Empirical Economics (2024) 66:2191–2247
https://doi.org/10.1007/s00181-023-02515-6

Benchmarking econometric and machine learning methodologies in nowcasting GDP

Daniel Hopp[1]

Received: 10 May 2023 / Accepted: 9 October 2023 / Published online: 4 November 2023

Abstract
Nowcasting can play a key role in giving policymakers timelier insight to data published with a significant time lag, such as final GDP figures. Currently, there are a plethora of methodologies and approaches for practitioners to choose from. However, there lacks a comprehensive comparison of these disparate approaches in terms of predictive performance and characteristics. This paper addresses that deficiency by examining the performance of 17 different methodologies in nowcasting US quarterly GDP growth, including all the methods most commonly employed in nowcasting, as well as some of the most popular traditional machine learning approaches. Performance was assessed over a 20-year period, from 2002 to 2022. This span encompassed two crises, the 2008 financial crisis and the COVID crisis, as well as extended tranquil periods. The two best-performing methodologies in the analysis were long short-term memory artificial neural networks (LSTM) and Bayesian vector autoregression (Bayesian VAR). To facilitate further application and testing of each of the examined methodologies, an open-source repository containing boilerplate code that can be applied to different datasets is published alongside the paper, available at: github.com/dhopp1/nowcasting_benchmark

Keywords Macroeconomic forecasting · Machine learning · Neural networks · Vector autoregression models · Econometric models · GDP · Bayesian methods · ARIMA models

1 Introduction

Gross domestic product's (GDP) importance in quantifying the size and performance of the economy cannot be understated. It is the go-to metric for government officials,

✉ Daniel Hopp
daniel.hopp@un.org

[1] UNCTAD, Palais des Nations ONU, Office E 10035, Avenue de la Paix 8, 1211 Genève 10, Switzerland

policymakers, and even the general public for insight to economic health, and by extension a myriad of related measures, such as social well-being (Dynan et al. 2018; IMF 2020; Kapoor and Debroy 2019). However, as much as we depend on this figure, the reality is that it is often published with a significant lag. This lag depends on the country, but in the USA, advanced estimates for an elapsed quarter are not released for at least one month afterward, with final estimates not appearing until three months afterward (Federal Reserve Bank of San Francisco 2005). This delay is related to the complexity of calculating GDP relative to other indicators, with its myriad of sources and adjustments. Policymakers can turn to other, faster-publishing indicators for a quicker look into the state of the economy, such as prices or industrial production. But GDP's comprehensive nature, which simultaneously delays its publication, is what makes it desirable as an indicator.

Given these characteristics, the utility of applying nowcasting to the case of GDP becomes clear. Nowcasting is the estimation of the current or near-current value of a target variable using data that are available more quickly. In essence, series that are available in a timelier manner than GDP can be used to estimate a model using historical data, when data for both GDP and these independent series are available. This model can then be used to obtain estimates for GDP well before advanced estimates are available, even while the quarter for prediction is ongoing. GDP's publication lag and salience as an indicator have rendered it perhaps the most common target variable for nowcasting applications. This makes it an ideal choice for a benchmark analysis comparing different nowcasting approaches.

The need for such an analysis stems from the existence of several disparate methodologies specifically employed for nowcasting, in addition to other commonly used econometric and machine learning techniques. A new practitioner, or an experienced practitioner looking to improve their models or expand to different datasets, is currently hard-pressed to find a starting point or an overview of nowcasting approaches when making their modeling decisions, though some existing papers will be discussed in Sect. 2. The need to refer to several papers, which may be applied to different datasets, to get a sense of different methodologies' performance and characteristics can obfuscate conclusions. It is thus the goal of this paper to consolidate results of the most common statistical, econometric, and machine learning methodologies in nowcasting using the closest thing to a benchmark dataset available in the field, nowcasting quarterly US GDP growth using explanatory variables from the Federal Reserve of Economic Data (FRED) as specified in Bok et al. (2018). This dataset has the additional benefit of a long publication history, dating back to 1947. This allows performance to be assessed over a 20-year span from 2002 to 2022. This period, containing both recessionary, volatile, and calm periods in US economic history, is used to test the performance of 17 different methodologies in nowcasting GDP growth. Detailed information on each is provided in Sect. 2.2. In alphabetical order, they are:

Autoregressive moving average (ARMA)
Bayesian mixed-frequency vector autoregression (Bayesian VAR)
Decision tree
DeepVAR
Dynamic factor model (DFM)
Elastic net
Gradient boosted trees
Lasso
Long short-term memory artificial neural networks (LSTM)
Midasml
Mixed data sampling regression (MIDAS)
Mixed-frequency vector autoregression (MF-VAR)
Multilayer perceptron feedforward artificial neural networks (MLP)
Ordinary least squares regression (OLS)
Random forest
Ridge regression
XGBoost

The primary goal of this paper is not only to shed light on these methods' relative performance and characteristics in nowcasting, but also to enable practitioners to take these findings and apply them to their own data. Consequently, an accompanying open-source repository has been created where boilerplate code in Python or R for each methodology can be found and easily adapted to different datasets (Hopp 2022b). With this tool, the barrier to trying out nowcasting on different applications is lowered. Additionally, trying out multiple methodologies to validate results and increase the chances of obtaining a well-performing model is simplified.

The rest of the paper will proceed in the following manner: Sect. 2 will provide further background on nowcasting and the methodologies employed; Sect. 3 will detail the data used; Sect. 4 will explain the modeling approach and how results were obtained; Sect. 5 will display and discuss results; and Sect. 6 will conclude.

2 Background

2.1 Nowcasting

The term nowcasting was first coined and applied in the meteorological domain in the early 1980 s to describe weather forecasting of the near future with information on current meteorological conditions (WMO 2017). It did not begin appearing in economic literature until the mid-2000 s, where the term became popularized after the publication of Giannone et al. (2005). The concept of obtaining real-time, data-based estimates of the macroeconomic situation predates 2005, however, with Mariano and Murasawa (2003) developing a coincident business cycle index based on monthly and quarterly series. This application was already very similar to what is considered economic nowcasting today, but did not directly nowcast GDP, rather equating its synthesized business index to "the smoothed estimate of latent monthly real GDP" (Mariano and Murasawa 2003). Post-2005, a wealth of papers began to be published

examining nowcasting different combinations of indicators, most commonly GDP, and geographies. Examples include Portuguese GDP (Morgado et al. 2007), European GDP (Giannone et al. 2009), global trade (Cantú 2018; Guichard and Rusticelli 2011), and German GDP (Marcellino and Schumacher 2010).

A further differentiating axis in the nowcasting literature, and that of primary concern in this paper, is the methodological approach employed. The most commonly used approach in nowcasting is perhaps the DFM, which is employed in a wealth of papers, including in Giannone et al. (2005). Other examples include Antolin-Diaz et al. (2020), Cantú (2018), and Guichard and Rusticelli (2011), among many others. Other commonly employed approaches include MIDAS (Kuzin et al. 2009; Marcellino and Schumacher 2010), MF-VAR (Kuzin et al. 2009), Bayesian VAR (Cimadomo et al. 2020), LSTMs (Hopp 2022a, c), and many more. Each of these methodologies has characteristics which make them suitable for use in the nowcasting context, which comes with its own particular data challenges, discussed below. But Richardson et al. (2021) additionally examined using common machine learning algorithms in predicting New Zealand GDP growth.

Other nowcasting papers using or comparing multiple methodologies at once include Barbaglia et al. (2022), Carriero et al. (2019), Jansen et al. (2016), and Kronenberg et al. (2021). In the case of Barbaglia et al. (2022), the use of big data and alternative data is examined in nowcasting GDP during COVID in four large European economies. Several methodologies are used, such as Bayesian VAR, MIDAS, and the DFM, but their outputs are combined rather than compared. Carriero et al. (2019) also looks at the performance of big data (100+ indicators) compared with the more usual 13–14 used in nowcasting to predict GDP and inflation in seven different countries. Methodologies compared include MIDAS and Bayesian VAR, but no machine learning techniques. Jansen et al. (2016) compares 11 analytical methodologies and one heuristic one in nowcasting euro area GDP from 1995–2011, coming closest in substance and approach to this paper. However, it does not include any machine learning techniques either. It also does not include performance during COVID in its results. Kronenberg et al. (2021) produces nowcasts for several European countries via forecast combination of DFM and MIDAS models, but focuses on model combination rather than comparison. These papers also lack a key feature of this contribution, namely the easy access of clear, generalizable code required to both reproduce results and apply any of their examined methodologies to novel datasets with step-by-step instructions.

It is drawing upon this literature that the 17 methodologies examined in this analysis were chosen: Bayesian VAR, DFM, LSTM, MF-VAR, MIDAS, and MLP were all included as they appear frequently in the nowcasting literature; ARMA was included as a baseline model; OLS, elastic net, Lasso, and ridge regression were included as perhaps the most popular regression technique in the case of the first and as augmentations of OLS which could render it more suitable for nowcasting in the case of the subsequent approaches; the decision tree, gradient boosted trees, random forest, and XGBoost were included as four popular machine learning techniques (Sarker 2021). DeepVAR and Midasml were included as newer methodologies combining econometric and machine learning approaches with promise for nowcasting (Babii et al. 2022; Salinas et al. 2020).

As mentioned earlier, nowcasting comes with its own set of data challenges, which each methodology needs to be able to handle. Details of this process for each methodology are outlined in the next section. The first challenge is mixed-frequency data, where all variables in the model are not recorded in the same frequency. In this analysis, for example, a mixture of quarterly and monthly variables was used to estimate a quarterly variable, GDP growth. The second is "ragged edges," or differences in missing variables at the end of series due to different publication schedules for each. The model needs some way to be able to handle partially complete data at the ends of time series. The third is the "curse of dimensionality," where there may be relatively more input variables to a model compared with training observations (Buono et al. 2017). This can lead to estimation and other errors in some methodologies, such as causing multicollinearity in OLS. If nowcasting is ever to leverage the power of big data and not be restricted to a handful of explanatory variables, this last challenge will be of particular importance.

2.2 Methodologies

The following sections will provide background information as well as references for further reading for each methodology. Due to the quantity of methodologies included in the analysis, it is not possible to include a comprehensive explanation of each. In-depth explanations of this nature are available via the references. The particular programming implementations utilized for each will also be discussed. The value for any hyperparameter not listed is the default value for the particular library. For information on how mixed frequencies, ragged edges, and time dependencies are handled across models, see Sect. 4. Methodologies are presented in alphabetical order.

2.2.1 ARMA

ARMA models are the simplest nowcasting approach examined in this paper, modeling a time series in terms of two main elements: an autoregressive (AR) component, where future values in a series are a function of its own p prior values, and a moving-average (MA) component, where the error terms of the series are a function of q prior error terms. For more information on the use of ARMA models for modeling stationary time series, see Mills (2019). ARMA is a univariate approach for modeling a time series, meaning that, unlike the other 16 methodologies, the ARMA model included only GDP growth's own history as an input variable. This parsimonious nature makes the model an attractive and commonly used benchmark in nowcasting applications, such as in Cimadomo et al. (2020) or Richardson et al. (2021).

For this analysis, the *auto.arima* function of the *pmdarima* (Smith 2021) Python library was used to determine the p and q lag orders of the ARMA model on the training set. See Sect. 3 for more information on the meaning of "training set." The *auto.arima* function determines the lag orders by fitting models with different lag permutations and recording their Akaike information criteria (AIC), then selecting the orders which minimize this value. See Liew (2004) for more information on AIC as well as lag

selection in ARMA models in general. The *ARIMA* function of *pmdarima* was then used to fit and generate final predictions.

2.2.2 Bayesian mixed-frequency vector autoregression

Standard vector autoregression (VAR) is similar to the univariate AR model discussed previously in Sect. 2.2.1, but generalized to consider multiple time series. Whereas in the univariate case, a variable's value is a function of its p prior values, in the multivariate case, a set of variables' prior values are a function of the set's prior values. Essentially, a vector is considered in the modeling rather than a scalar. For more information on VAR models, see Stock and Watson (2001).

In contrast to standard VARs, discussed further in Sect. 2.2.12, where model parameters are estimated and taken as fixed values, Bayesian VARs consider the parameters as random variables with an assigned prior probability. This approach helps to mitigate over-parameterization, the third data issue discussed in Sect. 2.1. Standard VARs struggle with this issue due to the high number of parameters required to estimate them, usually restricting their use to applications with less than 10 input variables (Bańbura et al. 2010). The introduction of Bayesian shrinkage has been shown to increase forecast accuracy in VAR models with as little as six input variables, many fewer than may be found in a typical nowcasting model. For more information on the concepts of Bayesian shrinkage and Bayesian statistics as applied to regression problems, see De Mol et al. (2008). Bayesian VAR's power in modeling complex dynamic systems and in handling over-parametrization and collinearity have made them a popular choice in the field of nowcasting, see, for instance, Bańbura et al. (2010), Cimadomo et al. (2020), or Schorfheide and Song (2015).

For this analysis, the *estimate_mfbvar* function of the *mfbvar* (Ankargren et al. 2021) R library was used to estimate and predict on a Bayesian VAR model. A Minnesota prior coupled with the inverse Wishart prior for the form of the error variance-covariance matrix was used, as performed in Cimadomo et al. (2020).

2.2.3 Decision tree

The decision tree is a commonly used, nonparametric algorithm in machine learning, due in part to its simplicity and interpretability. Decision trees are often employed as part of an ensemble approach, combining many decision trees as weak learners to produce a strong learner. Three of those tree-based ensemble approaches, gradient boosted trees, random forest, and XGBoost, will be examined in the coming sections. The basic premise of a decision tree does not differ from the standard semantic interpretation of the term; all data begin as one group at the "root" of the tree and are then split into "branches" at different nodes depending on their characteristics and the information gain from that split. This splitting can be very general, with, for instance, only a single split, or, at its most extreme, continuing until every observation sits alone on its own "leaf." Decision trees are normally not equipped to handle time series data, see Sect. 4 for more information on how this was addressed for the decision tree and other methodologies which do not natively handle time series. Simple decision trees have not been used for nowcasting, though tree-based ensembles in nowcasting were

examined in Soybilgen and Yazgan (2021) or Tiffin (2016). For more information on decision trees, see Patel and Prajapati (2018) or Scikit learn (2021a).

Decision trees have hyperparameters which usually need to be tuned depending on the application. In machine learning, hyperparameter refers to parameters which determine the macrostructure of a model and not the model's coefficients and parameters themselves. An example with decision trees is the max depth of the tree, or the number of splits the tree can have, which is then a given condition of the structure of the model independent of the data used to train it. Coefficients within the model, i.e., how to split the data, are then determined from the training data the model is fit with. Hyperparameter tuning refers to the process of establishing a performance metric, e.g., mean absolute error (MAE) or root-mean-square error (RMSE) in a regression application, testing out different hyperparameter combinations, recording their performance according to the performance metric, and selecting a final value for the algorithm's hyperparameters. For more information on hyperparameter tuning, see Probst et al. (2018).

For this analysis, the *DecisionTreeRegressor* function of the *sklearn* (Scikit learn 2021c) Python library was used. See Sect. 4 for more information on how hyperparameters were selected and the library's documentation for more information on specific hyperparameters. Hyperparameters in the final model were:

Hyperparameter	Value
Months lag included in the model	2
Criterion	Absolute error
Max depth	None
Minimum samples split	0.05
Minimum samples in a leaf	0.02

2.2.4 DeepVAR

The DeepVAR methodology gets its name from a portmanteau of "Deep" and "VAR." The former comes from "deep learning," i.e., relating to artificial neural networks, while the latter comes from vector autoregression. See Sect. 2.2.12 for more information on vector autoregression. It is a derivative of the DeepAR methodology adapted for use with multiple time series. The DeepAR methodology was first introduced in Salinas et al. (2020), initially envisioned as a forecasting tool for the business context, e.g., forecasting demand to plan inventory levels. The approach generates probabilistic forecasts via an autoregressive recurrent neural network (RNN). See Sect. 2.2.9 for more information on RNNs. Until now, applications of and research into DeepAR have focused primarily around pure forecasting. Its suitability for nowcasting has not been extensively explored until the writing of this paper.

For this analysis, the *DeepVAREstimator* function of the *gluonts* (Gluonts 2022) Python library was used to estimate and predict a DeepVAR model. See Sect. 4 for more information on how hyperparameters were selected and the library's documentation for more information on specific hyperparameters. Hyperparameters in the final model were:

Hyperparameter	Value
Months lag included in the model	12
Number of layers	4
Number of cells	40
Dropout rate	0.01
Learning rate	0.0001
Batch size	100
Epochs	10

2.2.5 Dynamic factor model

Dynamic factor models (DFMs) are commonly used in time series forecasting and nowcasting. They operate under the assumption that one or several latent underlying factors explain the development of multiple time series, which are a product of these latent factors plus an idiosyncratic error term. The latent factor could represent, for instance, the business cycle. DFMs are often estimated by assigning variables to various "blocks" (i.e., the factors), which can represent different aspects of the economy. For example, when nowcasting GDP, all variables can be assigned to one global block, while a subset of variables can be assigned to another block, representing, e.g., a geography or economic sector. For more information on the use of blocks in estimating DFMs, see Hallin and Liška (2011).

DFMs are one of the most commonly applied methodologies in nowcasting. Some examples include nowcasting Canadian GDP growth (Chernis and Sekkel 2017), global trade growth (Cantú 2018; Guichard and Rusticelli 2011), Russian GDP growth (Porshakov et al. 2016), and German GDP growth (Marcellino and Schumacher 2010), among many others. For more information on DFMs, see any of Bok et al. (2018), Cantú (2018), Giannone et al. (2005), or Stock and Watson (2002).

For this analysis, the *dfm* function of the *nowcastDFM* (Hopp and Cantú 2020) R library was used. This library was developed as an R implementation of the original MATLAB code published alongside Bok et al. (2018). This implementation of the DFM is modeled in state-space form under the assumption that all variables share common latent factors in addition to their own idiosyncratic components. Subsequently, the Kalman filter is applied and parameter estimates are obtained via maximum likelihood estimation. For more information on this particular modeling approach, see Bańbura and Rünstler (2011), Bok et al. (2018), and Cantú (2018). Blocks used were the same specified in Bok et al. (2018), representing global, soft, real and labor factors. Using a single, global block was also assessed, but was found to achieve worse results than Bok et al. (2018)'s block specification.

2.2.6 Elastic net

Elastic net is very similar to the OLS approach examined in Sect. 2.2.14. However, it seeks to address one of the principal issues with OLS: multicollinearity. Elastic net adds regularization to OLS through the linear combination of the L_1 and L_2 penalty terms. The L_1 regularization term penalizes the sum of absolute values of the model's

weights, while the L_2 term penalizes the sum of squares of weights. Regressions utilizing just the L_1 penalty term are called Lasso regressions, while those utilizing just the L_2 term are called ridge regressions. Lasso can reduce features' weights to zero, while ridge regression merely reduces them. As a result, Lasso is frequently used for variable selection. Elastic net seeks to leverage the features of both penalty terms, able to completely eliminate features as well as reduce their contribution.

The model then seeks to minimize not only the residual sum of squares, but also this penalty term. In this way, coefficients more often tend toward zero or are eliminated completely. Elastic net allows the inclusion of more input variables than standard OLS, an important characteristic for nowcasting. This penalty term necessitates the selection of two hyperparameter at estimation, an alpha term denoting the strength or degree of regularization as well as a term defining the ratio of L_1 to L_2 weight. This is the nomenclature used in the scikit-learn library implementation used for the analysis. It is used here for clarity when referring to the code. Other sources may refer to "alpha" as "lambda" and "L_1 ratio" as "alpha." For more information on elastic net in the context of nowcasting, see Eickmeier and Ng (2011). For more information on elastic net in general, see Zou and Hastie (2005).

For this analysis, the *ElasticNet* function of the *sklearn* (Scikit learn 2021c) Python library was used to estimate and predict with an elastic net model. See Sect. 4 for more information on how hyperparameters were selected and the library's documentation for more information on specific hyperparameters. Hyperparameters in the final model were:

Hyperparameter	Value
Months lag included in the model	4
Alpha	0.00001
L_1 ratio	0.25

2.2.7 Gradient boosted trees

Gradient boosted trees are an ensemble machine learning algorithm combining the result of several simpler decision tree models (see Sect. 2.2.3), similar to the random forest method discussed in Sect. 2.2.15. Gradient boosted trees combine individual decision trees sequentially, with each addition trained to reduce the errors of the previous iteration. The modeling approach has been shown to be powerful and high-performing in a variety of applications, counting as one of the most commonly winning algorithms in the Kaggle data science competition (Boehmke 2018). For more information on the gradient boosting approach, see Natekin and Knoll (2013).

For this analysis, the *GradientBoostingRegressor* function of the *sklearn* (Scikit learn 2021c) Python library was used. See Sect. 4 for more information on how hyperparameters were selected and the library's documentation for more information on specific hyperparameters. Hyperparameters in the final model were:

Hyperparameter	Value
Months lag included in the model	5
Loss	Absolute error
Number of estimators	100
Minimum samples split	0.05
Minimum samples in a leaf	0.05
Max depth	6
Max features	Square root

2.2.8 Lasso

Lasso is very similar to the elastic net described in Sect. 2.2.6, except only the L_1 penalty term is used. It is therefore identical to elastic net with an L_1 ratio of one. It only has one hyperparameter, an alpha term determining the degree of regularization. See Tibshirani (1996) for more information on Lasso regression.

For this analysis, the *Lasso* function of the *sklearn* (Scikit learn 2021c) Python library was used to estimate and predict with a Lasso model. See Sect. 4 for more information on how hyperparameters were selected and the library's documentation for more information on specific hyperparameters. Hyperparameters in the final model were:

Hyperparameter	Value
Months lag included in the model	4
Alpha	0.00001

2.2.9 Long short-term memory artificial neural network

Long short-term memory artificial neural networks (LSTMs) are a sub-class of artificial neural network (ANN). For more information on ANNs, see Sect. 2.2.13. The traditional feedforward network discussed in that section has information flowing through it unidirectionally, from the beginning of the network to the end. Recurrent neural networks (RNNs) were introduced to make ANNs more suitable for time series and situations where there is a temporal dependency in the data, for instance, in applications like speech processing. RNNs introduce a feedback loop to the traditional ANN architecture, where outputs of layers can be fed back into the network before a final output is obtained. For more information on RNNs, see Dematos et al. (1996) or Stratos (2020). Due to vanishing and exploding gradients, RNNs tend to have a short memory and thus are of limited use in the nowcasting context. For more information on vanishing and exploding gradients in RNNs, see Grosse (2017). LSTMs address this deficiency by introducing a memory cell and an input, output, and forget gate. Gradients are then able to flow through the network unchanged, allowing LSTMs to establish longer time dependencies than RNNs. For more information on LSTM architecture, see Brownlee (2018) or Chung et al. (2014).

Though LSTMs have been used before for nowcasting meteorological events (Shi et al. 2015), their fitness for economic nowcasting has only recently begun to be explored. One recent article examined their suitability in nowcasting global trade, finding them to produce superior results to DFMs (Hopp 2022a). An open-source library for nowcasting economic data using LSTMs in multiple programming languages was published alongside that paper. A second article compared their performance with the DFM in nowcasting global trade during the COVID crisis, again with the LSTM more often producing superior performance (Hopp 2022c).

For this analysis, the *LSTM* function of the *nowcast_lstm* (Hopp 2021) Python library was used. See Sect. 4 for more information on how hyperparameters were selected and the library's documentation for more information on specific hyperparameters. Hyperparameters in the final model were:

Hyperparameter	Value
Months lag included in the model	6
Fill NA function	Mean
Fill ragged edges function	Mean
Number of models	10
Train episodes	100
Batch size	50
Decay	0.98
Number of hidden states	10
Number of LSTM layers	1
Criterion	Mean squared error loss
Optimizer	Adam
Learning rate	0.01

2.2.10 Midasml

The Midasml methodology is a novel nowcasting approach which builds on MIDAS models by introducing Lasso regularization. See Sect. 2.2.11 for more information on MIDAS models. The methodology, along with an accompanying R library, was introduced in Babii et al. (2022). In the paper, the authors also carried out a nowcasting exercise predicting quarterly US GDP growth. They found the Midasml approach to outperform the DFM used by the Federal Reserve Bank of New York.

For this analysis, the *cv.sglfit* function of the *midasml* (Striaukas et al. 2022) R library was used to estimate and predict a Midasml model. See Sect. 4 for more information on how hyperparameters were selected and the library's documentation for more information on specific hyperparameters. Hyperparameters in the final model were:

Hyperparameter	Value
Low frequency lags	4
High frequency lags	9
Gamma	0.25
Legendre polynomial degree	3

2.2.11 Mixed data sampling regression

Introduced in 2004, the MIDAS framework was developed to address the issue of estimating time series regression models where the dependent variable is sampled at a lower frequency than the independent variables (Ghysels et al. 2004). Similar to mixed-frequency VAR, MIDAS can suffer from over-parametrization due to the necessity of including the lags of higher frequency variables in the model. To avoid this issue and obtain parsimony, MIDAS regression frequently employs a nonlinear weighting scheme for the lag coefficients (Kuzin et al. 2011). MIDAS' ability to handle mixed-frequency data as well as mitigate over-parametrization make them well suited for nowcasting, where the methodology has been employed for instance in nowcasting euro area GDP (Kuzin et al. 2009) or German GDP (Marcellino and Schumacher 2010).

For this analysis, the *midas_r* function of the *midasr* (Kvedaras 2021) R library was used employing an exponential Almon weighting function for lag coefficients. As specified in Kuzin et al. (2009), separate univariate models were estimated for each independent variable whose forecasts were then combined via a weighted mean. Weights were found using RMSE of the model on the training set, adjusted to discount the worst-performing univariate model, so that this model had no contribution to the combined forecast. See Clements and Galvão (2008) for more information on forecast combination in MIDAS models.

2.2.12 Mixed-frequency vector autoregression

As stated in Sect. 2.2.2, vector autoregression estimates a set or vector of variables as a function of the vector's own prior p values. Originally introduced in 1980 (Sims 1980), VAR models have become popular in the domain of macroeconomic forecasting due to their ability to capture the dynamics of complex systems using multiple time series (Stock and Watson 2001). They do, however, come with several limitations which limit their usefulness in the nowcasting context. Firstly, they do not natively handle mixed-frequency data. Secondly, they often suffer from over-parametrization, where the inclusion of additional variables leads to exponentially more parameters (Kuzin et al. 2009). The first issue can be addressed by three different approaches. One common approach is aggregating higher frequency indicators to the frequency of the lowest frequency indicator. Taking the case of estimating a VAR for a quarterly growth rate using monthly variables, the monthly variables can be transformed to a quarterly frequency by for instance averaging the growth rates for the three constituent months, or by calculating the full quarterly growth rate from the monthly values. A second approach is stacking the constituent months of a quarter into three separate series, so that one monthly series becomes three quarterly series composed of the time

series for months one, two, and three of each quarter. A final approach is estimating the model in the highest frequency of the dataset and interpolating lower-frequency variables into this higher frequency (Ghysels 2016). Any of these approaches yields a mixed-frequency VAR model (MF-VAR).

MF-VAR models have been used in nowcasting applications before, for instance, for nowcasting euro area GDP (Kuzin et al. 2009). However, due to the aforementioned issues with over-parametrization, Bayesian VARs are more often applied. See Sect. 2.2.2 for more information on how Bayesian VARs help address this issue.

For this analysis, the *VAR* function of the *PyFlux* (Taylor 2016) Python library was used to estimate and predict the MF-VAR model. See Sect. 4 for more information on how hyperparameters were selected and the library's documentation for more information on specific hyperparameters. Hyperparameters in the final model were:

Hyperparameter	Value
Months lag included in the model	3

2.2.13 Multilayer perceptron feedforward artificial neural network

Artificial neural networks (ANNs) have become extremely popular in recent years due to their exceptional performance in a variety of applications, ranging from image recognition to speech processing to self-driving cars. ANNs are made up of interconnected layers of nodes which receive data inputs, either from an external data source or from a previous layer, run the data through a series of weights or coefficients and a nonlinear activation layer, and then generate an output. This output can either be the final prediction of the model or serve as an input to a future layer. The model is trained by defining a cost function, calculating gradients with respect to this cost function, then adjusting weights in the direction of minimizing error according to the cost function. This process is called back propagation. The most common type of ANN is the feedforward multilayer perceptron (MLP), where information flows unidirectionally through the network. For more information on ANNs, see Sazli (2006) or Singh and Prajneshu (2008).

ANNs have been used for economic and forecasting applications, with good results. Examples include nowcasting US GDP (Loermann and Maas 2019), forecasting commodity prices (Kohzadi et al. 1996), and forecasting exchange rates (Falat and Pancikova 2015).

For this analysis, the *MLPRegressor* function of the *sklearn* (Scikit learn 2021c) Python library was used to estimate and predict with an MLP network. See Sect. 4 for more information on how hyperparameters were selected and the library's documentation for more information on specific hyperparameters. Hyperparameters in the final model were:

Hyperparameter	Value
Months lag included in the model	3
Hidden layer sizes	(100, 200)
Activation function	ReLU
Solver	Adam
Alpha	0.001
Initial learning rate	0.001
Learning rate type	Constant

2.2.14 Ordinary least squares regression

Ordinary least squares is one of the most commonly employed approaches in regression applications thanks to its simplicity, power, and interpretability. OLS estimates the parameters of a linear function mapping a set of input variables to an output variable. These values are determined by minimizing the sum of squared differences between the actual output variable and the estimate of the linear function. See Mahaboob et al. (2018) for more information on OLS.

Generally, OLS is not well suited for use in the nowcasting context as it contains no explicit provision for time series and often suffers from multicollinearity when higher numbers of input variables are used, rendering its coefficients unstable and violating a base assumption of the model. See Sect. 4 for more information on the adjustments to the data required to use OLS in this analysis.

For this analysis, the *LinearRegression* function of the *sklearn* (Scikit learn 2021c) Python library was used to estimate and predict with the OLS model. Hyperparameters in the final model were:

Hyperparameter	Value
Months lag included in the model	3

2.2.15 Random forest

Random forest is an ensemble machine learning algorithm that combines the results of multiple decision tree models. See Sect. 2.2.3 for more information on decision trees. Individual decision trees often have poor performance on their own, tending to overfit their training data if allowed too many layers. Random forest trains many decision trees on different subsets of the training data (bootstrap samples) or on all the training data and averages their predictions to obtain more generalizable estimates with lower variance. For bootstrap samples to be taken, the input data need to be independent, which is not the case with time series data. There is not currently an implementation of random forest available for dependent/time series data. Random forest models are not generally used in nowcasting, but were examined in Soybilgen and Yazgan (2021) and Tiffin (2016).

For this analysis, the *RandomForestRegressor* function of the *sklearn* (Scikit learn 2021c) Python library was used to estimate and predict with a random forest model. See

Sect. 4 for more information on how hyperparameters were selected and the library's documentation for more information on specific hyperparameters. Hyperparameters in the final model were:

Hyperparameter	Value
Months lag included in the model	4
Number of estimators	100
Criterion	Squared error
Max depth	None
Minimum samples split	0.01
Minimum samples in a leaf	0.01
Maximum features	Square root
Bootstrap	False

2.2.16 Ridge regression

Ridge regression is very similar to the elastic net described in Sect. 2.2.6, except only the L_2 penalty term is used. It is therefore identical to elastic net with an L_1 ratio of zero. It only has one hyperparameter, an alpha term determining the degree of regularization. See Tiffin (2016) for more information on ridge regression in the nowcasting context.

For this analysis, the *Ridge* function of the *sklearn* (Scikit learn 2021c) Python library was used to estimate and predict with a Ridge regression model. See Sect. 4 for more information on how hyperparameters were selected and the library's documentation for more information on specific hyperparameters. Hyperparameters in the final model were:

Hyperparameter	Value
Months lag included in the model	4
Alpha	0.1

2.2.17 XGBoost

XGBoost is an ensemble machine learning algorithm which is very similar to gradient boosted trees. As such, see Sect. 2.2.7 for more information on the approach's fundamentals. XGBoost differs from gradient boosted trees in some implementation and programming elements (parallel processing, improved data structures, etc.). Methodologically, the main difference is XGBoost's introduction of regularization. For more information on XGBoost, see Chen and Guestrin (2016).

For this analysis, the *XGBRegressor* function of the *XGBoost* (XGBoost Developers 2022) Python library was used. See Sect. 4 for more information on how hyperparameters were selected and the library's documentation for more information on specific hyperparameters. Hyperparameters in the final model were:

Hyperparameter	Value
Months lag included in the model	7
Number of estimators	1000
Eta	0.1
Max depth	3
Lambda	1
Alpha	0

3 Data

Data for US GDP, the target variable, and all explanatory variables were obtained from the Federal Reserve Economic Data (FRED) API (Boysel and Vaughan 2021). Explanatory variables were those specified in Bok et al. (2018), encompassing a variety of monthly and quarterly economic indicators such as goods and services exports, building permits, various price indices, and retail sales, among others. All series were obtained from FRED already seasonally adjusted, where applicable. All series were then transformed to period over period growth rates. Data for GDP range from the first quarter of 1947 to the third quarter of 2022. Availability start dates for explanatory variables range from January 1947 to November 2009. It was decided to use the latest available data for the analysis rather than real-time data. Real-time data refer to the data as it appeared in the past. The latest available data may be subject to later revisions. This decision was taken due to the fact that historical vintages are not available for many datasets. Developing the testing and training structure around their availability would severely restrict the generalizability of the resources this analysis provides. A structure that relies only on the latest data coupled with publication lags for series to create artificial vintages is generalizable to any nowcasting application, as well as to new data sources where real-time historical data may not be available.

Before proceeding, it is necessary to define the terms training, validation, and test periods or sets. These terms refer to an important concept in model evaluation, and machine learning in general, of assessing model performance on data the model was not trained on. The training set refers to the data used to train or estimate the model, while the test set refers to the data the trained model's performance is ultimately evaluated on. The purpose of splitting the data in these sets is to ensure the generalizability of the model and to avoid overfitting. A model may perform very well on predicting values from the data it was trained on, but may generalize very poorly to new data it has not seen before. The validation set is similar to the test set, but is used exclusively in selecting hyperparameters for algorithms that require them. The idea is to further divide the training set into a validation set so that different hyperparameters can be tested and selected based on their performance in predicting this validation set. Once hyperparameters are selected, the model can be retrained with the full training set and finally assessed on the test set. The logic of utilizing a separate validation and test set is to avoid information leakage and thus overfitting to the test set. A model with hyperparameters selected based on performance on the test set may be overfit to that set and not generalize well. For more information on these concepts, see Scikit learn (2021b).

The validation period for the analysis dated from Q1 1995 to Q4 2001. The test period dated from Q1 2002 to Q3 2022. The training period was a rolling one, varying depending on which time period was being nowcast, e.g., for nowcasting Q1 2002, the training period dated from Q1 1947 to Q4 2001, and so on. The test period contains two of the largest economic shocks in US history, the 2008 financial crisis and the COVID crisis, making it ideal to test models' performance during volatile periods. It also contains numerous tranquil periods, important in ensuring performance during normal, calmer economic times as well. Of the variables specified in Bok et al. (2018), not all had a long enough time series to be included in the test and validation periods. As a result, only variables with a time series dating back to at least Q1 1993 were included in the analysis.

4 Modeling approach

For methodologies that required hyperparameters, they were chosen by constructing grids of candidate hyperparameters, training many models with all combinations of those hyperparameters, then assessing their accuracy via RMSE on the validation set. The best-performing combinations of hyperparameters were then selected.

To obtain the results in Sect. 5, a single model for each methodology outlined in Sect. 2.2 was fit for each quarter in the test period using the hyperparameters (if applicable) obtained via the approach described in the previous paragraph. That is, the models were retrained recursively with the data as it would have appeared the month before the start of the period. This approach simulates the real-world conditions facing a nowcasting application. Model inference was then applied at five different vintages, simulating the data as it would have appeared two months before the target period, one month before, the month of, one month after, and two months after. For instance, if the target period was Q1 2002, the model was trained with the data as it would have appeared in December 2001. The trained model then made predictions on the data as it would have appeared in January, February, March, April, and May 2002. The data "as it would have appeared" refer to generating predictions on simulated data vintages. A simulated data vintage refers to the artificial introduction of missing values to simulate the data as it would have appeared at different points in the past. Testing model performance at different time points is important, as series in a live nowcasting model will rarely have complete data for the quarter at the time of prediction. A nowcast for a given quarter will most likely begin in media res, or even before the quarter has begun. A nowcasting model must be robust to this lack of information earlier in the prediction period. GDP actuals are ostensibly already available two months after the quarter, but this value is an advanced estimate liable to revision, so it is worth evaluating the models even at this advanced time point. Information for publication lags of each of the variables was obtained from Bok et al. (2018).

Another characteristic of this dataset is that not all series have a publication history as long as GDP's. This raises the question of how to handle these series in training the model. There are three approaches that can be taken. The first approach is to simply drop any series with a start date later than the first quarter of 1947, the start date of our target variable. This is an approach that drops columns or series from our data. The

second is to drop any observations earlier than the latest start date of all the series. This is an approach that drops rows from our data. The final approach is imputing the missing values using various techniques, such as mean imputation or expectation maximization (EM) (Soley-Bori 2013). For this analysis, the third approach, filling missing values with the mean, was employed, though all three approaches were tested empirically.

The first, dropping variables with a start date later than the first quarter of 1947, left only five explanatory variables out of the 28 specified in Bok et al. (2018). Performance was consequently worse across all methodologies. The second, dropping observations before the latest variable start date in the dataset, also obtained worse performance across all methodologies. That left the third option, along with the decision of how to fill missing values. Two of the approaches outlined in Soley-Bori (2013) were tested: filling values with the mean and filling them using EM imputation. The two approaches yielded comparable results, with some methodologies performing slightly better with one approach and others with the other. The decision to use mean-filling was then taken based on two factors: the aforementioned comparable empirical performance and for consistency with the filling ragged edges approach outlined below. It is important to note that the missing values were filled with the mean of the series only up to the end of the training data, not the full dataset. This is important to prevent data leakage from the test set to the models.

Ragged edges refer to missing values at the ends of time series in the data due to differing publication schedules and nowcasting at different times throughout the prediction period. For instance, a nowcast for the second quarter performed in May must have all values for June missing, unless the variable is some kind of forecast or plan. There are many different approaches to dealing with ragged edges, and three were tested for this analysis. The first, ultimately chosen, was filling missing values with the series mean. The second was filling missing values with n-step ahead forecasts of univariate ARMA models for each series. The third was using EM imputation. All three approaches had similar empirical performance, again with certain methodologies performing better with one approach and others with a different one, but differences were marginal. The decision to use mean-filling was then taken based on two considerations: first, the aforementioned parity in performance, and a second, practical consideration. Filling missing values with the mean is a very light computational task, occurring nearly instantaneously. EM imputation is a complex quantitative approach whose calculation time scales with both the number of observations and the number of variables. Because each data vintage, that is, combination of target period and simulated lag, has a unique pattern of missing values, this requires a run of the algorithm for each one. This quickly becomes computationally expensive and impractical for assessing model performance. Imputing ragged edges on one data vintage via the EM method takes roughly 3.5 min, for instance. This needs to be performed for 83 quarters and five lags, resulting in 415 different data vintages. This translates to a run time of more than 24 h. This process needs to be rerun each time a variable is added or removed from the model, a frequent occurrence in the model selection process. The situation is similar, though not as severe, for ARMA ragged edge filling. An ARMA model needs to be fit for each data vintage, though need not be rerun if variables are added or removed from the model, as the series models are univariate. Consequently,

mean-filling was selected for dealing with ragged edges for all methodologies, improving comparability. The sole exception was the DFM, where use of EM imputation in combination with the Kalman filter is an essential component of the implementations in Bok et al. (2018) and Hopp and Cantú (2020).

The final data issue to address in the analysis was how to handle mixed-frequency time series for the methodologies which do not natively handle them. As described in Sect. 2.2.12, there are three approaches that can be taken. Higher-frequency indicators can be aggregated to a lower frequency, they can be stacked into separate series, or lower-frequency indicators can be converted to a higher frequency, with missing values interpolated. For this analysis, the first two approaches were tested, as a good implementation of the interpolation method detailed in Kuzin et al. (2011) could not be found. For the first approach, two methods of aggregation were tested: taking the average growth rate of the three months constituting a quarter and calculating the full quarterly growth rate via each individual month's growth rate. Stacking monthly series was found to perform better than aggregating across methodologies, so was the approach utilized. Methodologies where stacking the time series was relevant were the decision tree, elastic net, gradient boosted trees, Lasso, MF-VAR, MLP, OLS, ridge regression, random forest, and XGBoost.

5 Results

5.1 Numerical and graphical results

Tables 1 and 2 display MAE and RMSE of each methodology as a proportion of the benchmark ARMA model's MAE and RMSE. Additional tabular results can be found in Appendix A.1. Additional graphical results can be found in Appendix B.2.

Table 3 displays the average revision between two data vintages of each methodology. For instance, we can see that when moving from one data vintage to another, i.e., more data are available to the model, ridge regression revised its nowcast by 0.18 percentage points, while OLS' revisions were on average 83% higher, at 0.33 percentage points. An ideal model would have low values in Tables 1 and 2, indicating good predictive performance, and a low value in Table 3, indicating lower month-to-month revisions and more consistent predictions.

With five different data vintages and accounting for revision volatility, it can be difficult to emerge with an overall picture of relative performance between the models. Figures 1 and 2 plot average MAE and RMSE over the five vintages on the Y-axis and average revision on the X-axis. The ideal model would appear in the lower left corner, with low MAE and RMSE and low revisions. Conversely, a poor model would appear in the upper right corner, achieving high MAE and RMSE while being extremely volatile. In terms of both MAE and RMSE, the two best performers were the LSTM and Bayesian VAR. It is notable, however, that the LSTM achieved a similar level of accuracy to the Bayesian VAR with significantly lower (less than half) revisions. In fact, Bayesian VAR was the most volatile of the 17 methodologies, rivalled only by MF-VAR and OLS. Other relatively well-performing models included Midasml, MIDAS, and elastic net.

Table 1 MAE as a proportion of ARMA model

Vintage	−2	−1	0	1	2
Bayesian VAR	0.90	0.76	0.60*	0.65*	0.53*
Decision Tree	1.00	1.01	1.06	1.02	0.92
DeepVAR	0.91	0.93	0.92	0.88	0.88
DFM	0.92	0.84	0.83	0.83	0.85
Elastic net	0.93	0.78	0.77	0.81	0.62
Gradient boost	0.92	0.91	0.88	0.79	0.74
Lasso	0.99	0.84	0.83	0.84	0.70
LSTM	0.90	0.69*	0.69	0.67	0.67
MF-VAR	0.95	0.85	0.80	0.85	0.73
MIDAS	0.89*	0.79	0.77	0.74	0.62
Midasml	0.94	0.76	0.72	0.73	0.73
MLP	1.06	0.88	0.91	0.89	0.86
OLS	0.94	0.86	0.79	0.88	0.66
Random forest	0.90	0.89	0.86	0.79	0.74
Ridge	0.98	0.92	0.89	0.87	0.86
XGBoost	0.91	0.89	0.88	0.85	0.69

This table shows MAE on the test set as a proportion of the ARMA model's MAE per data vintage. A value higher than 1 denotes worse performance than the ARMA model, and a value lower than 1 denotes better performance
*Marks the best-performing model in the vintage (column)

To ensure the statistical robustness of the results, the Diebold–Mariano test for predictive accuracy was performed pairwise for each methodology combination at each data vintage. Figure 3 displays the results as a stacked bar plot, ordered from most pairwise significance to least, on average. For instance, a full bar of the 10% significance level dark gray says that the test found the model, at the 10% significance level, to produce better forecasts than every other methodology in a head-to-head comparison. A full bar of the light gray of no significance, on the other hand, says that the methodology was not found to produce better forecasts than any of the other methodologies, as is the case with the ARMA model. Additional matrices and heatmaps of each of the head-to-head test statistics are available in Appendix B.2 (Figs. 21, 22, 23, 24, 25). The tests' findings support the conclusions from Figs. 1 and 2, with the LSTM and Bayesian VAR achieving the highest level of significance, followed more distantly by MIDAS and elastic net. General observations follow in the next section.

5.2 Overall results commentary

On the aggregate, methodologies which natively handle time series performed better than those that did not. Compare the performance of methodologies like the LSTM, Midasml, and Bayesian VAR with that of the tree-based methods or the MLP. There were, however, exceptions to this trend. For example, elastic net was a good performer while treating lags as independent. This implies that, while helpful, modeling time

Table 2 RMSE as a proportion of ARMA model

Vintage	−2	−1	0	1	2
Bayesian VAR	0.79	0.57	0.31*	0.34*	0.30*
Decision Tree	0.97	0.88	0.89	0.85	0.71
DeepVAR	0.94	0.84	0.85	0.83	0.82
DFM	0.75*	0.51	0.50	0.52	0.54
Elastic net	0.92	0.56	0.54	0.50	0.35
Gradient boost	0.94	0.82	0.82	0.78	0.74
Lasso	0.95	0.59	0.56	0.53	0.44
LSTM	0.81	0.40*	0.38	0.35	0.34
MF-VAR	0.94	0.66	0.59	0.53	0.50
MIDAS	0.86	0.56	0.55	0.56	0.42
Midasml	0.84	0.46	0.44	0.44	0.44
MLP	0.89	0.55	0.52	0.50	0.48
OLS	0.93	0.62	0.53	0.50	0.39
Random forest	0.93	0.80	0.79	0.76	0.73
Ridge	0.90	0.68	0.65	0.60	0.62
XGBoost	0.93	0.82	0.82	0.77	0.63

This table shows RMSE on the test set as a proportion of the ARMA model's RMSE per data vintage. A value higher than 1 denotes worse performance than the ARMA model, and a value lower than 1 denotes better performance

*Marks the best-performing model in the vintage (column)

dependencies is not essential to achieving good performance in nowcasting. Including lags of prior time periods can be an effective means of generating nowcasts with methodologies lacking time series elements. The fact that the elastic net was the best methodology to employ this technique indicates that regularization may play an important role in moderating the impact of the numerous additional lagged input variables.

Figures 1 and 2 also illustrate a positive relationship between predictive accuracy and volatility. This makes intuitive sense, as a model needs to be able to respond to new signals in the data to have accurate predictions. Only in tranquil periods can a model obtain good performance with minimal revisions. Some models over- and underperform relative to their volatility, however. If we picture a regression line in Figs. 1 and 2, we can see that some methodologies overperform relative to their volatility. Methods like MIDAS and the LSTM achieve better results with lower volatility than expected from the regression line. Methods like MF-VAR and OLS underperform, needing more volatility to achieve similar performance to much less volatile methods like the DFM or gradient boosted trees.

Next, we will discuss the role of nonlinearity in model performance. In the analysis, there are cases of both linear and nonlinear models performing both poorly and well. This prevents us from making a general statement such as "nonlinear models are better in nowcasting applications." The reality is that it depends on other aspects of the model as well. Elastic net and Bayesian VAR show that linear models are perfectly capable

Table 3 Average month-to-month revision (percentage points)

Methodology	Revision (%)
ARMA	0.05
Bayesian VAR	0.39
Decision tree	0.19
DeepVAR	0.08
DFM	0.12
Elastic net	0.2
Gradient boost	0.1
Lasso	0.21
LSTM	0.18
MF-VAR	0.34
MIDAS	0.1
Midasml	0.15
MLP	0.2
OLS	0.33
Random forest	0.08
Ridge	0.18
XGBoost	0.16

This table shows the average month-to-month revision of each methodology in percentage points. That is, how much the methodology's prediction changes when new data are made available to it. For example, if a methodology predicts growth of 1%, 0.5%, 0.25%, 1.2%, and 1.5% growth for the −2, −1, 0, 1, and 2 vintages, that implies changes of 0.5, 0.25, 0.95, and 0.3 percentage points, respectively. This results in an average revision of 0.5 percentage points

of high performance in nowcasting. They also imply that feature regularization may play a more important role than linearity. Conversely, gradient boosted trees and the MLP show that nonlinearity is not a guarantee of high performance. The existence of counterexamples for all three of the primary axes differentiating the models, handling of time series, volatility, and linearity, shows that the best way to ensure a particular methodology is right for a particular dataset is to test it empirically.

A final important consideration to make is the performance of the methodologies in volatile versus tranquil periods. Tables 4, 5, 6, and 7 duplicate the results for Table 2, but for the periods dating from 2002–2007, 2008–2009, 2010–2019, and 2020–2022, respectively. 2002–2007 corresponds to a tranquil period prior to the 2008 financial crisis, 2008–2009 covers said crisis, 2010–2019 covers the tranquil period between the financial crisis and COVID, and 2020–2022 covers the COVID crisis.

Unsurprisingly, no single methodology is the best across all these periods. In the tranquil period from 2002–2007, gradient boost, XGBoost, and the LSTM are the highest performers. From 2008–2009, the LSTM and Bayesian VAR are far and away the best performers, while gradient boost and XGBoost struggle. In the tranquil period from 2010–2019, gradient boost and XGBoost are again the top performers. Finally, during COVID, the LSTM and Bayesian VAR again dominate. This indicates that while

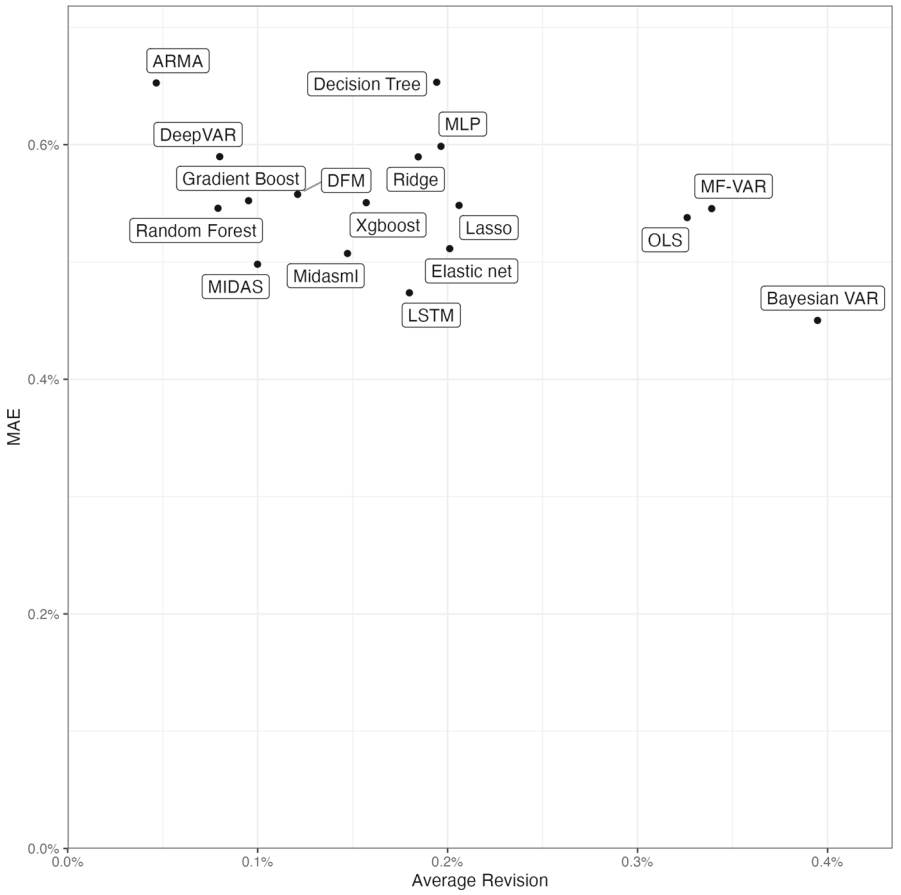

Fig. 1 Average MAE versus average revision. *Note*: This plot shows average revision plotted against average MAE across the five data vintages. A lower value is better for both axes. A value in the top left corner indicates a model that is very stable (does not revise its predictions very much), but not very accurate. A value in the top right corner indicates a model that is neither accurate, nor stable. A value in the bottom right corner indicates a model that is accurate, but very volatile. A value in the bottom left corner is ideal, indicating a model that is both accurate and stable

the boosting methodologies may be quite suitable for placid periods, they are not well suited for handling crises. The LSTM and Bayesian VAR, in turn, are undeniably the best performers by far during the two crisis periods. Each then does very well during one tranquil period, and worse in the other.

In terms of runtime, this can vary wildly for many of the methodologies depending on the hyperparameters chosen. But generally, the time taken to train a model and generate inferences is not significant, taking less than a few seconds, but at most a minute, for almost all of the methodologies. The most notable exception is the DFM, where runtime depends heavily on the speed with which EM convergence occurs. In practice, this can take anywhere from a few seconds to upward of 20 min.

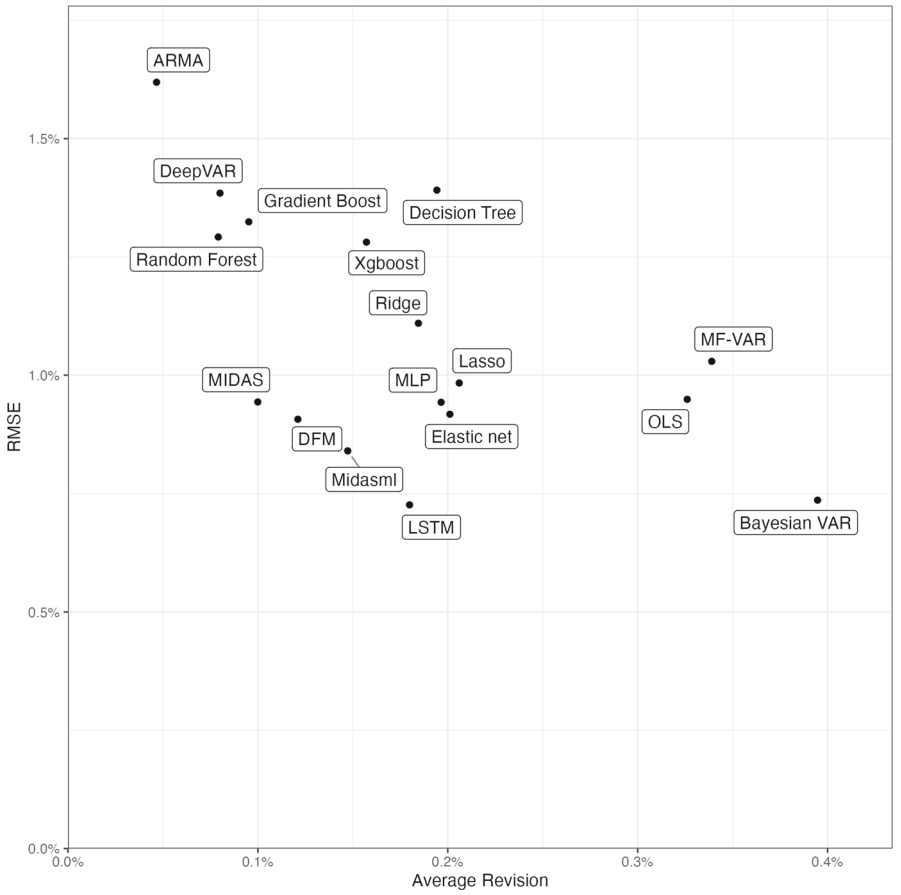

Fig. 2 Average RMSE versus average revision. *Note*: This plot shows average revision plotted against average RMSE across the five data vintages. A lower value is better for both axes. A value in the top left corner indicates a model that is very stable (does not revise its predictions very much), but not very accurate. A value in the top right corner indicates a model that is neither accurate, nor stable. A value in the bottom right corner indicates a model that is accurate, but very volatile. A value in the bottom left corner is ideal, indicating a model that is both accurate and stable

The next section contains detailed observations for each methodology drawing on these quantitative results as well as the graphical outputs in Appendix B.2.

5.3 Individual methodologies' results commentary

ARMA

A detailed plot of the ARMA model's predictions is available in Fig. 4. As the univariate benchmark, the ARMA model was unsurprisingly the worst performer in the analysis. In periods when GDP growth rates hewed very close to their long-term averages with minimal variation, such as between 2012 and 2019, the ARMA model

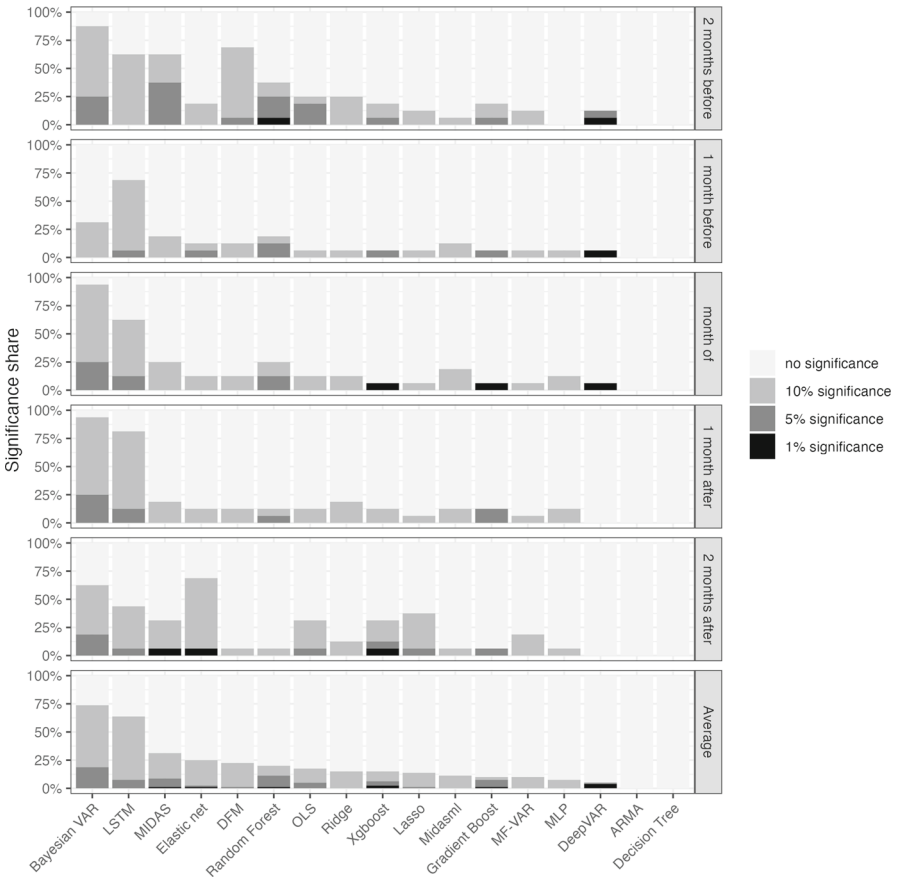

Fig. 3 Share of statistical significance in pairwise Diebold–Mariano tests. *Note*: This plot shows the share of statistical significance in the pairwise Diebold–Mariano tests. An empty bar indicates that the methodology did not obtain statistical significance under the alternative hypothesis that it was a better forecast than another methodology in any case. A full bar (color depending on significance threshold) indicates that the methodology was found to be a better forecast than all other 16 methodologies. Methodologies are sorted by average significance share across the five vintages. (Color figure online)

performed middle of the pack. However, its weakness was apparent during periods with large shocks. It predicted large declines in the third quarter of 2020, belatedly reacting to the strong decline of the second quarter of 2020 even though the quarter saw the largest growth rate recorded in the series. The approach's usefulness as a standalone methodology is limited in nowcasting for all but the most predictable series, as evidenced by its worst shown in Table 2.

Bayesian VAR

A detailed plot of the Bayesian VAR model's predictions is available in Fig. 5. The Bayesian VAR model was highly performant in the analysis, ranking as the best performer in terms of MAE and second-best, behind the LSTM, in terms of RMSE. Its ability to pick up both 2020 Q2's large decline and 2020 Q3's large growth was

rivalled only by the LSTM. The methodology produced, however, the most volatile predictions of all the methodologies. Most methodologies' revisions were clustered around 0.1 and 0.2 percentage points. The Bayesian VAR's revisions were two to four times higher, nearing 0.4 percentage points. Only MF-VAR and OLS rivalled that degree of volatility. High volatility in a nowcast can be an issue, as large revisions in predictions over time can make it difficult to communicate and rely on forecasts. The Bayesian VAR has the additional benefit of being able to produce uncertainty measures in its predictions, as the methodology does not produce a point estimate but rather a distribution of estimates.

Decision tree

A detailed plot of the decision tree model's predictions is available in Fig. 6. The decision tree did not perform particularly well in the analysis, especially at earlier data vintages. Generally, the tree learned most of its information from the last month of each quarter, so it was only able to start producing estimates significantly different from the mean at the one month after vintage. In tranquil periods, its predictions were relatively accurate at the two months after vintage, but this did not carry over during the COVID crisis. It was unable to produce predictions significantly higher or lower than those it had previously seen in training, thus being unable to produce estimates of large contraction in 2020 Q2 despite the strong negative signals in the data. All of these shortcomings led to the second-worst MAE and the third-worst RMSE of the analysis. However, the methodology could be useful in cases where the full complement of input data is available relatively quickly, e.g., if a monthly variable is being nowcast using extremely timely indicators.

DeepVAR

A detailed plot of the DeepVAR model's predictions is available in Fig. 7. DeepVAR generally performed poorly in the analysis. Its performance was akin to a slightly improved ARMA model, performing quite well during tranquil periods, but wholly unable to capture the volatile movements of the 2008 financial crisis or COVID crisis. Its predictions were relatively consistent but only able to capture small trends in the data, such as slightly elevated growth rates in the aftermath of the COVID crisis. This may be a result of its focus as a forecasting methodology, making it less suited to the ragged edges paradigm common in nowcasting. It could also be that its nowcasting performance could be improved with more adjustments than those performed for this analysis. In this implementation, however, its use is best limited to those cases where an ARMA model is applicable.

DFM

A detailed plot of the DFM model's predictions is available in Fig. 8. Surprisingly given its ubiquity, the DFM was a middling performer in the analysis, ranking middle of the pack in terms of MAE and fourth-best in terms of RMSE. It displayed a positive bias over the tranquil period from 2012 to 2019. It was able to capture the strong movements during the two crises earlier than most other methodologies, coupling this early insight with the fourth-least volatile revisions. DFMs and specifically the *nowcastDFM* implementation have the additional benefit of being able to give

interpretability to the changes in their predictions. Via the *gen_news* function, new data's contributions to changes in predictions can be obtained.

Elastic net

A detailed plot of the elastic net model's predictions is available in Fig. 9. Elastic net was the best performing of the three penalized OLS approaches. Its predictive performance was on par with MIDAS and Midasml, while being more volatile than those two approaches. It was able to capture the large movements during COVID relatively well and boasted all-around solid performance for those looking for an OLS-based model. It retained and improved upon the predictive power of OLS while substantially reigning in that methodology's volatility.

Gradient boosted trees

A detailed plot of the gradient boosted trees model's predictions is available in Fig. 10. The three ensemble tree-based methodologies all had similar performance. Gradient boost obtained the worst predictive performance of the three by a small margin, but was less volatile than XGBoost. Gradient boost performed quite well during the tranquil periods from 2002–2007 and 2010–2019, as evidenced in Fig. 10 and Tables 4 and 6, but was wholly unequipped to deal with both the financial crisis and the COVID crisis. Similar to the decision tree, it was unable to produce estimates far outside the bounds of what it had seen previously, leading to particularly poor performance during COVID.

Like the other tree-based methods, gradient boosted trees seemed to learn most of its information from the latest available data, making its two months after vintage predictions quite accurate, but its earlier predictions hewed very close to the mean. One approach tested to mitigate this behavior which particularly benefited gradient boosted trees was training a different model for each data vintage. That is, training five separate models, each only with information that would be available at that vintage. This forces the model to learn more information on data available earlier. Full details and specifications can be found in (Hopp 2022b).

Lasso

A detailed plot of the Lasso model's predictions is available in Fig. 11. Lasso performed better than ridge regression, but worse than elastic net in the analysis. It was able to capture large movements during the crisis periods, particularly at the 2 months ahead vintage, but significantly overestimated Q2 2020's decline. Its estimates were comparable in volatility to elastic net's, but were worse in predictive accuracy.

LSTM

A detailed plot of the LSTM model's predictions is available in Fig. 12. The LSTM performed well in the analysis, ranking the best in terms of RMSE and second-best in terms of MAE, behind the Bayesian VAR. It was, however, significantly less volatile, with average revisions less than half the Bayesian VARs. Performance was especially strong in the COVID crisis. It was rivalled by perhaps only MIDAS and Midasml in terms of combining accurate predictions with low volatility in revisions. The implementation has the added benefits of being able to produce prediction uncertainty via the *interval_predict* function and produce interpretability via the *feature_contribution* and *gen_news* functions. It also works with any combination of variable frequencies, e.g., nowcasting a yearly variable using monthly variables, or a daily variable using

hourly variables. This is not true of many of the other methodologies' implementations, notably the DFM and Bayesian VAR, both of which only work with combinations of quarterly and monthly variables.

MF-VAR

A detailed plot of the MF-VAR model's predictions is available in Fig. 13. The MF-VAR model boasted average performance in the analysis but coupled this with the second-most volatile predictions, behind only Bayesian VAR. It sometimes produced outlier predictions, such as largely overestimating Q2 2020's decline, and overestimating Q1 2021's growth.

MIDAS

A detailed plot of the MIDAS model's predictions is available in Fig. 14. MIDAS was a well-performing methodology, ranking behind only the LSTM and Bayesian VAR in terms of RMSE and in Fig. 3. It had trouble predicting extreme values during both the financial crisis and COVID crisis, but was particularly non-volatile in its revisions, ranking third-least volatile. This combination of solid accuracy and extremely low volatility makes it a compelling choice for nowcasting.

Midasml

A detailed plot of the Midasml model's predictions is available in Fig. 15. Midasml was a top three methodology in the analysis, ranking fourth- and third-best in terms of MAE and RMSE, respectively. It combined this with below-average volatility and did a good job picking up recessionary signals during the financial and COVID crises.

MLP

A detailed plot of the MLP model's predictions is available in Fig. 16. The MLP displays perhaps the biggest discrepancy between its MAE and RMSE ranking, coming in third worst in the former but middle of the pack in the latter. This, coupled with observing Fig. 16, implies the methodology is adequate at making predictions during tumultuous periods, but worse suited to making predictions during tranquil times. It did well in predicting Q2 2020's decline, but struggled, like many methodologies, in predicting the degree of recovery in Q3 2020. Its inferior performance to the LSTM, at the cost of no additional volatility, implies there is substantial information gain from explicitly incorporating a temporal component into the neural network architecture in the nowcasting context.

OLS

A detailed plot of the OLS model's predictions is available in Fig. 17. Given the high number of variables plus lags included in the modeling, it is not surprising to see OLS as a middling and volatile performer in the analysis. Though the method's predictions were quite accurate in Q2 and Q3 2020. It beats out both Lasso and ridge regression in terms of predictive accuracy, but at the cost of substantially more volatility. Elastic net is the best choice of OLS-derived model in the analysis, offering improved predictive accuracy coupled with lower volatility.

Random forest

A detailed plot of the random forest model's predictions is available in Fig. 18. Random forest suffered from the same issues as gradient boosted trees and the decision tree,

having difficulty predicting anything other than the mean until the two months ahead vintage and unable to predict extreme values during the crises. Similar to gradient boosted trees, before COVID it was good at prediction at the 2 month ahead vintage.

Ridge regression
A detailed plot of the ridge regression model's predictions is available in Fig. 19. While ridge regression was the least volatile OLS-derived model in the analysis, it was also the least accurate. Its lower overall volatility did not stop it from producing overly extreme predictions for Q2 2020, predicting a contraction more than double the actual value, even at the 2 months after vintage.

XGBoost
A detailed plot of the XGBoost model's predictions is available in Fig. 20. Unsurprisingly, XGBoost generated similar results to gradient boosted trees, though it was a bit more volatile. Performance and prediction characteristics between the two were very close. Choosing to use one over the other in this case may come down to implementation considerations, with XGBoost offering some performance improvements over gradient boosted trees.

6 Conclusion

Never before have nowcasting practitioners had so many options when it comes to selecting which methodology to use in their applications. This paper has attempted to ease that selection process by providing comparative results in nowcasting US GDP growth over a 20-year period, covering two of the most volatile periods in US economic history. The two best-performing methodologies were found to be the LSTM and Bayesian VAR, the latter coming with the caveat of having the highest revisions upon receiving new data in the analysis.

A primary aim of this paper was not only comparing methodologies on a benchmark dataset, but enabling others to use and explore those methodologies themselves. Results and characteristics found in this analysis will not hold for every dataset. Practitioners should use these findings as a starting point to try multiple of the methodologies on their own data. To that end, the exact code used to produce these results for each methodology is published on GitHub at (Hopp 2022b). This includes not only the function calls together with hyperparameters, but the code for getting the data and transforming it, the code necessary for generating and testing on artificial lags, and the code for generating predictions with new data. Once one's data are in a standardized format, with a date column, observations in rows, and series in separate columns, they can follow the self-contained boilerplate code in each methodology's Jupyter Notebook to adapt it to their own application or use as a springboard for their own research.

Beyond predictive performance, other factors such as computation time, time frequencies handled, and implementations available need to be taken into account when selecting a nowcasting methodology. More information on these specific characteristics in the implementations used for this analysis is available in Hopp (2022b). With

more tools, transparency, and context, existing practitioners may be able to try different nowcasting approaches to validate their models and new ones may be able to enter the field with novel applications.

Acknowledgements The author would like to thank Anu Peltola for her valuable comments and feedback.

Funding No funds, grants, or other support were received.

Declarations

Conflict of interest The author has no relevant financial or non-financial interests to disclose.

Ethical approval This article does not contain any studies with human participants or animals performed by any of the authors.

Appendix

A.1 Additional tabular results

See Tables 4, 5, 6 and 7.

Table 4 RMSE as a proportion of ARMA model, 2002–2007

Vintage	−2	−1	0	1	2
Bayesian VAR	1.19	1.19	1.10	1.25	1.18
Decision tree	1.00	0.99	1.09	1.16	1.37
DeepVAR	0.91	1.00	0.96	0.94	1.01
DFM	0.98	1.06	1.02	1.01	1.03
Elastic net	0.98	0.94	0.96	1.11	0.95
Gradient boost	0.92	0.88	0.86	0.82*	0.81*
Lasso	1.01	0.98	1.00	1.12	0.96
LSTM	1.01	0.86*	0.86*	0.89	0.81
MF-VAR	0.94	0.96	1.03	1.23	1.50
MIDAS	0.97	0.95	0.96	0.93	0.84
Midasml	1.02	0.95	1.00	0.96	0.96
MLP	1.37	1.30	1.29	1.28	1.26
OLS	0.94	0.94	1.07	1.35	1.60
Random Forest	0.89	0.89	0.89	0.86	0.83
Ridge	1.05	0.94	0.94	0.96	0.91
XGBoost	0.91*	0.94	0.92	1.03	1.12

This table shows RMSE on the test set as a proportion of the ARMA model's RMSE per data vintage for the period from 2002 to 2007. A value higher than 1 denotes worse performance than the ARMA model, and a value lower than 1 denotes better performance
* Marks the best-performing model in the vintage (column)

Table 5 RMSE as a proportion of ARMA model, 2008–2009

Vintage	−2	−1	0	1	2
Bayesian VAR	0.43*	0.78	0.58	0.53	0.47
Decision tree	1.08	1.30	1.30	1.00	0.48
DeepVAR	0.90	1.06	1.07	0.99	0.99
DFM	0.78	0.93	0.89	0.85	0.84
Elastic net	0.89	0.93	0.88	0.92	0.38
Gradient boost	0.99	1.17	1.01	0.79	0.68
Lasso	1.00	1.07	1.02	1.10	0.44
LSTM	0.78	0.66	0.55*	0.48*	0.47
MF-VAR	1.06	1.19	0.75	1.15	0.65
MIDAS	0.87	0.93	0.83	0.71	0.44
Midasml	0.54	0.62*	0.75	0.72	0.72
MLP	1.02	0.95	0.82	0.66	0.61
OLS	1.01	1.34	0.81	1.17	0.64
Random forest	0.97	1.10	0.94	0.71	0.61
Ridge	0.87	0.79	0.64	0.56	0.52
XGBoost	0.98	1.11	1.10	0.84	0.33*

This table shows RMSE on the test set as a proportion of the ARMA model's RMSE per data vintage for the period from 2008 to 2009. A value higher than 1 denotes worse performance than the ARMA model, and a value lower than 1 denotes better performance
* Marks the best-performing model in the vintage (column)

Table 6 RMSE as a proportion of ARMA model, 2010–2019

Vintage	−2	−1	0	1	2
Bayesian VAR	1.09	0.91*	0.96	0.85	0.72
Decision tree	1.05	1.06	1.10	0.98	0.97
DeepVAR	0.92	0.93	0.90	0.84	0.85
DFM	1.17	1.12	1.12	1.13	1.15
Elastic net	0.98	0.99	1.01	1.08	0.85
Gradient boost	0.91*	0.91	0.90	0.74*	0.64
Lasso	1.10	1.14	1.15	1.14	0.91
LSTM	1.11	1.20	1.25	1.25	1.21
MF-VAR	1.04	1.08	1.03	1.04	0.83
MIDAS	1.03	1.00	0.99	0.98	0.89
Midasml	1.00	1.00	0.93	0.91	0.91
MLP	1.42	1.38	1.48	1.58	1.49
OLS	0.98	1.11	1.03	1.07	0.86
Random forest	0.93	0.94	0.93	0.86	0.79
Ridge	1.17	1.26	1.29	1.33	1.28
XGBoost	0.91	0.85	0.85*	0.84	0.64*

This table shows RMSE on the test set as a proportion of the ARMA model's RMSE per data vintage for the period from 2010 to 2019. A value higher than 1 denotes worse performance than the ARMA model, and a value lower than 1 denotes better performance
* Marks the best-performing model in the vintage (column)

Table 7 RMSE as a proportion of ARMA model, 2020–2022

Vintage	−2	−1	0	1	2
Bayesian VAR	0.79*	0.52	0.19*	0.24	0.22*
Decision tree	0.96	0.85	0.85	0.83	0.70
DeepVAR	0.94	0.82	0.83	0.82	0.81
DFM	0.72	0.42	0.42	0.45	0.46
Elastic net	0.92	0.50	0.48	0.41	0.30
Gradient boost	0.94	0.79	0.80	0.78	0.75
Lasso	0.93	0.51	0.48	0.42	0.40
LSTM	0.79	0.31*	0.28	0.23*	0.23
MF-VAR	0.92	0.59	0.54	0.42	0.43
MIDAS	0.85	0.50	0.49	0.52	0.37
Midasml	0.85	0.41	0.37	0.38	0.37
MLP	0.83	0.43	0.40	0.37	0.36
OLS	0.92	0.52	0.46	0.37	0.27
Random forest	0.92	0.77	0.77	0.76	0.73
Ridge	0.89	0.64	0.61	0.55	0.58
XGBoost	0.93	0.80	0.80	0.75	0.63

This table shows RMSE on the test set as a proportion of the ARMA model's RMSE per data vintage for the period from 2020 to 2022. A value higher than 1 denotes worse performance than the ARMA model, and a value lower than 1 denotes better performance

* Marks the best-performing model in the vintage (column)

B.2 Additional graphical results

See Figs. 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20, 21, 22, 23, 24, and 25.

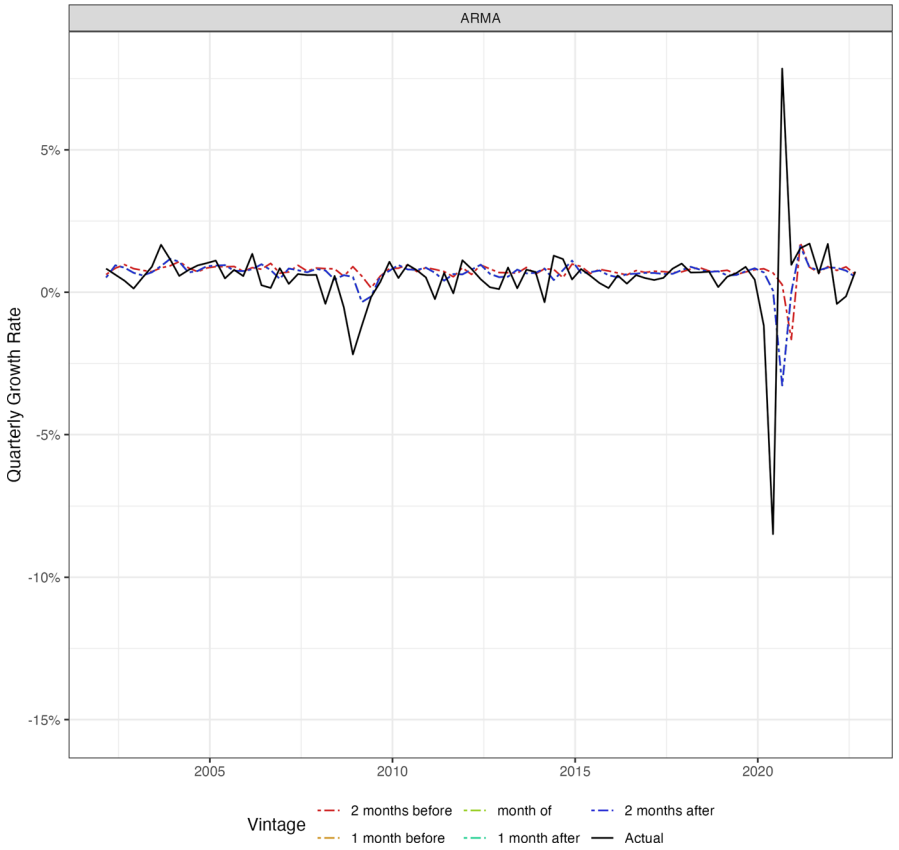

Fig. 4 Nowcasts (part 1 of 17). *Note*: This plot shows the predictions of the methodology at each quarter in the test period. The black line denotes what the actual quarterly growth rate was, while the various colored dotted lines denote the predictions at various data vintages. Dotted lines clustered together that track the black line closely indicate a well-performing model. Dotted lines spread out quite different from each other and far from the black line indicate a poorly performing model. (Color figure online)

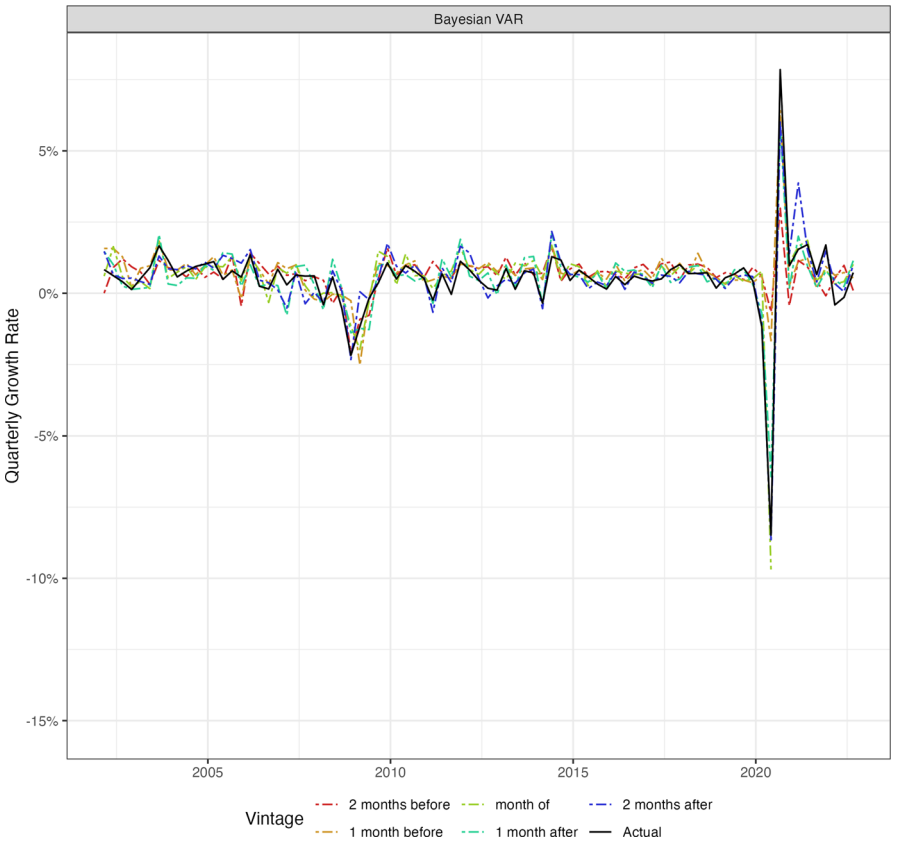

Fig. 5 Nowcasts (part 2 of 17). *Note:* This plot shows the predictions of the methodology at each quarter in the test period. The black line denotes what the actual quarterly growth rate was, while the various colored dotted lines denote the predictions at various data vintages. Dotted lines clustered together that track the black line closely indicate a well-performing model. Dotted lines spread out quite different from each other and far from the black line indicate a poorly performing model. (Color figure online)

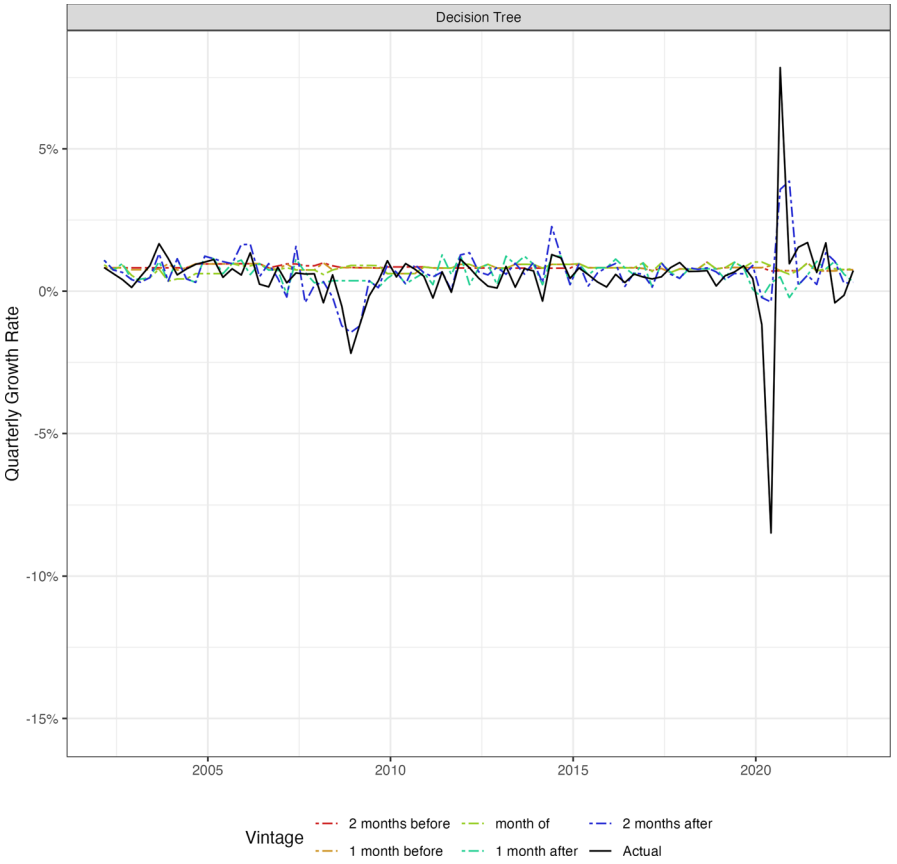

Fig. 6 Nowcasts (part 3 of 17). *Note*: This plot shows the predictions of the methodology at each quarter in the test period. The black line denotes what the actual quarterly growth rate was, while the various colored dotted lines denote the predictions at various data vintages. Dotted lines clustered together that track the black line closely indicate a well-performing model. Dotted lines spread out quite different from each other and far from the black line indicate a poorly performing model. (Color figure online)

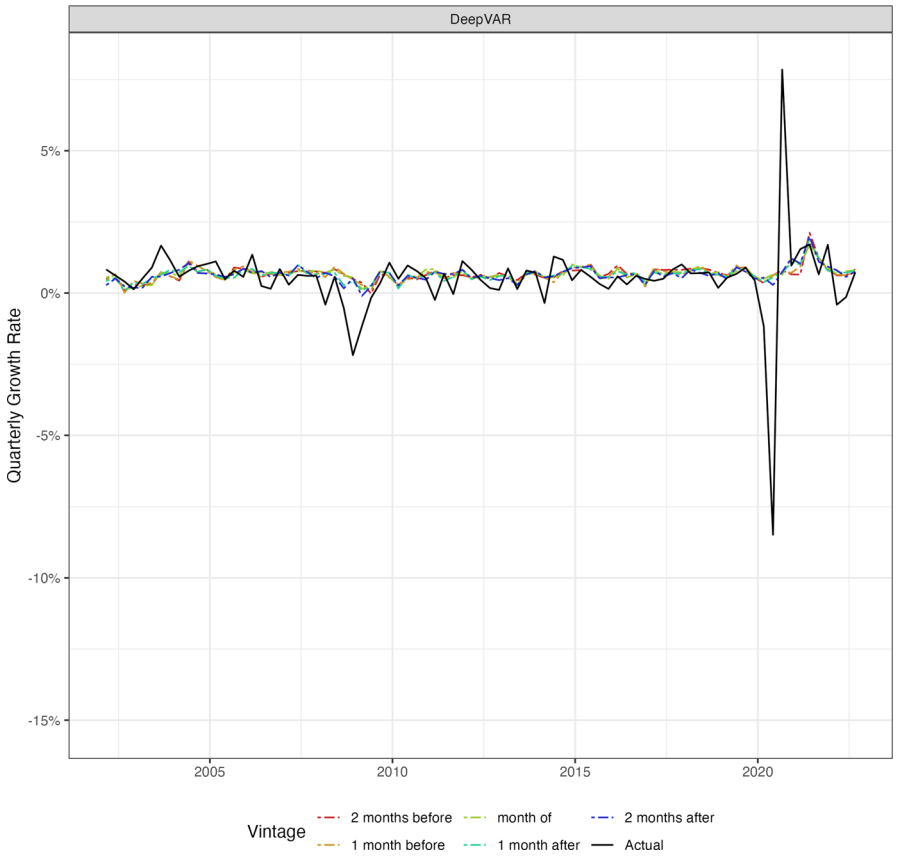

Fig. 7 Nowcasts (part 4 of 17). *Note*: This plot shows the predictions of the methodology at each quarter in the test period. The black line denotes what the actual quarterly growth rate was, while the various colored dotted lines denote the predictions at various data vintages. Dotted lines clustered together that track the black line closely indicate a well-performing model. Dotted lines spread out quite different from each other and far from the black line indicate a poorly performing model. (Color figure online)

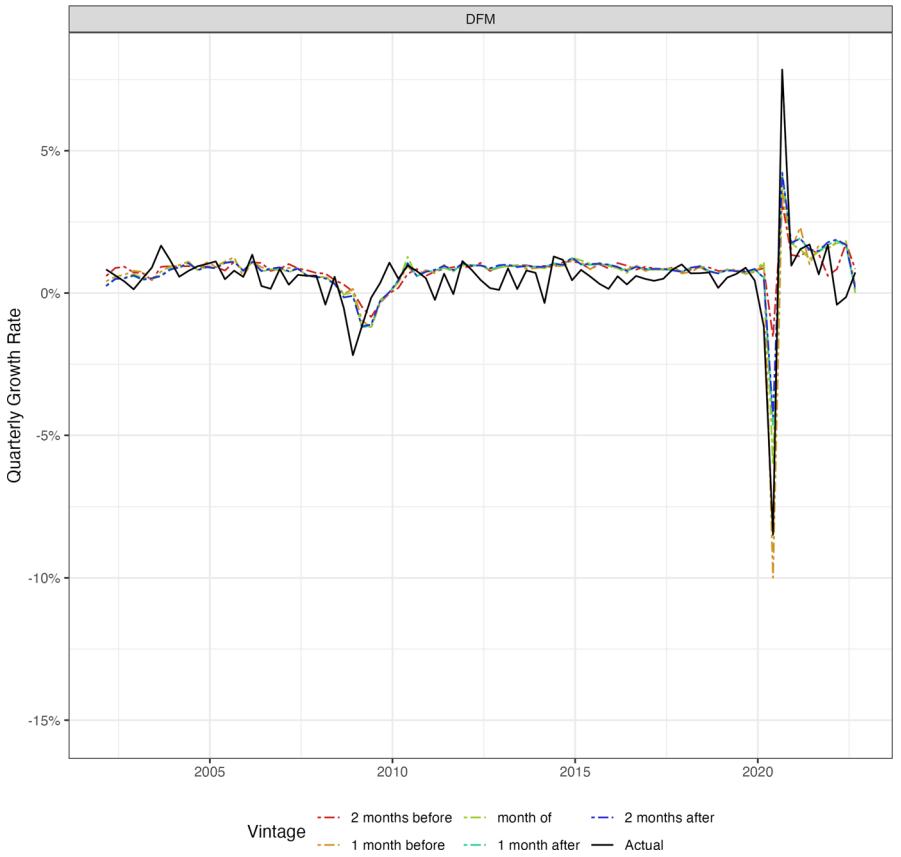

Fig. 8 Nowcasts (part 5 of 17). *Note*: This plot shows the predictions of the methodology at each quarter in the test period. The black line denotes what the actual quarterly growth rate was, while the various colored dotted lines denote the predictions at various data vintages. Dotted lines clustered together that track the black line closely indicate a well-performing model. Dotted lines spread out quite different from each other and far from the black line indicate a poorly performing model. (Color figure online)

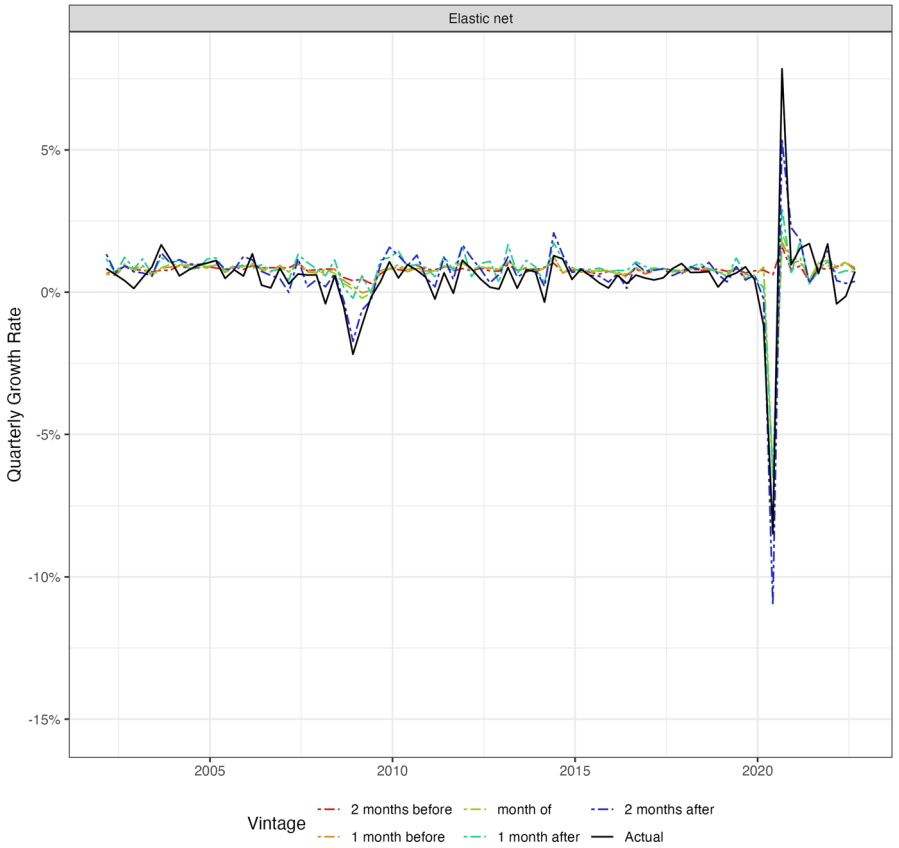

Fig. 9 Nowcasts (part 6 of 17). *Note*: This plot shows the predictions of the methodology at each quarter in the test period. The black line denotes what the actual quarterly growth rate was, while the various colored dotted lines denote the predictions at various data vintages. Dotted lines clustered together that track the black line closely indicate a well-performing model. Dotted lines spread out quite different from each other and far from the black line indicate a poorly performing model. (Color figure online)

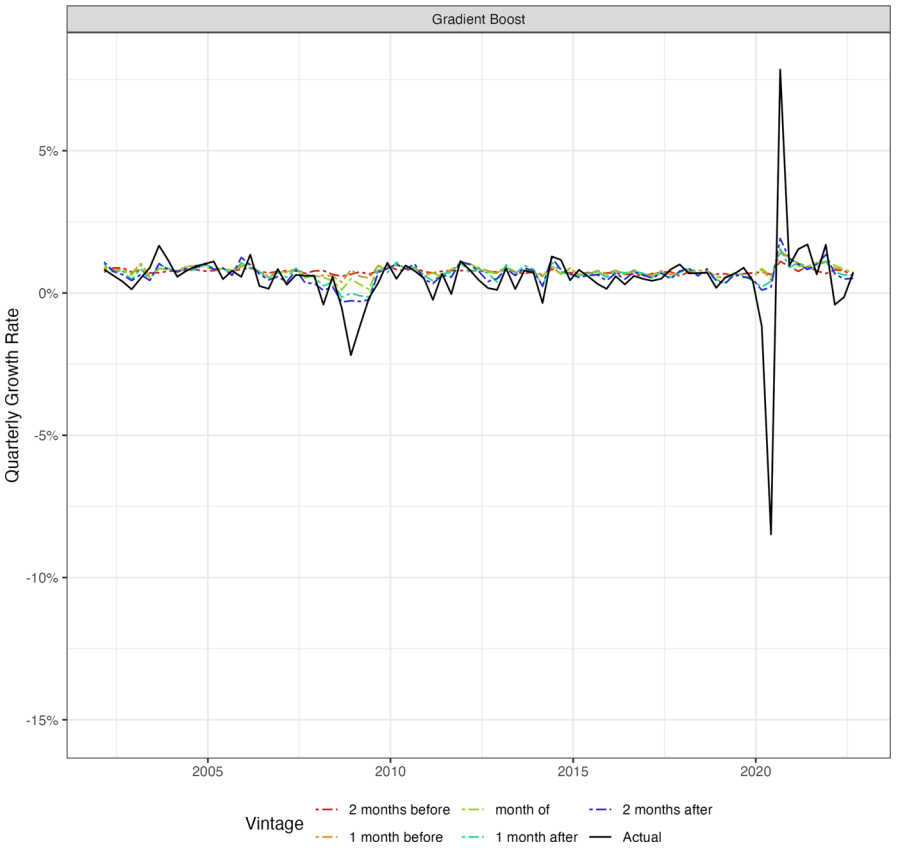

Fig. 10 Nowcasts (part 7 of 17). *Note*: This plot shows the predictions of the methodology at each quarter in the test period. The black line denotes what the actual quarterly growth rate was, while the various colored dotted lines denote the predictions at various data vintages. Dotted lines clustered together that track the black line closely indicate a well-performing model. Dotted lines spread out quite different from each other and far from the black line indicate a poorly performing model. (Color figure online)

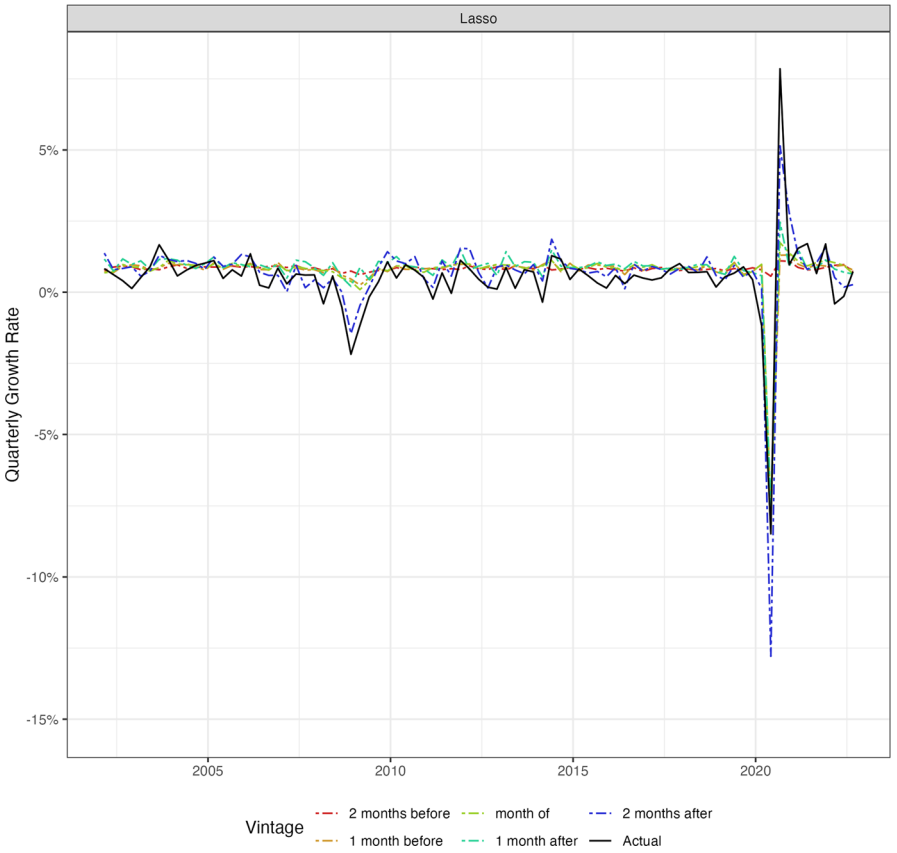

Fig. 11 Nowcasts (part 8 of 17). *Note*: This plot shows the predictions of the methodology at each quarter in the test period. The black line denotes what the actual quarterly growth rate was, while the various colored dotted lines denote the predictions at various data vintages. Dotted lines clustered together that track the black line closely indicate a well-performing model. Dotted lines spread out quite different from each other and far from the black line indicate a poorly performing model. (Color figure online)

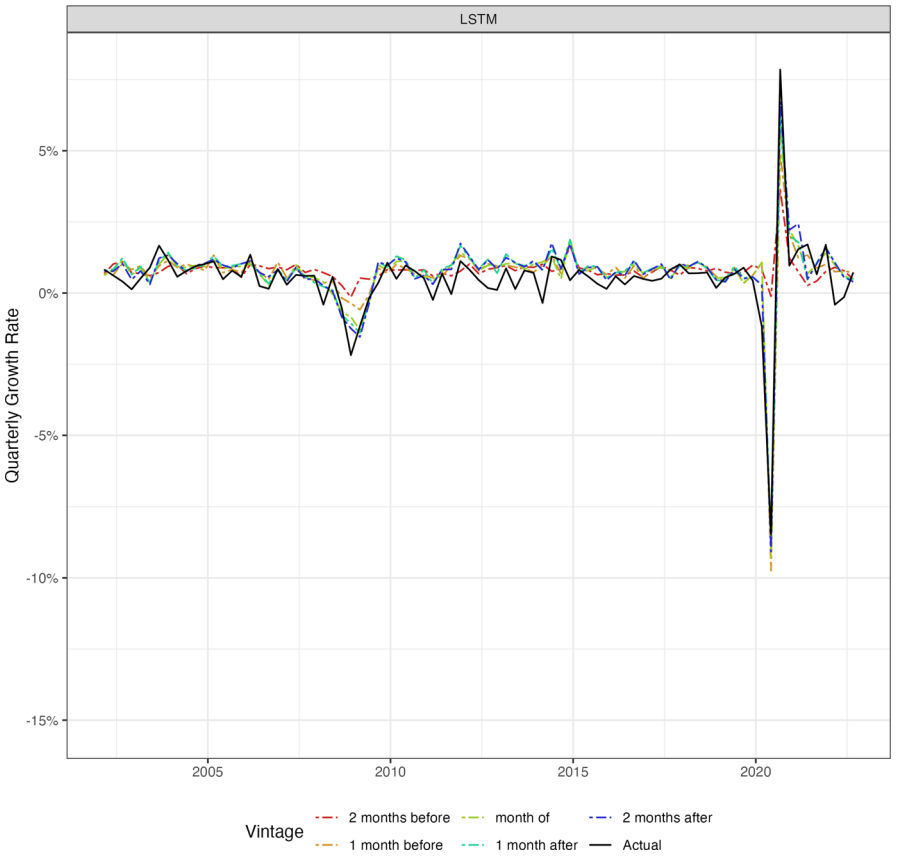

Fig. 12 Nowcasts (part 9 of 17). *Note*: This plot shows the predictions of the methodology at each quarter in the test period. The black line denotes what the actual quarterly growth rate was, while the various colored dotted lines denote the predictions at various data vintages. Dotted lines clustered together that track the black line closely indicate a well-performing model. Dotted lines spread out quite different from each other and far from the black line indicate a poorly performing model. (Color figure online)

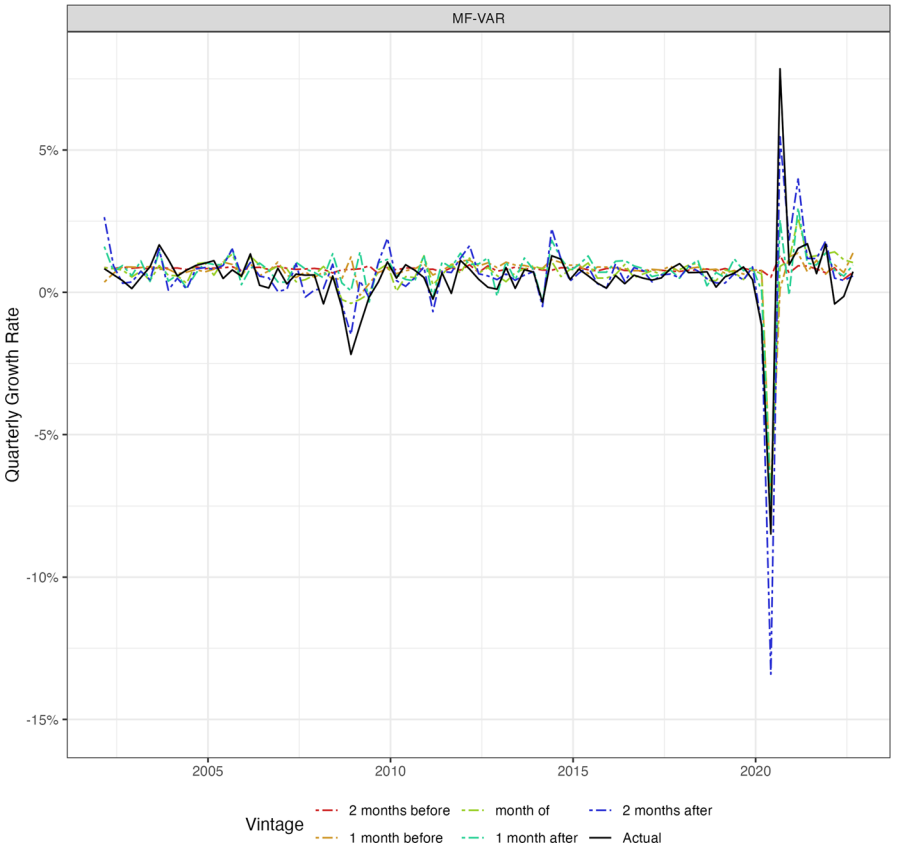

Fig. 13 Nowcasts (part 10 of 17). *Note*: This plot shows the predictions of the methodology at each quarter in the test period. The black line denotes what the actual quarterly growth rate was, while the various colored dotted lines denote the predictions at various data vintages. Dotted lines clustered together that track the black line closely indicate a well-performing model. Dotted lines spread out quite different from each other and far from the black line indicate a poorly performing model. (Color figure online)

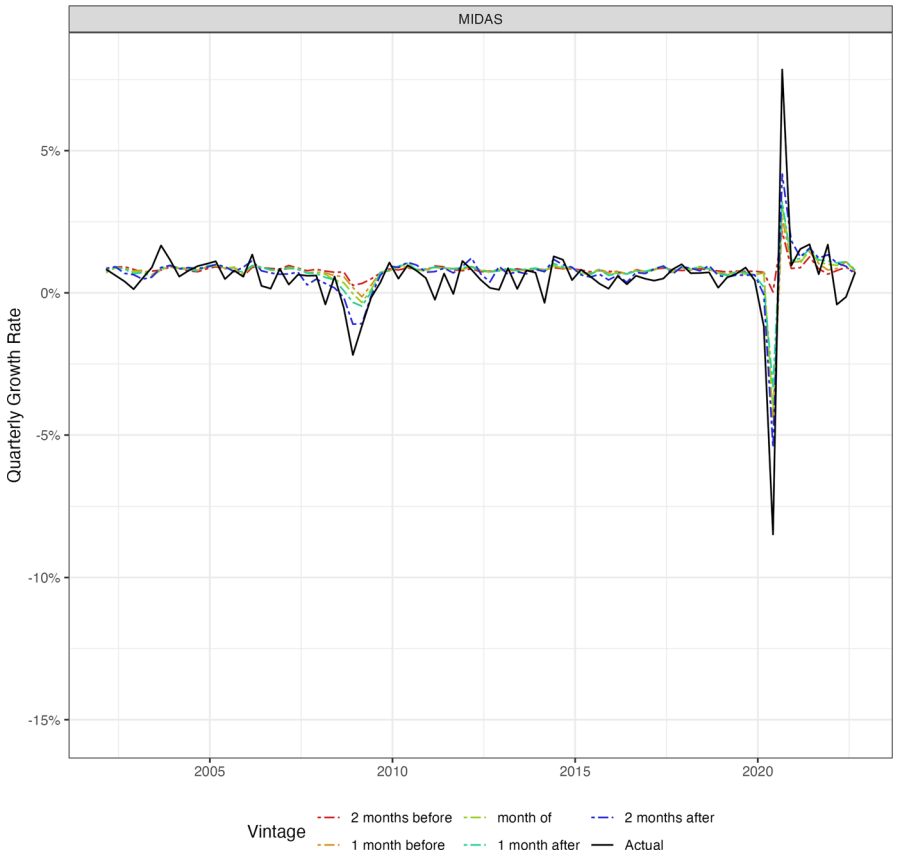

Fig. 14 Nowcasts (part 11 of 17). *Note*: This plot shows the predictions of the methodology at each quarter in the test period. The black line denotes what the actual quarterly growth rate was, while the various colored dotted lines denote the predictions at various data vintages. Dotted lines clustered together that track the black line closely indicate a well-performing model. Dotted lines spread out quite different from each other and far from the black line indicate a poorly performing model. (Color figure online)

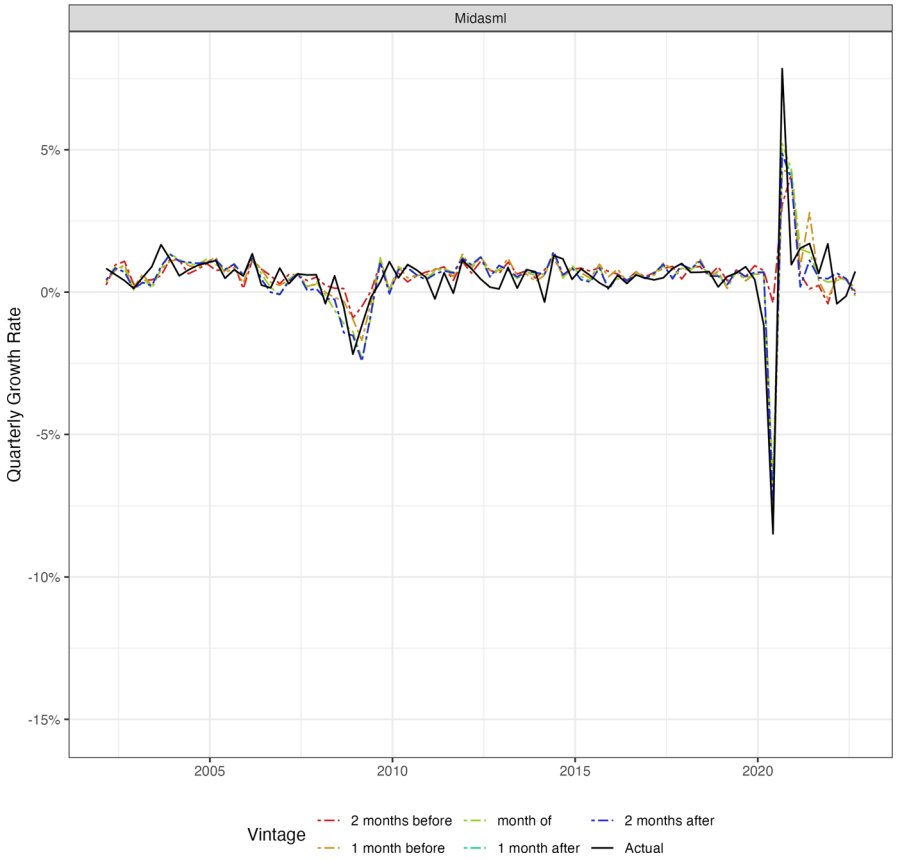

Fig. 15 Nowcasts (part 12 of 17). *Note*: This plot shows the predictions of the methodology at each quarter in the test period. The black line denotes what the actual quarterly growth rate was, while the various colored dotted lines denote the predictions at various data vintages. Dotted lines clustered together that track the black line closely indicate a well-performing model. Dotted lines spread out quite different from each other and far from the black line indicate a poorly performing model. (Color figure online)

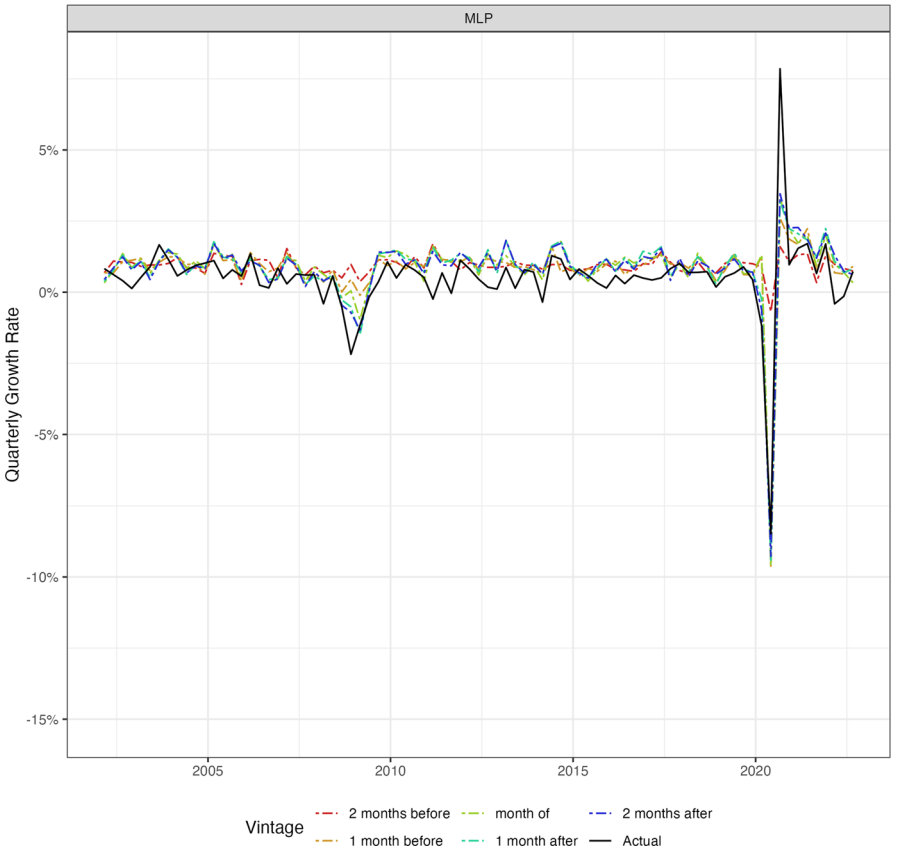

Fig. 16 Nowcasts (part 13 of 17). *Note*: This plot shows the predictions of the methodology at each quarter in the test period. The black line denotes what the actual quarterly growth rate was, while the various colored dotted lines denote the predictions at various data vintages. Dotted lines clustered together that track the black line closely indicate a well-performing model. Dotted lines spread out quite different from each other and far from the black line indicate a poorly performing model. (Color figure online)

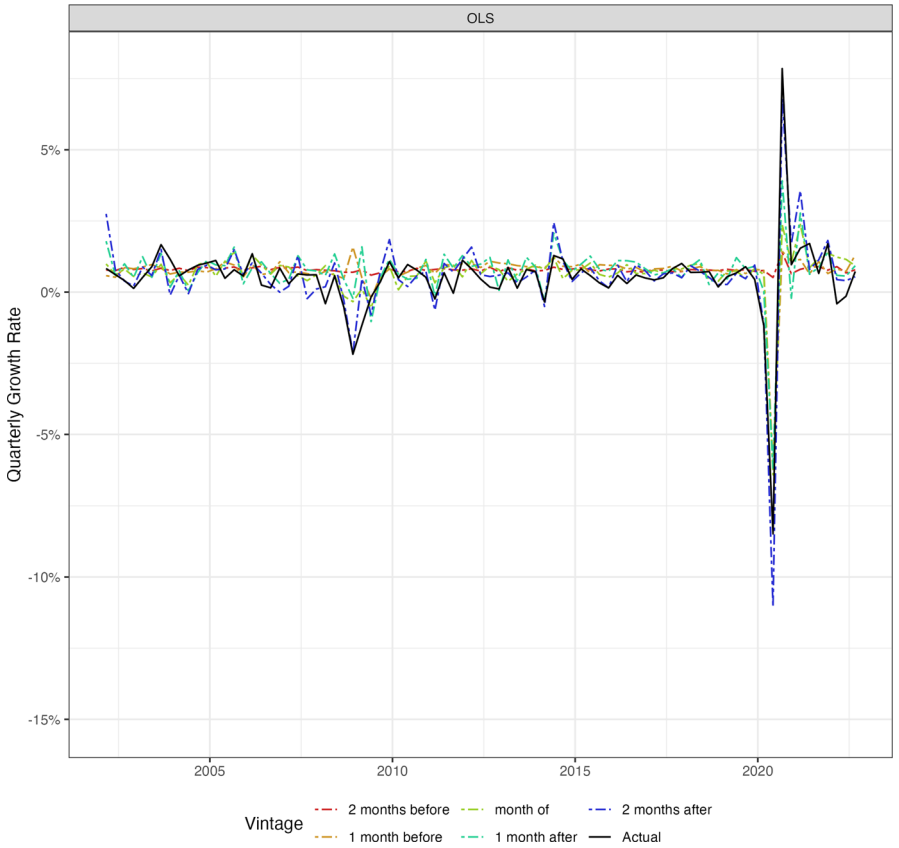

Fig. 17 Nowcasts (part 14 of 17). *Note*: This plot shows the predictions of the methodology at each quarter in the test period. The black line denotes what the actual quarterly growth rate was, while the various colored dotted lines denote the predictions at various data vintages. Dotted lines clustered together that track the black line closely indicate a well-performing model. Dotted lines spread out quite different from each other and far from the black line indicate a poorly performing model. (Color figure online)

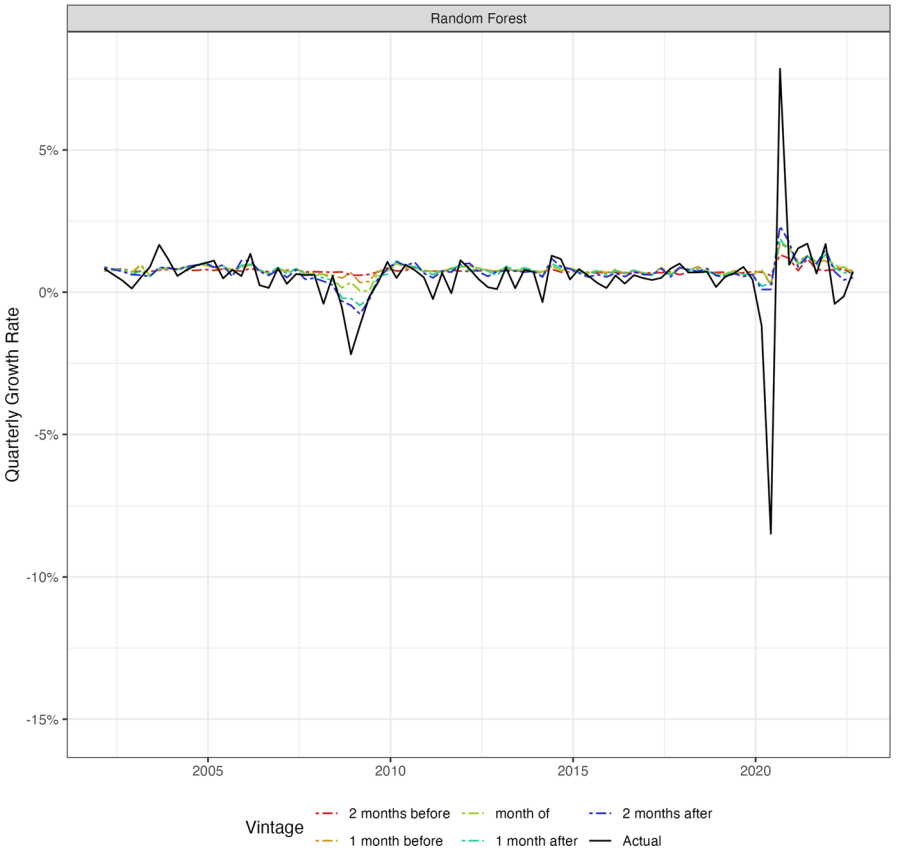

Fig. 18 Nowcasts (part 15 of 17). *Note*: This plot shows the predictions of the methodology at each quarter in the test period. The black line denotes what the actual quarterly growth rate was, while the various colored dotted lines denote the predictions at various data vintages. Dotted lines clustered together that track the black line closely indicate a well-performing model. Dotted lines spread out quite different from each other and far from the black line indicate a poorly performing model. (Color figure online)

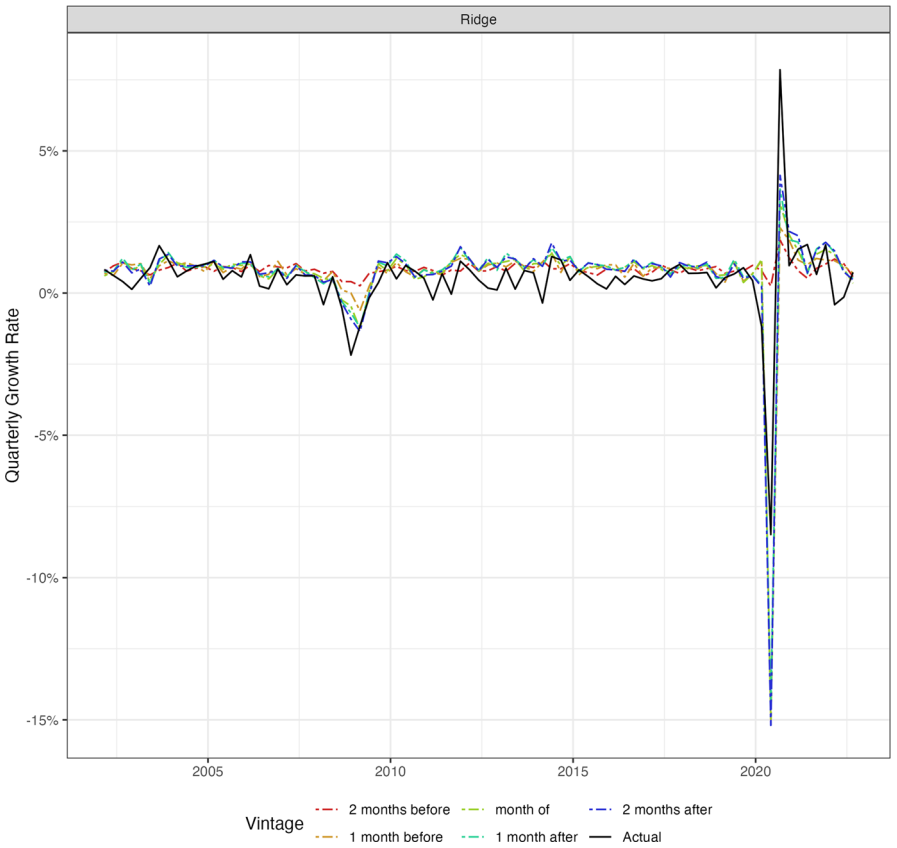

Fig. 19 Nowcasts (part 16 of 17). *Note:* This plot shows the predictions of the methodology at each quarter in the test period. The black line denotes what the actual quarterly growth rate was, while the various colored dotted lines denote the predictions at various data vintages. Dotted lines clustered together that track the black line closely indicate a well-performing model. Dotted lines spread out quite different from each other and far from the black line indicate a poorly performing model. (Color figure online)

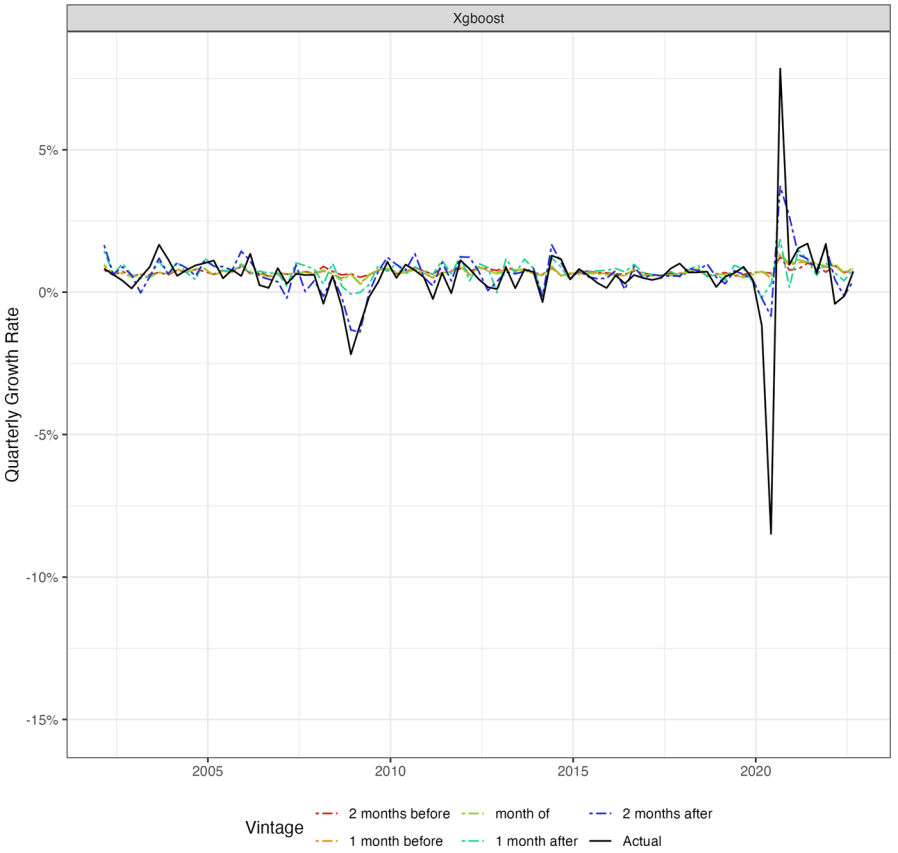

Fig. 20 Nowcasts (part 17 of 17). *Note:* This plot shows the predictions of the methodology at each quarter in the test period. The black line denotes what the actual quarterly growth rate was, while the various colored dotted lines denote the predictions at various data vintages. Dotted lines clustered together that track the black line closely indicate a well-performing model. Dotted lines spread out quite different from each other and far from the black line indicate a poorly performing model. (Color figure online)

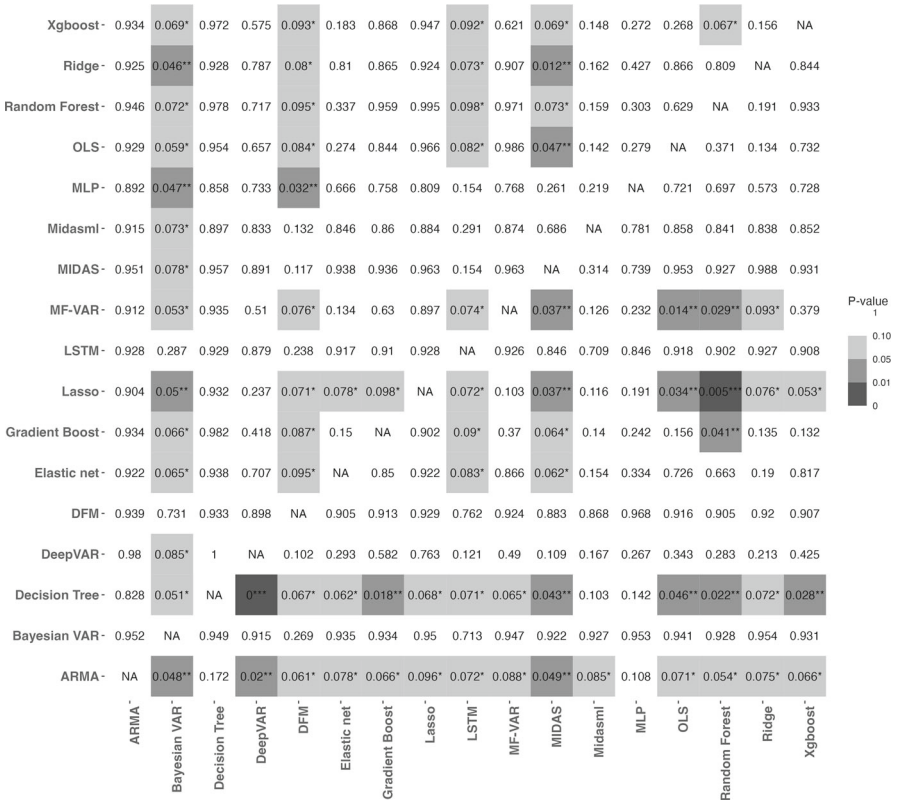

Fig. 21 2 months before vintage, pairwise Diebold–Mariano test statistics. Alternative hypothesis that the column methodology has superior predictions to the row methodology. *Note*: This plot shows the p values of the pairwise Diebold–Mariano test. The alternative hypothesis of the tests is that the methodology in the column has superior predictions to the methodology in the row. Significant values are marked both by color and with an asterisk(s). A methodology whose column is completely shaded (good) means that it achieved significance in each of the pairwise tests, while if the column is completely white (bad),it did not achieve significance in any of the pairwise tests. The inverse is true for rows; a completely shaded row means every other methodology achieved significance compared to it (bad), while a completely white row means no other methodology achieved significance compared to it (good). $^{***}p < 0.01$, $^{**}p < 0.05$, $^{*}p < 0.1$. (Color figure online)

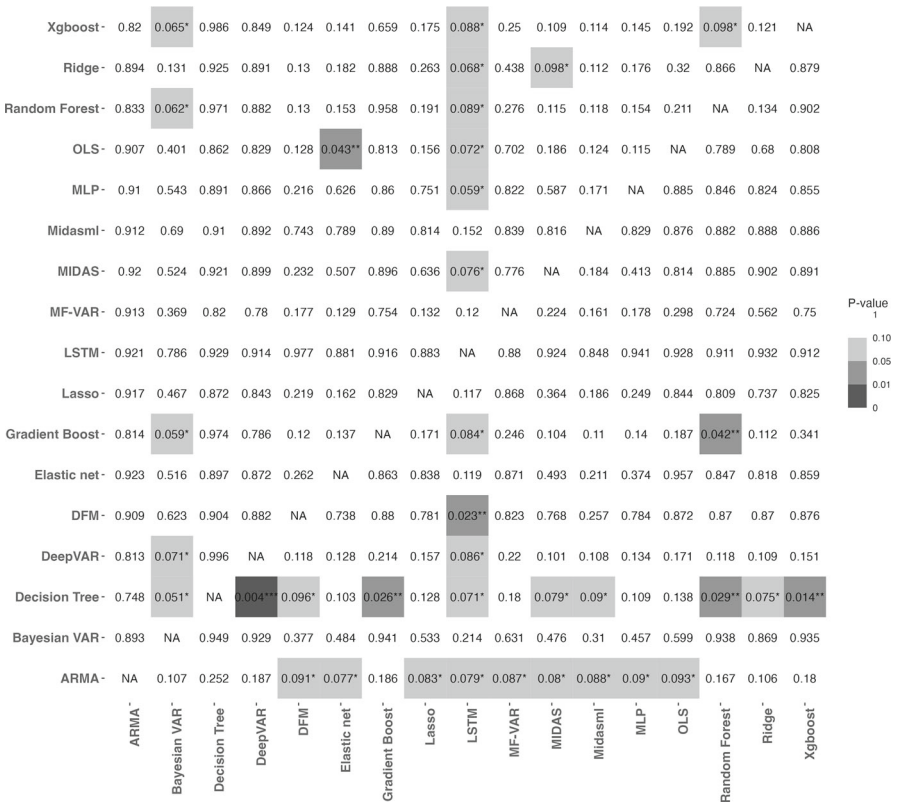

Fig. 22 1 month before vintage, pairwise Diebold–Mariano test statistics. Alternative hypothesis that the column methodology has superior predictions to the row methodology. *Note*: This plot shows the p values of the pairwise Diebold–Mariano test. The alternative hypothesis of the tests is that the methodology in the column has superior predictions to the methodology in the row. Significant values are marked both by color and with an asterisk(s). A methodology whose column is completely shaded (good) means that it achieved significance in each of the pairwise tests, while if the column is completely white (bad),it did not achieve significance in any of the pairwise tests. The inverse is true for rows; a completely shaded row means every other methodology achieved significance compared to it (bad), while a completely white row means no other methodology achieved significance compared to it (good). $^{***}p < 0.01$, $^{**}p < 0.05$, $^{*}p < 0.1$. (Color figure online)

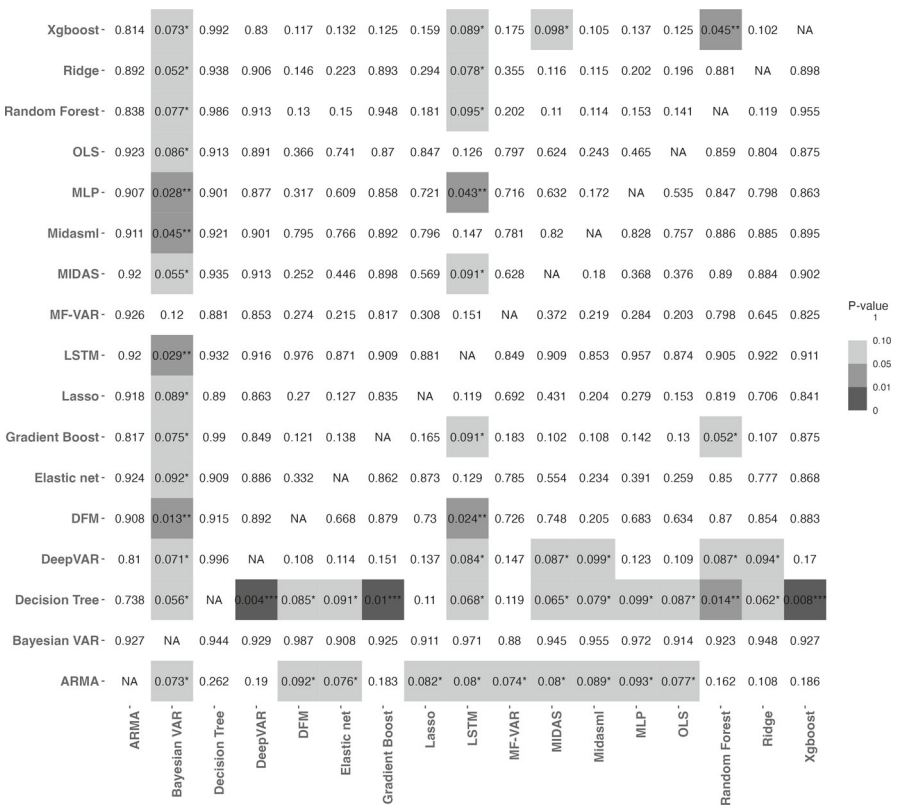

Fig. 23 Month of vintage, pairwise Diebold–Mariano test statistics. Alternative hypothesis that the column methodology has superior predictions to the row methodology. *Note*: This plot shows the p values of the pairwise Diebold–Mariano test. The alternative hypothesis of the tests is that the methodology in the column has superior predictions to the methodology in the row. Significant values are marked both by color and with an asterisk(s). A methodology whose column is completely shaded (good) means that it achieved significance in each of the pairwise tests, while if the column is completely white (bad),it did not achieve significance in any of the pairwise tests. The inverse is true for rows; a completely shaded row means every other methodology achieved significance compared to it (bad), while a completely white row means no other methodology achieved significance compared to it (good). $***p < 0.01$, $**p < 0.05$, $*p < 0.1$. (Color figure online)

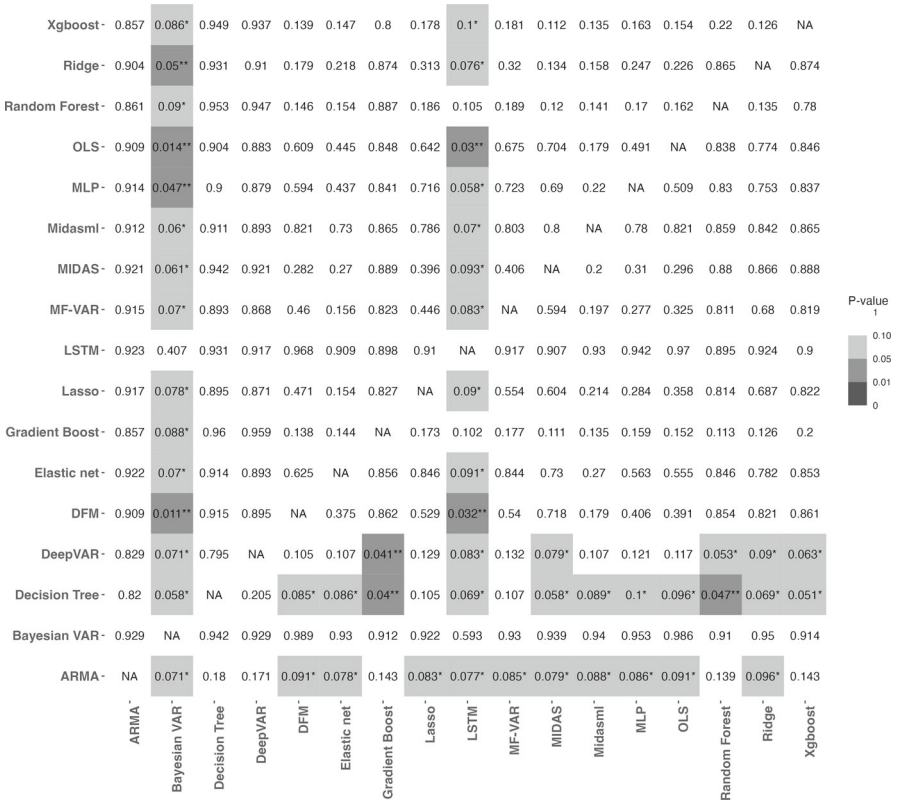

Fig. 24 1 month after vintage, pairwise Diebold–Mariano test statistics. Alternative hypothesis that the column methodology has superior predictions to the row methodology. *Note*: This plot shows the p values of the pairwise Diebold–Mariano test. The alternative hypothesis of the tests is that the methodology in the column has superior predictions to the methodology in the row. Significant values are marked both by color and with an asterisk(s). A methodology whose column is completely shaded (good) means that it achieved significance in each of the pairwise tests, while if the column is completely white (bad),it did not achieve significance in any of the pairwise tests. The inverse is true for rows; a completely shaded row means every other methodology achieved significance compared to it (bad), while a completely white row means no other methodology achieved significance compared to it (good). $^{***}p < 0.01$, $^{**}p < 0.05$, $^{*}p < 0.1$. (Color figure online)

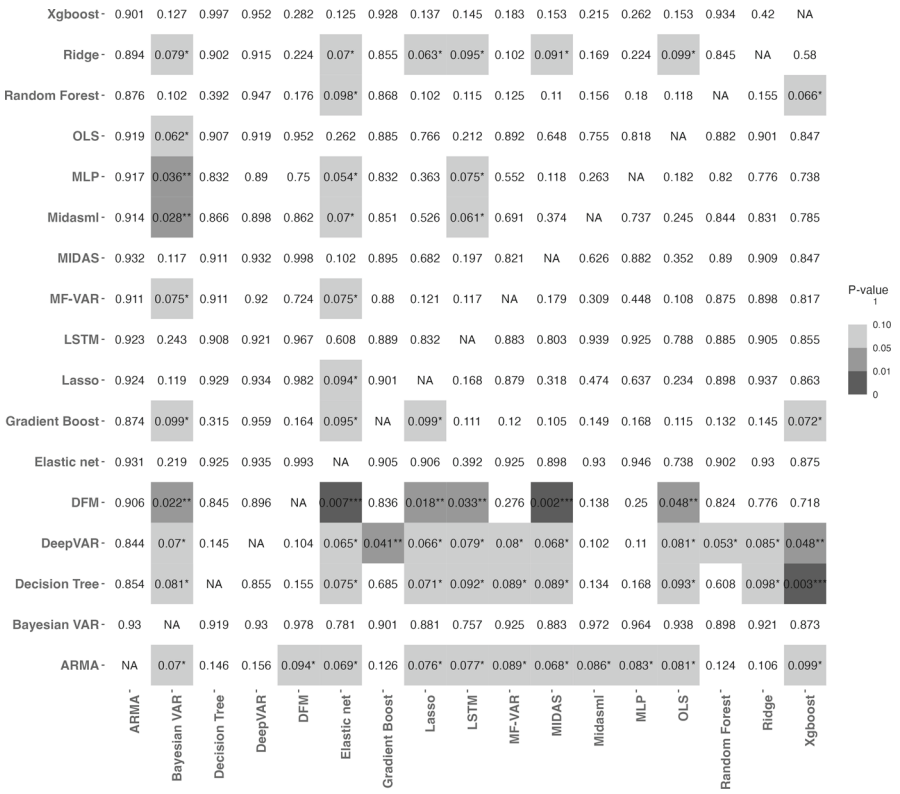

Fig. 25 2 months after vintage, pairwise Diebold–Mariano test statistics. Alternative hypothesis that the column methodology has superior predictions to the row methodology. *Note*: This plot shows the p values of the pairwise Diebold–Mariano test. The alternative hypothesis of the tests is that the methodology in the column has superior predictions to the methodology in the row. Significant values are marked both by color and with an asterisk(s). A methodology whose column is completely shaded (good) means that it achieved significance in each of the pairwise tests, while if the column is completely white (bad), it did not achieve significance in any of the pairwise tests. The inverse is true for rows; a completely shaded row means every other methodology achieved significance compared to it (bad), while a completely white row means no other methodology achieved significance compared to it (good). $^{***}p < 0.01$, $^{**}p < 0.05$, $^{*}p < 0.1$. (Color figure online)

References

Ankargren S, Yang Y, Kastner G (2021). Package 'mfbvar'. https://cran.r-project.org/web/packages/mfbvar/mfbvar.pdf

Antolin-Diaz J, Drechsel T, Petrella I (2020) Advances in nowcasting economic activity: secular trends, large shocks and new data. SSRN. https://papers.ssrn.com/sol3/papers.cfm?abstract_id=3669854

Babii A, Ghysels E, Striaukas J (2022) Machine learning time series regressions with an application to nowcasting. J Bus Econ Stat 40:1094–1106. https://doi.org/10.1080/07350015.2021.1899933

Bańbura M, Rünstler G (2011) A look into the factor model black box: Publication lags and the role of hard and soft data in forecasting GDP. Int J Forecast 27:333–346. https://doi.org/10.1016/j.ijforecast.2010.01.011

Bańbura M, Giannone D, Reichlin L (2010) Large Bayesian vector auto regressions. J Appl Economet 25:71–92. https://doi.org/10.1002/jae.1137

Barbaglia L, Frattarolo L, Onorante L, Pericoli FM, Ratto M, Pezzoli LT (2022) Testing big data in a big crisis: nowcasting under covid-19. Int J Forecast. https://doi.org/10.1016/j.ijforecast.2022.10.005
Boehmke B (2018) Gradient boosting machines. http://uc-r.github.io/gbm_regression
Bok B, Caratelli D, Giannone D, Sbordone AM, Tambalotti A (2018) Macroeconomic nowcasting and forecasting with big data. Annu Rev Econ 10:615–643. https://doi.org/10.1146/annurev-economics-080217-053214
Boysel S, Vaughan D (2021) Package 'fredr'. https://cran.r-project.org/web/packages/fredr/fredr.pdf
Brownlee J (2018) Deep learning for time series forecasting: predict the future with MLPs, CNNs and LSTMs in python. Machine Learning Mastery. https://books.google.ch/books?id=o5qnDwAAQBAJ
Buono D, Mazzi G, Marcellino M, Kapetanios (2017) Big data types for macroeconomic nowcasting. Eurona. https://ec.europa.eu/eurostat/cros/system/files/euronaissue1-2017-art4.pdf
Cantú F (2018) Estimation of a coincident indicator for international trade and global economic activity. Technical Report 27 UNCTAD. https://unctad.org/system/files/official-document/ser-rp-2018d9_en.pdf
Carriero A, Galvão AB, Kapetanios G (2019) A comprehensive evaluation of macroeconomic forecasting methods. Int J Forecasting 35:1226–1239. https://doi.org/10.1016/j.ijforecast.2019.02.007
Chen T, Guestrin C (2016) XGBoost. In: Proceedings of the 22nd ACM SIGKDD international conference on knowledge discovery and data mining. ACM. https://doi.org/10.1145/2939672.2939785
Chernis T, Sekkel R (2017) A dynamic factor model for nowcasting Canadian GDP growth. Empir Econ 53:217–234. https://doi.org/10.1007/s00181-017-1254-1
Chung J, Gulcehre C, Cho K, Bengio Y (2014) Empirical evaluation of gated recurrent neural networks on sequence modeling. CoRR, arXiv:1412.3555
Cimadomo J, Giannone D, Lenza M, Sokol A, Monti F (2020) Nowcasting with large Bayesian vector autoregressions. Working Paper Series 2453 European Central Bank. https://ideas.repec.org/p/ecb/ecbwps/20202453.html
Clements MP, Galvão AB (2008) Macroeconomic forecasting with mixed-frequency data: forecasting output growth in the United States. J Bus Econ Stat 26:546–554
De Mol C, Giannone D, Reichlin L (2008) Forecasting using a large number of predictors: Is Bayesian shrinkage a valid alternative to principal components? J Econom 146:318–328
Dematos G, Boyd MS, Kermanshahi B, Kohzadi N, Kaastra I (1996) Feedforward versus recurrent neural networks for forecasting monthly Japanese yen exchange rates. Financ Eng Japan Markets 3:59–75. https://doi.org/10.1007/BF00868008
Dynan K, Sheiner L, Fiscal BIH Co, Policy M (2018) GDP as a measure of economic well-being. Hutchins Center working paper. https://books.google.co.uk/books?id=rOe3vQEACAAJ
Eickmeier S, Ng T (2011) Forecasting national activity using lots of international predictors: an application to New Zealand. Int J Forecasting 27:496–511. https://doi.org/10.1016/j.ijforecast.2009
Falat L, Pancikova L (2015) Quantitative modelling in economics with advanced artificial neural networks. Procedia Econ Finance 34:194–201. https://doi.org/10.1016/S2212-5671(15)01619-6
Federal Reserve Bank of San Francisco (2005) Why is there such a time lapse getting the latest report on Gross Domestic Product (GDP)? In May 2005, we are just getting the preliminary GDP report for the quarter that ended in March 2005
Ghysels E (2016) Macroeconomics and the reality of mixed frequency data. Econom Anal Mixed Freq Data Sampling 193:294–314. https://doi.org/10.1016/j.jeconom.2016.04.008
Ghysels E, Santa-Clara P, Valkanov R (2004) The MIDAS touch: mixed data sampling regression models. In: CIRANO working papers CIRANO. https://EconPapers.repec.org/RePEc:cir:cirwor:2004s-20
Giannone D, Reichlin L, Simonelli S (2009) Nowcasting euro area economic activity in real time: the role of confidence indicators. Natl Inst Econ Rev 210:90–97. https://doi.org/10.1177/0027950109354413
Giannone D, Reichlin L, Small D (2005) Nowcasting GDP and inflation: the real time informational content of macroeconomic data releases. Technical Report Centre for Economic Policy Research. https://cepr.org/active/publications/discussion_papers/dp.php?dpno=5178
Gluonts (2022) GluonTS - probabilistic time series modeling in python. https://ts.gluon.ai/stable/
Grosse R (2017) Lecture 15: exploding and vanishing gradients. http://www.cs.toronto.edu/~rgrosse/courses/csc321_2017/readings/L15%20Exploding%20and%20Vanishing%20Gradients.pdf
Guichard S, Rusticelli E (2011) A dynamic factor model for world trade growth. In: OECD Economics Department Working Papers, https://doi.org/10.1787/5kg9zbvvwqq2-en. https://www.oecd-ilibrary.org/content/paper/5kg9zbvvwqq2-en

Hallin M, Liška R (2011) Dynamic factors in the presence of blocks. J Econom 163:29–41. https://doi.org/10.1016/j.jeconom.2010.11.004
Hopp D (2021) nowcastLSTM. https://github.com/dhopp1/nowcastLSTM
Hopp D (2022) nowcasting_benchmark. https://github.com/dhopp1/nowcasting_benchmark
Hopp D (2022) Economic nowcasting with long short-term memory artificial neural networks (LSTM). J Off Stat 38:847–873. https://doi.org/10.2478/jos-2022-0037
Hopp D (2022) Performance of LSTM neural networks in nowcasting global trade during the COVID-19 crisis. Stat J IAOS 38:1–14. https://doi.org/10.3233/SJI-210911
Hopp D, Cantú F (2020) nowcastDFM. https://github.com/dhopp1-UNCTAD/nowcastDFM publication Title: GitHub repository
IMF (2020) Gross domestic product: an economy's all. https://www.imf.org/external/pubs/ft/fandd/basics/gdp.htm
Jansen WJ, Jin X, Winter JMd (2016) Forecasting and nowcasting real GDP: comparing statistical models and subjective forecasts. Int J Forecast 32:411–436. https://doi.org/10.1016/j.ijforecast.2015.05.008
Kapoor A, Debroy B (2019) GDP is not a measure of human well-being. https://hbr.org/2019/10/gdp-is-not-a-measure-of-human-well-being
Kohzadi N, Boyd MS, Kermanshahi B, Kaastra I (1996) A comparison of artificial neural network and time series models for forecasting commodity prices. Neurocomputing 10:169–181. https://doi.org/10.1016/0925-2312(95)00020-8
Kronenberg P, Mikosch H, Neuwirth S (2021) The nowcasting lab. Technical Report KOF Swis Economic Institute Zurich. https://www.gpeari.gov.pt/documents/35086/229852/2022-15.11-nov-The+Nowcasting+Lab_Paper.pdf/456c3af1-ce10-1b73-494e-b840b8a0d05d?t=1668621627180
Kuzin VN, Marcellino M, Schumacher C (2009) MIDAS versus mixed-frequency VAR: nowcasting GDP in the euro area. Discussion Paper Series 1: Economic Studies 2009,07 Deutsche Bundesbank. https://ideas.repec.org/p/zbw/bubdp1/7576.html
Kuzin V, Marcellino M, Schumacher C (2011) MIDAS vs. mixed-frequency VAR: nowcasting GDP in the euro area. Int J Forecast 27:529–542. https://doi.org/10.1016/j.ijforecast.2010.02.006
Kvedaras V (2021) Package 'midasr'. https://cran.r-project.org/web/packages/midasr/midasr.pdf
Liew V (2004) Which lag selection criteria should we employ? Econ Bull 3:1–9
Loermann J, Maas B (2019) Nowcasting US GDP with artificial neural networks. MPRA Paper 95459 University Library of Munich, Germany. https://ideas.repec.org/p/pra/mprapa/95459.html
Mahaboob B, Bhumireddy V, Narayana C, Sankar J, Balasiddamuni P (2018) A treatise on ordinary least squares estimation of parameters of linear model. Int J Eng Technol (UAE), 7. https://doi.org/10.14419/ijet.v7i4.10.21216
Marcellino M, Schumacher C (2010) Factor MIDAS for nowcasting and forecasting with ragged-edge data: a model comparison for German GDP*. Oxford Bull Econ Stat 72:518–550. https://doi.org/10.1111/j.1468-0084.2010.00591.x
Mariano RS, Murasawa Y (2003) A new coincident index of business cycles based on monthly and quarterly series. J Appl Economet 18:427–443. https://doi.org/10.1002/jae.695
Mills TC (2019) Chapter 3: ARMA models for stationary time series. In: Mills TC (ed), Applied time series analysis (pp 31–56). Academic Press. https://doi.org/10.1016/B978-0-12-813117-6.00003-X
Morgado AJ, Nunes LC, Salvado S (2007) Nowcasting an economic aggregate with disaggregate dynamic factors: an application to Portuguese GDP. GEE Papers 0002 Gabinete de Estratégia e Estudos, Ministério da Economia. https://ideas.repec.org/p/mde/wpaper/0002.html
Natekin A, Knoll A (2013) Gradient boosting machines, a tutorial. Front Neurorobotics. https://doi.org/10.3389/fnbot.2013.00021
Patel H, Prajapati P (2018) Study and analysis of decision tree based classification algorithms. Int J Comput Sci Eng 6:74–78. https://doi.org/10.26438/ijcse/v6i10.7478
Porshakov A, Ponomarenko A, Sinyakov A (2016) Nowcasting and short-term forecasting of Russian GDP with a dynamic factor model. J New Econ Assoc 30:60–76
Probst P, Bischl B, Boulesteix A-L (2018) Tunability: importance of Hyperparameters of machine learning algorithms. arXiv:1802.09596
Richardson A, Mulder TvF, Vehbi T (2021) Nowcasting GDP using machine-learning algorithms: a real-time assessment. Int J Forecast 37:941–948. https://doi.org/10.1016/j.ijforecast.2020.10.005
Salinas D, Flunkert V, Gasthaus J, Januschowski T (2020) DeepAR: Probabilistic forecasting with autoregressive recurrent networks. Int J Forecast 36:1181–1191. https://doi.org/10.1016/j.ijforecast.2019.07.001

Sarker IH (2021) Machine learning: algorithms, real-world applications and research directions. SN Comput. Sci. 2:160. https://doi.org/10.1007/s42979-021-00592-x
Sazli M (2006) A brief review of feed-forward neural networks. Commun. Faculty Sci. Univ. Ankara 50:11–17. https://doi.org/10.1501/0003168
Schorfheide F, Song D (2015) Real-time forecasting with a mixed-frequency VAR. J Bus Econ Stat 33:366–380. https://doi.org/10.1080/07350015.2014.954707
Scikit learn (2021a) 1.10. Decision Trees. https://scikit-learn.org/stable/modules/tree.html
Scikit learn (2021b) 3.1. Cross-validation: evaluating estimator performance. https://scikit-learn.org/stable/modules/cross_validation.html
Scikit learn (2021c) scikit-learn machine learning in python. https://scikit-learn.org/stable/index.html
Shi X, Chen Z, Wang H, Yeung D-Y, Wong W-k, Woo W-C (2015) Convolutional LSTM network: a machine learning approach for precipitation nowcasting. arXiv:1506.04214
Sims CA (1980) Macroeconomics and reality. Econometrica 48:1–48. https://doi.org/10.2307/1912017
Singh R, Prajneshu (2008) Artificial neural network methodology for modelling and forecasting maize crop yield. Agric Econ Res Rev 21
Smith T (2021) pmdarima: ARIMA estimators for Python. https://alkaline-ml.com/pmdarima/index.html
Soley-Bori M (2013) Dealing with missing data: Key assumptions and methods for applied analysis. Technical Report No. 4 Boston University Boston. https://www.bu.edu/sph/files/2014/05/Marina-tech-report.pdf
Soybilgen B, Yazgan E (2021) Nowcasting US GDP using tree-based ensemble models and dynamic factors. Comput Econ 57:387–417. https://doi.org/10.1007/s10614-020-10083-5
Stock JH, Watson MW (2001) Vector autoregressions. J Econ Perspect 15:101–115. https://doi.org/10.1257/jep.15.4.101
Stock JH, Watson MW (2002) Forecasting using principal components from a large number of predictors. J Am Stat Assoc 97:1167–1179. https://doi.org/10.1198/016214502388618960
Stratos K (2020) Feedforward and recurrent neural networks. http://www1.cs.columbia.edu/~stratos/research/neural.pdf
Striaukas J, Babii A, Ghysels E, Kostrov A (2022) midasml: Estimation and prediction methods for high-dimensional mixed frequency time series data. https://cran.r-project.org/web/packages/midasml/index.html
Taylor R (2016) PyFlux. https://pyflux.readthedocs.io/en/latest/index.html
Tibshirani R (1996) Regression shrinkage and selection via the Lasso. J R Stat Soc Ser B (Methodological) 58:267–288
Tiffin A (2016) Seeing in the dark: a machine-learning approach to nowcasting in Lebanon. IMF Work Pap 16:1. https://doi.org/10.5089/9781513568089.001
WMO (2017) Guidelines for nowcasting techniques. Technical Report 1198 WMO. https://library.wmo.int/doc_num.php?explnum_id=3795
XGBoost Developers (2022) XGBoost Documentation. https://xgboost.readthedocs.io/en/stable/index.html
Zou H, Hastie T (2005) Regularization and variable selection via the elastic net. J R Stat Soc Ser B (Stat Methodol) 67:301–320