GRA 19703

Master Thesis

Thesis Master of Science

Using artificial intelligence in economic policy forecasting

Navn:	Mats Hagland, Benjamin Lian
Start:	15.01.2021 09.00
Finish:	01.07.2021 12.00

USING ARTIFICAL INTELLIGENCE IN ECONOMIC POLICY FORECASTING

Master Thesis

by

Mats Hagland and Benjamin Lian

MSc in Business with Major in Finance and MSc in Business with Major in Finance

Oslo, June 28, 2021

ABSTRACT

We hypothesize that machine learning algorithms are better equipped at forecasting policy rates. To test this hypothesis, we gathered several machine learning algorithms and compared their forecasts of the Norwegian policy rate against Norges Bank's own forecasts. The hypothesis builds upon the idea of machine learning as a general tool. Therefore, we tested a broad set of machine learning algorithms instead of developing a hyper specific model. The machine learning algorithms we tested were the elastic net algorithm, the decision tree algorithm, the long short-term memory neural network, the convolutional neural network, and an ensemble learner. Consistent with our hypothesis, the algorithms did indeed exhibit lower prediction errors than the benchmark. A deeper analysis of the results indicated that this is due to their ability to better adjust to drastic changes in the economy and that Norges Bank's model performs better during stable economic periods.

Acknowledgements

This master thesis marks the end of five years of study at BI Norwegian Business School. We want to express our deepest gratitude to the people who have helped us along the way. First, we would like to thank our supervisor, Professor Kjell Jørgensen, for his guidance and support throughout the master's thesis. In addition, we would like to thank Dr. Rogelio A. Mancisidor for his assistance with machine learning algorithms, and Sissel and Ola Lie for their advice during the writing process. Finally, we would like to take this opportunity to express our gratitude towards family and friends for their help and support through not only this thesis, but the entire master's program at BI.

Contents

1 Introduction

The financial industry is changing. Digital tools and high computing power have provided us with techniques and methods that have otherwise been unavailable. Easy and affordable access to powerful computers has made it possible to process large datasets in a precise and efficient way that challenges conventional models. Among other things, this has led to the emergence of high-frequency trading, decentralized banks, and cryptocurrencies. Machines have not only led to changing business models, but also made it possible to revisit old research to explore whether new techniques are able to change or improve results. Here, machine learning is particularly interesting. In short, machine learning is about using statistical models in combination with the computer's high computational power to find patterns in data. This makes it particularly useful for finding shrouded patterns that are difficult to identify with traditional methods.

Making accurate economic predictions is a difficult but important task. Both policy makers and consumers utilize economic outlooks for decision making. Central banks use these projections to decide whether to stimulate or depress economic growth. Stock market participants make decisions regarding stock prices in conjunction with similar forecasts. Consequently, accuracy in economic outlooks is pertinent to proper financial decision making.

One of the most prominent signals of the state of an economy is the central bank's policy rate. This is because it acts as an intermediary between the government and the economy. In Norway, the policy rate is the interest rate commercial banks receive on deposits. The interest rates that the commercial banks are exposed to will affect the products that these banks offer. Hence, a change in the policy rate will affect the whole Norwegian economy, from consumers' spending behavior to the Norwegian Krone. This makes the policy rate particularly interesting to forecast. In essence, forecasting policy rate decisions is analogous to forecasting the economy.

As of writing this thesis, the Norwegian central bank, Norges Bank, uses a dynamic stochastic general equilibrium (DSGE) model for policy rate forecasting. At the same time, machine learning has been implemented in several parts of economics and finance. Despite the application of machine learning in related fields, machine learning has yet to be implemented in conjunction with policy rates. To further the discussion on utilizing machine learning in fiscal policy decisions, we wanted to test the efficacy of these algorithms in such environments. In fact, we hypothesize that machine learning algorithms can better predict the policy rate than conventional DSGE models.

To test this hypothesis, we compared several machine learning algorithms to Norges Bank's DSGE forecasts: (i) the elastic net algorithm, (ii) the decision tree algorithm, (iii) long short-term memory, (iv) convolutional neural network, and (v) ensemble learner. The purpose of this thesis was not to create a hyper optimized machine learning model. Rather, we wanted to investigate the efficacy of machine learning as a tool. Therefore, we opted to test several well-known algorithms without excessive optimization. This means that this research, and subsequently its results, only pertain to the models themselves.

Our results confirmed our hypothesis; the machine learning algorithms forecasted the Norwegian policy rate more accurately than Norges Bank's DSGE model. Aggregated over several time-steps, the worst machine learning algorithm predicted the policy rate with approximately six percent higher accuracy than Norges Bank. However, a deeper analysis of the forecasts show that this is mostly due to Norges Bank's model's inadequacy to adjust to drastic changes in the economy. Conversely, the machine learning algorithms are better equipped at adjusting to these changes, which is reflected in the final forecasting score.

In the subsequent chapters, we will examine prior research on machine learning and interest rates. Furthermore, we will present the methodological approach related to choosing, constructing, and tuning the different machine learning algorithms. Next, we showcase the details surrounding the data preparation stage of this thesis. Then, the results from our analysis are presented. Lastly, the thesis ends with a discussion of the results and conclusive thoughts related to the thesis.

2 Literature review

2.1 The policy rate

The policy rate, also known as the Folio rate, is normally set every six weeks at Norges Bank's interest rate meeting. The policy rate is the central bank's most important instrument for stabilizing inflation and developments in the Norwegian economy (Norges Bank, n.d.) because it corresponds to the interest rate that commercial banks receive on their deposits with the central bank. This will in turn control the interest rate that the commercial banks offer to their customers on loans and deposits. If the central bank lowers the policy rate, it will lead to cheaper loans, which in turn stimulates increased consumption. With higher consumption, unemployment declines and inflation rises. At the same time, it will weaken the currency because foreign players relocate assets out of the country to a country with higher interest rates. Conversely, these effects are reversed if the policy rate is increased. Consequently, Norges Bank's interest rate decision is significant for the Norwegian economy.

The Norwegian economy is governed by an explicit inflation target, which is currently 2 percent (Norges Bank, 2020a). It states that "Inflation targeting shall be forward-looking and flexible so that it can contribute to high and stable output and employment, and to counteracting financial imbalances" (Lovdata, 2019).

To understand how central banks set policy rates, one must first understand the difference between rule-based monetary policy and discreet monetary policy. Rule-based monetary policy refers to placing restrictions on the authorities' ability to control the economy. In this scenario, the central bank follows a rule that defines which measures can be implemented, e.g., Friedman's k-percent rule. This rule states that the central bank will increase the money supply at a constant rate, regardless of cyclical fluctuations (Friedman, 1960). The main advantage of setting such rules is higher predictability of the economy.

The other side of monetary policy is discreet monetary policy. Under discretion, a monetary authority is free to act in accordance with its own judgment. This means that the authorities must define the state of the economy and act accordingly. With this form of monetary policy, the central bank will use the tools at their disposal to satisfy the economic requests put forth by the authorities. An example of this is that the central bank lowers the policy rate to support the authorities' desire for lower unemployment.

Norges Bank practices a combination of the two policies. Under normal circumstances, they rely on an economic model named NEMO to indicate the appropriate interest rate level and to forecast the economy. However, the policy rate decision is ultimately made by a team of experts from Norges Bank. This relationship between discreet and rule-based monetary policy is especially interesting during crises, which we will analyze in detail in later chapters.

2.1.1 The Taylor rule

It is a challenging task to predict policy rates based on discrete monetary policies using machine learning. That is because the methods used to make the decision are partly based on subjectivity. In addition, the relative importance and size of the feature space varies across time. However, this does not mean that the policy rate is completely unpredictable. It turns out that monetary policy rules can be indicative of policy rate developments.

A well-known rule is the Taylor rule (Taylor, 1993). The advantage of this rule is that it contains few explanatory variables while still producing sufficient estimates. The rule states that if actual inflation is higher than the inflation target, the policy rate should be raised, and vice versa. In addition, he includes a variable that adjusts for the pressure in the economy. Taylor formulated the rule as follows:

$$r = p + \frac{1}{2}y + \frac{1}{2}(p - 2) + 2 \tag{1}$$

Where r is the US federal funds rate, p is the rate of inflation over the previous four quarters, and y is the percent deviation of real GDP from a target. The US Federal Funds rate does not correspond to the Norwegian policy rate. Thus, the Norwegian version includes a neutral real interest rate in equilibrium. A neutral interest rate is the interest rate that in itself does not provide increased or reduced price and cost growth in the economy (Lønning & Olsen, 2000).

The adjusted Taylor rule looks like this:

$$i = r* + \pi* + \beta_1(\pi - \pi*) + \beta_2(y - y*) \tag{2}$$

Where i is the policy rate, $r*$ is the neutral real interest rate in equilibrium, π and $\pi*$ are the actual inflation and target inflation, $y - y*$ is the output gap, and β are the coefficients.

2.1.2 Norges Bank's model

Norges Bank's model for forecasting the policy rate is called NEMO (Norwegian Economic Model). NEMO is a type of dynamic stochastic general equilibrium model, often abbreviated as DSGE. DSGE models are widely used by monetary authorities for policy analysis and forecasting (Vitek, 2017). The model is dynamic and stochastic as the endogenous variables are probabilistic and the paths that the solution creates are dependent upon future stochastic shocks (Brubakk & Sveen, 2009). These shocks are supposed to be analogous to boom-bust cycles in the economy. Furthermore, general equilibrium implies that the market systems in the model at all times will stabilize supply and demand in equilibrium. Hence, one could stipulate that the system models the Norwegian economy on a smaller scale.

The model consists of a system of processes as visualized in Figure 1. Different parts of the system correspond to different parts of a simplified version of the Norwegian economy, such as the oil sector, households, and capital producers. These segments then aggregate the economic output of the system (Kravik & Paulsen, 2017):

$$Y = (A - Q + I + X - M)\,\frac{1}{1 - log(z)} \tag{3}$$

Where X is total export, z is an inventory shock to the mainland economy, A is final retail goods, Q is domestic intermediate goods, I is investments, M is imported intermediate goods.

The policy rate is derived by minimizing a loss function contingent on Norges Bank monetary policy mandate and preferences (Alstadheim et al., 2010). The loss function for optimal policy can be simplified to the following

(Olsen, 2011):

$$L = (\pi - \pi*)^2 + \lambda(Y - Y*)^2 \tag{4}$$

Where $(\pi - \pi*)^2$ is the squared inflation gap and $(Y - Y*)^2$ is the squared output gap. The trade-off between stabilizing inflation and avoiding output gap volatility is expressed by λ. This is essentially a version of the Taylor rule.

In essence, the model emulates the Norwegian economy through different actors such as intermediary producers and households, for which the policy rate is fitted to reduce the inflation- and output gap in the model. Hence, the policy rate is a function of the economic model's future development.

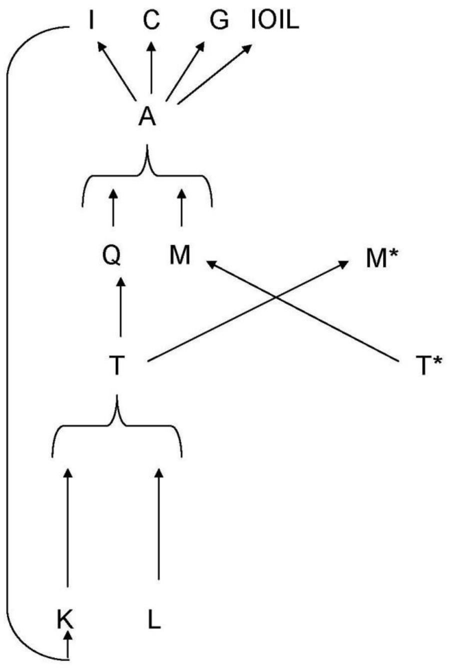


Figure 1: Birds eye view of NEMO (Brubakk et al., 2006). K and L are inputs in the production of intermediate goods T, respectively capital services and differentiated labor. $T*$ is exported intermediary goods and $M*$ is imported intermediate goods. These three inputs; T, $T*$, and $M*$ are inputs corresponding to domestic intermediate goods, Q, and imported intermediate goods, $M*$. Q and M are inputs in A, which is the final retail good. A can be used for consumption, C, Investment, I, government spending, G, and oil investment, $IOIL$.

2.2 Machine learning algorithms

As mentioned in the introduction, virtually any research has been conducted on the use of machine learning to predict policy rates. On the other hand, researchers have been studying machine learning's ability to predict interest rates and yield curves. A yield curve is a line of interest rates for bonds with identical credit quality, but different maturities. The curve signals the market's expectations of the economy, much like the policy rate. No one knows exactly which factors affect the yield curve and the policy rate, but it is reasonable to assume that they possess many similarities. Thus, we believe that a natural starting point would be to investigate the methods for using machine learning in conjunction with interest rates and yield curves.

Oh and Han, 2000 used a combination of change points and backpropagation neural network (BPN) to predict US interest rates. Their hypothesis was that the interest rate movement has several change points due to monetary policy and by including these in the model, they would achieve higher accuracy. They find that the model outperforms the pure BPN model.

Zimmermann et al., 2002 claim that an Error Correction Neural Network (ECNN) model is an appropriate model for predicting systems with noise and missing parameters. They agree with Oh and Han that pure neural networks do not achieve the highest possible accuracy. Therefore, they introduce a variant-invariant separation through a bottleneck neural network to account for high-dimensional problems. The researchers concluded that their modified ECNN model outperforms classical machine learning algorithms such as the Recurrent Neural Network (RNN) and the Multilayer Perceptron (MLP).

MLP is often referred to as the standard neural network. It belongs within the category of feedforward neural networks, but to be considered an MLP the system must consist of at least three layers (input, hidden layer, and output). While Zimmermann et al. use MLP as a benchmark for their model, Hong and Han, 2002 use MLP as a starting point in their paper. In the paper, they introduce a data collector called Knowledge-Based News Miner in combination with the MLP algorithm to study the Korean interest rate. They find that a combination of neural networks and event information produces better results than a Random Walk and an MLP without event information. Hong and Han's findings are supported by Yasir et al., 2020 which also finds event information together with a convolutional neural network (CNN) model to be a viable method of predicting interest rates. CNNs have been primarily used for image classification and computer vision. However, recent studies have shown great results using CNNs for financial forecasting, for instance asset price predictions (Sezer & Ozbayoglu, 2018) and macroeconomic indicator forecasting (Smalter Hall & Cook, 2017).

Support Vector Machines (SVM) have also been used to predict interest rate movements. SVMs are praised for their robustness and ability to solve classification problems. Jacovides, 2008 tested SVM against an MLP and found that the SVM produced more accurate results. Results from Gogas et al., 2015 support Jacovides' findings. They forecasted the yield curve of American interest rates with the intention of identifying recessions. Although their model produced some "false alarms", the model accurately predicted all recessions.

Machine learning has also had a wide variety of applications across finance and macroeconomics beyond interest rate predictions. The long short-term memory (LSTM) algorithm, first developed by Hochreiter and Schmidhuber, 1997, is considered by many to be a modern artificial neural network. LTSM is an extension of the classic RNN. The algorithm's advantage is its ability to store long-term information. This makes it attractive for processing time series where important information may lie in trends. Little research has been done on LSTM and interest rates. However, research has been done on LSTM's ability to predict the stock market. The results from these reports suggest that LSTM performs better than traditional methods ((Sirignano & Cont, 2019); (Lanbouri & Achchab, 2019); (Z. Zou & Qu, 2020); (Qiu et al., 2020)). Kim and Swanson, 2014 showed that hybrid shrinking methods, such as elastic net, do particularly well in predicting macroeconomic and financial variables. Kuzey et al., 2014 showed that decision tree algorithms can be used to determine the relative importance of firm performance metrics. West et al., 2005 showed that ensemble methods were better equipped at generalization of financial decision making such as bankruptcy classification and credit scoring.

Given the findings in this review, there is much to suggest that machine learning predicts interest rates better than traditional methods. In fact, we have not found a single research article that concludes otherwise. We assume that this is related to the fact that the use of machine learning in finance is relatively new and that there is some confirmation bias in the research. In addition, we note that there are many different machine learning algorithms, each with its own specialty. In the subsequent review of methodology, we will present the algorithms we believe are well suited for the specific task at hand.

3 Methodology

Machine learning is a subfield of artificial intelligence that focuses on computer algorithms' ability to learn from data by itself and ultimately predict its outputs. This is different from the data modeling paradigm of traditional statistical tools such as linear regressions, which requires several assumptions regarding the underlying process (Breiman et al., 2001). This can make machine learning particularly attractive for forecasting purposes as these algorithms can capture the underlying pattern in the data without necessarily knowing it (Zhang & Hu, 1998). In addition, the era of big data has further prompted the relevancy of such algorithmic models because of its efficacy in handling large amounts of information (Zhou et al., 2017).

Our hypothesis is that machine learning algorithms are able to predict the policy rate with higher accuracy than Norges Bank's model. In this chapter, we will explain our methodological approach to testing the hypothesis. First, we describe general considerations related to performing a test using machine learning algorithms. Then we describe the functionalities and peculiarities of the selected algorithms.

3.1 Training the machine

Several elements of defining machine learning algorithms are recognizable to researchers working with classical regression models. Data processing and model evaluation are in principle the same, while parameter definition and error minimization are to some degree dissimilar. In the six upcoming subchapters, we will highlight key elements of defining and running machine learning algorithms. Train-test split and under- versus overfitting should be familiar subjects to researchers working with classical regression models, while cross-validation, hyperparameter tuning, and gradient descent may be perceived as new topics.

3.1.1 Train-test split

Whether you use the classical linear regression model or a neural network for predictions, it is crucial to split the data into at least two parts; (i) training data and (ii) testing data. It is important to differentiate between these two subsets to get an unbiased estimate of the model's performance. In extreme cases, one could define a model that performs perfectly in-sample but fails miserably when applied to out-of-sample data. In-sample data, or the training data, is used to tune the parameters of the model, while the out-of-sample data, or the test data, is used to measure the efficacy of the model.

There is no definite answer as to how one should split the data. The main concern is the variance of the training performance versus the variance of the test performance. The models require enough training observations to tune the parameters during training, as well as enough test observations to attain a robust evaluation of the generalizability of the model. This is less of a problem as the number of observations increase, because any split would result in a large amount of observations on either side of the split. Our dataset consisted of 159 observations, and we allocated 80 percent of the data to training and 20 percent to testing

3.1.2 Under- versus overfitting

The ultimate goal of machine learning models is to generalize the pattern in the training data in such a way that the fitted model is able to make correct predictions on unseen data. This process is often referred to as generalization. Underfitting and overfitting are symptoms of a model that is unable to generalize properly. Underfitting means that the model is too simplistic to explain the underlying process for which we are modeling. Overfitting is the opposite; the model is too complex, making it too rigid to adapt to new information (Figure 2).

Overfitting is more common than underfitting for machine learning models. This is linked to several aspects of how the data is structured and how models are defined. For example, when fitting a model, the model builder must define the number of epochs. Epochs refers to how many times the training set is passed through a neural network. A low epoch-number restricts the model's ability to learn patterns, while a large number overfits the training data and causes errors in test predictions. This balancing act is known as the bias-variance trade-off. In Figure 3, we see that the prediction error of the training set declines with the model complexity. This is also true for the test error until the model becomes too complex and the test error increases.

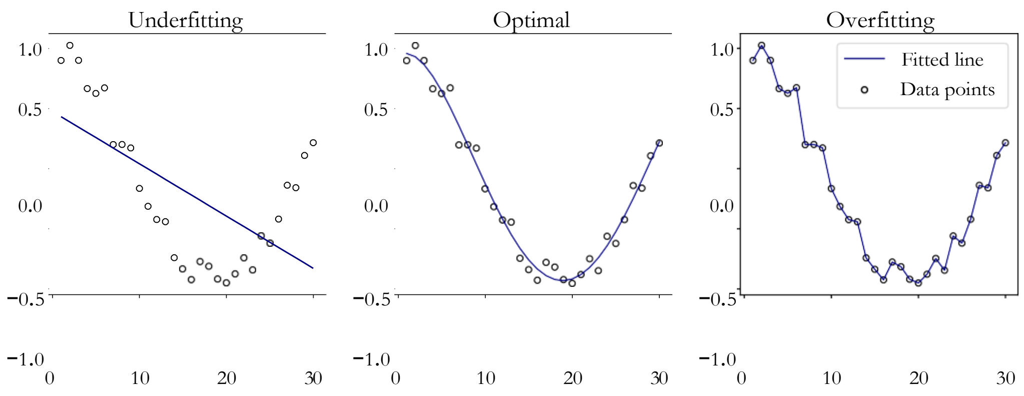


Figure 2: Example of overfitting and underfitting. The left plot corresponds to an underfitted model This is visualized by the fitted line deviating substantially from the data points. The mid plot corresponds to an optimized model. This model is fitted to assimilate the underlying process, but not too complex to warrant high out-of-sample error. This is visualized by a smoother curve that follows the trend of the data points. The right plot corresponds to an overfitted model. Here, the fitted line perfectly follows the data points. This will most likely result in poor out-of-sample performance as the model is too rigid to adapt to new data.

There exist techniques that inhibit sub-optimal model construction related to overfitting and underfitting. A common preventive technique used with neural networks is early-stopping. Early-stopping is a function that monitors the validation loss of the model and stops the algorithm from learning when a minimum target of improvement in performance is reached. You are in essence trying to withhold the model from proceeding beyond the optimum point in Figure 3. We used early-stopping for both of our neural networks.

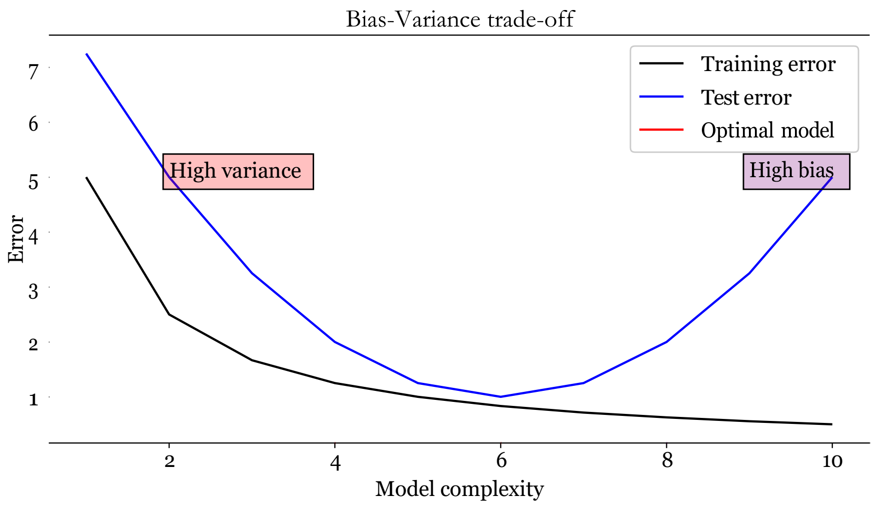


Figure 3: Bias-Variance trade-off. Overfitting can be defined as when the validation loss starts to increase and training loss decreases. This means that the model starts to fit itself

to noise in the training data. This results in lower training error but higher test error.

3.1.3 Cross-validation

Cross-validation is used to evaluate the generalizability of the model. This process validates the parameters of the model, making sure that they are optimally adjusted to perform well out-of-sample, rather than in-sample. Consequently, this reduces the chance of overfitting. A commonly used cross-validation method is *k*-fold cross-validation. For this method, the data is shuffled and partitioned into *k* subsets, in which one subset is kept for validation while the model tunes its parameters on the rest. This is not an appropriate method for this analysis, because we must consider the temporal aspect of our data. Hence, we instead use walk-forward validation (Figure 4). Walk-forward validation is similar to *k*-fold cross-validation, in that the dataset is partitioned into subsets. However, the data is not shuffled at the start and the order of the subsets is kept intact. The model is then trained on the subsets in chronological order, adding the test subsets to the training subset as the subsets are trained on.

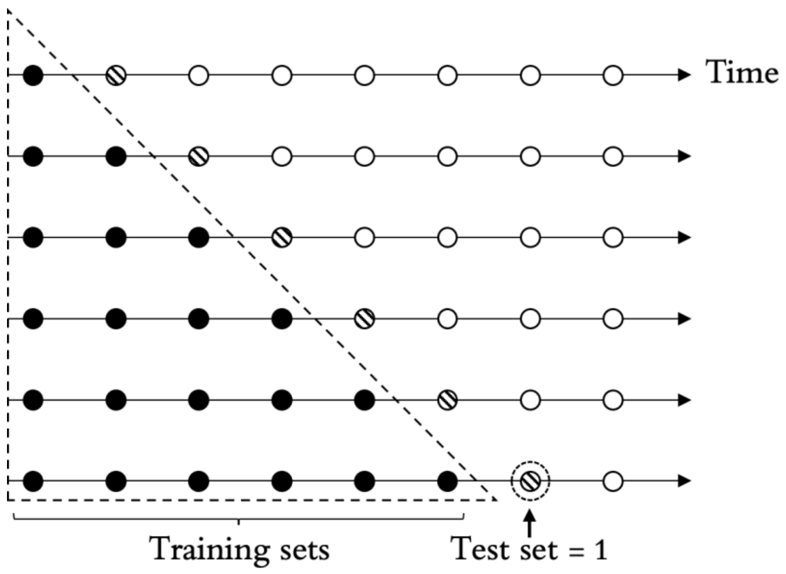


Figure 4: Example of walk-forward-validation (Hyndman & Athanasopoulos, 2018). This example consists of a dataset partitioned into eight folds. The temporal order of the folds goes from left to right. For the first validation, the first fold of the dataset is used as training data, while the second fold is used as a validation set. In the next iteration, the first and second fold are used as a training set, while the third fold is used as a test set. This iterative process proceeds until there are no folds left or a certain criteria is met, such as number of folds left untouched.

3.1.4 Hyperparameter tuning

Hyperparameters are parameters manually set by the model builder before the model is fitted. Conversely, model parameters are adjusted automatically during the learning process. An example of a hyperparameter is the number of layers in a neural network. This is a choice the model builder makes before the learning process, which affects how the model is tuned and how it learns underlying patterns. The weights of a neural network on the other hand, is an example of model parameters. These parameters are adjusted during the learning process and will affect how the input from a previous layer is transferred to another.

We can optimally adjust these hyperparameters by splitting the training data into a section called validation set. Here, we train the model while adjusting the hyperparameters for each training session and choose the hyperparameters that optimize our predictions.

3.1.5 Gradient descent

Some may argue that gradient descent is the most important element to differentiate machine learning from classical regressions. Gradient descent is an optimization algorithm that tunes the weights of the parameters in an artificial neural network to minimize a cost function. This minimization process is a function of the gradient of the dataset, which is the direction and rate of fastest increase at a specific point in a graph. Hence, the gradient can indicate the shortest path to a minimum- or a maximum point.

Since we want to minimize the errors of our neural networks, we want to find the minimum point of the cost function. Consequently, we must use the negative gradient as a compass for which direction to proceed. Backpropagation, short for backward propagation of errors, is a well-suited tool to compute the gradient. Backpropagation is a method for calculating the gradient more efficiently, in which the gradient is computed by going backwards in the network, using information from the prior layer. The technical details of backpropagation are outside the scope of this thesis. However, it is important to highlight the overall function of gradient descent and backpropagation, and to understand how these algorithms interact to fine-tune neural networks. A way to think about this relationship is that gradient descent is doing the learning itself, while backpropagation is outside the learning process, guiding where the learning should go.

The gradient descent is illustrated in Figure 5. It starts by placing the ball at a random starting point by assigning the weights some arbitrary values. Next, the gradient is computed, for instance through backpropagation. The ball is then shifted in the direction of the negative gradient. The algorithm reiterates until the algorithm is stopped manually or by a built-in function such as early-stopping.

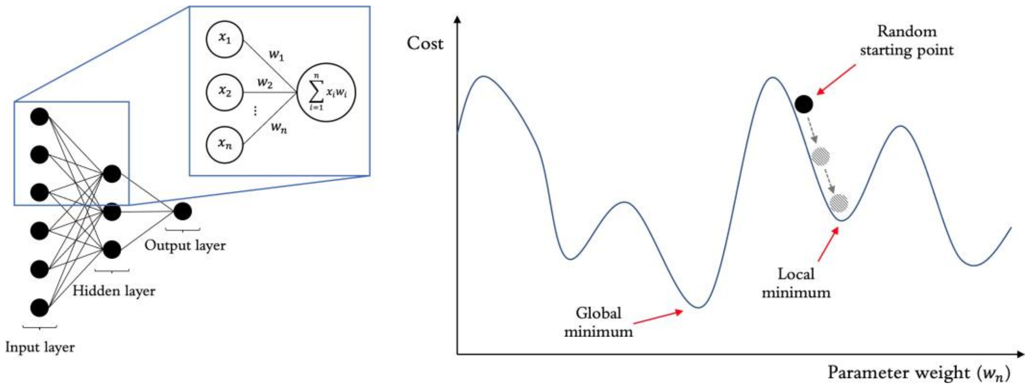


Figure 5: Illustration of gradient descent in a neural network. The left figure is a simplified version of a fully connected neural network illustrating how the parameter weights are the links between the nodes. The right figure illustrates the relationship between the cost and the parameter weights, and how the backpropagation function affects the gradient descent.

While studying Figure 5, one might notice that there are several minimum points. Since the optimal parameter weight is where the cost is minimized, we prefer the ball to move towards the global minimum. However, if the ball is placed so that the ball "falls" towards a local minimum point, the fitted model ends up being unoptimized. And since this process is unobservable, there is no way to detect such problems. One can only guess that this is a problem and try to counteract it.

3.1.6 Measuring efficacy

So far, we have discussed data preparation and how to prepare the model for training. As previously mentioned, the model is trained by minimizing a cost function. The cost-function of choice will have drastic consequences for how the model is fitted, which subsequently affects the results. Hence, differences across cost functions and the most appropriate one for our hypothesis is a relevant discussion.

There are numerous cost-functions one can use. Perhaps the two most utilized regression cost-functions are the mean absolute errors (MAE) and the mean squared errors (MSE). Both measures respect the issue of cancelling out errors by either squaring or taking the absolute value of the errors. However, the difference between the measures lies in their sensitivity to outliers. The MSE is more sensitive towards outliers because it squares the errors, which exponentially increases the cost-function for greater residuals. Hence, it makes more sense to use the MSE if large outliers are problematic. Using the MSE will ensure that the model adjusts itself towards lessening large deviations from the true value, rather than purely looking at all errors proportionally equal.

Given the outlier sensitivity of the different cost functions, we assess the MSE to be the more suitable cost function for the purpose of this thesis. As previously argued, the policy rate affects several aspects of the economy, from inflation to pricing in the financial markets. Furthermore, policy rate forecasts themselves are an indicator for the ensuing economic growth, affecting assumptions regarding future cash flows. Hence, having proper forecasts of the policy rate might enable governments to more easily prepare for the future state of the economy and lead to less volatile financial systems. Consequently, we presuppose that it is better to facilitate forecasts that predict the general direction of the policy rate rather than minimize smaller errors throughout the whole prediction period, as we believe this reduces the chance of large deviations due to unexpected changes.

3.2 Machine learning algorithms

We have selected five machine learning algorithms to test our hypothesis. The algorithms are listed in Table 1. Every algorithm is suitable for processing large amounts of data and for making time series predictions. Each has their own specialties which we will describe in detail in subsequent subchapters.

We used Python and the machine learning packages “Keras” and “sklearn” to define and run the algorithms. When defining the CNN- and LSTM models, we used a built-in Keras function called "Sequential". This feature enabled us to stack layers of machine learning functions one after the other, which in turn let us build complex and customized models. One can theoretically build an infinite number of different versions of the algorithms. Our model building strategy was therefore to define relatively simplistic models with only the necessary function layers. This is consistent with the purpose of this thesis, which is to test the efficacy of machine learning algorithms as a forecasting tool, not to optimize the algorithms to the specific task at hand.

Selected algorithms	**Type of machine learning algorithm**
Elastic net	Regularization
Convolutional neural network	Deep learning (neural network)
Decision tree	Classification and regression tree (CART)
Long short-term memory	Deep learning (neural network)
Ensemble learner	Stacked generalization

Table 1: Selected machine learning algorithms and their algorithm type.

3.2.1 Elastic net

The elastic net algorithm is closely related to the classical linear regression. The main difference is that it uses regularization. Regularization, in the context of machine learning, is a technique that expands the cost function of the algorithm to improve out-of-sample accuracy. The classical linear regression is sensitive towards bias, which means that the coefficients of such models typically exhibit a substantial amount of variance. This tends to make classical linear regressions poorly equipped to generalize beyond in-sample data, especially if the underlying data is high-dimensional. The elastic net algorithm tries to circumvent this issue by trading variance for bias by employing two regularization techniques: (i) lasso and (ii) ridge regularization.

(i) The lasso regularization estimates the coefficients of the model subject to the sum of the absolute value of the coefficients (Tibshirani, 1996). We can write the cost function as:

$$\hat{\beta} = arg\ \min \sum_{i=1}^{n} (y - \mathbf{X}\hat{\beta})^2 + \lambda \sum_{j=1}^{p} |\hat{\beta}| \tag{5}$$

The process of estimating λ is sometimes referred to as regression shrinkage. This regularization method tends to produce zero-coefficients.

(ii) The ridge regularization is particularly good at minimizing the coefficients that are correlated with each other, which in addition to improving precision, can reduce multicollinearity. We can write the cost function as:

$$\hat{\beta} = arg\ \min \sum_{i=1}^{n} (y - \mathbf{X}\hat{\beta})^2 + \lambda \sum_{j=1}^{p} (\hat{\beta})^2 \tag{6}$$

Suppose the elastic net algorithm finds that GDP growths are irrelevant to predicting the policy rate and that cross-country CPIs are highly correlated. Then, the lasso regularization would set the parameters of GDP growth to zero, while the ridge regularization would reduce the coefficients related to the aforementioned CPI variables proportionally.

We use the elastic net algorithm because it solves three problems. (i) The ridge regression usually fails to come up with parsimonious models as it never sets any of the parameters equal to zero (H. Zou & Hastie, 2005). Furthermore, (ii) the lasso regression will at most include n number of parameters out of p candidates (Efron et al., 2004). (iii) The elastic net algorithm, as well as the other regularization algorithms, reduce overfitting by shrinking, or even eliminating, coefficients. The elastic net algorithm estimates the coefficients of the regression given the following constraint:

$$\hat{\beta} = arg\ \min \sum_{i=1}^{n} (y - \mathbf{X}\hat{\beta})^2 + \lambda \sum_{j=1}^{p} (1 - \alpha)(\hat{\beta})^2 + \alpha|\hat{\beta}| \tag{7}$$

Coding-wise, we used scikit-learn's "ElasticNetCV". This package allows us to optimize the parameters using cross-validation. We used grid search for hyperparameter tuning of alpha and lambda. Grid search comprehensively searches the whole hyperparameter space to find the optimal values. We tuned the elastic net models using values of alpha from 0 to 1 with 0.1 increments.

For lambda, we used values on an exponential scale, starting from 1e-5 to 100.

3.2.2 Convolutional neural network

A neural network (NN) is a type of machine learning algorithm that mimics the brain's structure. It is an interconnected set of nodes that take some input which interacts with the neurons of the structure and ultimately produce some output (left-hand figure in Figure 5). These outputs are then calibrated in conjunction with its adjacency to the real value of the task. However, the nodes do not consider the order of the observations without any further modification. Hence, we found it reasonable to utilize a CNN, because they are better equipped for capturing spatial and temporal relationships.

The CNN algorithm has two particularly attractive attributes: (i) it is computationally efficient because it reduces the sample size, and (ii) it can find complex patterns in the dataset as the layers of the network will focus on smaller subset of the underlying process for which it can generalize to the data set as a whole (Ketkar & Santana, 2017).

The CNN algorithm is by far the most sophisticated algorithm used to test our hypothesis. Thus, a thorough explanation of its features and structure is beside the purpose of this thesis. We will, however, highlight key elements of how samples are managed throughout each layer in the algorithm, and clarify the intention of the algorithm's demeanor. The algorithm requires a three-dimensional dataset to operate as intended. Thus, we first altered the dataset so that each input contains several successive observations, as opposed to just one. A single input in the CNN algorithm is referred to as a "sample". Then we defined a CNN algorithm with six layers: A convolution layer, a pooling layer, a flattening layer, a dropout layer, and two dense layers. The purpose of the first four layers is to reduce the sample size and identify trends in the data, while the last two layers constitute a fully connected neural network. The one-dimensional convolution layer applies a pre-specified number of random kernels that "slides" along each sample (blue rectangle in Figure 6). The kernel is a type of filter that attempts to detect features and trends in the dataset. For example, the network may be able to identify how a long-term rise in consumer confidence combined with a decline in the Swedish three-month treasury bill results in an increase in the policy rate. The pooling layer further reduces the size of the sample by dropping all values but the largest within the defined windows. Next, the sample is flattened to facilitate a neural layer. The dropout layer is a regularization method that makes the CNN models more robust. The last two layers are fully connected layers that function as the traditional MLP neural network.

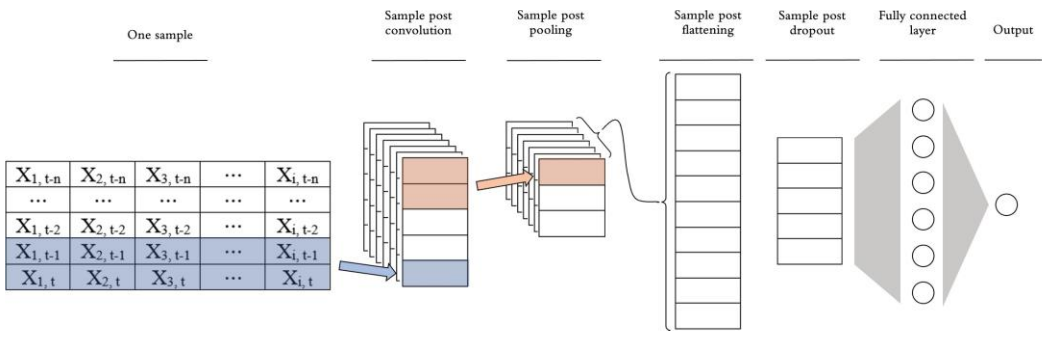


Figure 6: Illustration of how the CNN algorithm processes one sample. Each section in the illustration, except the "one sample"-section, corresponds to a layer in the CNN algorithm.

As previously mentioned, the CNN algorithm is defined using the "sequential" function within the Keras package. Each layer is modestly customized to our dataset using hyperparameter tuning. We tuned the CNN models by testing variations of activation functions, number of filters, kernel sizes, dropout rates, pool sizes, and number of dense nodes.

3.2.3 Decision tree regression

A decision tree is a well-known and widely used strategic tool for decision-making due to its ability to present processes in a simple and coherent way. The essence of the model is to understand action patterns and illustrate that an event has one or more outcomes that are related to each other. The decision-making tool has gradually been adopted by the field of machine learning because of its predictive abilities.

Decision tree regressions have several advantages compared to other machine learning algorithms. First, the algorithm is easy to define and understand. Second, it has an innate ability to select features. Thirdly, it requires little computer power, relatively speaking. Lastly, non-linear relationships between features will not affect the model's performance. On the other hand, decision tree regressions are prone to disadvantages. The model is easily overfitted and may be subject to poor model variance. Variance refers to how much the prediction changes when you change the underlying data. Decision tree regressions will also create biased trees if certain classes dominate.

Two factors are considered when building a decision tree regression model: which features should be included in the model and which conditions should affect the outcome space. At the root of the tree, the ability of all variables to predict the dependent variable is assessed using a cost function. The variable with the lowest cost is defined as the best predictor and is set as the root variable. The root variable is then split into branches and then eventually leaves using the cost function. The tree stops growing when the cost function is reduced to a minimum. The goal of the algorithm is to make a structure of economic variables that is able to predict the policy rate. For instance, it could be that the decision tree algorithm discovers that certain stock markets are the only features worth utilizing to predict the policy rate, thus making a decision tree consisting of OBX, S&P 500, DAX, FTSE, and Russell 1000.

A decision tree model trained on many features is prone to overfitting because the model will most likely find combinations of features that always reduce the in-sample cost function to zero. With 687 variables in our dataset, this problem is apparent. There are several ways to treat overfitting and it is mainly about limiting the tree's ability to split branches. For the decision tree models, we have chosen three parameters to reduce overfitting. The restrictions are placed on the depth of the tree, the number of samples per split and the number of samples per leaf. The parameters are defined in combination with the "GridSearchCV" function which is a hyperparameter tuning tool within sklearn. Figure 7 is a simplified illustration of a decision tree with depth = 2, minimum samples per split = 130 and minimum samples per leaf = 40. This figure showcases how decision trees make predictions; it will make a decision if a set of conditions are met. For example, if the change in OBX is less than 5 percent, and the change in S&P 500 is greater than 13 percent, then the decision tree predicts the policy rate to be 3.5 percent (Figure 7).

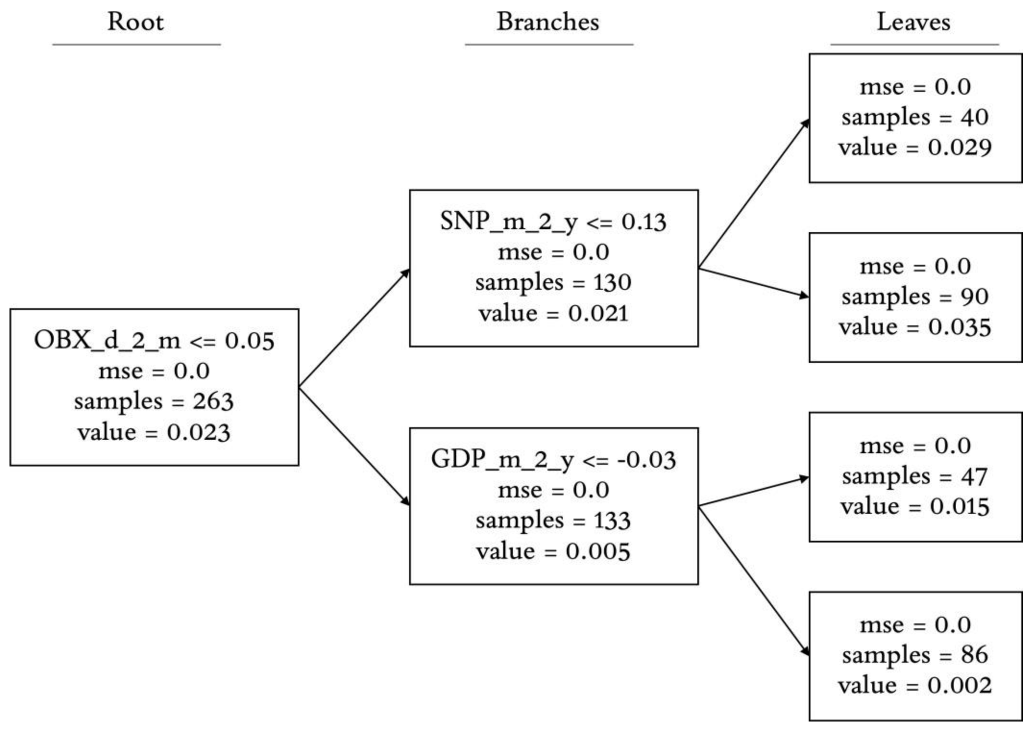


Figure 7: Example of a fitted decision tree. Each observation follows a path from left to right which is determined by the condition within the branches. The value of the output is then determined by the value of the leaves.

3.2.4 Long short-term memory

The long short-term memory (LSTM) neural network was first introduced by Hochreiter and Schmidhuber, 1997. LSTM models are trained using backpropagation through time and aims to solve the short-term memory problem of RNNs, formally referred to as the vanishing gradient problem. When passing information through an RNN, some information from previous steps is lost in the backpropagation process. RNNs use gradients to update the network and the problem arises as the gradient values deflate to insignificant values.

LSTM models solve the vanishing gradient problem by passing information through iterations and defining its importance. This is accomplished with gate units within memory cells. A memory cell consists of multiplicative input-, forget- and output gates (Figure 8). "Multiplicative" refers to how the sample vector is handled. A memory cell has three inputs and outputs: The observation X_i, candidate (input and output), hidden state (input and output), and the model output $\hat{y}$. The observation and candidate input are first combined and then passed through the gates (bottom-left corner in Figure 8). Information passed through the forget gate is subject to a sigmoid activation function and later combined with candidate inputs passing through the cell state. The purpose of the forget gate is to decide whether to keep or dismiss the information. Information passed through the input gate is transformed with both a sigmoid- and a tanh function. The tanh function regulates the network, and the output from the sigmoid function determines the importance. The input gate-output is then combined with the candidate information passing through the cell state. Lastly, the information is passed through an output gate and combined with the candidate information to produce a hidden state output and the prediction of y.

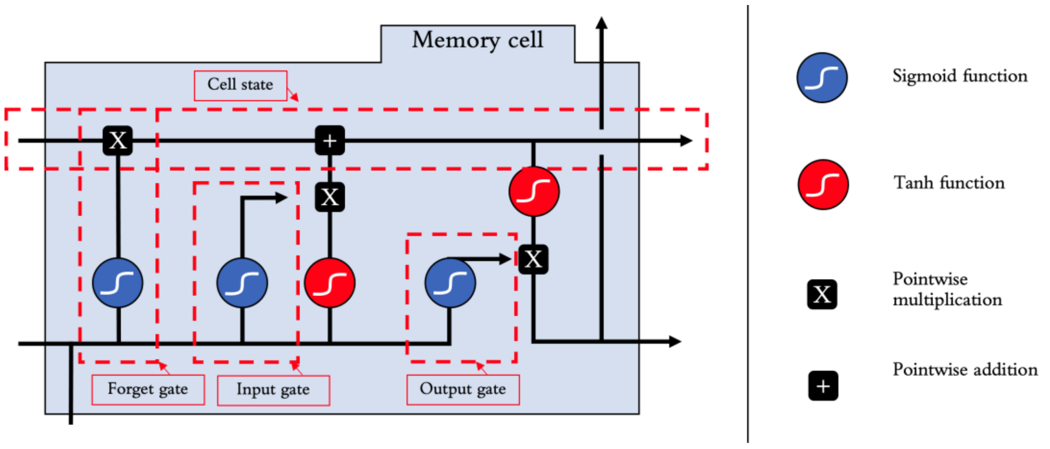


Figure 8: Illustration of a memory cell within the LSTM network. A new observation is passed to the cell from the bottom and information from previous iterations is passed from left to right on the horizontal inputs.

The details and technicalities of how inputs and outputs in the memory cell are treated are regarding the variable's behavior from the past is either important or worthless for the present. Important information is passed forward while worthless information is suppressed. In practical terms, this means that the models will use information on OBX' behavior in the past to predict the policy rate if that is important. For example, the models may learn that a negative change in the index one year ago is important for the policy rate today, while a positive change in GDP six months ago is worthless. Thus, the LSTM is able to identify long-term trends as opposed to other algorithms that treat each observation independently.

Our LSTM algorithm consisted of few sequential layers. We chose to use two LSTM layers because the initial testing proved that an additional layer increased the models' predictive ability. The only parameter used to tune the models was the number of units in each LSTM layer. One unit is equivalent to one memory cell.

3.2.5 Ensemble learner

An Ensemble Learning algorithm constructs a set of machine learning models and bases its prediction on a combination of the outputs from these models dependent on their optimal weighting (Dietterich, 2002). The ensemble learner algorithm's upper-hand relative to other machine learning algorithms is analogous to diversification in finance; by including several learning techniques (assets) in our model (portfolio), we reduce the noise (idiosyncratic risk) that is present in a single learning technique. More specifically, it reduces computational and statistical variance while reducing the chance of overfitting.

An ensemble learner algorithm has a set of base-models for which the meta-model fits itself to. The base-models are not chosen at random. In fact, it is an important part of the construction of the algorithm. When choosing which algorithms to include, it is important to construct a diverse set of machine learning techniques to facilitate the diversification benefit mentioned prior (Kuncheva & Whitaker, 2003). However, it is imperative to not add irrelevant algorithms either, even though they might increase the diversity of the set (Gashler et al., 2008). Based on these requirements, we have chosen the following algorithms displayed in Table 2, ranging from the standard linear regression model to the non-linear deep neural networks of LSTM and CNN.

When constructing the ensemble learner, the data is split in two parts; one for the base-models and one for the meta-model. Again, it is important to differentiate the subsets to get an unbiased estimate of the out-of-sample performance of the meta-model. The base-models are then fitted to the first part of the data. Furthermore, the meta-model is fitted to the out-of-sample predictions of the base-models. This dataset is then split further into training and testing. The meta-model in our case is an elastic net algorithm identical to the one used in former parts of the analysis.

Algorithm	**Algorithm type**
Linear regression	Modeling linear relationship with ordinary least squared.
Elastic net	Regression analysis that combines the lasso and ridge regularizations.
K nearest neighbor	Models relationships between variables by averaging observations of adjacent data points.
Decision tree	Using binary decision trees for predictions.
Adaptive boosting	Ensemble method where the base-learners are fitted sequentially.
Bagging regressor	Ensemble method where data for the base-learners are randomly sampled with replace-ment.
Random forest	Ensemble method where the base-learners are made up of decision trees.
Extra trees	Same as random forest but with random op-timization of tree split.
Deep neural network	Collection of interconnected inputs that produce outputs.
Long-short-term memory	Temporally sequenced neural network with longer memory.
Convolutional neural network	Regularized neural network through sub-samples.

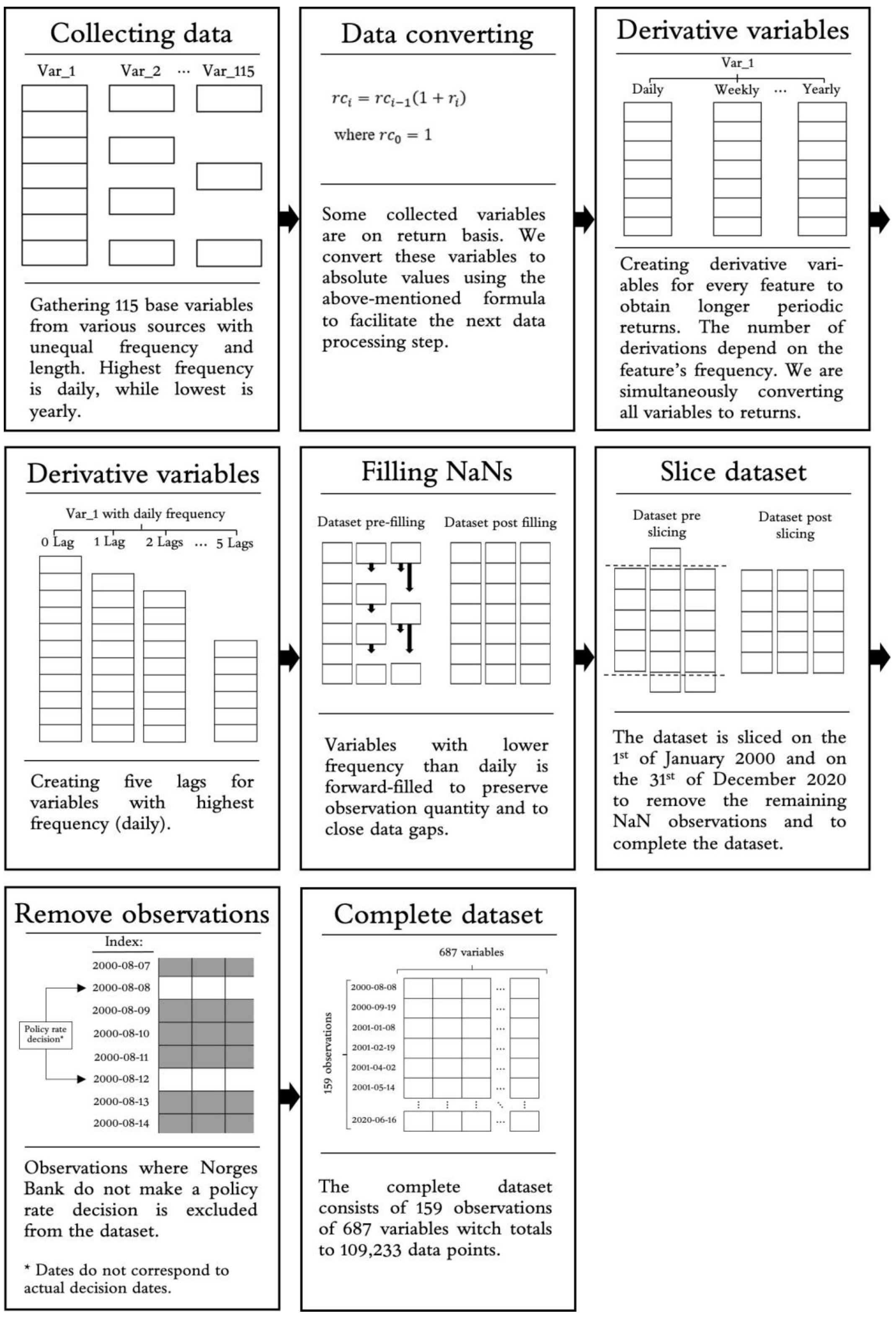
Table 2: Chosen base algorithms for the ensemble learner.

4 Data

Variable selection and proper data processing are instrumental in the success of machine learning. Our overall strategy for variable selection and processing was to include all variables that may influence the policy rate. This resulted in a high-dimensional feature space. We justify using many variables with the fact that machine learning algorithms have a built-in ability to detect and select predictive features, and that we want to utilize the algorithms' ability to detect economic contexts that may appear unrelated. This strategy mainly affected two data processing areas; the number of variables selected, and the number of observations included. We also employed a strict data gathering criteria in order to preserve data quality. This criteria entailed to exclusively use highly reliable sources. Our sources include: International Monetary Fund, Statistics Norway, Bloomberg, Oslo Stock Exchange, OECD, Finans Norge, and central banks such as Norges Bank, the European Central Bank, Bank of England, and the Federal Reserve[1].

Figure 9 shows an overview of the data preparation. We started by collecting data on 115 base variables and made conversions on some variables to get a homogeneous format. Then we created periodic returns and lags on variables depending on their frequency, and forward-filled NaN-observations[2]. Finally, we sliced the dataset on policy rate decisions to get a complete dataset. We will explain each step in detail in the following sections.

[1]See Exhibit 1 to see which variables are related to which source.
[2]NaN stands for Not a Number, which refers to a data point that is undefined.



Collecting data

Gathering 115 base variables from various sources with unequal frequency and length. Highest frequency is daily, while lowest is yearly.

Data converting

$rc_i = rc_{i-1}(1 + r_i)$

where $rc_0 = 1$

Some collected variables are on return basis. We convert these variables to absolute values using the above-mentioned formula to facilitate the next data processing step.

Derivative variables

Creating derivative variables for every feature to obtain longer periodic returns. The number of derivations depend on the feature's frequency. We are simultaneously converting all variables to returns.

Derivative variables

Creating five lags for variables with highest frequency (daily).

Filling NaNs

Variables with lower frequency than daily is forward-filled to preserve observation quantity and to close data gaps.

Slice dataset

The dataset is sliced on the 1st of January 2000 and on the 31st of December 2020 to remove the remaining NaN observations and to complete the dataset.

Remove observations

Observations where Norges Bank do not make a policy rate decision is excluded from the dataset.

* Dates do not correspond to actual decision dates.

Complete dataset

The complete dataset consists of 159 observations of 687 variables witch totals to 109,233 data points.

Figure 9: Overview of the data preparation.

4.1 Collecting base variables

When studying macroeconomic measures, especially the policy rate, it is important to consider the country itself and its relationship to other countries. Norway is a small and open economy, which means we must look for indicators that do not only pertain to Norway.

We started the data preparation by collecting a wide variety of international and domestic explanatory variables. We used Norwegian macroeconomic indicators such as the Consumer Price Index (CPI), Norway's Gross Domestic Product (GDP), and Producer Price Index (PPI). We also collected other countries' macroeconomic indicators such as interest rates, GDPs, and current accounts. Furthermore, we used stock market data for some of the bigger stock exchanges such as S&P 500, DAX, and OSEBX. Additional variables included oil prices, lending rates, industry indicators, and currencies. Our data collecting strategy was to include all variables that may, either by themself or in combination with others, affect the policy rate. In total, our dataset consisted of 115 base variables with varying length and frequency[3][4]. Issues related to the latter are covered in later sections.

To avoid look-ahead bias, we shifted the variables in conjunction with their publication schedule. Look-ahead bias occurs when using unavailable data at the time of prediction. For instance, the Norwegian household consumption is part of the national accounts. The national accounts are updated monthly but with 30 days publication lag. Thus, we shifted this variable 30 days backwards, e.g., the observation on December 12th, 2012 is first observed January 11th, 2013.

4.2 Derivative variables

In addition to the 115 base variables, we created 572 derivative variables. The number of derivative variables related to a specific base variable and the derivative variables themselves, depend on the frequency of the base variable. For instance, a variable on daily frequency produces derivative variables of daily change, weekly change, bi-weekly change, et cetera. A variable of monthly

[3]See Exhibit 1 for full list of base variables.
[4]See Exhibit 2 for correlation matrix.

frequency produces derivative variables of monthly change, bi-monthly change, quarterly change, and so forth. The variables that were already on a return-basis were converted to absolute values to produce derivative variables. This was achieved by computing the cumulative return of the variable:

$$rc_i = rc_{i-1}(1 + r_i), \text{ where } rc_0 = 1 \tag{8}$$

The resulting dataset consisted of 687 explanatory variables, all on return-basis. Since all the variables were in returns, we circumvented the issue of non-stationarity. Stationarity refers to a stochastic process in which the mean and variance of that process do not change over time. This can result in spurious coefficients as the model picks up on the trend in the process.

Stationarity is strictly necessary for the models to function as intended. A decision tree model, for example, would fail to make predictions if variables are increasing over time. However, taking first differences in combination with reducing frequency of variables has unintended side-effects on rare occasions. This is evident in for example the US Yield Spread. Figure 10 shows both the cumulative return and the first difference of the yield spread. The cumulative return is evidently non-stationary, which suggests differencing to be an appropriate approach to achieve stationarity. As the right-hand figure suggests, the process seems to be stationary, but we simultaneously create extreme outliers.

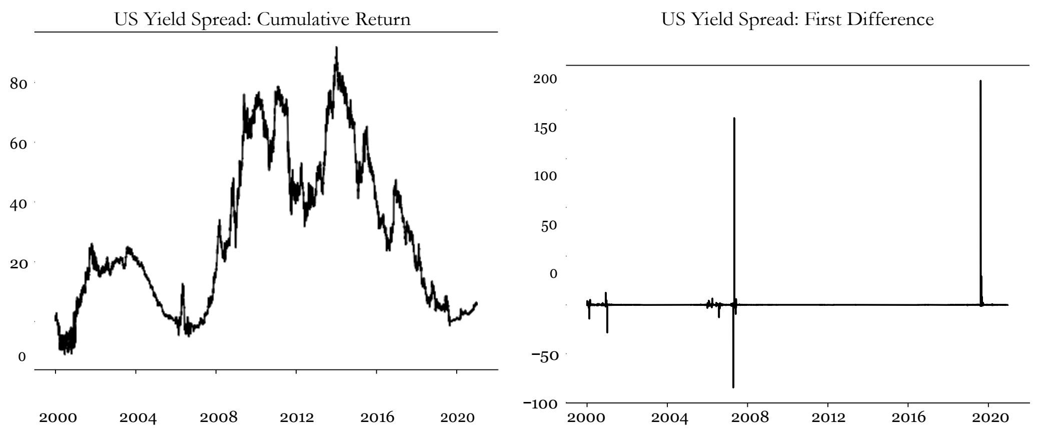


Figure 10: Comparison of cumulative- and first difference of the US Yield Spread from January 2000 to December 2020.

We encountered a handful of observations with values equal to infinity or minus infinity. Instead of removing the observation, we replaced these data points with zero. Some may argue that such data management is incorrect, as infinity does not equal zero. However, we assessed the benefit of including all the other values within these observations as greater than the cost of some of the observations containing potentially "incorrect values".

In addition to creating derivative variables based on periodic returns, we created lags of daily variables. We assessed a week of lags ($n = 5$) as appropriate to capture enough information while simultaneously not overcrowding the dataset with irrelevant variables. The reason we included lags of variables with daily frequency was to capture movements in leading indicators. The idea is that a sudden movement in, for example, the OBX index can influence the policy rate a few days after the event. Since the policy rate is not fully market-driven, there will usually be a lag in the policy rate movement. Figure 11 shows how the OBX index and the policy rate moved at the beginning of the covid-19 pandemic. It is clear that the policy rate reacts with a lag in relation to the OBX index.

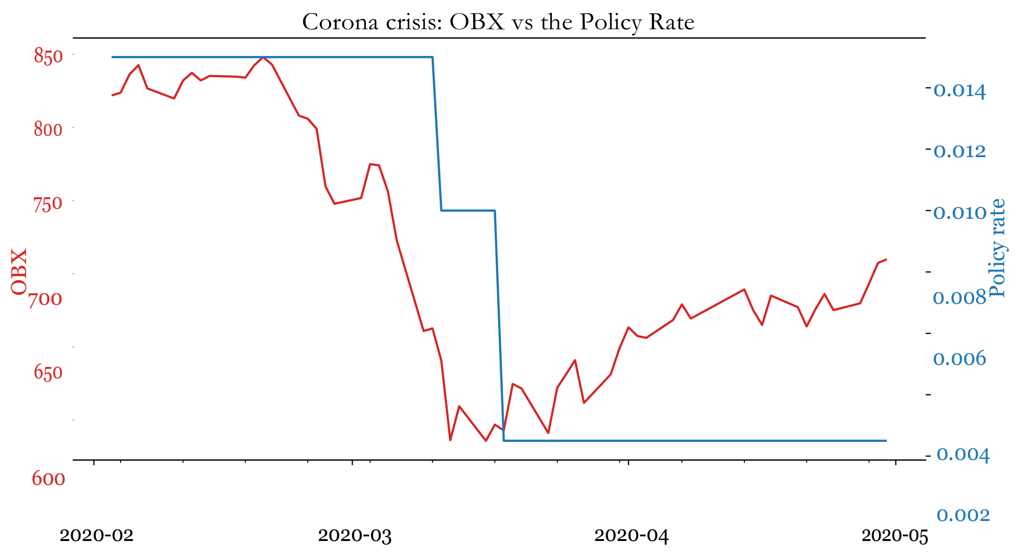


Figure 11: Plot of the OBX index and the policy rate from February 1st, 2020 to May 1st, 2020.

4.3 The dependent variable

The dependent variable in this study is the Norwegian policy rate. Norges Bank first offered interest rates on banks' deposits in January 1991. Before 1991, the central bank used the D-loan rate, which is equivalent to the interest rate on the bank's loans (Norges Bank, 2015). We assessed the D-loan rate as dissimilar to the current policy rate and therefore found it inappropriate to combine them. Consequently, the first observation was in January 1991.

To avoid look-ahead bias and to make sure that we were able to compare our models to Norge Bank's own forecasts properly, we shifted the policy rate two observations back in time. There are two reasons behind this decision. Firstly, as indicated in the latest monetary policy report, Norges Bank uses data up until the date of the committee's rate decision meeting, which is a day prior to the publication itself (Norges Bank, 2021). Additionally, we assume that Norges Bank is not able to gather intraday closing data for all their variables. For instance, we do not expect Norges Bank's meeting to be after the closing of S&P 500. Thus, we shifted the variable once more for a total of two shifts.

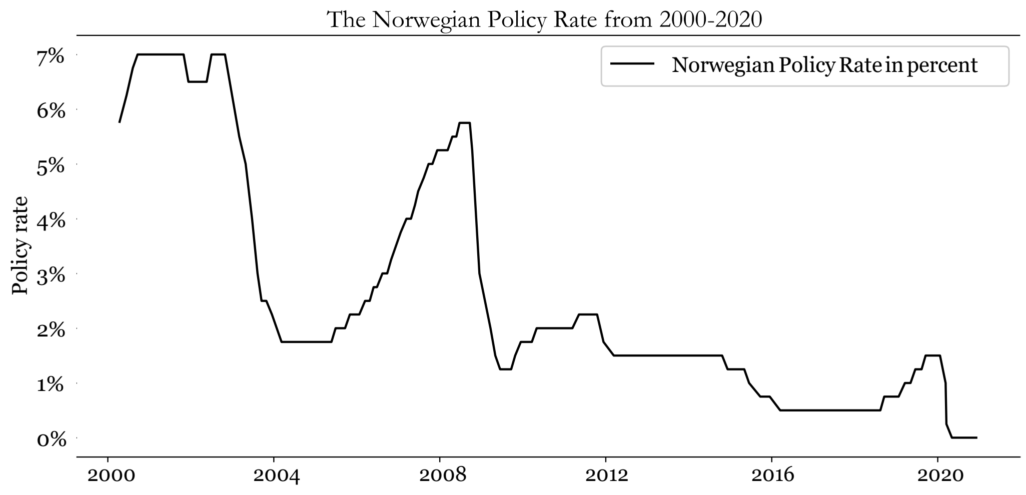


Figure 12: The Norwegian policy rate in percent from January 1st, 2000 to December 31st, 2020.

Based on the plot in Figure 12, the policy rate seems to be non-stationary. The rate is trending downwards, and the variance is not constant within the selected time period. To test its stationarity, we used an augmented Dickey-Fuller test. The result is presented in Table 3. We see that the null hypothesis is not rejected on both the 1 percent, 5 percent, and the 10 percent significance level. Thus, the policy rate is non-stationary.

Then, we tested if taking first differences makes the policy rate stationary. Since the *p*-value is 0.0011, we reject the null hypothesis on all presented significance levels. This implies that we succeeded in making the process stationary. Thus, our models will be constructed to forecast a change in the policy rate rather than the absolute value of the policy rate.

	Policy rate	**First difference**
ADF statistic	-2.5257	-4.0612
p-value	0.1093	0.0011
1%	-3.4727	-3.4760
5%	-2.8801	-2.8816
10%	-2.5767	-2.5774
n	160	159

Table 3: Augmented Dickey-Fuller test of the policy rate and of the first difference of the policy rate.

4.4 Heterogeneous feature frequency

The data we collected were of varied frequency. We solved this by fitting the variables to a complete data frame, removing observations with all NaNs, and forward-filling missing observations. For instance, if the observation on January 1st was 100 and the observation on January 10th was 105, the observations between January 1st and January 10th equals 100. Some may argue that filling missing data points is an inappropriate method. We argue, however, that removing observations is worse. Macroeconomic variables such as GDP are published with low frequency which produce severe gaps in the data set. If we were to remove all observations containing NaNs, the total number of observations would be close to zero, which defeats the purpose of using machine learning. By forward-filling NaNs, each observation contains the most recently observed information of each variable.

4.5 Data length

The collected data were of varied length as well. The algorithms require a balanced dataset in order to be fitted, which means that we had to do a cost-benefit analysis regarding which variables to include versus the number of observations to cut from the dataset. The economic union of the European Union (EU) was established in 1998, which means that most of the data related to the EU is first observed around the year 2000. Thus, we found it appropriate to exclude data before January 2000 as it allows us to keep important variables related to the EU. This resulted in the exclusion of 10 base variables[5]. Furthermore, we excluded observations after August 1st, 2020. We did this because of the publication inconsistencies. Extending the data length beyond this point would exclude important variables.

4.6 Data frequency

At this stage of the data preparation, we had to make a decision regarding the data frequency. We could either keep daily observations or cut the dataset to accommodate a certain frequency. The benefit of the former is that the dataset then consists of far more observations, which is an important prerequisite for most machine learning algorithms to properly fit the data. On the other hand, this would result in a dataset with mostly repeated data points and a dependent variable that primarily consists of zeros. This partly defeats the purpose of exposing the machine learning algorithms to many different environments. Additionally, the machine learning algorithms are then optimized to predict a near constant process. Pre-testing showed that the algorithms struggled to do so and mostly predicted no change for the whole testing period. Hence, we chose to cut the data frame in such a way that the algorithms instead predicted subsequent policy rate decisions.

Cutting the dataset such that each observation corresponds to a policy rate decision was done to emulate Norges Bank's own forecasts. As previously mentioned, Norges Bank publishes their forecasts for each policy rate decision, where the forecasts are partly based on the current policy rate decision. Similarly, by cutting the data set by each decision, the models essentially make forecasts at each policy rate decision, using the information currently available at that time.

4.7 Data preparation output

The resulting data set contained 687 explanatory variables and 159 observations for a total sum of 109,233 data points (Figure 13). The first observation is on August 8th, 2000, while the last is on June 16th, 2020. We believe that

[5]See Exhibit 3 for a list of all variables that were deleted.

the number of data points is sufficient to do robust testing of the machine learning algorithms.

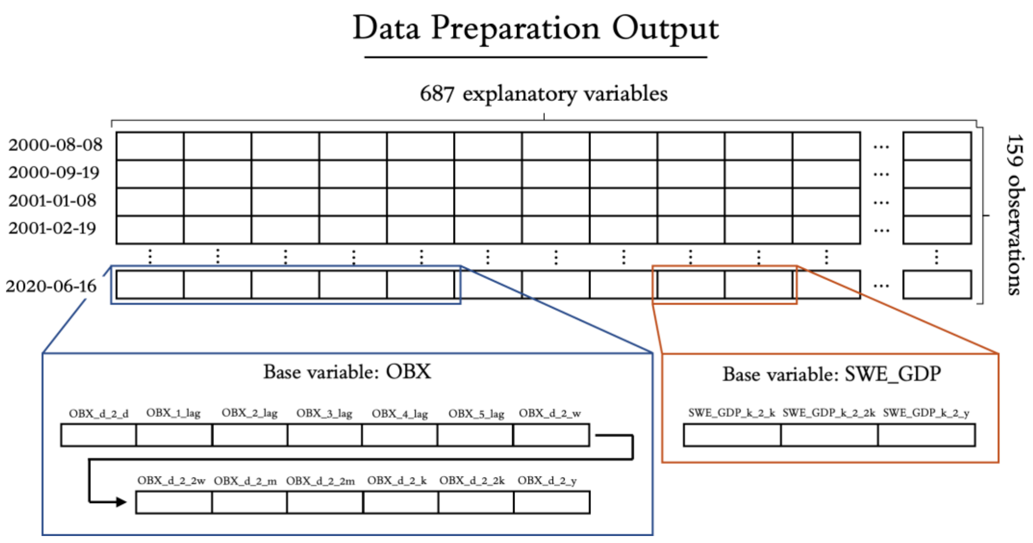


Figure 13: Illustration of the output from the data preparation.

5 Results

When Norges Bank publicizes their policy rate decisions, they create a forecast of the policy rate from the date of publicization to 15 quarters in the future. To assess the algorithms and to properly compare the algorithms to Norges Bank's own forecasts, we performed multi-step forecasting as well. However, instead of forecasting 15 quarters in the future, we limited our forecast to four policy rate decisions ahead. The rationale behind this decision was that; (i) the probability of having enough quality information to forecast that far into the future seemed low, and (ii) the practicality of forecasts beyond four decisions is sparse given the amount of time between decision-making and outcome.

To do multi-step forecasting, we opted to create several models for each algorithm. Each model within each algorithm forecasts the policy rate for a specific step-ahead. The periods for which we forecasted ranged from nowcasting to four decisions ahead, resulting in five models per algorithm. Thus, we created 25 different models (Figure 14).

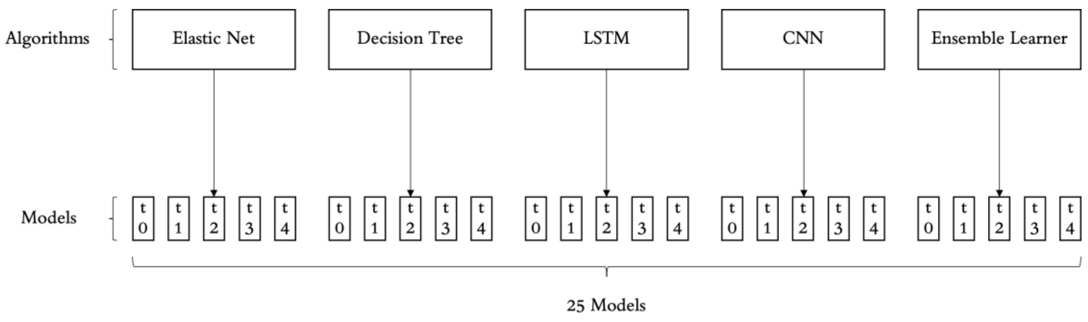


Figure 14: Multi-step forecasting construction. Five models with different time-steps are constructed per machine learning algorithm.The time-steps range from $t = 0$ to $t = 4$. $t = 0$ corresponds to nowcasting, while $t = 4$ corresponds to forecasts of four policy rate decisions in the future.

By combining the out-of-sample predictions for all the models, we produced a data table with multi-step predictions for each observation across all algorithms (Figure 15). For instance, the observation with the date "2016-12-13" had columns corresponding to nowcasting at that date, the prediction for one decision ahead of 2016-12-13, the prediction for two decisions ahead of 2016-12-13, and so forth for all the algorithms. This allowed us to easily plot and analyze the multi-step predictions. See Figure 17 for a sample of an observation. These observations, and subsequently the plots, are supposed to mirror

Norges Bank's visualization of the policy rate forecast[6].

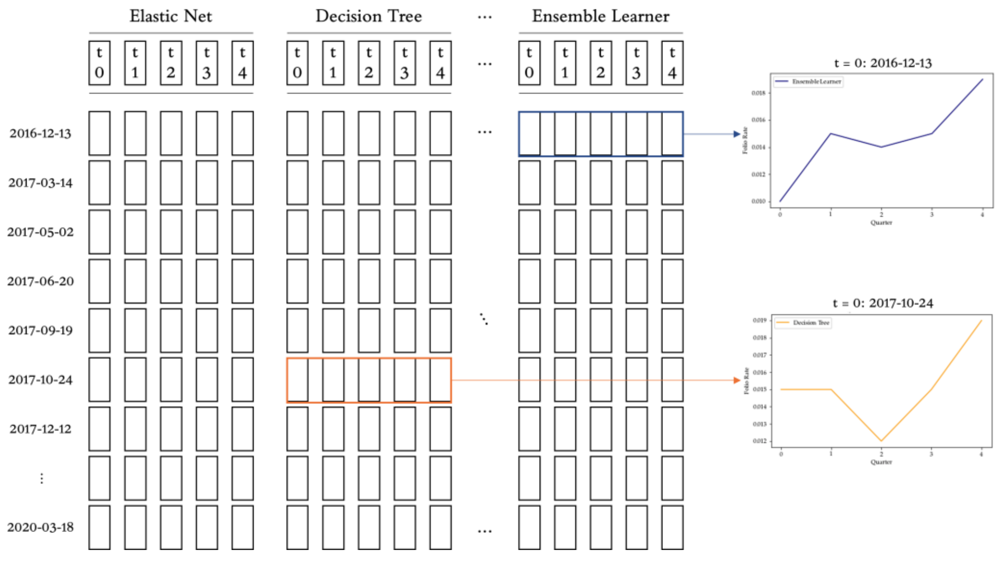


Figure 15: Complete data frame of multi-step predictions. Each algorithm employs five models for time-steps $t = 0$ to $t = 4$. Concatenating the model outputs results in a data frame with 25 columns where each row corresponds to the time-step prediction at that time. For instance, $t = 0$ predictions in the row "2016-12-13" corresponds to the algorithms' predictions for that date. Furthermore, $t = 1$ predictions in the same row corresponds to the algorithms' prediction for the next policy rate decision, i.e., March 14th, 2017. This allows us to easily plot the figures on the right-hand side. One can think of one row as the predictions every model makes at that particular date, in which the time until the prediction occurs will vary proportionally from $t = 0$ to $t = 4$.

As mentioned in Section 3.1.6: Measuring efficacy, we opted to use the MSE as a cost-function for tuning the models. We could use the same cost-function to assess the out-of-sample model performance. However, we instead chose to use the root mean squared error (RMSE). RMSE is an extension of MSE. It represents the standard deviation of the residuals and is sometimes preferable as it is scaled in accordance with the data (Hyndman & Koehler, 2006). Hence, RMSE is doing the same computation as the MSE by squaring the residuals. However, by taking the root of the MSE, we make sure that the measure is more comprehensible and on a scale that is equal to the policy rate.

5.1 Benchmark

We have chosen to compare the results from our algorithms to that of Norges Bank's own predictions. Norges Bank generally forecasts the policy rate on a quarterly basis. This means that we must either make predictions on a

[6]See Exhibit 4 for reference to example of plot.

frequency equal to the frequency of Norges Bank's quarterly forecasts or adjust Norges Bank's forecasts to reflect its predictions at each policy rate decision. We opted for the latter to extend the out-of-sample data size, increasing the number of observations from 14 to 26. Our out-of-sample predictions start at 2016-11-01, which means that the first out-of-sample rate decision is 2016-12-13. We gathered the benchmark data from Norges Bank's monetary policy reports (Norges Bank, 2021).

5.2 Zero-to-four step-ahead predictions

Table 4 presents the zero-to-four step-ahead RMSEs for the algorithms and the benchmark. $t = 0$-predictions are nowcasting errors, while $t = 4$-predictions are errors of four step-ahead predictions. The table also presents the best algorithm for each time-step and the percentage difference between the best algorithm and the benchmark.

	$t = 0$	$t = 1$	$t = 2$	$t = 3$	$t = 4$
Norges Bank	0.002768	0.004041	0.004607	0.005562	0.006305
LSTM	0.002023	0.003177	0.004563	0.005084	0.005873
Ensemble Learner	0.001849	0.003379	0.004681	0.005547	0.006003
CNN	0.002521	0.003667	0.004581	0.004971	0.006239
Elastic Net	0.001725	0.002949	0.004188	0.005076	0.006166
Decision Tree	0.001592	0.002887	0.004034	0.005098	0.005650
Best Model	DT	DT	DT	CNN	DT
Difference	0.5531	0.3362	0.1326	0.0915	0.1096

Table 4: RMSEs across time-steps. DT refers to the decision tree algorithm. The algorithms are showcased vertically, while the time-steps models are represented horizontally. The next to last row corresponds to the best model for the specific time step. The last row corresponds to the difference between the best model and Norges Bank's model for the specific time step. This is computed by the logarithm of the values corresponding to the two models: $ln\left(\frac{NB_i}{Y_i}\right)$, where Y_i is the best-performing algorithm for the specific time-step i.

There are three aspects of the model performances that are evident by investigating Table 4. Firstly, the decision tree algorithm consistently outperforms the other algorithms across step-aheads. The only step for which the decision tree algorithm ceases to outperform is at $t = 3$. However, the difference be-

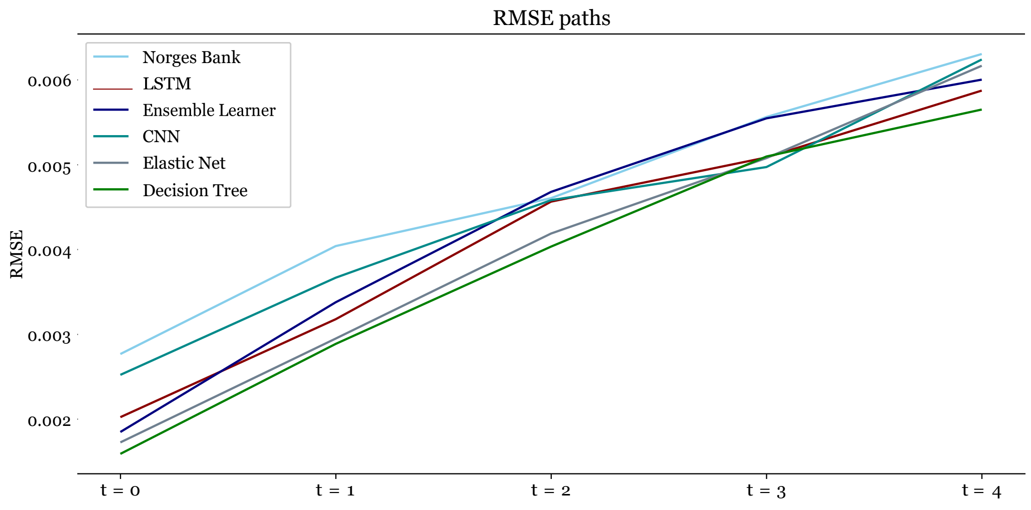


Figure 16: RMSE paths. This is a visualization of Table 4.

tween the decision tree algorithm and the relative best algorithm is minute. Secondly, the benchmark is outperformed by all algorithms across time-steps, except for the ensemble learner's three step-ahead model. Thirdly, the difference between the algorithm performances and the benchmark decreases across steps. Nevertheless, the differences are substantial, especially for nowcasting in which the decision tree algorithm predicts the policy rate with 55 percent higher accuracy than Norges Bank's model.

Figure 17 is supposed to be akin to Exhibit 4 in the appendix and is only one "window" of zero-to-four step-ahead predictions out of 26. One "window" is equal to one row in Figure 15. The plot might indicate that Norges Bank's model is better at predicting zero-to-four steps-ahead relative to the machine learning algorithms. However, Table 5 shows the average RMSE across steps. If the assertion above regarding the benchmark's overall performance were to be true, then the benchmark's average RMSE should be lower than the other algorithm's average RMSEs. In fact, we see the opposite; the decision tree algorithm has the lowest average RMSE while Norges Bank's model has the highest average RMSE, confirming the evidence presented in Table 4.

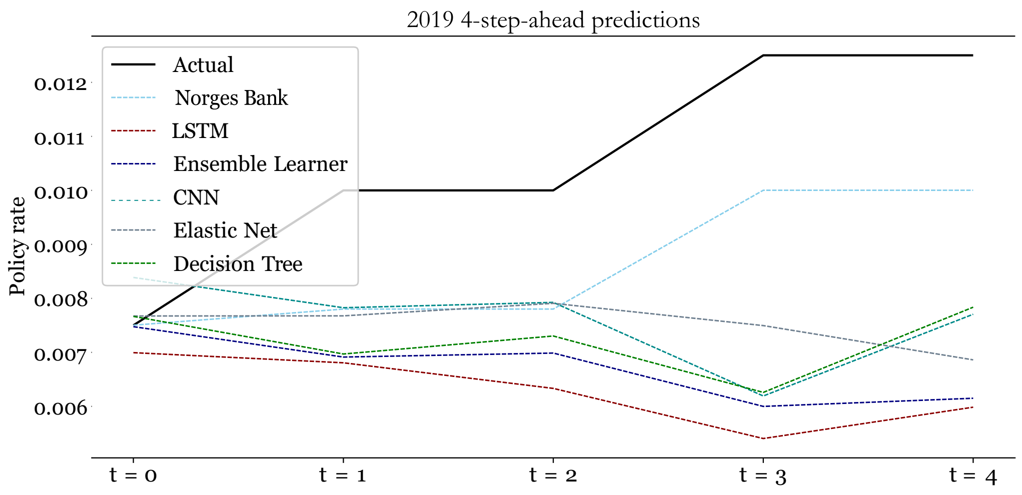


Figure 17: Zero-to-four time step predictions. This plot shows one image of the algorithms' forecast of the policy rate paths at a particular date, in this case January 22nd, 2019.

Algorithm	$\overline{RMSE}$
Decision Tree	0.003852
Elastic net	0.004021
LSTM	0.004144
Ensemble learner	0.004292
CNN	0.004396
Norges Bank	0.004657

Table 5: Averaged RMSE across time steps. It is ranked from best to worst.

The mean of the prediction errors might not portray the full story. It sheds light on the aggregated performance across time, but it does not provide a detailed description of the forecasting ability for each policy rate decision. Thus, a more thorough analysis of each observation is interesting. The following Figure 18 shows the number of times each model "won" a prediction. "Winning" a prediction implies that a model predicted a value closest to the actual value relative to other models. This computation is done across all steps.

Norges Bank's model wins most of the observations across all time-steps. This indicates that Norges Bank's model is more consistent, but its total RMSE suffers because of large outliers. Total won predictions does not mirror Table 5. Although the elastic net- and the ensemble learner algorithm exhibit lower aggregated RSME scores, the neural networks LSTM and CNN win substantially more observations. This indicates that the neural networks behave similarly to Norges Bank's model, but to a lesser extent

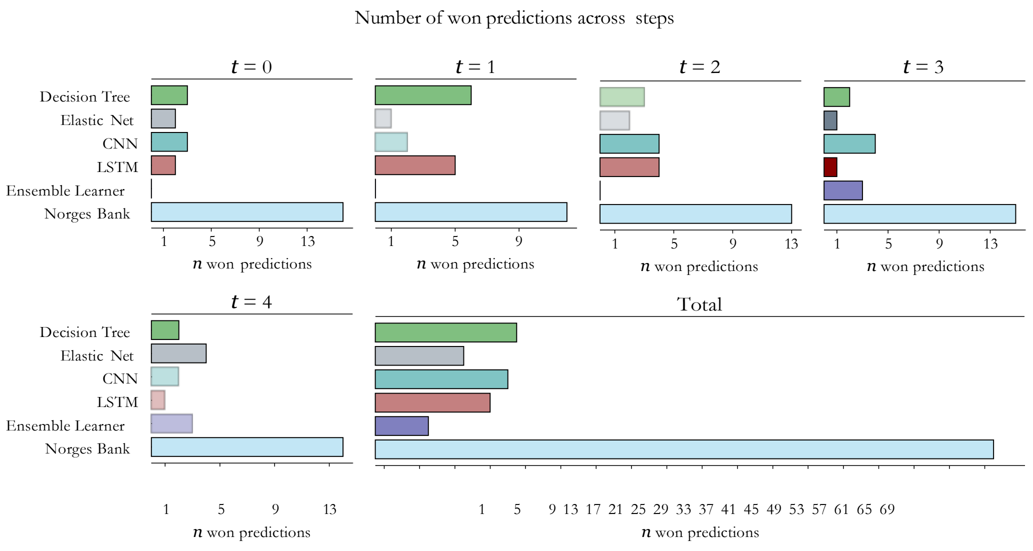


Figure 18: Number of won predictions across time steps. "Winning" a prediction implies that a model predicted a value closest to the actual value relative to other models.

5.3 One step-ahead predictions

We assess the one step-ahead predictions as particularly important, because it predicts far enough into the future such that adjustments can be made, but not too far as immediate adjustments have little practical effect. Figure 19's left plot shows the one step-ahead predictions for each algorithm, while the right plot shows the logarithmic cumulative squared errors. From these plots, we see that Norges Bank's model performs well during periods with constant policy rates but suffers when the policy rate changes substantially. For instance, as Norges Bank changed the policy rate from 50 basis points to 75 basis points in 2018, the cumulative errors sharply increased. The same happens when Norges Bank changed the policy rate from 1.5 percent to 25 basis points in the beginning of the covid-19 pandemic.

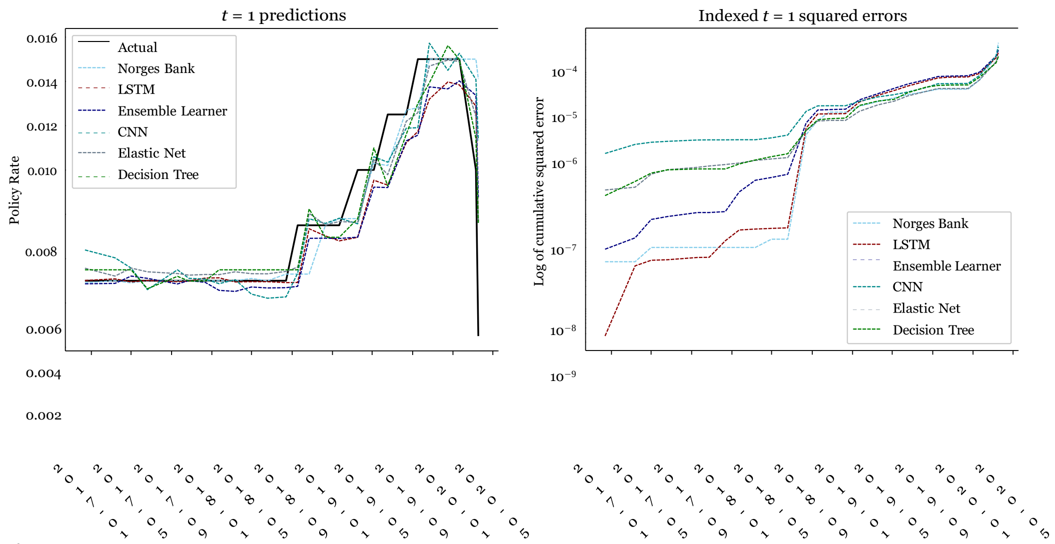


Figure 19: $t = 1$ forecasts and cumulative squared errors for the out-of-sample period. The squared errors are displayed on a logarithmic scale to better visualize the jumps in residuals. The index starts at 1, because the logarithm of a number less than 1 would result in an error. The out-of-sample period corresponds to dates between December 13th, 2016 to March 18th, 2020.

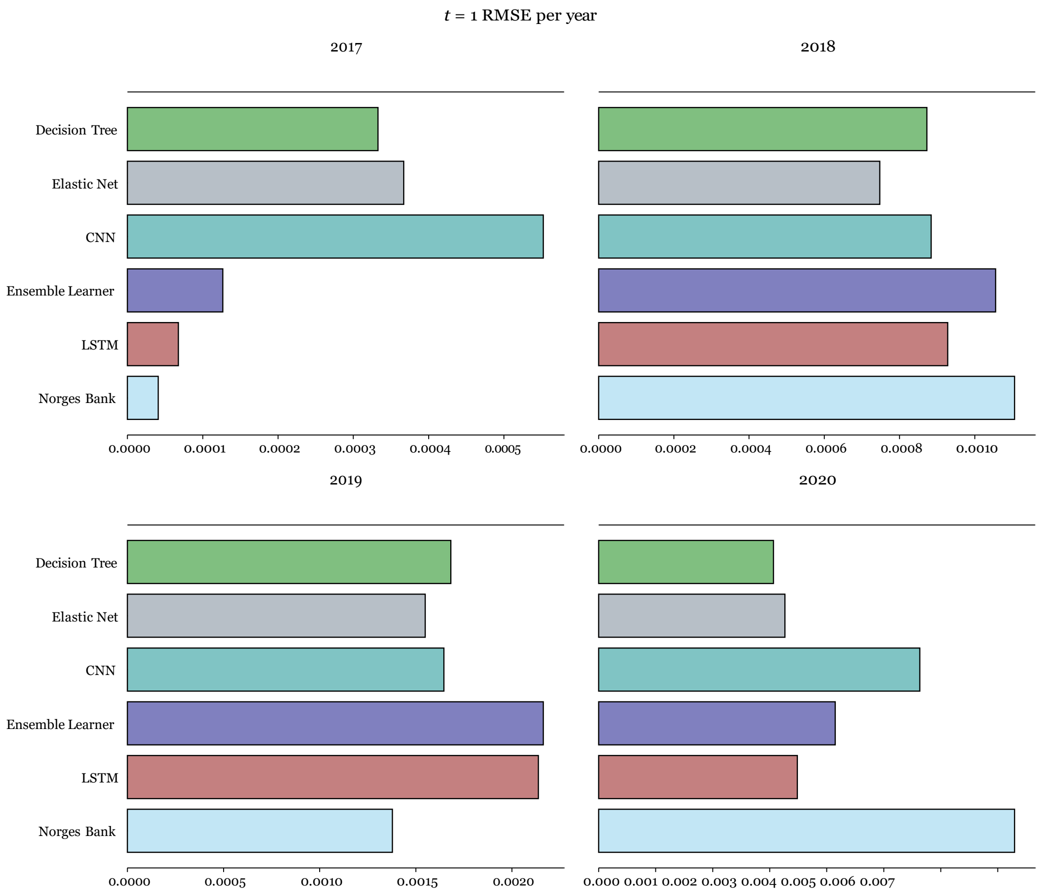


Figure 20: Per year RMSE of $t = 1$ predictions.

Figure 20 shows one step-ahead RMSE per year from 2017 to 2020. It displays changes across two-dimensions: (i) relative RMSE changes per year and (ii) idiosyncratic RMSE changes per year. The RMSE moves similarly for most of the algorithms. However, Norges Bank's model outperforms the other algorithms in both 2017 and 2019. 2017 was characterized by a constant policy rate and 2019 consisted of evenly spaced out policy rate increases of 25 basis points. On the other hand, Norges Bank's RMSE is substantially larger than the other algorithms in 2020. The ensemble learner's relative RMSE varies across the years. Most notably, the algorithm outperforms all other algorithms in 2020. We also note that the two best-performing algorithms, the decision tree- and the elastic net algorithm, exhibit the lowest RMSEs in 2020.

5.4 Uncertainty in RMSE measure

By studying the confidence interval of the RMSEs, we can analyze the uncertainty of the measures. Table 6 shows the 95 percent confidence intervals of the algorithms' one step-ahead root-squared errors. A 95 percent confidence interval means that with many repeated samples, the true RMSE will be between the limits of the confidence interval 95 percent of the samples (Brooks, 2019). For instance, the RMSE of the ensemble learner's one step-ahead model will lie between 8.52 and 44.11 basis points in 95 percent of the cases in a large sample.

	Norges Bank	**Ensemble learner**	**LSTM**	**CNN**	**Elastic net**	**Decision tree**
Mean	0.004041	0.003177	0.003379	0.003667	0.002949	0.002887
Lower CI	NaN	0.000852	0.000840	NaN	NaN	0.000842
Upper CI	0.005980	0.004411	0.004705	0.005263	0.004182	0.003995
n	26	26	26	26	26	26

Table 6: RMSE confidence intervals. Confidence interval is referred as CI. These are computed by squaring the errors and computing the confidence interval for these vectors. Next, the confidence intervals and the mean are transformed to values equivalent to the RMSEs by taking the root of the values. Some algorithms posses negative lower CI's of squared errors, which is impossible. These are therefore NaN.

The confidence intervals seem to be relatively equal across algorithms. Figure 21 shows boxplots of the algorithm's one step-ahead squared errors. Boxplots are compact and robust figures that are particularly useful for comparing distributions across groups (Stryjewski & Wickham, 2010). The median is given by the horizontal blue line, the first and fourth quartiles are given by the upper and lower edge of the box, and the upper and lower extremes are given by the upper and lower end of the vertical line. The extreme values are the minimum and maximum values in the distribution excluding outliers. Finally, the blue diamonds are the means of the distribution.

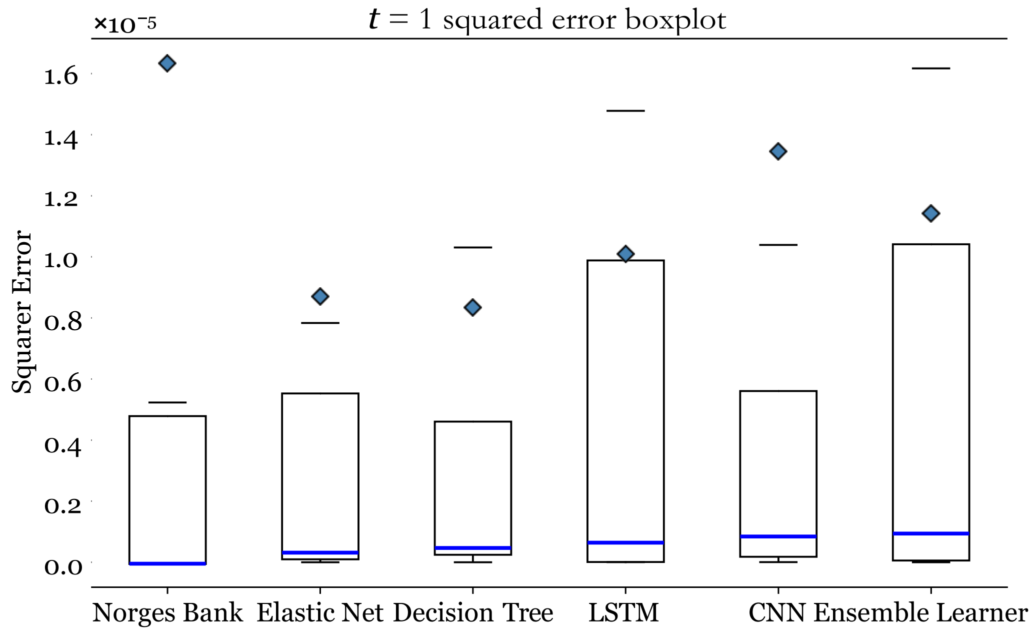


Figure 21: Boxplot of squared errors across algorithms. The median is given by the horizontal blue line, the first and fourth quartiles are given by the upper and lower edge of the box, and the upper and lower extremes are given by the upper and lower end of the vertical line. The extreme values are the minimum and maximum values in the distribution excluding outliers. Finally, the blue diamonds are the means of the distribution.

The dispersion between the benchmark's median- and average squared errors is particularly interesting. All the algorithms exhibit a higher mean- than median squared errors, which indicates positive skewness of the distributions. However, Norges Bank's model exhibits the widest difference between the two. Both the mean and the median can significantly differ if the data is skewed, for instance if it contains extreme outliers. Unlike the median, the mean is sensitive towards outliers, which means that a distribution with data points substantially above the median will have a mean exceeding the median and be positively skewed. This indicates two aspects of the distributions above. Firstly, the predictions of all the algorithms consist of small errors with infrequent larger errors. Secondly, this behavior is significantly more present in the benchmark. Additionally, as the errors are squared, outliers are even more pronounced.

The box plot's x-axis is ranked by the median of the distributions. This means that if we were to rank the algorithms based on the median of the errors, we would say that Norges Bank's model is the best for one step-ahead predictions, while the decision tree algorithm only places number three. This further confirms our assumption about Norges Bank's consistent predictions outside of large changes in the policy rate. Furthermore, a decision-maker with a different loss-function, such as MAE, would perhaps produce different results.

5.5 Robustness test

The policy rate is used as a tool to lessen the effect of financial crises on the economy. Therefore, it is interesting to study how the algorithms perform during crises. Figure 22 shows the algorithms' predictions prior to the Covid-19 pandemic and at the moment where governments began lowering their interest rates.

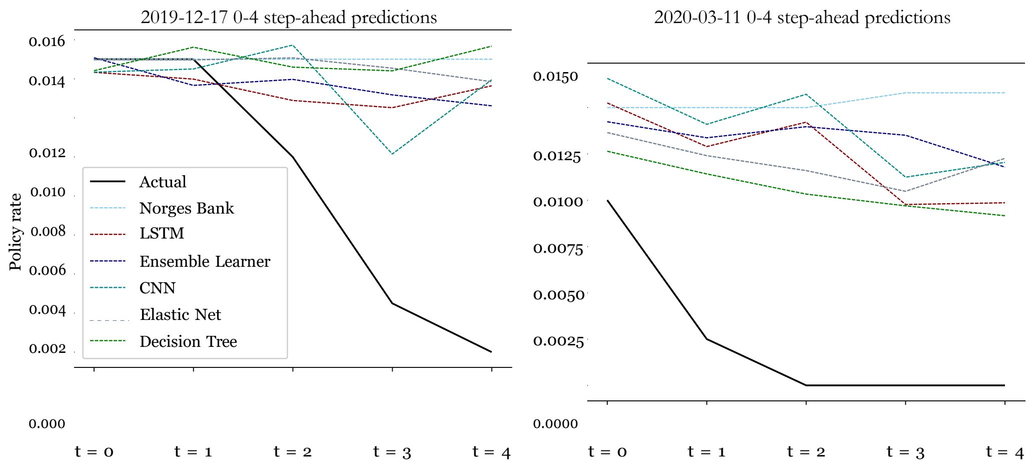


Figure 22: zero-to-four step-ahead predictions before and at the beginning of the covid-19 pandemic. $t = 0$ for plot 1 corresponds to December 17th, 2019 and $t = 0$ for plot 2 corresponds to March 11th, 2020.

From the left-hand plot, we see that none of the algorithms are particularly good at predicting the crisis. Most of them predict a stable rate at approximately 1.5 percent. When examining the second plot, we observe that the decision tree algorithm performs the best. However, none of the algorithms, including the decision tree algorithm, managed to predict the sharp decrease in policy rate. The decision tree algorithm seems to be able to predict a downward curve, but not to the extent of the actual policy rate.

Table 7 shows descriptive statistics of the one step-ahead errors of the algorithms pre-2020 and in-2020. All the algorithms suffer from the pandemic, but the increase in RMSE of Norges Bank's model is staggering. The pandemic could therefore explain some of its poor performance. Note, however, the small number of observations in 2020 ($n = 3$).

Pre covid-19	**Max**	**Min**	**Median**	**MAE**	**RMSE**
Norges Bank	0.002300	0.000000	0.000100	0.000883	0.001391
LSTM	0.003765	0.000006	0.000147	0.001243	0.001876
Ensemble Learner	0.003685	0.000009	0.000576	0.001365	0.001931
CNN	0.003223	0.000056	0.000727	0.001196	0.001598
Elastic Net	0.002799	0.000024	0.000410	0.001021	0.001412
Decision Tree	0.003388	0.000005	0.000533	0.001145	0.001536
During covid-19	**Max**	**Min**	**Median**	**MAE**	**RMSE**
Norges Bank	0.014100	0.005000	0.012500	0.010533	0.011256
LSTM	0.010386	0.003845	0.007670	0.007300	0.007778
Ensemble Learner	0.010869	0.004021	0.008767	0.007886	0.008390
CNN	0.011593	0.005268	0.011348	0.009403	0.009848
Elastic Net	0.009905	0.004904	0.007618	0.007476	0.007750
Decision Tree	0.008922	0.004986	0.007615	0.007175	0.007359

Table 7: Descriptive statistics of pandemic one step-ahead RMSE. Top is outside pandemic ($n = 23$) statistics while the bottom table is the inside pandemic ($n = 3$) performance. Max, min, and median measures are absolute errors.

Figure 23 shows prediction deviations across algorithms. These are one-to-four step-ahead predictions, which means that the predictions are made on December 17th, 2019. Notably, Norges Bank's model seizes to outperform the other algorithms as the pandemic starts to solidify. Furthermore, the ensemble learner and the LSTM algorithm have the lowest absolute errors across the five dates.

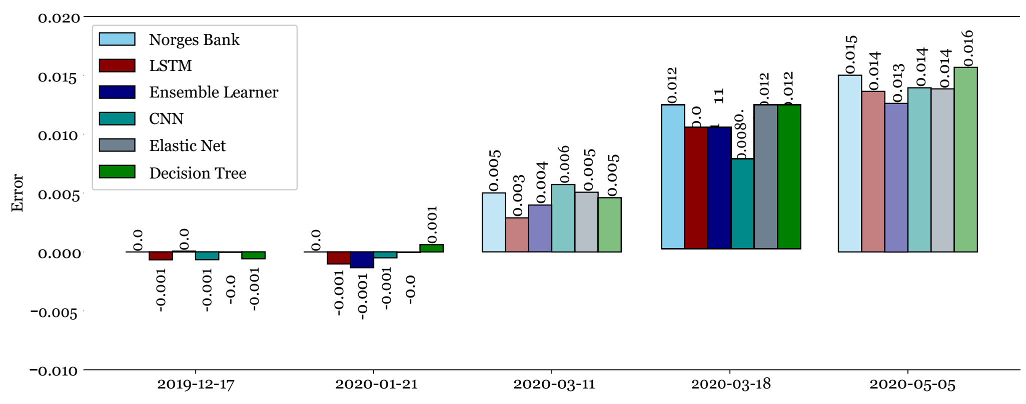


Figure 23: Pandemic errors. Each group of bar charts corresponds to the squared distance between the algorithms' predictions and the actual values for a specific decision.

6 Discussion

All the results presented so far have pointed toward a common denominator; the machine learning algorithms outperform Norges Bank's model due to better adaptability to shocks in the dependent variable. Norges Bank's model has the lowest RMSE when we exclude 2020 and its high RMSE is mostly caused by a few abnormal policy rate decisions. Beyond this, the results can be divided into three parts: (i) the decision tree algorithm and its dominant performance across time-steps, (ii) the elastic net and ensemble learner algorithms' short-term forecasting accuracy, and (iii) LSTM and CNN's long-term predictive precision.

6.1 Relative algorithm performance

6.1.1 The decision tree algorithm

Section 5.5 Robustness test presents how the different algorithms adapted to the covid-19 pandemic. This is the only financial crisis present in our test dataset. The training dataset, however, consists of two financial downturns: (i) the great recession in 07-08, and (ii) the drop in oil prices in 2015. The covid-19 pandemic destabilized the global economy due to the worrying economic outlook. In late February 2020, major stock markets fell as the number of covid-19 cases rose in countries such as Italy and South Korea. As the pandemic solidified, governments around the world sought monetary decision-making to alleviate economic contractions. For instance, the EU initiated a historic 1.8 trillion dollar stimulus package called NextGenerationEU (European Comission, 2021). Numerous countries lowered their policy rates to historic lows, including Norway. In other words, the global economic situation during 2020 was rarely reflected in the dataset prior to this period.

The decision tree algorithm performed the best on an aggregated level (Table 5). The algorithm consistently outperformed the other algorithms and exceeded the benchmark by approximately 55 percent for nowcasting predictions. Additionally, the decision tree's RMSE distribution is the least positively skewed, which indicates less outliers relative to the other algorithms. These findings, in addition to the algorithm's pandemic performance, might suggest that the decision tree algorithm exhibits greater adaptability to drastic changes in the economy. Kosina and Gama, 2012 argue for such traits in decision tree algorithms. They showed that decision rules models, which are similar to decision trees, exhibit high adaptive abilities, which could be due to its robustness to outliers and extraneous features. This robustness to superfluous information comes from pruning the branches of the tree (John, 1995), as described in Section 3: Methodology.

The fact that the decision tree algorithm performs well might also indicate that the relationship for which we are trying to model is nonlinear. Decision trees can be classified as a piecewise regression method, where each branch is its own regression. This means that the decision tree can opt to utilize different variable dependencies based on how the data looks at a certain point in time. Transferring this rationale to economic reasoning, this might suggest that the decision tree algorithm manages to identify when the economy is in a stable condition and when it is not. Döpke et al., 2017 showed that boosted regression trees performed well in predicting financial recessions, supporting the assertion above. This paper also showed that short-term interest rates, yield spreads, and stock market metrics are important indicators, which subsequently corresponds to the variable selection for our decision tree models[7].

In conclusion, our findings and prior research indicates that the decision tree's ability to quickly adapt to abrupt changes translates to a model well-equipped to predict the policy rate during volatile economic cycles. As a result, this algorithm outperforms every other algorithm, partly because our out-of-sample data consists of a volatile period in the global economy.

6.1.2 Elastic net and ensemble learner

The elastic net and the ensemble learner algorithms performed well up to three steps. For shorter predictions, the algorithms performed better relative to other algorithms, excluding the decision tree algorithm. Why these two algorithms performed well for this time interval is most likely different between the algorithms, but the similarities in RMSE paths might be due to the meta-model of the ensemble learner being an elastic net algorithm.

[7]See Exhibit 5 for decision tree models' trees.

As previously mentioned, the elastic net regularization can perform variable selection while simultaneously grouping correlated variables. Thus, its edge comes from efficiently trading variance for bias in a systematic manner, increasing its out-of-sample predictive power. Furthermore, the evidence mentioned prior might indicate that this regularization benefit ceases to exist as the time step increases. This assertion is further strengthened by how the intercept dominates the feature space of the elastic net algorithm as the prediction distance increases[8].

Unfortunately, we cannot read directly from Exhibit 6 and stipulate the importance of a variable or the magnitude of a specific relationship. This is due to the elastic net regularization and how it shrinks and excludes certain variables. However, we can get a sense of the overall relationship between the whole dataset and the output variable by examining several of the most weighted variables and their relative position, e.g., how the elastic net algorithm favors a higher intercept for longer predictions. The plot suggests that the most important variables for the elastic net algorithm are mostly international macroeconomic indicators and interest rates related to western countries. This is consistent with our assertion that the Norwegian economy is small and open.

We can do a similar analysis for the ensemble learner algorithm by looking at Exhibit 7 in the appendix. We see that the algorithm mostly depends on decision trees. This corresponds to our findings related to the decision tree algorithm's performance, which did particularly well nowcasting-wise.

The regularization paths for both algorithms can be seen in Figure 24. This figure illustrates how the models and the two algorithms differ in regularization. If $0 < \alpha < 1$, the model utilizes both regularization techniques. An α closer to zero indicates that the regularization is weighted more towards the ridge penalty, which is responsible for proportionally reducing correlated coefficients, while an α closer to one indicates that the regularization is weighted more towards the lasso penalty, which is responsible for variable selection. For instance, the elastic net algorithm employs an $\alpha = 0.9$ for its two step-ahead model, favoring the lasso regularization. Subsequently, the lambda path indicates the severity of the penalties. A λ equal to zero is equivalent to an

[8]See Exhibit 6 for elastic net coefficients.

ordinary least squares regression, while a high λ indicates severe penalty activation. For instance, the ensemble learner's two step-ahead model utilizes the highest lambda in the grid search, $\lambda = 100$, indicating that the regularization imposes severe penalties on the coefficients.

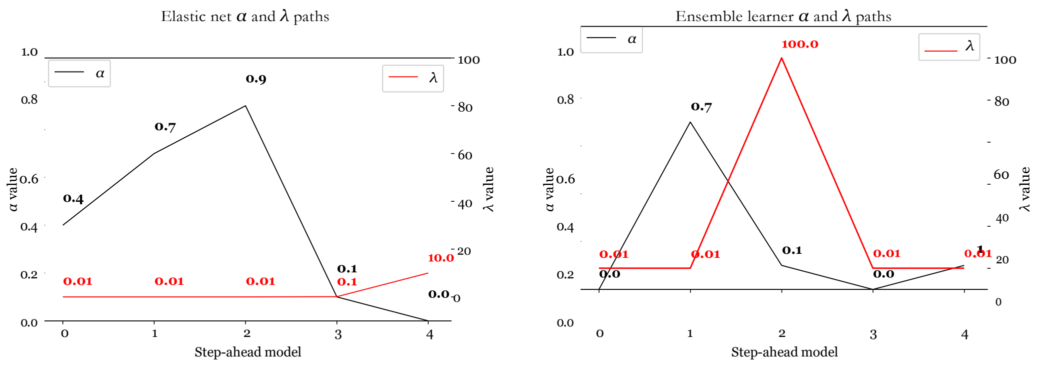


Figure 24: Elastic net and ensemble learner regularization paths. The two y-axes represents the magnitude of the elastic net algorithm parameters. Note that the ensemble learner utilizes an elastic net algorithm as meta-model. Hence, when we describe the elastic net algorithm here, it is referred to the general algorithm that is present in every model of the elastic net algorithm and the ensemble learner algorithm. The x-axis corresponds to every model related to the two algorithms.

Perhaps the clearest implication from these plots is that both algorithms utilize the regularizations to predict the policy rate. Both algorithms utilize the lasso and the ridge regularization except for longer predictions, where both algorithms favor the ridge regression.

Our dataset consists primarily of international macroeconomic data. Subsequently, the elastic net regularization is able to capitalize on the similarities between the variables. For instance, the correlation between GDPs is most likely high, in which the elastic net regularization groups these variables together and reduces their coefficients proportionally. This is also evident by the clustering of correlations visualized in the correlation plot[9].

It is interesting to note the similar results but differing model construction between the elastic net algorithm and the ensemble learner. Both algorithms discontinue to create sparse models for four step-ahead predictions, but the ensemble learner is less reliant on the intercept than the elastic net algorithm. The ensemble learner seems to be able to combine the input variables in a way to increase performance beyond just the intercept. It is therefore interesting to explore why this difference might occur. The only difference between the algorithms is in the dataset they are fitted to. The meta-model of the

[9]See Exhibit 2 for correlation matrix.

ensemble learner is fitted to the outputs from the different models that use the same dataset as the elastic net algorithm. This difference might explain the elastic net algorithm's reliance on the intercept, as the ensemble learner condenses information from the dataset from the beginning. The reason why the algorithms produce similar results might lie in the elastic net algorithm's inability to capture long-term trends. As opposed to other machine learning algorithms, the elastic net algorithm does not have an innate ability to map temporal relationships.

In conclusion, we know that both algorithms exhibit significant increases in RMSE for three- and four step-ahead predictions. For the elastic net algorithm, this seems to be due to its inability to map meaningful relationships between the input variables, because the algorithm mostly relies on the intercept for its predictions. For the ensemble learner on the other hand, the reduced performance seems to come from its inability to combine the different predictions from the base-models in a meaningful way, but still a combination that is more accurate than purely relying on the intercept. Nevertheless, the elastic net algorithm's ability to capture long-term trends might be the common denominator for the algorithms' poor long-term forecasting accuracy.

6.1.3 The neural networks

Both neural networks performed well relative to the benchmark (Table 4). In fact, the poorest performing algorithm, CNN, had an aggregated RMSE only 14.1 percent higher than the best performing algorithm. We have made several interesting discoveries after studying the results. First, both algorithms perform better for longer time steps. Second, the volatility of CNN's predictions is high compared to all other algorithms. Finally, we see that LSTM has a remarkably low RMSE throughout 2017 and behaves similarly to Norges Bank's model.

Both neural networks possess features that enable them to capture long-term trends. The CNN algorithm analyzes several observations per iteration, in contrast to the other algorithms who only process one observation at a time. This allows the algorithm to observe the feature space temporally and capture time-dependent relationships. LSTM, on the other hand, only processes one observation at a time, but sends information from previous iterations forward in time. We argue that CNN's- and LSTM's temporal qualities are reflected in the results. In fact, CNN has the best performing model for three step-ahead predictions and LSTM has an RMSE only 3.9 percent higher than the decision tree for four step-ahead predictions. Therefore, we argue that they are, to a certain extent, able to identify structural changes in the dataset which consequently allows them to better predict the long-term future.

The volatility of CNN's predictions is high (Figure 25). Volatility in in-sample predictions is not a problem by itself, but if the volatility reduces the out-of-sample accuracy, it may indicate that the model is underfitted. Based on analyzes of the CNN algorithm's structure, it appears that the volatility is due to the model construction. Through several initial layers in the CNN models, the sample size is reduced before the information is sent to a neural network. In our models, it appears that the pooling layer and the dropout rate have been too large, which has led to the natural networks having too few data points to process. After studying the results, we tested a CNN model with a lower dropout rate and without a pooling layer. The test shows that the in-sample predictions became less volatile, while the aggregate RMSE was 6.1 percent higher. This indicates that lower volatility in the predictions did not improve the accuracy of the predictions.

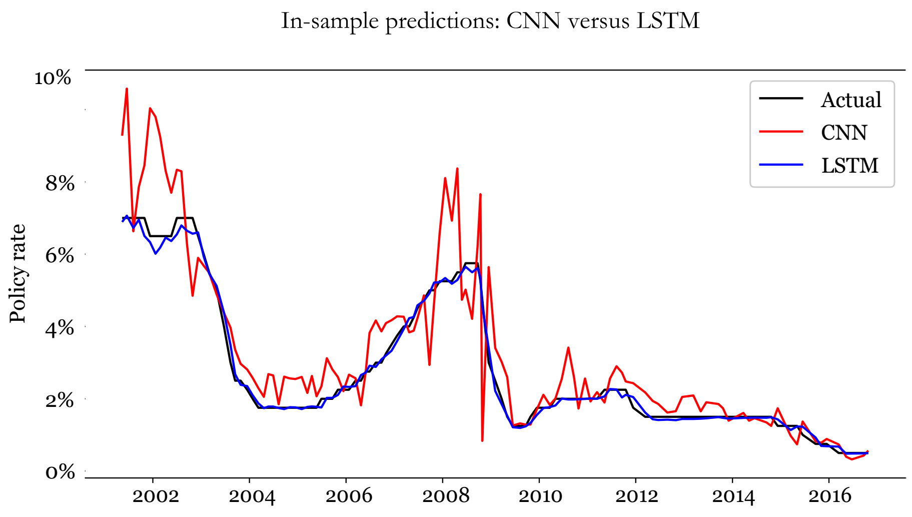


Figure 25: In-sample predictions for CNN and LSTM.

Figure 20 and the right-hand graph in Figure 19 show that the LSTM algorithm has the lowest RMSE in 2017 compared to the other algorithms, and that the cumulative RMSE behaves approximately the same as Norges Bank's model. What makes this finding interesting is that the policy rate level is flat equal to 50 basis points throughout 2017. In addition, both LSTM's- and Norges Bank's cumulative RMSE increased significantly on September 18th, 2018 because the interest rate was raised from 50 basis points to 75 basis points. This indicates that the LSTM algorithm has a significant dependence on previous observations of the policy rate and that the algorithm is less flexible than other algorithms such as the decision tree.

In conclusion, we succeeded in exploiting the temporal qualities of the LSTM- and CNN algorithm. This is reflected in their respective RMSEs for longer time steps, relative to the other algorithms. Nevertheless, it seems that the CNN algorithm was under-optimized given the volatility in in-sample predictions. Although the aggregated RMSE did not improve by reducing volatility, we believe that the algorithm can be improved by facilitating the structure of the data and spending more time on pre-testing. In addition, we see that the LSTM algorithm possesses similar characteristics to Norges Bank's model. Both models have a clear dependence on previous observations of the policy rate, and they struggle to adjust for structural changes in the dataset.

6.1.4 Norges Bank's model

Norges Bank's model emerges as the worst predictor for our forecasting period. It mostly presents the highest RMSE scores across time-steps and has the highest aggregated RMSE by approximately six percent. This is despite generally exhibiting superior performance across other analyzes. As previously mentioned, we believe that this is due to a handful of policy rate decisions and an accuracy measure that heavily penalizes large errors. Therefore, it is interesting to discuss Norges Bank's model to potentially explain some of its poor performance relative to the machine learning algorithms.

Norges Bank is not the only central bank utilizing a DSGE model for macroeconomic analysis and forecasting. DSGE models are generally favored by financial policy makers in part because of its transparency in highlighting economic forces and how these affect each other (Christiano et al., 2018). However, the DSGE model is not without controversy. Its inability to predict crises is among its flaws (Linde, 2018), which corresponds to our findings related to Norges Bank's model's ineffective response to shocks in the economy. In Figure 26, we plotted the nowcasting forecasts and the one step-ahead forecasts of Norges Bank's model and the decision tree algorithm. These figures show how quickly the decision tree algorithm is able to adapt to a shock relative to the benchmark.

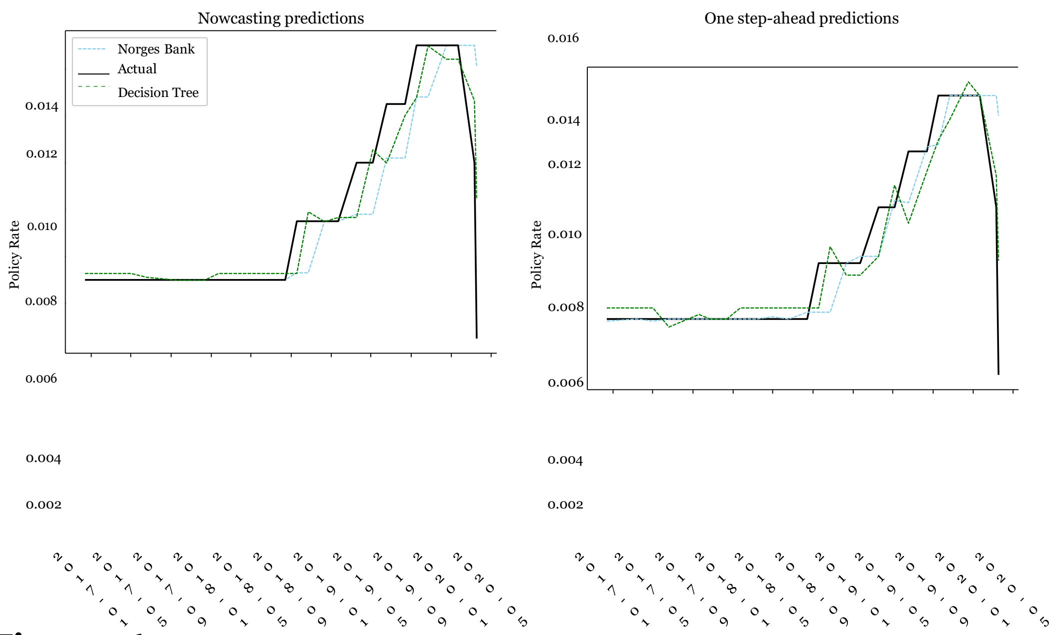


Figure 26: Nowcasting and one step-ahead predictions for Norges Bank's model and the decision tree algorithm.

This is not to say that Norges Bank's DSGE model fails at what it is trying to accomplish. The DSGE model is, on one hand, a forecasting tool. But it is just as much a tool for transparent policy making. DSGE models have for instance been useful for highlighting the benefits of fiscal stimulus during crises (Woodford, 2011). Moreover, DSGE models are not inept at forecasting. In fact, Norges Bank's model performed better than the machine learning algorithms during normal economic periods. Despite having an aggregated RMSE above the other algorithms, Norges Bank's model won the most observations during our out-of-sample period, predicting closest to the policy rate for approximately 54 percent of all observations (Figure 18). Furthermore, Norges Bank's model's RMSE is the lowest if we exclude the covid-19 pandemic (Table 7) and it outperformed the other algorithms during 2017 and 2019 (Figure 20). Lastly, it is evident from the accumulated squared errors that the model for the most part produces low residuals (Figure 19).

Summarized, we argue that Norges Bank's DSGE model is a versatile multi-faceted model used for broader fiscal policy making. But blindly following a mathematical model is usually not advised. Blanchard, 2016, former chief economist at the International Monetary Fund, argues that DSGE models should not be imperialistic, but rather inclusive to other models. Perhaps combining a DSGE model and machine learning algorithms is the next step. This macroeconomic framework could yield a tool which utilizes the transparency of the DSGE model and the flexibility of machine learning algorithms.

6.2 Improvements and further research

6.2.1 Machine learning disadvantages

Machine learning algorithms' ability to learn non-linear relationships, efficiently apply big data, and improve out-of-sample predictability is impressive. However, there are some disadvantages to machine learning relative to other statistical tools. Perhaps the biggest disadvantage is that most machine learning algorithms are so-called black-box algorithms. A black-box algorithm refers to the inability or difficulty of deciphering how the algorithm makes predictions. It is a process where the input and the outputs are known, but the internal transformation is unknown.

The deciphering problem becomes exponentially onerous as the technology improves, the complexity increases, and application of these algorithms expands (Castelvecchi, 2016). This problem can be a particular issue for applications where the interpretability of the underlying process is imperative, for instance autonomous vehicles or disease detection. The interpretability of the forces behind changes in the policy rate is most likely imperative to policy makers. One of the reasons why governments use DSGE models is that they are highly interpretable. Norges Bank is expected to be able to explain the motivation behind each decision, which makes black-box algorithms unattractive and will unlikely replace current models. They can however supplement current methods, which appear reasonable given that machine learning algorithms outperform Norges Bank's model during times of crisis.

An action is naturally followed by a reaction. The black box problem has issued increased research into explainable artificial intelligence (XAI). XAI is a paradigm shift in AI which aims to facilitate the creation of more transparent machine learning algorithms while maintaining high performance (Adadi &

Berrada, 2018). One approach is to create a set of machine learning models that collectively produce more explainable transformations (Gunning, 2017). For instance, the CNN algorithm is notoriously difficult to decipher. One could therefore use a sequential RNN model, such as an LSTM, to generate words and captions that describe the transformations that are happening. XAI might be the bridge between monetary policy decisions and machine learning where interpretability is imperative.

6.2.2 Model construction

This thesis studies machine learning's application in policy rate predictions. As a result, we focused on a broad range of machine learning algorithms that were neither chosen for perfect optimization nor optimized to perfection. On the contrary, the algorithms were chosen to capture the spectrum of machine learning and to focus on well-known algorithms. Furthermore, we chose to keep the standardization of the algorithms and not do extensive hyperparameterization. Hence, there may be custom versions of the selected algorithms, or untested algorithms, better suited for this particular application. For instance, (Vaswani et al., 2017) shows that transformers outperform CNN's and RNN's while requiring less time to train. Transformers are similar to other sequential algorithms but differ in their mechanism of attention. This means that it can reduce the impact of less important data while highlighting important ones making it attractive to users that require interpretability.

6.2.3 Features

In simplified terms, machine learning boils down to making a model do the same task repeatedly. This repeated process is an attempt at making the model used to the task and therefore better at doing that task in the future. This is why data is so important in machine learning. The model needs to get familiarized with the task by getting exposed to different examples of environments, which means that the data should in general contain plenty of observations. Furthermore, the number of features in the data should reflect the real environment of the task at hand, mapping the relevant variables that affect the dependent variable. Lastly, the environment needs to be readable to the machine, which poses numerous data formatting constraints.

Making sure the data is optimized for the proper task is generally difficult. Every task requires different data, which implies different feature structures, spaces, lengths, and dimensions. Hence, every choice made in Section 4: Data had an impact on the final result, and different choices would yield different results. For instance, we focused on gathering mostly macroeconomic data. In addition to this, we could have collected non-macroeconomic data, such as the number of airline tickets purchased per day. While the relationship between the policy rate and airline tickets seems unfounded, one could argue that the number of airline tickets purchased is a proxy for global economic stimulus. The point is that there could be an entire feature space that we have overlooked due to the obscure connection between the independent- and dependent variable.

On the other hand, reducing the feature space could also yield better results. This should not be an issue for algorithms well-equipped for feature space regularization such as the elastic net and decision tree, but for algorithms such as LSTM and CNN, the number of features can pose a problem. LSTM and CNN have some implicit feature selection functionality as LSTM suppresses irrelevant information and CNN compresses each sample. However, being inattentive to the models' data handling process may allow non-causal features to disturb the generalization of the models, making them prone to produce unoptimized predictions. Thus, reducing the feature space by being more restrictive in the variable selection process may reduce noise and yield more accurate predictions. One could, for example, make an LSTM model with only the variables used in Norges Bank's model.

The feature length of our dataset is also worth a discussion. As stated, machine learning algorithms generally require many observations to get familiarized with a task. In Section 4: Data, we excluded observations where Norges Bank did not make a policy rate decision, which left us with 127 training observations. This is, by machine learning standards, a small but sufficient amount. However, given the philosophy that "more is better", a larger dataset could have produced better results. There is an implicit limitation to the feature length given the amount of available policy rate decisions, but since the rate is constant until a change is made, one could theoretically use daily data.

There are several considerations to make when building such models, but it is nevertheless possible.

7 Conclusion

The purpose of this thesis was to explore various standardized machine learning algorithms and measure their ability to predict the Norwegian policy rate. The policy rate and its future development is an important variable to analyze because of its instrumental role in the economy. The forecasts of the policy rate can be seen as a proxy for the direction of the economy. Hence, a prediction of the policy rate is essentially a prediction of the direction of the economy. Therefore, improved research on current and new methods for forecasting policy rates are an attractive goal.

The hypothesis of this thesis is that machine learning algorithms can produce more accurate predictions than Norges Bank's own model, NEMO. This is a bold but compelling statement. To test the hypothesis, we first collected data on numerous economic variables with varying frequency and length in order to give the machine learning algorithms a wide set of features to extract patterns from. Then, we selected five well-known machine learning algorithms as candidates to compete against Norges Bank's model: (i) the elastic net algorithm, (ii) the decision tree algorithm, (iii) LSTM, (iv) CNN, and (vi) ensemble learner. These are algorithms with different properties and functions that are suitable for time series predictions.

The result of the test confirmed our hypothesis; every algorithm managed to forecast the policy rate better than Norges Bank's model. We can therefore claim that the machine learning algorithms predict the policy rate with greater accuracy than Norges Bank's model. However, this does not reflect the whole truth. A thorough analysis showed that Norges Bank made by far the most correct predictions and had the lowest RMSE when we excluded the covid-19 pandemic. Our results therefore indicate that our machine learning algorithms do not perform better than Norges Bank's model under normal circumstances. However, during crises, Norges Bank's model lacks the flexibility to adapt to sudden macroeconomic changes and the predictions become highly inaccurate. The machine learning algorithms, on the other hand, are more flexible and are able to anticipate the crisis to a greater extent.

In addition to exceeding the benchmark, the machine learning algorithms exhibited performance heterogeneity. The decision tree algorithm performed best overall. We believe that this is due to the algorithm's ability to quickly adapt to changes in economic conditions. The two neural networks CNN and LSTM performed better than the other algorithms for longer predictions, which we believe to be due to the algorithms' innate ability to remember temporal relationships. Surprisingly, the ensemble learner algorithm underperformed. A priori fitting, we believed that this algorithm would outperform the other algorithms by an effect analogous to the wisdom of the crowd.

We believe our results indicate the potential for machine learning algorithms in monetary policy decisions. However, the models described in this thesis are not optimized to the task at hand, which means that there most likely exists algorithms and models better equipped at forecasting the policy rate. Furthermore, the current state of machine learning does not elicit a replacement of current methods such as DSGE models. Machine learning might instead be a supplementary instrument to traditional policy tools.

References

Adadi, A., & Berrada, M. (2018). Peeking inside the black-box: a survey on explainable artificial intelligence (XAI). *IEEE access*, *6*, 52138–52160.

Alstadheim, R., Bache, I. W., Holmsen, A., Maih, J., & Røisland, Ø. (2010). *Monetary policy analysis in practice* (tech. rep. No. 11/2010). Norges Bank.

Bankson, C. A., & Holm, A. M. (2019). *Kunstig intelligens i makroøkonomisk prognosearbeid: En empirisk studie av hvor godt maskinlæring evner å predikere norsk økonomisk vekst* (Master's thesis).

Blanchard, O. (2016). Do DSGE models have a future? *Revista de Economı'a Institucional*, *18* (35), 39–46.

Breiman, L. et al. (2001). Statistical modeling: The two cultures (with comments and a rejoinder by the author). *Statistical science*, *16*(3), 199–231.

Brooks, C. (2019). *Introductory econometrics for finance* (4th ed.). Cambridge university press.

Brubakk, L., Husebø, T. A., Maih, J., Olsen, K., & Østnor, M. (2006). *Finding NEMO: Documentation of the Norwegian economy model* (tech. rep. No. 2006/6). Norges Bank.

Brubakk, L., & Sveen, T. (2009). *NEMO–a new macro model for forecasting and monetary policy analysis* (tech. rep.). Norges Bank.

Castelvecchi, D. (2016). Can we open the black box of AI? *Nature News*, *538* (7623), 20.

Christiano, L. J., Eichenbaum, M. S., & Trabandt, M. (2018). On DSGE models. *Journal of Economic Perspectives*, *32*(3), 113–40.

Dietterich, T. G. (2002). Ensemble Learning, The Handbook of Brain Theory and Neural Networks, MA Arbib. Cambridge, MA: MIT Press.

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

Efron, B., Hastie, T., Johnstone, I., Tibshirani, R., et al. (2004). Least angle regression. *Annals of statistics*, *32*(2), 407–499.

European Comission. (2021). The EU's 2021-2027 long-term budget & NextGenerationEU. *Publication Office of the European Union*. https://op.europa.eu/en/publication-detail/-/publication/d3e77637-a963-11eb-9585-01aa75ed71a1/language-en.

Friedman, M. (1960). *A program for monetary stability*. Fordham University Press.

Gashler, M., Giraud-Carrier, C., & Martinez, T. (2008). Decision tree ensemble: Small heterogeneous is better than large homogeneous, In *2008 seventh international conference on machine learning and applications*. IEEE.

Gogas, P., Papadimitriou, T., Matthaiou, M., & Chrysanthidou, E. (2015). Yield curve and recession forecasting in a machine learning framework. *Computational Economics*, *45* (4), 635–645.

Gunning, D. (2017). Explainable artificial intelligence (xai). *Defense Advanced Research Projects Agency (DARPA), nd Web*, *2*(2).

Hochreiter, S., & Schmidhuber, J. (1997). Long short-term memory. *Neural computation*, *9* (8), 1735–1780.

Hong, T., & Han, I. (2002). Knowledge-based data mining of news information on the internet using cognitive maps and neural networks. *Expert systems with applications*, *23* (1), 1–8.

Hyndman, R. J., & Athanasopoulos, G. (2018). *Forecasting: Principles and practice*. OTexts.

Hyndman, R. J., & Koehler, A. B. (2006). Another look at measures of forecast accuracy. *International journal of forecasting*, *22*(4), 679–688.

Jacovides, A. (2008). *Forecasting Interest Rates from the Term Structure: Support Vector Machines Vs Neural Networks* (Doctoral dissertation). University of Nottingham.

John, G. H. (1995). Robust Decision Trees: Removing Outliers from Databases, In *Kdd*.

Ketkar, N., & Santana, E. (2017). *Deep learning with python* (Vol. 1). Springer.

Kim, H. H., & Swanson, N. R. (2014). Forecasting financial and macroeconomic variables using data reduction methods: New empirical evidence. *Journal of Econometrics*, *178*, 352–367.

Kosina, P., & Gama, J. (2012). Handling time changing data with adaptive very fast decision rules, In *Joint european conference on machine learning and knowledge discovery in databases*. Springer.

Kravik, E. M., & Paulsen, K. (2017). *A complete documentation of Norges Bank's policy model NEMO* (tech. rep.). Norges Bank, Technical Report.

Kuncheva, L. I., & Whitaker, C. J. (2003). Measures of diversity in classifier ensembles and their relationship with the ensemble accuracy. *Machine learning*, *51* (2), 181–207.

Kuzey, C., Uyar, A., & Delen, D. (2014). The impact of multinationality on firm value: A comparative analysis of machine learning techniques. *Decision Support Systems*, *59*, 127–142.

Lanbouri, Z., & Achchab, S. (2019). A new approach for Trading based on Long-Short Term memory Ensemble technique. *International Journal of Computer Science Issues (IJCSI)*, *16* (3), 27–31.

Linde, J. (2018). DSGE models: still useful in policy analysis? *Oxford Review of Economic Policy*, *34* (1-2), 269–286.

Lønning, I., & Olsen, K. (2000). Pengepolitiske regler. *Penger og kreditt 2/2000*.

Lovdata. (2019). Bestemmelse om pengepolitikken (FOR-2019-12-13-1775). https://lovdata.no/dokument/LTI/forskrift/2019-12-13-1775.

Norges Bank. (n.d.). The policy rate. *norges-bank.no*. https://www.norges-bank.no/en/topics/Monetary-policy/Policy-rate/.

Norges Bank. (2015). Further information on the policy rate. *norges-bank.no*. https:// www. norges- bank. no/ en/ topics/ Monetary- policy/ Policy-rate/Policy-rate---more/.

Norges Bank. (2020a). Inflation. *norges-bank.no*. https://www.norges- bank.no/en/topics/Monetary-policy/Inflation/.

Norges Bank. (2020b). Monetary policy rate 01/20. *norges-bank.no*. https : //www.norges-bank.no/en/news-events/news-publications/Reports/Monetary- Policy- Report- with- financial- stability- assessment/ 2021 / mpr-12021/.

Norges Bank. (2021). Monetary Policy Report with financial stability assessment 1/2021. *norges-bank.no*. https://www.norges-bank.no/en/news-events/news-publications/Reports/Monetary-Policy-Report-with-financial-stability-assessment/2021/mpr-12021/.

Oh, K. J., & Han, I. (2000). Using change-point detection to support artificial neural networks for interest rates forecasting. *Expert systems with applications*, *19* (2), 105–115.

Olsen, Ø. (2011). Use of models and economic theory in Norges Bank [[Lecture at the Department of Economics, University of Oslo]]. *Norges Bank archive*. https://www.norges-bank.no/en/news-events/news-publications/Speeches/2011/08092011-Schweigaard-lecture/.

Qiu, J., Wang, B., & Zhou, C. (2020). Forecasting stock prices with long-short term memory neural network based on attention mechanism. *PloS one*, *15* (1), e0227222.

Sezer, O. B., & Ozbayoglu, A. M. (2018). Algorithmic financial trading with deep convolutional neural networks: Time series to image conversion approach. *Applied Soft Computing*, *70*, 525–538.

Sirignano, J., & Cont, R. (2019). Universal features of price formation in financial markets: Perspectives from deep learning. *Quantitative Finance*, *19* (9), 1449–1459.

Smalter Hall, A., & Cook, T. R. (2017). Macroeconomic indicator forecasting with deep neural networks. *Federal Reserve Bank of Kansas City Working Paper*, (17-11).

Stryjewski, L., & Wickham, H. (2010). 40 years of boxplots. https://vita.had.co.nz/papers/boxplots.pdf.

Taylor, J. B. (1993). Discretion versus policy rules in practice [North-Holland], In *Carnegie-rochester conference series on public policy*. North-Holland. Elsevier.

Tibshirani, R. (1996). Regression shrinkage and selection via the lasso. *Journal of the Royal Statistical Society: Series B (Methodological)*, *58*(1), 267–288.

Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, L., & Polosukhin, I. (2017). Attention is all you need. *arXiv preprint arXiv:1706.03762*.

Vitek, F. (2017). *Policy, risk and spillover analysis in the world economy: A panel dynamic stochastic general equilibrium approach.* International Monetary Fund.

West, D., Dellana, S., & Qian, J. (2005). Neural network ensemble strategies for financial decision applications. *Computers & operations research*, *32* (10), 2543–2559.

Woodford, M. (2011). Simple analytics of the government expenditure multiplier. *American Economic Journal: Macroeconomics*, *3*(1), 1–35.

Yasir, M., Afzal, S., Latif, K., Chaudhary, G. M., Malik, N. Y., Shahzad, F., & Song, O.-y. (2020). An efficient deep learning based model to predict interest rate using twitter sentiment. *Sustainability*, *12*(4), 1660.

Zhang, G., & Hu, M. Y. (1998). Neural network forecasting of the British pound/US dollar exchange rate. *Omega*, *26* (4), 495–506.

Zhou, L., Pan, S., Wang, J., & Vasilakos, A. V. (2017). Machine learning on big data: Opportunities and challenges. *Neurocomputing*, *237*, 350–361.

Zimmermann, H.-G., Tietz, C., & Grothmann, R. (2002). Yield curve forecasting by error correction neural networks and partial learning., In *Esann.* Citeseer.

Zou, H., & Hastie, T. (2005). Regularization and variable selection via the elastic net. *Journal of the royal statistical society: series B (statistical methodology)*, *67* (2), 301–320.

Zou, Z., & Qu, Z. (2020). Using LSTM in Stock prediction and Quantitative Trading.

8 Appendices

Exhibit 1: Overview of collected base variables. Lag refers to the publication lag of the variable, which is displayed in number of days.

Variable	Basis	Type	Source	Frequency	Lag
BoE base rate	Return	Interest Rate	Bank of England	Daily	0
EuroStoxx50	Return	Financial	Bloomberg	Daily	0
OBX, 25 Most Liquid Stocks	Return	Financial	Bloomberg	Daily	0
OSE25GI, Consumer Discretionary	Return	Financial	Bloomberg	Daily	0
OSE3030GI, Household & Personal Products	Return	Financial	Bloomberg	Daily	0
OSE10GI, Energy	Return	Financial	Bloomberg	Daily	0
OSE40GI, Finance	Return	Financial	Bloomberg	Daily	0
OSE35GI, Health Care	Return	Financial	Bloomberg	Daily	0
OSE20GI, Industry	Return	Financial	Bloomberg	Daily	0
OSE45GI, IT	Return	Financial	Bloomberg	Daily	0
OSE50GI, Telecommunication Services	Return	Financial	Bloomberg	Daily	0
OSE55GI, Utilities	Return	Financial	Bloomberg	Daily	0
OMX	Return	Financial	Bloomberg	Daily	0
DAX	Return	Financial	Bloomberg	Daily	0
FTSE	Return	Financial	Bloomberg	Daily	0
S&P500 High	Return	Financial	Bloomberg	Daily	0
S&P500 Low	Return	Financial	Bloomberg	Daily	0
S&P500 Adjusted Close	Return	Financial	Bloomberg	Daily	0
S&P500 Volume	Absolute	Financial	Bloomberg	Daily	0
Index of Industrial Production (Manufacturing, Absolute)	Absolute	Industry	Bloomberg	Monthly	45
Index of Industrial Production (Total)	Return	Industry	Bloomberg	Monthly	30
Index of Industrial Production (Manufacturing, Return)	Return	Industry	Bloomberg	Monthly	30

Exhibit 1: Overview of collected base variables. Lag refers to the publication lag of the variable, which is displayed in number of days.

Variable	Basis	Type	Source	Frequency	Lag
Housing Prices	Absolute	CPI	Bloomberg	Quarterly	12
China Real GDP	Return	National	Bloomberg	Quarterly	18
Aluminium Spot Price	Return	Commodity	Bloomberg	Daily	0
Aluminium Futures Price	Return	Commodity	Bloomberg	Daily	0
EU Economic Sentiment	Absolute	Survey	Bloomberg	Monthly	21
ISM Manufacturing Index	Absolute	Survey	Bloomberg	Monthly	5
US Conference Board Leading Index	Return	Survey	Bloomberg	Monthly	21
US Conference Board Leading Index	Absolute	Survey	Bloomberg	Monthly	21
Brent Crude Oil Spot Price	Absolute	Commodity	Bloomberg	Daily	0
Volatilty Index (VIX)	Return	Financial	Bloomberg	Daily	0
Norwegian 10-year Treasury Yield	Return	Interest Rate	Bloomberg	Daily	0
Fish Price Index	Return	Commodity	Bloomberg	Weekly	7
EU Quantitative Easing	Absolute	Financial	ECB	Yearly	0
US Quantitative Easing	Absolute	Financial	FED	Quarterly	0
Effective Federal Funds Rate	Return	Interest Rate	Federal Reserves	Daily	0
Norwegian yield-spread (10Y-2Y)	Return	Interest Rate	Federal Reserves	Daily	0
Norwegian Consumer Trust	Absolute	Survey	Finans Norge	Quarterly	-15
Norwegian Unemployment Rate	Return	CPI	SSB	Monthly	0
Norwegian CPI (Absolute)	Absolute	CPI	SSB	Monthly	10
Norwegian CPI (Return)	Return	CPI	SSB	Monthly	10

Exhibit 1: Overview of collected base variables. Lag refers to the publication lag of the variable, which is displayed in number of days.

Variable	**Basis**	**Type**	**Source**	**Frequency**	**Lag**
Norwegian CPI Year-to-Year Change	Return	CPI	SSB	Monthly	10
Norwegian CPI-AT	Return	CPI	SSB	Monthly	10
Norwegian CPI-ATE	Return	CPI	SSB	Monthly	10
Norwegian CPI-AE	Return	CPI	SSB	Monthly	10
Norwegian CPI-AEL	Return	CPI	SSB	Monthly	10
Denmark Real GDP	Return	National	OECD	Quarterly	30
France Real GDP	Return	National	OECD	Quarterly	30
Germany Real GDP	Return	National	OECD	Quarterly	30
Netherlands Real GDP	Return	National	OECD	Quarterly	30
Sweden Real GDP	Return	National	OECD	Quarterly	30
US Real GDP	Return	National	OECD	Quarterly	30
UK Real GDP	Return	National	OECD	Quarterly	30
EU Real GDP	Return	National	OECD	Quarterly	30
Denmark Real GDP Year-to-Year Change	Return	National	OECD	Quarterly	30
France Real GDP Year-to-Year Change	Return	National	OECD	Quarterly	30
Germany Real GDP Year-to-Year Change	Return	National	OECD	Quarterly	30
Netherlands Real GDP Year-to-Year Change	Return	National	OECD	Quarterly	30
Sweden Real GDP Year-to-Year Change	Return	National	OECD	Quarterly	30
US Real GDP Year-to-Year Change	Return	National	OECD	Quarterly	30
UK Real GDP Year-to-Year Change	Return	National	OECD	Quarterly	30
EU Real GDP Year-to-Year Change	Return	National	OECD	Quarterly	30
Norwegian Real GDP Year-to-Year-Change	Return	National	OECD	Quarterly	30
Sweden CPI	Return	CPI	IMF	Monthly	10
Germany CPI	Return	CPI	IMF	Monthly	10
UK CPI	Return	CPI	IMF	Monthly	10

US CPI	Return	CPI	IMF	Monthly	10

Exhibit 1: Overview of collected base variables. Lag refers to the publication lag of the variable, which is displayed in number of days.

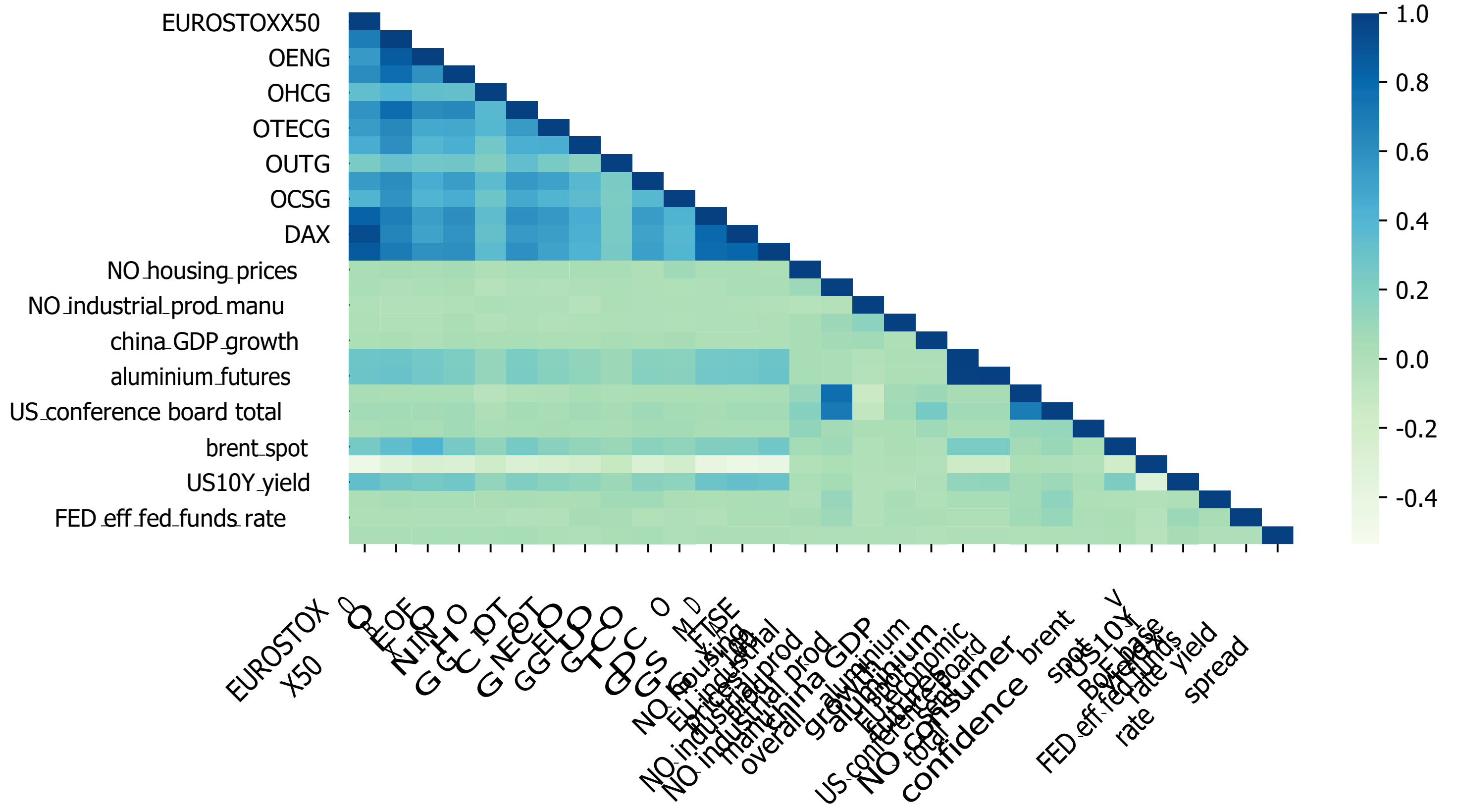
Variable	Basis	Type	Source	Frequency	Lag
Hong Kong CPI	Return	CPI	IMF	Monthly	10
China CPI	Return	CPI	IMF	Monthly	10
France CPI	Return	CPI	IMF	Monthly	10
Denmark CPI	Return	CPI	IMF	Monthly	10
Netherlands CPI	Return	CPI	IMF	Monthly	10
Norwegian Policy Rate	Return	Interest Rate	Norges Bank	Daily	0
ECB Folio Rate	Return	Interest Rate	ECB	Monthly	0
Hong Kong Folio Rate	Return	Interest Rate	IMF	Monthly	0
Sweden Folio Rate	Return	Interest Rate	IMF	Monthly	0
Denmark Folio Rate	Return	Interest Rate	IMF	Monthly	0
OSEBX Open	Return	Financial	Oslo Børs	Daily	0
OSEBX High	Return	Financial	Oslo Børs	Daily	0
OSEBX Low	Return	Financial	Oslo Børs	Daily	0
OSEBX Close	Return	Financial	Oslo Børs	Daily	0
Norway Household Consumption	Absolute	National	SSB	Quarterly	30
Norway Gross Fixed Capital Formation Oil	Absolute	National	SSB	Quarterly	30
Norway Gross Real Capital Formation	Absolute	National	SSB	Quarterly	30
Norway Export	Absolute	National	SSB	Quarterly	30
Norway Import	Absolute	National	SSB	Quarterly	30
Norway Domestic Debt	Absolute	National	SSB	Monthly	30
Norway Domestic Debt Provinces	Absolute	National	SSB	Monthly	31
Norway Domestic Debt Non-Financial Corporations	Absolute	National	SSB	Monthly	32
Norway Domestic Debt Households	Absolute	National	SSB	Monthly	33

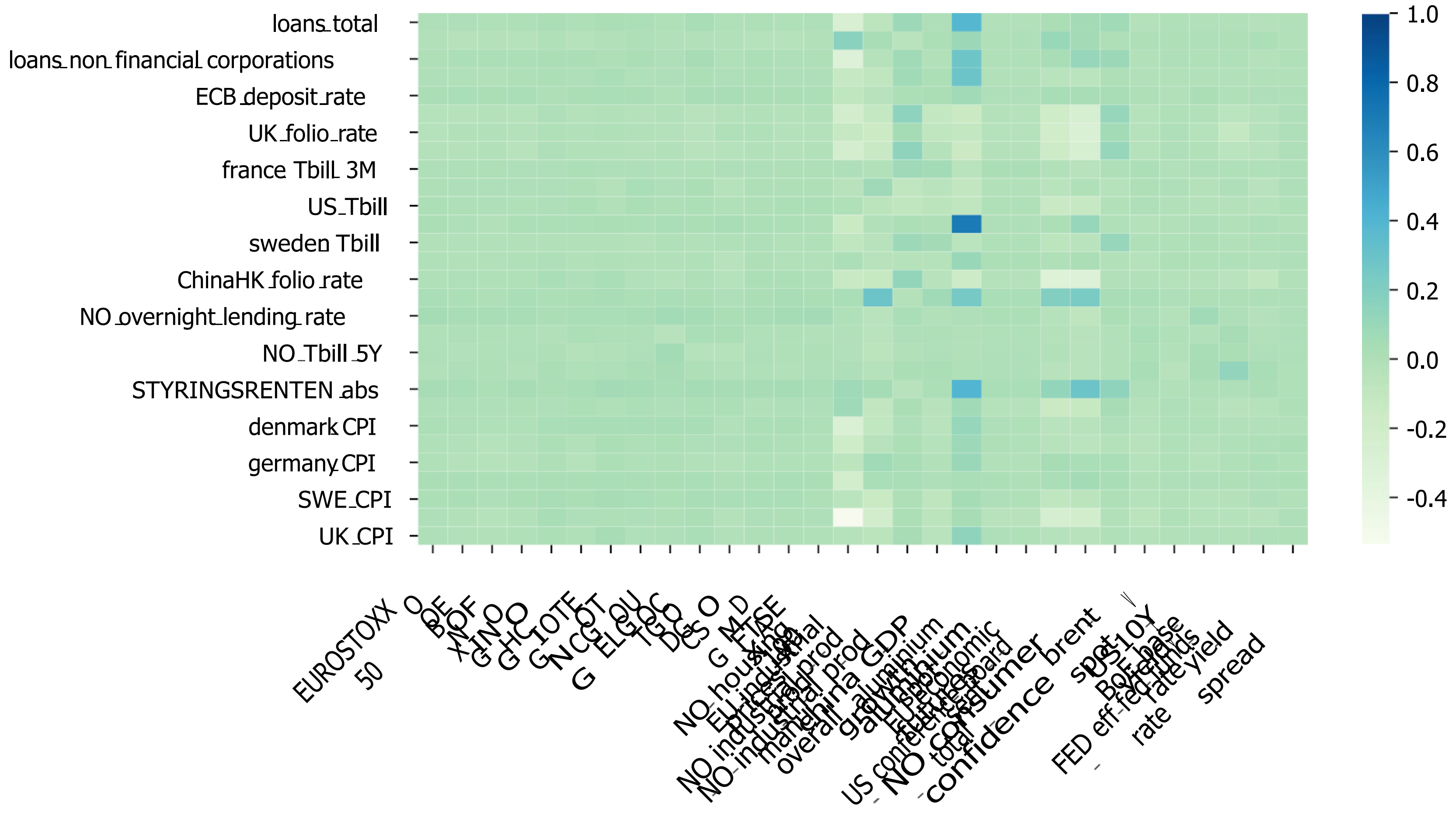
Exhibit 1: Overview of collected base variables. Lag refers to the publication lag of the variable, which is displayed in number of days.

Variable	Basis	Type	Source	Frequency	Lag
Spread Borrowing Rate/Policy Rate	Return	Interest Rate	SSB	Monthly	30
Power Prices	Absolute	Industry	SSB	Quarterly	30
Norway GDP Market Value	Absolute	National	SSB	Quarterly	30
UK Treasury Yield 3 Year	Return	Interest Rate	IMF	Monthly	0
France Treasury Yield 3 Year	Return	Interest Rate	IMF	Monthly	0
US Treasury Yield 3 Year	Return	Interest Rate	IMF	Monthly	0
Norway Treasury Yield 3 Year	Return	Interest Rate	IMF	Daily	0
Norway Treasury Yield 5 Year	Return	Interest Rate	IMF	Daily	0
Norway Treasury Yield 10 Year	Return	Interest Rate	IMF	Daily	0
UK Folio Rate	Return	Interest Rate	IMF	Monthly	0
UK Treasury Yield 1 Year	Return	Interest Rate	IMF	Monthly	0
US Treasury Yield 1 Year	Return	Interest Rate	IMF	Monthly	0
Sweden Treasury Yield 1 Year	Return	Interest Rate	IMF	Monthly	0
Hong Kong Treasury Yield 1 Year	Return	Interest Rate	IMF	Monthly	0
Norway Overnight Lending Rate	Return	Interest Rate	IMF	Daily	0
USD-to-NOK	Absolute	Currency	IMF	Daily	0
EUR-to-NOK	Absolute	Currency	IMF	Daily	0
GBP-to-NOK	Absolute	Currency	IMF	Daily	0
Norway Business Cycle	Return	National	SSB	Monthly	30

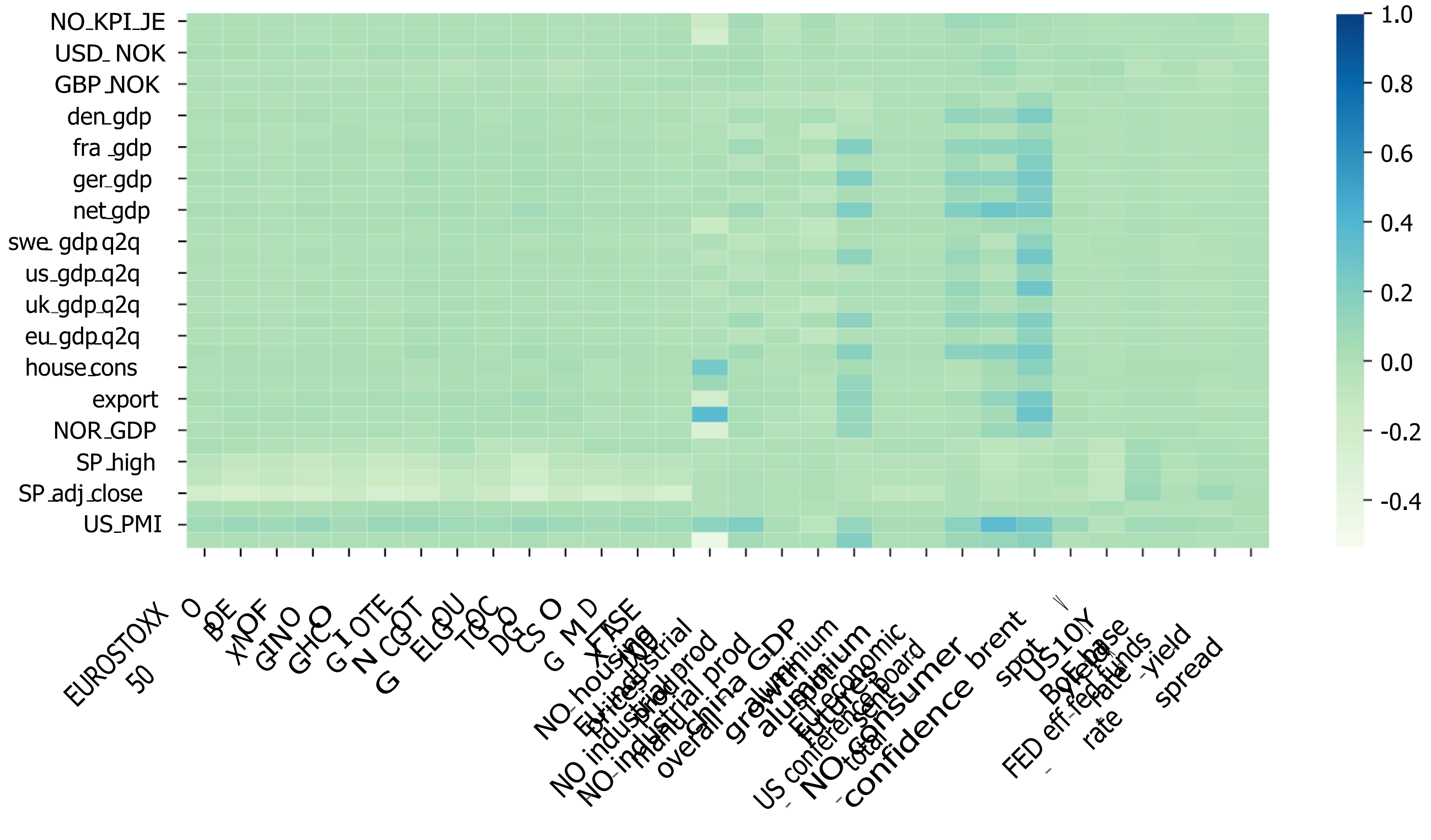
Exhibit 2: Correlation matrix of base variables. The full correlation matrix can be put together by places in the order of the index (*i*, *j*), where *i* refers to the row and *j* refers to the column.

(1, 1)

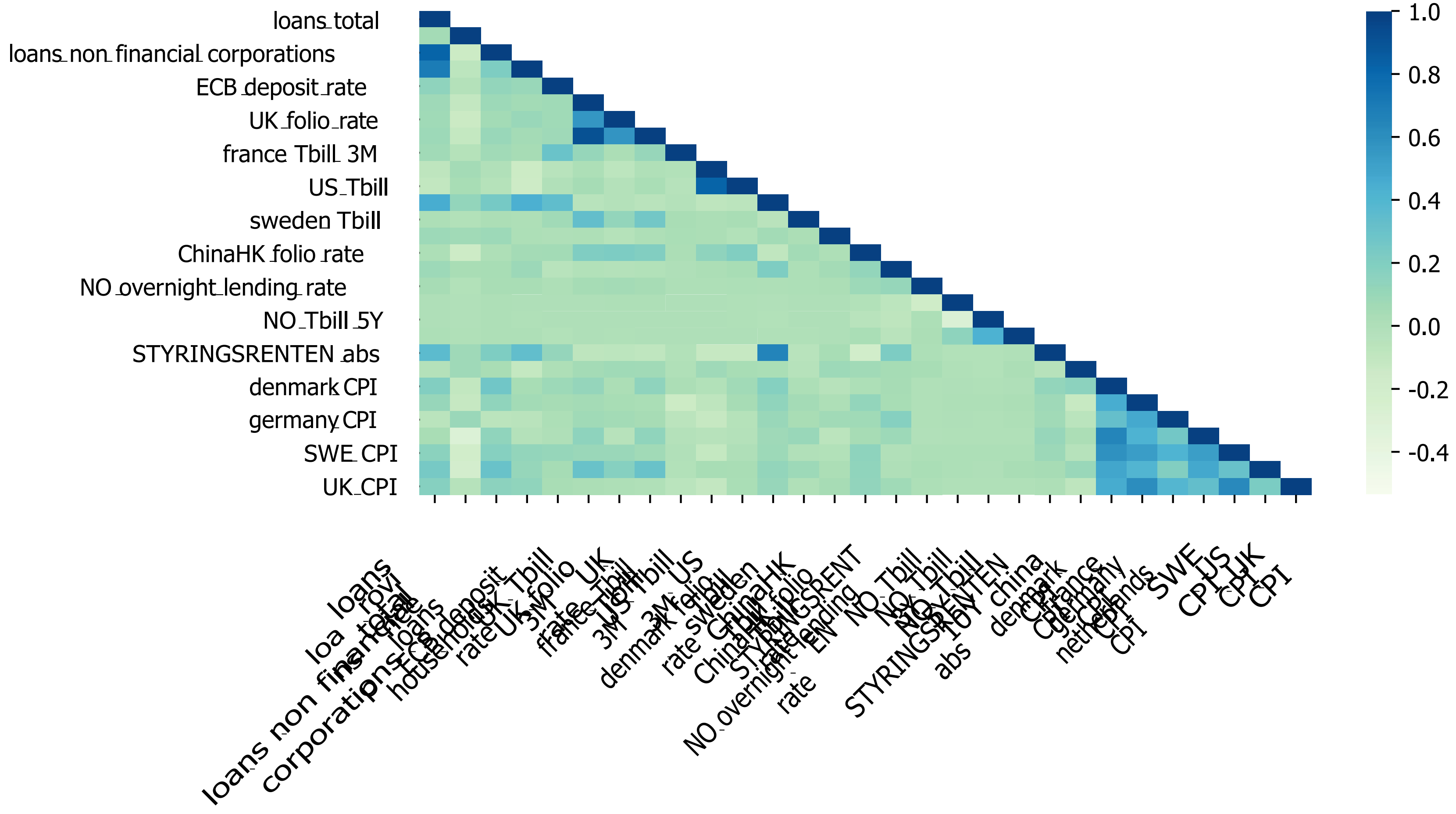



loans_total
loans_non financial_corporations
ECB_deposit_rate
UK_folio_rate
france Tbill_3M
US_Tbill
sweden Tbill
ChinaHK folio rate
NO overnight_lending_rate
NO_Tbill_5Y
STYRINGSRENTEN abs
denmark CPI
germany CPI
SWE_CPI
UK_CPI
EUROSTOXX 50
brent
spot
US 10Y
BoE base
FED eff fed funds rate
yield
spread
NO housing
china GDP
aluminium
confidence
1.0
0.8
0.6
0.4
0.2
0.0
-0.2
-0.4

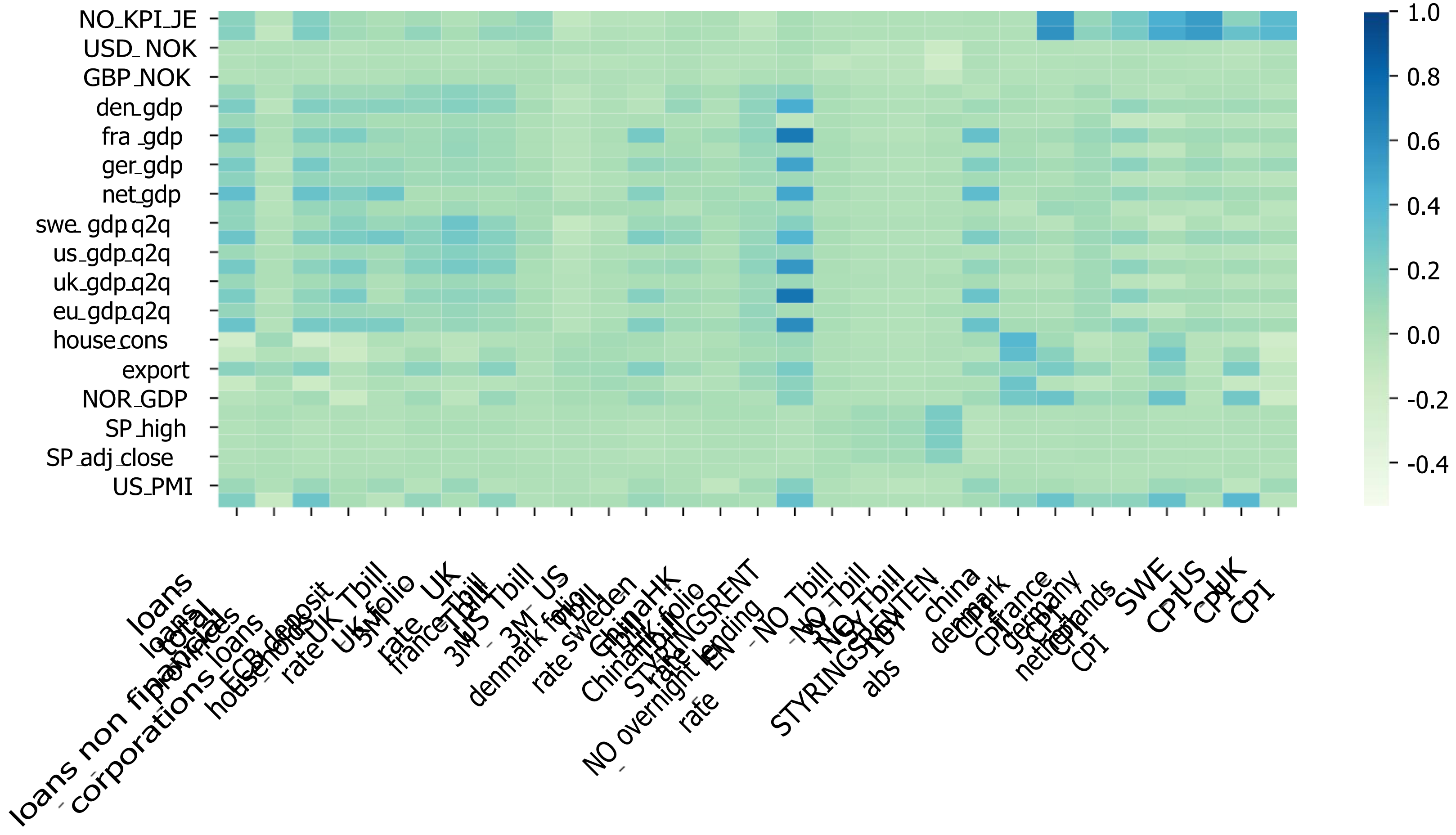
(2, 1)


NO_KPI_JE
USD_NOK
GBP_NOK
den_gdp
fra_gdp
ger_gdp
net_gdp
swe_gdp q2q
us_gdp_q2q
uk_gdp_q2q
eu_gdp_q2q
house_cons
export
NOR_GDP
SP_high
SP_adj_close
US_PMI
EUROSTOXX 50
FTSE
NO housing
EU industrial
NO industrial prod
NO industrial prod
manu
overall
china GDP growth
aluminium
aluminium futures
EU economic sentiment
US conference board total
NO consumer confidence
brent spot
US10Y
BoE base rate
FED eff fed funds rate
yield spread
1.0
0.8
0.6
0.4
0.2
0.0
-0.2
-0.4

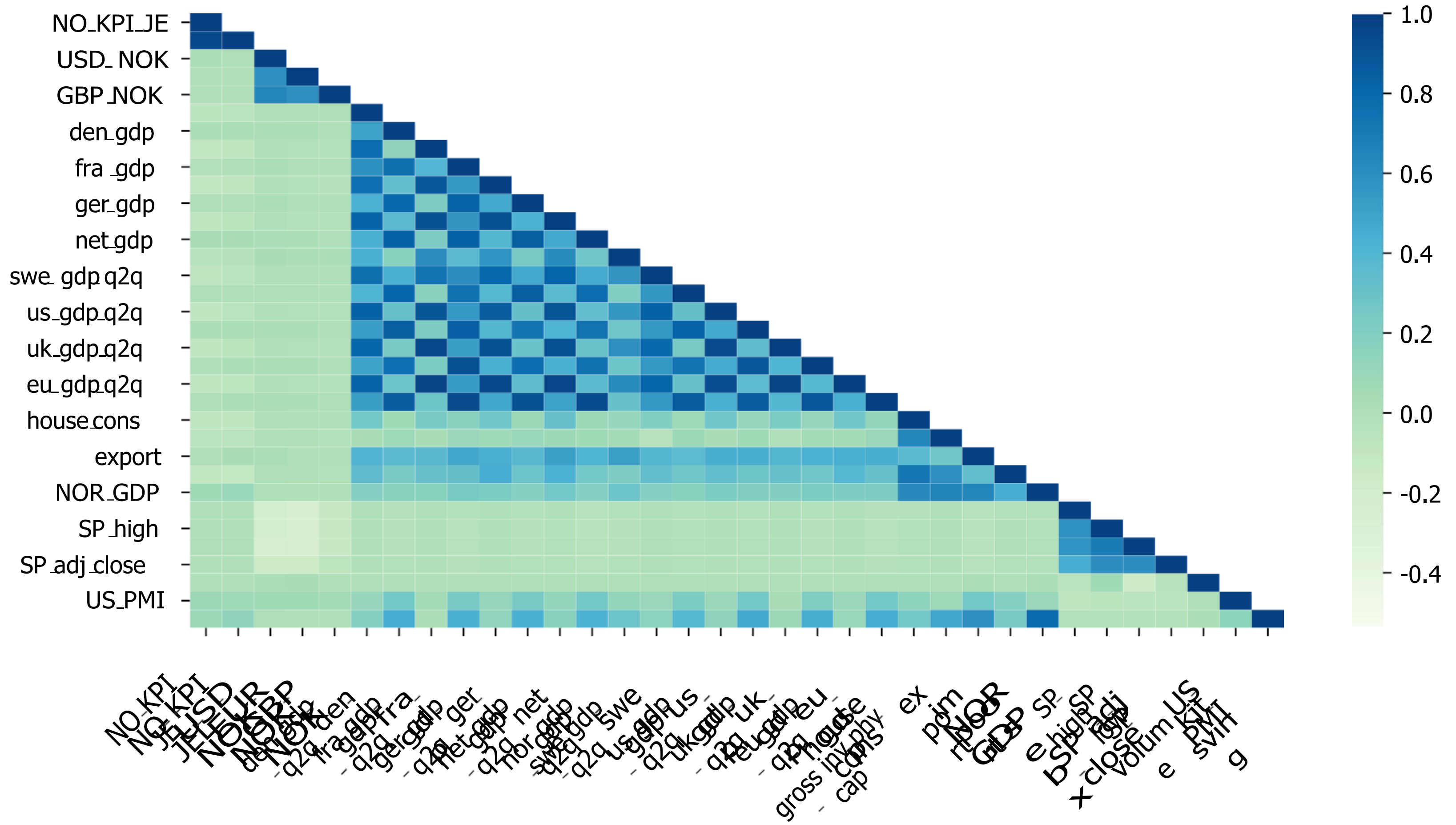
(3, 1)


loans_total
loans_non financial corporations
ECB_deposit_rate
UK_folio_rate
france Tbill_3M
US_Tbill
sweden Tbill
ChinaHK folio rate
NO overnight lending rate
NO_Tbill_5Y
STYRINGSRENTEN abs
denmark CPI
germany CPI
SWE_CPI
UK_CPI
1.0
0.8
0.6
0.4
0.2
0.0
-0.2
-0.4

(2,2)


NO_KPI_JE
USD_NOK
GBP_NOK
den_gdp
fra_gdp
ger_gdp
net_gdp
swe_gdp q2q
us_gdp_q2q
uk_gdp_q2q
eu_gdp_q2q
house cons
export
NOR_GDP
SP_high
SP_adj close
US_PMI
1.0
0.8
0.6
0.4
0.2
0.0
-0.2
-0.4
loans
loans total
loans non financial corporations
households loans
ECB deposit rate
UK Tbill 3M
UK folio
UK rate
france Tbill 3M
US Tbill 3M
US folio
denmark Tbill
rate sweden
China Tbill
ChinaHK
HK folio
STYRINGSRENT
rate lending
NO overnight rate
EN NO Tbill
NO Tbill
NO Tbill
STYRINGSRENTEN abs
NOY
china CPI
denmark CPI
france CPI
germany CPI
netherlands CPI
SWE CPI
US CPI
UK CPI

(3,2)


NO_KPI_JE
USD_NOK
GBP_NOK
den_gdp
fra_gdp
ger_gdp
net_gdp
swe_gdp q2q
us_gdp_q2q
uk_gdp_q2q
eu_gdp_q2q
house cons
export
NOR_GDP
SP_high
SP_adj_close
US_PMI
1.0
0.8
0.6
0.4
0.2
0.0
-0.2
-0.4

(3,3)

Exhibit 3: Variables excluded during dataframe slicing.

Variables excluded:
sweden_folio_rate_m_2_m
sweden_folio_rate_m_2_2m
sweden_folio_rate_m_2_k
sweden_folio_rate_m_2_2k
sweden_folio_rate_m_2_y
interest_rates_on_loans_m_2_m
interest_rates_on_loans_m_2_2m
interest_rates_on_loans_m_2_k
interest_rates_on_loans_m_2_2k
interest_rates_on_loans_m_2_y
FED_QE_k_2_k
FED_QE_q_2_2k
FED_QE_q_2_y
fish_pool_NOK_w_2_w
fish_pool_NOK_w_2_2w
fish_pool_NOK_w_2_m
fish_pool_NOK_w_2_2m
fish_pool_NOK_w_2_k
fish_pool_NOK_w_2_2k
fish_pool_NOK_w_2_y
cont.
ECB_QE_å_2_å
oil_inv_k_2_k
oil_inv_q_2_2k
oil_inv_q_2_y
EUR_NOK_d_2_y
NO_EL_price_k_2_k
NO_EL_price_q_2_2k
NO_EL_price_q_2_y
NO_KPI_JA_m_2_m
NO_KPI_JA_m_2_2m
NO_KPI_JA_m_2_k
NO_KPI_JA_m_2_2k
NO_KPI_JA_m_2_y
NO_KPI_JAE_m_2_m
NO_KPI_JAE_m_2_2m
NO_KPI_JAE_m_2_k
NO_KPI_JAE_m_2_2k
NO_KPI_JAE_m_2_y

Exhibit 4: Norges Bank's policy rate forecast from monetary policy report published on March 12th, 2020. (Norges Bank, 2020b)

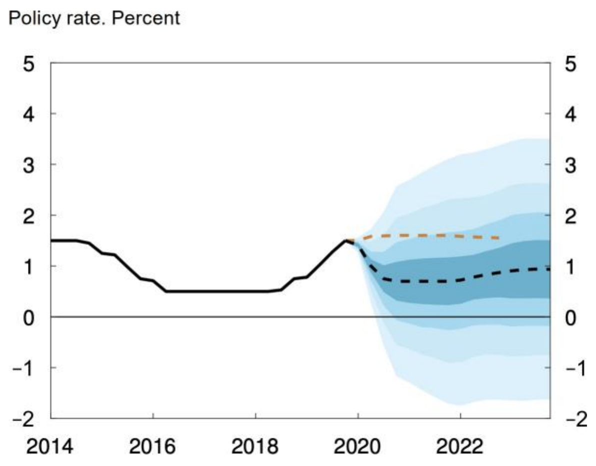


Exhibit 5: Decision trees

Exhibit 6.1: Decision tree output for t = 0 predictions.

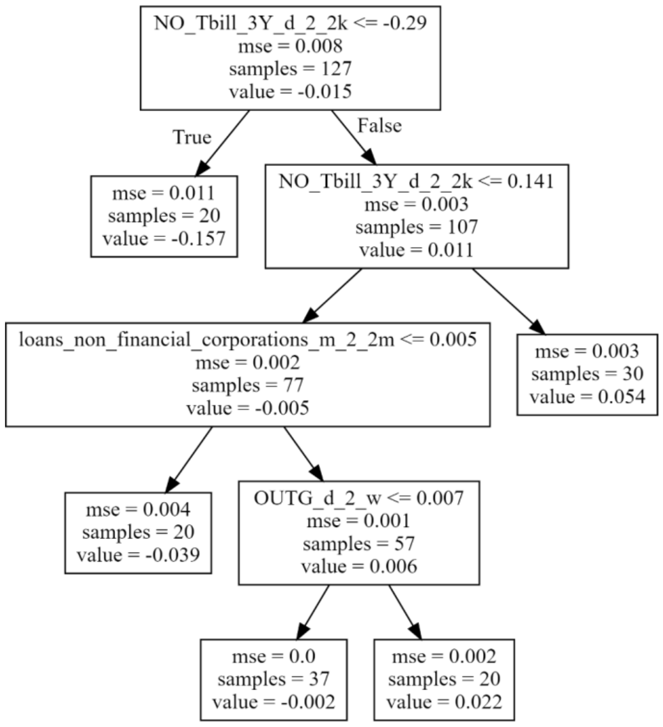


Exhibit 6.2: Decision tree output for t = 1 predictions.

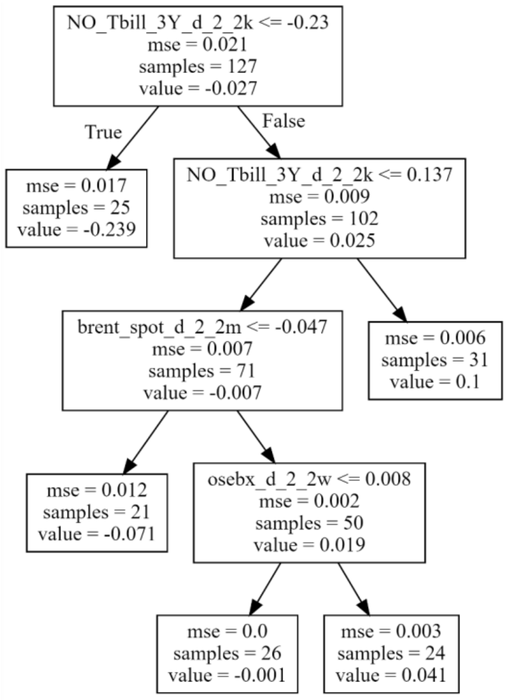


Exhibit 6.3: Decision tree output for $t = 2$ predictions.

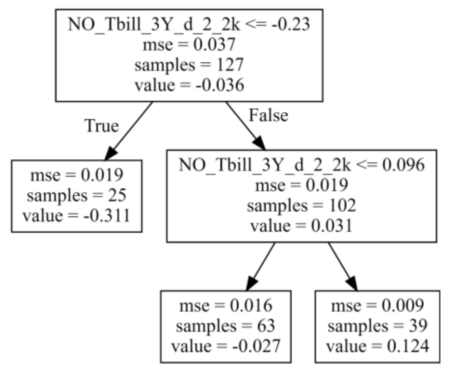


Exhibit 6.4: Decision tree output for $t = 3$ predictions.

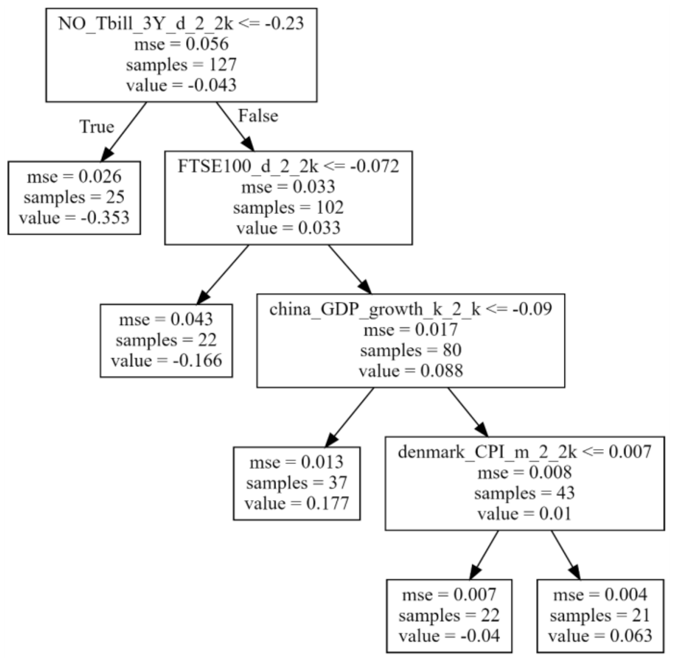


Exhibit 6.5: Decision tree output for t = 4 predictions.

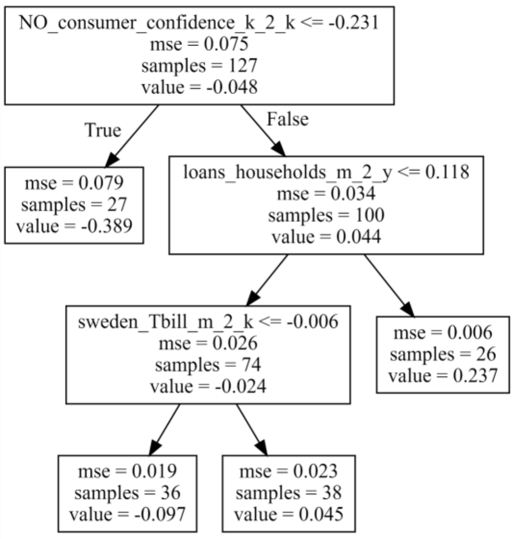


Exhibit 6: Elastic net coefficients for the nowcasting-, one-step-, and four-step model.

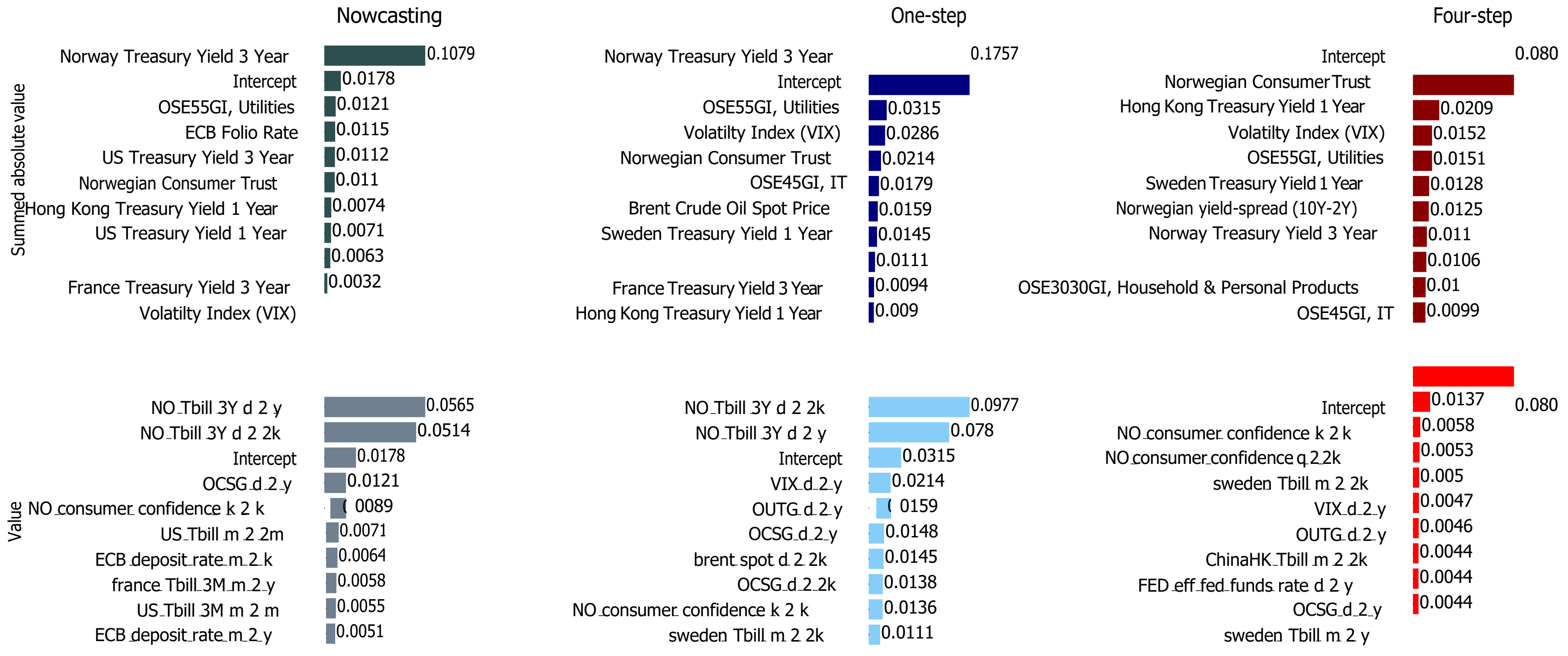


Exhibit 7: Ensemble learner coefficients across time-steps.

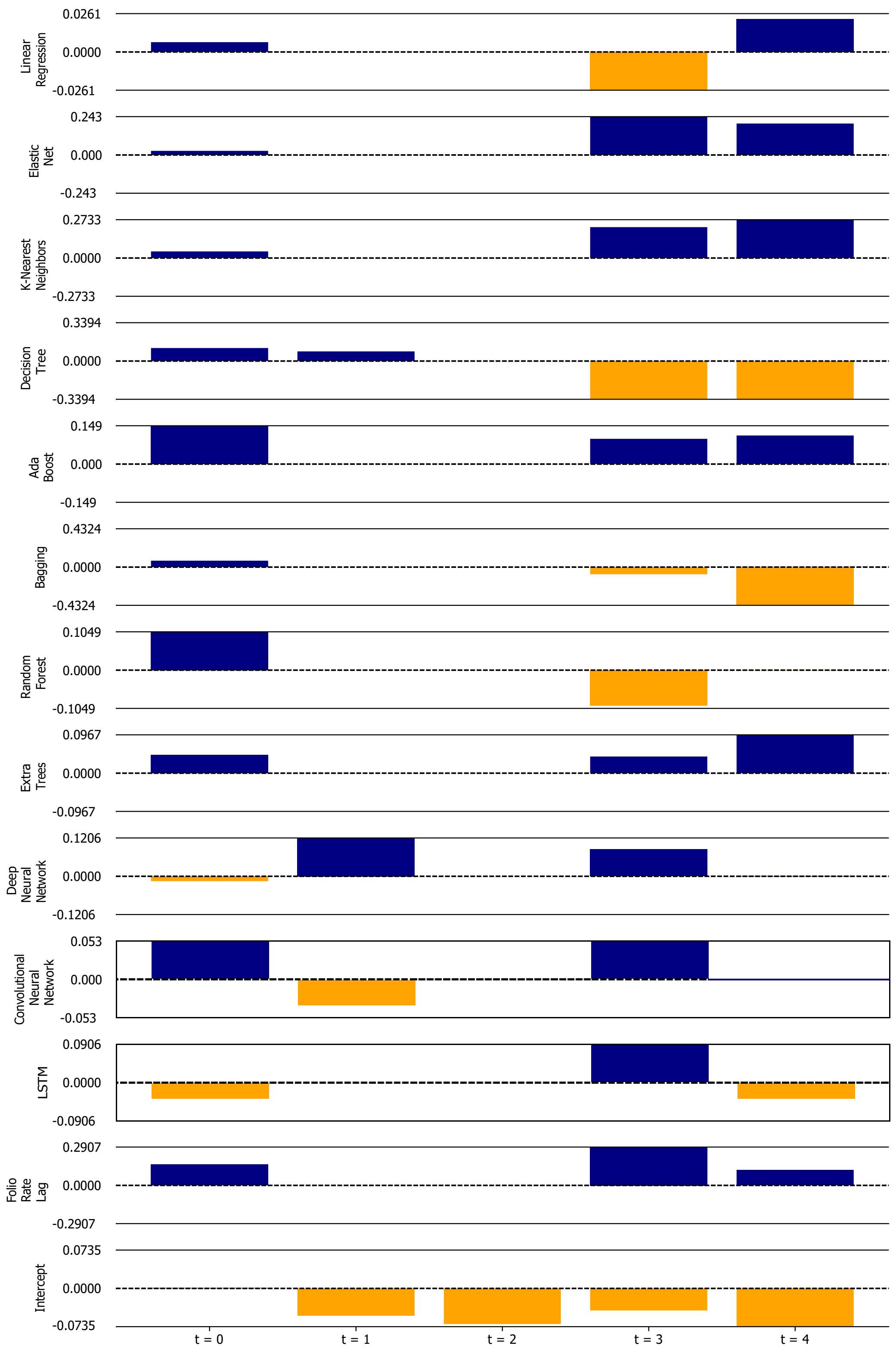


Exhibit 8: URL to github repository for Python codes and data sets.

Github repository:

https://github.com/BenjLian/Using-artificial-intelligence-in-economic-policy-forecasting.