U.S. recession forecasting with extreme gradient boosting

Rouven Beiner

August 13, 2024

Abstract

In this study, I forecast U.S. recessions using the extreme gradient boosting algorithm (XGBoost) within a real-time out of sample prediction setting, considering factors such as the availability and delay of data releases for each training period. The macroeconomic time series used as predictors are based on the reference series of the OECD CLI for the U.S.. The predictive ability of the model is assessed up to 4 future horizons on quarterly basis. A naive forecast and a baseline logit are used as benchmark models. To identify downturns, I calculate GDP turning points using a simple method based on the four business cycle phases. The results show that compared to the benchmark models, one year forecast accuracy can be significantly increased when using the XGBoost model.

Contents

List of Figures

List of Tables

1 Introduction

Forecasting economic phases, particularly recessions, is crucial for policymakers, businesses, and private agents to make informed decisions. Traditional GDP forecasting methods, such as parametric linear models or nonlinear models sometimes fail to capture the complex and dynamic non linear relationship of economic cycles and their predictors. In this study, I aim to enhance the prediction of U.S. recession phases by applying a modern machine learning technique, the XGBoost algorithm, which belongs to the family of gradient boosting decision trees.

The primary objective of this research is to construct a robust forecasting model that can accurately predict economic downturns by analyzing a wide array of macroeconomic indicators. The approach accounts for the delay of data releases, and simulates real-world forecasting conditions by re-calculating the business cycle for every training period. The study evaluates the out-of-sample prediction performance of the XGBoost model across different forecasting horizons and compares it with a baseline logit model and a naive benchmark forecast, which uses the last available data for future predictions. The results demonstrate that the XGBoost model significantly improves forecast accuracy compared to the benchmark models. The accuracy of the model for a one year forecast is 0.7, the precision (i.e., the proportion of predicted recession periods that are actual recession periods) is 0.69, and the recall (i.e., the proportion of actual recession periods that are correctly predicted as recession periods) is 0.67.

2 Literature

The idea that economic activity does not increase along a stable path but rather exhibits cyclical fluctuations around this path was put forward in the late 19th century. Since then, numerous economic theories about the causes of business cycles have emerged. Among the most prominent are the Keynesian approaches, which support active countercyclical policy intervention; the neo-classical theories, which argue for the ineffectiveness of such policies; and the new-Keynesian theory, which emphasizes contract-based wages and price stickiness. For the purposes of this analysis, it is sufficient to acknowledge the presence of economic oscillations around a stable growth path or trend. Initially described by Schumpeter, the business cycle is characterized by four phases: prosperity, recession, depression, and revival.

The prediction of economic phases in the business cycle has long been a subject of significant interest. Accurate monitoring and anticipation of economic trends are essential for effective policy formulation and economic planning. Consequently, researchers have been developing various prediction methods for decades, with early attempts tracing back to the mid-20th century. The world of GDP forecasting can be divided into two categories: Composite Leading Indicators (CLIs), which are typically developed through aggregation schemes (e.g. the OECD system of CLIs explained by Gyomai and Guidetti (2012)), and model-based approaches, including both linear and nonlinear models.

The system of OECD Composite Leading Indicators (CLIs) was developed in the 1970s to provide early signals of turning points in economic activity. The construction of the OECD CLI begins with the pre-selection of components that are considered economically relevant. This is followed by time series pre-processing, which includes seasonal adjustments, outlier detection, and business cycle identification using a Hodrick-Prescott (HP) band-pass filter. Then, the Bry-Boschan algorithm is employed to detect turning points (local minima and maxima) in the cyclical part of the series, and, at the same time, ensures minimum phase length and minimum cycle length conditions. In the second selection step, the statistical relevance of the pre-selected components is assessed. Experts evaluate these candidate component series manually using a set of statistical methods. For each component, the lead time, defined as the interval between turning points in the component and the reference series, is calculated. The goal is to find components with a lead time of six to nine months. Finally, the selected component indicators are combined and aggregated into a single composite indicator.

I critique this procedure for three main reasons. First, the selection of appropriate statistical components is done by "expert evaluation" and visual inspection with the help of simple statistical methods. This selection process could be significantly improved by employing more advanced statistical techniques. Second, the CLI does not provide precise forecasts for a specific horizon; it only serves as a general indicator for the next six to nine months. Third, the components are aggregated using equal weights, which fails to account for the varying importance of different components.

The second category of GDP forecasting methods are parametric models like linear regression models (e.g. ARIMA), nonlinear binary regression models (e.g.,

LOGIT, PROBIT) or nonlinear time series models (e.g., Markov-Switching (MS) and Self-Exciting Threshold Autoregressive (SETAR) models). For the case of Germany, Lehmann and Wohlrabe (2013) forecasted GDP at the regional level using an autoregressive distributed lag model, and Funke (1997) assessed the performance of selected predictors of recessions in Germany. The non linear parametric models are particularly suited for detecting turning points in the business cycle (Chauvet and Potter (2002),Chauvet and Potter (2005),Chauvet and Potter (2009),Bellégo and Ferrara (2009),Bellégo and Ferrara (2012),King et al. (2007)). Ferrara and Mazzi (2017) provide an overview of these techniques. The category of parametric models can be criticized because they assume a specific functional form, and maybe fall short of capturing the underlying complex relationship. Within the non parametric non linear models, some studies exploit the abilities Machine learning (ML) and artificial intelligence (AI) approaches. ML and AI approaches offer flexibility and the ability to model complex, nonlinear relationships in the data. Neural network techniques have been explored for economic forecasting, as discussed by Jagric (2003) and Qi (2001). However, neural networks typically require specific data structures and large datasets to perform effectively. Vrontos et al. (2021) compared in their study the predictive abilities of various ML algorithms compared to traditional econometric methods. The results strongly support the application of machine learning over more standard econometric techniques like probit/logit models in the prediction of U.S. recessions, as they achieve higher predictive accuracy across long-, medium-, and short-term forecast horizons. There is still limited literature on the application of ML methods, especially the XGBoost model to predict recessions. This paper aims to fill these gaps by employing the XGBoost algorithm, to predict recent U.S. recession phases.

3 Boosting

3.1 A brief introduction to boosting

The first simple boosting procedure was developed by Schapire (1990). A boosted model is constructed additively from weak learners that iteratively improve upon the errors of previous learners. A weak learner is a binary classifier with a predictive performance significantly better than random chance, with high probability. In Schapire's (1990) procedure, the first weak learner h_1 is trained on a dataset with N datapoints. The second weak learner h_2 is then trained on N datapoints half of which are misclassified by h_1. The third weak learner h_3 is trained on datapoints for which h_1 and h_2 disagree. The final boosted classifier is defined as $h_B = majorityvote(h_1, h_2, h_3)$ (Friedman et al. (2000)).

3.2 Gradient boosting

The main idea of gradient boosting is to construct a model in a iterative process by combining multiple weak (base) learners (typically basic decision trees), that reduce the (pseudo) errors made by the previous models. Thereby gradient boosting is a

generalization of boosting to arbitrary differentiable loss functions, see the seminal work of Friedman (2001).

Let $\{(x_i, y_i)\}_{i=1}^N$ be our dataset, where $x_i \in \mathbb{R}^P$ represents the i-th observation with P features, and $y_i \in \mathbb{R}$ is the corresponding target variable. Let $F(x)$ be the model we are trying to learn, which maps the feature space $\mathbb{R}^P$ to the target space $\mathbb{R}$. The model $F(x)$ is an additive expansion of the base learners

$$F(x) = \sum_{m=1}^{M} \gamma_m h_m(x, \beta) \tag{1}$$

where M is the total number of boosting iterations, $h_m(x)$ is the m-th base learner, γ_m is the corresponding weight for the m-th base learner and β are the parameters to be estimated by the base learner. For trees, the parameter vector contains identification of splitting variables, their splitting values, and the constants in the terminal nodes.

The model is initialized with a constant prediction. Assuming that each Y_t is a outcome of independent bernoulli random variables ($Y_t = 1$ denotes recession at time t), with probability of recession p_t, the initial prediction is chosen to be the log-odds of the recession probability:

$$F_0(x_i) = \log\left(\frac{\bar{y}}{1-\bar{y}}\right) \tag{2}$$

where $\bar{y}$ is the mean of the target values.

Within each iteration $m = 1$ to M, first the pseudo residuals are computed. The pseudo-residuals are the negative gradients of the loss function with respect to the current model's predictions. For each observation i, the pseudo-residual r_{im} is computed as:

$$r_{im} = -\left[\frac{\partial L(y_i, F(x_i))}{\partial F(x_i)}\right]_{F(x)=\hat{F}_{m-1}(x)} \tag{3}$$

In binary classification, a common choice for the loss function is the logistic loss (log loss), which measures the difference between the predicted probabilities and the actual binary outcomes. The logistic loss function is defined as:

$$L(y_i, p) = -\left[y_i \log(\hat{y}_i) + (1-y_i)\log(1-\hat{y}_i)\right] \tag{4}$$

where $y_i \in \{0, 1\}$ is the true binary label, $\hat{y}_i = \sigma(F_{m-1}(x_i))$ is the predicted probability from the current model, and σ is the sigmoid function:

$$\sigma(F_{m-1}(x_i)) = \frac{1}{1+e^{-F_{m-1}(x_i)}} = \frac{e^{F_{m-1}(x_i)}}{1+e^{F_{m-1}(x_i)}} \tag{5}$$

For logistic loss, the pseudo-residuals are:

$$r_{im} = \frac{\partial L(y_i, \hat{y}_i)}{\partial F_{m-1}(x_i)} = \hat{y}_i - y_i \tag{6}$$

A detailed derivation can be found in Appendix A. As a second step within the iteration m, a new base learner is fitted to the estimated pseudo-residuals by solving the following optimization problem:

$$\hat{\beta} = \arg\min_{ß} \sum_{i=1}^{N} (r_{im} - h(x_i, \beta))^2 \tag{7}$$

As a third step within iteration m, the model is updated:

$$\hat{F}_m(x_i) = \hat{F}_{m-1}(x_i) + \gamma_m h_m(x_i, \hat{\beta}) \tag{8}$$

And the updated predicted probability is:

$$\hat{y}_i = \sigma(\hat{F}_m(x_i)) \tag{9}$$

The weight (step size) γ_m can be determined through a line search that minimizes the loss function:

$$\gamma_m = \arg\min_{\gamma} \sum_{i=1}^{N} L(y_i, \hat{F}_{m-1}(x_i) + \gamma_m h_m(x_i, \hat{\beta})) \tag{10}$$

3.3 Extreme gradient boosting

Extreme Gradient Boosting (XGBoost) builds upon the principles of gradient boosting but introduces several enhancements to improve performance and efficiency. The algorithm was developed by Chen and Guestrin (2016). Similar to the gradient boosting algorithm, the XGBoost is also a additive model, that builds on the pseudo residuals of several base learners. But while the base learners in the GB minimizes the negative gradients of the loss function, the XGBoost uses a second-order Taylor expansion of the loss function around the current predictions $\hat{y}_{i,m-1}$ to evaluate different tree structures.

The regularized objective function in XGBoost is defined as:

$$\mathcal{L} = \sum_{i=1}^{n} l(y_i, \hat{y}_m(x_i)) + \sum_{m=1}^{M} \Omega(f_m) \tag{11}$$

where $l(y_i, \hat{y}_m(x_i))$ is a differentiable loss function (i.e. the logistic log loss), $\Omega(f_m)$ is the regularization term for the m-th tree that penalizes the complexity and helps to avoid overfitting. $\hat{y}_m(x_i)$ is the prediction for the i-th observation at iteration m and is given by: $\hat{y}_m(x_i) = \sum_{m=1}^{M} f_m(x_i) = \hat{y}_{i,m-1} + f_m(x_i)$. Since the model is trained in an additive manner, the objective function for iteration m is given by:

$$\mathcal{L}(m) = \sum_{i=1}^{n} l(y_i, \hat{y}_{i,m-1} + f_m(x_i)) + \Omega(f_m) \tag{12}$$

That means in each iteration m, the function $f_m(x_i)$ is greedily added that most improves our model according to the regularized objective (Eq. (11)). Using a

second-order Taylor expansion around the current prediction $\hat{y}_{i,m-1}$, the loss function can be approximated as:

$$\mathcal{L}(m) \approx \sum_{i=1}^{n} \left[l(y_i, \hat{y}_{m-1}) + g_i f_m(x_i) + \frac{1}{2} h_i f_m(x_i)^2 \right] + \Omega(f_m) \tag{13}$$

where the first and second derivatives (gradient statistics) of the logistic loss function with respect to the current prediction for observation i at iteration m are given by

$$g_i = \frac{\partial - l(y_i, \hat{y}_{m-1})}{\partial \hat{y}_{m-1}} = \hat{y}_{i,m-1} - y_i \tag{14}$$

and

$$h_i = \frac{\partial^2 - l(y_i, \hat{y}_{m-1})}{\partial \hat{y}_{m-1}^2} = \hat{y}_{i,m-1}(1 - \hat{y}_{i,m-1}) \tag{15}$$

The derivation can be found in Appendix A. The constant term does not contain information for the current base learner and can be removed for simplification:

$$\mathcal{L}(m) \approx \sum_{i=1}^{n} \left[g_i f_m(x_i) + \frac{1}{2} h_i f_m(x_i)^2 \right] + \Omega(f_m) \tag{16}$$

The regularization term $\Omega(f_m)$ for a tree model f_m with T leaves is given by:

$$\Omega(f_m) = \gamma T + \frac{1}{2} \lambda f_m \tag{17}$$

$$= \gamma T + \frac{1}{2} \lambda \sum_{j=1}^{T} w_j^2. \tag{18}$$

where w_j is the weight of leaf j of all leaves T and γ and λ are constants to control the degree of regularization.

Now, define $I_j = \{i | q(\mathbf{x}_i) = j\}$ as the instance set of leaf j. We can rewrite Eq. (16) by expanding Ω as follows:

$$\mathcal{L}(m) = \sum_{i=1}^{n} \left[g_i f_m(\mathbf{x}_i) + \frac{1}{2} h_i f_m(\mathbf{x}_i)^2 \right] + \gamma T + \frac{1}{2} \lambda \sum_{j=1}^{T} w_j^2 \tag{19}$$

and since $f_m(\mathbf{x}_i) = \sum_{j=1}^{T} w_j$ we can further rewrite the equation:

$$\mathcal{L}(m) = \sum_{j=1}^{T} \left[\left(\sum_{i \in I_j} g_i \right) w_j + \frac{1}{2} \left(\sum_{i \in I_j} h_i + \lambda \right) w_j^2 \right] + \gamma T \tag{20}$$

For a fixed tree structure $q(\mathbf{x})$, the optimal weight w_j^* of leaf j can be computed by optimizing the Eq. (20):

$$\frac{\partial \mathcal{L}(m)}{\partial w_j} = \sum_{i \in I_j} g_i + (\sum_{i \in I_j} h_i + \lambda) w_j^* = 0 \tag{21}$$

$$w_j^* = -\frac{\sum_{i \in I_j} g_i}{\sum_{i \in I_j} h_i + \lambda} \tag{22}$$

Now we can calculate the corresponding optimal value by plugging in w_j^* into Eq. (20):

$$\mathcal{L}^*(m,q) = -\frac{1}{2}\sum_{j=1}^{T}\frac{\left(\sum_{i\in I_j} g_i\right)^2}{\sum_{i\in I_j} h_i + \lambda} + \gamma T = -\frac{1}{2}\sum_{j=1}^{T}\frac{G_j^2}{H_j+\lambda} + \gamma T \tag{23}$$

Equation (23) can be used as a scoring function (or sometimes called the similarity score) to measure the quality of a tree structure q. The lower the score, the better the tree structure. The score is calculated by iterating through all leaves (end nodes) j of the tree and summing the gradients of the loss function g_i for all observations i that fall into each respective leaf j, yielding G_j. The squared sum of the gradients G_j^2 is then divided by the sum of the second derivatives plus an overfitting constant λ. The gradients g_i are derived from the loss function, with positive or negative values depending on whether the current model overestimates or underestimates the prediction for observation i. The scoring function shows that positive and negative gradients can cancel each other out, leading to a lower gradient sum G_j for the respective leaf. As a result, tree structures that group observations with gradients of the same sign (e.g., mostly positive) into the same leaf are favored, as a higher $G_i = \sum_{i\in I_j} g_i$ will, ceteris paribus, reduce the score, indicating a better tree structure. Conversely, a tree structure where gradients within the same leaf cancel each other out will result in a lower squared sum of gradients G_i and, ceteris paribus, a higher score, indicating a less favorable tree structure. In other words, a tree structure is preferred if, within each leaf, the predictions for the observations are either all overestimated or all underestimated. Figure 1 provides an example of a tree structure, along with the gradients and hessians of the leaves, given the data points in Table 1. Normally it is impossible to enumerate all the possible tree structures q. Therefore an algorithm that starts from a single leaf and iteratively adds branches to the tree is used instead. Each new potential split is evaluated by estimating the gain or reduction in loss (improvement in the objective function) which is calculated using:

$$\text{Gain} = \frac{1}{2}\left[\frac{G_L^2}{H_L+\lambda} + \frac{G_R^2}{H_R+\lambda} - \frac{G^2}{H+\lambda}\right] - \gamma \tag{24}$$

where G and H are the sum of gradients and Hessians for the node and G_L, H_L (G_R, H_R) are the sum of gradients and Hessians for the left (right) child node. λ is the regularization parameter to prevent overfitting and γ is the minimum loss reduction required to make a split. The overall gain from a new split is according to Eq. (24) the gain of the left child node plus the gain of the right child node. If the gains of both child nodes exceed the gain of the parent node, then the new split is justified. A constant threshold γ ensures that the gain is greater than a minimum value.

After the structure of the tree is determined (i.e. the split points are calculated with Eq. (24)), each leaf is weighted with the optimal weight w_j^*. The new tree's predictions $h_m(x_i, \hat{\beta})$ are added to the current model's predictions, weighted by the learning rate η:

$$F_M(x_i) = F_0(x_i) + \sum_{m=1}^{M} \eta \cdot h_m(x_i, \hat{\beta}) \tag{25}$$

Observation	Gradient Statistics	Variable
1	g_1, h_1	0.1
2	g_2, h_2	0.05
3	g_3, h_3	0.7
4	g_4, h_4	0.8
5	g_5, h_5	0.3

Table 1: XGBoost scoring example

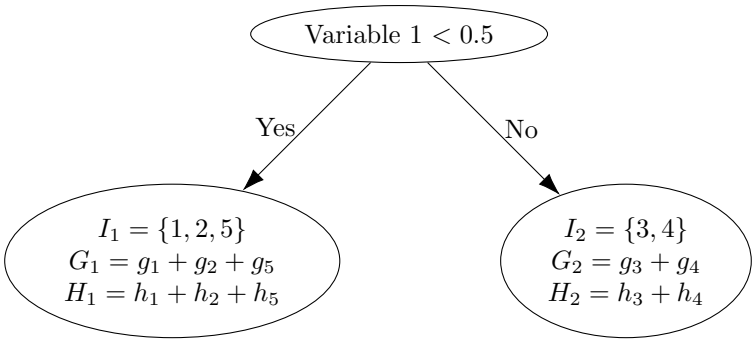

Figure 1: Decision tree with one split and gradient statistics

4 Data

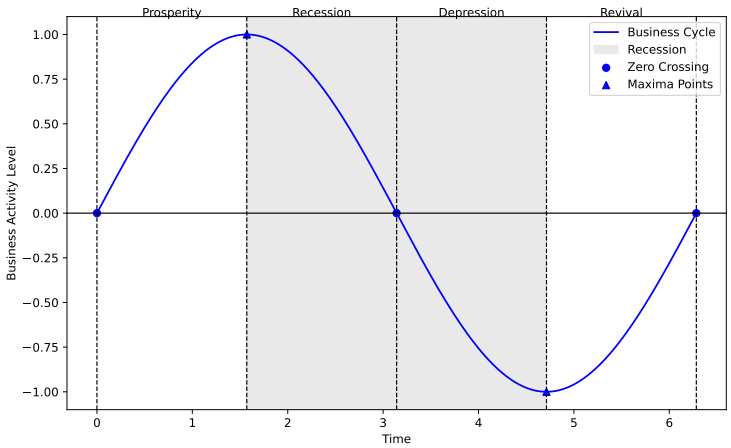

Figure 2: Business Cycle Phases, Zero Crossings and Maxima Points

The endogenous variable is a binary recession indicator derived from the U.S. GDP cycle, analyzed on a quarterly basis. For the identification of recession phases I employed a basic algorithm that processes the business cycle component of a time series according to the following steps: First, zero-crossings are identified. Then in a second step, the time series is divided into segments between consecutive zero crossings. Each segment is analyzed to identify the local maxima or minima, which serve as turning points within the segment. Then the four identified phases

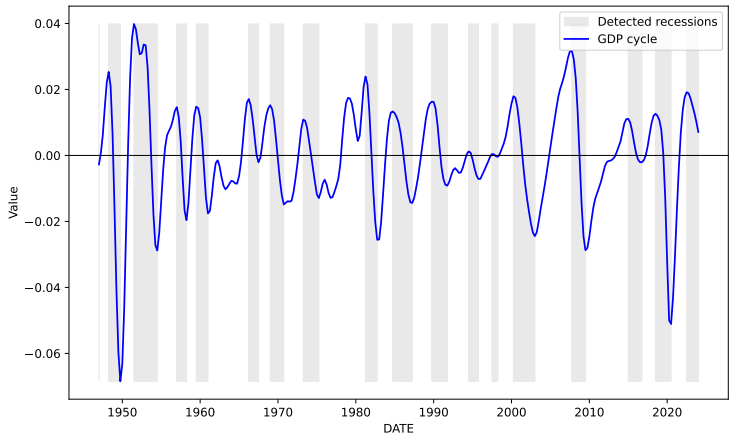

Figure 3: U.S. GDP business cycle and detected recessions

are labelled as follows: Prosperity, the period from a zero crossing to the next local maxima. Recession, the period from the local maxima to the subsequent zero crossing. The period of the local maxima is included in the recession section. Depression, the period from a zero crossing to the next local minima. Revival, the period from the local minima to the subsequent zero crossing. The period of the local minima is included in the Revival section. If the time series does not end at a zero crossing, the function identifies the last turning point and labels the phase up to the end of the series accordingly. As illustrated in Figure 2, the theoretical business phases, zero crossings, and turning points are depicted, while Figure 3 shows the U.S. GDP cycle and the corresponding identified business cycle phases. I aim to predict periods of low economic activity, specifically recession and depression. For simplicity, and in line with the National Bureau of Economic Research (NBER) definition, I will refer to both recession and depression as "recessions" in the following discussion. In general, this method detects more recessions than the conventional NBER method, as they also consider the duration and amplitude of economic downturns. According to the NBER, a recession is defined as a significant decline in economic activity that is widespread across the economy and lasts more than a few months. However, using the most recent published NBER's historical recession indicator series for time series cross validation is not appropriate. In a time series cross-validation context, training models on past periods necessitates the assumption that the data available now were also available at that time in the past. Given that the NBER's recession determinations can take between 4 and 21 months after the recession occurs (National Bureau of Economic Research (2024a)), this leads to the problem of future information leakage, which is further detailed in Section 3.

I used the component series of the OECD's Composite Leading Indicator for U.S. (OECD (2022)) as regressors in the model. These regressors have been proven to be economically relevant, thereby avoiding spurious correlations. They also exhibit leading properties, such as financial data (share prices, interest spreads), data on industrial production, and surveys of consumers and businesses.

The GDP data, which serves as the basis for the recession indicator, is only

available on a quarterly basis. For the other regressors (which are available on a monthly or daily basis), I estimated the average for each respective quarter. A critical aspect is the timeliness of the data. Most economic indicators are available with a lag of one month, whereas GDP data are available with a lag of one quarter. In the model, lagged regressors and the lagged GDP variable are included from the first lag up to the eighth lag, thus considering the previous two years. The availability is also considered for the naive benchmark model, as this model "knows" the data from the quarter before. Table 2 provides information about the regressors used in this

Variable	**Frequency**	**Unit**
Gross Domestic Product	Quarterly	Billions of Dollars, Seasonally Adjusted Annual Rate, log-transformed
Work Started (Construction, Dwellings and Residential Buildings)	Monthly	Number, Monthly level, Seasonally Adjusted
Manufacturers' New Orders (Durable Goods)	Monthly	Millions of Dollars, Seasonally Adjusted
NYSE Composite Index	Quarterly	Millions of Dollars, Not Seasonally Adjusted, log-transformed
Composite Consumer Confidence	Monthly	Normalised (Normal=100), Seasonally Adjusted
Hours Worked (Manufacturing)	Monthly	Hours, Seasonally Adjusted
Business Tendency Surveys (Manufacturing)	Monthly	Percent, Seasonally Adjusted
Interest spread (10-Year Treasury Minus Federal Funds Rate)	Daily	Percent, Not Seasonally Adjusted

Table 2: Economic Indicators

study, where IX = Index, PA = Percentage per annum, PB = Percentage balance, Y = Calendar and seasonally adjusted, Z = Not applicable, PT_LF = Percentage of labour force, RT = Ratio to trend, GY = Growth rate, over 1 year, GR = Growth rate. Most regressors are seasonally adjusted, and some are also detrended. In each training period, the regressors are detrended using the HP high-pass filter. The final set of regressors includes both the detrended levels of the regressors and their first differences (slopes). The variable $change\_count\_GDP$ estimates the duration of each business cycle phase up to the current period. The variable GDP_i is a dummy indicator representing the current cycle phase i. Additionally, an interaction term between the detrended levels and an indicator for whether the regressor is in phase 1 or 4 is included as $regressor\_phase14$.

5 Time series preparation

The business cycle component is extracted from the target variable (GDP) using a band-pass filter and the regressors are detrended using a high-pass filter. The Band-pass filtering is done with the Hodrick-Prescott (HP) filter which is applied twice to

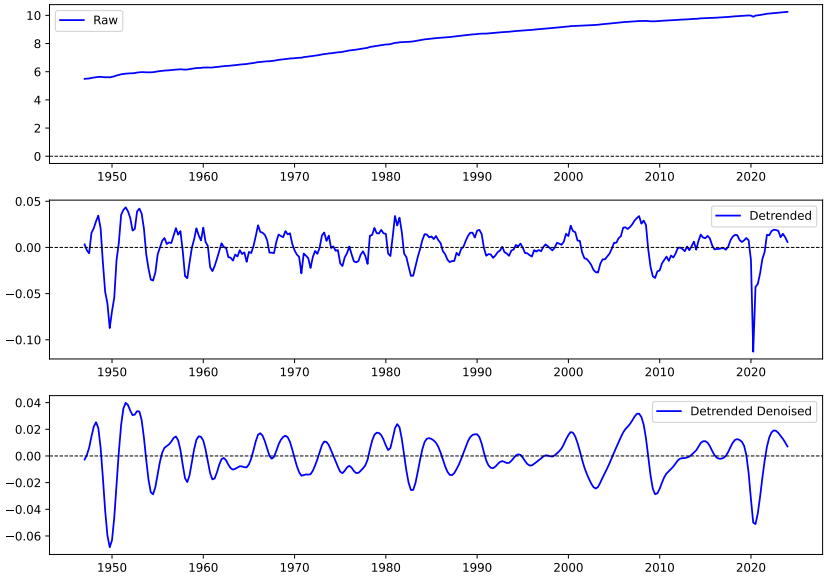

Figure 4: Hodrick-Prescott band-pass filter on GDP

the time series to extract the business cycle frequency component, as proposed by Pedersen (2001). A time series filter separates a time series into two components, often called trend and cycle of the filter. For the HP band-pass filter, first, the HP high-pass filter with $\lambda_1 = 2413.06$ removes the long-term trend, i.e. the trend of the filter is subtracted from the raw time series. Then, the HP low-pass filter with $\lambda_2 = 2.91$ is applied to the detrended series, i.e. the cycle of the filter is subtracted from the detrended series to get the final business cycle component. The HP filter minimizes the following objective function:

$$\min_{\{\tilde{y}_t\}_{t=1}^{T}} \left\{ \sum_{t=1}^{T} \left(y_t - \tilde{y}_t\right)^2 + \lambda \sum_{t=2}^{T-1} \left[(\tilde{y}_{t+1} - \tilde{y}_t) - (\tilde{y}_t - \tilde{y}_{t-1})\right]^2 \right\}$$

The frequency response function of the Hodrick-Prescott filter is given by:

$$R(\omega) = \frac{1}{1 + 4\lambda[1 - \cos(\omega)]^2}$$

Now solving for the HP-filter parameter λ:

$$\lambda = \frac{1 - R(\omega)}{4R(\omega)[1 - \cos(\omega)]^2}$$

It can be shown that in the case of the HP filter, the gain function equals the frequency response function. Using a gain of $R(\omega) = 0.5$ and replacing the frequency in radians (ω) with the ordinary frequency (f) multiplied by 2π, the equation can be simplified as follows:

$$\lambda = \frac{1 - 0.5}{4 \cdot 0.5 \cdot [1 - \cos(\omega)]^2} = \frac{0.25}{[1 - \cos(\omega)]^2} = \frac{0.25}{[1 - \cos(2\pi f)]^2}$$

The lambda parameters are set so that the frequency cut-off occurs at frequencies higher than 8 quarters and lower than 44 quarters. These cut-offs are based on the U.S. recession durations reported by the NBER, where all recessions from 1858 onwards last longer than two years and shorter than eleven years (National Bureau of Economic Research (2024b)) Therefore, the corresponding ordinary frequencies of $f_1 = 1/44$ and $f_2 = 1/8$ are used:

$$\lambda_1 = \frac{0.25}{\left(1 - \cos\left(\frac{1}{40}2\pi\right)\right)^2} = 2413.06$$

$$\lambda_2 = \frac{0.25}{\left(1 - \cos\left(\frac{1}{4}2\pi\right)\right)^2} = 2.91$$

To ensure that the model is placed in a real-world setting for each respective training period, it is essential to avoid leakage of future information. There are two potential "future-information-leakage" issues, resulting from the HP-filter and the recession detection algorithm. The filter leakage would be introduced if models which are trained for a smaller period ($t = 1, t = T - m$) are trained on time series where the filter is applied on the entire time series ($t = 1, t = T$). Reason for the leakage is that the HP filter uses values from both before and after (the future) the respective time period. Especially the right endpoints are influenced from this endpoint bias. Therefore the HP filter is applied again for each new training period. In a similar way the recession detection is applied again for each training period separately. Because after the last zero crossing, a recession can not be detected by guarantee. Lets imagine the last zero crossing from under zero to greater than zero appeared some periods ago and since then the business cycle only increased. Now we have a falling period which is smaller than the last maximum. Is this the beginning of a recession? This depends if the values fall further and cross the zero again, then we can for sure talk about a recession. But if the values decrease only for some periods, and then again increase and even reach new maxima, we are still in expansion phase. Therefore for all periods after the last zero crossing, the algorithm is dynamic, meaning that it can detect a recession, but if new maxima are reached again after some periods, the detected recession can be removed again. It is not appropriate to use the recession series which is calculated from the whole time series for models that use smaller training periods because at that time before, the recession could be not detected for sure.

6 Forecasting setup

Lags of the levels and slopes are created from the detrended regressors. The following feature selection procedure is then applied: First, a base XGBoost model with default parameters is estimated. The internally calculated feature importance values are used to filter out only the most important features, with the threshold set at the 90th percentile. This feature selection process is applied for each training period.

Decision trees, which serve as the base learners in XGBoost, offer two main advantages. First, they can handle diverse types of data with varying scales and ranges.

Second, they can model complex, nonlinear relationships non-parametrically by automatically capturing interactions without the need for manual feature engineering. However, they can overfit, identifying random relationships rather than true patterns. The potential issue of overfitting (bias-variance trade-off) is addressed with out-of-sample forecasting and conducting a grid search to find the optimal value for the *max_depth* parameter. While deeper trees can capture more complex patterns, they also run the risk of overfitting if they become too deep. The *max_depth* parameter determines the maximum depth of the subtrees in the XGBoost model. The results of the forecast accuracy for different *max_depth* values are displayed in Figure 5. The highest accuracy is achieved with a value of 4,11,12 and 13, indicating that the model prefers complex data patterns. However, there is no clear positive trend visible. The model's accuracy is evaluated using a standard out-of-sample forecasting approach through time series cross-validation. The initial training set contains the first 240 periods. For each forecast horizon h, a model is trained to predict the value at $t = 550 + h$. The training set is then updated by adding one more period in each iteration. The accuracy for a specific horizon is estimated based on the forecasts generated for each training period at that horizon. The concept of the basic time series cross-validation is illustrated in Figure 6. To account for delays in data releases, the models incorporate only lags. This ensures that the forecasts are based solely on information available up to the point of prediction. Forecast accuracy is computed by averaging the performance metrics over all test sets. The more advanced k-fold time series cross-validation, discussed by Bergmeir et al. (2018) is not used in this study.

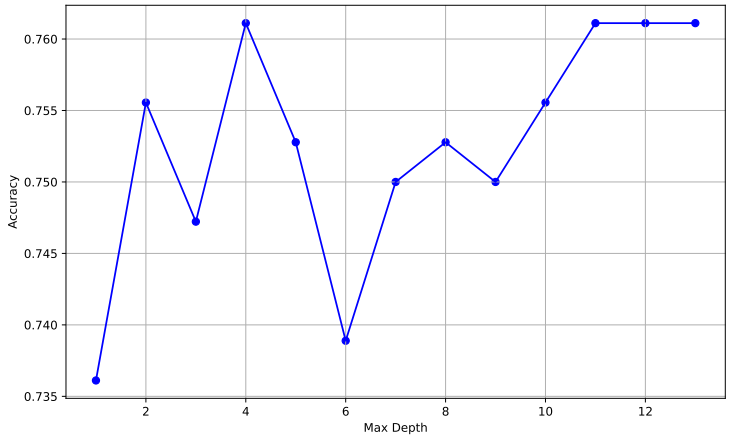

Figure 5: Grid Search Accuracy over Max Depth Values

7 Results and Discussion

Figure 7 illustrates the overall accuracy of the XGBoost model compared to the benchmark models: the naive forecast and the baseline logit model. The comparison is made across all prediction horizons and for XGBoost with *max_depth* = 4 and

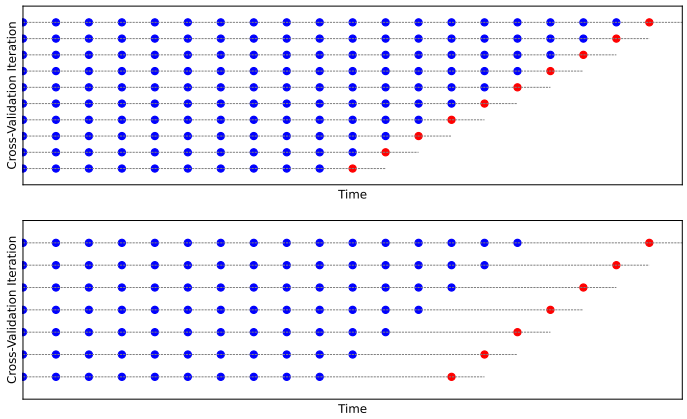

Figure 6: Time series cross validation for forecast horizons of 1 (top) and 4 (bottom).

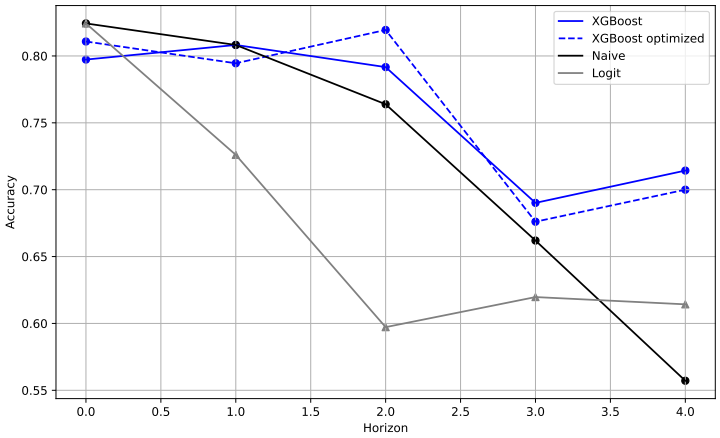

Figure 7: Model accuracy comparison over horizons

both thresholds (the default 0.5 and the optimized 0.4). Accuracy is measured as the proportion of correctly predicted labels. The XGBoost model outperforms the naive forecast from the second horizon on. In contrast, the logit model only outperforms the naive forecast in the fourth horizon. As expected, the naive forecast accuracy is relatively high at the beginning and monotonically falls with increasing horizons. However, the differences in accuracy between the XGBoost and the naive up to the third horizon are small and probably not significant. The accuracy values for the optimized XGBoost model at the fourth horizon are bootstrapped to determine if they significantly differ from those of the naive forecast. The 90% confidence intervals (CIs) from a bootstrap with 1,000 repetitions are shown in Figure 8. The accuracy of the naive forecast falls outside the 90% CI, indicating that the optimized XGBoost model's accuracy is significantly higher than that of the naive forecast, at least for the fourth horizon. The four quarters forecast XGBoost model with the optimized threshold has a accuracy of 0.7, a Recall (TPR) of: $\text{TPR} = \frac{\text{TP}}{\text{TP+FN}} = 0.69$

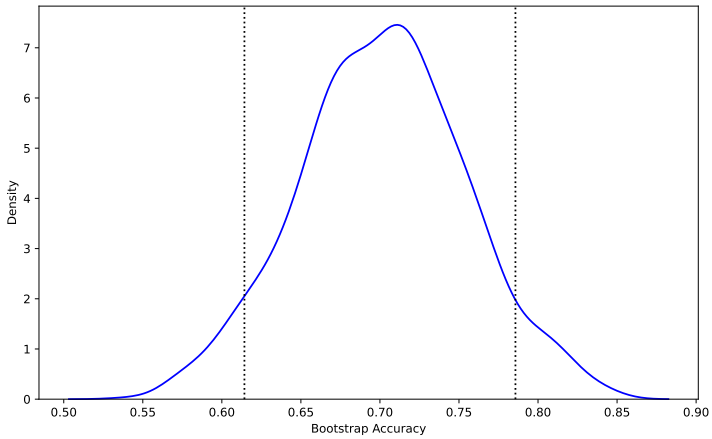

Figure 8: Bootstrapped accuracy distribution for optimized XGBoost (h=4)

and a precision of: Precision $= \frac{\text{TP}}{\text{TP+FP}} = 0.67$. The Recall, a.k.a True Positive Rate (TPR) is the ratio of correctly identified recession periods (TP) to the total number of actual recession periods (TP + FN). The precision is the proportion of all predicted recession periods that are actual recession periods.

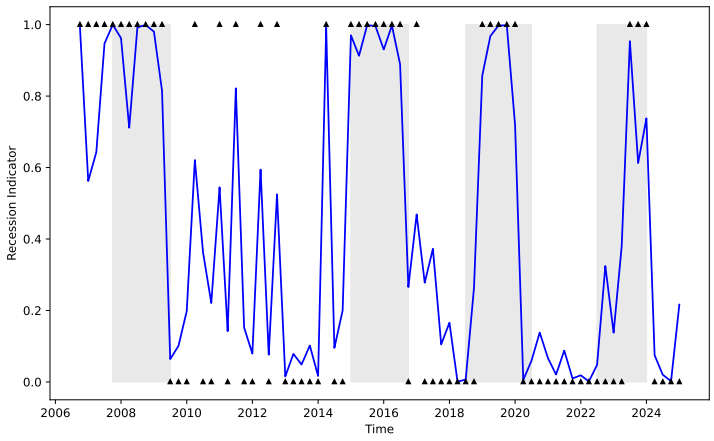

Figure 9: Comparison of actual and predicted recession probabilities over time

Figure 9 illustrates the actual and predicted probabilities of a recession occurring over time for a one-year forecast, along with the predicted labels using the optimized threshold. The XGBoost model predicted the recessions in 2008/2009 and 2015/2016 relatively good but could predict the most recent recessions in 2019/2020 and 2023 only with a forecast horizon of two and three quarters. This can be attributed to the fact that the last two recessions, caused by the COVID-19 pandemic and the Russia-Ukraine war, were not driven by typical economic factors. The recession 2008/2009 was predicted too early, and in the period between the recessions 2008/2009 and 2015/2016, some false positives occurred.

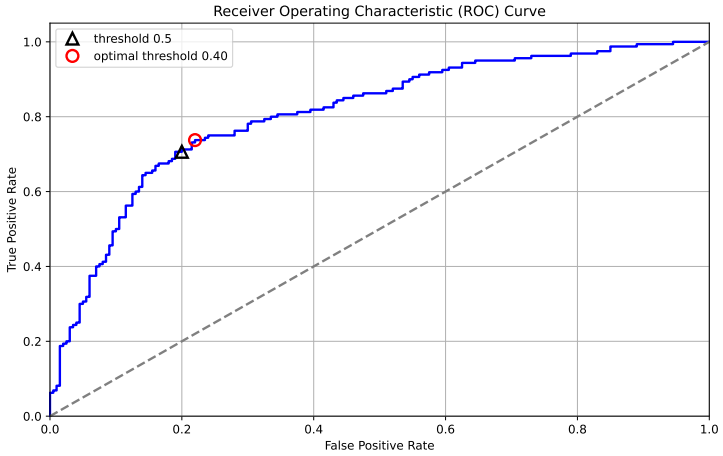

Figure 10: Receiver operator curve and threshold optimization

For the threshold optimization, the ROC curve is an essential tool. The ROC curve plots the false positive rate (FPR) against the true positive rate (TPR). The FPR represents the proportion of non-recession periods falsely classified as recessions, while the TPR indicates the proportion of actual recessions correctly identified. Reducing the FPR, while increasing the TPR is of interest. From the 0.5 threshold point on the ROC curve, a marginal increase in the FPR can lead to a bigger gain in the TPR. The optimal threshold, determined mathematically, is 0.4 (displayed in Figure 10). Graphically, the optimal point on the ROC curve is where the slope of the ROC equals the slope of the straight line from the bottom left corner to the top right corner. Figure 11 ranks the most important predictors. This

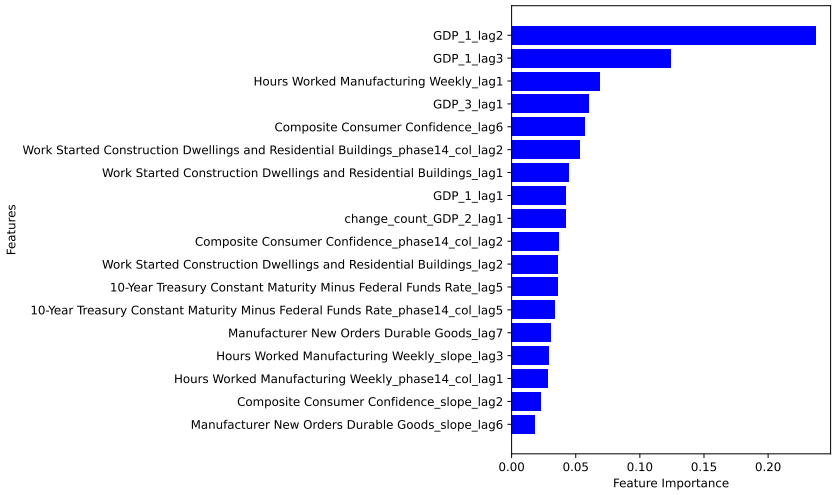

Figure 11: Most important predictor variables

ranking is based on the entire time range, which includes multiple recessions, each with different underlying causes. The feature importance score can tell only the importance of regressors, but can not tell if the causality is positive or negative. The

lagged GDP phase indicators rank high. For example, if the economy was in a state of prosperity in the preceding quarters, as indicated by the regressors GDP_1_lag1, GDP_1_lag2, and GDP_1_lag3, the likelihood of a recession is relatively high. Apart from these autoregressive predictors, the hours worked in Manufacturing, the work started in construction and the composite consumer confidence survey index rank highly in importance.

One challenge of using nonlinear models for recession forecasting is the relatively low number of recessions. Even with the progressive recession indicator used in this study, only 18 recessions occurred over the entire time period. Models trained on shorter time periods have even less data to learn from and identify relationships. In this study, the initial training set consists of 240 periods, with the subsequent 73 periods used to assess the model's accuracy. It is important to note that the accuracy of these models is sensitive to the size of the test set, as this determines which recessions are included. In this study, I used a very limited set of predictors. The forecasting performance could likely be improved by incorporating a larger set of predictors, as used by Vrontos et al. (2021) A further improvement could involve using an ensemble approach by combining forecasts from different models, which is often statistically preferred.

A Gradients and hessian of the logistic loss function

The probability function of the binary target variable $y_i \sim \mathcal{B}(1, \hat{y}_i)$ is defined as

$$f(y_i) = \hat{y}_i^{y_i}(1-\hat{y}_i)^{1-y_i} = \begin{cases} \hat{y}_i & \text{if } y_i = 1 \\ 1-\hat{y}_i & \text{if } y_i = 0 \end{cases}.$$

The likelihood function is defined as

$$L = \prod_{i=1}^{n} f(y_i) = \prod_{i=1}^{n} \hat{y}_i^{y_i}(1-\hat{y}_i)^{1-y_i}.$$

Taking the logarithm of the likelihood function leads to the log-likelihood

$$l = \ln L = \sum_{i=1}^{n} \ln\left(f(y_i)\right) = \sum_{i=1}^{n} \left[y_i F_{m-1}(x_i) - \ln\left(1+\exp(F_{m-1}(x_i))\right)\right],$$

where

$$\begin{aligned} \ln f(y_i) &= \ln\left(\hat{y}_i^{y_i}(1-\hat{y}_i)^{1-y_i}\right) \\ &= \ln\left(\hat{y}_i^{y_i}\right) + \ln\left((1-\hat{y}_i)^{1-y_i}\right) \\ &= y_i \ln(\hat{y}_i) + (1-y_i)\ln(1-\hat{y}_i) \\ &= y_i \ln(\hat{y}_i) + \ln(1-\hat{y}_i) - y_i \ln(1-\hat{y}_i) \\ &= y_i \ln\left(\frac{\hat{y}_i}{1-\hat{y}_i}\right) + \ln(1-\hat{y}_i). \end{aligned}$$

Because of

$$\begin{aligned} \hat{y}_i = P(y_i = 1) &= \frac{\exp(F_{m-1}(x_i))}{1+\exp(F_{m-1}(x_i))}, \\ \ln\left(\frac{\hat{y}_i}{1-\hat{y}_i}\right) &= F_{m-1}(x_i), \end{aligned}$$

and

$$1-\hat{y}_i = \frac{1}{1+\exp(F_{m-1}(x_i))},$$

it follows that

$$\begin{aligned} \ln f(y_i) &= y_i \ln\left(\frac{\hat{y}_i}{1-\hat{y}_i}\right) + \ln(1-\hat{y}_i) \\ &= y_i F_{m-1}(x_i) + \ln\left(\frac{1}{1+\exp(F_{m-1}(x_i))}\right) \\ &= y_i F_{m-1}(x_i) - \ln\left(1+\exp(F_{m-1}(x_i))\right) \end{aligned}$$

The log loss per sample is the negative log-likelihood: $-ln(f(y_i))$. The first derivative (gradient) of the log loss can be found by differentiation:

$$\begin{aligned} g_i &= \frac{\partial - \ln\left(f(y_i, F_{m-1}(x_i))\right)}{\partial F_{m-1}(x_i)} \\ &= -\left(y_i - \frac{1}{1+\exp(F_{m-1}(x_i))} \cdot \exp(F_{m-1}(x_i))\right) = -(y_i - \hat{y}_i). \end{aligned} \tag{26}$$

The second derivative (hessian) can be found by differentiating twice using the quotient rule:

$$\begin{aligned} h_i &= \frac{\partial g_i}{\partial F_{m-1}(x_i)} \\ &= -\frac{\partial(y_i - \hat{y}_i)}{\partial F_{m-1}(x_i)} \\ &= \frac{\partial \hat{y}_i}{\partial F_{m-1}(x_i)} \\ &= \hat{y}_i(1 - \hat{y}_i). \end{aligned} \tag{27}$$

References

Bellégo, C. and Ferrara, L. (2009). Forecasting euro area recessions using time-varying binary response models for financial variables. *SSRN Electronic Journal*, (259).

Bellégo, C. and Ferrara, L. (2012). Macro-financial linkages and business cycles: A factor-augmented probit approach. *Economic Modelling*, 29:1793–1797.

Bergmeir, C., Hyndman, R. J., and Koo, B. (2018). A note on the validity of cross-validation for evaluating autoregressive time series prediction. *Computational Statistics and Data Analysis*, 120:70–83.

Chauvet, M. and Potter, S. (2002). Predicting recessions: Evidence of the yield curve in the presence of structural breaks. *Economics Letters*, 77(2):245–253.

Chauvet, M. and Potter, S. (2005). Forecasting recessions using the yield curve. *Journal of Forecasting*, 24(2):77–103.

Chauvet, M. and Potter, S. (2009). Monitoring business cycles with structural breaks. *MPRA Paper*, (No. 15097).

Chen, T. and Guestrin, C. (2016). Xgboost: A scalable tree boosting system. *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*.

Ferrara, L. and Mazzi, G. (2017). *Handbook on Cyclical Composite Indicators for Business Cycle Analysis*. The Conference Board, Inc., Banque de France and Independent Expert, 1st edition.

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

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

Funke, N. (1997). Predicting recessions: Some evidence for germany. *Weltwirtschaftliches Archiv*, 133(1):90–102.

Gyomai, G. and Guidetti, E. (2012). Oecd system of composite leading indicators. This methodology guideline has been prepared by Gyorgy Gyomai and Emmanuelle Guidetti.

Jagric, T. (2003). Forecasting with leading economic indicators — a neural network approach. In *Operations Research Proceedings*, volume 2002, pages 486–491. Springer, Berlin, Heidelberg.

King, T. B., Levin, A. T., and Perli, R. (2007). Financial market perceptions of recession risk. Technical Report 2007-57, FEDS Working Paper.

Lehmann, R. and Wohlrabe, K. (2013). Forecasting gdp at the regional level with many predictors. *German Economic Review*, 16(2):226–254.

National Bureau of Economic Research (2024a). Business cycle dating. Accessed: 2024-08-04.

National Bureau of Economic Research (2024b). U.s. business cycle expansions and contractions. Accessed: 2024-08-08.

OECD (2022). Oecd composite leading indicators: Turning points of reference series and component series. Retrieved on May 22, 2024.

Qi, M. (2001). Predicting us recessions with leading indicators via neural network models. *International Journal of Forecasting*, 17:383–401.

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

Vrontos, S. D., Galakis, J., and Vrontos, I. D. (2021). Modeling and predicting u.s. recessions using machine learning techniques. *International Journal of Forecasting*, 37:647–671.