Nowcasting and forecasting US recessions: Evidence from the Super Learner

Maas, Benedikt

Hamburg University

September 2019

Nowcasting and forecasting US recessions: Evidence from the Super Learner

Benedikt Maas*

September 2019

Abstract

This paper introduces the Super Learner to nowcast and forecast the probability of a US economy recession in the current quarter and future quarters. The Super Learner is an algorithm that selects an optimal weighted average from several machine learning algorithms. In this paper, elastic net, random forests, gradient boosting machines and kernel support vector machines are used as underlying base learners of the Super Learner, which is trained with real-time vintages of the FRED-MD database as input data. The Super Learner's ability to categorise future time periods into recessions versus expansions is compared with eight different alternatives based on probit models. The relative model performance is evaluated based on receiver operating characteristic (ROC) curves. In summary, the Super Learner predicts a recession very reliably across all forecast horizons, although it is defeated by different individual benchmark models on each horizon.

JEL classification: C32, C53, C55, E32

*Hamburg University, Department of Economics, Von-Melle-Park 5, 20146 Hamburg, Germany; E-mail: *Benedikt.Maas@uni-hamburg.de*

1. Introduction

Accurately predicting turning points in the economy and identifying recessions is of great importance for economic agents, such as central bankers and policy makers. Previous research has shown that probit models and related approaches are powerful and reliable tools to classify recessions (see Estrella and Hardouvelis (1991), Estrella and Mishkin (1996), Estrella and Mishkin (1998), Stock and Watson (1989), and Liu and Moench (2016)). However, these approaches can only use a limited amount of forward-looking predictor variables. In contrast, machine learning algorithms are able to cope with a huge set of data and they can capture any non-linearities in the data.

A key question in the application of machine learning algorithms is the selection of the most predictive algorithm. However, there is still no consensus in the economic literature on which machine learning algorithm is most promising in detecting recessions. Against the backdrop of the Great Recession 2008/09, researchers have used different machine learning methods to predict recessions. For example, Ng (2014) and Döpke et al. (2017) use boosted regression trees (BRTs), whereas Gogas et al. (2015) use support vector machines (SVMs). By acknowledging that different algorithms have different advantages under certain circumstances, this paper takes an alternative path and develops a methodology that uses an ensemble of different algorithms to combine the best properties of each machine learning method. This ensemble learner is the so called "Super Learner" of van der Laan et al. (2007), which is an algorithm that selects an optimal weighted average of multiple machine learning algorithms. Jung et al. (2018) have been the first using the Super Learner for forecasting GDP growth, while this paper introduces the Super Learner as a tool for nowcasting and forecasting the likelihood of a US recession in the current quarter and in the following four quarters. The Super Learner is built from four widely-used machine learning algorithms, namely elastic net, random forests, gradient boosting machines (GBMs) and kernel support vector machines (KSVMs). These specific algorithms are chosen as so called "base learners" because they are the most commonly used classification tools in the machine learning literature.

Moreover, the prediction and exact identification of a recession is rather difficult and challenging in advance because of the so called "ragged edge" structure of the data, as described in Wallis (1986). When estimating macroeconomic variables in real-time, some data series have observations through the current period, whereas for others the most recent observations may only be available for a month or a quarter earlier, which creates an unbalanced dataset. To provide realistic time conditions for the creation of the nowcasts and forecasts, real-time vintages of the FRED-MD database are used as the underlying dataset. The nowcasts and forecasts of the quarterly probabilities of a recession within the current quarter and for future quarters are conducted after the second month of each quarter, where the Super Learner algorithm is trained with a large database on past values of the National Bureau of Economic Research (NBER) recession indicator.

The Super Learner's performance is compared with a total of eight benchmark models, which are all based on standard univariate and multivariate probit models. Four benchmark models are based on nowcast and forecast values of US GDP growth which are then used as predictor variables in the probit models. The nowcasts and forecasts of the GDP growth values are obtained from: i) a multilayer perceptron (MLP), ii) an extreme learning machine (ELM), iii) a dynamic factor model (DFM), and iv) the Survey of Professional Forecasters (SPF). Several studies have shown the predictive power of the term structure of Treasury yields to forecast recessions (Estrella and Hardouvelis, 1991; Estrella and Mishkin, 1998). Recently, Liu and Moench (2016) show that at short forecast horizons, adding a few leading financial and economic indicators can improve the recession predictability relative to forecasts based only on the Treasury term spread and its lags. Following their study, the remaining benchmark models are based on the term spread and its lagged value, which are combined with different economic and financial covariates: i) all nonfarm payroll employees, ii) returns on the S&P 500 common stock price index, iii) the spread between the yield of one-year constant maturity Treasuries and the federal funds rate, and iv) the spread between the yield of five-year constant maturity Treasuries and the federal funds rate.

Finally, the ability of the individual models to categorise current and future periods in recessions versus expansions is measured by receiver operating characteristic (ROC) curves and the corresponding area under receiver operating characteristic (AUROC) curves, as in Jordà and Taylor (2011) and Jordà and Taylor (2012). In summary, the Super Learner predicts a recession very reliably across all forecast horizons, although it is defeated by different benchmark models on each horizon.

The remainder of this paper is organised as follows. Section 2 introduces the Super Learner algorithm, the four different machine learning algorithms of the Super Learner, the benchmark models and the model evaluation procedure. Section 3 describes the applied data. Section 4 presents the empirical results. Finally, Section 5 concludes.

2. Methodology

This section contains a description of the empirical methods that have been used in this paper. First, the main machine learning algorithm—the Super Learner— is introduced and then the machine learning algorithms that form the Super Learner are briefly explained. Elastic net, random forests, GBMs and KSVMs are used as underlying base learners in this paper. These four machine learning algorithms are selected because they are considered to be the most common classification models within the machine learning literature. Furthermore, the benchmark models are presented, which are based on standard probit models with different predictor variables. Finally, the model evaluation measures—ROC curve and AUROC—are described.

2.1. Super Learner

The Super Learner is an algorithm that selects an optimal weighted average of multiple machine learning algorithms. It was first introduced in van der Laan et al. (2007) and Dudoit and van der Laan (2005), and is a generalisation of the stacking algorithm introduced by Wolpert (1992), which was adopted to model linear regressions by Breiman (1996b).[1] This algorithm can be used for both regression applications or classification problems, and it can handle large datasets.

In the context of prediction, the Super Learner algorithm applies a set of candidate prediction algorithms— the base learners—to the underlying dataset. The base learners can be any parametric or nonparametric supervised machine learning algorithm and the Super Learner theory does not require any specific level of diversity among the set of base learners. The Super Learner then works as a "metalearner" to find the optimal combination of the set of base learners. The metalearner algorithm is typically a method designed to minimise the cross-validated risk of some loss function—for example the mean squared prediction error. Because the set of predictions from the different base learners may be highly correlated, it is advisable to choose a metalearning method like the Super Learner which performs well in the presence of collinear predictions (Bühlmann et al., 2016).

In the following, the general Super Learner algorithm for prediction is described. The description that follows is based on Polley and van der Laan (2010) and van der Laan et al. (2007).

Suppose the learning dataset $X_i = (Y_i, W_i)$ for $i = 1, \dots, n$, where Y is the outcome of interest and W is a p-dimensional set of covariates. The objective is to estimate the function $\psi_0 = \mathrm{E}(Y|W)$. The function can be expressed as the minimiser of the expected loss:

$$\psi_0(W) = \arg\min_{\psi} \mathrm{E}\left[L(X, \psi(W))\right], \tag{2.1}$$

where the loss function is often the squared error loss: $L_2 : (Y - \psi(W))^2$. For a given problem, a "library" of various different prediction algorithms can be proposed. Denote the library as L and the cardinality of L as $K(n)$.

[1] Wolpert (1992) and Breiman (1996b) used the same underlying algorithm with different tuning parameters as base learners, whereas the Super Learner uses various different machine learning algorithms as base learners.

1. Fit each algorithm in L on the entire dataset $X = \{X_i : i = 1, \ldots, n\}$ to estimate $\widehat{\Psi}_k(W),\ k = 1, \ldots, K(n)$.

2. Split the dataset X into training and validation sample, according to a V-fold cross-validation scheme: split the ordered n observations into V-equal size groups, let the v-th group be the validation sample, and the remaining group the training sample, $v = 1, \ldots, V$. Define $T(v)$ to be the v^{th} training data split and $V(v)$ to be the corresponding validation data split. $T(v) = X \setminus V(v),\ v = 1, \ldots, V$.

3. For the v^{th} fold, fit each algorithm in L on $T(v)$ and save the predictions on the corresponding validation data, $\widehat{\Psi}_{k,T(v)}(W_i),\ X_i \in V(v)$ for $v = 1, \ldots, V$.

4. Stack the predictions from each algorithm together to create a n by K matrix, $Z = \left\{\widehat{\Psi}_{k,T(v)}\left(W_{V(v)}\right),\ v = 1, \ldots, V \text{ and } k = 1, \ldots, K\right\}$, where the notation $W_{V(v)} = (W_i : X_i \in V(v))$ is used for the covariate-vectors of the $V(v)$-validation sample.

5. Propose a family of weighted combinations of the candidate estimators indexed by weight-vector α:
$$m(z|\alpha) = \sum_{k=1}^{K} \alpha_k \widehat{\Psi}_{k,T(v)}\left(W_{V(v)}\right),\ \alpha_k \geq 0\ \forall k,\ \sum_{k=1}^{K} \alpha_k = 1. \tag{2.2}$$

6. Determine the α that minimises the cross-validated risk of the candidate estimator $\sum_{k=1}^{K} \alpha_k \widehat{\Psi}_k$ over all allowed α-combinations:
$$\widehat{\alpha} = \arg\min_{\alpha} \sum_{i=1}^{n} \left(Y_i - m(z_i|\alpha)\right)^2. \tag{2.3}$$

7. Combine $\widehat{\alpha}$ with $\widehat{\Psi}_k(W),\ k = 1, \ldots, K$ according to the family $m(z|\alpha)$ of weighted combinations to create the final Super Learner fit:
$$\widehat{\Psi}_{SL}(W) = \sum_{k=1}^{K} \widehat{\alpha}_k \widehat{\Psi}_k(W). \tag{2.4}$$

Theoretical results show that such an optimal learner will perform asymptotically as well as or better than any of the candidate base learners (van der Laan et al., 2007). This motivated the naming "Super Learner" since it provides a system of combining many estimators into an improved estimator (Polley and van der Laan, 2010).

The base learners, which build the Super Learner to nowcast and forecast US recessions in this paper, are briefly described in the following sections.

2.1.1. Elastic net

The first base learner to be presented is the elastic net. The elastic net algorithm was originally proposed by Zou and Hastie (2005) and is a combination of the ridge and least absolute shrinkage and selection operator (LASSO) regressions. Both approaches are forms of penalised regressions which generally improves ordinary least squares (OLS) regressions by using dimension reduction and variable selection approaches when dealing with large datasets. In the following, the ridge and LASSO regressions are both briefly stated, before the elastic net is presented. The description that follows is based on Tiffin (2016) and Jung et al. (2018).

In general, the ridge regression minimises the residual sum of squares (RSS) and also an additional shrinkage penalty term which decreases when the estimated coefficients of the regression become close to zero. When both the RSS and the shrinkage penalty are minimised, the optimal result will be achieved by shrinking those regressors of the dataset which are correlated. The overall minimisation problem is given as follows:

$$\widehat{\beta} = \arg \min_{\widehat{\beta}_j} \left[\underbrace{\sum_{i=1}^{n} \left(Y - X\widehat{\beta} \right)^2}_{\text{RSS}} + \underbrace{\lambda \sum_{j=1}^{p} \left(\widehat{\beta}_j \right)^2}_{\text{ridge penalty}} \right], \tag{2.5}$$

where n is the number of observations, p the number of explanatory variables and λ denotes the shrinkage penalty parameter, which will be determined by iterative cross-validation. A higher λ will lead to a stronger shrinkage, whereas a λ of zero produces the same result as standard OLS regression.

The LASSO was proposed by Tibshirani (1996) and also shrinks the coefficients of an OLS regression, but uses a different penalty term compared to the ridge regression. The overall minimisation problem is then given as follows:

$$\widehat{\beta} = \arg \min_{\widehat{\beta}_j} \left[\underbrace{\sum_{i=1}^{n} \left(Y - X\widehat{\beta} \right)^2}_{\text{RSS}} + \underbrace{\lambda \sum_{j=1}^{p} \left| \widehat{\beta}_j \right|}_{\text{LASSO penalty}} \right]. \tag{2.6}$$

In Equation (2.6), zero coefficients are possible if the parameter λ is large enough. Hence, the LASSO is able to select variables from the given dataset whereas the ridge regression only shrinks the coefficients close to zero and does not exclude them from the model.

Finally, the elastic net algorithm combines both penalty terms from Equations (2.5) and (2.6):

$$\widehat{\beta} = \arg \min_{\widehat{\beta}_j} \left[\underbrace{\sum_{i=1}^{n} \left(Y - X\widehat{\beta} \right)^2}_{\text{RSS}} + \lambda \sum_{j=1}^{p} \left[\underbrace{(1-\alpha) \left(\widehat{\beta}_j \right)^2}_{\text{ridge}} + \underbrace{\alpha \left| \widehat{\beta}_j \right|}_{\text{LASSO}} \right] \right], \tag{2.7}$$

where the parameter α determines the relative weights of the penalty terms. It is selected via cross-validation. In general, the elastic net combines the advantages from both the ridge regression and the LASSO and overcomes their individual weakness.[2] Zou and Hastie (2005) state that the elastic net is superior or at least as good as both standalone models in situations where the number of regressors exceed the number of observations ("fat data"), when the number of observations largely exceeds the number of regressors ("tall data") or when multiple variables are highly correlated.

2.1.2. Random forests

Beside elastic net, the Super Learner algorithm can also choose random forests as potential base learner. In general, random forests belong to the family of decision trees. They were proposed by Breiman (2001) and they are an advancement of the related classification tree algorithm called bagging (bootstrap aggregating) introduced by Breiman (1996a). In bagging, the decision trees are not completely independent of each other since all variables are considered at every split of the tree. Random forests overcome this feature by adding an additional layer of randomness.

Similar to bagging, random forests also construct each tree using a different bootstrap sample of the data, but they change how the classification trees are constructed. In bagging, each node is split using the best split among all variables, whereas in random forests, each node is split using the best among a subset of variables randomly chosen at that node. This strategy turns out to perform very well compared to other classifiers and is also robust against overfitting (Breiman, 2001).

The description about the random forests algorithm that follows is based on Efron and Hastie (2016):

Suppose a training set consisting of an $n \times p$ data matrix X and an n-vector of responses y.

1. Given the training dataset $d = (X, y)$, fix $m \leq p$ and the number of trees B.

[2]For extensive details about the ridge regression, the LASSO and the elastic net, see Zou and Hastie (2005) and Friedman et al. (2010).

2. For $b = 1, 2, \ldots, B$ do the following:

   a) Create a bootstrap version of the training data d_b^*, by randomly sampling the n rows with replacement n times.

   b) Grow a maximal-depth tree $\widehat{r}_b(x)$ using the data in d_b^*, sampling m of the p features at random prior to making each split.

   c) Save the tree, as well as the bootstrap sampling frequencies for each of the training observations.

3. Compute the random forests fit at any prediction point x_0 as the average

$$\widehat{r}_{rf}(x_0) = \frac{1}{B} \sum_{b=1}^{B} \widehat{r}_b(x_0). \tag{2.8}$$

4. Compute the "out-of-bag" error OOB_i for each response observation y_i in the training data, by using the fit $\widehat{r}_{rf}^{(i)}$, obtained by averaging only those $\widehat{r}_b(x_i)$ for which observation i was not in the bootstrap sample. The overall OOB error is the average of these OOB_i.

2.1.3. Gradient boosting machine

GBMs are based on multiple decision trees like random forests, but the treatment of the single trees is rather different. In general, the term "boosting" has been originally developed for classification problems. The first boosting algorithms were introduced by Schapire (1990), Freund (1995) and Freund and Schapire (1999) and they combine (or "boost") a number of weak classifiers (a classifier that predicts marginally better than random) into a superior ensemble classifier. One of the most popular boosting algorithm is the Adaboost ("Adaptive boosting") algorithm (Freund and Schapire, 1997), which was then connected by Friedman et al. (2000) to statistical concepts of loss functions and logistic regressions, showing that boosting can be interpreted as a forward stagewise additive model that minimises exponential loss.

Friedman (2001) developed a highly adaptable method for both classification and regression problems which he called "gradient boosting machine". The intuition of Friedman's GBMs taken from Kuhn and Johnson (2013) is the following:

Given a specific loss function and a weak learner like regression trees, the algorithm seeks to find an additive model that minimises the loss function. After an initialisation, the gradient (e.g., residual) is calculated, and a model is then fit to the residuals to minimise the loss function. The current model is added to the previous model, and the procedure continues for a user-specified number of iterations. When trees are used as the base learner, basic gradient boosting machine has three tuning parameters: number of iterations B, tree depth d , and shrinkage parameter ε.

A more formal presentation of the gradient boosting machine is briefly stated in the following. The notation is based on Efron and Hastie (2016). Suppose that we are interested in modeling $\mu(x) = \Pr(Y = 1|X = x)$ for a Bernoulli response variable. The idea is to fit a model of the form

$$\lambda(x) = G_B(x) = \sum_{b=1}^{B} g_b(x; y_b), \tag{2.9}$$

where $\lambda(x)$ is the natural parameter in the conditional distribution of $Y|X = x$, and $g_b(x; y_b)$ are functions like shallow trees. In the case of the Bernoulli response, we have

$$\lambda(X) = \log\left(\frac{\Pr(Y = 1|X = x)}{\Pr(Y = 0|X = x)}\right). \tag{2.10}$$

The gradient boosting algorithm works as follows:

1. Start with $\widehat{G}_0(x) = 0$, and set B and the shrinkage parameter $\varepsilon > 0$.

2. For $b = 1, 2, \ldots, B$ repeat the following steps:

   a) Compute the pointwise negative gradient of the loss function at the current fit:

   $$r_i = -\left.\frac{\partial L(y_i, \lambda_i)}{\partial \lambda_i}\right|_{\lambda_i = \widehat{G}_{b-l(x_i)}}, \quad l = 1, \ldots, n. \tag{2.11}$$

   b) Approximate the negative gradient by a depth-d tree by solving

   $$\min_{y} \sum_{i=1}^{n} (r_i - g(x_i; y))^2. \tag{2.12}$$

   c) Update $\widehat{G}_b(x) = \widehat{G}_{b-1}(x) + \widehat{g}_b(x)$, with $\widehat{g}_b = \varepsilon \times g(x; \widehat{y}_b)$.

3. Return the sequence $\widehat{G}_b(x)$, $b = 1, \ldots, B$.

2.1.4. Kernel support vector machine

The final base learner is a KSVM, which is a specific type of a SVM. A SVM can be used for classification and regression analysis and was first introduced by Vapnik (1998). In general, a SVM can be imagined as a surface that creates a boundary between points of data plotted in a multidimensional space. The goal of a SVM is to create a flat boundary called a hyperplane, which divides the space to create partitions on either side. SVMs are most easily understood when used for binary classification within a linear framework, which is how the method has been traditionally applied.

However, in many real-world applications, the relationships between variables are non-linear. Nevertheless, SVMs can still be trained on non-linear data through the addition of a so-called "slack" variable, which to some extent allows for misclassification, or more promisingly by the use of the so-called "kernel trick", leading to KSVMs.[3] The following brief explanation of KSVMs is based on Kecman (2005):

Consider the problem of binary classification, where the training data are given as

$$(x_1, y_1), (x_2, y_2), \ldots, (\mathrm{x}_l, y_l), \; x \in R^n, \; y \in \{+1, -1\}. \tag{2.13}$$

In the case of the classification of linearly separable data, the goal of a SVM is to find among all the hyperplanes that minimise the training error to find the one with the largest margin. This is done by—using the given training examples—finding parameters $w = [w_1, w_2, \ldots, w_n]^T$ and the scalar b of the following decision function $d(x, w, b)$

$$d(x, w, b) = w^T x + b = \sum_{i=1}^{n} w_i x_i + b, \tag{2.14}$$

where $x, w \in R^n$. After obtaining the weights, testing on unseen data x_p the vector machine produces output 0 according to an indicator function given as

$$i_F = o = \operatorname{sign}(d(x_p, w, b)), \tag{2.15}$$

where o stands for output. The decision rule of the binary classification task is as follows: if $d(x_p, w, b) > 0$, then x_p belongs to class 1 ($o = y_1 = +1$), and if $d(x_p, w, b) < 0$, then x_p belongs to class 2 ($o = y_2 = -1$).[4]

The intuition of the SVM works also for non-linear data, because KSVMs are able to map a problem into a higher dimension space using the kernel trick, making a non-linear relationship appear to be quite linear. The basic idea in designing non-linear KSVMs is to map input vectors $x \in R^n$ into vectors $\Phi(x)$ of a higher dimensional feature space F (where Φ represents mapping: $R^n \rightarrow R^f$), and to solve a linear classification problem in this feature space

$$x \in R^n \rightarrow \Phi(x) = [\phi_1(x), \phi_2(x), \ldots, \phi_n(x)]^T \in R^f. \tag{2.16}$$

[3] For the kernel trick, see Schölkopf et al. (2002).

[4] For a more detailed explanation of the classification of linearly separable data, see Kecman (2005).

The mapping $\Phi(x)$ is chosen in advance. By performing a mapping, in a Φ-space, the learning algorithm will be able to linearly separate images of x by applying the linear SVM framework. This approach also leads to an optimisation problem with similar constraints in a Φ-space. The solution for an indicator function $i_F(x) = \text{sign}\left(w^T\Phi(x) + b\right) = \text{sign}\left(\sum_{i=1}^{l} y_i\alpha_i\Phi^T(x_i)\Phi(x) + b\right)$, which is a linear classifier in a feature space, creates a non-linear separating hypersurface in the original space by the following indicator function:

$$\begin{aligned} i_F &= \text{sign}\left(\sum_{i=1}^{l} y_i\alpha_i\Phi^T(\mathbf{x}_i)\Phi(\mathbf{x}) + b\right) \\ &= \text{sign}\left(\sum_{i=1}^{l} y_i\alpha_i k(\mathbf{x}_i, \mathbf{x}) + b\right) \\ &= \text{sign}\left(\sum_{i=1}^{l} v_i k(\mathbf{x}_i, \mathbf{x}) + b\right), \end{aligned} \tag{2.17}$$

where v_i corresponds to the output layer weights of the SVM and $k(x_i, x)$ denotes the value of the kernel function. The kernel function is a function in input space. Thus, by using a kernel function, it is no longer necessary to know the mapping $\Phi(x)$. Instead, the required scalar products in a feature space $\Phi^T(x_i)\Phi\left(x_j\right)$ are calculated directly by computing kernels for given training data vectors in an input space. By using kernels, a KSVM can be constructed that operates in an infinite dimensional space and the extremely high dimensionality of a feature space F is avoided. In general, a kernel is a function K such that

$$K\left(x_i, x_j\right) = \Phi^T(x_i)\Phi\left(x_j\right). \tag{2.18}$$

However, there are many different kernels to choose from and, therefore, this paper uses the Gaussian radial basis function kernel of the following form:

$$K(x, x_i) = e^{\frac{1}{2}\left[(x-x_i)^T\Sigma^{-1}(x-x_i)\right]}, \tag{2.19}$$

which is a general purpose kernel and is typically chosen when no prior knowledge is available about the data.

2.2. Benchmark models

In this paper, the recession probabilities predicted by the Super Learner are compared with a total of eight different benchmark models. The benchmark models are all based on standard univariate and multivariate probit models of the following form:

$$\Pr(Y_{t+h} = 1 | X_t = x_{i,t}) = \Phi\left(\beta_0 + \sum_{i=1}^{k} \beta_i x_{i,t}\right), \tag{2.20}$$

where the dependent variable Y_{t+h} is the binary NBER recession indicator, h is the forecast horizon, Φ is the cumulative standard normal distribution function, and $x_{i,t}$ are up to k predictor variables.

The predictor variables are different across all the benchmark models. In four of the benchmark models, the predictor variables are nowcasts and forecasts of GDP growth. These nowcasts and forecasts are generated by two machine learning methods—to be precise a MLP and an ELM which are both types of artificial neural networks (ANNs)—by a standard DFM and taken from the SPF. Based on the research by Liu and Moench (2016), the remaining four benchmark models are multivariate probit models, where the term spread and its 6-month lag are combined with different economic and financial indicators as predictor variables. The term spread is defined in this paper as the difference between the ten-year and three-month Treasury yields.

A list of the benchmark probit models and a brief explanation of each approach is presented in the following:

1. Probit: MLP
   Following the approach described in Loermann and Maas (2019), nowcasts and forecasts of GDP growth are first obtained by a MLP, which is a special kind of a feedforward ANN. This network is trained on a large database via the gradient-based learning algorithm called "backpropagation" (Rumelhart et al., 1986) and can be best interpreted as a flexible and highly parametrised non-linear autoregressive distributed lag model (ARDL) model.[5] The estimated GDP values then go into the univariate probit model as described in Equation (2.20) to yield the recession probabilities for the different horizons.

2. Probit: ELM
   The procedure is the same as under 1, but the feedforward ANN is an ELM based on Huang et al. (2006). In contrast to a MLP, an ELM does not use the slow gradient-based learning algorithm backpropagation to tune parameters of the hidden nodes. The output weights of the hidden nodes are randomly chosen and, therefore, are learned in a single step, so that the ELM learns faster as the MLP. The nowcasts and forecasts of GDP values

[5]For details about the MLP, see Crone and Kourentzes (2010), Kourentzes et al. (2014) and Lachtermacher and Fuller (1995).

obtained from this ELM then go into the univariate probit model from Equation (2.20).

3. Probit: DFM
   First, the GDP growth values for the current quarter and for future quarters are estimated from a large database of potential predictor variables by a DFM based on Giannone et al. (2008), as done in Loermann and Maas (2019). Then, the estimated values are used in the probit model to get the recession probabilities.

4. Probit: SPF
   This benchmark model uses the nowcasts and forecasts of GDP growth published by the SPF, which are then used as predictor variables in the probit model. The Survey publishes its nowcast and forecasts of future GDP growth around the mid of every quarter.[6]

5. Probit: term spread + Emp: total
   This benchmark model is a multivariate probit model, where the predictor variables are the term spread, its 6-month lag and all nonfarm payroll employees.

6. Probit: term spread + S&P 500
   Beside the term spread and its 6-month lag, returns on the S&P 500 common stock price index are included as covariate in the probit model.

7. Probit: term spread + 1yr spread
   In addition to the term spread and its 6-month lag, this benchmark includes as an additional variable in the probit model the spread between the yield of one-year constant maturity Treasuries and the federal funds rate.

8. Probit: term spread + 5yr spread
   The procedure is the same as under 7, but the spread between the yield of five-year constant maturity Treasuries and the federal funds rate is used as an additional covariate next to the term spread and its lagged value.

2.3. Evaluating the models

To measure which model has the best ability to nowcast and forecast recessions, the ROC curve and its corresponding AUROC value are used in this paper. The basic ROC methodology was first introduced by Peterson et al. (1954), but has recently found its way into Economics.[7]

[6]For details about the SPF, see Croushore (1993).

[7]For applications of the ROC in Economics, see for example Jordà and Taylor (2011, 2012), Khandani et al. (2010), Liu and Moench (2016), and Pierdzioch et al. (2018).

The basic ROC methodology in the context of predicting recessions is summarised in the following, while the notation is adapted from Liu and Moench (2016):

1. Let
$$Z_t = \begin{cases} 1, & \text{if in recession} \\ 0, & \text{otherwise,} \end{cases} \tag{2.21}$$
denote the true, observed state of the economy. Let P_t be the prediction of Z_t—the recession probability—where $0 \leq P_t \leq 1$.

2. Define evenly spaced thresholds, denoted as C^*, along the interval $[0,1]$. For example, a set with 20 thresholds would be $C^* = \{0, 0.05, \ldots, 0.95, 1\}$.

3. For each given threshold C_i^*, record the model's predicted categories. Define the predicted categorisation $\widehat{Z}_t$ as follows:
$$\widehat{Z}_t = \begin{cases} 1, & \text{if } P_t \geq C_i^* \\ 0, & \text{if } P_t < C_i^*. \end{cases} \tag{2.22}$$

4. Comparing the true Z_t with the predicted categorisations $\widehat{Z}_t$, calculate the percentages of true positives (PTP) and the percentages of false positives (PFP).[8] Both fractions can be defined using the sum of two indicator variables:
$$\begin{aligned} PTP_i &= \frac{1}{n_R} \sum_{t=1}^{T} I_t^{tp}; \\ \text{where } I_t^{tp} &= \begin{cases} 1, & \text{if } Z_t = 1 \text{ and } \widehat{Z}_t = 1 \\ 0, & \text{otherwise,} \end{cases} \end{aligned} \tag{2.23}$$
$$\begin{aligned} PFP_i &= \frac{1}{n_E} \sum_{t=1}^{T} I_t^{fp}; \\ \text{where } I_t^{fp} &= \begin{cases} 1, & \text{if } Z_t = 0 \text{ and } \widehat{Z}_t = 1 \\ 0, & \text{otherwise,} \end{cases} \end{aligned} \tag{2.24}$$
where n_R is the number of times the true Z_t was in a recession and n_E is the number of times the true Z_t was not in a recession, such that $n_R + n_E = T$, where T is the total number of observations in the sample.

[8] In the empirical literature, the percentages of true positives (PTP) is also called 'sensitivity' and the percentages of false positives (PFP) is also called 'one minus sensitivity'.

5. For each C_i^* create a set of coordinates: (PFP_i, PTP_i).

6. After a coordinate is created for each threshold, we plot the coordinates across all thresholds, with the false positive rate on the x-axis and the true positive rate on the y-axis. We then connect these coordinates to trace out the ROC curve.

In summary, a model with 100 % accuracy would have a ROC curve that covers the upper left-hand corner. A model which is equivalent to a random guess would follow a 45° diagonal running from the bottom left-hand to the top right-hand corner.

Because it is visually very difficult to recognise which ROC curve gives the best overall predictive ability out of a set of different ROC curves, the curves are integrated and the resulting area under receiver operating characteristic (AUROC) curve is then compared. An AUROC value of one means that the model perfectly classifies a recession, where a value of 0.5 is equivalent to a random guess. In the empirical analysis in Section 4, the recession classification ability of the models will be compared by their implied AUROC values.

Given that the residuals from classification models that use a recession indicator as the dependent variable are likely to be autocorrelated, inference on the classification ability using the AUROC is given by the block bootstrap approach of Politis and Romano (1994), as done in Liu and Moench (2016) and Pierdzioch et al. (2018). The block bootstrap is implemented with 1000 simulation runs and—following Liu and Moench (2016)—a block length of eight years is used to retain the typical business cycle length. This procedure creates a comparable empirical distribution of AUROC values for each model and each forecast horizon.

3. Data

Before turning to the results, it is important to illustrate the data, particularly the challenges of real-time now- and forecasting.

Machine learning algorithms usually deal with large amounts of high-frequency data. In macroeconomics, important indicators such as GDP are collected only quarterly, while unemployment statistics and inflation rates are collected on a monthly basis. Furthermore, many indicators are published only with a time lag and are susceptible to revisions. These issues create many challenges for real-time nowcasting and forecasting of recessions. Therefore, most of the probit models used in this paper use quarterly GDP predictions as input variables, so the final recession probability of all prediction models in this paper is limited to the quarterly frequency.

Beginning with the introduction of the data, it can be noted that this study uses macroeconomic data as provided by the Federal Reserve Economic Data (FRED) database. To be precise, the data behind the DFM, the MLP, the ELM and the Super Learner is the same and comes from FRED-MD, the monthly database for Macroeconomic Research of the Federal Reserve Bank of St. Louis, which is described extensively in McCracken and Ng (2016). The time series used in the remaining probit models are also retrieved from FRED but the nowcasts and forecasts of the SPF are obtained from the Federal Reserve Bank of Philadelphia.[9]

In brief, FRED-MD is a large macroeconomic database that is designed for the empirical analysis of "big data". The database is publicly available and updated in real-time on a monthly basis.[10] It consists of 134 monthly time series and is classified into eight categories: (1) output and income; (2) labor market; (3) housing; (4) consumption, orders and inventories; (5) money and credit; (6) interest and exchange rates; (7) prices; and (8) stock market. A full list of the data and its transformation is given in Appendix A.2. The time series start in January 1959 and vintages of the whole database are available since August 1999.

Before training the Super Learner, the time series contained in the vintage of the FRED-MD database are transformed to be stationary, outliers are removed, and missing values are replaced by the expectation-maximization (EM) algorithm; however, the ragged edge structure at the end of the sample is still unchanged. Because the NBER recession dates are quarterly, the monthly time series need to be transformed into quarterly equivalents. This paper follows Giannone et al. (2008), who use a rational transfer function for this purpose.[11] To deal with the ragged edge problem at the end of the sample, these missing values are filled up by applying univariate $ARMA(p;q)$ forecasts of each single time series, where the lag lengths are selected via Akaike information criterion (AIC). This procedure is also done for the benchmark models. Then, considering the training of the Super Learner, the data is scaled on the interval $[0,1]$ and split into five folds for cross-validation.

When it comes to training the Super Learner algorithm and estimating the probit models, the treatment of the NBER recession dates is of interest. Because the NBER Business Cycle Dating Committee detects a recession with a certain delay[12], all models are trained with a delay of four

[9]The SPF can be downloaded from the following link: https://www.philadelphiafed.org/research-and-data/real-time-center/survey-of-professional-forecasters/.

[10]The FRED-MD database is available for download under the following link: https://research.stlouisfed.org/econ/mccracken/fred-databases/.

[11]Giannone et al. (2008) use the following rational transfer function to transform the data into a quarterly equivalent: $Y(z) = \frac{b(1)+b(2)z^{-1}+\cdots+b(n_b+1)z^{-n_b}}{1+a(2)z^{-1}+\cdots+a(n_a+1)z^{-n_a}} X(z)$, with $[a,b] = [1,(1,2,3,2,1)]$. For further details, see the Appendix of Giannone et al. (2008).

[12]The NBER Business Cycle Dating Committee states that "there is no fixed timing rule" when it comes to a determination of a recession. "The committee waits long enough so that the existence of a peak or trough is not in doubt, and until it can assign an accurate peak or trough date." See https://www.nber.org/cycles/

quarters before the final forecast is produced with the latest and most up-to-date data from the last update of the database. Furthermore, because the recession probability is to be predicted for the current quarter and the following four quarters, the NBER recession date in t is linked to the predictor variables in $t-h$, where h stands for the horizons $h = 0, 1, 2, 3, 4$.

The nowcasts and forecasts of the recession probabilities are conducted at the end of the second month of each quarter. To allow the Super Learner and also the probit models to learn and adapt from new data, all models are retrained in the next period after an update of the respective input data.

4. Empirical results

This section presents the empirical results. The recession probabilities obtained from the Super Learner algorithm are compared with the results of the different probit models for the different forecast horizons using the respective AUROC value. The higher the AUROC, the better the ability to determine recessions, whereby the AUROC value cannot exceed a value of one. For each model, the standard errors obtained from the block bootstrap approach after Politis and Romano (1994) are also reported.

Due to data transformation and the lagged behavior of the NBER recession dates, the training sample of the first out-of-sample nowcasting and forecasting exercise starts in 1959Q3 and goes on until 1998Q2, which covers a total of six recessions after the NBER recession indicator. The first nowcast is then conducted for 1999Q3 and the first out-of-sample forecasts are made for the following four quarters. The last nowcast is made in 2019Q1. In addition, it is shown which machine learning methods the Super Learner is composed of at the individual points in time per prediction horizon.

In this empirical task, the Super Learner consists of a total of 1289 models per forecast horizon. The high number of different models results from a multitude of different tuning parameters with which the individual four machine learning methods are trained before making the nowcasts and forecasts.[13]

Table 1 shows the empirical out-of-sample results. For horizon $h = 0$—the nowcast—the SPF reaches the highest AUROC value of 0.9937, followed by the Super Learner and the 'Probit:

`recessions_faq.html` for further information.

[13] As far as tuning parameters are concerned, random forests are trained with different number of trees = $[100, 150, \ldots, 1000]$. Gradient boosting machines are trained with number of trees = $[100, 200, \ldots, 1000]$, depth of the tree = $[1, 2, \ldots, 10]$, shrinkage parameter = $[0.001, 0.005, 0.01, 0.1]$, and minimum observations allowed per tree node = $[1, 3, 5]$. Kernel support vector machines are trained with the following tuning parameters: 'C'-constant = $\left[2^{-5}, 2^{-3}, \ldots, 2^{15}\right]$ and 'nu' parameter = $[0.001, 0.01, 0.1, 0.2, 0.5, 1]$.

DFM'-model, both with a value of 0.9810. All alternative models have a value above 0.94, which highlights that they are all very successful in nowcasting a recession in the current quarter.

When estimating the probability of a recession for future quarters, the AUROC value of the Super Learner decreases. For horizon $h = 1$, the Super Learner's AUROC of 0.9614 is only beaten by the 'Probit: term spread + S&P 500'-model with a value of 0.9646. For horizon $h = 2$, the Super Learner is beaten by 'Probit: term spread + Emp: total', 'Probit: term spread + S&P 500', 'Probit: term spread + 1yr spread', and 'Probit: term spread + 5yr spread'. For the longer horizons $h = 3$ and $h = 4$, the Super Learner clearly outperforms the MLP, the ELM, the DFM, and the SPF, which highlights the limited power of these approaches for detecting recessions in the long-term. However, the Super Learner is considerably beaten by the probit models using the one-year Treasury spread and the five-year Treasury spread as additional predictors. The probit model with the five-year Treasury spread has the overall highest AUROC value for the longer forecast horizons $h = 2$, $h = 3$, and $h = 4$, which illustrates the strong predictive power of the yield curve for the sample. However, since the financial crisis of 2007/08, the central banks have intervened significantly in the bond market through extensive asset purchase programmes, so that the credibility and thus also the forecasting ability of the yield curve can be questioned since that time. In contrast, the Super Learner algorithm used in this article is not affected by this problem because its predictions are based on the use of much larger and more diverse datasets. The Super Learner can therefore be described as a kind of all-rounder that can rely on large databases and which works well across all forecast horizons.

Table 1: Out-of-sample summary of AUROC values.

Model	$h=0$	$h=1$	$h=2$	$h=3$	$h=4$
Super Learner	0.9810	0.9614	0.8268	0.8342	0.7778
	(0.1130)	(0.1161)	(0.1006)	(0.1034)	(0.1012)
Probit: MLP	0.9444	0.9179	0.7843	0.5207	0.1211
	(0.1170)	(0.1105)	(0.1004)	(0.0902)	(0.1241)
Probit: ELM	0.9444	0.9147	0.7810	0.5688	0.1924
	(0.1122)	(0.1075)	(0.1032)	(0.0988)	(0.1211)
Probit: DFM	0.9810	0.9597	0.9150	0.7612	0.1410
	(0.1149)	(0.1175)	(0.1134)	(0.1016)	(0.1105)
Probit: SPF	**0.9937**	0.8969	0.6111	0.3566	0.1260
	(0.1111)	(0.1144)	(0.1068)	(0.0984)	(0.1263)
Probit: term spread + Emp: total	0.9365	0.8824	0.8497	0.8507	0.8259
	(0.1191)	(0.1260)	(0.1244)	(0.1179)	(0.1163)
Probit: term spread + S&P 500	0.9730	**0.9646**	0.9150	0.8192	0.7877
	(0.1146)	(0.1160)	(0.1188)	(0.1255)	(0.1260)
Probit: term spread + 1yr spread	0.9587	0.9452	0.9248	0.9088	0.8458
	(0.1311)	(0.1227)	(0.1218)	(0.1289)	(0.1334)
Probit: term spread + 5yr spread	0.9444	0.9356	**0.9461**	**0.9254**	**0.8690**
	(0.1259)	(0.1194)	(0.1185)	(0.1227)	(0.1309)

Notes: This table shows the out-of-sample AUROC values for the sample period 1999Q3 - 2019Q1. The numbers in bold correspond to the highest AUROC value at each forecast horizon *h*. The corresponding standard errors are reported in parentheses. Standard errors are computed based on the bootstrapped sampling distribution of the AUROC statistic. The bootstrap was implemented through the block bootstrap approach of Politis and Romano (1994) with 1000 simulation runs and a block length of eight years.

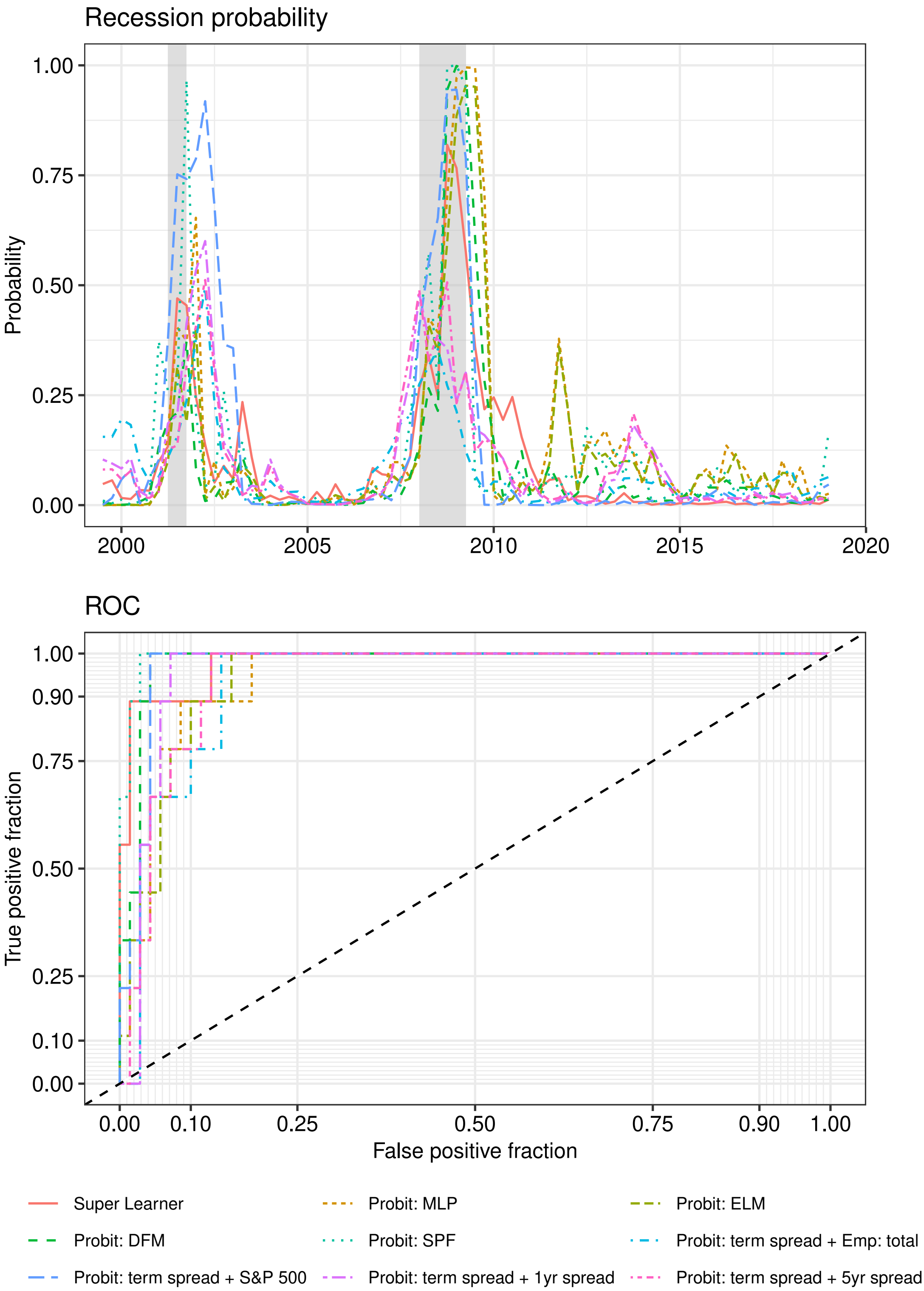
Figures 1-5 show the plotted recession probabilities and the corresponding ROC curves. All of the figures show that the Super Learner announces periods of economic turmoil in quite good time. In particular, it indicates an increased recession probability for all forecast horizons in the run-up to the 2008/09 recession. Especially for the horizons $h = 3$ and $h = 4$, the Super Learner shows an increased recession probability in the period leading up to the Great Recession when compared to the benchmark models, highlighting the strength of its use of a larger dataset and different underlying powerful classification tools. However, the recession probability determined for these horizons is more volatile than for the multivariate probit models, resulting in a lower AUROC value in each case.

It is of note that for the shorter forecast horizons $h = 0$ and $h = 1$, during the two NBER recessions the recession probability determined by the Super Learner is lower than that of some benchmarks. Nevertheless, considering these horizons, the Super Learner seems best to announce the approaching end of a recession because the probability of a recession decreases more strongly towards the end of each recession than most other benchmark models.

The plotted ROC curves visualise the AUROC results. Overall, they show that it is harder to predict recessions for the individual models with an increased forecast horizon because the ROC curves slide further and further in the direction of the lower right-hand corner, where at horizon $h = 4$ the ‘Probit: DFM’, ‘Probit: MLP’, ‘Probit: SPF’ and ‘Probit: ELM’ models perform clearly worse than a random guess.

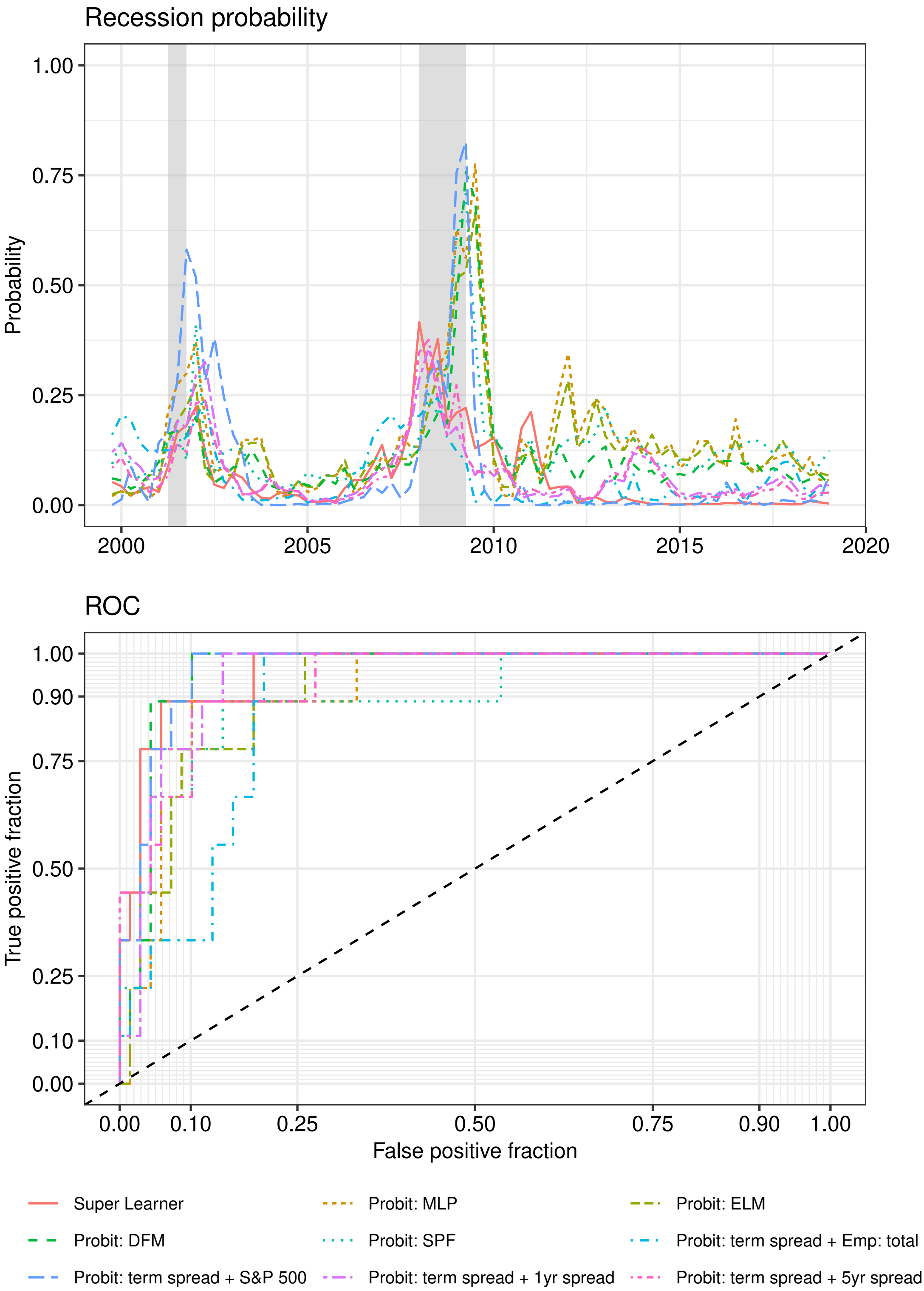
To gain further insight into which machine learning methods the Super Learner is composed of in each period—either elastic net, random forests, gradient boosting machines or kernel support vector machines—, the proportions of the four different methods are shown visually in the Figures 6-10 in Appendix A.1 for each forecast horizon. It is of note that at horizon $h = 0$ after 2008 the Super Learner consists only of gradient boosting machines and kernel support vector machines. Random forests are only used at the beginning of the sample and since 2004 elastic net has also been used. However, as the prediction horizon widens, the proportion of random forests increases, especially at the beginning of each sample. At horizon $h = 4$, this scheme is interrupted. Here, the image is quite mixed, with the gradient boosting machines making up the largest part, especially in the middle of the sample. In the last years of the sample, the Super Learner also consists of kernel support vector machines, underlining the high model flexibility of this ensemble learner.

Figure 1: Forecast horizon $h = 0$.



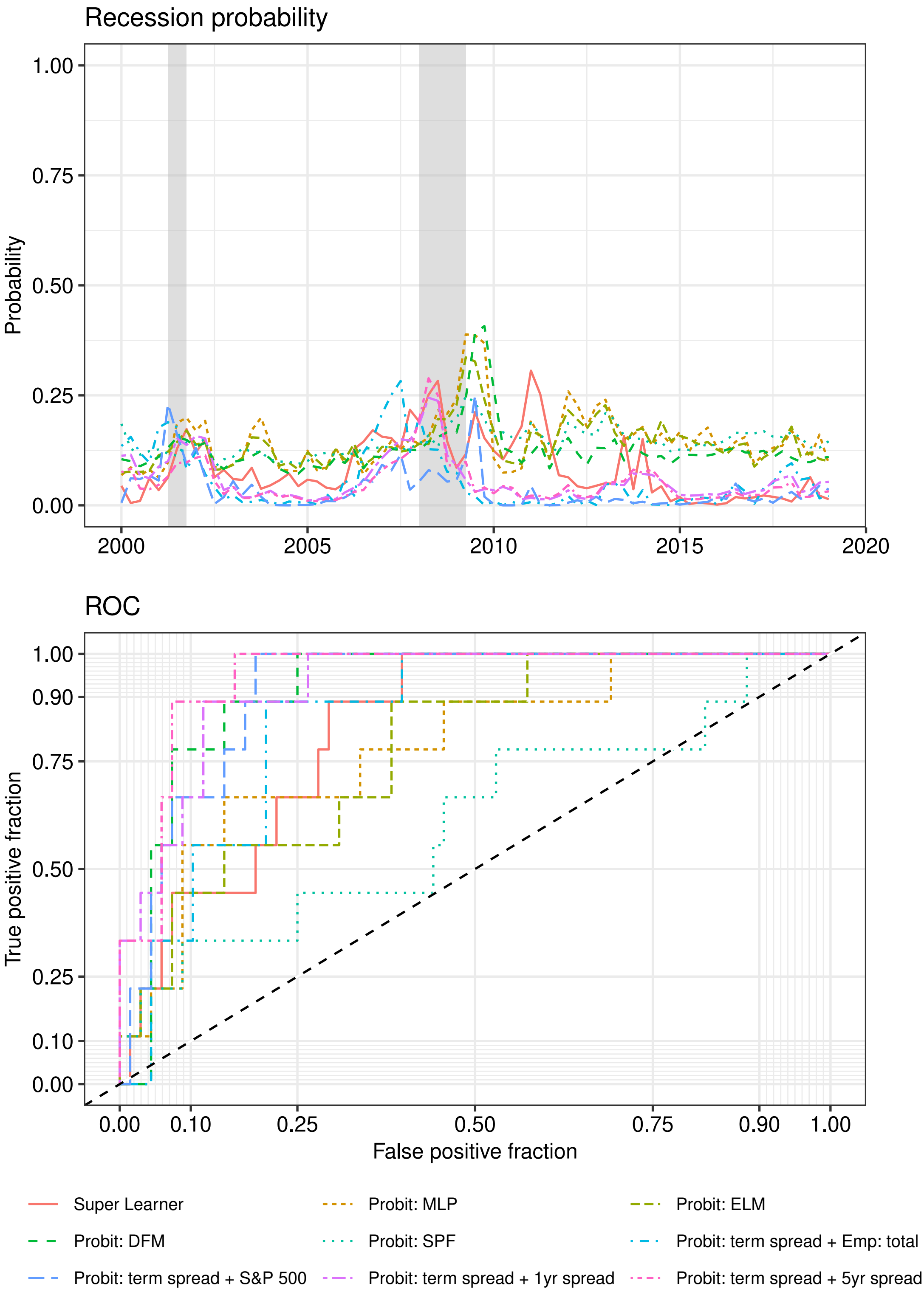
Notes: NBER recessions are highlighted by gray shading.

Figure 2: Forecast horizon $h = 1$.



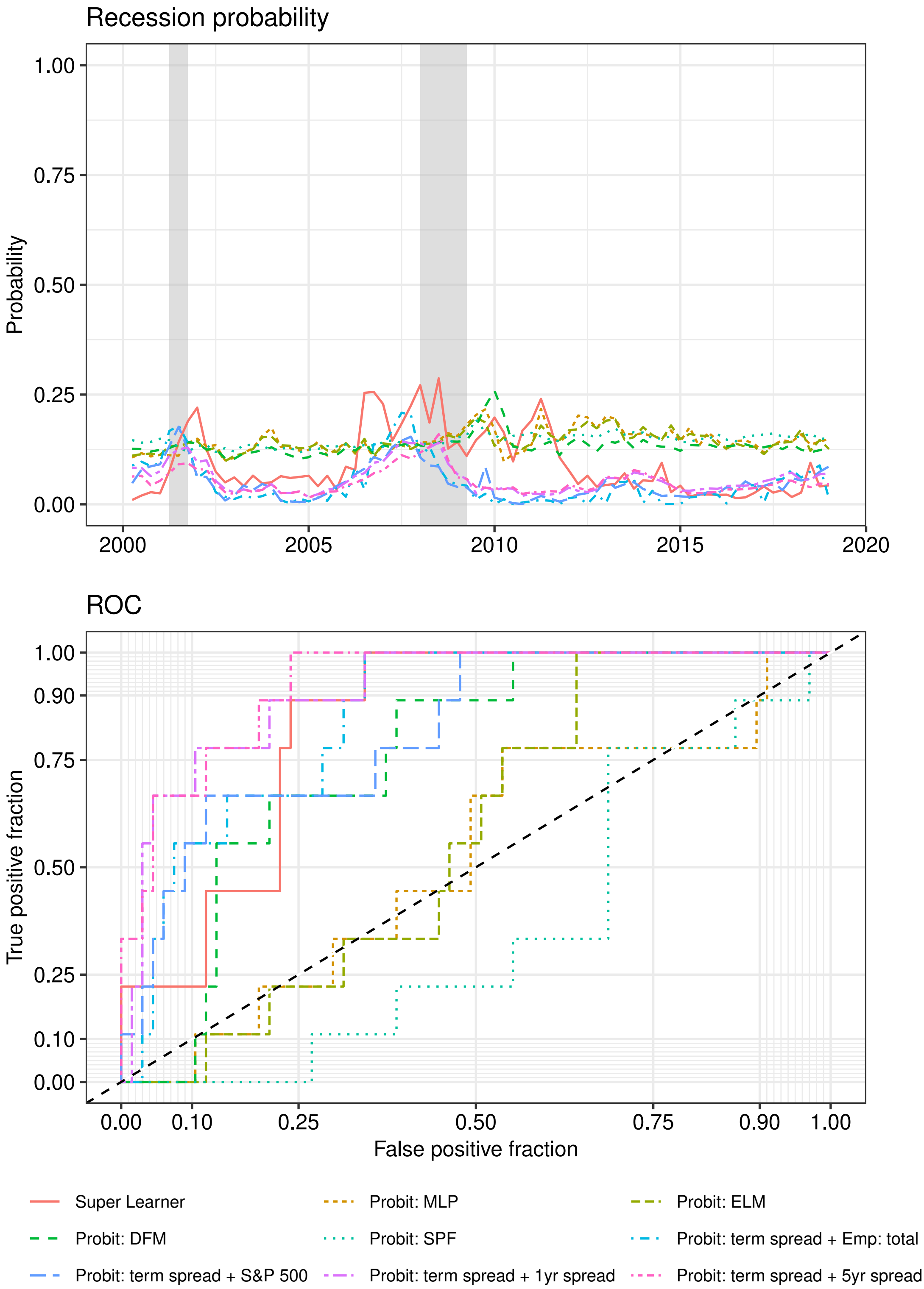
Notes: NBER recessions are highlighted by gray shading.

Figure 3: Forecast horizon $h = 2$.



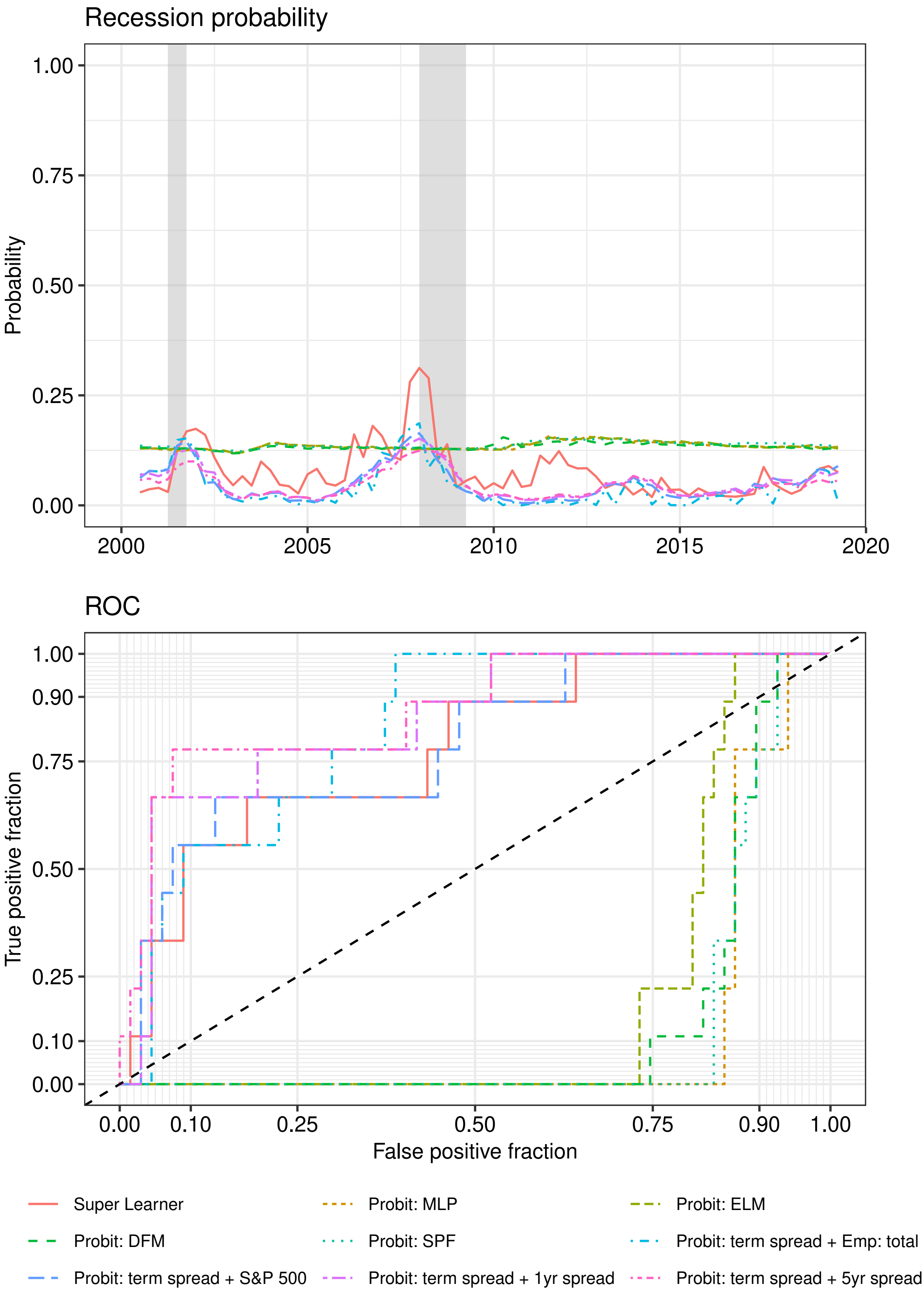
Notes: NBER recessions are highlighted by gray shading.

Figure 4: Forecast horizon $h = 3$.



Notes: NBER recessions are highlighted by gray shading.

Figure 5: Forecast horizon $h = 4$.



Notes: NBER recessions are highlighted by gray shading.

5. Conclusion

This paper applies the Super Learner algorithm to nowcast and forecast the probability of a recession in the US economy for the current quarter and for future quarters. The Super Learner is composed of four machine learning algorithms—namely, elastic net, random forests, gradient boosting machines and kernel support vector machines—and is trained with real-time vintages of the FRED-MD database. The obtained recession probabilities are compared with those of a total of eight benchmark models based on univariate and multivariate probit models. In four of the probit models, nowcasts and forecasts of GDP growth in the current quarter and the following quarters are the only predictor variables, while in the remaining four the term spread, its 6-month lag and an additional economic or financial indicator variable are incorporated as additional predictors. To measure which model overall has the best ability to predict recessions across all horizons, this article uses the ROC curve and the corresponding AUROC. The nowcasts and forecasts are conducted in real-time and are made at the end of the second month of each quarter.

In summary, the Super Learner presented in this paper can be described as a kind of all-rounder that can rely on large databases and which works well across all forecast horizons. When nowcasting a recession in the current quarter, all models are very successful in nowcasting a recession and the Super Learner is only beaten by the probit model with GDP nowcast published by the SPF as predictor variable. For the longer forecast horizons, the probit models including the term spread yield the highest AUROC values, where the model with the spread between the yield of five-year constant maturity Treasuries and the federal funds rate as additional predictor variable reaches the overall best results for horizon $h = 2$, $h = 3$, and $h = 4$.

However, since the financial crisis of 2007/08, central banks have intervened significantly in the bond market through extensive asset purchase programmes and, therefore, the credibility and the forecasting ability of the yield curve can be questioned since that time. In contrast, the Super Learner algorithm used in this article is not affected by this problem because its predictions are based on the use of much larger and more diverse datasets.

It will be interesting to see if the recession forecasts of the Super Learner will improve in the future if more data should become available for training, especially in times of economic uncertainty. In addition, the Super Learner can easily be extended by further classification models, so that the recession probabilities can be determined on the basis of further powerful models. In addition, the recession probabilities can also be determined in monthly frequency by the Super Learner. However, a larger computing capacity is required for the calculations because training the algorithm is very computational and time-consuming. This is left for further research.

References

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

Breiman, L. (1996b). Stacked regressions. *Machine Learning*, 24(1):49–64.

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

Bühlmann, P., Drineas, P., Kane, M., and van der Laan, M. J. (2016). *Handbook of Big Data*. CRC Press.

Crone, S. F. and Kourentzes, N. (2010). Feature selection for time series prediction–a combined filter and wrapper approach for neural networks. *Neurocomputing*, 73(10-12):1923–1936.

Croushore, D. D. (1993). Introducing: the Survey of Professional Forecasters. *Business Review-Federal Reserve Bank of Philadelphia*, 6:3–15.

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

Dudoit, S. and van der Laan, M. J. (2005). Asymptotics of cross-validated risk estimation in estimator selection and performance assessment. *Statistical Methodology*, 2(2):131–154.

Efron, B. and Hastie, T. (2016). *Computer Age Statistical Inference*, volume 5. Cambridge University Press.

Estrella, A. and Hardouvelis, G. A. (1991). The term structure as a predictor of real economic activity. *The Journal of Finance*, 46(2):555–576.

Estrella, A. and Mishkin, F. S. (1996). The yield curve as a predictor of US recessions. *Current Issues in Economics and Finance*, 2(7).

Estrella, A. and Mishkin, F. S. (1998). Predicting US recessions: financial variables as leading indicators. *Review of Economics and Statistics*, 80(1):45–61.

Freund, Y. (1995). Boosting a weak learning algorithm by majority. *Information and Computation*, 121(2):256–285.

Freund, Y. and Schapire, R. E. (1997). A decision-theoretic generalization of on-line learning and an application to boosting. *Journal of Computer and System Sciences*, 55(1):119–139.

Freund, Y. and Schapire, R. E. (1999). Adaptive game playing using multiplicative weights. *Games and Economic Behavior*, 29(1-2):79–103.

Friedman, J., Hastie, T., and Tibshirani, R. (2010). Regularization paths for generalized linear models via coordinate descent. *Journal of Statistical Software*, 33(1):1–22.

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

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

Giannone, D., Reichlin, L., and Small, D. (2008). Nowcasting: the real-time informational content of macroeconomic data. *Journal of Monetary Economics*, 55(4):665–676.

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

Huang, G.-B., Zhu, Q.-Y., and Siew, C.-K. (2006). Extreme learning machine: theory and applications. *Neurocomputing*, 70(1-3):489–501.

Jordà, Ò. and Taylor, A. M. (2011). Performance evaluation of zero net-investment strategies. Technical report, National Bureau of Economic Research.

Jordà, Ò. and Taylor, A. M. (2012). The carry trade and fundamentals: Nothing to fear but feer itself. *Journal of International Economics*, 88(1):74–90.

Jung, J.-K., Patnam, M., and Ter-Martirosyan, A. (2018). An algorithmic crystal ball: Forecasts-based on machine learning. IMF Working Paper 230.

Kecman, V. (2005). Support vector machines: an introduction. In *Support vector machines: theory and applications*, pages 1–47. Springer.

Khandani, A. E., Kim, A. J., and Lo, A. W. (2010). Consumer credit-risk models via machine-learning algorithms. *Journal of Banking & Finance*, 34(11):2767–2787.

Kourentzes, N., Barrow, D. K., and Crone, S. F. (2014). Neural network ensemble operators for time series forecasting. *Expert Systems with Applications*, 41(9):4235–4244.

Kuhn, M. and Johnson, K. (2013). *Applied Predictive Modeling*, volume 26. Springer.

Lachtermacher, G. and Fuller, J. D. (1995). Back propagation in time-series forecasting. *Journal of Forecasting*, 14(4):381–393.

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

Loermann, J. and Maas, B. (2019). Nowcasting US GDP with artificial neural networks. MPRA Working Paper.

McCracken, M. W. and Ng, S. (2016). FRED-MD: a monthly database for macroeconomic research. *Journal of Business & Economic Statistics*, 34(4):574–589.

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

Peterson, W., Birdsall, T., and Fox, W. (1954). The theory of signal detectability. *Transactions of the IRE professional group on information theory*, 4(4):171–212.

Pierdzioch, C., Reid, M. B., and Gupta, R. (2018). On the directional accuracy of inflation forecasts: evidence from South African survey data. *Journal of Applied Statistics*, 45(5):884–900.

Politis, D. N. and Romano, J. P. (1994). The stationary bootstrap. *Journal of the American Statistical association*, 89(428):1303–1313.

Polley, E. C. and van der Laan, M. J. (2010). Super Learner in prediction. U.C. Berkeley Division of Biostatistics Working Paper Series 226.

Rumelhart, D. E., Hinton, G. E., and Williams, R. J. (1986). Learning representations by back-propagating errors. *Nature*, 323(6088):533.

Schapire, R. E. (1990). The strength of weak learnability. *Machine Learning*, 5(2):197–227.

Schölkopf, B., Smola, A. J., Bach, F., et al. (2002). *Learning with Kernels: Support Vector Machines, Regularization, Optimization, and Beyond*. MIT press.

Stock, J. H. and Watson, M. W. (1989). New indexes of coincident and leading economic indicators. *NBER Macroeconomics Annual*, 4:351–394.

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

Tiffin, A. (2016). Seeing in the dark: a machine-learning approach to nowcasting in Lebanon. IMF Working Paper 56.

van der Laan, M. J., Polley, E. C., and Hubbard, A. E. (2007). Super Learner. *Statistical Applications in Genetics and Molecular Biology*, 6(1).

Vapnik, V. (1998). *Statistical Learning Theory*. Wiley, New York.

Wallis, K. F. (1986). Forecasting with an econometric model: the “ragged edge” problem. *Journal of Forecasting*, 5(1):1–13.

Wolpert, D. H. (1992). Stacked generalization. *Neural Networks*, 5(2):241–259.

Zou, H. and 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.

A. Appendix

A.1. Additional figures

Figure 6: Forecast horizon $h = 0$.

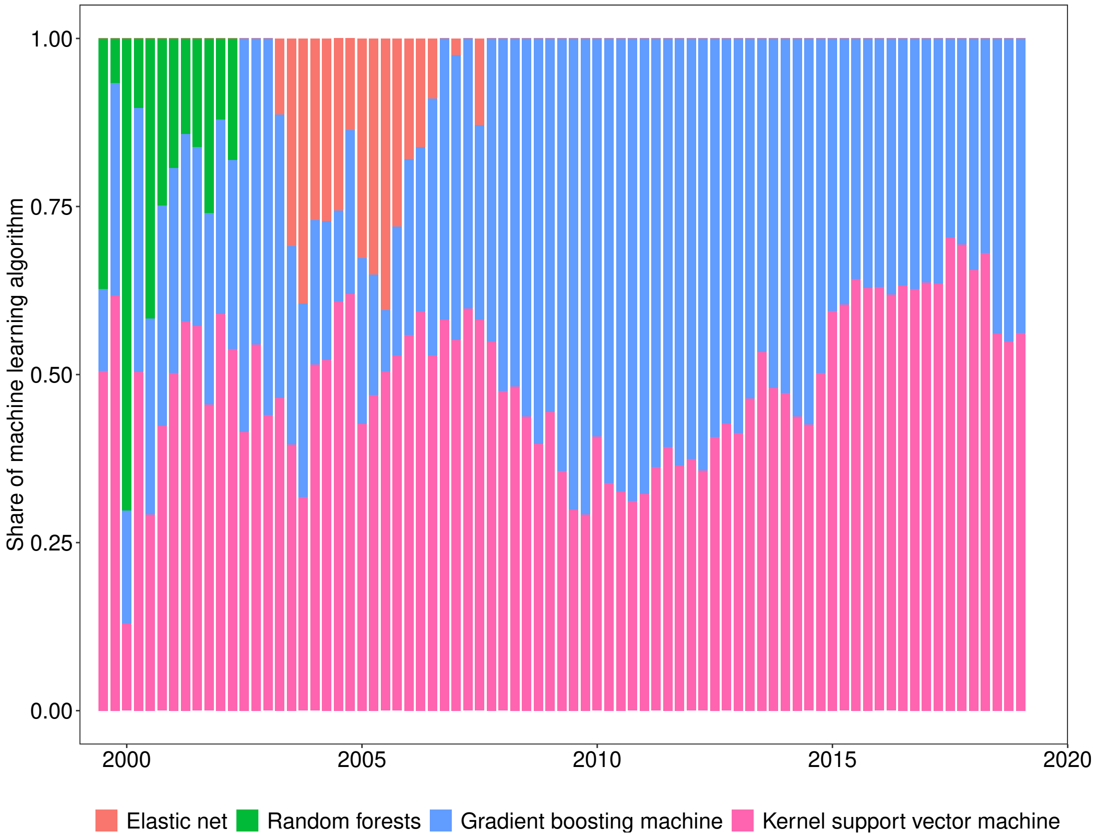


Figure 7: Forecast horizon $h = 1$.

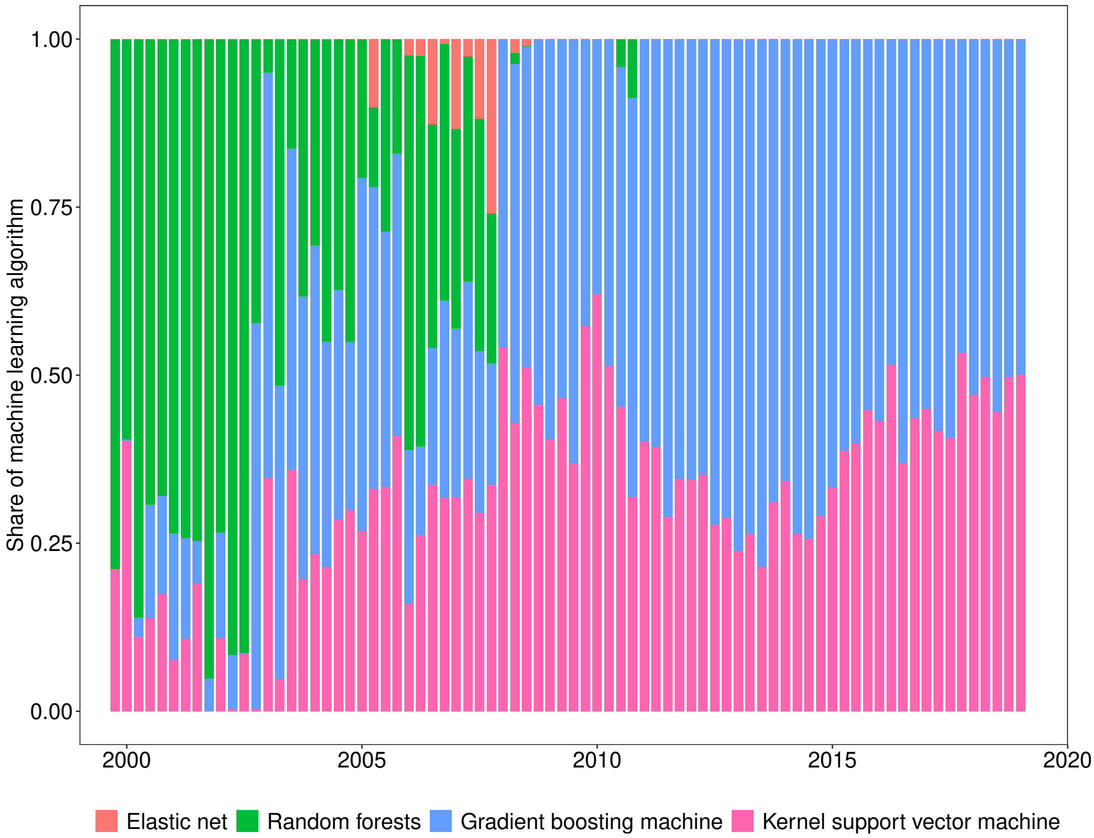


Figure 8: Forecast horizon $h = 2$.

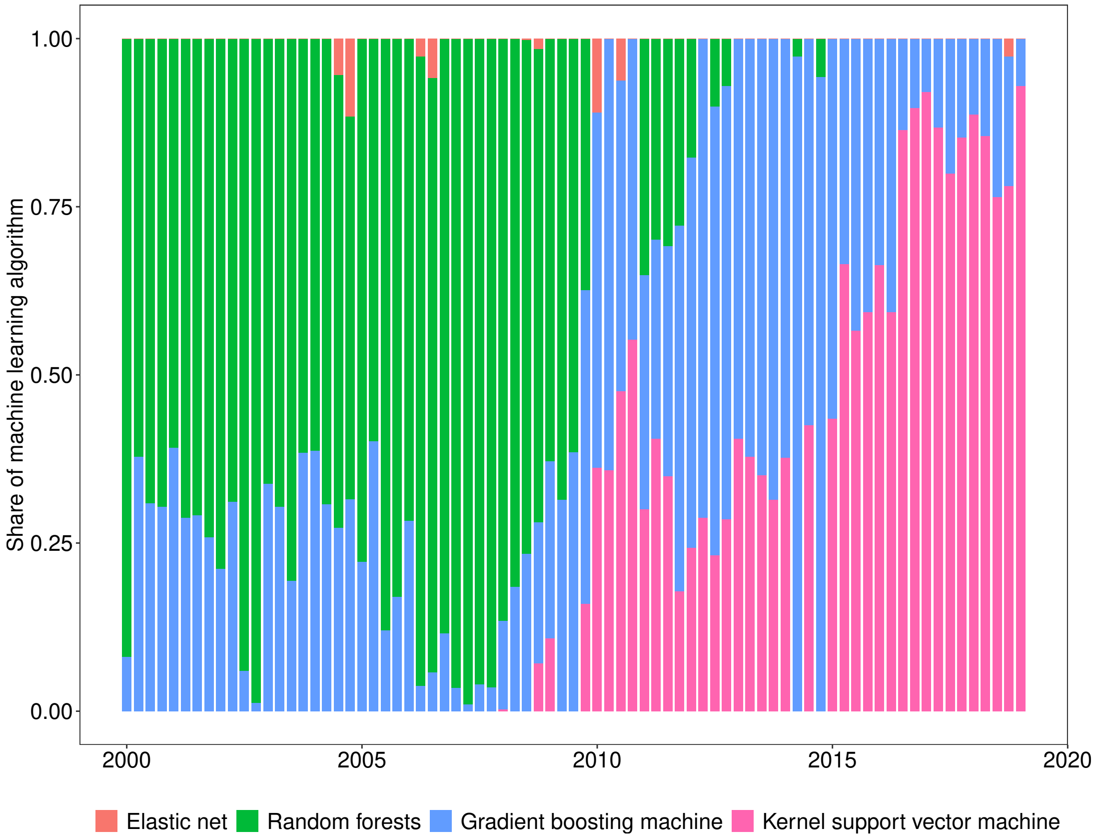


Figure 9: Forecast horizon $h = 3$.

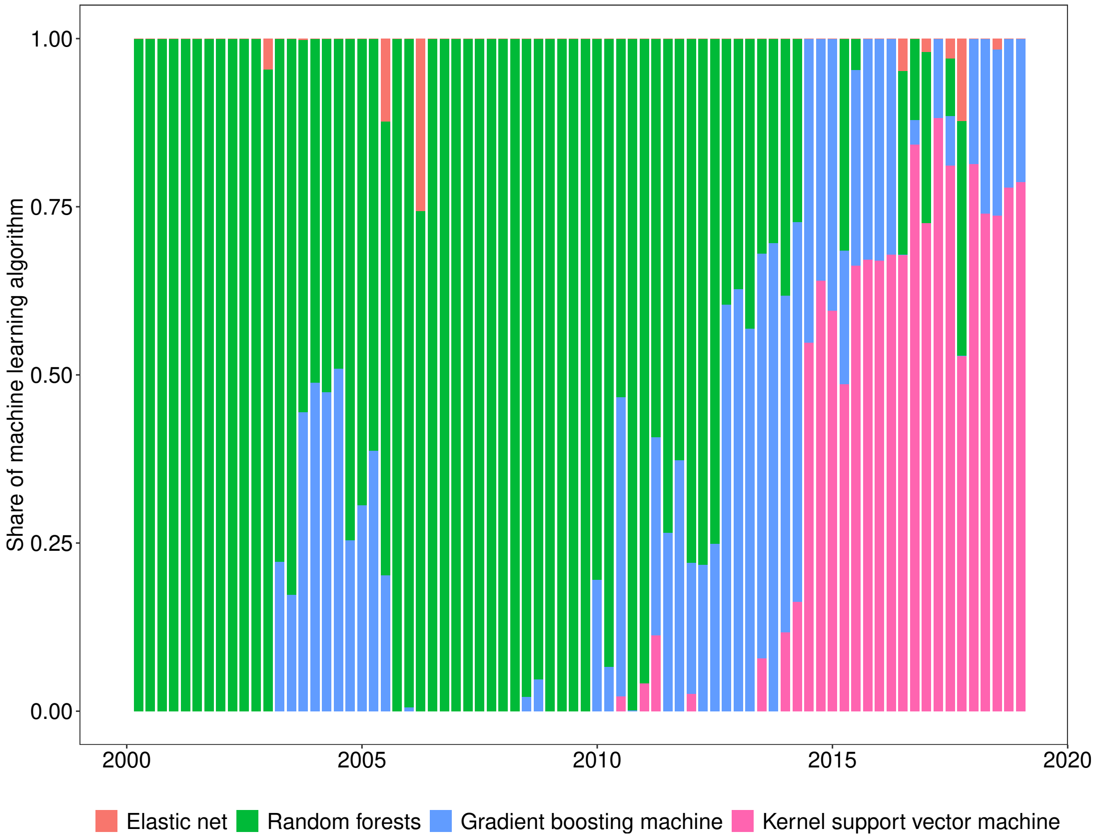
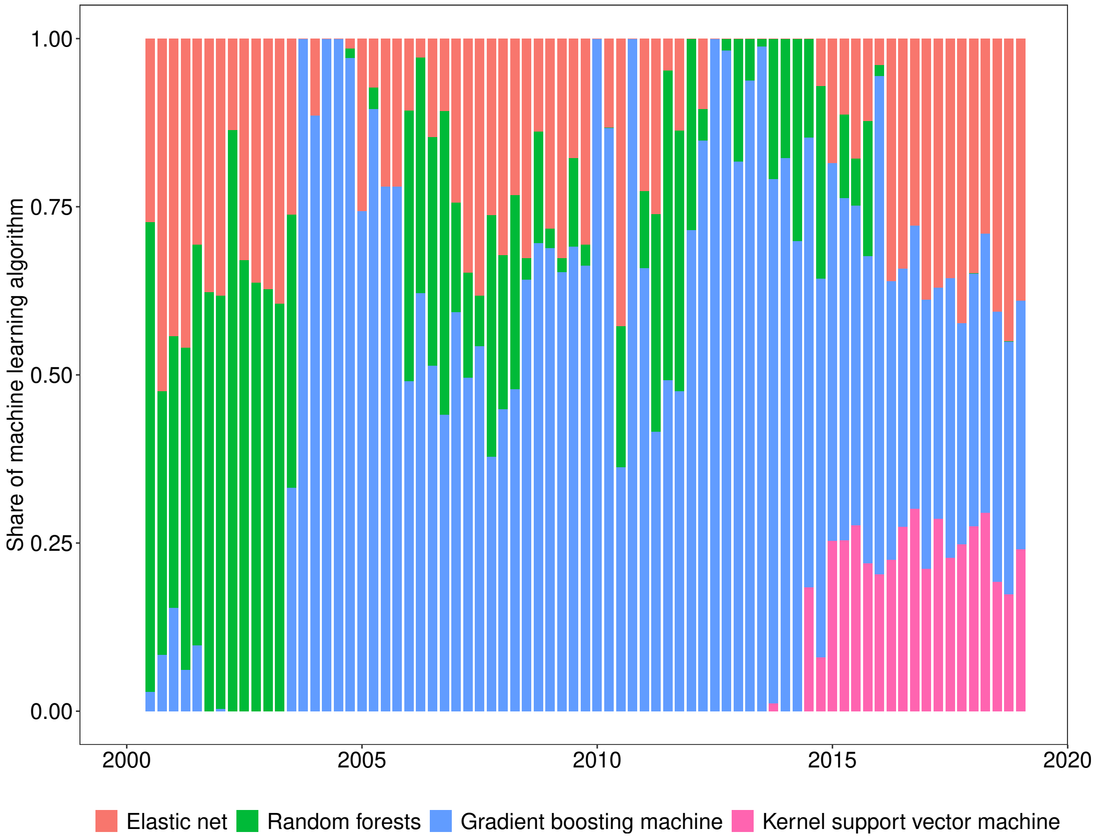


Figure 10: Forecast horizon $h = 4$.



A.2. FRED-MD database

The TCODE column denotes the following data transformation for a series x: (1) no transformation; (2) Δx_t; (3) $\Delta^2 x_t$; (4) $log(x_t)$; (5) $\Delta log(x_t)$; (6) $\Delta^2 log(x_t)$; (7) $\Delta(x_t/x_{t-1} - 1.0)$. The FRED column gives mnemonics in FRED followed by a short description.

Some series require adjustments to the raw data available in FRED. These variables are tagged by an asterisk to indicate that they have been adjusted and thus differ from the series from the source. For a detailed summary of the adjustments see McCracken and Ng (2016).

Group 1. Output and income.

	ID	*tcode*	*FRED*	*Description*
1	1	5	RPI	Real Personal Income
2	2	5	W875RX1	Real personal income ex transfer receipts
3	6	5	INDPRO	IP Index
4	7	5	IPFPNSS	IP: Financial Products and Nonindustrial Supplies
5	8	5	IPFINAL	IP: Final Products (Market Group)
6	9	5	IPCONGD	IP: Consumer Goods
7	10	5	IPDCONGD	IP: Durable Consumer Goods
8	11	5	IPNCONGD	IP: Nondurable Consumer Goods
9	12	5	IPBUSEQ	IP: Business Equipment
10	13	5	IPMAT	IP: Materials
11	14	5	IPDMAT	IP: Durable Materials
12	15	5	IPNMAT	IP: Nondurable Materials
13	16	5	IPMANSICS	IP: Manufacturing (SIC)
14	17	5	IPB51222s	IP: Residential Utilities
15	18	5	IPFUELS	IP: Fuels
16	19	1	NAPMPI	ISM Manufacturing: Production Index
17	20	2	CUMFNS	Capacity Utilization: Manufacturing

Group 2: Labor market.

	ID	*tcode*	*FRED*	*Description*
1	21*	2	HWI	Help-Wanted Index for United States
2	22*	2	HWIURATIO	Ratio of Help Wanted/No. Unemployed
3	23	5	CLF160OV	Civilian Labor Force
4	24	5	CE160V	Civilian Employment
5	25	2	UNRATE	Civilian Unemployment Rate
6	26	2	UEMPMEAN	Average Duration of Unemployment (Weeks)
7	27	5	UEMPLT5	Civilians Unemployed - Less Than 5 Weeks
8	28	5	UEMP5TO14	Civilians Unemployed for 5-14 Weeks
9	29	5	UEMP15OV	Civilians Unemployed - 15 Weeks and Over
10	30	5	UEMP15T26	Civilians Unemployed for 15-26 Weeks
11	31	5	UEMP27OV	Civilians Unemployed for 27 Weeks and Over
12	32*	5	CLAIMSx	Initial Claims
13	33	5	PAYEMS	All Employees: Total nonfarm
14	34	5	USGOOD	All Employees: Goods-Producing Industries
15	35	5	CES1021000001	All Employees: Mining and Logging: Industries
16	36	5	USCONS	All Employees: Construction
17	37	5	MANEMP	All Employees: Manufacturing
18	38	5	DMANEMP	All Employees: Durable Goods
19	39	5	NDMANEMP	All Employees: Nondurable Goods
20	40	5	SRVPRD	All Employees: Service-Providing Industries
21	41	5	USTPU	All Employees: Trade, Transportation and Utilities
22	42	5	USWTRADE	All Employees: Wholesale Trade
23	43	5	USTRADE	All Employees: Retail Trade
24	44	5	USFIRE	All Employees: Financial Activities
25	45	5	USGOVT	All Employees: Government
26	46	1	CES0600000007	Avg Weekly Hours: Goods-Producing
27	47	2	AWOTMAN	Avg Weekly Overtime Hours: Manufacturing
28	48	1	AWHMAN	Avg Weekly Hours: Manufacturing
29	49	1	NAPMEI	ISM Manufacturing: Employment Index
30	127	6	CES0600000008	Avg Hourly Earnings: Goods-Producing
31	128	6	CES2000000008	Avg Hourly Earnings: Construction
32	129	6	CES3000000008	Avg Hourly Earnings: Manufacturing

Group 3: Housing.

	ID	*tcode*	*FRED*	*Description*
1	50	4	HOUST	Housing Starts: Total New Privately Owned
2	51	4	HOUSTNE	Housing Starts: Northeast
3	52	4	HOUSTMW	Housing Starts: Midwest
4	53	4	HOUSTS	Housing Starts: South
5	54	4	HOUSTW	Housing Starts: West
6	55	4	PERMIT	New Private Housing Permits (SAAR)
7	56	4	PERMITNE	New Private Housing Permits: Northeast (SAAR)
8	57	4	PERMITMW	New Private Housing Permits: Midwest (SAAR)
9	58	4	PERMITS	New Private Housing Permits: South (SAAR)
10	59	4	PERMITW	New Private Housing Permits: West (SAAR)

Group 4: Consumption, orders and inventories.

	ID	*tcode*	*FRED*	*Description*
1	3	5	DPCERA3M086SBEA	Real personal consumption expenditures
2	4*	5	CMRMTSPLx	Real Manu. and Trade Industries Sales
3	5*	5	RETAILx	Retail and Food Services Sales
4	60	1	NAPM	ISM: PMI Composite Index
5	61	1	NAPMNOI	ISM: New Orders Index
6	62	1	NAPMSDI	ISM: Supplier Deliveries Index
7	63	1	NAPMII	ISM: Inventories Index
8	64	5	ACOGNO	New Orders for Consumer Goods
9	65*	5	AMDMNOx	New Orders for Durable Goods
10	66*	5	ANDENOx	New Orders for Nondefense Capital Goods
11	67*	5	AMDMUOx	Unfilled Orders for Durable Goods
12	68*	5	BUSINVx	Total Business Inventories
13	69*	2	ISRATIOx	Total Business: Inventories to Sales Ratio
14	130*	2	UMSCENTx	Consumer Sentiment Index

Group 5: Money and credit.

	ID	*tcode*	*FRED*	*Description*
1	70	6	M1SL	M1 Money Stock
2	71	6	M2SL	M2 Money Stock
3	72	5	M2REAL	Real M2 Money Stock
4	73	6	AMBSL	St. Louis Adjusted Monetary Base
5	74	6	TOTRESNS	Total Reserves of Depository Institutions
6	75	7	NONBORRES	Reserves of Depository Institutions
7	76	6	BUSLOANS	Commercial and Industrial Loans
8	77	6	REALLN	Real Estate Loans at All Commercial Banks
9	78	6	NONREVSL	Total Nonrevolving Credit
10	79*	2	CONSPI	Nonrevolving consumer credit to Personal Income
11	131	6	MZMSL	MZM Money Stock
12	132	6	DTCOLNVHFNM	Consumer Motor Vehicle Loans Outstanding
13	133	6	DTCTHFNM	Total Consumer Loans and Leases Outstanding
14	134	6	INVEST	Securities in Bank Credit at All Commercial Banks

Group 6: Interest and exchange rates.

	ID	*tcode*	*FRED*	*Description*
1	84	2	FEDFUNDS	Effective Federal Funds Rate
2	85*	2	CP3Mx	3-Month AA Financial Commercial Paper Rate
3	86	2	TB3MS	3-Month Treasury Bill
4	87	2	TB6MS	6-Month Treasury Bill
5	88	2	GS1	1-Year Treasury Rate
6	89	2	GS5	5-Year Treasury Rate
7	90	2	GS10	10-Year Treasury Rate
8	91	2	AAA	Moody's Seasoned Aaa Corporate Bond Yield
9	92	2	BAA	Moody's Seasoned Baa Corporate Bond Yield
10	93*	1	COMPAPFFx	3-Month Commercial Paper Minus FEDFUNDS
11	94	1	TB3SMFFM	3-Month Treasury C Minus FEDFUNDS
12	95	1	TB6SMFFM	6-Month Treasury C Minus FEDFUNDS
13	96	1	T1YFFM	1-Year Treasury C Minus FEDFUNDS
14	97	1	T5YFFM	5-Year Treasury C Minus FEDFUNDS
15	98	1	T10YFFM	10-Year Treasury C Minus FEDFUNDS
16	99	1	AAAFFM	Moody's Aaa Corporate Bond Minus FEDFUNDS
17	100	1	BAAFFM	Moody's Baa Corporate Bond Minus FEDFUNDS
18	101	5	TWEXMMTH	Trade Weighted U.S. Dollar Index: Major Currencies
19	102*	5	EXSZUSx	Switzerland / U.S. Foreign Exchange Rate
20	103*	5	EXJPUSx	Japan / U.S. Foreign Exchange Rate
21	104*	5	EXUSUKx	U.S. / U.K. Foreign Exchange Rate
22	105*	5	EXCAUSx	Canada / U.S. Foreign Exchange Rate

Group 7: Prices.

	ID	*tcode*	*FRED*	*Description*
1	106	6	WPSFD49207	PPI: Finished Goods
2	107	6	WPSFD49502	PPI: Finished Consumer Goods
3	108	6	WPSID61	PPI: Intermediate Materials
4	109	6	WPSID62	PPI: Crude Materials
5	110*	6	OILPRICEx	Crude Oil, spliced WTI and Cushing
6	111	6	PPICMM	PPI: Metals and metal products
7	112	1	NAPMPRI	ISM Manufacturing: Prices Index
8	113	6	CPIAUCSL	CPI: All Items
9	114	6	CPIAPPSL	CPI: Apparel
10	115	6	CPITRNSL	CPI: Transportation
11	116	6	CPIMEDSL	CPI: Medical Care
12	117	6	CUSR0000SAC	CPI: Commodities
13	118	6	CUSR0000SAD	CPI: Durables
14	119	6	CUSR0000SAS	CPI: Service
15	120	6	CPIULFSL	CPI: All Items less Food
16	121	6	CUSR0000SA0L2	CPI: All Items less Shelter
17	122	6	CUSR0000SA0L5	CPI: All Items less Medical Care
18	123	6	PCEPI	Personal Cons. Expend.: Chain Index
19	124	6	DDURRG3M086SBEA	Personal Cons. Expend.: Durable Goods
20	125	6	DNDGRG3M086SBEA	Personal Cons. Expend.: Nondurable Goods
21	126	6	DSERRG3M086SBEA	Personal Cons. Expend.: Services

Group 8: Stock market.

	ID	*tcode*	*FRED*	*Description*
1	80*	5	S&P 500	S&P's Common Stock Price Index: Composite
2	81*	5	S&P: indust	S&P's Common Stock Price Index: Industrials
3	82*	2	S&P div yield	S&P's Composite Common Stock: Dividend Yield
4	83*	5	S&P PE ratio	S&P's Composite Common Stock: Price-Earnings Ratio