(12)	0.476	0.172	0.515	0.159	0.689	0.185	0.638	0.165
VOL(1,9)	0.487	0.177	0.475	0.150	0.658	0.181	0.572	0.151
VOL(1,12)	0.509	0.183	0.520	0.170	0.700	0.193	0.634	0.173
VOL(2,9)	0.449	0.164	0.460	0.147	0.632	0.175	0.582	0.155
VOL(2,12)	0.486	0.176	0.523	0.163	0.689	0.185	0.670	0.175
VOL(3,9)	0.463	0.165	0.463	0.139	0.635	0.171	0.583	0.147
VOL(3,12)	0.543	0.194	0.566	0.179	0.736	0.199	0.742	0.196
Panel C: Benchmarks								
Const	0.413	0.138			0.578	0.141		
HA	0.438	0.120			0.588	0.136		

Notes: See the notes to Tables 4.3 and 4.4 in the body text.

A.2 Transaction costs involved

In the following Tables A.2.1 and A.2.2, we impose transaction costs along the lines of Neely et al. (2014) in the evaluation stage when computing the resulting portfolio returns and the values of different evaluation metrics. We consider two transaction costs scenarios (see Marquering and Verbeek, 2004; Rossi, 2018): Low transaction costs case means that 0.1% of the value traded is lost as a result of rebalancing portfolio weights whereas high transaction costs corresponds the case of 0.5%.

As discussed in Section 4.3.3, here we take the standpoint that asset allocation decisions are first determined at each step so that the potential impact of transaction costs is not taken into account. This corresponds the idea of continuous portfolio rebalancing. Integrating transaction costs to optimal portfolio weight determination requires several additional steps and detailed examination, which are hence left for the future research. First of all, to determine whether it is optimal to update the portfolio weight from period t to $t+1$ at all needs to be addressed. As discussed in the main text, say that the portfolio weight for the risky asset is 0.7 at time t (i.e. $w_t = 0.7$) and then the optimal prediction (not acknowledging the impact of transaction costs) is to change it to 0.75 for the period $t+1$. Is this change of 0.05 large enough that we should really rebalance the portfolio again or should we just continue with the weight 0.7? Seemingly, as also pointed out already by Brandt (1999), the impact of the investment horizon must also be addressed simultaneously when addressing this question, whereas throughout this study we concentrate on the one-month horizon.

The above points show that the potential impact of the transaction costs in various stages of our analysis is much more fundamental to the method than is possible to deal with in this paper yet, given all the other advancements. Even though these extensions are left for the future research, below we report the results when taking the same standpoint as in the past two-step statistical approaches. That is, the impact of transaction costs is taken into account in the evaluation stage when evaluating the resulting portfolio returns and utility levels. This view is natural in the two-step statistical approach where the predictive regressions do not contain utility optimization, whereas transaction costs would be an important part of direct portfolio optimization problem.

Table A.2.1: In-sample results with low transaction costs when $\gamma = 5$ and $\gamma = 3$ with the portfolio weight upper bound $w^{\max} = 1$.

	$\gamma = 5$				$\gamma = 3$			
	Utility boosting		Linear, constr. weights		Utility boosting		Linear, constr. weights	
Variable	Util(%)	Sharpe	Util(%)	Sharpe	Util(%)	Sharpe	Util(%)	Sharpe
Panel A: Macroeconomic variables								
DP	0.680	0.183	0.579	0.136	0.808	0.186	0.710	0.147
DY	0.725	0.199	0.579	0.137	0.850	0.199	0.704	0.146
EP	0.653	0.180	0.567	0.134	0.736	0.172	0.697	0.144
DE	0.591	0.159	0.586	0.139	0.698	0.158	0.686	0.138
RVOL	0.716	0.201	0.597	0.136	0.815	0.196	0.695	0.139
BM	0.711	0.200	0.566	0.132	0.819	0.182	0.702	0.143
NTIS	0.657	0.179	0.581	0.137	0.775	0.177	0.701	0.143
TBL	0.666	0.184	0.632	0.164	0.891	0.210	0.726	0.160
LTY	0.632	0.180	0.604	0.150	0.784	0.191	0.697	0.148
LTR	0.702	0.188	0.632	0.155	0.845	0.198	0.748	0.161
TMS	0.690	0.190	0.652	0.168	0.909	0.219	0.773	0.170
DFY	0.600	0.162	0.585	0.137	0.702	0.162	0.705	0.143
DFR	0.695	0.200	0.586	0.147	0.857	0.214	0.711	0.152
INFL	0.672	0.186	0.574	0.138	0.863	0.204	0.690	0.141
Panel B: Technical indicators								
MA(1,9)	0.626	0.177	0.605	0.163	0.729	0.178	0.692	0.158
MA(1,12)	0.649	0.187	0.645	0.181	0.758	0.187	0.742	0.182
MA(2,9)	0.635	0.181	0.620	0.169	0.739	0.180	0.710	0.166
MA(2,12)	0.670	0.195	0.670	0.190	0.779	0.195	0.773	0.192
MA(3,9)	0.645	0.183	0.635	0.174	0.749	0.182	0.716	0.167
MA(3,12)	0.609	0.168	0.590	0.152	0.711	0.167	0.682	0.146
MOM(9)	0.611	0.170	0.592	0.154	0.714	0.170	0.679	0.147
MOM(12)	0.613	0.169	0.592	0.154	0.718	0.169	0.684	0.148
VOL(1,9)	0.614	0.170	0.583	0.150	0.711	0.169	0.670	0.145
VOL(1,12)	0.631	0.178	0.613	0.167	0.736	0.178	0.709	0.166
VOL(2,9)	0.606	0.168	0.591	0.155	0.706	0.166	0.694	0.152
VOL(2,12)	0.613	0.171	0.604	0.162	0.722	0.173	0.711	0.164
VOL(3,9)	0.600	0.165	0.580	0.147	0.701	0.164	0.681	0.144
VOL(3,12)	0.645	0.181	0.636	0.173	0.753	0.182	0.739	0.176
Panel C: Benchmarks								
Const	0.570	0.141			0.698	0.144		
HA	0.569	0.134			0.695	0.141		

Notes: In these results, the low transaction cost scenario (0.10% of the traded portfolio value) is taken into account when computing the values of economic goodness-of-fit measures.

Table A.2.2: In-sample results with high transaction costs when $\gamma = 5$ and $\gamma = 3$ with the portfolio weight upper bound $w^{\max} = 1$.

	$\gamma = 5$				$\gamma = 3$			
	Utility boosting		Linear, constr. weights		Utility boosting		Linear, constr. weights	
Variable	Util(%)	Sharpe	Util(%)	Sharpe	Util(%)	Sharpe	Util(%)	Sharpe
Panel A: Macroeconomic variables								
DP	0.663	0.177	0.571	0.134	0.786	0.179	0.705	0.145
DY	0.701	0.190	0.571	0.135	0.818	0.189	0.699	0.144
EP	0.641	0.176	0.559	0.132	0.727	0.169	0.692	0.143
DE	0.584	0.155	0.580	0.137	0.689	0.155	0.683	0.138
RVOL	0.662	0.180	0.578	0.130	0.757	0.177	0.683	0.135
BM	0.686	0.190	0.559	0.130	0.801	0.177	0.699	0.142
NTIS	0.642	0.173	0.575	0.135	0.760	0.172	0.699	0.142
TBL	0.656	0.180	0.625	0.161	0.867	0.203	0.720	0.158
LTY	0.624	0.176	0.599	0.148	0.766	0.184	0.692	0.146
LTR	0.603	0.155	0.523	0.120	0.744	0.167	0.658	0.135
TMS	0.671	0.183	0.632	0.161	0.861	0.204	0.758	0.165
DFY	0.589	0.157	0.578	0.135	0.686	0.156	0.702	0.143
DFR	0.616	0.171	0.524	0.126	0.771	0.186	0.670	0.141
INFL	0.614	0.165	0.556	0.132	0.787	0.181	0.679	0.138
Panel B: Technical indicators								
MA(1,9)	0.592	0.163	0.571	0.149	0.690	0.163	0.662	0.148
MA(1,12)	0.621	0.175	0.614	0.168	0.729	0.176	0.710	0.170
MA(2,9)	0.608	0.170	0.591	0.158	0.710	0.170	0.682	0.157
MA(2,12)	0.646	0.185	0.642	0.179	0.754	0.186	0.744	0.182
MA(3,9)	0.619	0.173	0.609	0.164	0.721	0.172	0.690	0.158
MA(3,12)	0.590	0.161	0.574	0.146	0.692	0.161	0.670	0.143
MOM(9)	0.587	0.160	0.571	0.146	0.686	0.159	0.663	0.142
MOM(12)	0.594	0.162	0.573	0.147	0.696	0.161	0.668	0.143
VOL(1,9)	0.570	0.153	0.543	0.136	0.662	0.152	0.637	0.135
VOL(1,12)	0.593	0.163	0.572	0.151	0.695	0.163	0.669	0.152
VOL(2,9)	0.578	0.157	0.565	0.145	0.680	0.157	0.672	0.146
VOL(2,12)	0.589	0.162	0.579	0.152	0.697	0.164	0.689	0.157
VOL(3,9)	0.576	0.156	0.561	0.140	0.674	0.156	0.668	0.141
VOL(3,12)	0.621	0.172	0.611	0.164	0.728	0.173	0.714	0.168
Panel C: Benchmarks								
Const	0.567	0.140			0.697	0.144		
HA	0.564	0.133			0.693	0.141		

The full sample results in Tables A.2.1 and A.2.2 show that when comparing to the main conclusions of Tables 4.3 and 4.4 the best predictors are still essentially the same. In other words, transaction costs naturally diminish the economic gains in different predictors quite symmetrically. The essential differences between the utility boosting and the current two-step approach are still valid. Less persistent variables (DFR and INFL) naturally suffer somewhat more on inevitably stronger impact of transaction costs (i.e. due to more active portfolio rebalancing), especially in the high cost scenario. On the other hand, as technical indicators are defined as binary variables, there is especially in

full sample estimations no portfolio rebalancing in most months as the value of the predictor remains the same (i.e. due to persistent runs of either zeros and ones as defined in Section 3.1).

Table A.2.3: Out of-sample results with low transaction costs when $\gamma = 5$ and $\gamma = 3$ with the portfolio weight upper bound $w^{\max} = 1$.

	$\gamma = 5$				$\gamma = 3$			
	Utility boosting		Linear, constr. weights		Utility boosting		Linear, constr. weights	
Variable	Util(%)	Sharpe	Util(%)	Sharpe	Util(%)	Sharpe	Util(%)	Sharpe
Panel A: Macroeconomic variables								
DP	0.420	0.131	0.242	0.030	0.518	0.134	0.293	0.049
DY	0.413	0.128	0.256	0.039	0.470	0.119	0.285	0.046
EP	0.403	0.146	0.497	0.166	0.514	0.151	0.586	0.164
DE	0.366	0.121	0.296	0.084	0.418	0.109	0.435	0.108
RVOL	0.434	0.137	0.377	0.103	0.496	0.125	0.519	0.126
BM	0.368	0.107	0.411	0.115	0.499	0.131	0.506	0.126
NTIS	0.446	0.150	0.369	0.113	0.539	0.148	0.508	0.131
TBL	0.418	0.133	0.470	0.155	0.556	0.145	0.590	0.153
LTY	0.429	0.146	0.461	0.142	0.514	0.138	0.580	0.150
LTR	0.396	0.124	0.383	0.113	0.505	0.132	0.477	0.120
TMS	0.412	0.135	0.426	0.143	0.519	0.138	0.605	0.157
DFY	0.396	0.122	0.414	0.121	0.445	0.111	0.538	0.137
DFR	0.462	0.168	0.450	0.144	0.578	0.173	0.552	0.150
INFL	0.435	0.139	0.381	0.114	0.581	0.158	0.519	0.133
Panel B: Technical indicators								
MA(1,9)	0.506	0.191	0.469	0.162	0.602	0.184	0.553	0.157
MA(1,12)	0.488	0.184	0.522	0.194	0.601	0.189	0.614	0.193
MA(2,9)	0.487	0.186	0.472	0.171	0.588	0.181	0.545	0.161
MA(2,12)	0.536	0.207	0.559	0.212	0.647	0.210	0.654	0.210
MA(3,9)	0.462	0.169	0.439	0.155	0.567	0.172	0.501	0.147
MA(3,12)	0.465	0.169	0.449	0.153	0.549	0.161	0.519	0.146
MOM(9)	0.478	0.176	0.483	0.167	0.590	0.178	0.540	0.155
MOM(12)	0.480	0.176	0.482	0.164	0.595	0.176	0.556	0.157
VOL(1,9)	0.481	0.177	0.433	0.148	0.572	0.175	0.514	0.146
VOL(1,12)	0.516	0.191	0.473	0.170	0.604	0.187	0.555	0.167
VOL(2,9)	0.454	0.166	0.440	0.153	0.558	0.168	0.534	0.154
VOL(2,12)	0.490	0.179	0.499	0.173	0.597	0.177	0.590	0.170
VOL(3,9)	0.468	0.169	0.444	0.145	0.536	0.154	0.524	0.143
VOL(3,12)	0.529	0.195	0.541	0.193	0.643	0.197	0.633	0.190
Panel C: Benchmarks								
Const	0.435	0.139			0.563	0.146		
HA	0.443	0.133			0.551	0.141		

Notes: As in Tables A.2.1 and A.2.2., transaction costs are taken into account during the out-of-sample forecasting evaluation.

Table A.2.4: Out-of-sample results with high transaction costs when $\gamma = 5$ and $\gamma = 3$ with the portfolio weight upper bound $w^{\max} = 1$.

	$\gamma = 5$				$\gamma = 3$			
	Utility boosting		Linear, constr. weights		Utility boosting		Linear, constr. weights	
Variable	Util(%)	Sharpe	Util(%)	Sharpe	Util(%)	Sharpe	Util(%)	Sharpe
Panel A: Macroeconomic variables								
DP	0.394	0.121	0.233	0.025	0.491	0.126	0.281	0.044
DY	0.383	0.117	0.243	0.031	0.441	0.110	0.269	0.039
EP	0.369	0.129	0.487	0.161	0.466	0.132	0.578	0.161
DE	0.344	0.113	0.283	0.080	0.385	0.098	0.423	0.104
RVOL	0.397	0.123	0.358	0.097	0.445	0.110	0.503	0.122
BM	0.337	0.093	0.404	0.112	0.466	0.119	0.501	0.125
NTIS	0.427	0.143	0.357	0.109	0.517	0.141	0.499	0.128
TBL	0.401	0.128	0.465	0.153	0.534	0.139	0.588	0.153
LTY	0.414	0.140	0.455	0.140	0.494	0.131	0.577	0.149
LTR	0.296	0.090	0.262	0.075	0.393	0.098	0.366	0.089
TMS	0.381	0.125	0.412	0.139	0.481	0.127	0.595	0.155
DFY	0.364	0.109	0.404	0.117	0.399	0.096	0.530	0.135
DFR	0.421	0.149	0.406	0.128	0.511	0.149	0.512	0.138
INFL	0.376	0.118	0.343	0.102	0.517	0.137	0.487	0.124
Panel B: Technical indicators								
MA(1,9)	0.483	0.181	0.449	0.154	0.574	0.173	0.535	0.151
MA(1,12)	0.458	0.172	0.496	0.183	0.570	0.177	0.586	0.182
MA(2,9)	0.457	0.174	0.447	0.161	0.557	0.170	0.521	0.153
MA(2,12)	0.510	0.196	0.537	0.202	0.619	0.199	0.629	0.201
MA(3,9)	0.432	0.157	0.412	0.145	0.537	0.161	0.471	0.137
MA(3,12)	0.445	0.161	0.434	0.148	0.525	0.153	0.505	0.142
MOM(9)	0.456	0.166	0.465	0.160	0.562	0.168	0.525	0.150
MOM(12)	0.462	0.168	0.468	0.159	0.571	0.168	0.544	0.153
VOL(1,9)	0.439	0.159	0.395	0.133	0.528	0.158	0.478	0.134
VOL(1,12)	0.474	0.174	0.432	0.153	0.560	0.171	0.509	0.151
VOL(2.9)	0.426	0.154	0.413	0.142	0.527	0.157	0.508	0.145
VOL(2,12)	0.471	0.171	0.480	0.165	0.572	0.168	0.572	0.164
VOL(3,9)	0.446	0.159	0.426	0.139	0.507	0.144	0.509	0.138
VOL(3,12)	0.505	0.186	0.520	0.184	0.618	0.188	0.609	0.182
Panel C: Benchmarks								
Const	0.432	0.138			0.561	0.146		
HA	0.439	0.131			0.548	0.140		

In Tables A.2.3 and A.2.4, we present the corresponding out-of-sample forecasting results as in Tables 4.7 and 4.8 in the main text but now incorporating also transaction costs in the evaluation stage. The obtained realized utility levels naturally drop due to transaction costs. The drop appears to be about the same magnitude in both the utility boosting and the 'linear' two-step statistical approach. As in the in-sample results above, the smaller amount of trading activity in the case of technical indicators imply that the impact of transaction costs is substantially smaller versus the macroeconomic variables where partly quite erratic behaviour (such as the IT bubble during

the millennium 2000) complicates their use as predictors.

A.3 Splines as baselearner function

Regression trees, which are now used as the base learner function in the customized gradient boosting in the main analysis, have several advantages. Trees are highly flexible, easily interpretable and computationally efficient. However, trees provide only one possible alternative in this respect. This section presents results when the base learner function is changed to a slightly more constrained spline-based alternative (i.e. using so called P-splines).

Table A.3.1: Spline-based in-sample predictive results with $\gamma = 5$ or $\gamma = 3$ and $w^{\max} = 1$.

	$\gamma = 5$			$\gamma = 3$		
Variable	Util(%)	CER(%)	Sharpe	Util(%)	CER(%)	Sharpe
Panel A: Macroeconomic variables						
DP	0.651	1.309	0.172	0.762	0.929	0.172
DY	0.647	1.270	0.170	0.763	0.953	0.171
EP	0.648	1.434	0.174	0.779	1.304	0.175
DE	0.588	0.556	0.150	0.706	0.211	0.148
RVOL	0.619	1.000	0.164	0.708	0.233	0.151
BM	0.566	0.341	0.145	0.741	0.705	0.160
NTIS	0.655	1.419	0.173	0.771	1.095	0.170
TBL	0.638	1.289	0.170	0.740	0.772	0.168
LTY	0.603	0.849	0.164	0.740	0.675	0.161
LTR	0.649	1.244	0.168	0.780	1.207	0.170
TMS	0.681	1.728	0.186	0.793	1.368	0.181
DFY	0.597	0.750	0.158	0.684	-0.035	0.150
DFR	0.605	0.909	0.162	0.720	0.514	0.160
INFL	0.643	1.321	0.173	0.766	1.114	0.177
Panel B: Technical indicators						
MA(1,9)	0.633	1.529	0.181	0.739	1.055	0.181
MA(1,12)	0.655	1.835	0.189	0.766	1.405	0.190
MA(2,9)	0.641	1.625	0.184	0.749	1.162	0.183
MA(2,12)	0.675	2.103	0.197	0.786	1.651	0.197
MA(3,9)	0.650	1.698	0.185	0.756	1.208	0.184
MA(3,12)	0.612	1.163	0.169	0.717	0.722	0.169
MOM(9)	0.616	1.255	0.172	0.721	0.788	0.172
MOM(12)	0.617	1.232	0.171	0.724	0.800	0.170
VOL(1,9)	0.623	1.296	0.174	0.724	0.785	0.173
VOL(1,12)	0.639	1.555	0.181	0.748	1.130	0.181
VOL(2,9)	0.612	1.183	0.171	0.719	0.754	0.171
VOL(2,12)	0.618	1.291	0.173	0.731	0.931	0.175
VOL(3,9)	0.604	1.049	0.167	0.708	0.583	0.166
VOL(3,12)	0.649	1.627	0.182	0.759	1.226	0.183

Table A.3.2: Spline-based out of-sample predictive results with $\gamma = 5$ or $\gamma = 3$ and $w^{\max} = 1$.

	$\gamma = 5$			$\gamma = 3$		
Variable	Util(%)	CER(%)	Sharpe	Util(%)	CER(%)	Sharpe
Panel A: Macroeconomic variables						
DP	0.431	0.084	0.135	0.545	-0.094	0.143
DY	0.395	-0.425	0.120	0.529	-0.311	0.137
EP	0.446	0.804	0.164	0.584	0.678	0.174
DE	0.348	-0.604	0.112	0.433	-1.352	0.111
RVOL	0.434	0.147	0.136	0.521	-0.522	0.131
BM	0.401	-0.440	0.118	0.484	-0.953	0.122
NTIS	0.489	1.152	0.165	0.615	0.917	0.167
TBL	0.392	-0.128	0.130	0.543	-0.041	0.141
LTY	0.424	0.228	0.138	0.505	-0.579	0.131
LTR	0.404	-0.237	0.127	0.535	-0.167	0.138
TMS	0.429	0.306	0.141	0.544	0.078	0.145
DFY	0.391	-0.457	0.117	0.462	-1.140	0.119
DFR	0.441	0.538	0.150	0.503	-0.517	0.132
INFL	0.460	0.525	0.147	0.576	0.320	0.153
Panel A: Technical indicators						
MA(1,9)	0.505	1.689	0.187	0.605	1.144	0.183
MA(1,12)	0.508	1.953	0.193	0.618	1.513	0.195
MA(2,9)	0.481	1.514	0.180	0.589	1.067	0.181
MA(2,12)	0.553	2.573	0.213	0.660	2.063	0.214
MA(3,9)	0.473	1.373	0.173	0.570	0.794	0.172
MA(3,12)	0.459	1.011	0.164	0.562	0.588	0.166
MOM(9)	0.475	1.244	0.172	0.581	0.843	0.174
MOM(12)	0.480	1.268	0.173	0.585	0.852	0.173
VOL(1,9)	0.478	1.328	0.174	0.590	0.979	0.181
VOL(1,12)	0.499	1.674	0.184	0.611	1.333	0.190
VOL(2,9)	0.456	1.028	0.165	0.559	0.583	0.167
VOL(2,12)	0.493	1.429	0.178	0.592	0.960	0.173
VOL(3,9)	0.470	1.055	0.166	0.556	0.401	0.160
VOL(3,12)	0.527	1.972	0.194	0.645	1.678	0.199

Table A.3.3: Spline-based in-sample predictive results for the predictor groups with $\gamma = 5$ or $\gamma = 3$ and $w^{\max} = 1$.

	$\gamma = 5$			$\gamma = 3$		
Variables	Util(%)	CER(%)	Sharpe	Util(%)	CER(%)	Sharpe
MACRO	0.840	3.819	0.252	0.959	3.504	0.253
TECH	0.686	2.227	0.200	0.822	2.040	0.207
ALL	0.851	4.016	0.261	0.993	3.975	0.270

Table A.3.4: Spline-based out of-sample predictive results for the predictor groups with $\gamma = 5$ or $\gamma = 3$ and $w^{\max} = 1$.

	$\gamma = 5$			$\gamma = 3$		
Variables	Util(%)	CER(%)	Sharpe	Util(%)	CER(%)	Sharpe
MACRO	0.501	1.139	0.166	0.582	0.411	0.158
TECH	0.511	1.907	0.194	0.588	1.106	0.183
ALL	0.541	1.747	0.187	0.613	0.901	0.173

The results in Tables A.3.1–A.3.4 are largely similar to the ones obtained with regression trees in Tables 4.3–4.4 and 4.7–4.9. This concerns single predictors and multivariate predictor groups. Here using all the predictors (ALL) even lead to somewhat superior performance than obtained with the regression tree case, further strengthening the usefulness of the utility boosting method and its internal model selection capability.

A.4 GARCH model and realized variance based volatility proxies

In accordance with various closely related return predictability studies (see, e.g., Campbell and Thompson, 2008; Rapach et al., 2010; Rapach and Zhou, 2013; Neely et al., 2014), we have used a five-year rolling window (moving average) based volatility proxy computed using historical excess returns. In this section, we consider two alternative ways to extract the necessary volatility proxy σ_t^2: The Generalized Autoregressive Conditional Heteroskedastic (GARCH) model and an alternative realized variance-based approach.

In this section, we concentrate on the full sample results. Specifically here, as this is already a robustness check for the main analysis, detailed out-of-sample forecasting results, containing continuous parameter updating, depends crucially on finding the optimal GARCH model specification as well and its detailed forecasting performance evaluation. Hence, we believe relying on the maximum data availability provides the best additional value over the main results.

In the GARCH model specification for the excess market returns, the conditional variance h_t^2 is extracted with a GARCH error term, combined with the constant conditional mean in line with the evidence of Welch and Goyal (2008), among others (i.e. no mean return predictability). Formally, we set

hence

$$r_{e,t} = \alpha_0 + h_t \, \nu_t,$$

where ν_t is an independent and identically distributed error term with zero mean and unit variance ($\nu_t \sim \mathrm{iid}(0,1)$). Following the large majority of past GARCH specifications, the normality assumption (i.e. $\nu_t \sim \mathrm{nid}(0,1)$) is assumed for maximum likelihood estimation purposes, and the conditional variance is assumed to follow the GARCH(1,1) process

$$h_t^2 = c + b_1 h_{t-1}^2 + a_1 u_{t-1}^2, \qquad c > 0, b_1 \geq 0, a_1 > 0,$$

where $u_t = r_{e,t} - \alpha_0$. There are, of course, various extensions available for the above specifications of the conditional mean and conditional variance, including, for example, asymmetric conditional variance equations (asymmetric GARCH models) and non-Gaussian innovations, but here we restrict ourselves to this commonly used specification. Finally, the extracted estimate of the conditional variance $\widehat{h}_t^2$ acts as our volatility proxy in the procedures described in Section 2.

Table A.4.1 presents the (full sample) results obtained with our benchmark selections $\gamma = 5$ and $w^{\max} = 1$ when the GARCH-based volatility proxy is employed. It turns out that within technical indicators the same indicators perform the best (MA(1,12), MA(2,12) and VOL(3,12)) as in the case of main volatility proxy. In the case of macroeconomic variables, some changes occur as now EP and TMS are among the best performing variables in this setting. All in all, all the same main conclusions on the superiority of the utility boosting over the traditional two-step 'linear' approach are intact (seen as almost uniformly higher Sharpe ratios and realized utility levels).

Table A.4.1: In-sample predictive results with $\gamma = 5$ and $w^{\max} = 1$ when using the extracted conditional variance from the GARCH(1,1) model as the underlying volatility proxy σ_t^2.

	Utility boosting			Linear, constrained weights			
Variable	Util(%)	CER(%)	Sharpe	Util(%)	Sharpe	t-val	adj-R^2
Panel A: Macroeconomic variables							
DP	0.664	1.692	0.186	0.562	0.142	1.76	0.28
DY	0.656	1.633	0.184	0.563	0.144	1.87	0.33
EP	0.706	2.375	0.204	0.554	0.140	0.77	0.04
DE	0.579	0.711	0.157	0.537	0.134	0.69	-0.02
RVOL	0.690	2.229	0.209	0.555	0.145	2.80	0.63
BM	0.666	1.755	0.186	0.536	0.133	0.58	-0.07
NTIS	0.637	1.451	0.177	0.532	0.131	0.27	-0.10
TBL	0.637	1.503	0.184	0.584	0.155	2.26	0.55
LTY	0.625	1.308	0.180	0.563	0.146	1.49	0.18
LTR	0.678	1.895	0.190	0.601	0.155	2.68	0.69
TMS	0.704	2.280	0.202	0.595	0.154	1.90	0.41
DFY	0.667	1.848	0.191	0.526	0.129	0.38	-0.08
DFR	0.603	1.078	0.169	0.575	0.154	0.96	0.08
INFL	0.622	1.302	0.174	0.540	0.135	0.35	-0.09
Panel B: Technical indicators							
MA(1,9)	0.621	1.541	0.178	0.584	0.159	1.63	0.29
MA(1,12)	0.662	2.039	0.192	0.628	0.175	1.97	0.55
MA(2,9)	0.642	1.751	0.184	0.606	0.166	1.83	0.39
MA(2,12)	0.683	2.293	0.199	0.654	0.184	2.34	0.76
MA(3,9)	0.645	1.739	0.183	0.606	0.164	1.78	0.41
MA(3,12)	0.614	1.316	0.171	0.569	0.148	1.02	0.08
MOM(9)	0.612	1.320	0.172	0.571	0.151	1.11	0.11
MOM(12)	0.617	1.350	0.171	0.571	0.150	1.10	0.12
VOL(1,9)	0.603	1.253	0.170	0.575	0.153	1.30	0.16
VOL(1,12)	0.627	1.583	0.179	0.596	0.162	1.73	0.39
VOL(2,9)	0.613	1.368	0.174	0.578	0.154	1.31	0.18
VOL(2,12)	0.620	1.502	0.177	0.587	0.158	1.49	0.29
VOL(3,9)	0.600	1.162	0.168	0.562	0.146	0.95	0.04
VOL(3,12)	0.642	1.693	0.181	0.609	0.164	1.93	0.52
Panel C: Benchmarks							
Const	0.556	0.425	0.143				
HA	0.529		0.131				

As another realized volatility proxy and robustness check for the 5-year rolling window volatility, we utilize RVOL as defined in Table 4.1 (see Mele (2007), without annualizing RVOL). This means that the lagged RVOL (lagged by one month) then acts as our volatility proxy σ_t^2. As a 12-month moving average volatility estimator, RVOL is also rather persistent, as the 60-month (5-year) moving average in the main analysis. However, due to the shorter computation window the 12-month estimator naturally reacts somewhat sharper to volatility changes.

Table A.4.2: In-sample predictive results with $\gamma = 5$ and $w^{\max} = 1$ when RVOL is the volatility proxy.

	Utility boosting			Linear, constrained weights			
Variable	Util(%)	CER(%)	Sharpe	Util(%)	Sharpe	t-val	adj-R^2
Panel A: Macroeconomic variables							
DP	0.667	1.877	0.186	0.585	0.142	1.76	0.28
DY	0.639	1.556	0.174	0.588	0.144	1.87	0.33
EP	0.728	2.740	0.207	0.571	0.136	0.77	0.04
DE	0.584	0.972	0.157	0.547	0.128	0.69	-0.02
RVOL	0.587	0.982	0.160	0.563	0.144	2.80	0.63
BM	0.707	2.396	0.205	0.550	0.129	0.58	-0.07
NTIS	0.646	1.724	0.180	0.535	0.123	0.27	-0.10
TBL	0.629	1.599	0.180	0.587	0.148	2.26	0.55
LTY	0.726	2.740	0.218	0.570	0.140	1.49	0.18
LTR	0.681	2.068	0.190	0.596	0.146	2.68	0.69
TMS	0.735	2.897	0.215	0.596	0.147	1.90	0.41
DFY	0.590	1.036	0.159	0.537	0.124	0.38	-0.08
DFR	0.622	1.418	0.170	0.587	0.148	0.96	0.08
INFL	0.601	1.165	0.164	0.550	0.129	0.35	-0.09
Panel B: Technical indicators							
MA(1,9)	0.615	1.639	0.176	0.590	0.155	1.63	0.29
MA(1,12)	0.654	2.122	0.190	0.628	0.172	1.97	0.55
MA(2,9)	0.627	1.732	0.180	0.609	0.163	1.83	0.39
MA(2,12)	0.673	2.360	0.197	0.651	0.180	2.34	0.76
MA(3,9)	0.636	1.838	0.181	0.610	0.162	1.78	0.41
MA(3,12)	0.606	1.403	0.168	0.571	0.142	1.02	0.08
MOM(9)	0.601	1.373	0.168	0.577	0.145	1.11	0.11
MOM(12)	0.623	1.587	0.173	0.587	0.148	1.10	0.12
VOL(1,9)	0.598	1.359	0.168	0.587	0.149	1.30	0.16
VOL(1,12)	0.621	1.711	0.178	0.601	0.161	1.73	0.39
VOL(2,9)	0.601	1.435	0.171	0.581	0.149	1.31	0.18
VOL(2,12)	0.608	1.559	0.174	0.588	0.155	1.49	0.29
VOL(3,9)	0.589	1.207	0.163	0.562	0.138	0.95	0.04
VOL(3,12)	0.631	1.759	0.178	0.607	0.160	1.93	0.52
Panel C: Benchmarks							
Const	0.562	0.611	0.143				
HA	0.541		0.125				

Table A.4.2 shows that when relying on the RVOL-based volatility proxy, basically the same empirical findings between the methods and predictor-specific performances arise as already pointed out in the GARCH case. With the technical indicators, the same indicators are again performing the best whereas with the macroeconomic variables there is more variation. Interestingly in both robustness checks (GARCH and RVOL), EP and TMS stand out much more clearer among the macroeconomic variables than in the case of slowly evolving 5-year moving average volatility proxy.

Instead of the predictor-specific considerations, let us still consider multi-

variate results with different volatility proxies. Table A.4.3 reports the same groups of predictors and analysis settings as in Table 4.5 in the main text, but now for the GARCH and RVOL-based volatility proxies.

Table A.4.3: In-sample predictive results with $\gamma = 5$ and $w^{\max} = 1$ when using the extracted conditional variance from the GARCH(1,1) model and RVOL as the volatility proxy.

	Utility boosting			Linear (PCA), constrained weights		
Variables	Util(%)	CER(%)	Sharpe	Util(%)	Sharpe	adj-R^2
Panel A: GARCH-based volatility proxy						
MACRO	0.934	5.223	0.305	0.584	0.158	0.54
TECH	0.676	2.186	0.198	0.614	0.169	0.45
ALL	0.944	5.377	0.313	0.650	0.188	0.92
Panel B: RVOL-based volatility proxy						
MACRO	0.896	4.866	0.292	0.605	0.159	0.54
TECH	0.648	1.982	0.191	0.605	0.161	0.45
ALL	0.958	5.664	0.321	0.645	0.183	0.92

The main results are largely the same as in Table 4.5. Larger information sets of predictors stabilizes the performances over certain noise accompanied to predictor-specific results. Compared with Table 4.5, maybe the only slight change is that now TECH appears to contribute the combined ALL case marginally stronger both in the utility boosting and in the two-step linear PCA-based approaches.

A.5 First differences of macroeconomic predictors

The descriptive statistics in Table 4.2 shows that majority of macroeconomic predictive variables are highly persistent. The only exceptions are LTR, DFR and INFL. This section considers whether taking the first differences of the persistent macroeconomic predictors affect the out-of-sample forecasting results.

Table A.5.1: Out-of-sample results in predictor-specific models with $\gamma = 5$ and $\gamma = 3$ ($w^{\max} = 1$) after taking the first differences of the persistent macroeconomic predictors.

	$\gamma = 5$				$\gamma = 3$			
	Utility boosting		Linear, constr. weights		Utility boosting		Linear, constr. weights	
Variable	Util(%)	Sharpe	Util(%)	Sharpe	Util(%)	Sharpe	Util(%)	Sharpe
ΔDP	0.510	0.182	0.438	0.139	0.649	0.194	0.520	0.139
ΔDY	0.417	0.128	0.430	0.125	0.510	0.130	0.521	0.131
ΔEP	0.380	0.123	0.353	0.104	0.491	0.135	0.478	0.123
ΔDE	0.492	0.172	0.440	0.138	0.624	0.184	0.546	0.145
ΔRVOL	0.396	0.127	0.419	0.125	0.519	0.138	0.535	0.137
ΔBM	0.441	0.142	0.456	0.139	0.536	0.141	0.554	0.146
ΔNTIS	0.435	0.139	0.416	0.124	0.530	0.145	0.561	0.145
ΔTBL	0.417	0.136	0.452	0.135	0.527	0.136	0.541	0.138
ΔLTY	0.428	0.138	0.435	0.130	0.541	0.142	0.545	0.139
ΔTMS	0.418	0.138	0.406	0.121	0.457	0.120	0.542	0.139
ΔDFY	0.421	0.137	0.464	0.139	0.546	0.144	0.569	0.147

Notes: In this table, we report the out-of-sample forecasting results when taking the first differences of the persistent macroeconomic predictive variables (see Tables 4.1 and 4.2). Compare the out-of-sample forecasting results (with the levels) in Tables 4.7 and 4.8.

Table A.5.2: Out-of-sample predictive results of the predictor groups (MACRO and ALL) with $\gamma = 5$ and $w^{\max} = 1$ using different volatility proxies and taking the first differences of persistent macroeconomic predictors.

	Utility boosting			Linear (PCA), constrained weights		
Variables	Util(%)	CER(%)	Sharpe	Util(%)	CER(%)	Sharpe
Panel A: 5-year moving average volatility proxy						
MACRO	0.538	1.795	0.186	0.375	-0.985	0.108
ALL	0.553	2.035	0.195	0.371	-0.520	0.119
Panel B: GARCH-based volatility proxy						
MACRO	0.571	1.444	0.205	0.377	-1.356	0.118
ALL	0.574	1.552	0.209	0.357	-1.375	0.116
Panel C: RVOL-based volatility proxy						
MACRO	0.564	1.454	0.199	0.416	-1.003	0.123
ALL	0.594	1.893	0.215	0.375	-1.244	0.115

Forecasting results with the changes in the dividend-price ratio (ΔDP), and partly also the differenced payout ratio (ΔDE), pops out from the results. In other words, using the changes of the strongly asset pricing-motivated dividend-price ratio supports its use in portfolio weight determination even further. Together with the fact that less persistent inflation (INFL) and partly also default return spread (DFR) perform well in Tables 4.7 and 4.8, this all

suggests that less persistent state variables seem generally more useful in the utility boosting than highly persistent ones.

Multivariate results in Table A.5.2 (cf. Table 4.9)) again strongly support the usefulness of the utility boosting method. Using all the predictors, including thus also the technical indicators (TECH), show that the additional predictive value can be obtained over using just the MACRO variables (here the first differences of the persistent macro variables augmented with LTR, DFR and INFL). It is also noteworthy how the multivariate results for utility boosting in Panel A are slightly higher using each evaluation criteria when compared to the ones obtained without differencing in Table 4.9 of the main text.

Appendix B: Comparison to Brandt and Santa-Clara (2006)

In this Appendix, we briefly state the main points of Brandt and Santa-Clara (2006) relevant to the comparison with our method and analysis. Their study is probably the closest one to our approach in terms of the general idea of direct portfolio weight determination. However, their approach is still in various ways very different and aiming to answer somewhat different demand than ours. First of all, their empirical context is closely connected to potentially large cross-sections of assets. That is at the centre of attention also in Brandt (1999), Aït-Sahalia and Brandt (2001) and Brandt et al. (2009) whereas here we are introducing a new empirical approach in a simple setting between one risky asset and the risk-free rate.

Methodologically, in contrast to our highly flexible, nonlinear and advanced machine learning-motivated approach, Brandt and Santa-Clara (2006) assume that the optimal portfolio weights are linear in parameters

$$w_t = \boldsymbol{\theta}\boldsymbol{x}_{t-1}, \tag{B.1}$$

where $\boldsymbol{\theta}$ is the (row) vector of parameter coefficients. Likewise in (4.2), they consider the decision making problem of an investor who maximizes the conditional expected value of a quadratic utility function over the next period's wealth. When writing their objective function in terms of our notation, we get

$$\max_{\boldsymbol{\theta}} \left\{ r_{f,t} + \boldsymbol{\theta}\boldsymbol{x}_{t-1} r_{e,t} - \frac{\gamma}{2}(\boldsymbol{\theta}\boldsymbol{x}_{t-1})^2 r_{e,t}^2 \right\}. \tag{B.2}$$

In our single risky asset setting (market portfolio and risk-free rate), the solution of the above optimization problem leads to the estimate

$$\widehat{\boldsymbol{\theta}} = \frac{1}{\gamma}\Big(\sum_{t=1}^{T}(\boldsymbol{x}_{t-1}\boldsymbol{x}_{t-1}')\, r_{e,t}^2\Big)^{-1}\sum_{t=1}^{T}\boldsymbol{x}_{t-1}r_{e,t}. \tag{B.3}$$

From this solution, the (empirical) weight invested in risky asset is obtained by adding the corresponding products of elements of $\widehat{\boldsymbol{\theta}}$ and $\boldsymbol{x}_{t-1}$. This solution depends only on the data and does not require any assumptions about the distribution of stock returns, apart from stationarity (as in our utility boosting approach) and that returns are assumed non-i.i.d. (identically and independently distributed).

The final solution (B.3) relies on the selected (simple) conditional variance specification in (B.2) (cf. equation (4.2)). This does not allow a straightforward use of rolling window (moving-average) based volatility proxy σ_t^2 (as in the utility boosting) which importantly acknowledges the well-documented volatility clustering in asset prices and found an important ingredient in the past two-step statistical approaches (see Section 4.2.1). Together with this point, another point not allowing for a direct comparison to our empirical perspective, framed by the recent large branch of return predictability studies, is the fact that the solution (B.3) does not automatically respect the pre-determined bounds (4.8), which are guaranteed and an important internal ingredient in the utility boosting approach.

Appendix C: Tuning customized gradient boosting

As was shown in the proposed algorithm in Section 4.2.3, boosting builds the final model in a forward stagewise manner by adding new base learner functions to the ensemble that best fit the negative gradient of the loss function. The final additive boosting ensemble can be written as a sum of the base learner functions

$$F_M(\boldsymbol{x}_{t-1}) = \sum_{m=1}^{M} \upsilon h_m(\boldsymbol{x}_{t-1}).$$

This equation also summarizes the tuning parameters related to gradient boosting, which are the base learner function $h_m(\boldsymbol{x}_{t-1})$ and the amount of

iterations M. In order to make the optimization in functional space feasible, the base learners are assumed to belong to certain parameterized class of functions. Regression trees and smoothing splines are two common choices. Simple linear functions is another alternative, leading to the linear final model.

Regression trees split the predictor space into J disjoint rectangles and attach a simple constant as the functional estimate in each of these rectangles. This is the method that we incorporate to the practical algorithm in Section 4.2.3. Mathematically these J-terminal node regression trees can be written as

$$h_m(\boldsymbol{x}_{t-1}; \{c_{jm}, R_{jm}\}_{j=1}^{J}) = \sum_{j=1}^{J} c_{jm} I(\boldsymbol{x}_{t-1} \in R_{jm}),$$

where $c_{jm} \in \mathbb{R}$ is the functional estimate in region R_{jm} at the mth iteration. The complexity of the regression tree base learner can be controlled by the amount of terminal nodes J, and the amount of observations required at each terminal node. The simplest regression tree with two terminal nodes is sufficient in our predictor-specific analysis. Each of the terminal nodes must contain at least 10 observations, a value commonly used in previous literature.

The optimal amount of iterations M could be determined by setting aside part of the dataset (i.e. excluding this part from fitting the model) and then this separate test set is used to evaluate the generalization ability of the model. To this end K-fold cross-validation is a commonly used method where the dataset is randomly split into K non-overlapping folds. Each of the K folds (concretely $K = 5$ in this study) is used as a test set once while the model is trained using the remaining $K - 1$ folds. Using the utility function presented in (4.17), the validation utility is the average utility produced by the K independent folds

$$CV = \frac{1}{K} \sum_{k=1}^{K} \bar{u}_k^{(-k)},$$

where $\bar{u}_k$ is the average utility when the data points of fold k are used as an independent test set and not in fitting the model. The cross-validation estimate for the amount of iterations M is the one producing the maximum validation utility.

Friedman (2002) introduces an additional randomization step to the algorithm in Section 4.2.3 to further enhance the generalization ability of the

algorithm. At each of the m repeats a random subsample is drawn without replacement from the entire dataset. The pseudo-residuals and the new base learner is then constructed using this random sample instead of the entire dataset. A subsampling rate of one half is a commonly used alternative.

UNIVERSITY OF TURKU

ISBN 978-951-29-8222-6 (PRINT)
ISBN 978-951-29-8223-3 (PDF)
ISSN 2343 3159 (Painettu/Print)
ISSN 2343 3167 (Verkkojulkaisu/Online)

Painosalama Oy, Turku, Finland 2020