Problem Set 2 - Forecasting Using Machine Learning

Due date: October 28, 2022

The aim of this problem set is to compare a range of models for forecasting macroeconomic data. Researchers typically have access to a large number of potential macroeconomic series that could be used for prediction. Perhaps the most common approach, at least since the work of Stock and Watson (2002), is to reduce the dimensionality of the problem by extracting a small subset of principal components from the matrix of predictors. More recently, regularized estimators and other machine learning tools have been used. We will compare a range of different techniques for forecasting changes in the US unemployment rate.

The Data

We will take data from the FRED-MD database. The database contains monthly observations of 134 economic indicators. The outcome variable of interest for us is the unemployment rate - *'UNRATE'*.

Sample selection and transformation: We will use data from January 1960 to December 2019 (most series should available after 1960, and we end the data in 2019 to avoid the latest recession, which will be tough to predict). You may still wish to drop several other series if there are missing observations; you should end up with about 120-125 series in total.

We will also want to first transform many of the series to ensure that they are stationary. Suggested transformations are listed for each series in the data file and accompanying explanations are in the 'FRED-MD Working Paper' that is available on the database website. For example, many series will be first differenced either in levels or in logs. Your outcome variable should be

$$Y_{t+1} = \Delta UNRATE_{t+1} = UNRATE_{t+1} - UNRATE_t$$

Some explanation for MSE calculations and CV

Below you will be asked to evaluate models by comparing their out-of-sample MSE. The estimation of models will sometimes require decisions about tuning parameter that may be made with cross-validation. This section contains brief explanations of these two procedures.

Evaluating out-of-sample forecast performance

For each of our chosen models, we will evaluate its predictive performance by using it to predict one-month-ahead changes in the unemployment rate. Let Y_t be the monthly change in the unemployment rate in period t and let X_t be a set of predictors that are observed at time t; this may include observations of variables at t as well as lagged values. Denote by T_0, the observation corresponding to December 1989 and T the final period in the data (i.e. the total number of periods). We will generate a prediction of Y_{t+1} using X_t for each $t = T_0, \ldots, T-1$. Specifically, for each model we do the following for every $t = T_0, \ldots, (T-1)$:

1. Estimate the model using all data up to period t. This will mean the final observation in your data set will be (Y_t, X_{t-1}).
2. Use the model to predict one month ahead, i.e. use X_t to construct the prediction $\widehat{Y}_{t+1}$

This generates a series of one-step-ahead predictions, and corresponding prediction errors. The mean-squared prediction error for the model is given by

$$MSE = \frac{1}{T - T_0} \sum_{t=T_0}^{T-1} \left(Y_{t+1} - \widehat{Y}_{t+1}\right)^2$$

Cross-validation in time series models

The standard method for cross-validation is a k-fold design in which observations are randomly sorted into k groups, with predicted values for each group constructed using estimates fitted on the remaining $k-1$ groups. With time series, this method loses validity due to the potential for serial correlation in observations over time (essentially we can no longer claim that the model errors are independent of the data we are evaluating the models on). A more principled method is to separate the data at some point in time t_0, fitting the model using observations $t \leq t_0$ and evaluating on $t > t_0$ (or even leaving a gap between the training and evaluation periods to ensure low dependence).

In practice, it seems that the standard k-fold method does not seem to suffer from the dependency issue and outperforms the single splitting method (see for example, Bergmeir and Benítez (2012)). An alternative is to do a k-fold 'blocked' design in which the k-fold are constructed from blocks of consecutive observations. This would seem to be a sensible compromise and apparently works well in practice. When asked to do cross-validation of parameters below, you are free to choose whatever approach is convenient for you (methods for ML models in R, Python etc. typically have cross-validation already built in to the function so you can just use this - it will do the random splits by default).

Forecasting models

Autoregression

A good baseline predictor to compare other methods to is a univariate autoregression, i.e.

$$Y_t = \sum_{j=1}^{J} \beta_j Y_{t-j} + e_t$$

Compute the forecasting MSE of the AR model. You can select the number of lags J using cross-validation or some other procedure, e.g. AIC/BIC, but you should report which method you use.

Factor Model

Another good baseline, since it is such a ubiquitous method in macro forecasting, is principal components regression. Here we regress Y_t on the first r principal components of X_{t-1}. There are data-driven methods for choosing r, e.g. see Bai and Ng (2001), but for simplicity may just choose r by inspection.

Using data up to T_0, compute the eigendecomposition of $X'X$ and plot the corresponding eigenvalues in descending order. Comment on the results - do the eigenvalues appear sparse or dense, do they decay quickly or slowly?

Choose some number of principal components r, based on inspection of the above plot, or otherwise. To construct forecasts using the principal components, do the following. For each time period $t = T_0$ onwards, you will: (1) compute the decomposition $X = F\Lambda'$ using data up to time $t-1$ (where F is the $(t-1) \times p$ matrix of factors, and Λ the $p \times p$ matrix of loadings, with $\Lambda'\Lambda/p = I$); (2) regress Y_t on $F_{t-1}^{(r)}$, where $F_{t-1}^{(r)}$ is the $r \times 1$ vector that contains the $(t-1)$-th row and the first r columns of F, to give regression coefficients $\widehat{\beta}$; (3) construct the forecast prediction $\widehat{Y}_{t+1} = \widehat{\beta}'(\widehat{\Lambda} X_t)$. See Stock and Watson (2002) for more on this estimator.

Compute the forecasting MSE of the principal components model.

You can also combine the factor model with the autoregression model. For example you could regress the outcome on r factors as well as J lags of itself. You could also include lags of the factors themselves in your model. Feel free to explore these alternative as well.

Regularized linear models

Using the data up to $t = T_0$, use the blocked k-fold cross-validation method to estimate the MSE for a LASSO model over some grid of chosen values for the LASSO penalty parameter λ. Plot the results.

You should ensure that you have chosen an appropriate grid of values for λ, and in particular that you have included the value of the penalty that minimizes MSE.

Compute the forecasting MSE of the LASSO predictor as well as one other choice of regularized linear predictor.

Note that, you should really be updating the choice of penalty parameter over time, as the optimal λ will change as more data is included in your estimation period. It is likely to be computationally difficult to redo the cross-validation every time a new period of data is added. You could instead choose to reevaluate the choice of tuning parameter less frequently, e.g. every five years. The optimal tuning parameter should not change too quickly over time anyway, so this seems sensible.

Other choices of regularized linear models include ridge, elastic net and LAVA. You could also consider changing the set of predictors from the original variables X to the factors you estimated above, F. LASSO is not invariant to rotations of the data - it is possible that the coefficients on X are not sparse, while the coefficients on F are sparse (or vice versa).

Nonlinear models

Compute the forecasting MSE of one choice of nonlinear ML predictor, e.g. random forest, boosted tree, or neural network.

You may choose any model and make any choices for the relevant tuning parameters that you like, but explain your choices.

Ensemble methods

We can aggregate our existing set of models to create an ensemble predictor of the form

$$\tilde{g}(X) = \sum_{j=1}^{J} \alpha_j \widehat{g}_j(X)$$

The weightings α_j can be chosen by simply regressing the outcome on the predictions from each of the J models in our test data.

Compute the forecasting MSE for an ensemble method that combines all of the models you have estimated above.

Comparing the models

Report the MSE for each model you estimate in a table. Comment on the results. How do the ML models compare to the standard autoregressive or principal components predictors?

How does the ensemble method compare to the individual estimators. Comment on the weights α_j given to each model in the ensemble predictor (these weights will differ in each period you estimate the model, so just summarize them in some way, e.g. a plot or averages).

Compare the predictive performance of models during expansionary and recessionary periods (use the NBER definition for recession periods) by taking the average squared prediction error only during recession months. Do the models perform better or worse in at predicting unemployment in recessions? Does the ranking of models in terms of their MSE change?

A note on computational issues:

Computation should not be too much of an issue, but if you find the above procedure is very slow you can reduce the number of estimations by jumping forward by one year at each step rather than one month, i.e. estimate using data up to Dec-1989 and predict Jan-1990, then estimate up to Dec-1990 and predict Jan-1991. This will give you fewer estimates to average over when computing the MSE and so will be noisier, but it's just a pset so don't spend days running code.

Extensions

These are optional, but would be interesting to consider if this is an area of interest for you.

- Repeat the above, but try to predict a different outcome, for example industrial production or inflation. Are there notable differences in the predictability of the different macroeconomic outcomes? Does the ranking of models in terms of their predictive performance change?

- Repeat the above, but try to predict three-month, six-month, or one year ahead changes in the unemployment rate. Your outcome variable should now be, for example, the yearly change in unemployment rate, i.e. $UNRATE_{t+12} - UNRATE_t$, with predictor variables measurable at time t. Are longer term or shorter term changes in the unemployment rate more predictable? Do different models perform better/worse at different horizons?

- Augment your predictor set with lags of the variables. For example, if Z_t contains all of the observed series at time t, use $X_t = Z_t, Z_{t-1}, Z_{t-2}$. You may also consider comparing different choices of the number of lags (you can do this on a smaller subset of the models, rather than all of them). Do the additional lags help with prediction or just add noise?

- Try any other methods that we have not covered in class that you are aware of. One good choice is kernel ridge regression, which is equivalent to running ridge regression on a set of basis functions of your variables and allows for interesting nonlinearities in the prediction function.

References

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

[2] Bai, J., and Ng, S.(2001), "Determining the Number of Factors in Approximate Factor Models," *Econometrica* **70**, 191–221.

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