Machine Learning Methods and Time Series: a Through Comparison Study via Simulation and U.S. Inflation Forecasting

Klaus Boesch[a], Flavio A. Ziegelmann[b,*]

[a]*Instituto Federal Sul-rio-grandense, klausboesch@gmail.com*
[b]*Universidade Federal do Rio Grande do Sul - Departamento de Estatistica, flavioaz@mat.ufrgs.br*

Abstract

In this paper our main goal is to compare time series forecasting accuracy for several candidates within a wide collection of machine learning methods. In addition to the vast number of existing methods in the literature, we propose a new variation of the Elastic Net (ENet), the Weighted Lag Adaptive ENet (WLadaENet), which combines the popular Ridge Regression with a regularization method tailored for time series, the WLAdaLASSO (Konzen and Ziegelmann, 2016). The motivating idea is to increasingly penalize the coefficients of lagged variables as the lag increases. To achieve our goal, we carry out Monte Carlo simulation studies as well as a real data analysis of USA inflation. In our Monte Carlo implementations, the WLadaENet presents a good performance both in terms of variable selection when the true model is sparse and in terms of forecasting accuracy even when the model is not sparse and nonlinearities are included. WLadaENet also performs well to forecast the USA inflation. Nevertheless, L_2Boost is the best inflation forecaster for the USA data in the analyzed period.

Keywords: Time series, Machine learning, WLadaENet, Forecasting, Inflation

1. Introduction

High-dimensional machine methods have become increasingly important in the literature over the last few decades. In the context of time series analysis, the main reasons for this are certainly i) the forecasting accuracy gain obtained from a model which is built having a large number of potential covariates to select from, and ii) the availability of modern

*Corresponding author

methods capable of performing selection, estimation and forecasting in high-dimensional problems.

Many of the traditional methods present some drawbacks within the high-dimensional context (see Tibshirani, 1996; Fan, 2014), for instance:

1. Estimates often have low bias but large variance, reducing the accuracy of the forecasts.
2. They can lack model interpretability. Considering a large set of predictors, it is preferable to determine a smaller subset of predictors that most explain the variability of the response variable.
3. When the number of predictors exceeds the sample size, ordinary least squares (OLS) does not produce an identifiable solution.
4. Spurious correlation can play a strong negative role.

Machine learning (ML) methods are able to handle some of the issues above. These methods are designed to improve out-of-sample prediction. Gu et al. (2020) and Mullainathan and Spiess (2017) point out some key characteristics of (high-dimensional) ML methods: ML contains a diverse collection of high-dimensional methods for statistical prediction that combine two elements, namely, regularization and empirical tuning. The high-dimensional nature of these methods enhances their flexibility relative to more traditional econometric prediction techniques (Gu et al., 2020). ML methods typically have a regularizing element associated with them and the empirical tuning allows one to choose the level of regularization appropriately (Mullainathan and Spiess, 2017).

In a seminal work, Tibshirani (1996) introduces the least absolute shrinkage and selection operator (LASSO), which soon became a benchmark in the regularization literature. Zou (2006) investigates the oracle properties of LASSO, showing that it is not consistent in variable selection in some situations. Then the adaLASSO (adaptive LASSO) is proposed by Zou (2006), with distinct penalties for each of the coefficients, inheriting consistency in variable selection under very general assumptions. Medeiros and Mendes (2016) study the asymptotic properties of adaLASSO in sparse, high-dimensional, linear time-series models and find that its properties allow the adaLASSO to be applied to a variety of applications in empirical finance and macroeconomics. The authors also present an application to forecast U.S. inflation using many predictors, where adaLASSO delivers superior forecasts compared to traditional benchmarks such as autoregressive and factor models. Medeiros and Vasconcelos (2016) employ high-dimensional methods (LASSO-family and *bagging*) to forecast macroeconomic variables, and show that these methods provide smaller forecast errors than autoregressive and factor models. Medeiros et al. (2016) use LASSO and adaLASSO to forecast Brazilian inflation and observe that LASSO-based methods have the smallest errors for short-horizon forecasts. Konzen and Ziegelmann (2016) test

LASSO-type penalty methods for covaraite selection and forecasting, and propose the WLadaLASSO (which is based on further penalizing lagged predictors) method, having good results for U.S. risk premium and U.S. inflation forecasting. Medeiros et al. (2019) added a variety of high-dimensional methods to forecast U.S. inflation, such as bagging, Ridge Regression, Elastic Net (and its adaptive version) and Jackknife Model Averaging (JMA), finding that Random Forests (RF) robustly outperforms the other methods. Gu et al. (2020) provide a comparative analysis of methods in the machine learning repertoire, using these methods to forecast stock returns, and identify the best performing methods as trees-based models and neural networks.

In this work we employ a variety of ML methods (including methods based on shrinkage, regression trees and boosting) to perform time series forecasting. In particular, we propose a method we call WLadaENet (weighted lag adaptive Elastic Net), which combines quadratic regularization (ridge) with adaptive weighted LASSO shrinkage similarly to adaENet introduced by Zou and Zhang (2009), but further penalizes coefficients of higher lag order terms as the WLadaLASSO proposed by Konzen and Ziegelmann (2016). To compare the finite sample performances of the ML methods considered in our work, particularly against WLadaENet, a Monte Carlo (MC) simulation study is carried out, with three different data-generating processes. In summary, the MC studies highlight the good performance of our proposal. WLadaENet presents a good performance in terms of variable selection when the model is sparse and in terms of forecasting even when the model is not sparse and nonlinearities are included. Furthermore, in our empirical analysis of the USA inflation forecasting, WLadaENet performs well, been only slightly behind L_2Boost.

Besides this introduction, the paper comprises 3 more sessions. Section 2 describes all the forecasting machine learning methods implemented in this work, including our novel WLadaENet proposal. In Section 3 all the Monte Carlo (MC) simulations as well as empirical analysis are performed. Section 4 brings the concluding remarks.

2. Machine Learning Methods

2.1. Traditional Shrinkage Methods

Considering the linear time series model

$$y_{t+h} = \beta_0 + \beta_1 x_{t,1} + ... + \beta_q x_{t,q} + u_{t+h}, \quad t = 1, ..., T, \tag{1}$$

where the forecast of y_{t+h}, denoted as $\hat{y}_{t+h}$ is given by

$$\hat{y}_{t+h} = \hat{\beta}_0 + \hat{\beta}_1 x_{t,1} + ... + \hat{\beta}_q x_{t,q}, \quad t = 1, ..., T. \tag{2}$$

This section presents a class of methods that shrink the regression coefficients by imposing a penalty on their size, i.e, the coefficients in 1 are obtained by solving the following problem:

$$\hat{\beta}(\boldsymbol{\lambda}) = \underset{\beta_0,...,\beta_q}{\arg\min} \left\{ \underbrace{\sum_{t=1}^{T-h} \left(y_{t+h} - \beta_0 - \sum_{j=1}^{q} \beta_j x_{t,j} \right)^2}_{RSS(\beta_0,...,\beta_q)} + p(\beta_1, ..., \beta_q; \boldsymbol{\lambda}) \right\}, \tag{3}$$

where $\hat{\beta}(\boldsymbol{\lambda}) = (\hat{\beta}_0, \hat{\beta}_1(\boldsymbol{\lambda}), ..., \hat{\beta}_q(\boldsymbol{\lambda}))$, $p(\cdot)$ is a penalty function and $\boldsymbol{\lambda}$ is a vector of tuning parameters. In Table 1 we summarize some very traditional shrinkage methods along with their specific penalty functions.

Table 1: Penalty functions ($\hat{\beta}_j^*$ are obtained in a previous first stage)

Method	**Penalty:** $p(\beta_1, ..., \beta_q; \boldsymbol{\lambda})$
Ridge	$\lambda \sum_{j=1}^{q} \beta_j^2$
LASSO	$\lambda \sum_{j=1}^{q} \lvert\beta_j\rvert$
ENet	$\lambda[\rho \sum_{j=1}^{q} \lvert\beta_j\rvert + (1-\rho) \sum_{j=1}^{q} \beta_j^2]$
adaLASSO	$\lambda \sum_{j=1}^{q} \lvert\hat{\beta}_j^*\rvert^{-\tau} \lvert\beta_j\rvert$
adaENet	$\lambda[\rho \sum_{j=1}^{q} \lvert\hat{\beta}_j^*\rvert^{-\tau} \lvert\beta_j\rvert + (1-\rho) \sum_{j=1}^{q} \beta_j^2]$
WLadaLASSO	$\lambda \sum_{j=1}^{q} (\lvert\hat{\beta}_j^*\rvert e^{-\alpha l_j})^{-\tau} \lvert\beta_j\rvert$
WLadaENet	$\lambda[\rho \sum_{j=1}^{q} (\lvert\hat{\beta}_j^*\rvert e^{-\alpha l_j})^{-\tau} \lvert\beta_j\rvert + (1-\rho) \sum_{j=1}^{q} \beta_j^2]$

2.2. Weighted Lag Adaptive Elastic Net (WLadaENet)

We propose a method that we call Weighted Lag Adaptive Elastic Net (WLadaENet), a combination between adaENet and WLadaLASSO, where the idea is similar to the adaptive Elastic Net, but penalizing further the coefficients of higher-lagged covariates in a time series context. The penalization of WLadaENet is given by

$$p(\cdot) = \lambda \left\{ \rho \sum_{j=1}^{q} \hat{\omega}_j^* |b_j| + (1-\rho) \sum_{j=1}^{q} b_j^2 \right\}. \tag{4}$$

Here, the estimator is obtained as the solution of the following minimization problem:

$$\hat{\beta}^{WLadaENet} = \underset{b_0,...,b_q}{\arg\min} \left\{ RSS(b_0, ..., b_q) + \lambda \left[\rho \sum_{j=1}^{q} \hat{\omega}_j^* |b_j| + (1-\rho) \sum_{j=1}^{q} b_j^2 \right] \right\}, \tag{5}$$

where $\omega_j^* = \left[\left(|\hat{\beta}_j^*|\right) e^{-\alpha l_j}\right]^{-\tau}$, if either OLS or Ridge are employed in the first stage. In turn, if LASSO or ENet are employed, then $\hat{\omega}_j^* = \left[\left(|\hat{\beta}_j^*| + T^{-1}\right) e^{-\alpha l_j}\right]^{-\tau}$. Similarly to WLadaLASSO, $\tau > 0$, $\alpha \geq 0$, l_j represents the lag order and $\hat{\beta}_j^*$, $j = 1, \ldots, q$, are the coefficient estimates of the first stage.

2.3. Ensemble Methods

According to Hastie et al. (2009), ensemble learning consists in constructing a prediction method $\hat{F}_h$ by combining the strengths of simple base estimators $\hat{f}_{h,b}$, such that the prediction rule is given by

$$\hat{y}_{t+h} = \hat{F}_h(\boldsymbol{x}_t) = \sum_{b=1}^{B} \lambda_b \hat{f}_{h,b}(\boldsymbol{x}_t), \tag{6}$$

where λ_b can be seen as a learning rate or a weight.

2.3.1. Complete Subset Regression (CSR)

Subset selection methods retain only a subset of the covariates, eliminating the others from the model. Usually, least squares regression is employed to estimate the coefficients of the retained covariates. Elliott et al. (2013) introduce the Complete Subset Regression which, for a given set of potential covariates, combines forecasts from all possible linear regressions with a fixed number of covariates. For a set of K covariates, or predictor candidates, there are $n_{k,K} = K!/[(K-k)!k!]$ combinations of $k \leq K$ variables. Elliott et al. (2015) consider large-dimensional sets of potential predictors where CSR is unfeasible and indicate a pre-testing procedure as a possible solution.

Consider the linear model $y_{t+h} = \gamma^T \boldsymbol{z}_t + \delta^T \boldsymbol{w}_t + u_{t+h}$, $t = 1, \ldots, T$, where $\boldsymbol{z}_t$ is a $Px1$ vector of predictors which are always included in the forecast equation, and $\boldsymbol{w}_t$ is a $Kx1$ vector of potential predictors. We follow Garcia et al. (2017) and Medeiros et al. (2019) and use a pre-testing procedure where for each variable in $\boldsymbol{w}_t$ we fit a linear regression of y_{t+h} by OLS and use the absolute values of the t-statistic to select the $\tilde{K} < K$ most relevant variables. The CSR forecast is given by

$$\hat{y}_{t+h} = B^{-1} \sum_{b=1}^{B} \hat{\beta}_b^T \boldsymbol{x}_t = \underbrace{\left[B^{-1}\left(\sum_{b=1}^{B} \hat{\beta}_b^T\right)\right]}_{(\hat{\beta}^{CSR})^T} \boldsymbol{x}_t, \tag{7}$$

where $B = n_{k,\tilde{K}} = \tilde{K}!/[(\tilde{K}-k)!k!]$, $\boldsymbol{x}_t = (\boldsymbol{z}_t, \boldsymbol{w}_t)$ and $\hat{\beta}_b = (\hat{\gamma}_b^T, \hat{\delta}_b^T)^T$, $b = 1, ..., B$.

2.3.2. Component-Wise L_2 Boosting (L_2Boost)

Boosting is a procedure that combines the outputs of many *base* or *weak learners* iteratively in order to achieve high accuracy (Bühlmann and Yu, 2003; Hastie et al., 2009). According to Friedman (2002) gradient boosting constructs additive regression models by sequentially fitting a simple parameterized function (base learner) to current "pseudo"-residuals by least-squares at each iteration, where the "pseudo" residuals are the gradient of the loss functional being minimized, with respect to the model values at each training data point, evaluated at the current step. The boosting technology builds an ensemble model by conducting a regularized and supervised search in a high-dimensional space of weak learners (Hastie et al., 2009).

Bühlmann and Yu (2003) present a computationally simple variant of boosting algorithms, L_2Boost, which is constructed from a functional gradient descent algorithm employing the L_2-loss function. For L_2Boost method the loss function is given by $L(y_{t+h}, \hat{F}(\boldsymbol{x}_t)) = [y_{t+h} - \hat{F}(\boldsymbol{x}_t)]^2/2$ such that the gradient of $L(.)$ is $\hat{u}_{t+h} = y_{t+h} - \hat{F}(\boldsymbol{x}_t), \quad t = 1, ..., T-h$. Based on Bühlmann and Yu (2003), and Bai and Ng (2009) the Componentwise L_2Boost algorithm sequentially fits q simple linear models until a stopping rule is reached. The L_2Boost forecast is given by

$$\hat{y}_{t+h} = \hat{\beta}_0 + \hat{\beta}_{1,B^*} x_{t,1} + ... + \hat{\beta}_{q,B^*} x_{t,q}. \tag{8}$$

In order to avoid overfit Bühlmann (2006) proposes a stopping rule using the corrected AIC_c criterion. For time series, Bai and Ng (2009) use the BIC as a stopping rule.

2.3.3. Random Forest (RF)

Breiman (2001) proposes the Random Forest to improve on the variance reduction capability of *bagging* predictors (Breiman, 1996) by reducing the correlation between the trees. Regression trees partition the space of predictors into disjoint regions $\{R_k\}_{k=1}^K$. According to Hastie et al. (2009) a regression tree with K regions (terminal nodes or leaves) can be formally defined by the equation

$$T(\boldsymbol{x}_t; \boldsymbol{\theta}) = \sum_{k=1}^{K} \beta_k \mathbb{I}_{R_k}(\boldsymbol{x}_t), \tag{9}$$

with parameters $\boldsymbol{\theta} = \{R_k, \beta_k\}_{k=1}^K$ and where $\mathbb{I}_{R_k}(.)$ is a product of indicator functions such that $\mathbb{I}_{R_k}(\boldsymbol{x}_t)$ indicates whether $\boldsymbol{x}_t \in R_k$.

Minimizing the sum of squares, $\hat{\beta}_k$ is given by the average of those y_{t+h} in region $\hat{R}_k$

$$\hat{\beta}_k = \frac{\sum_{t=1}^{T-h} \mathbb{I}_{\hat{R}_k}(\boldsymbol{x}_t) y_{t+h}}{\sum_{t=1}^{T-h} \mathbb{I}_{\hat{R}_k}(\boldsymbol{x}_t)}. \tag{10}$$

The RF prediction is an ensemble of trees predictions, resulting in

$$\hat{y}_{t+h} = \hat{F}_B^{RF}(\boldsymbol{x}_t) = B^{-1} \sum_{b=1}^{B} \underbrace{\left[\sum_{k=1}^{K_b} \hat{\beta}_{k,b} \mathbb{I}_{\hat{R}_{k,b}}(\boldsymbol{x}_t) \right]}_{T_b(\boldsymbol{x}_t, \hat{\boldsymbol{\theta}}_b)} . \tag{11}$$

2.3.4. Boosting Trees (B.Trees)

For boosting regression trees, Friedman (2001) considers the case where each base learner is a K-terminal node regression tree. So the update at each iteration has the form

$$\hat{F}_b(\boldsymbol{x}_t) = \hat{F}_{b-1}(\boldsymbol{x}_t) + \lambda \sum_{k=1}^{K} \hat{\beta}_{k,b} \mathbb{I}_{\hat{R}_{k,b}}(\boldsymbol{x}_t), \tag{12}$$

where $\{\hat{R}_{k,b}\}_{k=1}^{K}$ are the regions defined by the terminal nodes of the tree at the b_{th} iteration. The trees are constructed to predict the pseudo-responses $\{\hat{u}_{t+h,b}\}_{t=1}^{T-h}$ by least squares. The $\{\hat{\beta}_{k,b}\}_{k=1}^{K}$ are the corresponding least squares coefficients

$$\hat{\beta}_{k,b} = \frac{\sum_{t=1}^{T-h} \mathbb{I}_{\hat{R}_{k,b}}(\boldsymbol{x}_t) \hat{u}_{t+h,b}}{\sum_{t=1}^{T-h} \mathbb{I}_{\hat{R}_{k,b}}(\boldsymbol{x}_t)} . \tag{13}$$

The forecast of y_{t+h} is given by

$$\hat{y}_{t+h} = \hat{F}_B^{B.Trees}(\boldsymbol{x}_t) = \hat{\beta}_0 + \lambda \sum_{b=1}^{B} \underbrace{\left[\sum_{k=1}^{K} \hat{\beta}_{k,b} \mathbb{I}_{R_{k,b}}(\boldsymbol{x}_t) \right]}_{T_b(\boldsymbol{x}_t, \hat{\boldsymbol{\theta}}_b)}, \tag{14}$$

where, for all b, $K_b = K = d + 1$ and $\lambda \in (0, 1]$.

2.4. Factor Methods

The following methods avoid high-dimensional problems by using the common factors. We consider the following model:

$$w_{t,j} = \lambda_j^T G_t + e_{t,j} , \tag{15}$$

where G_t is the vector of common factors, λ_j is a vector of loadings associated with G_t, and $e_{t,j}$ is the idiosyncratic component of $w_{t,j}$, $t = 1, ..., T - h$ and $j = 1, ..., q$.

2.4.1. Factors

Here, the h-period-ahead forecast, using data for $t = 1, \ldots, T - h$, is given by

$$y_{t+h} = \gamma^T \boldsymbol{z}_t + \delta^T \tilde{g}_t + u_{t+h} \ , \tag{16}$$

where $\boldsymbol{z}_t$ is a vector of predetermined variables, $\tilde{g}_t$ includes lags of $\hat{g}_t$, $\hat{g}_t \subseteq \hat{G}_t$, and $\hat{G}_t$ are the principal components estimates of the vector G_t in the factor model. The key task in factors based methods is the correct specification of the number of factors. In this work we follow Medeiros et al. (2019). The forecast can be written as

$$\hat{y}_{t+h} = \hat{\beta}_{BIC}^T \boldsymbol{x}_t \ , \tag{17}$$

where $\boldsymbol{x}_t = (\boldsymbol{z}_t, \tilde{g}_t)$, $\boldsymbol{z}_t$ includes lags of y_t, $\tilde{g}_t$ include lags of $\hat{g}_t$ and $\hat{\beta}_{BIC} = (\hat{\gamma}_{BIC}^T, \hat{\delta}_{BIC}^T)^T$ is selected using the BIC from a coefficients set $\{\hat{\beta}^{OLS}(l)\colon\ l = 1, .., L\}$ estimated by OLS for each lag.

2.4.2. Boosting Factors

To select predictors from a large set of candidates, where they have no natural ordering, Bai and Ng (2009) propose the use of *boosting* in factor-augmented autoregressions. The *boosting* algorithm employed is the same presented earlier for the L_2Boost method, where the BIC value is used as a stopping rule to prevent overffiting. The forecast is given by

$$\hat{y}_{t+h} = (\hat{\beta}^{L_2 Boost})^T \boldsymbol{x}_t \ , \tag{18}$$

where $\boldsymbol{x}_t = (\boldsymbol{z}_t, \tilde{g}_t)$, $\boldsymbol{z}_t$ includes lags of y_t, $\tilde{g}_t$ include lags of $\hat{g}_t$ and $\hat{\beta}^{L_2 Boost}$ is estimated through L_2Boost method.

3. Numerical Implementations

In order to evaluate the forecasting performance of the statistical methods presented earlier, we carry out several numerical exercises, including Monte Carlo simulations and an empirical data analysis. All implementations are performed using software R. For all shrinkage methods we used the package *glmnet*. The RF method is implemented using the package *randomForestSRC* and B.Trees using the package *gbm*. For the shrinkage methods the parameters λ and $\alpha \in \{0, 0.5, 1, ..., 10\}$, whereas the order of WL methods are selected according to the BIC. The parameter ρ of ENet and its adaptive versions is set to 1/2 (1/3 in *glmnet* function). For the adaptive methods we employ Ridge Regression in the first stage to compute the weights ω_j. For the CSR method we fix $\tilde{K} = 20$ and $k = 4$, following Medeiros et al. (2019) and using the first four lags of y_t as fixed controls. The method L_2Boost employs the minimum BIC as the stopping rule, where we use $\lambda = 0.2$

and the maximum number of iterations $B = 10q$. For RF and B.Trees we use the number of trees $B = 500$ and the minimum number of observations by leaf $n_{min} = 15$. The number of splits of B.Trees is fixed at $d = 2$, while for the RF this parameter is not fixed. RF is implemented employing non-overlapping blocks bootstrap where the block size is 4. In the factor methods we use the first four Principal Components.

The parameters of the AR model are estimated by Ordinary Least Squares (OLS) and the order p is determined by BIC. A set of AR models $\{AR(p) : p \in \{1, ..., L\} \subset \mathbb{N}\}$ is estimated and the selected model is the one which has minimum BIC. For the empirical applications the number of lags used is $L = 4$.

3.1. Metrics of Prediction Performance

Following Garcia et al. (2017) and Medeiros et al. (2019) in this work we employ a direct forecast approach such that the h periods ahead response variable, y_{t+h} is modeled as a function of a set of predictors measured at up to time t, considering the following general model:

$$y_{t+h} = F_h(\boldsymbol{x}_t) + u_{t+h}, \quad h = 1, \ldots, H, \quad t = 1, \ldots, T, \tag{19}$$

where y_{t+h} is the dependent variable in period $t + h$ and $\boldsymbol{x}_t = (x_{t,1}, ..., x_{t,q})^T \in \mathbb{X} \subseteq \mathbb{R}^q$ is a set of covariates which contains only variables observed and available at time t. $F_h(.)$ is the mapping between covariates and y_{t+h} and u_{t+h} is the forecasting error. There is a different mapping $F_h(.)$ for each forecasting horizon h and for each method. We use a fixed length rolling-window scheme for all methods. Adopting a notation similar to Medeiros et al. (2019), the direct forecast equation is given by

$$\hat{y}_{t+h|t} = \hat{F}_{h,t-T_h^w+1:t}(\boldsymbol{x}_t), \tag{20}$$

where $\hat{F}_{h,t-T_h^w+1:t}$ is the estimated target function based on data from time $t - T_h^w + 1$ up to t and T_h^w is the window size. The window size varies depending on the forecasting horizon h and the number of lagged variables used in the method.

All methods are evaluated based on a fixed number $T_{PF} = T - T_0$ of point forecasts. For each forecast horizon $h = 1, ..., H$ the methods are compared according the root mean square error (RMSE) and the mean absolute error (MAE), which are defined as follows:

$$\text{RMSE} = \sqrt{T_{PF}^{-1} \sum_{t=T_0+1}^{T} (\hat{u}_{t+h})^2} \quad \text{and} \quad \text{MAE} = T_{PF}^{-1} \sum_{t=T_0+1}^{T} |\hat{u}_{t+h}|, \tag{21}$$

where $\hat{u}_{t+h} = y_{t+h} - \hat{y}_{t+h}$.

Besides, Superior Predictive Ability (Hansen, 2005; Quaedvlieg, 2019) and Model Confidence Set (Hansen et al., 2011) are performed to choose the best methods.

3.2. Monte Carlo Simulation

In this section we analyze and compare the performance of almost all the methods presented before through a Monte Carlo simulation study. As our simulated data does not have a factor structure we do not report results for factors methods in this section. We perform Monte Carlo simulations with 1000 replications, simulating $n = 10$ independent time series with an AR(1) structure:

$$x_{t,j} = \phi x_{t,j} + \epsilon_{t,j}, \tag{22}$$

where $\phi = 0.5$ and $\epsilon_{t,j} \sim \mathcal{NIID}(0, 1)$, $j = 1, ..., n$.

We consider three different data-generating processes (DGP).

DGP1: sparse model.

$$\begin{aligned} y_t = {} & 0.8y_{t-1} + 0.6x_{t-1,1} + 0.3x_{t-2,1} - 0.5x_{t-1,2} - 0.2x_{t-2,2} \\ & + 0.4x_{t-1,3} + 0.3x_{t-2,3} + 0.4x_{t-1,4} - 0.3x_{t-1,5} + 0.2x_{t-1,6} + u_t, \quad t = 1, ..., T, \end{aligned} \tag{23}$$

where $u_t \sim \mathcal{N}(0, 1)$ and all u_t are mutually independent.

DGP2: dense linear model.

$$\text{DGP 2: } y_t = \sum_{\ell=1}^{L} a_\ell \left(y_{t-\ell}\right) + \sum_{\ell=1}^{L} \sum_{j=1}^{n} b_{\ell,j} \left(x_{t-\ell,j}\right) + u_t, \quad t = 1, ..., T, \tag{24}$$

where $a_\ell = 0.8 \left(-0.5\right)^{\ell-1}$, $b_{\ell,j} = \left(0.5\right)^{\ell} \left(-1\right)^{\ell+j-1}$, $u_t \sim \mathcal{N}(0, 1)$ and all u_t are mutually independent. In this case, for each value of L, we generate and analyse the simulated data employing all lagged variables from 1 to L, thus all coefficients are nonzero.

DGP3: dense nonlinear model.

$$\text{DGP 3: } y_t = \sum_{l=1}^{L} \left\{a_l \left[g \left(y_{t-l}; a_l\right)\right]\right\} + \sum_{l=1}^{L} \sum_{j=1}^{n} \left\{b_{l,j} \left[g \left(x_{t-l,j}; b_{l,j}\right)\right]\right\} + u_t \ , \ t = 1, ..., T, \tag{25}$$

where $g(z; c) = z/(1 + |c|z^2)$. In terms of Taylor's expansion $g(z; c) = z - |c|z^3 + |c|^2 z^5 - |c|^3 z^7 + |c|^4 z^9$..., such that the DGP3 is similar to the DGP2 plus a nonlinear part.

3.2.1. Variable and Model Selection

Restricted to linear models, for the first two DGP specifications we additionally report in Table 2 some statistics related to model/variable selection.

Table 2: Simulation results: Descriptive statistics of models selection for DGP 1 and DGP 2

	DGP 1: Sparse model						DGP 2: Dense model					
	4 candidate lags			12 candidate lags			4 candidate lags			12 candidate lags		
T	150	500	1000	150	500	1000	150	500	1000	150	500	1000
	FVCI			FVCI			FVCI			FVCI		
Ridge	0.2273	0.2273	0.2273	0.0758	0.0758	0.0758	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**
LASSO	0.8928	0.9342	0.9472	0.9490	0.9744	0.9790	0.4114	0.4625	0.8274	0.1643	0.1300	0.1323
ENet	0.7389	0.7740	0.7749	0.8746	0.9142	0.9176	0.4873	**0.5516**	**0.9073**	0.1932	0.1546	0.1535
adaLASSO	0.9339	0.9771	0.9850	0.9575	0.9906	0.9946	0.3316	0.3567	0.5960	0.1392	0.1013	0.1131
adaENet	0.9225	0.9727	0.9825	0.9459	0.9887	0.9935	0.3469	0.3669	0.6058	0.1518	0.1044	0.1135
WLadaLASSO	0.9562	0.9900	0.9943	**0.9680**	0.9967	0.9981	0.2941	0.3874	0.5320	0.0851	0.1310	0.1674
WLadaENet	**0.9566**	**0.9906**	**0.9952**	0.9668	**0.9968**	**0.9984**	0.2904	0.3874	0.5334	0.0847	0.1317	0.1674
CSR	0.6230	0.6667	0.6763	0.8415	0.8563	0.8588	0.4545	0.4545	0.4545	0.1515	0.1515	0.1515
L_2Boost	0.8329	0.8771	0.8867	0.8202	0.9015	0.9153	**0.4938**	0.5328	0.6928	**0.3612**	**0.2578**	**0.2793**
	TMI			TMI			TMI			TMI		
Ridge	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**
LASSO	0.5922	0.9947	**1.0000**	0.2995	0.9757	**1.0000**	0.0000	0.0013	0.0714	0.0000	0.0000	0.0000
ENet	**0.7800**	**0.9990**	**1.0000**	0.4745	0.9907	**1.0000**	0.0000	**0.0037**	**0.1518**	0.0000	0.0000	0.0000
adaLASSO	0.4075	0.9529	0.9990	0.0565	0.9153	0.9987	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000
adaENet	0.4397	0.9543	0.9990	0.0643	0.9278	0.9990	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000
WLadaLASSO	0.4243	0.9735	**1.0000**	0.0637	0.9717	**1.0000**	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000
WLadaENet	0.4563	0.9793	**1.0000**	0.0829	0.9786	**1.0000**	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000
CSR	0.0001	0.0012	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000
L_2Boost	0.7574	0.9998	**1.0000**	**0.6485**	**0.9988**	**1.0000**	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000
	FRVI			FRVI			FRVI			FRVI		
Ridge	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**
LASSO	0.9451	0.9995	**1.0000**	0.8756	0.9975	**1.0000**	0.4114	0.4625	0.8274	0.1643	0.1300	0.1323
ENet	**0.9766**	0.9999	**1.0000**	0.9300	0.9991	**1.0000**	0.4873	**0.5516**	**0.9073**	0.1932	0.1546	0.1535
adaLASSO	0.9145	0.9952	0.9999	0.7698	0.9913	0.9999	0.3316	0.3567	0.5960	0.1392	0.1013	0.1131
adaENet	0.9246	0.9953	0.9999	0.7924	0.9927	0.9999	0.3469	0.3669	0.6058	0.1518	0.1044	0.1135
WLadaLASSO	0.9009	0.9973	**1.0000**	0.7007	0.9971	**1.0000**	0.2941	0.3874	0.5320	0.0851	0.1310	0.1674
WLadaENet	0.9070	0.9979	**1.0000**	0.7050	0.9978	**1.0000**	0.2904	0.3874	0.5334	0.0847	0.1317	0.1674
CSR	0.6705	0.7667	0.7878	0.4542	0.5516	0.5678	0.4545	0.4545	0.4545	0.1515	0.1515	0.1515
L_2Boost	0.9728	**1.0000**	**1.0000**	**0.9586**	**0.9999**	**1.0000**	**0.4938**	0.5328	0.6928	**0.3612**	**0.2578**	**0.2793**
	FIVE			FIVE			FIVE			FIVE		
Ridge	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	-	-	-	-	-	-
LASSO	0.8775	0.9150	0.9317	0.9550	0.9725	0.9773	-	-	-	-	-	-
ENet	0.6690	0.7075	0.7087	0.8700	0.9072	0.9108	-	-	-	-	-	-
adaLASSO	0.9396	0.9717	0.9806	0.9728	0.9905	0.9942	-	-	-	-	-	-
adaENet	0.9219	0.9661	0.9773	0.9585	0.9884	0.9930	-	-	-	-	-	-
WLadaLASSO	**0.9725**	0.9878	0.9926	**0.9899**	0.9966	0.9979	-	-	-	-	-	-
WLadaENet	0.9712	**0.9884**	**0.9938**	0.9883	**0.9968**	**0.9982**	-	-	-	-	-	-
CSR	0.6090	0.6372	0.6435	0.8733	0.8813	0.8826	-	-	-	-	-	-
L_2Boost	0.7917	0.8410	0.8534	0.8089	0.8934	0.9084	-	-	-	-	-	-
	NIV			NIV			NIV			NIV		
Ridge	44.0000	44.0000	44.0000	132.0000	132.0000	132.0000	**44.0000**	**44.0000**	**44.0000**	**132.0000**	**132.0000**	**132.0000**
LASSO	13.6171	12.8855	12.3235	14.2468	13.3297	12.770	18.101	20.3482	36.4060	21.6831	17.162	17.4666
ENet	21.0193	19.944	19.9051	25.1581	21.3117	20.8805	21.4407	**24.2682**	**39.9232**	25.4983	20.4126	20.259
adaLASSO	11.1996	10.9134	10.6582	**11.0108**	11.0684	10.7089	14.5902	15.6959	26.2219	18.3707	13.3685	14.9258
adaENet	11.9012	11.1070	10.7692	12.9883	11.3414	10.8542	15.2643	16.1446	26.6532	20.0439	13.7855	14.985
WLadaLASSO	9.9441	10.3865	10.2527	8.2447	10.3812	10.2553	12.9418	17.0468	23.4098	11.2356	17.2919	22.1009
WLadaENet	**10.0491**	**10.3726**	**10.212**	8.4804	**10.3736**	**10.2171**	12.778	17.0455	23.4694	11.1768	17.3846	22.0988
CSR	20.0000	20.0000	20.0000	20.0000	20.0000	20.0000	20.0000	20.0000	20.0000	20.0000	20.0000	20.0000
L_2Boost	16.8083	15.4067	14.9853	32.9025	23.0002	21.1782	**21.728**	23.4424	30.4811	**47.6759**	**34.0347**	**36.8712**

Note: statistics related to model/variable selection, where FVCI is the average fraction of variables correctly identified; TMI is the fraction of replications where the true model is included; FRVI is the average fraction of relevant variables included; FIVE is the fraction of irrelevant variables excluded and NIV is the number of included variables. For each specification, the value of statistic for the best performing method is highlighted in bold. Ridge Regression always select all variables and does not exclude any, while CSR always selects $\tilde{K} = 20$ variables and excludes the remaining ones.

Although the Ridge Regression is not a variable selection method, we included its statistics as a reference (when all variables are selected) to compare to the performances of other methods. For each method, sample size and lag order, we report: (1) the average fraction of variables correctly identified (FVCI); (2) the fraction of replications where the true model is included (TMI); (3) the average fraction of relevant variables included (FRVI); (4) the average fraction of irrelevant variables excluded (FIVE) and (5) the average number of included variables (NIV). While Ridge Regression, by construction, always select all variables and does not exclude any, the CSR always selects $\tilde{K} = 20$ variables and excludes the remaining ones. For each specification, the value of the best performing method is highlighted in bold.

Considering DGP1, the adaptive methods, especially WLadaENet, have the best performances for the statistics FVCI, FIVE and NIV. When we consider the statistics TMI and FRV, the methods ENet and L_2Boost perform better than the other methods, while CSR rarely includes the true model (TMI near or equal to zero). For DGP2, as all coefficients are different from zero, the statistic FIVE is zero for all methods and for each specification. For the TMI statistic (besides Ridge) only LASSO and ENet present values different from zero but very small, only for $L = 4$ and $T \geq 500$. ENet, L_2Boost (for $T = 150$ and 500) and LASSO (for $T = 1000$) present the best performance in terms of the FVCI, FRVI and NIV statistics when $L = 4$. For $L = 12$, L_2Boost has the best performance (excluding Ridge) for FVCI, FRVI and NIV.

3.2.2. Forecasting

In terms of forecasting, we consider the three DGP specifications. We remove the last 10 observations of each simulated time series data end employ the methods to perform the one-step-ahead out-of-sample forecast for these observations using a rolling-window scheme, where the window has size $T - 10$. We analyze situations where $L \in \{4, 12\}$ and $T \in \{150, 500, 1000\}$.

Table 3 shows the results for the one-step-ahead forecasts. We report the mean values of RMSE and MAE across replications, and indicate in bold the method with the lowest forecasting error for each specification. The cells in gray/blue indicate that the method is included in the 50% Model Confidence Set (MCS) using the squared/absolute error as loss function. For DGP1 and DGP2 the methods with best performance in terms of errors are WLadaLASSO and WLadaENet, except for DGP2 when $L = 4$ and $T = 1000$, where ENet has the lowest errors and is the only method included in the MCS. As we use all point forecasts of all replications and, consequently, this data is very informative, we have at most three methods included in the MCS and one or two methods in most cases. For DGP3, which is nonlinear, when we have four lags the methods RF, B.Trees and L_2Boost have the best performance for the smallest simulated sample, and are the

only methods included in the MCS. For the moderate and the largest sample sizes, Ridge has the lowest errors and is the only method in the MCS. Finally, for 12 lags, the model becomes approximately sparse, then WLadaLASSO and WLadaENet present the lowest errors, being the best performing methods.

Table 3: Simulation results: Forecasting accuracy for the DGP 1, 2 and 3

Mean of RMSEs	DGP 1: Sparse model						DGP 2: Dense model						DGP 3: Nonlinear model					
(Mean of MAEs)	4 candidate lags			12 candidate lags			4 candidate lags			12 candidate lags			4 candidate lags			12 candidate lags		
T	150	500	1000	150	500	1000	150	500	1000	150	500	1000	150	500	1000	150	500	1000
RW	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000
	(1.0000)	(1.0000)	(1.0000)	(1.0000)	(1.0000)	(1.0000)	(1.0000)	(1.0000)	(1.0000)	(1.0000)	(1.0000)	(1.0000)	(1.0000)	(1.0000)	(1.0000)	(1.0000)	(1.0000)	(1.0000)
AR	0.9262	0.9175	0.9159	0.9269	0.9176	0.9159	0.9109	0.8983	0.8967	0.9172	0.9035	0.9027	0.7565	0.7525	0.7504	0.7558	0.7501	0.75
	(0.9275)	(0.9200)	(0.9183)	(0.9282)	(0.9199)	(0.9183)	(0.9105)	(0.8986)	(0.8972)	(0.9205)	(0.9074)	(0.9061)	(0.7671)	(0.7632)	(0.7608)	(0.7695)	(0.7634)	(0.7636)
Ridge	0.7274	0.6486	0.6342	1.8334	0.7171	0.6655	0.6007	0.523	0.5126	1.0237	0.5739	0.5338	0.7619	**0.6616**	**0.6479**	0.7334	0.7266	0.6733
	(0.7254)	(0.6451)	(0.6323)	(1.8488)	(0.7189)	(0.6652)	(0.5998)	(0.5225)	(0.5126)	(1.0286)	(0.5722)	(0.5337)	(0.7748)	**(0.6723)**	**(0.6588)**	(0.7444)	(0.7352)	(0.684)
LASSO	0.6456	0.5912	0.5809	0.6990	0.6023	0.5855	0.5906	0.5302	0.4952	0.6521	0.5520	0.5331	0.7491	0.6887	0.6683	0.7494	0.708	0.679
	(0.6443)	(0.5914)	(0.5809)	(0.6962)	(0.6021)	(0.5856)	(0.5883)	(0.529)	(0.4959)	(0.6513)	(0.5539)	(0.5339)	(0.759)	(0.6984)	(0.6782)	(0.7649)	(0.7187)	(0.6897)
ENet	0.6641	0.5989	0.5842	0.7572	0.6201	0.5954	0.5972	0.5282	**0.4921**	0.6964	0.5637	0.5404	0.7553	0.6938	0.6697	0.7535	0.7193	0.6829
	(0.6622)	(0.5970)	(0.5831)	(0.7530)	(0.6182)	(0.5942)	(0.5946)	(0.5264)	**(0.4934)**	(0.6947)	(0.5655)	(0.5413)	(0.7652)	(0.7039)	(0.6794)	(0.7691)	(0.7313)	(0.6937)
adaLASSO	0.6301	0.5836	0.5758	0.7115	0.586	0.5762	0.5696	0.5229	0.5002	0.6342	0.5259	0.5159	0.7406	0.681	0.6626	0.7441	0.6866	0.6649
	(0.6290)	(0.5840)	(0.5763)	(0.7099)	(0.5869)	(0.5766)	(0.5676)	(0.5219)	(0.5011)	(0.6328)	(0.5269)	(0.5169)	(0.7521)	(0.6914)	(0.6732)	(0.7592)	(0.6963)	(0.6753)
adaENet	0.6310	0.5842	0.5760	0.728	0.5867	0.5764	0.5722	0.5235	0.5000	0.6482	0.5275	0.5167	0.7441	0.6811	0.6624	0.7449	0.6891	0.6664
	(0.6297)	(0.5844)	(0.5765)	(0.7254)	(0.5872)	(0.5768)	(0.5699)	(0.5223)	(0.5006)	(0.6475)	(0.5286)	(0.5175)	(0.7552)	(0.6912)	(0.6726)	(0.7606)	(0.6989)	(0.6764)
WLadaLASSO	**0.6168**	**0.5799**	**0.5746**	0.6634	**0.5805**	**0.5747**	0.5548	**0.5155**	0.4966	0.5476	0.5093	0.4905	0.7263	0.6700	0.6591	0.7241	**0.6662**	0.6554
	(0.6151)	**(0.5804)**	(0.5751)	(0.6635)	**(0.5806)**	**(0.5754)**	**(0.5537)**	**(0.5149)**	(0.4972)	**(0.5469)**	(0.5102)	(0.4907)	(0.7368)	(0.6811)	(0.6692)	(0.7366)	**(0.6748)**	(0.6654)
WLadaENet	0.6170	0.5804	**0.5746**	**0.6633**	0.5806	0.5749	**0.5546**	0.5157	0.4969	**0.5474**	**0.5088**	**0.4904**	0.7258	0.6698	0.659	**0.72**	0.6663	**0.6553**
	(0.6156)	(0.5808)	**(0.575)**	**(0.6632)**	**(0.5806)**	**(0.5754)**	**(0.5537)**	**(0.5149)**	(0.4976)	**(0.5469)**	**(0.5097)**	**(0.4906)**	(0.7356)	(0.6806)	(0.6696)	**(0.7326)**	(0.6749)	**(0.665)**
CSR	1.3605	1.3028	1.2857	1.4464	1.3489	1.3245	0.9554	0.9366	0.9308	0.9750	0.9313	0.9259	0.7302	0.7233	0.7223	0.7367	0.7233	0.7207
	(1.3633)	(1.3082)	(1.2897)	(1.4459)	(1.3522)	(1.3261)	(0.9583)	(0.9379)	(0.9323)	(0.9775)	(0.9371)	(0.9310)	(0.7405)	(0.7331)	(0.7323)	(0.7497)	(0.736)	(0.7332)
L_2Boost	0.6265	0.5852	0.5772	0.6660	0.5909	0.5800	0.5762	0.5248	0.5057	0.6230	0.5331	0.5192	0.7202	0.6761	0.6621	0.7449	0.6818	0.6657
	(0.6263)	(0.5856)	(0.5775)	(0.6658)	(0.5921)	(0.5805)	(0.5739)	(0.5236)	(0.5060)	(0.6227)	(0.5342)	(0.5205)	(0.7306)	(0.6859)	(0.6722)	(0.7558)	(0.691)	(0.6758)
RF	1.3050	1.0176	0.9151	1.3858	1.0666	0.9561	0.9367	0.8420	0.8008	0.9805	0.8739	0.8303	0.7196	0.691	0.6776	0.7322	0.706	0.6918
	(1.2686)	(0.9999)	(0.9039)	(1.3495)	(1.0495)	(0.9450)	(0.9349)	(0.8382)	(0.7974)	(0.9823)	(0.8755)	(0.8315)	(0.7295)	(0.7014)	(0.6874)	(0.7452)	(0.7179)	(0.7028)
B.Trees	1.4524	1.0613	0.9660	1.6040	1.0661	0.9676	0.9410	0.8390	0.8177	1.0001	0.8353	0.8110	**0.7194**	0.687	0.675	0.7327	0.6957	0.6769
	(1.4109)	(1.0401)	(0.9504)	(1.5656)	(1.0446)	(0.9518)	(0.9400)	(0.8354)	(0.8140)	(1.0004)	(0.8357)	(0.8121)	**(0.7292)**	(0.6984)	(0.685)	(0.7476)	(0.7075)	(0.6878)

Note: The table shows the means of root mean squared errors (RMSE) and means of mean absolute errors (MAE) in parenthesis for the forecasts across replications, relative to the Random Walk (RW). The values in bold indicate the method with lowest values of RMSE and MAE for each horizon. Cells in gray/blue indicate that the method is included in the 50% MCS constructed based on the T_{max} statistic using the squared (or absolute) errors.

3.3. Empirical Analysis: U.S. Inflation

In this section we employ all statistical methods seen previously to forecast U.S. inflation. We use the variables from the FRED-MD[1] database compiled by McCracken and Ng (2016), performing forecasts for the Consumer Price Index (CPI) in log change. The dataset span ranges from January 1960 to December 2018, having 708 monthly observations and 122 variables classified in 8 groups as seen in Table 4.

Our in-sample period spans from January 1960 to December 2012, leading to an in-sample size of 636 observations, with $q = 488$ covariates (considering 122 variables and 4 lags for each). Therefore, we have 72 out-of-sample observations to evaluate our forecasts, from January 2013 to December 2018 (see Table 5). Furthermore, Figure 1 presents the time series plot of U.S. inflation, where the red interval indicates the period for which we perform the forecasts.

[1]The FRED-MD is a large database containing monthly observations of macroeconomic variables, designed for the analysis of *big data* and is updated in real-time thorough the FRED database. The FRED-MD is available at https://research.stlouisfed.org/econ/mccracken/fred-databases/.

Table 4: Variable Groups Summary

	Description	**Number of variables**
Group 1	Output and income	16
Group 2	Labor market	31
Group 3	Housing	10
Group 4	Consumption, orders, and inventories	7
Group 5	Money and credit	13
Group 6	Interest and exchange rates	21
Group 7	Prices	20
Group 8	Stock market	4

Table 5: Empirical application summary

	Variables	**Lags**	**q**	**In-sample**	**Sample size**	**Out-of-Sample**	**Point forecasts**
CPI U.S. large sample	122	4	488	Jan 1960-Dec 2012	636	Jan 2013-Dec 2018	72

In Table 6 we report the results for the inflation forecast accuracy, all relative to the Random Walk (RW). This table shows the root mean squared error (RMSE) and the mean absolute error (MAE) for all forecasting methods. These forecast accuracy measures are displayed for horizons from 1 to 12 months ahead as well as for the cumulative inflation for 3, 6 and 12 months ahead. The gray/blue cells indicate that the method is included in

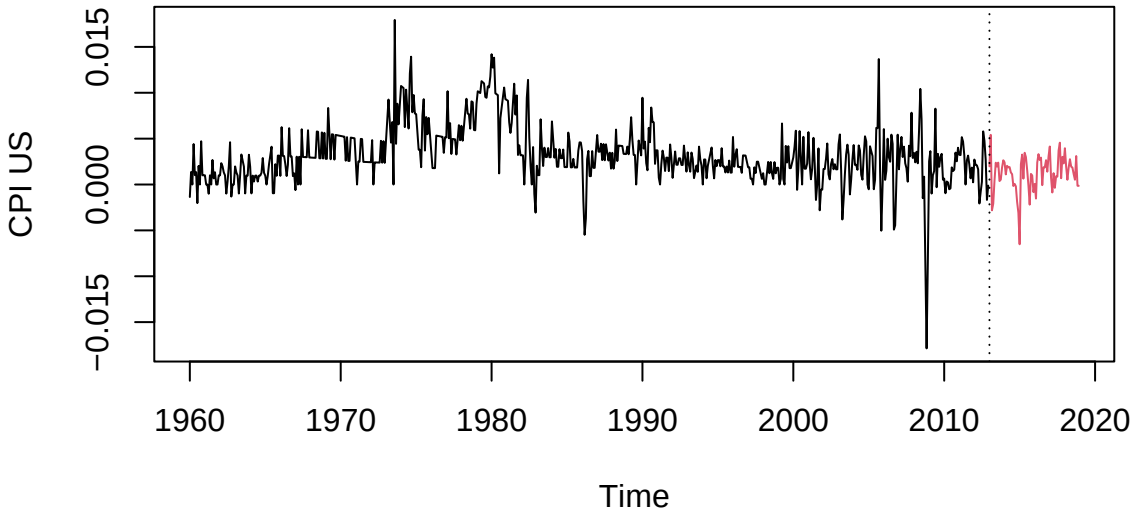

Figure 1: Time series U.S. inflation

the 50% Model Confidence Set (MCS) using the squared (or absolute in brackets) error as loss function. As it can be seen, the RW, AR and Ridge are almost never included in the MCS. On the opposite side, L_2Boost is the only method that is always in the MCS. Our proposal, WLadaENet, comes second in that respect, only not being in the MCS for horizon 6 under the RMSE loss.

Table 6: Forecasting Accuracy for Predicting U.S. Inflation from 2013 to 2018: RMSE, MAE and MCS

	Consumer price index (U.S.) 2013-2018 Forecast horizon														
RMSE/(MAE)	1	2	3	4	5	6	7	8	9	10	11	12	3m	6m	12m
RW	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00
	(1.00)	(1.00)	(1.00)	(1.00)	(1.00)	(1.00)	(1.00)	(1.00)	(1.00)	(1.00)	(1.00)	(1.00)	(1.00)	(1.00)	(1.00)
AR	0.87	0.79	0.81	0.86	0.86	0.78	0.72	0.73	0.75	0.84	0.87	0.78	0.90	0.97	1.43
	(0.92)	(0.81)	(0.82)	(0.85)	(0.89)	(0.75)	(0.66)	(0.67)	(0.72)	(0.81)	(0.78)	(0.68)	(0.94)	(0.95)	(1.38)
Ridge	0.96	0.91	0.83	0.96	0.98	0.85	0.87	0.92	0.95	1.08	1.03	0.89	0.85	0.82	1.21
	(1.00)	(0.89)	(0.84)	(0.96)	(1.03)	(0.87)	(0.83)	(0.89)	(0.96)	(1.11)	(0.98)	(0.81)	(0.87)	(0.84)	(1.28)
LASSO	0.78	0.76	0.75	0.81	0.79	0.71	0.65	0.67	0.69	0.77	0.73	0.74	0.79	0.80	1.08
	(0.80)	(0.76)	(0.70)	(0.78)	(0.79)	(0.67)	(0.59)	(0.61)	(0.65)	(0.71)	(0.66)	(0.66)	(0.80)	(0.75)	(1.02)
ENet	0.79	0.76	0.76	0.80	0.83	0.70	0.65	0.67	0.70	0.78	0.75	0.74	0.81	0.82	1.12
	(0.81)	(0.77)	(0.72)	(0.77)	(0.80)	(0.66)	(0.59)	(0.61)	(0.65)	(0.73)	(0.68)	(0.64)	(0.83)	(0.77)	(1.04)
adaLASSO	0.80	**0.72**	0.75	0.79	0.80	0.72	0.68	0.69	0.71	0.76	0.72	0.71	0.77	0.76	1.00
	(0.80)	(0.73)	(0.71)	(0.75)	(0.78)	(0.68)	(0.62)	(0.62)	(0.67)	(0.73)	(0.66)	(0.67)	(0.75)	(0.69)	(0.91)
adaENet	0.79	0.73	0.75	0.79	0.80	0.73	0.67	0.69	0.68	0.75	0.72	0.70	0.78	0.78	1.01
	(0.79)	(0.73)	(0.71)	(0.75)	(0.79)	(0.70)	(0.61)	(0.62)	(0.64)	(0.72)	(0.65)	(0.66)	(0.77)	(0.72)	(0.94)
WLadaLASSO	**0.74**	0.74	**0.72**	0.78	0.78	0.72	0.64	0.69	0.68	0.76	**0.71**	0.70	0.76	0.75	0.96
	(0.73)	(0.74)	(0.69)	(0.73)	(0.76)	(0.68)	(0.59)	(0.62)	(0.64)	(0.73)	(0.66)	(0.66)	(0.75)	(0.68)	(0.90)
WLadaENet	**0.74**	0.74	0.75	**0.75**	0.79	0.73	0.64	0.69	**0.67**	0.74	**0.71**	**0.68**	0.78	0.75	0.96
	(0.74)	(0.76)	(0.72)	**(0.71)**	(0.76)	(0.69)	**(0.56)**	**(0.60)**	(0.63)	(0.71)	**(0.64)**	(0.63)	(0.78)	(0.68)	(0.90)
CSR	0.80	0.73	0.74	0.77	0.77	0.69	0.66	0.67	**0.67**	**0.73**	0.73	0.70	0.77	0.75	1.00
	(0.84)	(0.74)	(0.70)	(0.76)	(0.78)	(0.67)	(0.62)	(0.64)	(0.64)	**(0.70)**	(0.68)	(0.63)	(0.78)	(0.70)	(0.94)
L_2Boost	0.78	0.73	**0.72**	0.79	**0.74**	**0.67**	**0.62**	**0.64**	**0.67**	0.75	0.74	0.72	**0.73**	**0.70**	**0.93**
	(0.77)	**(0.72)**	**(0.68)**	(0.75)	**(0.72)**	(0.64)	**(0.56)**	(0.61)	(0.65)	(0.73)	(0.69)	(0.65)	**(0.72)**	**(0.64)**	**(0.87)**
RF	0.78	0.73	0.73	0.78	0.77	0.69	0.64	0.67	0.69	0.76	0.75	0.71	0.78	0.77	1.02
	(0.81)	(0.73)	(0.69)	(0.73)	(0.76)	(0.64)	**(0.56)**	**(0.60)**	(0.63)	**(0.70)**	(0.68)	**(0.62)**	(0.78)	(0.70)	(0.91)
B.Trees	0.81	0.75	0.76	0.78	0.77	0.68	0.64	0.69	0.70	0.78	0.78	0.73	0.84	0.81	1.13
	(0.86)	(0.73)	(0.72)	(0.74)	(0.76)	**(0.63)**	(0.58)	**(0.60)**	**(0.61)**	(0.71)	(0.69)	(0.63)	(0.85)	(0.76)	(1.05)
Factors	0.83	0.80	0.80	0.82	0.77	0.73	0.71	0.70	0.71	0.76	0.75	0.73	0.86	0.90	1.23
	(0.86)	(0.81)	(0.78)	(0.81)	(0.78)	(0.70)	(0.67)	(0.65)	(0.67)	(0.71)	(0.68)	(0.65)	(0.87)	(0.85)	(1.21)
B.Factors	0.81	0.76	0.76	0.81	0.80	0.73	0.68	0.70	0.71	0.78	0.77	0.73	0.82	0.85	1.19
	(0.85)	(0.77)	(0.76)	(0.81)	(0.82)	(0.70)	(0.61)	(0.64)	(0.67)	(0.72)	(0.68)	(0.64)	(0.83)	(0.82)	(1.15)
Mean	0.77	0.73	0.73	0.77	0.76	0.70	0.65	0.66	0.68	0.75	0.73	**0.68**	0.77	0.75	0.95
	(0.78)	(0.73)	(0.70)	(0.73)	(0.76)	(0.67)	(0.59)	(0.62)	(0.65)	(0.71)	(0.67)	**(0.62)**	(0.77)	(0.69)	**(0.87)**
Median	0.78	0.73	0.74	0.78	0.77	0.70	0.65	0.67	0.68	0.75	0.72	0.70	0.77	0.76	1.00
	(0.79)	(0.73)	(0.70)	(0.75)	(0.77)	(0.67)	(0.59)	(0.61)	(0.64)	**(0.70)**	(0.65)	**(0.62)**	(0.77)	(0.71)	(0.93)

Note: The table shows the root mean squared errors (RMSE) and mean absolute errors (MAE) in parenthesis for the forecasts, relative to the Random Walk (RW). The values in bold indicate the method with lowest values of RMSE and MAE for each horizon. Cells in gray/blue indicate that the method is included in the 50% MCS constructed based on the T_{max} statistic using the squared/absolute errors.

We also apply the Superior Predictive Analysis (SPA) tests of i) Hansen (2005) to compare the forecasting methods for each forecasting horizon (plus the cumulative ones) and ii) Quaedvlieg (2019)) for the Uniform and Average Multi-Horizon SPA versions. Table 7 reports on its left-hand-side the p-values of Hansen (2005) SPA test using each method as benchmark for each forecasting horizon. Quaedvlieg (2019) SPA test p-values,

for both Uniform and Average Multi-Horizon versions are reported on the right-hand-side of this table, having only RW and AR as benchmarks. Panel (a) presents the p-values for the test using RMSE and Panel (b) using MAE. The null hypothesis for both single and multi-horizon is that the benchmark method is not inferior. The gray cells indicate that the null hypothesis is rejected at the 0.05 significance level. For the single-horizon SPA test the methods LASSO, ENet, adaLASSO, adaENet, WLadaLASSO, WLadaENet, L_2Boost, RF and B.Trees are statistically not inferior to the others for all forecasting horizons. The multi-horizon tests show that RW is not uniformly inferior to the Ridge, while AR is also not uniformly inferior to Ridge and Factors model and is not on average inferior to Ridge.

Table 7: Superior Predictive Ability Test (U.S. Inflation, 2013-2018)

Panel (a): Squared errors

	Hansen's test – Forecasting horizon															Quaedvlieg's test			
Benchmark	1	2	3	4	5	6	7	8	9	10	11	12	3m	6m	12m	Unif.(RW)	Avg.(RW)	Unif.(AR)	Avg.(AR)
RW	0.012	0.017	0.008	0.059	0.062	0.005	0.003	0.003	0.001	0.013	0.004	0.004	0.046	0.020	0.723	-	-	0.930	0.992
AR	0.042	0.105	0.089	0.222	0.224	0.072	0.199	0.286	0.296	0.046	0.087	0.232	0.080	0.020	0.055	0.000	0.008	-	-
Ridge	0.005	0.049	0.044	0.069	0.027	0.004	0.020	0.024	0.017	0.002	0.004	0.068	0.253	0.320	0.245	0.292	0.045	0.879	0.969
LASSO	0.290	0.414	0.509	0.379	0.425	0.160	0.167	0.271	0.528	0.240	0.642	0.376	0.496	0.261	0.298	0.000	0.003	0.000	0.005
ENet	0.185	0.325	0.341	0.429	0.336	0.313	0.235	0.394	0.376	0.193	0.311	0.373	0.370	0.272	0.271	0.000	0.003	0.000	0.007
adaLASSO	0.226	0.971	0.466	0.499	0.450	0.055	0.151	0.122	0.212	0.480	0.757	0.536	0.542	0.334	0.646	0.000	0.001	0.000	0.011
adaENet	0.221	0.889	0.437	0.492	0.433	0.043	0.163	0.138	0.837	0.640	0.910	0.712	0.450	0.278	0.418	0.000	0.004	0.000	0.010
WLadaLASSO	0.928	0.745	0.876	0.395	0.388	0.078	0.518	0.115	0.779	0.493	0.888	0.674	0.735	0.551	0.760	0.000	0.000	0.000	0.009
WLadaENet	0.806	0.649	0.457	0.931	0.529	0.053	0.523	0.179	0.988	0.786	0.945	0.890	0.556	0.512	0.759	0.000	0.003	0.000	0.005
CSR	0.134	0.918	0.486	0.558	0.608	0.451	0.272	0.474	0.907	0.990	0.610	0.746	0.466	0.446	0.722	0.000	0.003	0.000	0.009
L_2Boost	0.341	0.861	0.844	0.328	0.943	0.820	0.948	0.938	0.751	0.550	0.453	0.471	0.964	0.985	0.972	0.000	0.000	0.000	0.012
RF	0.322	0.801	0.756	0.503	0.511	0.439	0.640	0.329	0.640	0.461	0.343	0.555	0.437	0.371	0.392	0.000	0.002	0.000	0.008
B.Trees	0.143	0.429	0.345	0.538	0.442	0.634	0.404	0.171	0.376	0.201	0.144	0.456	0.317	0.316	0.256	0.000	0.001	0.000	0.005
Factors	0.083	0.119	0.110	0.122	0.428	0.081	0.045	0.081	0.187	0.414	0.378	0.439	0.181	0.051	0.099	0.000	0.002	0.136	0.009
B.Factors	0.122	0.370	0.342	0.102	0.157	0.023	0.053	0.079	0.188	0.245	0.143	0.420	0.184	0.100	0.153	0.000	0.002	0.000	0.004
Mean	0.331	0.903	0.834	0.539	0.651	0.192	0.230	0.414	0.882	0.820	0.595	0.999	0.582	0.414	0.613	0.000	0.002	0.000	0.004
Median	0.372	0.963	0.587	0.388	0.355	0.163	0.299	0.268	0.828	0.721	0.872	0.831	0.491	0.308	0.442	0.000	0.004	0.000	0.005

Panel (b): Absolute errors

	Hansen's test – Forecasting horizon															Quaedvlieg's test			
Benchmark	1	2	3	4	5	6	7	8	9	10	11	12	3m	6m	12m	Unif.(RW)	Avg.(RW)	Unif.(AR)	Avg.(AR)
RW	0.002	0.007	0.002	0.011	0.024	0.000	0.000	0.000	0.000	0.004	0.000	0.000	0.024	0.004	0.534	-	-	0.941	0.994
AR	0.022	0.142	0.035	0.134	0.080	0.064	0.154	0.300	0.038	0.077	0.110	0.354	0.025	0.013	0.045	0.000	0.005	-	-
Ridge	0.006	0.060	0.029	0.013	0.004	0.006	0.006	0.024	0.004	0.002	0.002	0.040	0.161	0.119	0.112	0.395	0.037	0.848	0.966
LASSO	0.181	0.434	0.516	0.087	0.146	0.203	0.340	0.846	0.390	0.782	0.722	0.474	0.295	0.225	0.346	0.000	0.001	0.000	0.007
ENet	0.079	0.493	0.449	0.262	0.316	0.455	0.328	0.864	0.316	0.590	0.362	0.594	0.270	0.188	0.320	0.000	0.000	0.000	0.002
adaLASSO	0.170	0.863	0.505	0.429	0.180	0.107	0.100	0.538	0.160	0.585	0.634	0.294	0.674	0.542	0.798	0.000	0.001	0.004	0.014
adaENet	0.207	0.819	0.418	0.442	0.132	0.065	0.135	0.671	0.516	0.736	0.823	0.384	0.572	0.331	0.677	0.000	0.004	0.002	0.008
WLadaLASSO	0.920	0.742	0.712	0.781	0.452	0.150	0.249	0.522	0.496	0.586	0.679	0.355	0.765	0.696	0.894	0.000	0.002	0.004	0.007
WLadaENet	0.700	0.540	0.328	0.945	0.406	0.146	0.718	0.846	0.655	0.849	0.991	0.722	0.481	0.654	0.874	0.000	0.001	0.000	0.012
CSR	0.025	0.877	0.621	0.345	0.192	0.276	0.061	0.398	0.485	0.882	0.375	0.846	0.399	0.462	0.731	0.000	0.002	0.000	0.010
L_2Boost	0.422	0.918	0.893	0.531	0.982	0.692	0.744	0.687	0.351	0.474	0.331	0.437	0.957	0.937	0.811	0.000	0.002	0.008	0.018
RF	0.145	0.812	0.807	0.641	0.515	0.705	0.860	0.920	0.747	0.903	0.389	0.944	0.512	0.494	0.637	0.000	0.005	0.000	0.007
B.Trees	0.119	0.755	0.349	0.529	0.500	0.819	0.497	0.833	0.927	0.705	0.275	0.812	0.221	0.225	0.316	0.000	0.006	0.000	0.002
Factors	0.083	0.179	0.091	0.048	0.277	0.130	0.127	0.178	0.181	0.726	0.352	0.614	0.082	0.034	0.088	0.000	0.003	0.102	0.012
B.Factors	0.038	0.386	0.192	0.036	0.017	0.033	0.076	0.309	0.161	0.588	0.319	0.683	0.186	0.082	0.141	0.000	0.000	0.000	0.006
Mean	0.167	0.962	0.674	0.851	0.522	0.298	0.224	0.797	0.281	0.891	0.601	0.998	0.576	0.547	0.974	0.000	0.003	0.000	0.009
Median	0.129	0.944	0.653	0.450	0.297	0.251	0.299	0.939	0.521	0.996	0.911	0.995	0.527	0.387	0.747	0.000	0.000	0.000	0.009

Note: The table reports the p-values of Hansen (2005) SPA test on the left using each method as benchmark for each forecasting horizon. The p-values of the uniform and average multi-horizon Quaedvlieg (2019) SPA tests are also reported (right) using RW and AR as benchmarks. Panel (a) uses RMSE whereas Panel (b) uses MAE. The null hypothesis is that the benchmark method is not inferior. The gray cells indicate that the null hypothesis is rejected at the 0.05 significance level.

Figure 2, inspired by Medeiros et al. (2019), shows the plots of the variables relative importance (aggregated by variable groups) for all twelve forecasting horizons and for all methods, except the univariate and factors based methods.

Similarly to Medeiros et al. (2019), for methods based on the linear model (shrinkage methods, CSR and L_2Boost) the relative importance is computed as the average coefficient

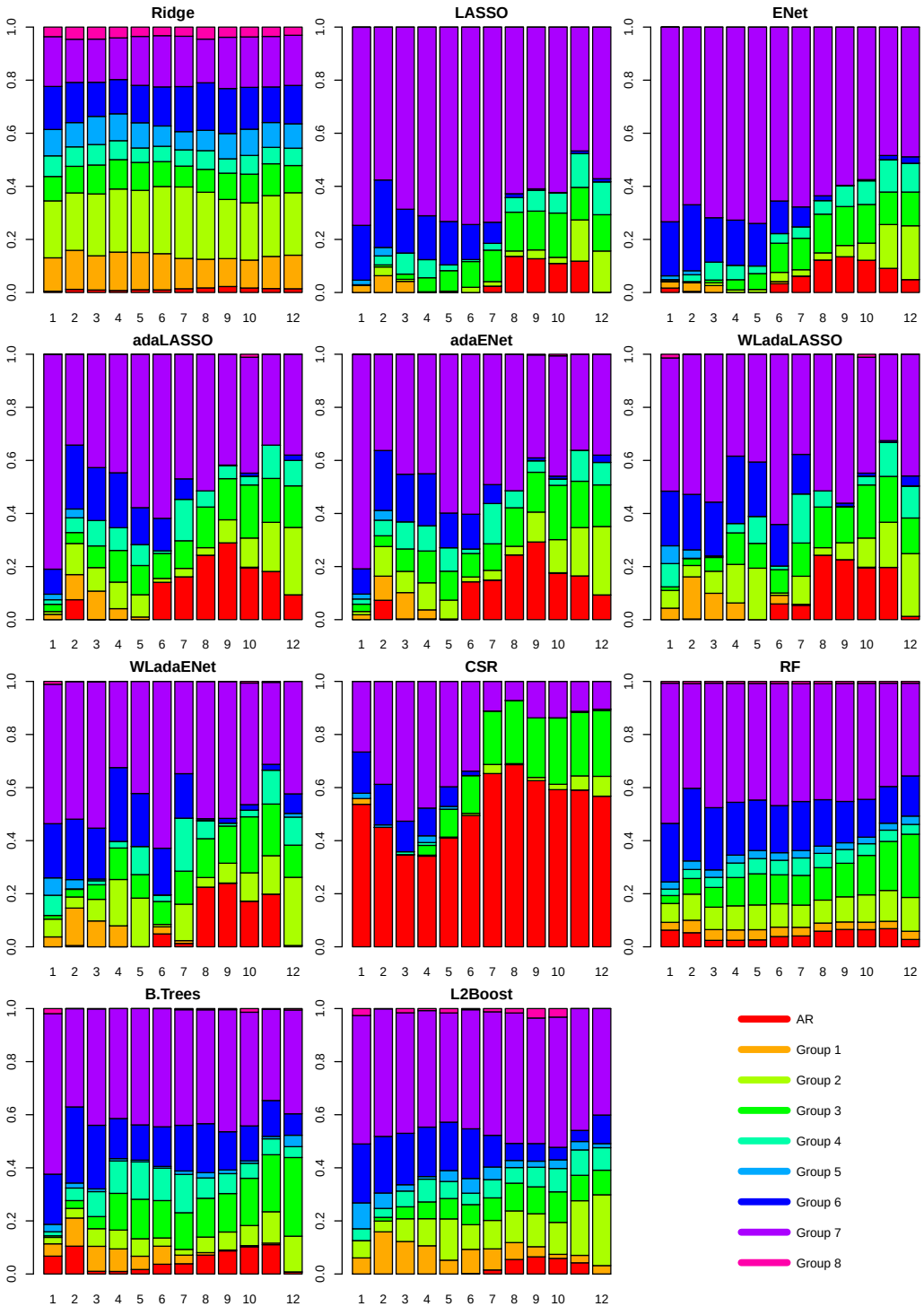

Figure 2: Variable importance for U.S. CPI

size. All variables are standardized as zero mean and unity variance. For RF the Out-of-Bag (OOB) samples are used to compute the variable importance, while for B.Trees all samples are employed. The OOB samples are the observations which are not selected in the bootstrap sample process, for each $b = 1, ..., B$. The samples are passed down the b_{th} tree when it is grown and the accuracy is recorded, then the values for the variable j are permuted at random, and the accuracy is computed once more. Hence, the decrease in accuracy is averaged over all trees and is used as a measure of the importance of variable j in the forest Hastie et al. (2009). AR terms (autorregressive, or past inflation) are very relevant for CSR. Prices (Group 7) are, in general, quite relevant for all methods. Interest and exchange rates (Group 6) also play an important role, especially for RF and B.Trees.

Finally, Figure 3 illustrates the forecasts for horizons 1 and 12 of the methods that had good performances in our analysis.

4. Concluding Remarks

Our two-fold study aims i) studying a variety of machine learning methods capable of performing time series forecasting and ii) proposing a new method, which we name WLadaENet, tailored for time series analysis. In order to evaluate the forecasting performances of these methods we carry out several numerical exercises, including Monte Carlo simulations and an empirical data analysis of the U.S.inflation.

Through our Monte Carlo implementation, we simulate three different data-generating process, trying to explore different levels of sparsity and different degrees of nonlinearities in the model. WLadaLASSO and WLadaENet (our new proposal) have the best performance in terms of variable selection and forecast in most cases, even when nonlinearities are present.

For the U.S. inflation forecasts, the more modern ML methods have statistically superior relative performances against the simpler RW and AR benchmarks. Overall, considering all forecast horizons L_2Boost has the best performance followed closely by our proposal, the WLadaENet.

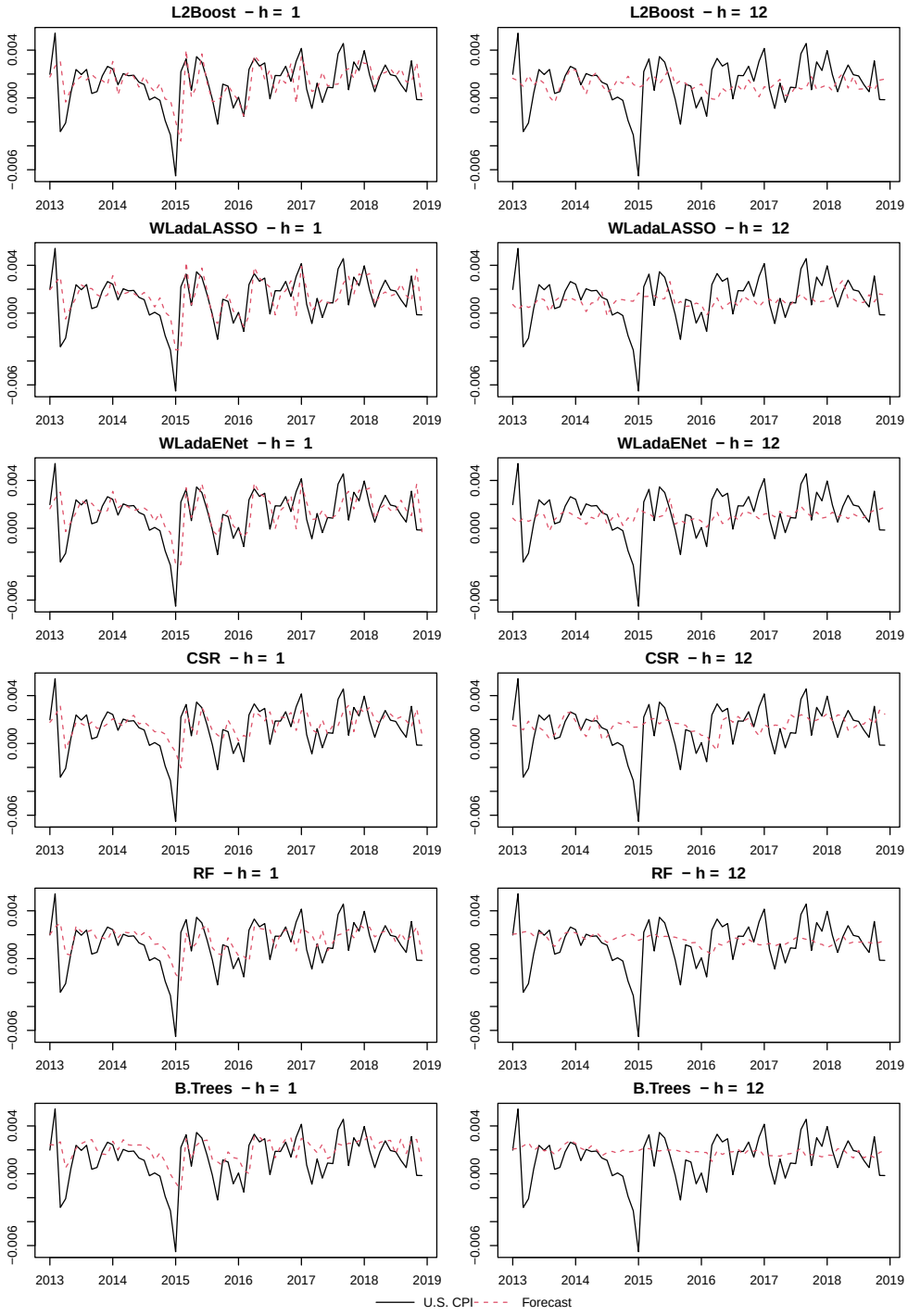

Figure 3: U.S. CPI time series forecasts

References

Bai, J., Ng, S., 2009. Boosting diffusion indices. Journal of Applied Econometrics 24, 607–629.

Breiman, L., 1996. Bagging Predictors. Machine Learning 24, 123–140.

Breiman, L., 2001. Random Forests. Machine Learning 45, 5–32.

Bühlmann, P., 2006. Boosting for High-Dimensional Linear Models. The Annals of Statistics 34, 559–583.

Bühlmann, P., Yu, B., 2003. Boosting With the L2 Loss. Journal of the American Statistical Association 98, 324–339.

Elliott, G., Gargano, A., Timmermann, A., 2013. Complete subset regressions. Journal of Econometrics 177, 357 – 373. Dynamic Econometric Modeling and Forecasting.

Elliott, G., Gargano, A., Timmermann, A., 2015. Complete subset regressions with large-dimensional sets of predictors. Journal of Economic Dynamics and Control 54, 86 – 110.

Fan, J., 2014. Features of big data and sparsest solution in high confidence set, in: Lin, X., Genest, C., Banks, D.L., Molenberghs, G., Scott, D.W., Wang, J.L. (Eds.), Past, Present and Future of Statistical Science. 1 ed.. Chapman & Hall, New York. chapter 34, pp. 507–523.

Friedman, J.H., 2001. Greedy Function Approximation: A Gradient Boosting Machine. The Annals of Statistics 29, 1189–1232.

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

Garcia, M.G.P., Medeiros, M.C., Vasconcelos, G.F.R., 2017. Real-time inflation forecasting with high-dimensional models: The case of Brazil. International Journal of Forecasting 33, 679–693.

Gu, S., Kelly, B., Xiu, D., 2020. Empirical Asset Pricing via Machine Learning. The Review of Financial Studies 33, 2223–2273.

Hansen, P.R., 2005. A test for superior predictive ability. Journal of Business & Economic Statistics 23, 365–380.

Hansen, P.R., Lunde, A., Nason, J.M., 2011. The model confidence set. Econometrica 79, 453–497.

Hastie, T., Tibshirani, R., Friedman, J.H., 2009. The elements of statistical learning: data mining, inference and prediction. 2 ed., Springer, New York.

Konzen, E., Ziegelmann, F.A., 2016. LASSO-Type Penalties for Covariate Selection and Forecasting in Time Series. Journal of Forecasting 35, 592–612.

McCracken, M.W., Ng, S., 2016. Fred-md: A monthly database for macroeconomic research. Journal of Business & Economic Statistics 34, 574–589.

Medeiros, M.C., Mendes, E.F., 2016. l1-regularization of high-dimensional time-series models with non-gaussian and heteroskedastic errors. Journal of Econometrics 191, 255 – 271.

Medeiros, M.C., Vasconcelos, G.F.R., 2016. Forecasting macroeconomic variables in data-rich environments. Economics Letters 138, 50 – 52.

Medeiros, M.C., Vasconcelos, G.F.R., Freitas, E., 2016. Forecasting Brazilian Inflation with High-Dimensional Models. Brazilian Review of Econometrics 36, 223 – 254.

Medeiros, M.C., Vasconcelos, G.F.R., Veiga, , Zilberman, E., 2019. Forecasting inflation in a data-rich environment: The benefits of machine learning methods. Journal of Business & Economic Statistics , 1–22.

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

Quaedvlieg, R., 2019. Multi-horizon forecast comparison. Journal of Business & Economic Statistics 0, 1–14.

Tibshirani, R., 1996. Regression Shrinkage and Selection via the Lasso. Journal of the Royal Statistical Society. Series B (Methodological) 58, 267–288.

Zou, H., 2006. The adaptive lasso and its oracle properties. Journal of the American Statistical Association 101, 1418–1429.

Zou, H., Zhang, H.H., 2009. On the adaptive elastic-net with a diverging number of parameters. The Annals of Statistics 37, 1733–1751.