How is Machine Learning Useful for Macroeconomic Forecasting?*

Philippe Goulet Coulombe[1†] Maxime Leroux[2] Dalibor Stevanovic[2‡]
Stéphane Surprenant[2]

[1]University of Pennsylvania
[2]Université du Québec à Montréal

This version: February 28, 2019

Abstract

We move beyond *Is Machine Learning Useful for Macroeconomic Forecasting?* by adding the *how*. The current forecasting literature has focused on matching specific variables and horizons with a particularly successful algorithm. To the contrary, we study a wide range of horizons and variables and learn about the usefulness of the underlying features driving ML gains over standard macroeconometric methods. We distinguish 4 so-called features (nonlinearities, regularization, cross-validation and alternative loss function) and study their behavior in both the data-rich and data-poor environments. To do so, we carefully design a series of experiments that easily allow to identify the treatment effects of interest. The simple evaluation framework is a fixed-effects regression that can be understood as an extension of the Diebold and Mariano (1995) test. The regression setup prompt us to use a novel visualization technique for forecasting results that conveys all the relevant information in a digestible format. We conclude that **(i)** more data and non-linearities are very useful for real variables at long horizons, **(ii)** the standard factor model remains the best regularization, **(iii)** cross-validations are not all made equal (but K-fold is as good as BIC) and **(iv)** one should stick with the standard L_2 loss.

Keywords: Machine Learning, Big Data, Forecasting.

*The third author acknowledges financial support from the Fonds de recherche sur la société et la culture (Québec) and the Social Sciences and Humanities Research Council.

†Corresponding Author: gouletc@sas.upenn.edu. Department of Economics, UPenn.

‡Corresponding Author: dstevanovic.econ@gmail.com. Département des sciences économiques, UQAM.

1 Introduction

The intersection of Machine Learning (ML) with econometrics has become an important research landscape in economics. ML has gained prominence due to the availability of large data sets, especially in microeconomic applications, Athey (2018). However, as pointed by Mullainathan and Spiess (2017), applying ML to economics requires finding relevant tasks. Despite the growing interest in ML, little progress has been made in understanding the properties of ML models and procedures when they are applied to predict macroeconomic outcomes.[1] Nevertheless, that very understanding is an interesting econometric research endeavor *per se*. It is more appealing to applied econometricians to upgrade a standard framework with a subset of specific insights rather than to drop everything altogether for an off-the-shelf ML model.

A growing number studies have applied recent machine learning models in macroeconomic forecasting.[2] However, those studies share many shortcomings. Some focus on one particular ML model and on a limited subset of forecasting horizons. Other evaluate the performance for only one or two dependent variables and for a limited time span. The papers on comparison of ML methods are not very extensive and do only a forecasting horse race without providing insights on why some models perform better.[3] As a result, little progress has been made to understand the properties of ML methods when applied to macroeconomic forecasting. That is, so to say, the black box remains closed. The objective of this paper is to bring an understanding of each method properties that goes beyond the coronation of a single winner for a specific forecasting target. We believe this will be much more useful for subsequent model building in macroeconometrics.

Precisely, we aim to answer the following question. What are the key features of ML modeling that improve the macroeconomic prediction? In particular, no clear attempt has been made at understanding why one algorithm might work and another one not. We address this question by designing an *experiment* to identify important characteristics of machine learning and big data techniques. The exercise consists of an extensive pseudo-out-of-sample forecasting horse race between many models that differ with respect to the four

[1] Only the unsupervised statistical learning techniques such as principal component and factor analysis have been extensively used and examined since the pioneer work of Stock and Watson (2002a). Kotchoni et al. (2017) do a substantial comparison of more than 30 various forecasting models, including those based on factor analysis, regularized regressions and model averaging. Giannone et al. (2017) study the relevance of sparse modelling (Lasso regression) in various economic prediction problems.

[2] Nakamura (2005) is an early attempt to apply neural networks to improve on prediction of inflation, while Smalter and Cook (2017) use deep learning to forecast the unemployment. Diebold and Shin (2018) propose a Lasso-based forecasts combination technique. Sermpinis et al. (2014) use support vector regressions to forecast inflation and unemployment. Döpke et al. (2015) and Ng (2014) aim to predict recessions with random forests and boosting techniques. Few papers contribute by comparing some of the ML techniques in forecasting horse races, see Ahmed et al. (2010), Ulke et al. (2016) and Chen et al. (2019).

[3] An exception is Smeekes and Wijler (2018) who compare performance of sparse and dense models in presence of non-stationary data.

main features: nonlinearity, regularization, hyperparameter selection and loss function. To control for big data aspect, we consider data-poor and data-rich models, and administer those *patients* one particular ML *treatment* or combinations of them. Monthly forecast errors are constructed for five important macroeconomic variables, five forecasting horizons and for almost 40 years. Then, we provide a straightforward framework to back out which of them are actual game-changers for macroeconomic forecasting.

The main results can be summarized as follows. First, non-linearities either improve drastically or decrease substantially the forecasting accuracy. The benefits are significant for industrial production, unemployment rate and term spread, and increase with horizons, especially if combined with factor models. Nonlinearity is harmful in case of inflation and housing starts. Second, in big data framework, alternative regularization methods (Lasso, Ridge, Elastic-net) do not improve over the factor model, suggesting that the factor representation of the macroeconomy is quite accurate as a mean of dimensionality reduction.

Third, the hyperparameter selection by K-fold cross-validation does better on average that any other criterion, strictly followed by the standard BIC. This suggests that ignoring information criteria when opting for more complicated ML models is not harmful. This is also quite convenient: K-fold is the built-in CV option in most standard ML packages. Fourth, replacing the standard in-sample quadratic loss function by the $\bar{\epsilon}$-insensitive loss function in Support Vector Regressions is not useful, except in very rare cases. Fifth, the marginal effects of big data are positive and significant for real activity series and term spread, and improve with horizons.

The state of economy is another important ingredient as it interacts with few features above. Improvements over standard autoregressions are usually magnified if the target falls into an NBER recession period, and the access to data-rich predictor set is particularly helpful, even for inflation. Moreover, the pseudo-out-of-sample cross-validation failure is mainly attributable to its underperformance during recessions.

These results give a clear recommendation for practitioners. For most variables and horizons, start by reducing the dimensionality with principal components and then augment the standard diffusion indices model by a ML non-linear function approximator of choice. Of course, that recommendation is conditional on being able to keep overfitting in check. To that end, if cross-validation must be applied to hyperparameter selection, the best practice is the standard K-fold.

In the remainder of this papers we first present the general prediction problem with machine learning and big data in Section 2. The Section 3 describes the four important features of machine learning methods. The Section 4 presents the empirical setup, the Section 5 discuss the main results and Section 6 concludes. Appendices A, B, C, D and E contain, respectively: tables with overall performance; robustness of treatment analysis; additional figures; description of cross-validation techniques and technical details on forecasting models.

2 Making predictions with machine learning and big data

To fix ideas, consider the following general prediction setup from Hastie et al. (2017)

$$\min_{g \in \mathcal{G}} \{\hat{L}(y_{t+h}, g(Z_t)) + pen(g; \tau)\}, \quad t = 1, \ldots, T \tag{1}$$

where y_{t+h} is the variable to be predicted h periods ahead (target) and Z_t is the N_Z-dimensional vector of predictors made of H_t, the set of all the inputs available at time t. Note that the time subscripts are not necessary so this formulation can represent any prediction problem. This setup has four main features:

1. $\mathcal{G}$ is the space of possible functions g that combine the data to form the prediction. In particular, the interest is how much non-linearities can we allow for? A function g can be parametric or nonparametric.

2. $pen()$ is the penalty on the function g. This is quite general and can accommodate, among others, the Ridge penalty of the standard by-block lag length selection by information criteria.

3. τ is the set of hyperparameters of the penalty above. This could be λ in a LASSO regression or the number of lags to be included in an AR model.

4. $\hat{L}$ the loss function that defines the optimal forecast. Some models, like the SVR, feature an in-sample loss function different from the standard l_2 norm.

Most of (Supervised) machine learning consists of a combination of those ingredients. This formulation may appear too abstract, but the simple predictive regression model can be obtained as a special case. Suppose a quadratic loss function $\hat{L}$, implying that the optimal forecast is the conditional expectation $\mathrm{E}(y_{t+h}|Z_t)$. Let the function g be parametric and linear: $y_{t+h} = Z_t\beta + error$. If the number of coefficients in β is not too big, the penalty is usually ignored and (1) reduces to the textbook predictive regression inducing $\mathrm{E}(y_{t+h}|Z_t) = Z_t\beta$ as the optimal prediction.

2.1 Predictive Modeling

We consider the *direct* predictive modeling in which the target is projected on the information set, and the forecast is made directly using the most recent observables. This is opposed to *iterative* approach where the model recursion is used to simulate the future path of the variable.[4] Also, the direct approach is the only one that is feasible for all ML models.

[4] Marcellino et al. (2006) conclude that the direct approach provides slightly better results but does not dominate uniformly across time and series.

We now define the forecast objective. Let Y_t denote a variable of interest. If $\ln Y_t$ is a stationary, we will consider forecasting its average over the period $[t+1, t+h]$ given by:

$$y_{t+h}^{(h)} = (1/h) \sum_{k=1}^{h} y_{t+k}, \tag{2}$$

where $y_t \equiv \ln Y_t$ if Y_t is strictly positive. Most of the time, we are confronted with I(1) series in macroeconomics. For such series, our goal will be to forecast the average annualized growth rate over the period $[t+1, t+h]$, as in Stock and Watson (2002b) and McCracken and Ng (2016). We shall therefore define $y_{t+h}^{(h)}$ as:

$$y_{t+h}^{(h)} = (1/h)\ln(Y_{t+h}/Y_t). \tag{3}$$

In cases where $\ln Y_t$ is better described by an I(2) process, we define $y_{t+h}^{(h)}$ as:

$$y_{t+h}^{(h)} = (1/h)\ln(Y_{t+h}/Y_{t+h-1}) - \ln(Y_t/Y_{t-1}). \tag{4}$$

In order to avoid a cumbersome notation, we use y_{t+h} instead of $y_{t+h}^{(h)}$ in what follows, but the target is always the average (growth) over the period $[t+1, t+h]$.

2.2 *Data-poor* versus *data-rich* environments

Large time series panels are now widely constructed and used for macroeconomic analysis. The most popular is FRED-MD monthly panel of US variables constructed by McCracken and Ng (2016). Fortin-Gagnon et al. (2018) have recently proposed similar data for Canada, while Boh et al. (2017) has constructed a large macro panel for Euro zone. Unfortunately, the performance of standard econometric models tends to deteriorate as the dimensionality of the data increases, which is the well-known curse of dimensionality. Stock and Watson (2002a) first proposed to solve the problem by replacing the large-dimensional information set by its principal components. See Kotchoni et al. (2017) for the review of many dimension-reduction, regularization and model averaging predictive techniques. Another way to approach the dimensionality problem is to use Bayesian methods (Kilian and Lütkepohl (2017)). All the shrinkage schemes presented later in this paper can be seen as a specific prior. Indeed, some of our Ridge regressions will look very much like a direct version of a Bayesian VAR with a Litterman (1979) prior.[5]

Traditionally, as all these series may not be relevant for a given forecasting exercise, one will have to preselect the most important candidate predictors according to economic the-

[5] Giannone et al. (2015) have shown that a more elaborate hierarchical prior can lead the BVAR to perform as well as a factor model

ories, the relevant empirical literature and own heuristic arguments. Even though the machine learning models do not require big data, they are useful to discard irrelevant predictors based on statistical learning, but also to digest a large amount of information to improve the prediction. Therefore, in addition to treatment effects in terms of characteristics of forecasting models, we will also compare the predictive performance of small versus large data sets. The data-poor, defined as H_t^-, will only contain a finite number of lagged values of the dependent variable, while the data-rich panel, defined as H_t^+ will also include a large number of exogenous predictors. Formally, we have

$$H_t^- \equiv \{y_{t-j}\}_{j=0}^{p_y} \quad and \quad H_t^+ \equiv \left[\{y_{t-j}\}_{j=0}^{p_y}, \{X_{t-j}\}_{j=0}^{p_f}\right]. \tag{5}$$

The analysis we propose can thus be summarized in the following way. We will consider two standard models for forecasting.

1. The H_t^- model is the *autoregressive direct* (AR) model, which is specified as:

$$y_{t+h} = c + \rho(L)y_t + e_{t+h}, \quad t = 1, \dots, T, \tag{6}$$

   where $h \geq 1$ is the forecasting horizon. The only hyperparameter in this model is p_y, the order of the lag polynomial $\rho(L)$.

2. The H_t^+ workhorse model is the autoregression augmented with diffusion indices (ARDI) from Stock and Watson (2002b):

$$\begin{aligned} y_{t+h} &= c + \rho(L)y_t + \beta(L)F_t + e_{t+h}, \quad t = 1, \dots, T & (7) \\ X_t &= \Lambda F_t + u_t & (8) \end{aligned}$$

   where F_t are K *consecutive static* factors, and $\rho(L)$ and $\beta(L)$ are lag polynomials of orders p_y and p_f respectively. The feasible procedure requires an estimate of F_t that is usually obtained by principal components analysis (PCA).

Then, we will take these models as two different types of "patients" and will administer them one particular ML treatment or combinations of them. That is, we will upgrade (hopefully) these models with one or many features of ML and evaluate the gains/losses in both environments.

Beyond the fact that the ARDI is a very popular macro forecasting model, there are additional good reasons to consider it as one benchmark for our investigation. While we discuss four features of ML in this paper, it is obvious that the big two are shrinkage (or dimension reduction) and non-linearities. Both goes in completely different directions. The first deals with data sets that have a low observations to regressors ratio while the latter is especially useful when that same ratio is high. Most nonlinearities are created with basis expansions

which are just artificially generated additional regressors made of the original data. That is quite useful in a data-poor environments but is impracticable in data-rich environments where the goal is exactly the opposite, that is, to decrease the effective number of regressors.

Hence, the only way to afford non-linear models with wide macro datasets is to compress the data beforehand and then use the compressed predictors as inputs. Each compression scheme has an intuitive economic justification of its own. Choosing only a handful of series can be justified by some DSGE model that has a reduced-form VAR representation. Compressing the data according to a factor model adheres to the view that are only a few key drivers of the macroeconomy and those are not observed. We choose the latter option as its forecasting record is stellar. Hence, our non-linear models implicitly postulate that a sparse set of latent variables impact the target variable in a flexible way. To take PCs of data to feed them afterward in a NL model is also a standard thing to do from a ML perspective.

2.3 Evaluation

The objective of this paper is to disentangle important characteristics of the ML prediction algorithms when forecasting macroeconomic variables. To do so, we design an *experiment* that consists of a pseudo-out-of-sample forecasting horse race between many models that differ with respect to the four main features above: nonlinearity, regularization, hyperparameter selection and loss function. To create variation around those *treatments*, we will generate forecasts errors from different models associated to each feature.

To test this paper's hypothesis, suppose the following model for forecasting errors

$$e^2_{t,h,v,m} = \alpha_m + \psi_{t,v,h} + v_{t,h,v,m} \tag{9a}$$

$$\alpha_m = \alpha_F + \eta_m \tag{9b}$$

where $e^2_{t,h,v,m}$ are squared prediction errors of model m for variable v and horizon h at time t. $\psi_{t,v,h}$ is a fixed effect term that demean the dependent variable by "forecasting target", that is a combination of t, v and h. α_F is a vector of $\alpha_{\mathcal{G}}$, $\alpha_{pen()}$, α_τ and $\alpha_{\hat{L}}$ terms associated to each feature. We re-arrange equation (9) to obtain

$$e^2_{t,h,v,m} = \alpha_F + \psi_{t,v,h} + u_{t,h,v,m}. \tag{10}$$

H_0 is now $\alpha_f = 0 \quad \forall f \in F = [\mathcal{G},\ pen(),\ \tau,\ \hat{L}]$. In other words, the null is that there is no predictive accuracy gain with respect to a base model that does not have this particular feature.[6] Very interestingly, by interacting α_F with other fixed effects or even variables, we

[6]Note that if we are considering two models that differ in one feature and run this regression for a specific (h, v) pair, the t-test on the sole coefficients amounts to a Diebold and Mariano (1995) test – conditional on having the proper standard errors.

can test many hypothesis about the heterogeneity of the "ML treatment effect". Finally, to get interpretable coefficients, we use a linear combination of $e^2_{t,h,v,m}$ by (h,v) pair that makes the final regressand $(h,v,m)-$specific average a pseudo-out-of-sample R^2.[7] Hence, we define $R^2_{t,h,v,m} \equiv 1 - \frac{e^2_{t,h,v,m}}{\frac{1}{T}\sum_{t=1}^{T}(y_{v,t+h} - \bar{y}_{v,h})^2}$ and run

$$R^2_{t,h,v,m} = \dot{\alpha}_F + \dot{\psi}_{t,v,h} + \dot{u}_{t,h,v,m}. \tag{11}$$

On top of providing coefficients $\dot{\alpha}_F$ interpretable as marginal improvements in OOS-R^2's, the approach has the advantage of standardizing *ex-ante* the regressand and thus removing an obvious source of (v,h)-driven heteroscedasticity. Also, a positive α_F now means (more intuitively) an improvement rather than the other way around.

While the generality of (10) and (11) is appealing, when investigating the heterogeneity of specific partial effects, it will be much more convenient to run specific regressions for the multiple hypothesis we wish to test. That is, to evaluate a feature f, we run

$$\forall m \in \mathcal{M}_f: \quad R^2_{t,h,v,m} = \dot{\alpha}_f + \dot{\phi}_{t,v,h} + \dot{u}_{t,h,v,m} \tag{12}$$

where $\mathcal{M}_f$ is defined as the set of models that differs only by the feature under study f.

3 Four features of ML

In this section we detail the forecasting approaches to create variations for each characteristic of machine learning prediction problem defined in (1).

3.1 Feature 1: selecting the function g

Certainly an important feature of machine learning is the whole available apparatus of non-linear function estimators. We choose to focus on applying the Kernel trick and Random Forests to our two baseline models to see if the non-linearities they generate will lead to significant improvements.

3.1.1 Kernel Ridge Regression

Since all models considered in this paper can easily be written in the dual form, we can use the kernel trick (KT) in both data-rich and data-poor environments. It is worth noting that Kernel Ridge Regression (KRR) has several implementation advantages. First, it has a closed-form solution that rules out convergence problems associated with models trained

[7] Precisely: $\frac{1}{T}\sum_{t=1}^{T} 1 - \frac{e^2_{t,h,v,m}}{\frac{1}{T}\sum_{t=1}^{T}(y_{v,t+h} - \bar{y}_{v,h})^2} = R^2_{h,v,m}$

with gradient descent. Second, it is fast to implement given that it implies inverting a TxT matrix at each step (given tuning parameters) and T is never quite large in macro. Since we are doing an extensive POOS exercise for a long period of time, these qualities are very helpful.

We will first review briefly how the KT is implemented in our two benchmark models. Suppose we have a Ridge regression direct forecast with generic regressors Z_t

$$\min_{\beta} \sum_{t=1}^{T} (y_{t+h} - Z_t\beta)^2 + \lambda \sum_{k=1}^{K} \beta_k^2.$$

The solution to that problem is $\hat{\beta} = (Z'Z + \lambda I_k)^{-1} Z'y$. By the representer theorem of Smola and Schölkopf (2004), β can also be obtained by solving the dual of the convex optimization problem above. The dual solution for β is $\hat{\beta} = Z'(ZZ' + \lambda I_T)^{-1} y$. This equivalence allows to rewrite the conditional expectation in the following way:

$$\hat{E}(y_{t+h}|Z_t) = Z_t\hat{\beta} = \sum_{i=1}^{t} \hat{\alpha}_i \langle Z_i, Z_t \rangle$$

where $\hat{\alpha} = (ZZ' + \lambda I_T)^{-1} y$ is the solution to the dual Ridge Regression problem. For now, this is just another way of getting exactly the same fitted values.

Let's now introduce a general non-linear model. Suppose we approximate it with basis functions $\phi()$

$$y_{t+h} = g(Z_t) + \varepsilon_{t+h} = \phi(Z_t)'\gamma + \varepsilon_{t+h}.$$

The so-called Kernel trick is the fact that there exist a reproducing kernel $K()$ such that

$$\hat{E}(y_{t+h}|Z_t) = \sum_{i=1}^{t} \hat{\alpha}_i \langle \phi(Z_i), \phi(Z_t) \rangle = \sum_{i=1}^{t} \hat{\alpha}_i K(Z_i, Z_t).$$

This means we do not need to specify the numerous basis functions, a well-chosen Kernel implicitly replicates them. For the record, this paper will be using the standard radial basis function kernel

$$K_\sigma(\mathbf{x}, \mathbf{x}') = \exp\left(-\frac{\|\mathbf{x} - \mathbf{x}'\|^2}{2\sigma^2}\right)$$

where σ is a tuning parameter to be chosen by cross-validation.

Hence, by using the corresponding Z_t, we can easily make our data-rich or data-poor model non-linear. For instance, in the case of the factor model, we can apply it to the regres-

sion equation to implicitly estimate

$$y_{t+h} = c + g(Z_t) + \varepsilon_{t+h}, \tag{13}$$
$$Z_t = \left[\{y_{t-0}\}_{j=0}^{p_y}, \{F_{t-j}\}_{j=0}^{p_f}\right], \tag{14}$$
$$X_t = \Lambda F_t + u_t. \tag{15}$$

In terms of implementation, this means extracting factor via PCA and then get

$$\hat{E}(y_{t+h}|Z_t) = K_\sigma(Z_t, Z)(K_\sigma(Z, Z) + \lambda I_T)^{-1} y. \tag{16}$$

The final set of tuning parameters for such a model is $\tau = \{\lambda, \sigma, p_y, p_f, n_f\}$.

3.1.2 Random forests

Another way to introduce non-linearity in the estimation of the predictive equation is to use regression trees instead of OLS. Recall the ARDI model:

$$y_{t+h} = c + \rho(L)y_t + \beta(L)F_t + \varepsilon_{t+h},$$
$$X_t = \Lambda F_t + u_t,$$

where y_t and F_t, and their lags, constitute the informational set Z_t. This form is clearly linear but one could tweak the model by replacing it by a regression tree. The idea is to split sequentially the space of Z_t into several regions and model the response by the mean of y_{t+h} in each region. The process continues according to some stopping rule. As a result, the tree regression forecast has the following form:

$$\hat{f}(Z) = \sum_{m=1}^{M} c_m \mathbf{I}_{(Z \in R_m)}, \tag{17}$$

where M is the number of terminal nodes, c_m are node means and $R_1, ..., R_M$ represent a partition of feature space. In the diffusion indices setup, the regression tree would estimate a non-linear relationship linking factors and their lags to y_{t+h}. Once the tree structure is known, this procedure can be related to a linear regression with dummy variables and their interactions.

Instead of just using one single tree, which is known to be subject to overfitting, we use Random forests which consist of a certain number of trees using a subsample of observations but also a random subset of regressors for each tree.[8] The hyperparameter to be cross-

[8]Only using a subsample of observations would be a procedure called Bagging. Also selecting randomly regressors has the effect of decorrelating the trees and hence improving the out-of-sample forecasting accuracy.

validated is the number of trees. The forecasts of the estimated regression trees are then averaged together to make one single prediction of the targeted variable.

3.2 Feature 2: selecting the regularization

In this section we will only consider models where dimension reduction is needed, which are the models with H_t^+ – that is, more information than just the past values of y_t. The traditional shrinkage method used in macroeconomic forecasting is the ARDI model that consists of extracting principal components of X_t and to use them as data in an ARDL model. Obviously, this is only one out of many ways to compress the information contained in X_t to run a well-behaved regression of y_{t+h} on it. De Mol et al. (2008) compares Lasso, Ridge and ARDI and finds that forecasts are very much alike. This section can be seen as extending the scope of their study by consider a wider range of models in a updated forecasting experiment that includes the Great Recession (theirs end in 2003).

In order to create identifying variations for $pen()$ treatment, we need to generate multiple different shrinkage schemes. Some will also blend in selection, some will not. The alternative shrinkage methods consider in this section will all be specific special cases of a standard Elastic Net (EN) problem:

$$\min_{\beta} \sum_{t=1}^{T} (y_{t+h} - Z_t\beta)^2 + \lambda \sum_{k=1}^{K} \left(\alpha|\beta_k| + (1-\alpha)\beta_k^2 \right) \tag{18}$$

where $Z_t = B(H_t)$ is some transformation of the original predictive set X_t. $\alpha \in [0,1]$ can either be fixed or found via cross-validation (CV) while $\lambda > 0$ always needs to be obtained by CV. By using different B operators, we can generate shrinkage schemes. Also, by setting α to either 1 or 0 we generate LASSO and Ridge Regression respectively. Choosing α by CV also generate an intermediary regularization scheme of its own. All these possibilities are reasonable alternatives to the traditional factor hard-thresholding procedure that is ARDI.

Each type of shrinkage in this section will be defined by the tuple $S = \{\alpha, B()\}$. To begin with the most straightforward dimension, for a given B, we will evaluate the results for $\alpha \in \{0, \hat{\alpha}_{CV}, 1\}$. For instance, if B is the identity mapping, we get in turns the LASSO, Elastic Net and Ridge shrinkage.

Let us now turn to detail different resulting $pen()$ when we vary $B()$ for a fixed α. Three alternatives will be considered.

1. **(Fat Regression)**: First, we will consider the case $B_1() = I()$ as mentioned above. That is, we use the entirety of the untransformed high-dimensional data set. The results of Giannone et al. (2017) point in the direction that specifications with a higher α should do better, that is, sparse models do worse than models where every regressor is kept but shrunk to zero.

2. **(Big ARDI)** Second, we will consider the case where $B_2()$ corresponds to first rotating $X_t \in \mathbb{R}^N$ so that we get N uncorrelated F_t. Note here that contrary to the standard ARDI model, we do not throw out factors according to some information criteria or a scree test: we keep them all. Hence, F_t has exactly the same span as X_t. If we were to run OLS (without any form of shrinkage), using $\phi(L)F_t$ versus $\psi(L)X_t$ would not make any difference in term of fitted values. However, when shrinkage comes in, a similar $pen()$ applied to a rotated regressor space implicitly generates a new penalty. Comparing LASSO and Ridge in this setup will allow to verify whether sparsity emerges in a rotated space. That is, this could be interpreted as looking whether the 'economy' has a sparse DGP, but in a different regressor space that the original one. This correspond to the dense view of the economy, which is that observables are only driven by a few key fundamental economic shocks.

3. **(Principal Component Regression)** A third possibility is to rotate H_t^+ rather than X_t and still keep all the factors. H_t^+ includes all the relevant pre-selected lags. If we were to just drop the F_t using some hard-thresholding rule, this would correspond to Principal Component Regression (PCR). Note that $B_3() = B_2()$ only when no lags are included. Here, the F_t have a different interpretation since they are extracted from multiple t's data whereas the standard factor model used in econometrics typically extract principal components out of X_t in a completely contemporaneous fashion.

To wrap up, this means the tuple S has a total of 9 elements. Since we will be considering both POOS-CV and K-fold CV for each of these models, this leads to a total of 18 models.

Finally, to see clearly through all of this, we can describe where the benchmark ARDI model stands in this setup. Since it uses a hard thresholding rule that is based on the eigenvalues ordering, it cannot be a special case of the Elastic Net problem. While it is clearly using B_2, we would need to set $\lambda = 0$ and select F_t *a priori* with a hard-thresholding rule. The closest approximation in this EN setup would be to set $\alpha = 1$ and fix the value of λ to match the number of consecutive factors selected by an information criteria directly in the predictive regression (20) or using an analytically calculated value based on Bai and Ng (2002). However, this would still not impose the ordering of eigenvalues: the Lasso could happen to select a F_t associated to a small eigenvalue and yet drop one F_t associated with a bigger one.

3.3 Feature 3: Choosing hyperparameters τ

The conventional wisdom in macroeconomic forecasting is to either use AIC or BIC and compare results. It is well known that BIC selects more parsimonious models than AIC. A relatively new kid on the block is cross-validation, which is widely used in the field of

machine learning. The prime reason for the popularity of CV is that it can be applied to any model, which includes those for which the derivation of an information criterion is impossible. An another appeal of the method is its logical simplicity. However, as AIC and BIC, it relies on particular assumptions in order to be well-behaved.

It is not quite obvious that CV should work better only because it is "out of sample" while AIC and BIC are "in sample". All model selection methods are actually approximations to the OOS prediction error that relies on different assumptions that are sometime motivated by different theoretical goals. Also, it is well known that asymptotically, these methods have quite similar behavior.[9] For instance, one can show that Leave-one-out CV (a special case of k-fold) is asymptotically equivalent to Takeuchi Information criterion (TIC), Claeskens and Hjort (2008). AIC is a special case of TIC where we need to assume in addition that all models being considered are at least correctly specified. Thus, under the latter assumption, Leave-one-out CV is asymptotically equivalent to AIC. Hence, it is impossible *a priori* to think of one model selection technique being the most appropriate for macroeconomic forecasting.

For samples of small to medium size encountered in macro, the question of which one is optimal in the forecasting sense is inevitably an empirical one. For instance, Granger and Jeon (2004) compared AIC and BIC in a generic forecasting exercise. In this paper, we will compare AIC, BIC and two types of CV for our two baseline models. The two types of CV are relatively standard. We will first use POOS CV and then k-fold CV. The first one will always behave correctly in the context of time series data, but may be quite inefficient by only using the end of the training set. The latter is known to be valid only if residuals autocorrelation is absent from the models as shown in Bergmeir et al. (2018). If it were not to be the case, then we should expect k-fold to under-perform. The specific details of the implementation of both CVs is discussed in appendix D.

The contributions of this section are twofold. First, it will shed light on which model selection method is most appropriate for typical macroeconomic data and models. Second, we will explore how much of the gains/losses of using ML can be attributed to widespread use of CV. Since most non-linear ML models cannot be easily tuned by anything else than CV, it is hard for the researcher to disentangle between gains coming from the ML method itself or just the way it is tuned.[10] Hence, it is worth asking the question whether some gains from ML are simply coming from selecting hyperparameters in a different fashion using a method which assumptions are more fit with the data at hand. To investigate that, a natural first step is to look at our benchmark macro models, AR and ARDI, and see if using CV to

[9] Hansen and Timmermann (2015) show equivalence between test statistics for OOS forecasting performance and in-sample Wald statistics.

[10] Zou et al. (2007) show that the number of remaining parameters in the LASSO is an unbiased estimator of the degrees of freedom and derive LASSO-BIC and LASSO-AIC criteria. Considering these as well would provide additional evidence on the empirical debate of CV vs IC.

select hyperparameters gives different selected models and forecasting performances.

3.4 Feature 4: Selecting the loss function

With the exception of the support vector regression (SVR), all of our estimators for the predictive function $g \in \mathcal{G}$ use a quadratic loss function. The objective of this section is to evaluate the importance of a $\bar{\epsilon}$-insensitive loss function for macroeconomic predictions. However, this is not so easily done since the SVR is different from an ARDI model in multiple aspects. Namely, it

- uses a different in-sample loss function;
- (usually) uses a kernel trick in order to obtain non-linearities and
- has different tuning parameters.

Hence, we must provide a strategy to isolate the effect of the first item. That is, if the standard RBF kernel SVR works well, we want to know whether is the effect of the kernel or that of the loss-function. First, while the SVR is almost always used in combination with a kernel trick similar to what described in the previous sections, we will also obtain results for a linear SVR. That isolates the effect of the kernel. Second, we considered the Kernel Ridge Regression earlier. The latter only differs from the Kernel-SVR by the use of different in-sample loss functions. That identifies the effect of the loss function. To sum up, in order to isolate the "treatment effect" of a different in-sample loss function, we will obtain forecasts from

1. the linear SVR with H_t^-;
2. the linear SVR with H_t^+;
3. the RBF Kernel SVR with H_t^- and
4. the RBF Kernel SVR with H_t^+.

What follows is a bird's eye overview of the underlying mechanics of the SVR. As it was the case for the Kernel Ridge regression, the SVR estimator approximates the function $g \in G$ with basis functions. That is, the DGP is still $y_{t+h} = \alpha + \gamma' \phi(Z_t) + \epsilon_{t+h}$. We opted to use the ν-SVR variant which implicitly defines the size $2\bar{\epsilon}$ of the insensitivity tube of the loss

function. The hyperparameter ν is selected by cross validation. This estimator is defined by:

$$\min_{\gamma} \frac{1}{2}\gamma'\gamma + C\left[\sum_{j=1}^{T}(\xi_j + \xi_j^*) + T\nu\bar{\epsilon}\right]$$
$$s.t. \begin{cases} y_{t+h} - \gamma'\phi(Z_t) - c \leq \bar{\epsilon} + \xi_t \\ \gamma'\phi(Z_t) + c - y_{t+h} \leq \bar{\epsilon} + \xi_t^* \\ \xi_t, \xi_t^* \geq 0. \end{cases}$$

Where ξ_t, ξ_t^* are slack variables, $\phi()$ is the basis function of the feature space implicitly defined by the kernel used, T is the size of the sample used for estimation and C is an hyperparameter. In case of the RBF Kernel, an additional hyperparameter, σ, has to be cross-validated. Associating Lagrange multipliers λ_j, λ_j^* to the first two types of constraints, we can derive the dual problem (Smola and Schölkopf (2004)) out of which we would find the optimal weights $\gamma = \sum_{j=1}^{T}(\lambda_j - \lambda_j^*)\phi(Z_j)$ and the forecasted values

$$\hat{E}(y_{t+h}|Z_t) = \hat{c} + \sum_{j=1}^{T}(\lambda_j - \lambda_j^*)\phi(Z_j)\phi(Z_j) = \hat{c} + \sum_{j=1}^{T}(\lambda_j - \lambda_j^*)K(Z_j, Z_t). \tag{19}$$

Let us now turn to the resulting loss function of such a problem. Along the in-sample forecasted values, there is an upper bound $\hat{E}(y_{t+h}|Z_t) + \bar{\epsilon}$ and lower bound $\hat{E}(y_{t+h}|Z_t) - \bar{\epsilon}$. Inside of these bounds, the loss function is null. Let $e_{t+h} := \hat{E}(y_{t+h}|Z_t) - y_t$ be the forecasting error and define a loss function using a penalty function $P_{\bar{\epsilon}}$ as $\hat{L}_{\bar{\epsilon}}(\{e_{t+h}\}_{t=1}^{T}) := \frac{1}{T}\sum_{t=1}^{T} P_{\bar{\epsilon}}(e_{t+h})$. For the ν-SVR, the penalty is given by:

$$P_{\bar{\epsilon}}(\epsilon_{t+h|t}) := \begin{cases} 0 & if \quad |e_{t+h}| \leq \bar{\epsilon} \\ |e_{t+h}| - \bar{\epsilon} & otherwise \end{cases}.$$

For other estimators, the penalty function is quadratic $P(e_{t+h}) := e_{t+h}^2$. Hence, the rate of the penalty increases with the size of the forecasting error, whereas it is constant and only applies to excess errors in the case of the ν-SVR. Note that this insensitivity has a nontrivial consequence for the forecasting values. The Karush-Kuhn-Tucker conditions imply that only support vectors, i.e. points lying inside the insensitivity tube, will have nonzero Lagrange multipliers and contribute to the weight vector. In other words, all points whose errors are too big are effectively ignored at the optimum. Smola and Schölkopf (2004) call this the *sparsity* of the SVR. The empirical usefulness of this property for macro data is a question we will be answering in the coming sections.

To sum up, the Table 1 shows a list of all forecasting models and highlights their relation-

ship with each of four features discussed above. The computational details on every model in this list are available in Appendix E.

Table 1: List of all forecasting models

Models	Feature 1: selecting the function g	Feature 2: selecting the regularization	Feature 3: optimizing hyperparameters τ	Feature 4: selecting the loss function
Data-poor models				
AR,BIC	Linear		BIC	Quadratic
AR,AIC	Linear		AIC	Quadratic
AR,POOS-CV	Linear		POOS CV	Quadratic
AR,K-fold	Linear		K-fold CV	Quadratic
RRAR,POOS-CV	Linear	Ridge	POOS CV	Quadratic
RRAR,K-fold	Lineal	Ridge	K-fold CV	Quadratic
RFAR,POOS-CV	Nonlinear		POOS CV	Quadratic
RFAR,K-fold	Nonlinear		K-fold CV	Quadratic
KRRAR,POOS-CV	Nonlinear	Ridge	POOS CV	Quadratic
KRRAR,K-fold	Nonlinear	Ridge	K-fold CV	Quadratic
SVR-AR,Lin,POOS-CV	Linear		POOS CV	$\bar{\epsilon}$-insensitive
SVR-AR,Lin,K-fold	Linear		K-fold CV	$\bar{\epsilon}$-insensitive
SVR-AR,RBF,POOS-CV	Nonlinear		POOS CV	$\bar{\epsilon}$-insensitive
SVR-AR,RBF,K-fold	Nonlinear		K-fold CV	$\bar{\epsilon}$-insensitive
Data-rich models				
ARDI,BIC	Linear	PCA	BIC	Quadratic
ARDI,AIC	Linear	PCA	AIC	Quadratic
ARDI,POOS-CV	Linear	PCA	POOS CV	Quadratic
ARDI,K-fold	Linear	PCA	K-fold CV	Quadratic
RRARDI,POOS-CV	Linear	Ridge-PCA	POOS CV	Quadratic
RRARDI,K-fold	Linear	Ridge-PCA	K-fold CV	Quadratic
RFARDI,POOS-CV	Nonlinear	PCA	POOS CV	Quadratic
RFARDI,K-fold	Nonlinear	PCA	K-fold CV	Quadratic
KRRARDI,POOS-CV	Nonlinear	Ridge-PCR	POOS CV	Quadratic
KRRARDI,K-fold	Nonlinear	Ridge-PCR	K-fold CV	Quadratic
$(B_1, \alpha = \hat{\alpha})$,POOS-CV	Linear	EN	POOS CV	Quadratic
$(B_1, \alpha = \hat{\alpha})$,K-fold	Linear	EN	K-fold CV	Quadratic
$(B_1, \alpha = 1)$,POOS-CV	Linear	Lasso	POOS CV	Quadratic
$(B_1, \alpha = 1)$,K-fold	Linear	Lasso	K-fold CV	Quadratic
$(B_1, \alpha = 0)$,POOS-CV	Linear	Ridge	POOS CV	Quadratic
$(B_1, \alpha = 0)$,K-fold	Linear	Ridge	K-fold CV	Quadratic
$(B_2, \alpha = \hat{\alpha})$,POOS-CV	Linear	EN-PCA	POOS CV	Quadratic
$(B_2, \alpha = \hat{\alpha})$,K-fold	Linear	EN-PCA	K-fold CV	Quadratic
$(B_2, \alpha = 1)$,POOS-CV	Linear	Lasso-PCA	POOS CV	Quadratic
$(B_2, \alpha = 1)$,K-fold	Linear	Lasso-PCA	K-fold CV	Quadratic
$(B_2, \alpha = 0)$,POOS-CV	Linear	Ridge-PCA	POOS CV	Quadratic
$(B_2, \alpha = 0)$,K-fold	Linear	Ridge-PCA	K-fold CV	Quadratic
$(B_3, \alpha = \hat{\alpha})$,POOS-CV	Linear	EN-PCR	POOS CV	Quadratic
$(B_3, \alpha = \hat{\alpha})$,K-fold	Linear	EN-PCR	K-fold CV	Quadratic
$(B_3, \alpha = 1)$,POOS-CV	Linear	Lasso-PCR	POOS CV	Quadratic
$(B_3, \alpha = 1)$,K-fold	Linear	Lasso-PCR	K-fold CV	Quadratic
$(B_3, \alpha = 0)$,POOS-CV	Linear	Ridge-PCR	POOS CV	Quadratic
$(B_3, \alpha = 0)$,K-fold	Linear	Ridge-PCR	K-fold CV	Quadratic
SVR-ARDI,Lin,POOS-CV	Linear	PCA	POOS CV	$\bar{\epsilon}$-insensitive
SVR-ARDI,Lin,K-fold	Linear	PCA	K-fold CV	$\bar{\epsilon}$-insensitive
SVR-ARDI,RBF,POOS-CV	Nonlinear	PCA	POOS CV	$\bar{\epsilon}$-insensitive
SVR-ARDI,RBF,K-fold	Nonlinear	PCA	K-fold CV	$\bar{\epsilon}$-insensitive

Note: PCA stands for Principal Component Analysis, EN for Elastic Net regularizer, PCR for Principal Component Regression.

4 Empirical setup

This section presents the data and the design of the pseudo-of-sample experiment used to generate the treatment effects above.

4.1 Data

We use historical data to evaluate and compare the performance of all the forecasting models described previously. The dataset is FRED-MD, publicly available at the Federal Reserve of St-Louis's web site. It contains 134 monthly US macroeconomic and financial indicators observed from 1960M01 to 2017M12. Many macroeconomic and financial indicators are usually very persistent or not stationary. We follow Stock and Watson (2002b) and McCracken and Ng (2016) in the choice of transformations in order to achieve stationarity. The details on the dataset and the series transformation are all in McCracken and Ng (2016).

4.2 Variables of Interest

We focus on predicting five macroeconomic variables: Industrial Production (INDPRO), Unemployment rate (UNRATE), Consumer Price Index (INF), difference between 10-year Treasury Constant Maturity rate and Federal funds rate (SPREAD) and housing starts (HOUST). These are standard candidates in the forecasting literature and are representative macroeconomic indicators of the US economy. In particular, we treat INDPRO as an $I(1)$ variable so we forecast the average growth rate over h periods as in equation (3). We follow the literature and treat the price index as $I(2)$, so the target is the average change in inflation defined by equation (4). The unemployment rate is considered $I(1)$ and we target the average first-difference as in (3) but without logs. The spread and housing starts are modeled as $I(0)$ and the targets are constructed as in (2).

4.3 Pseudo-Out-of-Sample Experiment Design

The pseudo-out-of-sample period is 1980M01 - 2017M12. The forecasting horizons considered are 1, 3, 9, 12 and 24 months. Hence, there are 456 evaluation periods for each horizon. All models are estimated recursively with an expanding window.

Hyperparameter fine tuning is done with in-sample criterion (AIC and BIC) and using two types of cross validation (POOS CV and k-fold). The in-sample model selection is standard, we only fix the upper bounds for the set of HPs. In contrast, the CV can be very computationally extensive in a long time series evaluation period as in this paper. Ideally, one would re-optimize every model, for every target variable and for each forecasting horizon, for every out-of-sample period. As we have 456 evaluation observations, five variables,

five horizons and many models, this is extremely demanding especially for the POOS CV where the CV in the validation set mimics the out-of-sample prediction in the test sample. Therefore, for POOS CV case, the POOS period consists of last five years in the validation set. In case of k-fold CV, we set $k = 5$. We re-optimize hyperparameters every two years. This is reasonable since as it is the case with parameters, we do not expect hyperparameters to change drastically with the addition of a few data points.

Appendix D describes both cross-validation techniques in details, while the information on upper / lower bounds and grid search for hyperaparameters for every model is available in Appendix E.

4.4 Forecast Evaluation Metrics

Following a standard practice in the forecasting literature, we evaluate the quality of our point forecasts using the root Mean Square Prediction Error (MSPE). The standard Diebold-Mariano (DM) test procedure is used to compare the predictive accuracy of each model against the reference (ARDI,BIC) model.

We also implement the Model Confidence Set (MCS) introduced in Hansen et al. (2011). The MCS allows us to select the subset of best models at a given confidence level. It is constructed by first finding the best forecasting model, and then selecting the subset of models that are not significantly different from the best model at a desired confidence level. We construct each MCS based on the quadratic loss function and 4000 bootstrap replications. As expected, we find that the $(1 - \alpha)$ MCS contains more models when α is smaller. Following Hansen et al. (2011), we present the empirical results for 75% confidence interval.

These evaluation metrics are standard outputs in a forecasting horse race. They allow to verify the overall predictive performance and to classify models according to DM and MCS tests. Regression analysis from section 2.3 will be used to distinguish the marginal treatment effect of each ML ingredient that we try to evaluate here.

5 Results

We present the results in several ways. First, for each variable, we show standard tables containing the relative root MSPEs (to AR,BIC model) with DM and MCS outputs, for the whole pseudo-out-of-sample and NBER recession periods. Second, we evaluate the marginal effect of important features of ML using regressions described in section 2.3.

5.1 Overall Predictive Performance

Tables 3 - 7, in Appendix A, summarize the overall predictive performance in terms of root MSPE relative to the reference model AR,BIC. The analysis is done for the full out-of-sample as well as for NBER recessions taken separately (i.e., when the target belongs to a recession episode). This address two questions: is ML already useful for macroeconomic forecasting and when?[11]

In case of industrial production, Table 3 shows that the SVR-ARDI with linear kernel and K-fold cross-validation is the best at $h = 1$. Big ARDI version with Lasso penalty and K-fold CV minimizes the MSE 3-month ahead, while the kernel ridge AR with K-fold is best for $h = 9$. At longer horizons, the ridge ARDI is the best option with an improvement of more than 10%. During recessions, the ARDI with CV is the best for all horizons except the one-year ahead where the minimum MSE is obtained with RRARDI,K-fold. Ameliorations with respect to AR,BIC are much larger during economic downturns, and the MCS selects less models.

Results for the unemployment rate, table 4, highlight the performance of nonlinear models: Kernel ridge and Random forests. Improvements with respect to the AR,BIC model are bigger for both full OOS and recessions. MCSs are narrower than in case of INDPRO. Similar pattern is observed during NBER recessions. Table 5 summarizes results for the Spread. Nonlinear models are generally the best, combined with H_t^+ predictors' set. Occasionally, autoregressive models with the kernel ridge or SVR specifications produce minimum MSE.

In case of inflation, table 6 shows that simple autoregressive models are the best for the full out-of-sample, except for $h = 1$. It changes during recessions where ARDI models improve upon autoregressions for horizons of 9, 12 and 24. This finding is similar to Kotchoni et al. (2017) who document that ARMA(1,1) is in general the best forecasting model for inflation change. Finally, housing starts are best predicted with data-poor models, except for short horizons and few cases during recessions. Nonlinearity seems to help only for 2-year ahead forecasting during economic downturns.

Overall, using data-rich models and nonlinear g functions seems to be a game changer for predicting real activity series and term spread, which is itself usually a predictor of the business cycle (Estrella and Mishkin (1998)). SVR specifications are occasionally among the best models as well as the shrinkage methods from section 3.2. When predicting inflation change and housing starts, autoregressive models are generally preferred, but are dominated by data-rich models during recessions. These findings suggest that machine learning treatments and data-rich models can ameliorate predictions of important macroeconomic variables. In addition, their marginal contribution depends on the state of the economy.

[11] The knowledge of the models that have performed best historically during recessions is of interest for practitioners. If the probability of recession is high enough at a given period, our results can provide an ex-ante guidance on which model is likely to perform best in such circumstances.

5.2 Disentangling ML Treatment Effects

The results in the previous section does not allow easily to disentangle the marginal effects of important features of machine learning as presented in section 3, which is the most important goal of this paper. Before we employ the evaluation strategy depicted in section 2.3, we first use a Random forest as an exploration tool. Since creating the relevant dummies and interaction terms to fully describe the environment is a hard task in presence of many treatment effects, a regression tree well suited to reveal the potential of ML features in explaining the results from our experiment. We report the importance of each features in what is a potentially a very non-linear model.[12] For instance, the tree could automatically create interactions such as $I(NL = 1) * I(h \leq 12)$, that is, some condition on non-linearities and horizon forecast.

Figure 1 plots the relative importance of machine learning features in our macroeconomic forecasting experiment. The space of possible interaction is constructed with dummies for horizon, variable, recession periods, loss function and H_t^+, and categorical variables nonlinearity, shrinkage and hyperameters' tuning that follow the classification as in Table 1. As expected, target variables, forecasting horizons, the state of economy and data richness are important elements. Nonlinearity is relevant, which confirms our overall analysis from the previous section. More interestingly, interactions with shrinkage and cross-validation emerge as very important ingredients for macroeconomic forecasting, something that we might have underestimated from tables containing relative MSE. Loss function appears as the least important feature.

Despite its richness in terms of interactions among determinants, the Random forest analysis does not provide the sign of the importance of each feature not it measures their

[12] The importance of each ML ingredient is obtain with feature permutation. The following process describes the estimation of out-of-bag predictor importance values by permutation. Suppose a random forest of B trees and p is the number of features.

1. For tree b, $b = 1, ..., B$:
   (a) Identify out-of-bag observations and indices of features that were split to grow tree b, $s_b \subseteq 1, ..., p$.
   (b) Estimate the out-of-bag error $u^2_{t,h,v,m,\mathbf{b}}$.
   (c) For each feature x_j, $j \in s_b$:
      i. Randomly permute the observations of x_j.
      ii. Estimate the model squared errors, $u^2_{t,h,v,m,\mathbf{b,j}}$, using the out-of-bag observations containing the permuted values of x_j.
      iii. Take the difference $d_{\mathbf{bj}} = u^2_{t,h,v,m,\mathbf{b,j}} - u^2_{t,h,v,m,\mathbf{b}}$.
2. For each predictor variable in the training data, compute the mean, $\bar{d}_j$, and standard deviation, σ_j, of these differences over all trees, $j = 1, ..., p$.
3. The out-of-bag predictor importance by permutation for x_j is $\bar{d}_j / \sigma_j$

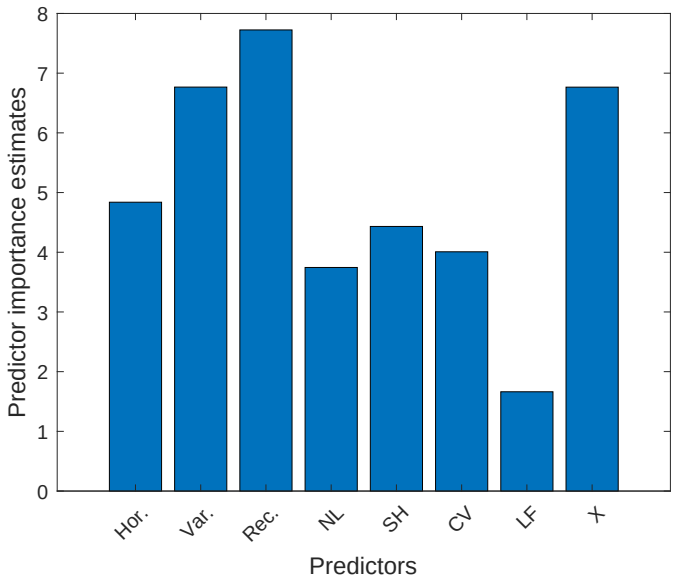

Figure 1: This figure presents predictive importance estimates. Random forest is trained to predict $R^2_{t,h,v,m}$ defined in (11) and use out-of-bags observations to assess the performance of the model and compute features' importance. NL, SH, CV and LF stand for nonlinearity, shrinkage, cross-validation and loss function features respectively. A dummy for H_t^+ models, X, is included as well.

marginal contributions. To do so, and armed with insights from the Random forest analysis, we turn now to regression analysis described in section 2.3, .

Figure 2 shows the distribution of $\dot{\alpha}_F^{(h,v)}$ from equation (11) done by (h, v) subsets. Hence, here we allow for heterogeneous treatment effects according to 25 different targets. This figure highlights by itself the main findings of this paper. **First**, non-linearities either improve drastically forecasting accuracy or decrease it substantially. There is no middle ground, as shown by the area around the 0 line being quite uncrowded. The marginal benefits of data-rich models seems roughly to increase with horizons for every variables except inflation. The effects are positive and significant for INDPRO, UNRATE and SPREAD at the last three horizons. **Second**, standard alternative methods of dimensionality reduction do not improve on average over the standard factor model. **Third**, the average effect of CV is 0. However, as we will see in section 5.2.3, the averaging in this case hides some interesting and relevant differences between K-fold and POOS CVs, that the Random forest analysis in Figure 1 has picked up. **Fourth**, on average, dropping the standard in-sample squared-loss function for what the SVR proposes is not useful, except in very rare cases. **Fifth** and lastly, the marginal benefits of data-rich models (X) increase with horizons for INDPRO, UNRATE and SPREAD. For INF and HOUST, benefits are on average non-statistically different from zero. Note that this is almost exactly like the picture we described for NL. Indeed, visually, it seems like the results for X are a compressed-range version of NL that was translated to the right. Seeing NL models as data augmentation via some basis expansions, we can conclude

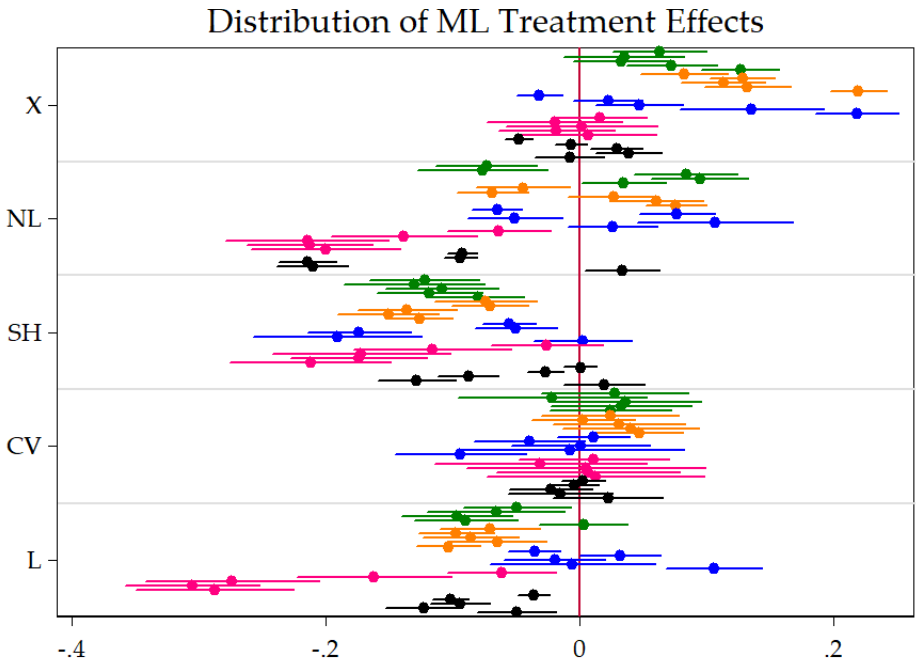

Figure 2: This figure plots the distribution of $\grave{\alpha}_F^{(h,v)}$ from equation (11) done by (h, v) subsets. That is, we are looking at the average partial effect on the pseudo-OOS R^2 from augmenting the model with ML features, keeping everything else fixed. X is making the switch from data-poor to data-rich. Finally, variables are INDPRO, UNRATE, SPREAD, INF and HOUST. Within a specific color block, the horizon increases from $h = 1$ to $h = 24$ as we are going down. As an example, we clearly see that the partial effect of X on the R^2 of INF increases drastically with the forecasted horizon h. SEs are HAC. These are the 95% confidence bands.

that for INDPRO, UNRATE and SPREAD at longer horizons, we either need to augment the AR(p) model with more regressors either created from the lags of the dependent variable itself or coming from additional data. The possibility of joining these two forces to create a "data-filthy-rich" model is studied in section 5.2.1.

It turns out these findings are somewhat robust as graphs included in the appendix section B show. ML treatment effects plots of very similar shapes are obtained for data-poor models only (Figure 12), data-rich models only (Figure 13), recessions periods (Figure 14) and the last 20 years of the forecasting exercise (Figure 15).

Finally, Figure 3 aggregates by h and v in order to clarify whether variable or horizon heterogeneity matters most. Two facts detailed earlier and now are quite easy to see. For both X and NL, the average marginal effect increase in h. Now, the effect sign (or it being statistically different from 0) is truly variable-dependent: the first three variables are those that benefit the most from both additional information and non-linearities. This grouping should not come as a surprise since the 3 variables all represent real activity.

In what follows we break down averages and run specific regressions as in (12) to study how homogeneous are the $\grave{\alpha}_F$'s reported above.

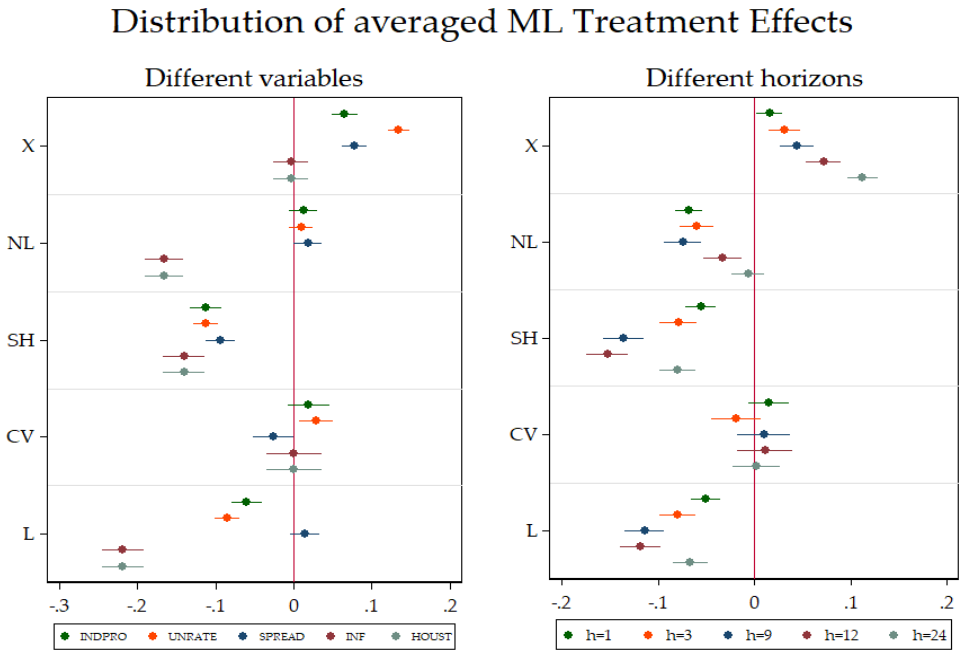

Figure 3: This figure plots the distribution of $\dot{\alpha}_F^{(v)}$ and $\dot{\alpha}_F^{(h)}$ from equation (11) done by h and v subsets. That is, we are looking at the average partial effect on the pseudo-OOS R^2 from augmenting the model with ML features, keeping everything else fixed. X is making the switch from data-poor to data-rich. However, in this graph, $v-$specific heterogeneity and $h-$specific heterogeneity have been integrated out in turns. SEs are HAC. These are the 95% confidence bands.

5.2.1 Non-linearities

Figure 4 suggests that non-linearities can be very helpful at forecasting both UNRATE and SPREAD in the data rich-environment. The marginal effects of Random Forests and KRR are almost never statistically different for data-rich models, suggesting that the common NL feature is the driving force. However, this is not the case for data-poor models where only KRR shows R^2 improvements for UNRATE and SPREAD, except for INDPRO where both non-linear features has similar positive effects. Nonlinearity is harmful for predicting inflation change and housing, irrespective of data size.

Figure 5 suggest that non-linearities are more useful for longer horizons in data rich environment while they can be harmful in short-horizons. Note again that both non-linear models follow the same pattern for data-rich models with Random Forest always being better (but never statistically different from KRR). For data-poor models, it is KRR that has a (statistically significant) growing advantage as h increases.

Seeing NL models as data augmentation via some basis expansions, we can join the two facts together to conclude that the need for a complex and "data-filthy-rich" model arise for INDPRO, UNRATE and SPREAD at longer horizons.

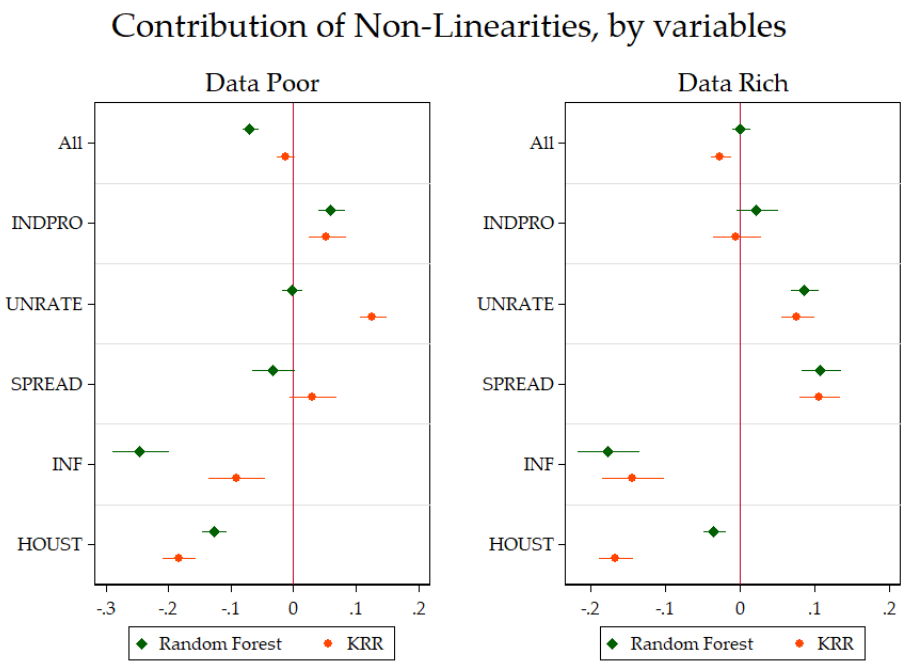

Figure 4: This compares the two NL models averaged over all horizons. The unit of the x-axis are improvements in OOS R^2 over the basis model. SEs are HAC. These are the 95% confidence bands.

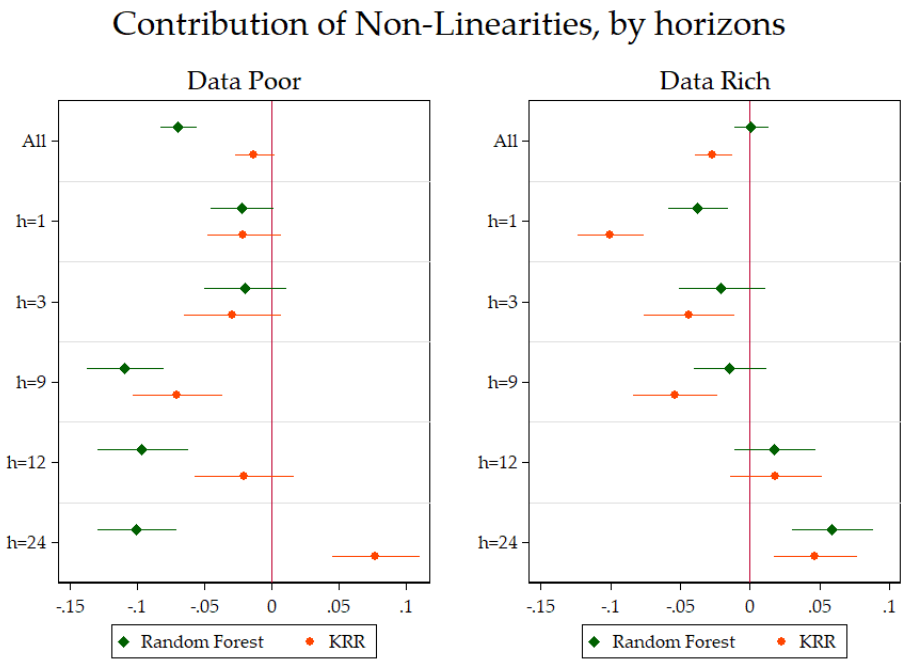

Figure 5: This compares the two NL models averaged over all variables. The unit of the x-axis are improvements in OOS R^2 over the basis model. SEs are HAC. These are the 95% confidence bands.

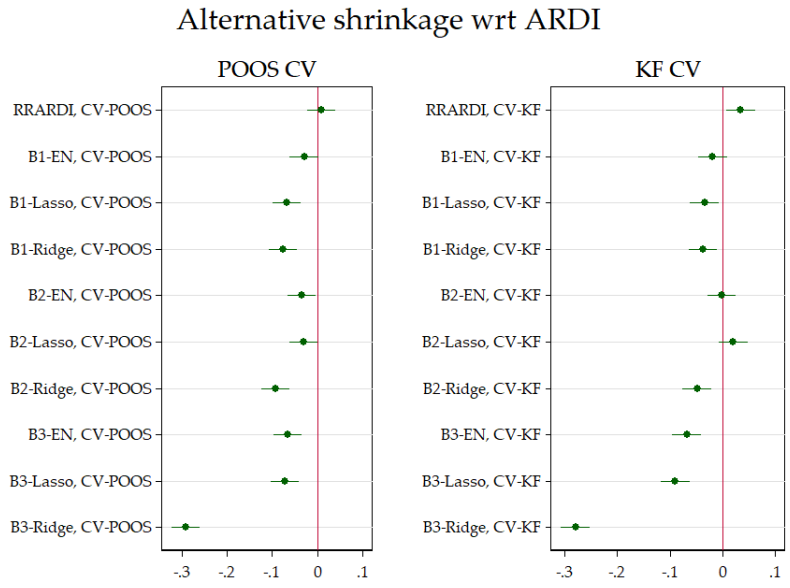

Figure 6: This compares models of section 3.2 averaged over all variables and horizons. The unit of the x-axis are improvements in OOS R^2 over the basis model. The base models are ARDIs specified with POOS-CV and KF-CV respectively. SEs are HAC. These are the 95% confidence bands.

5.2.2 Alternative Dimension Reduction

Figure 6 shows that the ARDI reduces dimensionality in a way that certainly works well with economic data: all competing schemes do at most as good on average. It is overall safe to say that on average, all shrinkage schemes give similar or lower performance. No clear superiority for the Bayesian versions of some of these models was also documented in De Mol et al. (2008). This suggests that the factor model view of the macroeconomy is quite accurate in the sense that when we use as a mean of dimensionality reduction, it extracts the most relevant information to forecast the relevant time series. This is good news. The ARDI is the simplest model to run and results from the preceding section tells us that adding non-linearities to an ARDI can be quite helpful. For instance, B_1 models where we basically keep all regressors do approximately as well as the ARDI when used with CV-POOS. However, it is very hard to consider non-linearities in this high-dimensional setup. Since the ARDI does a similar (or better) job of dimensionality reduction, it is both convenient for subsequent modeling steps and does not loose relevant information.

Obviously, the deceiving average behavior of alternative (standard) shrinkage methods does not mean there cannot be interesting (h, v) cases where using a different dimensionality reduction has significant benefits as discussed in section 5.1 and Smeekes and Wijler (2018). Furthermore, LASSO and Ridge can still be useful to tackle specific time series econometrics problems (other than dimensionality reduction), as shown with time-varying parameters in Goulet Coulombe (2019).

Figure 6 indicates that the RRARDI-KF performs quite well with respect to ARDI-KF. Figure 7, in next section, shows that the former ends up considering many more total regressors than the latter – but less than RRARDI-POOS. However, the interesting question is whether RRARDI-KF is better on average than *any* ARDIs considered in this paper. The answers turns out to be a strong yes in Figure 16, in Appendix C. Does that superiority still holds when breaking things down by h and v? Figure 17 procures another strong yes.

5.2.3 Hyperparameter Optimization

Figure 7 shows how many total regressors are kept by different model selection methods. As expected, BIC is almost always the lower envelope of each of these graphs and is the only true guardian of parsimony in our setup. AIC also selects relatively sparse models. It is also quite visually clear that both cross-validations favors larger models. Most likely as a results of expanding window setup, we see a common upward trends for all model selection methods. Finally, CV-POOS has quite a distinctive behavior. It is more volatile and seems to select bigger models in similar times for all series (around 1990 and after 2005). While K-fold also selects models of considerable size, it does so in a more slowly growing fashion. This is not surprising given the fact that K-fold samples from all available data to build the CV criterion: adding new data points only gradually change the average. CV-POOS is a short rolling window approach that offers flexibility against structural hyperparameters change at the cost of greater variance and vulnerability of rapid change of regimes in the data.

Following intuition, the Ridge regression ARDI models are most often richer than their non-penalized counterparts. When combined with CV-KF, we get the best ARDI (on average), as seen in Figure 16. For instance, we see in Figure 17 that the RR-ARDI-KF performs quite well for INDPRO. Figure 7 informs us that it is because that specific factor model has constantly more lags and factors (up to 120) than any other version of the ARDI model considered in this paper.

We know that different model selection methods lead to quite different models, but what about their predictions? Table 2 tells many interesting tales. The models included in the regressions are the standard linear ARs and ARDIs (that is, excluding the Ridge versions) that have all been tuned using BIC, AIC, CV-POOS and CV-KF. First, we see that overall, only CV-POOS is distinctively worse. We see that this is attributable mostly to recessions in *both* data-poor and data-rich environments – with 6.91% and 8.25% losses in OOS-R^2 respectively. However, CV-POOS is still doing significantly worse by 2.7% for data-rich models even in expansion periods. For data-poor models, AIC and CV-KF have very similar behavior, being slightly worse than BIC in expansions and significantly better in recessions. Finally, for data rich models, CV-KF does better than any other criterion on average and that difference is 3.87% and statistically significant in recessions. This suggest that this particular form of ML treatment effect is useful.

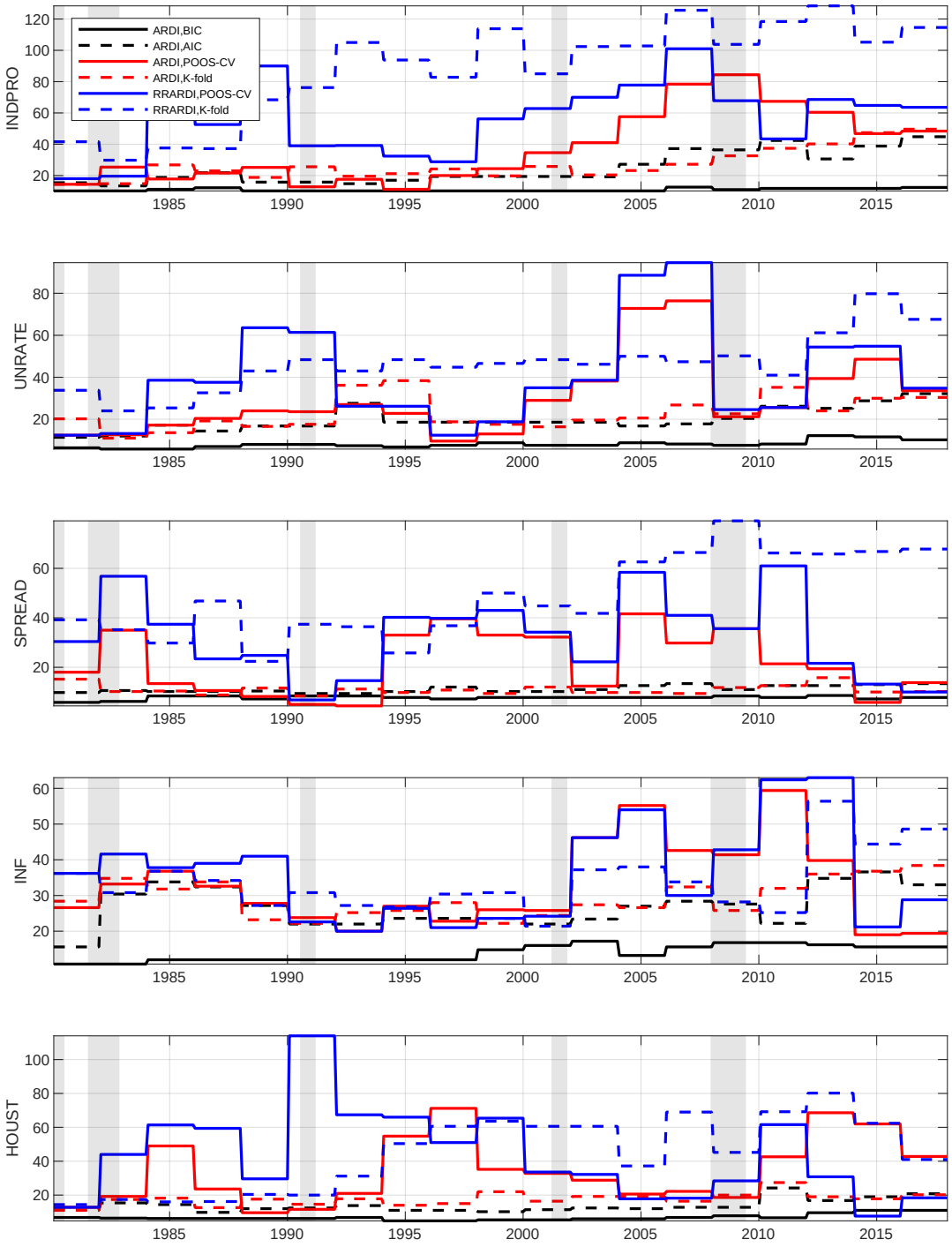

Figure 7: This shows the total number of regressors for the linear ARDI models. Results averaged across horizons.

Another conclusion is that, for that class of models, we can safely opt for either BIC or CV-KF. Assuming some degree of external validity beyond that model class, we can be reassured that the quasi-necessity of leaving ICs behind when opting for more complicated ML models is not harmful.

Table 2: CV comparaison

	(1) All	(2) Data-rich	(3) Data-poor	(4) Data-rich	(5) Data-poor
CV-KF	-0.00927	0.706	-0.725	0.230	-1.092*
	(0.586)	(0.569)	(0.443)	(0.608)	(0.472)
CV-POOS	-2.272***	-3.382***	-1.161**	-2.704***	-0.312
	(0.586)	(0.569)	(0.443)	(0.608)	(0.472)
AIC	-0.819	-0.867	-0.771	-0.925	-1.258*
	(0.676)	(0.657)	(0.511)	(0.702)	(0.546)
CV-KF * Recessions				3.877*	2.988*
				(1.734)	(1.348)
CV-POOS * Recessions				-5.525**	-6.914***
				(1.734)	(1.348)
AIC * Recessions				0.470	3.970*
				(2.002)	(1.557)
Observations	136800	68400	68400	68400	68400

Standard errors in parentheses. Units are percentage of OOS-R^2.

* $p < 0.05$, ** $p < 0.01$, *** $p < 0.001$

We will now consider models that are usually always tuned by CV and compare the performance of the two CVs by horizon and variables.

Since we are now pooling multiple models, including all the alternative shrinkage models, if a clear pattern only attributable to a certain CV existed, it would most likely appear in Figure 8. What we see are two things. First, CV-KF is at least as good as CV-POOS on average for every variables and horizons, irrespective of the informational content of the regression. When there is statistically significant difference – which happens quite often – it is always in favor of CV-KF. These effects are magnified when we concentrate on the data-rich environment.

Figure 9's message has the virtue of clarity. CV-POOS's failure is mostly attributable to its poor record in recessions periods for the first three variables at any horizon. Note that this is the same subset of variables that benefits from adding in more data (X) and on-linearities as discussed in 5.2.1.

Intuitively, by using only recent data, CV-POOS will be more robust to gradual structural change but will perhaps have an Achilles heel in regime switching behavior. If optimal hyperparameters are state-dependent, then a switch from expansion to recession at time t

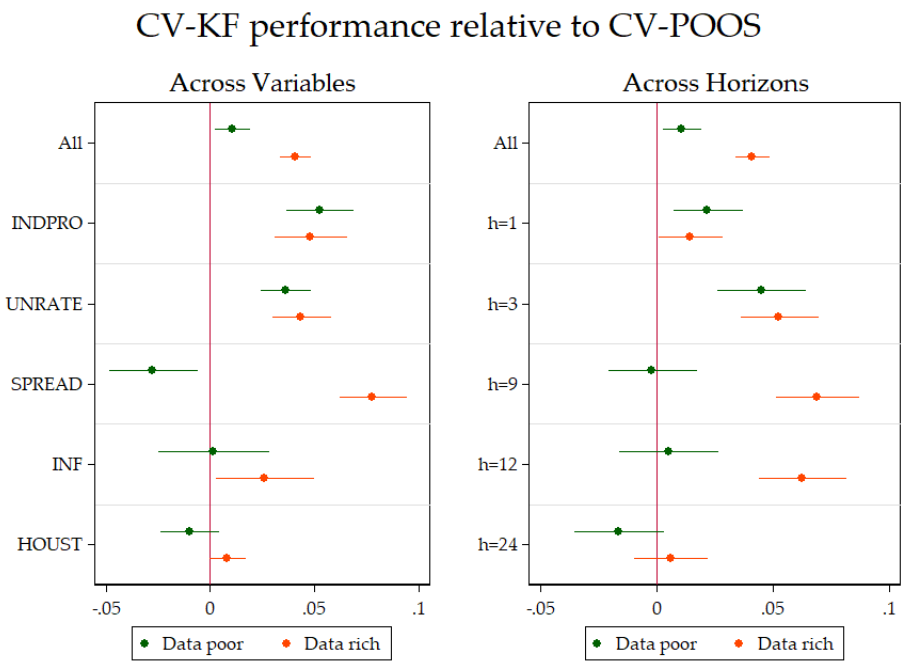

Figure 8: This compares the two CVs procedure averaged over all the models that use them. The unit of the x-axis are improvements in OOS R^2 over the basis model. SEs are HAC. These are the 95% confidence bands.

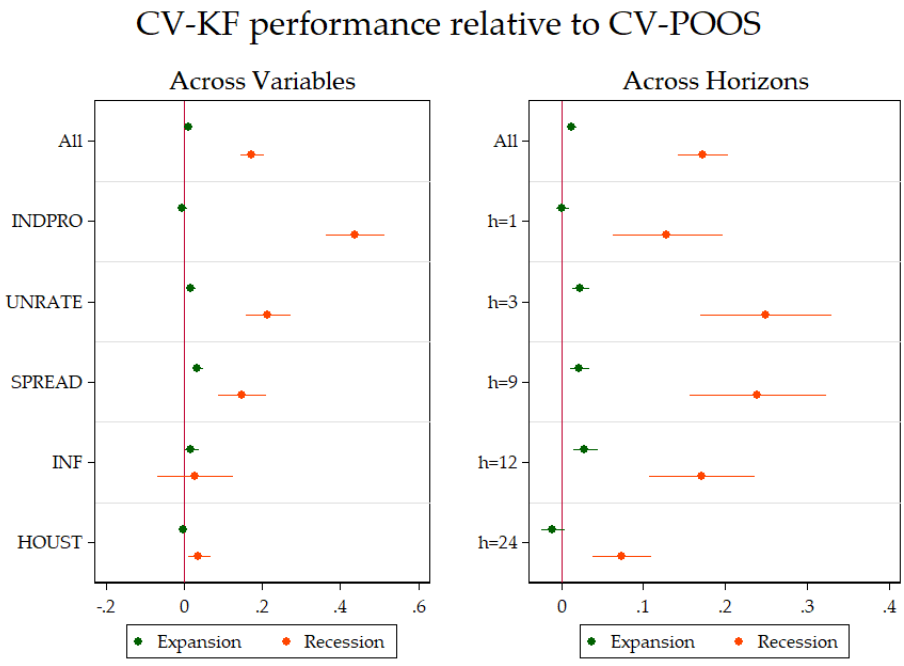

Figure 9: This compares the two CVs procedure averaged over all the models that use them. The unit of the x-axis are improvements in OOS R^2 over the basis model. SEs are HAC. These are the 95% confidence bands.

can be quite harmful. K-fold, by taking the average over the whole sample, is less immune to such problems. Since results in 5.1 point in the direction that smaller models are better in expansions and bigger models in recessions, the behavior of CV and how it picks the effective complexity of the model can have an important effect on overall predictive ability. This is exactly what we see in Figure 9: CV-POOS is having a hard time in recessions with respect to K-fold.[13]

5.2.4 Loss Function

In this section, we investigate whether replacing the l_2 norm as an in-sample loss function for the SVR machinery helps in forecasting. We again use as baseline models ARs and ARDIs trained by the same corresponding CVs. The very nature of this ML feature is that the model is less sensible to extreme residuals, thanks to the ϵ-insensitivity tube. We first compare linear models in Figure 10. Clearly, changing the loss function is mostly very harmful and that is mostly due to recessions period. However, in expansion, the linear SVR is better on average than a standard ARDI for UNRATE and SPREAD, but these small gains are clearly offset (on average) by the huge recession losses.

The SVR (or the better-known SVM) is usually used in its non-linear form. We hereby compare KRR and SVR-NL to study whether the loss function effect could reverse when a non-linear model is considered. Comparing these models makes sense since they both use the same kernel trick (with a RBF kernel). Hence, like linear models of Figure 10, models in Figure 11 only differ by the use of a different loss function $\hat{L}$. It turns out conclusions are exactly the same as for linear models with the negative effects being slightly larger. Furthermore, Figures 18 and 19 confirm that these findings are found in both the data-rich and the data-poor environments. Hence, these results confirms that $\hat{L}$ is not the most salient feature of ML, at least for macroeconomic forecasting. If researchers are interested in using its associated kernel trick to bring in non-linearities, they should rather use the lesser-known KRR.

[13]Of course, CV-POOS has hyper-hyperparameters of its own as described in detail in the appendix D and these can change moderately the outcome. For instance, considering an expanding *test* window in the cross-validation recursive scheme could reduce greatly its volatility. However, the setup of the CV-POOS used in this paper corresponds to what is standard in the literature.

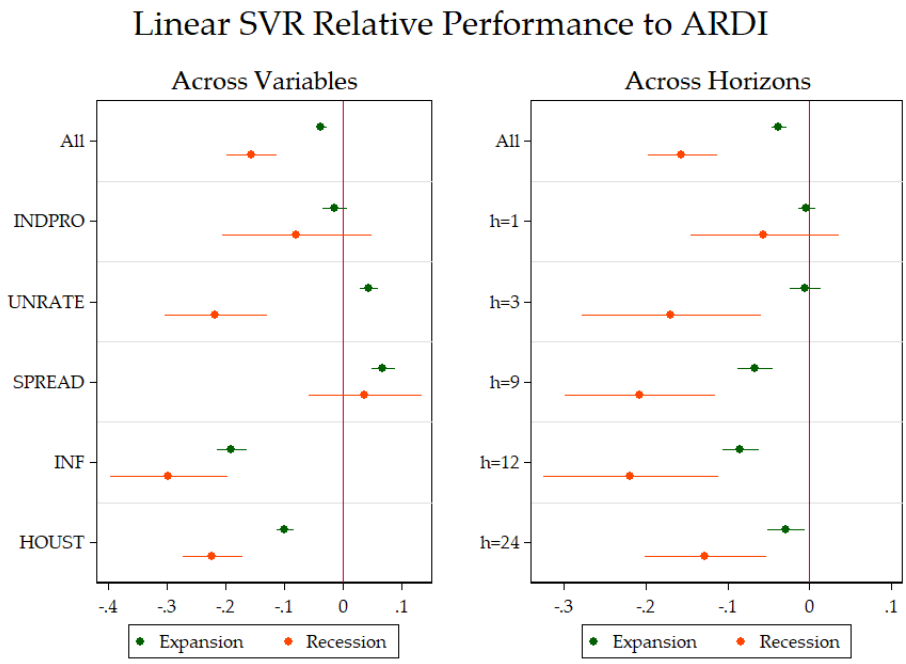

Figure 10: This graph display the marginal (un)improvments by variables and horizons to opt for the SVR in-sample loss function in **both the data-poor and data-rich environments**. The unit of the x-axis are improvements in OOS R^2 over the basis model. SEs are HAC. These are the 95% confidence bands.

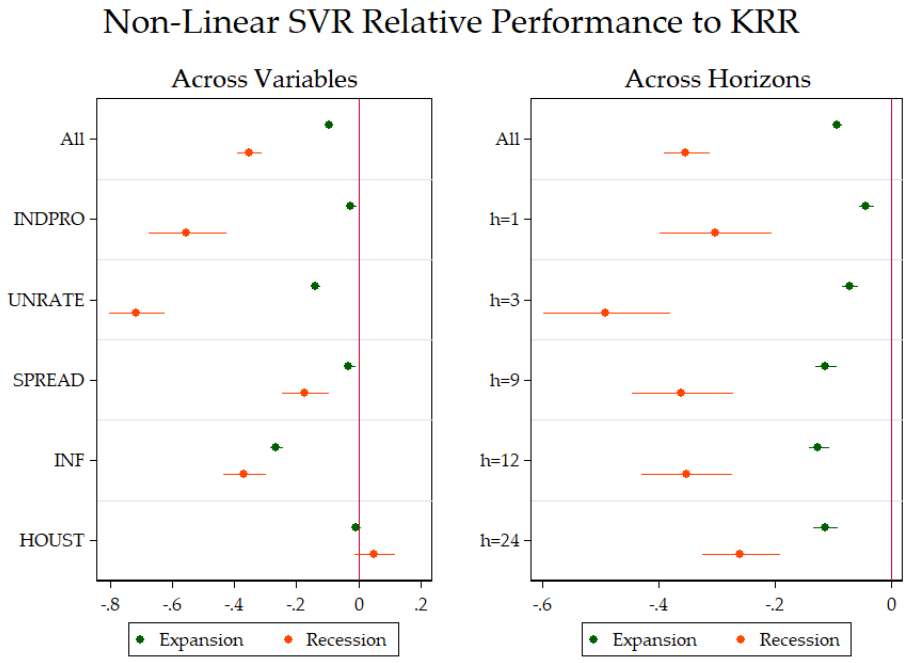

Figure 11: This graph display the marginal (un)improvments by variables and horizons to opt for the SVR in-sample loss function in **both recession and expansion periods**. The unit of the x-axis are improvements in OOS R^2 over the basis model. SEs are HAC. These are the 95% confidence bands.

6 Conclusion

In this papers we have studied important underlying features driving machine learning techniques in the context of macroeconomic forecasting. We have considered many machine learning methods in a substantive POOS setup over almost 40 years for 5 key variables and 5 different horizons. We have classified these models by "features" of machine learning: non-linearities, regularization, cross-validation and alternative loss function. The four aspects of ML are nonlinearities, regularization, cross-validation and alternative loss function. The data-rich and data-poor environments were considered. In order to recover their marginal effects on forecasting performance, we designed a series of experiments that easily allow to identify the treatment effects of interest.

The first result point in the direction that non-linearities are the true game-changer for the data rich environment, especially when predicting real activity series and at long horizons. This gives a stark recommendation for practitioners. It recommends for most variables and horizons what is in the end a partially non-linear factor model – that is, factors are still obtained by PCA. The best of ML (at least of what considered here) can be obtained by simply generating the data for a standard ARDI model and then feed it into a ML non-linear function of choice. The second result is that the standard factor model remains the best regularization. Third, if cross-validation has to be applied to select models' features, the best practice is the standard K-fold. Finally, one should stick with the standard L_2 loss function.

References

Ahmed, N. K., Atiya, A. F., Gayar, N. E., and El-Shishiny, H. (2010). An empirical comparison of machine learning models for time series forecasting. *Econometric Reviews*, 29(5):594–621.

Athey, S. (2018). The impact of machine learning on economics. *The Economics of Artificial Intelligence, NBER volume*, Forthcoming.

Bai, J. and Ng, S. (2002). Determining the number of factors in approximate factor models. *Econometrica*, 70(1):191–221.

Bergmeir, C. and Benítez, J. M. (2012). On the use of cross-validation for time series predictor evaluation. *Information Sciences*, 191:192–213.

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

Boh, S., Borgioli, S., Coman, A. B., Chiriacescu, B., Koban, A., Veiga, J., Kusmierczyk, P.,

Pirovano, M., and Schepens, T. (2017). European macroprudential database. Technical report, IFC Bulletins chapters, 46.

Chen, J., Dunn, A., Hood, K., Driessen, A., and Batch, A. (2019). Off to the races: A comparison of machine learning and alternative data for predicting economic indicators. Technical report, Bureau of Economic Analysis.

Claeskens, G. and Hjort, N. L. (2008). *Model selection and model averaging*. Cambridge University Press, Cambridge, U.K.

De Mol, C., Giannone, D., and Reichlin, L. (2008). Forecasting using a large number of predictors: Is bayesian shrinkage a valid alternative to principal components? *Journal of Econometrics*, 146:318–328.

Diebold, F. X. and Mariano, R. S. (1995). Comparing predictive accuracy. *Journal of Business and Economic Statistics*, 13:253–263.

Diebold, F. X. and Shin, M. (2018). Machine learning for regularized survey forecast combination: Partially-egalitarian lasso and its derivatives. *International Journal of Forecasting*, forthcoming.

Döpke, J., Fritsche, U., and Pierdzioch, C. (2015). Predicting recessions with boosted regression trees. Technical report, George Washington University, Working Papers No 2015-004, Germany.

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

Fortin-Gagnon, O., Leroux, M., Stevanovic, D., and Surprenant, S. (2018). A large canadian database for macroeconomic analysis. Technical report, Department of Economics, UQAM.

Giannone, D., Lenza, M., and Primiceri, G. (2015). Prior selection for vector autoregressions. *Review of Economics and Statistics*, 97(2):436–451.

Giannone, D., Lenza, M., and Primiceri, G. (2017). Macroeconomic prediction with big data: the illusion of sparsity. Technical report, Federal Reserve Bank of New York.

Goulet Coulombe, P. (2019). Sparse and dense time-varying parameters using machine learning. Technical report.

Granger, C. W. J. and Jeon, Y. (2004). Thick modeling. *Economic Modelling*, 21:323–343.

Hansen, P., Lunde, A., and Nason, J. (2011). The model confidence set. *Econometrica*, 79(2):453–497.

Hansen, P. R. and Timmermann, A. (2015). Equivalence between out-of-sample forecast comparisons and wald statistics. *Econometrica*, 83(6):2485–2505.

Hastie, T., Tibshirani, R., and Friedman, J. (2017). *The Elements of Statistical Learning*. Springer Series in Statistics. Springer-Verlag, New York.

Kilian, L. and Lütkepohl, H. (2017). *Structural vector autoregressive analysis*. Cambridge University Press.

Kotchoni, R., Leroux, M., and Stevanovic, D. (2017). Macroeconomic forecast accuracy in a data-rich environment. Technical report, CIRANO, 2017s-05.

Litterman, R. B. (1979). Techniques of forecasting using vector autoregressions. Technical report.

Marcellino, M., Stock, J. H., and Watson, M. W. (2006). A comparison of direct and iterated multistep ar methods for forecasting macroeconomic time series. *Journal of Econometrics*, 135:499–526.

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

Mullainathan, S. and Spiess, J. (2017). Machine learning: An applied econometric approach. *Journal of Economic Perspectives*, 31(2):574–589.

Nakamura, E. (2005). Inflation forecasting using a neural network. *Economics Letters*, 86(3):373–378.

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

Sermpinis, G., Stasinakis, C., Theolatos, K., and Karathanasopoulos, A. (2014). Inflation and unemployment forecasting with genetic support vector regression. *Journal of Forecasting*, 33(6):471–487.

Smalter, H. A. and Cook, T. R. A. (2017). Macroeconomic indicator forecasting with deep neural networks. Technical report, Federal Reserve Bank of Kansas City.

Smeekes, S. and Wijler, E. (2018). Macroeconomic forecasting using penalized regression methods. *International Journal of Forecasting*, 34(3):408–430.

Smola, A. J. and Schölkopf, B. (2004). A tutorial on support vector regression. *Statistics and computing*, 14(3):199–211.

Stock, J. H. and Watson, M. W. (2002a). Forecasting using principal components from a large number of predictors. *Journal of the American Statistical Association*, 97:1167–1179.

Stock, J. H. and Watson, M. W. (2002b). Macroeconomic forecasting using diffusion indexes. *Journal of Business and Economic Statistics*, 20(2):147–162.

Tashman, L. J. (2000). Out-of-sample tests of forecasting accuracy: an analysis and review. *International Journal of Forecasting*, 16(4):437–450.

Ulke, V., Sahin, A., and Subasi, A. (2016). A comparison of time series and machine learning models for inflation forecasting: empirical evidence from the USA. *Neural Computing and Applications*, 1.

Zou, H., Hastie, T., and Tibshirani, R. (2007). On the “degrees of freedom" of the Lasso. *The Annals of Statistics*, 35(5):2173–2192.

A Detailed overall predictive performance

Table 3: Industrial Production: Relative Root MSPE

	Full Out-of-Sample					NBER Recessions Periods				
Models	h=1	h=3	h=9	h=12	h=24	h=1	h=3	h=9	h=12	h=24
Data-poor (H_t^-) models										
AR,BIC	1	**1.000**	1	**1.000**	1	1	1	1	1	1
AR,AIC	0.991*	**1.000**	0,999	**1.000**	1	0.987*	1	1	1	1
AR,POOS-CV	0,998	**1.044****	0,988	**0.998**	1.030*	1.012*	1.086***	0.989*	1,001	1.076**
AR,K-fold	0.991*	**1.000**	0,998	**1.000**	1.034*	0.987*	1	1	1	1.077**
RRAR,POOS-CV	1,043	**1.112***	1.028*	1.026**	**0.973****	1.176**	1.229**	1.040*	1,005	0.950***
RRAR,K-fold	0.985*	**1.019****	0,998	**1.005***	1.033**	1,022	1.049***	1.009**	1.006**	1.061**
RFAR,POOS-CV	0,999	**1.031**	**0.977**	**0.951**	0,992	1,023	1,043	0.914**	0.883**	1,002
RFAR,K-fold	1,004	**1.020**	**0.939***	**0.933****	**0.988**	1,031	1,012	0.871***	0.892***	0.962**
KRR-AR,POOS-CV	1,032	**1.017**	**0.901***	0,995	**0.949**	1.122*	1,019	**0.791*****	0.890***	0.887***
KRR,AR,K-fold	1,017	**1.056**	**0.903***	**0.959**	**0.934***	1.147*	1,136	**0.799*****	0.861***	0.887**
SVR-AR,Lin,POOS-CV	0,993	**1.046*****	1.043**	1.062***	**0.970****	1.026*	1.094***	1.066***	1.067***	0.943***
SVR-AR,Lin,K-fold	0.977**	**1.017**	1.050**	1.068***	**0.976****	1,001	1.047**	1.068***	1.074***	0.964***
SVR-AR,RBF,POOS-CV	1,055	1.134**	1,042	1.042*	0,987	1.162**	1.224**	0.945**	0.955**	0.937***
SVR-AR,RBF,K-fold	1,053	**1.145****	1,004	**0.971**	**0.945*****	1.253***	1.308***	0.913***	0.911***	0.949**
Data-rich (H_t^+) models										
ARDI,BIC	**0.946***	**0.991**	1,037	1,004	**0.968**	**0.801*****	**0.807*****	0.887**	0.833***	0.784***
ARDI,AIC	0.959*	**0.968**	1,017	**0.998**	**0.943**	0.840***	**0.803*****	**0.844****	**0.798****	**0.768*****
ARDI,POOS-CV	**0.934****	**1.042**	0,999	**1.020**	**0.925**	**0.807*****	**0.704*****	**0.767*****	**0.829****	**0.706*****
ARDI,K-fold	**0.940***	**0.977**	1,013	**0.982**	**0.941**	**0.787*****	**0.812*****	**0.841****	**0.808****	**0.730*****
RRARDI,POOS-CV	0.966*	**1.087**	0,984	**0.947**	**0.882****	0.925**	0.900*	0.878***	**0.761*****	**0.728*****
RRARDI,K-fold	**0.934*****	**0.940**	**0.931**	**0.911**	**0.919***	0.863***	**0.766*****	**0.816*****	**0.760*****	**0.718*****
RFARDI,POOS-CV	**0.957****	**1.034**	**0.951**	**0.940**	**0.903****	0.874***	**0.847****	**0.845*****	**0.799*****	0.834***
RFARDI,K-fold	**0.961****	**1.024**	**0.944**	**0.928***	**0.901****	0.902***	**0.841****	**0.844*****	**0.813*****	**0.758*****
KRR-ARDI,POOS-CV	1,005	**1.067**	0,959	**0.912****	0,974	1,099	1,126	0.858***	**0.810*****	0.912*
KRR,ARDI,K-fold	**0.973**	**0.988**	**0.910***	**0.929***	**0.945**	1,017	0,97	**0.823*****	0.858***	0.808***
($B_1, \alpha = \hat{\alpha}$),POOS-CV	0,993	**1.122***	1,072	**0.969**	**0.940**	1,066	1,152	0,99	0.890**	0.873**
($B_1, \alpha = \hat{\alpha}$),K-fold	**0.921*****	**0.972**	0,973	**0.961**	0,991	0.871***	0.847***	0.922*	**0.822*****	**0.778*****
($B_1, \alpha = 1$),POOS-CV	0,997	**1.108***	1.071*	**1.003**	**0.929***	1,055	1,165	1	0,949	0.828***
($B_1, \alpha = 1$),K-fold	**0.961****	**1.024**	1,039	1,015	**0.975**	0,964	0,946	0,964	0.911**	0.802***
($B_1, \alpha = 0$),POOS-CV	1,003	**0.963**	**0.969**	**0.996**	0,982	1,067	**0.791*****	**0.851*****	0,962	0,916
($B_1, \alpha = 0$),K-fold	**0.934*****	**0.918****	**0.938**	**0.930**	**0.932**	0.871***	**0.793*****	**0.838*****	**0.788*****	0.771***
($B_2, \alpha = \hat{\alpha}$),POOS-CV	1,041	**1.099**	1.078*	1,061	0,985	1.189**	1.177*	1,015	0,963	0.923*
($B_2, \alpha = \hat{\alpha}$),K-fold	**0.963****	**0.975**	0,971	**0.992**	0,97	0,994	0.932*	0.928**	0.876***	0.821***
($B_2, \alpha = 1$),POOS-CV	1,026	**1.096**	1,075	1.103*	**0.922****	1.136*	1,157	1,005	0,991	0.897**
($B_2, \alpha = 1$),K-fold	**0.943*****	**0.916****	**0.948**	**0.954**	0,975	0.899***	0.811***	0.875***	**0.825*****	0.830***
($B_2, \alpha = 0$),POOS-CV	1,013	1.099**	1.102*	1.107*	**0.969**	1.091*	1.116*	1,026	0,942	0.870**
($B_2, \alpha = 0$),K-fold	0,981	**1.010**	1,017	1,029	1	0,971	1,008	0,943	0.871**	0.825***
($B_3, \alpha = \hat{\alpha}$),POOS-CV	1,038	**1.106***	1,042	**1.002**	**0.933***	1.128**	1,16	1,014	0.928*	0.852***
($B_3, \alpha = \hat{\alpha}$),K-fold	**0.945*****	**1.003**	1,082	1,038	**0.932**	0.922**	0,977	0.878**	0.823***	0.784***
($B_3, \alpha = 1$),POOS-CV	1.077*	**1.121***	1,034	1,033	0,974	1.175**	1,131	0,996	0.917**	0.901**
($B_3, \alpha = 1$),K-fold	**0.950****	**0.978**	1,074	1.088*	1,031	0.929**	0.883**	**0.841*****	0.846**	0.833**
($B_3, \alpha = 0$),POOS-CV	1,037	1.183***	1.148**	1.127*	1,008	1.166**	1.246**	1.129*	1,028	0.862***
($B_3, \alpha = 0$),K-fold	1.305**	1.517**	1.079**	1,008	1.110***	1,238	1.438**	1,003	0,96	1
SVR-ARDI,Lin,POOS-CV	**0.945****	**1.008**	1,081	**0.967**	**0.948**	0.907*	0.815***	0.869**	**0.799*****	**0.737*****
SVR-ARDI,Lin,K-fold	**0.911*****	**0.920***	1,015	**0.987**	0,984	**0.821*****	**0.761*****	0.866**	**0.805*****	**0.743*****
SVR-ARDI,RBF,POOS-CV	1,045	1.140**	0,995	**0.979**	**0.962**	1.158*	1.226**	0,952	0.907***	0.894***
SVR-ARDI,RBF,K-fold	1.072*	1.132***	**0.948**	**0.956**	**0.918*****	1.290***	1.205***	**0.824*****	0.843***	0.861***

*Note: The numbers represent the relative, with respect to AR,BIC model, root MSPE. Models retained in model confidence set are in bold, the minimum values are underlined, while ***, **, * stand for 1%, 5% and 10% significance of Diebold-Mariano test.*

Table 4: Unemployment rate: Relative Root MSPE

	Full Out-of-Sample					NBER Recessions Periods				
Models	h=1	h=3	h=9	h=12	h=24	h=1	h=3	h=9	h=12	h=24
Data-poor (H_t^-) models										
AR,BIC	1	1	1	1	1	1	1	1	1	1
AR,AIC	0,991	0,984	0,988	0.993***	1	0,958	0.960**	0.984*	1	1
AR,POOS-CV	0,989	1.042*	0,996	0,996	0,98	0,977	1.103*	0.981**	0,998	1,024
AR,K-fold	0,987	0,984	0.990*	0.994***	0.974**	0.982*	0.960**	1	1,001	1
RRAR,POOS-CV	1,005	1.050*	1,008	0,99	0.979*	1.083**	1.151**	1,006	1,006	0,997
RRAR,K-fold	0.984*	0.982*	0,994	0,996	0,993	0,983	0,984	0,992	1,001	1,029
RFAR,POOS-CV	0,999	1,011	0,987	1,002	1	1.107**	1,046	0.921**	0,962	0,997
RFAR,K-fold	0,982	0,986	0,977	0,987	1,003	0,966	0,971	0.911***	0.947**	0.946**
KRR-AR,POOS-CV	**0.982**	1,012	**0.892****	**0.837*****	0.821***	1,093	1,151	0.858***	**0.747*****	0.841***
KRR,AR,K-fold	**0.925*****	**0.862*****	**0.842*****	**0.828*****	**0.803*****	0.828***	0.804***	**0.772*****	**0.736*****	0.846***
SVR-AR,Lin,POOS-CV	1,037	1,068	1	0,987	0,97	1.237**	1.231***	1.064***	1.093***	1.165***
SVR-AR,Lin,K-fold	0,985	0.975*	0,992	0,991	0,991	0,988	1,008	1.064***	1.092***	1.205***
SVR-AR,RBF,POOS-CV	1.044*	1.087**	1.088***	1.036*	1.048***	1.208**	1.177**	1.120***	1.082***	1.119***
SVR-AR,RBF,K-fold	1.034**	1.114**	1.064**	1.052**	1,011	1.136***	1.248***	1.065***	1.074***	1.075***
Data-rich (H_t^+) models										
ARDI,BIC	**0.937****	**0.893****	0,938	0,939	0.875***	**0.690*****	**0.715*****	0.798***	**0.782*****	0.783***
ARDI,AIC	**0.933****	**0.878*****	0,928	0,953	0.893**	**0.720*****	**0.719*****	0.798***	**0.799*****	0.787***
ARDI,POOS-CV	**0.930*****	**0.918***	0,931	0,937	0.869***	**0.731*****	**0.675*****	0.821**	**0.785*****	0.768***
ARDI,K-fold	**0.947***	**0.893****	0,97	0,964	0.928*	**0.677*****	**0.665*****	0.805***	**0.807*****	0.772***
RRARDI,POOS-CV	**0.918*****	0,937	0,97	0,955	0.861***	**0.709*****	0.756***	0.890**	0.832***	**0.754*****
RRARDI,K-fold	**0.940****	**0.876****	**0.901***	**0.911***	0.897**	**0.744*****	**0.676*****	**0.788*****	0.819***	**0.745*****
RFARDI,POOS-CV	**0.925*****	**0.919****	**0.870*****	**0.879****	**0.782*****	**0.708*****	**0.739*****	**0.715*****	**0.736*****	0.785***
RFARDI,K-fold	**0.947****	**0.902*****	**0.857*****	**0.846*****	**0.776*****	**0.763*****	**0.751*****	**0.761*****	**0.725*****	**0.696*****
KRR-ARDI,POOS-CV	0,993	**0.990**	**0.883****	**0.835*****	**0.779*****	1,066	1,092	0.800***	**0.734*****	**0.766*****
KRR,ARDI,K-fold	**0.940*****	**0.882*****	**0.841*****	**0.810*****	**0.802*****	0,938	0.889*	0.791***	**0.739*****	0.841***
$(B_1, \alpha = \hat{\alpha})$,POOS-CV	**0.910*****	0.936**	0,945	0,975	0.929**	0.771**	0.834**	0.879**	0.841***	0.853***
$(B_1, \alpha = \hat{\alpha})$,K-fold	**0.920*****	**0.871*****	0.913*	0,933	0,967	**0.813***	0.738***	0.839***	**0.736*****	**0.712*****
$(B_1, \alpha = 1)$,POOS-CV	**0.928*****	0,975	1.092*	1,041	0.906***	0.861**	0.886**	0,986	0.906*	0.837***
$(B_1, \alpha = 1)$,K-fold	**0.912*****	**0.888*****	0,994	0,984	0.924**	0.798***	0.745***	0,906	0.834***	0.766***
$(B_1, \alpha = 0)$,POOS-CV	**0.922*****	0,947	0,982	0,961	0.934*	0.802**	0.809***	0.893*	0,897	0.843**
$(B_1, \alpha = 0)$,K-fold	**0.921*****	**0.876*****	**0.893****	0.911**	0.929*	0.824**	0.788**	0.820***	**0.771*****	**0.731*****
$(B_2, \alpha = \hat{\alpha})$,POOS-CV	**0.950****	**0.922****	0,971	1,002	0.856***	0,898	0.856**	0.902*	0.857***	0.831***
$(B_2, \alpha = \hat{\alpha})$,K-fold	**0.930*****	**0.867*****	0.898**	0.917**	0.892***	0.827***	0.773***	0.869***	0.831***	**0.750*****
$(B_2, \alpha = 1)$,POOS-CV	**0.944*****	**0.900*****	0,952	0,996	0.892***	0.787***	0.770***	0.889**	0.869**	0,978
$(B_2, \alpha = 1)$,K-fold	**0.937*****	**0.914*****	**0.849*****	0.915**	0.872***	**0.736*****	0.791***	0.837***	0.873**	0.842***
$(B_2, \alpha = 0)$,POOS-CV	**0.979**	1,001	1,031	1,011	0.946*	1,085	1,061	0,947	0.826***	0.835**
$(B_2, \alpha = 0)$,K-fold	**0.957****	**0.939****	0,988	1,004	0,951	0,991	1,006	0,97	0.875**	**0.763*****
$(B_3, \alpha = \hat{\alpha})$,POOS-CV	0,975	0,974	0,984	0.922*	0.882***	0,999	0,969	0.916**	0.893**	0.834**
$(B_3, \alpha = \hat{\alpha})$,K-fold	0.955***	**0.872*****	0,945	0,94	0.897**	0.882***	0.772***	0.840***	0.835***	0.829**
$(B_3, \alpha = 1)$,POOS-CV	**0.912*****	0,953	0,981	0,949	0.910**	0.793**	0.843***	0.895**	0.923*	0,919
$(B_3, \alpha = 1)$,K-fold	**0.937*****	0.923***	0,954	0,939	0.917*	0.811***	0.821***	0.852***	0.823***	0.838**
$(B_3, \alpha = 0)$,POOS-CV	1.197**	1.144***	1.318**	1.233**	1	1.470**	1.269***	1.288**	1,123	**0.787*****
$(B_3, \alpha = 0)$,K-fold	1.219*	1,039	1.108**	1.236**	1.082*	1,417	1,077	1,095	1,232	0,971
SVR-ARDI,Lin,POOS-CV	**0.919*****	**0.921***	**0.906***	0,947	0.911**	**0.748*****	0.734***	0.861**	**0.794*****	0.835***
SVR-ARDI,Lin,K-fold	**0.939*****	**0.864*****	**0.877****	**0.858*****	0.888***	0.777***	0.726***	**0.777*****	**0.750*****	0.791***
SVR-ARDI,RBF,POOS-CV	1,035	1.133**	1,018	0.945**	0.926***	1.182**	1.278***	1,017	0.885***	0.931*
SVR-ARDI,RBF,K-fold	1,032	1,032	0,954	0.921**	0.870***	1,102	1,058	0.894***	0.864***	0.874**

*Note: The numbers represent the relative, with respect to AR,BIC model, root MSPE. Models retained in model confidence set are in bold, the minimum values are underlined, while ***, **, * stand for 1%, 5% and 10% significance of Diebold-Mariano test.*

Table 5: Term spread: Relative Root MSPE

	Full Out-of-Sample					NBER Recessions Periods				
Models	h=1	h=3	h=9	h=12	h=24	h=1	h=3	h=9	h=12	h=24
Data-poor (H_t^-) models										
AR,BIC	1	**1.000**	**1.000**	**1.000**	**1.000**	1	1	1	1	1
AR,AIC	1.002*	**0.998**	1.053*	1.034**	1.041**	1,002	1,001	1,034	0,993	**0.972**
AR,POOS-CV	1,002	1.140*	**1.005**	**0.988**	1.035*	1	1,017	0.873**	0.872**	**0.973**
AR,K-fold	1.054*	1.065*	**0.998**	**1.000**	1.034*	1,041	1	0,907	1	0,983
RRAR,POOS-CV	1.012**	1.145*	**1.011**	1.016*	1,016	1,011	1,015	0.966**	0.987*	**0.930****
RRAR,K-fold	1.046*	**0.997**	1,043	**0.972**	1,021	1,025	0,997	0,995	0.820**	**0.954***
RFAR,POOS-CV	1,006	**0.899**	1.110**	0,996	1.086**	0,908	0,839	1,042	0.713**	1.048*
RFAR,K-fold	0,986	**0.929**	1.124***	1,014	1.083**	**0.892**	0,793	1,006	0.754*	1.053*
KRR-AR,POOS-CV	1.203*	**0.876**	**0.978**	**0.868****	**0.887*****	**0.894**	0.703*	0.776***	0.658**	**0.945**
KRR,AR,K-fold	1.203*	**0.867***	**0.936***	**0.871****	**0.894*****	**0.879**	0,708	0.791***	0.665**	**0.954**
SVR-AR,Lin,POOS-CV	0,999	**0.973**	**0.995**	1,025	**0.964***	0,975	0,806	0,989	0.922*	**0.998**
SVR-AR,Lin,K-fold	0,995	**0.916**	**0.990**	**0.984**	**0.955****	**0.966**	0,706	0,998	0,972	**0.949****
SVR-AR,RBF,POOS-CV	1,019	**0.853***	1,055	**0.928**	**0.953**	**0.786**	0,739	0,888	0.688**	**0.885*****
SVR-AR,RBF,K-fold	**1.005**	**0.879**	1.161***	0,998	1,052	**0.786**	**0.668**	0,957	0.732*	**0.990**
Data-rich (H_t^+) models										
ARDI,BIC	**0.953**	0,971	**0.979**	**0.930**	**0.892*****	0,921	0,9	0.790***	**0.633*****	1,049
ARDI,AIC	**0.970**	**0.956**	**1.019**	**0.944**	**0.917****	**0.929**	0,867	0.814***	0.647***	1,076
ARDI,POOS-CV	**0.934**	1,039	1,063	1,036	**1.000**	**0.900**	0,939	0,973	0,868	1,105
ARDI,K-fold	**0.963**	**0.936**	**0.980**	**0.934**	**0.955**	**0.892**	0,897	0.788***	**0.647*****	1,114
RRARDI,POOS-CV	0,972	1,022	1.095*	1,003	1,025	0,948	0,925	0,998	0.827*	1.183*
RRARDI,K-fold	0,97	0,99	**0.983**	**0.955**	**0.962**	0,934	0,967	**0.777*****	0.682**	1,102
RFARDI,POOS-CV	1,002	**0.832***	**0.956**	**0.856****	**0.919****	**0.817**	**0.684**	**0.790*****	**0.626*****	**0.965**
RFARDI,K-fold	1,061	**0.872***	1,014	**0.887***	**0.950**	**0.863**	0,736	0.856*	0.652**	**0.961**
KRR-ARDI,POOS-CV	1.311***	**0.908**	**1.018**	**0.852****	**0.907*****	**0.904**	**0.663***	**0.716*****	**0.603*****	**0.931**
KRR,ARDI,K-fold	1.376**	**0.947**	**0.968**	**0.846****	**0.877*****	**0.862**	**0.686**	**0.766*****	**0.575*****	**0.930***
$(B_1, \alpha = \hat{\alpha})$,POOS-CV	1,043	0,988	1.164***	1.085*	**0.924****	0,979	0,907	1,075	0,951	1,026
$(B_1, \alpha = \hat{\alpha})$,K-fold	0,995	0,981	1.087*	1,028	**0.939**	0,94	0,929	0,91	0.789**	1,003
$(B_1, \alpha = 1)$,POOS-CV	1,068	**0.954**	1.185**	1,074	**0.991**	**0.915**	0,86	1.399*	0.807*	1.104**
$(B_1, \alpha = 1)$,K-fold	1,041	**0.937**	**0.961**	**0.982**	**0.943****	0,937	0,854	0.789***	0.664**	1,01
$(B_1, \alpha = 0)$,POOS-CV	1.426**	1.138*	1.215***	1,052	**0.967**	1.473*	1,018	1.283*	0.808*	1,07
$(B_1, \alpha = 0)$,K-fold	1.359*	1,041	1,038	**0.980**	**0.905****	1,404	1,044	0.869*	0.737**	**0.978**
$(B_2, \alpha = \hat{\alpha})$,POOS-CV	0,987	0,979	1.149***	1,01	1,008	**0.879**	0,874	0,97	0.731**	1,09
$(B_2, \alpha = \hat{\alpha})$,K-fold	0,979	**0.959**	**1.006**	**0.965**	**0.937****	**0.872**	0,885	0.819**	0.666**	**0.912****
$(B_2, \alpha = 1)$,POOS-CV	1.121*	1,086	**1.008**	1	**0.965**	1,002	1,078	0.840***	0.766**	**0.955**
$(B_2, \alpha = 1)$,K-fold	1,006	**0.916**	**0.983**	**0.918**	**0.901*****	**0.880**	0,815	0.811***	0.648**	1,024
$(B_2, \alpha = 0)$,POOS-CV	1.136**	1,018	1.150***	1,045	**1.010**	0,971	0,908	1,086	0,838	**0.958**
$(B_2, \alpha = 0)$,K-fold	1.146**	**0.952**	1,042	**0.961**	**0.954**	0,946	0,879	0.887*	0.789*	**0.900****
$(B_3, \alpha = \hat{\alpha})$,POOS-CV	1.183*	1.145**	1.157***	1,034	**0.949**	0,923	1,012	0.826***	**0.623*****	0,993
$(B_3, \alpha = \hat{\alpha})$,K-fold	1.176*	1,074	1.167***	1,048	**0.929***	0,956	1,023	0.812**	0.754**	**0.975**
$(B_3, \alpha = 1)$,POOS-CV	1.305***	1.302***	1.140***	0,981	**0.948**	1,042	1,071	0.855**	0.656***	1,009
$(B_3, \alpha = 1)$,K-fold	1.222*	1,086	1.093**	1,039	**0.972**	1,004	0,889	0,913	0.736**	**0.976**
$(B_3, \alpha = 0)$,POOS-CV	1.775***	1.335***	1.208***	1.362*	**0.990**	1.379**	0,977	1,093	1,244	0,968
$(B_3, \alpha = 0)$,K-fold	1.644***	1.189**	1.153***	1,019	**0.997**	1.565***	0,795	0,907	0.737**	**0.980**
SVR-ARDI,Lin,POOS-CV	**0.927**	**0.988**	**1.024**	**0.941**	**0.962**	**0.888**	0,953	0.856**	0.729**	1.114*
SVR-ARDI,Lin,K-fold	**0.975**	**0.924**	**1.007**	**0.928**	**0.899*****	**0.944**	0,851	0.842***	0.681**	1,03
SVR-ARDI,RBF,POOS-CV	1.733***	1,031	1,061	**0.935**	**0.900*****	1.377**	0,799	**0.761*****	0.651**	**0.931**
SVR-ARDI,RBF,K-fold	1.692***	**1.002**	1,034	**0.933**	**0.948***	1.377**	0,802	0.795**	**0.648****	**0.976**

*Note: The numbers represent the relative, with respect to AR,BIC model, root MSPE. Models retained in model confidence set are in bold, the minimum values are underlined, while ***, **, * stand for 1%, 5% and 10% significance of Diebold-Mariano test.*

Table 6: CPI Inflation: Relative Root MSPE

	Full Out-of-Sample					NBER Recessions Periods				
Models	h=1	h=3	h=9	h=12	h=24	h=1	h=3	h=9	h=12	h=24
Data-poor (H_t^-) models										
AR,BIC	**1.000**	**1.000**	**1.000**	1	1	1	**1.000**	1	1	1
AR,AIC	**0.965*****	**1.000**	**1.000**	**0.969****	0.968**	0,998	**0.998**	1	0,998	0,972
AR,POOS-CV	**0.977**	**0.997**	**0.977**	**0.957****	**0.943*****	0,978	**0.992**	1	0,979	0,977
AR,K-fold	**0.966*****	**0.986**	**0.974***	**0.967****	0.970**	0,998	**0.975****	1	1,007	0,992
RRAR,POOS-CV	**0.962*****	**1.003**	**0.978**	**0.976**	**0.946*****	0.963**	**0.990**	0,998	0,977	0,971
RRAR,K-fold	**1.020***	**1.015**	1.182*	**0.981**	0,986	0,999	**1.008**	1,077	0,999	1.002*
RFAR,POOS-CV	**1.012**	1.165**	1.285***	1.279***	1.274***	0,931	1,147	1,08	1.089**	1.256***
RFAR,K-fold	1,021	1.188***	1.269***	1.285***	1.310***	0,959	1.159*	1,042	1.125**	1.266**
KRR-AR,POOS-CV	**0.993**	1.144**	1.288***	1.330**	1.263**	**0.884***	1,143	0,944	1,053	1.279**
KRR,AR,K-fold	**1.004**	**1.116***	1.226**	1.234**	1,163	0.893*	1,14	0,911	0,952	0,93
SVR-AR,Lin,POOS-CV	**1.010**	1.191***	1.376***	1.365***	1.337***	0.902*	1,13	1.148*	1.146**	1.222**
SVR-AR,Lin,K-fold	**1.003**	1.201***	1.354***	1.355***	1.321***	0.901*	1.139*	1.123*	1.149**	1.220**
SVR-AR,RBF,POOS-CV	1.053*	1.260***	1.539***	1.515***	1.471***	0,905	1.191*	1,122	1.166**	1.337***
SVR-AR,RBF,K-fold	1,039	1.253***	1.462***	1.488***	1.416***	0,906	1.185*	1.119*	1.222**	1.263*
Data-rich (H_t^+) models										
ARDI,BIC	**0.958**	**1.005**	**1.022**	**1.006**	1,004	0,928	**0.938**	**0.745****	**0.688****	0.568**
ARDI,AIC	**0.975**	**1.014**	**1.033**	1,04	1,033	0,958	**0.923**	**0.740****	**0.661****	**0.474****
ARDI,POOS-CV	**0.979**	**1.113**	**1.005**	1,041	1,014	0,963	1,041	**0.789****	**0.711*****	0.530**
ARDI,K-fold	**0.945**	**1.015**	**1.008**	1,045	1,009	0,906	**0.927**	**0.753****	**0.662****	**0.456****
RRARDI,POOS-CV	**0.975**	**1.099**	**0.984**	1,075	1,058	0,937	1,034	**0.789****	**0.688*****	0.504**
RRARDI,K-fold	**0.943***	**1.012**	**1.046**	**0.935**	1,044	0.894**	**0.913**	**0.767***	**0.658*****	**0.465****
RFARDI,POOS-CV	**1.025**	1.165**	1.254***	1.261***	1.179**	0,936	1,143	0,982	0,948	0,985
RFARDI,K-fold	**1.024**	1.178**	1.267***	1.274***	1.214**	0,955	1,176	0,976	0,95	1,014
KRR-ARDI,POOS-CV	**0.978**	1.149**	1.300***	1.233**	1.161*	0.902*	1,131	0,996	0,968	0,917
KRR,ARDI,K-fold	**1.004**	1.137**	1.271***	1.244**	1,157	0.904*	1,117	0,984	0.882*	0,954
$(B_1, \alpha = \hat{\alpha})$,POOS-CV	**0.954***	**1.062***	1.112**	1.087*	1,086	0.873*	1,053	0,924	0.824**	0,804
$(B_1, \alpha = \hat{\alpha})$,K-fold	**0.937****	**1.047**	1.136**	1,062	1,125	**0.863***	1,074	0,986	0.877*	0,812
$(B_1, \alpha = 1)$,POOS-CV	**0.977**	**1.136***	1.185***	1.096*	1.217**	0.880*	1,213	1,04	0,921	0,873
$(B_1, \alpha = 1)$,K-fold	**0.978**	**1.138***	1.154**	1.091*	1,093	0,915	1,222	1,075	0,96	0.592*
$(B_1, \alpha = 0)$,POOS-CV	**0.977**	**1.052**	1.612*	1.250***	1.266**	**0.904**	**1.005**	2,139	1,067	0,786
$(B_1, \alpha = 0)$,K-fold	**0.948**	1,315	1.109**	1.359**	1,412	**0.839***	1,57	0,961	1,355	0,76
$(B_2, \alpha = \hat{\alpha})$,POOS-CV	**0.970**	**1.051***	1.089*	1.104*	1.122*	0.891*	1,027	0,885	0,89	0,844
$(B_2, \alpha = \hat{\alpha})$,K-fold	**0.977**	**1.039**	1.086*	1,079	1.112*	0,904	1,033	0,891	0,901	0,871
$(B_2, \alpha = 1)$,POOS-CV	**0.997**	**1.075****	1.106*	1.115*	1.151*	0,913	1,066	0,902	0,923	0,888
$(B_2, \alpha = 1)$,K-fold	**0.973**	**1.053***	1.137**	1,099	1.191**	0,919	1,022	0,886	0,921	0,958
$(B_2, \alpha = 0)$,POOS-CV	**1.001**	1.104**	1.136***	1.186***	1.327***	0,935	1,093	0,988	1,071	0,889
$(B_2, \alpha = 0)$,K-fold	**0.993**	1.085**	1.144***	1.132**	1.216***	0,923	1,076	0,965	1,045	0,896
$(B_3, \alpha = \hat{\alpha})$,POOS-CV	**0.973**	1.128***	1.211***	1.274**	1.225***	**0.819****	1,09	1,023	1.095*	0,918
$(B_3, \alpha = \hat{\alpha})$,K-fold	**0.976**	**1.098*****	1.231***	1.211***	1.125**	**0.837****	1,061	1,108	1,013	0,864
$(B_3, \alpha = 1)$,POOS-CV	**0.999**	**1.117*****	1.219***	1.313***	1.232***	**0.830****	1,076	1,08	0,976	0,876
$(B_3, \alpha = 1)$,K-fold	**0.990**	1.134***	1.260***	1.241***	1.176***	**0.853***	1,054	1.132*	1,069	0,862
$(B_3, \alpha = 0)$,POOS-CV	**1.085**	1.400**	1.330***	1.468***	1.267***	**0.900**	1,056	1.173**	1.490**	0,831
$(B_3, \alpha = 0)$,K-fold	**0.976**	1.276**	1.328***	1.392***	1.244***	**0.864***	1,064	1.348**	1.439***	0,858
SVR-ARDI,Lin,POOS-CV	1,019	1.134**	1.200***	1.326**	1.189*	0,969	1,055	1	0,974	0,735
SVR-ARDI,Lin,K-fold	**0.964**	1.108**	1.191**	1.200**	1,167	**0.865****	1,084	1,018	0.769**	0,689
SVR-ARDI,RBF,POOS-CV	1,04	1.263***	1.534***	1.515***	1.461***	0,906	1,153	1.119*	1.195**	1.243*
SVR-ARDI,RBF,K-fold	1,037	1.253***	1.522***	1.500***	1.433***	0,907	1,155	1.124*	1.187*	1.252*

*Note: The numbers represent the relative, with respect to AR,BIC model, root MSPE. Models retained in model confidence set are in bold, the minimum values are underlined, while ***, **, * stand for 1%, 5% and 10% significance of Diebold-Mariano test.*

Table 7: Housing starts: Relative Root MSPE

	Full Out-of-Sample					NBER Recessions Periods				
Models	h=1	h=3	h=9	h=12	h=24	h=1	h=3	h=9	h=12	h=24
Data-poor (H_t^-) models										
AR,BIC	1	**1.000**	**1.000**	**1.000**	**1.000**	1	**1.000**	1	1	1
AR,AIC	**0.995**	**0.987**	**1.048***	**1.008**	**1.041***	**0.956*****	**0.976***	1,007	0,97	**0.955**
AR,POOS-CV	1.010*	**1.002**	**1.064****	**1.011**	**1.049*****	0.989*	**1.011***	1.025**	0,991	0,981
AR,K-fold	**0.992**	**0.986**	**1.029**	**0.997**	**1.031**	**0.957*****	**0.973****	0,999	0,954	**0.925****
RRAR,POOS-CV	1.028**	**1.022**	**0.983**	**0.905***	**0.954**	0,987	**1.018**	**0.943**	**0.912***	1,05
RRAR,K-fold	**0.990**	**0.986***	**1.035**	**0.992**	**1.024**	**0.935*****	**0.974****	1,013	0,965	**0.924****
RFAR,POOS-CV	1.231**	1.267**	1.260**	1.271**	1.086**	1.463*	1.588**	1,058	1,01	**0.941***
RFAR,K-fold	1.179*	1.255**	1.314**	1.264**	**1.058***	1.452*	1.586**	1.073*	0,973	**0.883****
KRR-AR,POOS-CV	2.008***	1.811***	1.567***	1.384***	**0.980****	2.141***	1.927***	1.088*	0,944	0.981*
KRR,AR,K-fold	1.293**	**1.325***	1.430**	1.165**	**1.001**	1.684**	1.778**	1.107**	0,937	0.974**
SVR-AR,Lin,POOS-CV	1.220**	1.411**	1.307**	1.284***	**1.040**	1.286*	1.600*	1.056*	1.108**	1.114*
SVR-AR,Lin,K-fold	1.219**	1.501**	1.313***	1.240***	**1.011**	1.449*	1.573*	1,037	1,05	1,024
SVR-AR,RBF,POOS-CV	1.281***	**1.315****	1.336**	1.251***	**1.014**	1.517**	1.627**	1.092*	1	0,962
SVR-AR,RBF,K-fold	1.155***	1.443***	1.421***	1.338***	1.080***	1.265**	1.624*	1.088*	1,043	1,011
Data-rich (H_t^+) models										
ARDI,BIC	**0.951****	**0.958**	**1.034**	**1.026**	**1.045**	**0.929**	1.104*	0,96	**0.862****	1,064
ARDI,AIC	**0.979**	**0.975**	**1.018**	**1.034**	1.078**	**0.958**	1.151**	**0.936**	0.905*	1,036
ARDI,POOS-CV	0,996	**0.981**	**1.077***	**1.040**	1.091**	0,962	1.133*	0,968	0,915	0,987
ARDI,K-fold	**0.979**	**0.989**	**1.028**	**1.037**	1.075***	0,977	1.154**	0,953	**0.885***	1,041
RRARDI,POOS-CV	1,004	**1.044**	**1.042**	**1.012**	**1.004**	**0.939***	1.216**	**0.942***	**0.907**	1.096*
RRARDI,K-fold	**0.970**	**0.981**	**1.043**	**1.034**	**0.964**	**0.932**	1.136**	**0.915****	**0.867****	1,01
RFARDI,POOS-CV	1.224*	**1.238***	**1.174****	**1.141***	**1.004**	1.608*	1.602**	1,019	0,926	0,969
RFARDI,K-fold	1,131	**1.216***	1.209**	1.128**	**0.973**	1.401*	1.615**	1,046	0.929*	**0.940**
KRR-ARDI,POOS-CV	1.346***	1.456**	1.348**	1.242**	**1.002**	1.562**	1.887**	1,028	0,98	**0.963*****
KRR,ARDI,K-fold	1.466***	1.410**	1.474**	1.300**	**1.022**	1.856***	1.907**	1.106**	0,981	1
$(B_1, \alpha = \hat{\alpha})$,POOS-CV	1.102**	**1.110**	1.158**	1.126**	**1.032**	1.197**	1.387***	1,029	0,947	1,001
$(B_1, \alpha = \hat{\alpha})$,K-fold	1.057*	**1.050**	1.135**	1.110**	**1.009**	1.124*	1.281***	0,987	0.929**	**0.980**
$(B_1, \alpha = 1)$,POOS-CV	**0.960**	**1.038**	1.274***	1.158***	**0.989**	**0.940**	1.317**	1,017	0.901**	1,009
$(B_1, \alpha = 1)$,K-fold	**0.950***	**1.022**	1.203***	1.163**	**0.981**	**0.945**	1.285**	0,971	**0.890****	1,019
$(B_1, \alpha = 0)$,POOS-CV	1.191***	1.214**	1.136**	1.203**	**1.055***	1.227*	1.594***	0,975	0.914*	0,99
$(B_1, \alpha = 0)$,K-fold	1.091*	**1.155**	**1.104***	**1.085**	**1.044**	1,198	1.536**	**0.946**	**0.863****	0,982
$(B_2, \alpha = \hat{\alpha})$,POOS-CV	1,053	**1.092**	**1.040**	**1.054**	**1.018**	1.170*	1.368***	0,969	0,936	0.958*
$(B_2, \alpha = \hat{\alpha})$,K-fold	1.058*	**1.020**	**1.064**	**1.033**	1.057**	1.183**	1.253**	0,989	0.909*	0,999
$(B_2, \alpha = 1)$,POOS-CV	1,015	**1.009**	**1.079**	1.101**	**1.024**	1,036	1.234**	0,962	0.904**	**0.909****
$(B_2, \alpha = 1)$,K-fold	0,991	**0.994**	**1.019**	**1.009**	**1.013**	0,981	1.226**	**0.931**	**0.841*****	1,005
$(B_2, \alpha = 0)$,POOS-CV	1.224***	1.293**	1.202***	1.168**	**1.002**	1.378**	1.639**	1,007	**0.887*****	**0.897****
$(B_2, \alpha = 0)$,K-fold	1.147***	**1.175**	**1.070**	**1.006**	**0.995**	1.261***	1.589**	0,973	**0.884*****	0,955
$(B_3, \alpha = \hat{\alpha})$,POOS-CV	1.380***	1.283**	1.223**	1.147**	**1.046**	1.531***	1.566***	1.089*	0,978	0.943*
$(B_3, \alpha = \hat{\alpha})$,K-fold	1.420***	1.248***	1.186**	1.176**	**1.049**	1.370***	1.436***	1,027	0,972	0,993
$(B_3, \alpha = 1)$,POOS-CV	1.132**	**1.033**	1.181**	1.145**	**1.036**	1.175*	1.179*	**0.933***	**0.906****	**0.926***
$(B_3, \alpha = 1)$,K-fold	1.085**	**1.061**	1.169**	1.159**	**1.081**	1,013	1.257**	0,954	0,923	0,998
$(B_3, \alpha = 0)$,POOS-CV	1.595***	1.468***	1.352***	1.204**	**1.041****	1.588***	1.622***	**0.956**	**0.908***	1,013
$(B_3, \alpha = 0)$,K-fold	1.505***	1.367***	1.375***	1.252**	**0.984**	1.634***	1.490***	1.104*	1,037	0,952
SVR-ARDI,Lin,POOS-CV	1.231***	1.207**	1.179**	1.152**	1.134***	1.333**	1.386***	1,087	1,026	1,023
SVR-ARDI,Lin,K-fold	1.208***	1.442**	1.107*	1,101	**1.074****	1,097	1.222**	**0.925***	**0.846****	1,007
SVR-ARDI,RBF,POOS-CV	1.621***	1.778***	1.425***	1.263**	**0.985**	1.745***	1.956**	1,077	0,919	1
SVR-ARDI,RBF,K-fold	1.675***	1.802***	1.467***	1.267**	**1.023**	1.808***	1.993**	1,079	0.924*	0,989

*Note: The numbers represent the relative, with respect to AR,BIC model, root MSPE. Models retained in model confidence set are in bold, the minimum values are underlined, while ***, **, * stand for 1%, 5% and 10% significance of Diebold-Mariano test.*

B Robustness of Treatment Effects Graphs

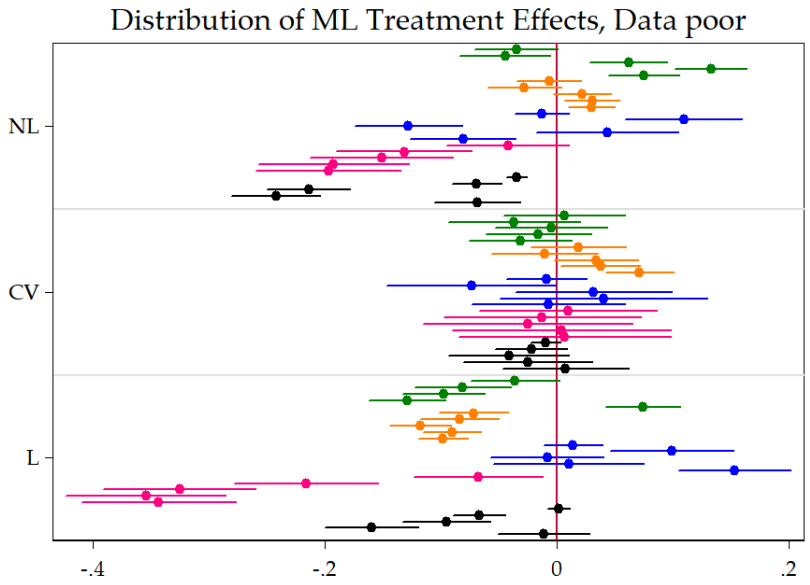

Figure 12: This figure plots the distribution of $\dot{\alpha}_F^{(h,v)}$ from equation 11 done by (h, v) subsets. The subsample under consideration here is **data-poor models**. The unit of the x-axis are improvements in OOS R^2 over the basis model. Variables are INDPRO, UNRATE, SPREAD, INF and HOUST. Within a specific color block, the horizon increases from $h = 1$ to $h = 24$ as we are going down. SEs are HAC. These are the 95% confidence bands.

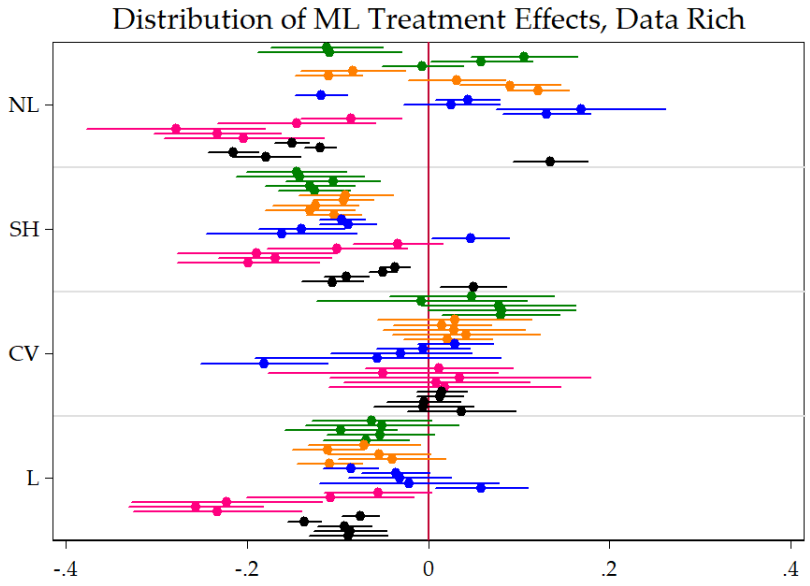

Figure 13: This figure plots the distribution of $\dot{\alpha}_F^{(h,v)}$ from equation 11 done by (h, v) subsets. The subsample under consideration here is **data-rich models**. The unit of the x-axis are improvements in OOS R^2 over the basis model. Variables are INDPRO, UNRATE, SPREAD, INF and HOUST. Within a specific color block, the horizon increases from $h = 1$ to $h = 24$ as we are going down. SEs are HAC. These are the 95% confidence bands.

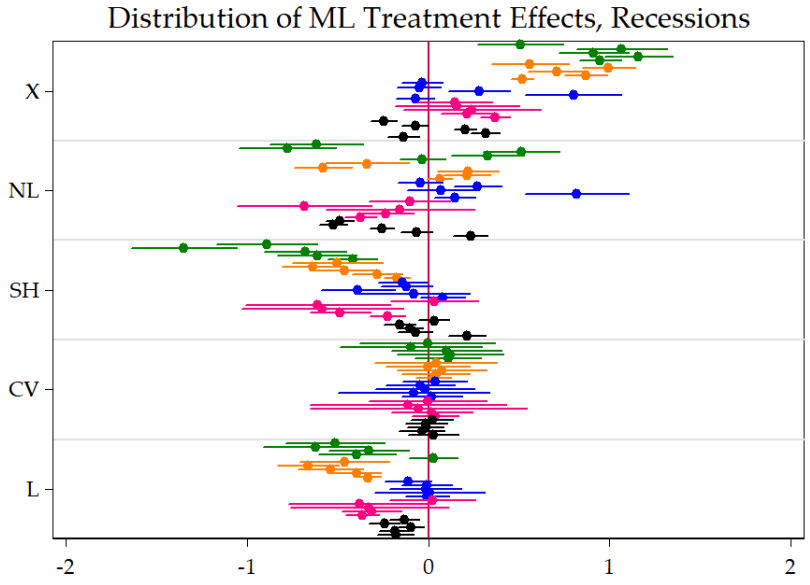

Figure 14: This figure plots the distribution of $\dot{\alpha}_F^{(h,v)}$ from equation 11 done by (h, v) subsets. The subsample under consideration here are **recessions**. The unit of the x-axis are improvements in OOS R^2 over the basis model. Variables are INDPRO, UNRATE, SPREAD, INF and HOUST. Within a specific color block, the horizon increases from $h = 1$ to $h = 24$ as we are going down. SEs are HAC. These are the 95% confidence bands.

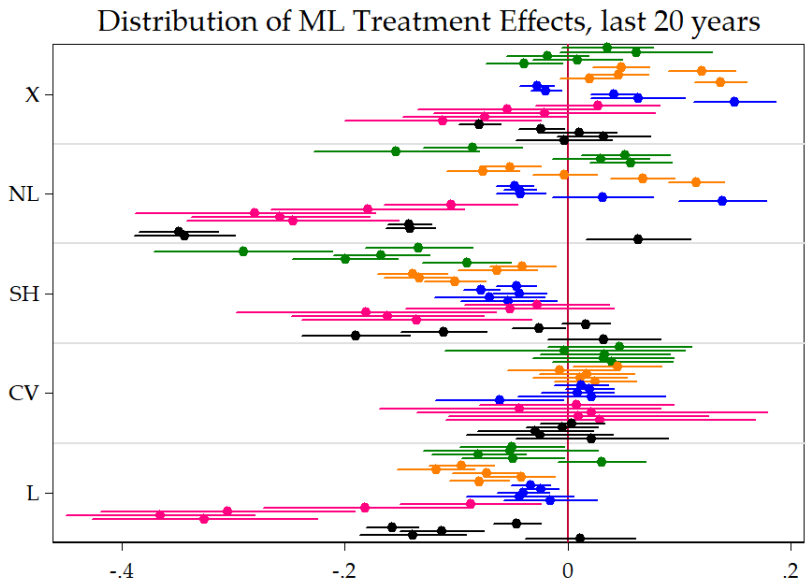

Figure 15: This figure plots the distribution of $\grave{\alpha}_F^{(h,v)}$ from equation 11 done by (h, v) subsets. The subsample under consideration here are **the last 20 years**. The unit of the x-axis are improvements in OOS R^2 over the basis model. Variables are INDPRO, UNRATE, SPREAD, INF and HOUST. Within a specific color block, the horizon increases from $h = 1$ to $h = 24$ as we are going down. SEs are HAC. These are the 95% confidence bands.

C Additional Graphs

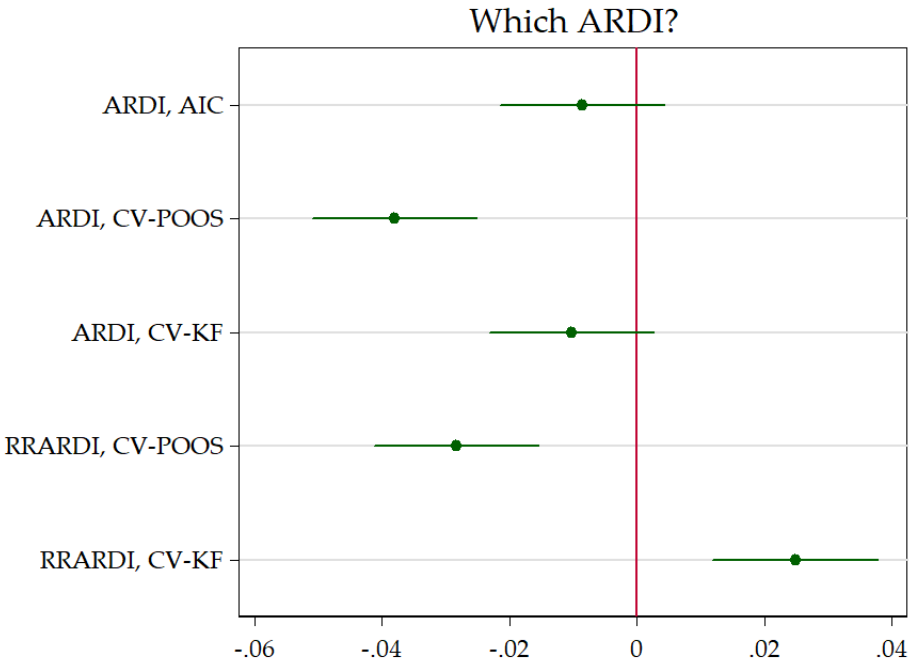

Figure 16: This graph displays the marginal improvements of different ARDIs with respect to the baseline ARDI-BIC. The unit of the x-axis are improvements in OOS R^2 over the basis model. SEs are HAC. These are the 95% confidence bands.

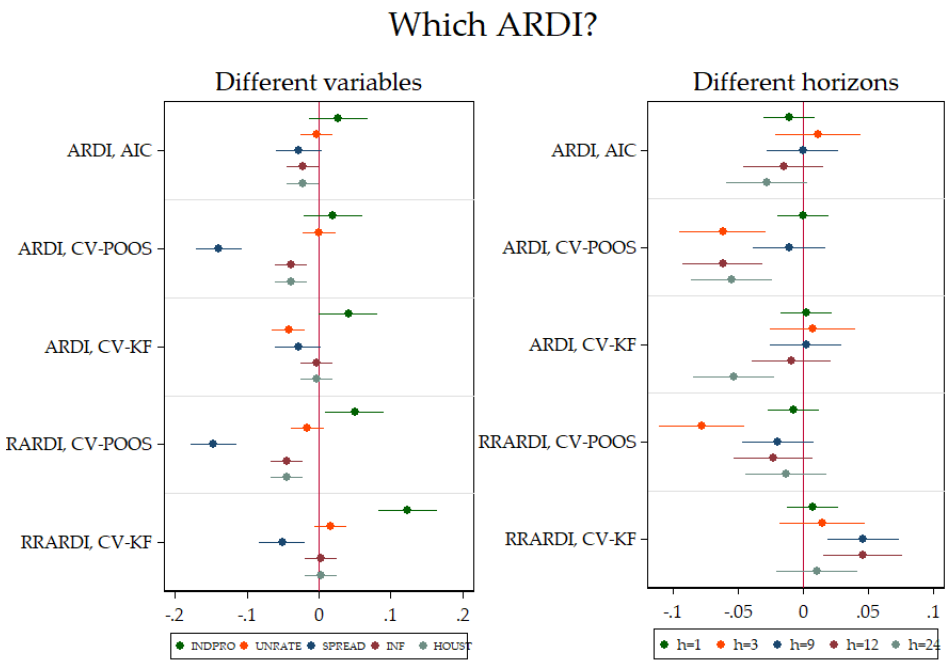

Figure 17: This graph displays the marginal improvements of different ARDIs with respect to the baseline ARDI-BIC by variables and horizons. The unit of the x-axis are improvements in OOS R^2 over the basis model. SEs are HAC. These are the 95% confidence bands.

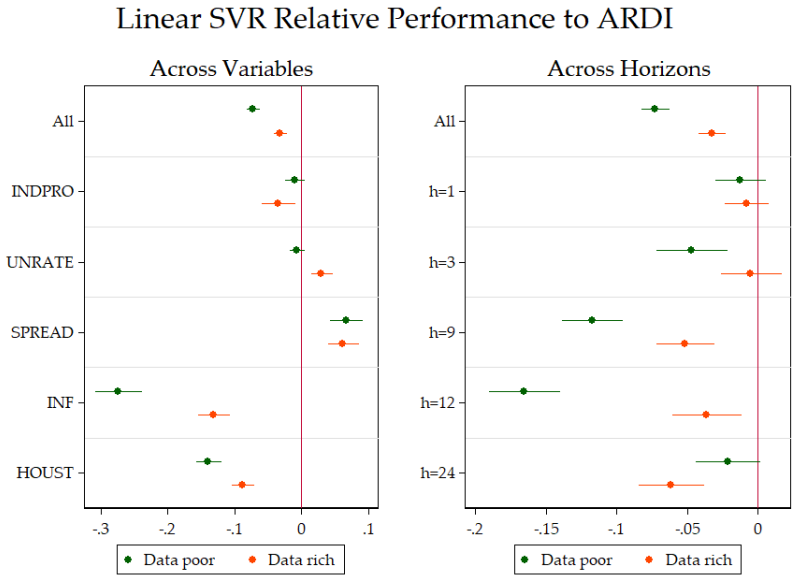

Figure 18: This graph display the marginal (un)improvments by variables and horizons to opt for the SVR in-sample loss function in comparing the data-poor and data-rich environments for linear models. The unit of the x-axis are improvements in OOS R^2 over the basis model. SEs are HAC. These are the 95% confidence bands.

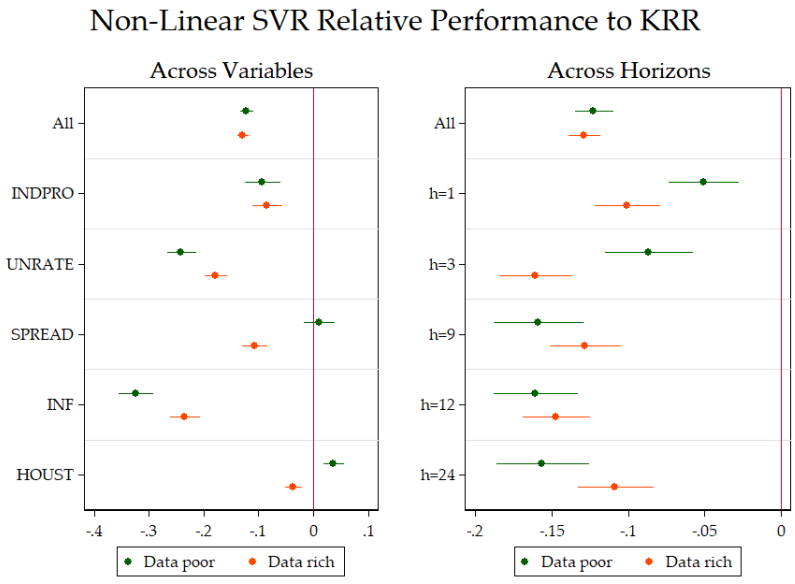

Figure 19: This graph display the marginal (un)improvments by variables and horizons to opt for the SVR in-sample loss function in comparing the data-poor and data-rich environments for non-linear models. The unit of the x-axis are improvements in OOS R^2 over the basis model. SEs are HAC. These are the 95% confidence bands.

D Detailed Implementation of Cross-validations

All of our models involve some kind of hyperparameter selection prior to estimation. To curb the overfitting problem, we use two distinct methods that we refer to loosely as cross-validation methods. To make it feasible, we optimize hyperparameters every 24 months as the expanding window grows our in-sample set. The resulting optimization points are the same across all models, variables and horizons considered. In all other periods, hyperparameter values are frozen to the previous values and models are estimated using the expanded in-sample set to generate forecasts.

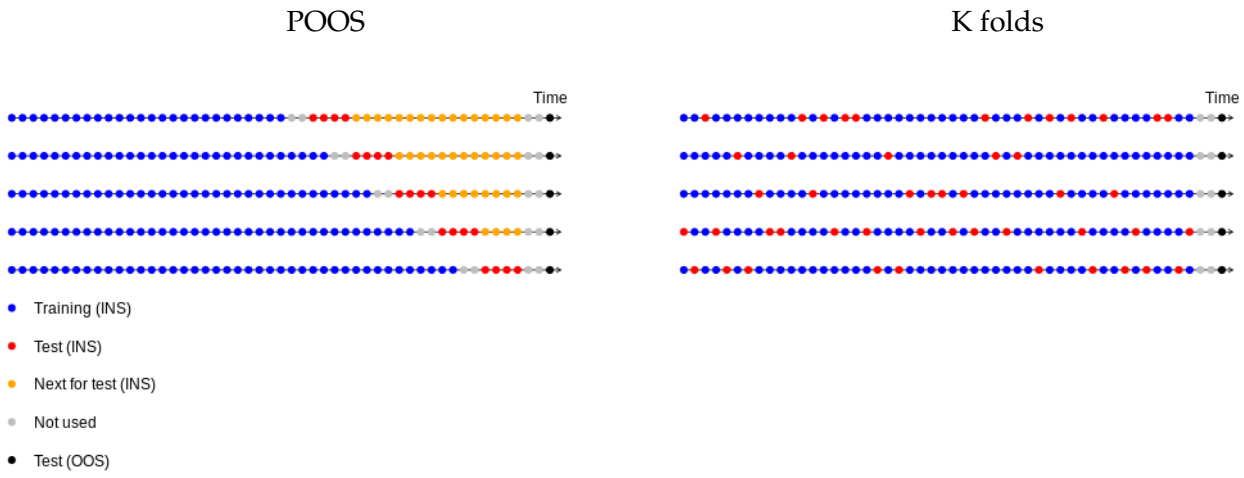

Figure 20: Illustration of cross-validation methods

Notes: Figures are drawn for 3 months forecasting horizon and depict the splits performed in the in-sample set. The pseudo-out-of-sample observation to be forecasted here is shown in black.

The first cross-validation method we consider mimics in-sample the pseudo-out-of-sample comparison we perform across model. For each set of hyperparameters considered, we keep the last 60 months as a comparison window. Models are estimated every 12 months, but the training set is gradually expanded to keep the forecasting horizon intact. This exercice is thus repeated 5 times. Figure 20 shows a toy example with smaller jumps, a smaller comparison window and a forecasting horizon of 3 months, hence the gaps. Once hyperparameters have been selected, the model is estimated using the whole in-sample set and used to make a forecast in the pseudo-out-of-sample window we use to compare all models (the black dot in the figure). This approach is a compromise between two methods used to evaluate time series models detailed in Tashman (2000), rolling-origin recalibration and rolling-origin updating.[14] For a simulation study of various cross-validation methods in a

[14] In both cases, the last observation (the origin of the forecast) of the training set is rolled forward. However, in the first case, hyperparameters are recalibrated and, in the second, only the information set is updated.

time series context, including the rolling-origin recalibration method, the reader is referred to Bergmeir and Benítez (2012). We stress again that the compromise is made to bring down computation time.

The second cross-validation method, K-fold cross-validation, is based on a re-sampling scheme (Bergmeir et al. (2018)). We chose to use 5 folds, meaning the in-sample set is randomly split into five disjoint subsets, each accounting on average for 20 % of the in-sample observations. For each one of the 5 subsets and each set of hyperparameters considered, 4 subsets are used for estimation and the remaining corresponding observations of the in-sample set used as a test subset to generate forecasting errors. This is illustrated in figure 20 where each subsets is illustrated by red dots on different arrows.

Note that the average mean squared error in the test subset is used as the performance metric for both cross-validation methods to perform hyperparameter selection.

E Forecasting models in detail

E.1 Data-poor (H_t^-) models

In this section we describe forecasting models that contain only lagged values of the dependent variable, and hence use a small amount of predictors, H_t^-.

Autoregressive Direct (AR) The first univariate model is the so-called *autoregressive direct* (AR) model, which is specified as:

$$y_{t+h}^{(h)} = c + \rho(L)y_t + e_{t+h}, \quad t = 1, \dots, T,$$

where $h \geq 1$ is the forecasting horizon. The only hyperparameter in this model is p_y, the order of the lag polynomial $\rho(L)$. The optimal p is selected in four ways: (i) Bayesian Information Criterion (AR,BIC); (ii) Akaike Information Criterion (AR,AIC); (iii) Pseudo-out-of-sample cross validation (AR,POOS-CV); and (iv) K-fold cross validation (AR,K-fold). The lag order is selected from the following subset $p_y \in \{1, 3, 6, 12\}$. Hence, this model enters the following categories: linear g function, no regularization, in-sample and cross-validation selection of hyperparameters and quadratic loss function.

Ridge Regression AR (RRAR) The second specification is a penalized version of the previous AR model that allows potentially more lagged predictors by using Ridge regression. The model is written as in (E.1), and the parameters are estimated using Ridge penalty. The Ridge hyperparameter is selected with two cross validation strategy, which gives two models: RRAR,POOS-CV and RRAR,K-fold. The lag order is selected from the following subset

$p_y \in \{1, 3, 6, 12\}$ and for each of these value we choose the Ridge hyperparameter. This model creates variation on following axes: linear g, Ridge regularization, cross-validation for tuning parameters and quadratic loss function.

Random Forests AR (RFAR) A popular way to introduce nonlinearities in the predictive function g is to use a tree method that splits the predictors space in a collection of dummy variables and their interactions. Since a standard tree regression is prompt to the overfitt, we use instead the random forest approach described in Section 3.1.2. As in the literature we set the number of predictors in each tree to one third of all the predictors and the observations in each set are sampled with replacement to get as many observations in the trees as in the full sample. The number of lags of y_t, is chosen from the subset $p_y \in \{1, 3, 6, 12\}$ with cross-validation while the number of trees is selected internally with out-of-bag observations. This model generates nonlinear approximation of the optimal forecast, without regularization, using both CV techniques with the quadratic loss function: RFAR,K-fold and RFAR,POOS-CV.

Kernel Ridge Regression AR (KRRAR) This specification adds a nonlinear approximation of the function g by using the Kernel trick as in Section 3.1.1. The model is written as in (13) and (14) but with the autoregressive part only

$$y_{t+h} = c + g(Z_t) + \varepsilon_{t+h},$$
$$Z_t = \left[\{y_{t-0}\}_{j=0}^{p_y}\right],$$

and the forecast is obtained using the equation (16). The hyperparameters of Ridge and radial basis function kernel are selected by two cross-validation procedure, which gives two forecasting specifications: KRRAR,POOS-CV and KRRAR,K-fold. Z_t consist of y_t and its p_y lags, $p_y \in \{1, 3, 6, 12\}$. This model is representative of a nonlinear g function, Ridge regularization, cross-validation to select τ and quadratic $\hat{L}$.

Support Vector Regression AR (SVR-AR) We use the SVR model to create variation among the loss function dimension. In the data-poor version the predictors set Z_t contains y_t and a number of lags chosen from $p_y \in \{1, 3, 6, 12\}$. The hyperparameters are selected with both cross-validation techniques, and we consider two kernels to approximate basis functions, linear and RBF. Hence, there are four versions: SVR-AR,Lin,POOS-CV, SVR-AR,Lin,K-fold, SVR-AR,RBF,POOS-CV and SVR-AR,RBF,K-fold. The forecasts are generated from equation (19).

E.2 Data-rich (H_t^+) models

We now describe forecasting models that use a large dataset of predictors, including the autoregressive components, H_t^+.

Diffusion Indices (ARDI) The reference model in the case of large predictor set is the autoregression augmented with diffusion indices from Stock and Watson (2002b):

$$y_{t+h}^{(h)} = c + \rho(L)y_t + \beta(L)F_t + e_{t+h}, \quad t = 1, \dots, T \tag{20}$$
$$X_t = \Lambda F_t + u_t \tag{21}$$

where F_t are K *consecutive static* factors, and $\rho(L)$ and $\beta(L)$ are lag polynomials of orders p_y and p_f respectively. The feasible procedure requires an estimate of F_t that is usually done by PCA.[15] The optimal values of hyperparamters p, K and m are selected in four ways: (i) Bayesian Information Criterion (ARDI,BIC); (ii) Akaike Information Criterion (ARDI,AIC); (iii) Pseudo-out-of-sample cross validation (ARDI,POOS-CV); and (iv) K-fold cross validation (ARDI,K-fold). These are selected from following subsets: $p_y \in \{1, 3, 6, 12\}$, $K \in \{3, 6, 10\}$, $p_f \in \{1, 3, 6, 12\}$. Hence, this model enters the following categories: linear g function, PCA regularization, in-sample and cross-validation selection of hyperparameters and quadratic loss function.

Ridge Regression Diffusion Indices (RRARDI) As for the small data case, we explore how a regularization affects the predictive performance of the reference model ARDI above. The predictive regression is written as in (20) and p_y, p_f and K are selected from the same subsets of values as for the ARDI case above. The parameters are estimated using Ridge penalty. All the hyperparameters are selected with two cross validation strategies, giving two models: RRARDI,POOS-CV and RRARDI,K-fold. This model creates variation on following axes: linear g, Ridge regularization, cross-validation for tuning parameters and quadratic loss function.

Random Forest Diffusion Indices (RFARDI) We also explore how nonlinearities affect the predictive performance of the ARDI model. The model is as in (20) but a Random Forest of regression trees are used. and only cross-validate the number of trees with K-fold. The ARDI hyperparameters are chosen from the grid as in the linear case, together with the number of trees. Both POOS and K-fold CV are used to generate two forecasting models: RFARDI,POOS-CV and RFARDI,K-fold. This model generates nonlinear treatment, with PCA regularization, using both CV techniques with the quadratic loss function.

[15] See Stock and Watson (2002a) for technical details on the estimation of F_t as well as their asymptotic properties.

Kernel Ridge Regression Diffusion Indices (KRRARDI) As for the autoregressive case, we can use the Kernel trick to generate nonlinear predictive functions g. The model is represented by equations (13) - (15) and the forecast is obtained using the equation (16). The hyperparameters of Ridge and radial basis function kernel, as well as p_y, K and p_f are selected by two cross-validation procedures, which gives two forecasting specifications: KRRARDI,POOS-CV and KRRARDI,K-fold. We use the same grid as in ARDI case for discrete hyperparameters. This model is representative of a nonlinear g function, Ridge regularization with PCA, cross-validation to select τ and quadratic $\hat{L}$.

Support Vector Regression ARDI (SVR-ARDI) We use four versions of the SVR model: (i) SVR-ARDI,Lin,POOS-CV; (ii) SVR-ARDI,Lin,K-fold; (iii) SVR-ARDI,RBF,POOS-CV; and (iv) SVR-ARDI,RBF,K-fold. The SVR hyperparameters are selected with cross validation while the ARDI hyperparameters are chosen using a grid that search in the same subsets as the ARDI model. The forecasts are generated from equation (19). This model creates variations in all categories: nonlinear g, PCA regularization, two sets of cross validation and $\bar{\epsilon}$-insensitive loss function.

E.2.1 Generating shrinkage schemes

The rest of the forecasting models relies on using different B operators to generate variations across shrinkage schemes, as depicted in section 3.2.

B_1: taking all observables H_t^+ When B is identity mapping, we consider $Z_t = H_t^+$ in the Elastic Net problem (18), where H_t^+ is defined by (5). The following lag structures for y_t and X_t are considered, $p_y \in \{1, 3, 6, 12\}$ $p_f \in \{1, 3, 6, 12\}$, and the exact number is cross-validated. The hyperparameter λ is always selected by two cross validation procedures, while we consider three cases for α: $\hat{\alpha}$, $\alpha = 1$ and $\alpha = 0$, which correspond to EN, Ridge and Lasso specifications respectively. In case of EN, α is also cross-validated. This gives six combinations: $(B_1, \alpha = \hat{\alpha})$,POOS-CV; $(B_1, \alpha = \hat{\alpha})$,K-fold; $(B_1, \alpha = 1)$,POOS-CV; $(B_1, \alpha = 1)$,K-fold; $(B_1, \alpha = 0)$,POOS-CV and $(B_1, \alpha = 0)$,K-fold. They create variations within regularization and hyperparameters' optimization.

B_2: taking all principal components of X_t Here $B_2()$ rotates X_t into N factors, F_t, estimated by principal components, which then constitute Z_t to be used in (18). Same lag structures and hyperparameters' optimization from the B_1 case are used to generate the following six specifications: $(B_2, \alpha = \hat{\alpha})$,POOS-CV; $(B_2, \alpha = \hat{\alpha})$,K-fold; $(B_2, \alpha = 1)$,POOS-CV; $(B_2, \alpha = 1)$,K-fold; $(B_2, \alpha = 0)$,POOS-CV and $(B_2, \alpha = 0)$,K-fold.

B_3: taking all principal components of H_t^+ Finally, $B_3()$ rotates H_t^+ by taking all principal components, where H_t^+ lag structure is to be selected as in the B_1 case. Same variations and hyperparameters' selection are used to generate the following six specifications: $(B_3, \alpha = \hat{\alpha})$,POOS-CV; $(B_3, \alpha = \hat{\alpha})$,K-fold; $(B_3, \alpha = 1)$,POOS-CV; $(B_3, \alpha = 1)$,K-fold; $(B_3, \alpha = 0)$,POOS-CV and $(B_3, \alpha = 0)$,K-fold.