UNIVERSITY OF CALIFORNIA

Los Angeles

Improving Automated Time Series Forecasting with the use of Model Ensembles

A thesis submitted in partial satisfaction

of the requirements for the degree

Master of Applied Statistics

by

Christopher Meade

2019

ABSTRACT OF THE THESIS

Improving Automated Time Series
Forecasting with the use
of Model Ensembles

by

Christopher Meade
Master of Applied Statistics
University of California, Los Angeles, 2019
Professor Jack W. Carlyle, Chair

There currently exist several black box software libraries for the automatic forecasting of time series. Popular among these are the forecast and bsts packages for R, which have functions to automatically fit several common classes of time series models, such as the autoregressive integrated moving average (ARIMA) and the family of exponential smoothing models, among others. It is often the case that what one gains from the ease in fitting these automatic methods comes at the cost of predictive performance. In this paper, we propose several methods to improve the prediction accuracy of automatic time series forecasting, all of which relate to creating ensembles of models automatically fit from these packages. We explore different ways that one can construct these ensembles and evaluate each on a benchmark time series dataset. In addition, we release an R software library that implements the methods discussed within this paper.

The thesis of Christopher Meade is approved.

Robert M. Stevenson

Richard L. Baker

David G. Cantor

Mario Gerla

Jack W. Carlyle, Committee Chair

University of California, Los Angeles

2019

To my mother . . .

who—among so many other things—

saw to it that I learned to touch-type

while I was still in elementary school

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

ACKNOWLEDGMENTS

(Acknowledgments omitted for brevity.)

VITA

1974–1975	Campus computer center "User Services" programmer and consultant, Stanford Center for Information Processing, Stanford University, Stanford, California.
1974–1975	Programmer, Housing Office, Stanford University. Designed a major software system for assigning students to on-campus housing. With some later improvements, it is still in use.
1975	B.S. (Mathematics) and A.B. (Music), Stanford University.
1977	M.A. (Music), UCLA, Los Angeles, California.
1977–1979	Teaching Assistant, Computer Science Department, UCLA. Taught sections of Engineering 10 (beginning computer programming course) under direction of Professor Leon Levine. During summer 1979, taught a beginning programming course as part of the Freshman Summer Program.
1979	M.S. (Computer Science), UCLA.
1979–1980	Teaching Assistant, Computer Science Department, UCLA.
1980–1981	Research Assistant, Computer Science Department, UCLA.
1981–present	Programmer/Analyst, Computer Science Department, UCLA.

PUBLICATIONS

MADHOUS Reference Manual. Stanford University, Dean of Student Affairs (Residential

Education Division), 1978. Technical documentation for the MADHOUS software system used to assign students to on-campus housing.

CHAPTER 1

Introduction

MAS Thesis Draft Chris Meade

Ensemble Methods to Improve Automated Time Series Forecasting

MAS Thesis Introduction Draft Chris Meade

Ensemble Methods to Improve Automated Time Series Forecasting

Abstract

There currently exist several black box software libraries for the automatic forecasting of time series. Popular among these are the forecast and bsts packages for R, which have functions to automatically fit several common classes of time series models, such as the autoregressive integrated moving average (ARIMA) and the family of exponential smoothing models, among others. It is often the case that what one gains from the ease in fitting these automatic methods comes at the cost of predictive performance. In this paper, we propose several methods to improve the prediction accuracy of automatic time series forecasting, all of which relate to creating ensembles of models automatically fit from these packages. We explore different ways that one can construct these ensembles and evaluate each on a benchmark time series dataset. In addition, we release an R software library that implements the methods discussed within this paper.

Introduction

Several scenarios exist that necessitate automatic time series forecasting. A manufacturing firm may need to generate monthly production forecasts for each of its products, of which there could be thousands. A hedge fund manager may need to forecast the price of a security every 30 seconds using the latest available stock data. In either case, the high volume of time series datasets or the frequency at which forecasts must be made represent too high a cost for a statistician to manually apply the classical supervised methodology to fit an appropriate, causal, and invertible ARIMA model to each time series. In the first scenario, using such a methodology would consume too many manhours of work, while in the second, a human could simply not keep up with the data.

In such situations, automatic black box time series forecasting methods, like those implemented in the popular forecast R software library by Rob J. Hyndman and the bsts package by Steven L. Scott, deliver a compelling value proposition adequate forecasts can be made almost instantly by anyone. Even if these automatic methods do not perform as well as the ideal, manually fitted ARIMA model, the value they create by reducing forecast costs (in hours of work or in time to model) can easily surpass the cost associated with implementing a less-than-ideal forecast. In the first scenario given above, 1000 adequate production-ready forecasts are better than 100 excellent forecasts and 900 missing values because the statistician ran out of time.

The work presented in this paper deals with how these black box automatic methods can improved, so that a forecaster need not settle with adequate performance while still enjoying the benefits that black box methods provide. To that end, we explore several ways to create ensembles of four of the popular black box models included in the forecast and bsts packages to increase prediction accuracy over any single automatically fit model alone. To evaluate the performance of the ensembles, they will be tested with the M3-Competition dataset, which contains 3,003 univariate time series with mostly monthly, quarterly, and yearly periods. Each series in the dataset is split into a training and test set, the latter of which will be used to evaluate loss using root mean squared error (RMSE), mean absolute percent error (MAPE), and mean absolute error (MAE).

Related Work:

Members of the Ensemble

The time series ensembles proposed in this paper will be composed of combinations of four classes of models that can be fit automatically using the ‘forecast‘ package in R. They are as follows:

Expand on: ‘auto.arima()‘: iterates over given parameter space of model order, estimates parameters for each, and chooses best based on information criterion (AIC, AICc, or BIC)

‘ets()‘: short for error, trend, seasonal, the ‘ets()‘ function. Fit 30 different classes of exponential smoothing models to data, chooses best on basis of AIC.

‘bsts()‘:

‘thetaf()‘: fit model using theta method described in Assimakopoulos, 2000.

The Case for Ensembles

The use of ensembles to improve prediction accuracy is nothing new in the realm of statistical modeling. In 1996, Leo Breiman introduced the concept of bootstrap aggregation, shortened to bagging (Breiman, 1996). Given a training set L, Bagging uses the bootstrap to create new training sets Li, i = 1,, B. A given machine learning model model is then fit to each of the new training sets. The final decision is made by taking the average of the B models in the case of regression, or by majority vote for classification. The method, one of the first implementations of ensemble learning, was successful, with Breiman concluding What one loses [] is a simple and interpretable structure. What one gains is increased accuracy.

In the case of bagging, it was found that an ensemble of models performed better than any single learner. Dietterich (2002) gives three possible explanations of the improved predictive performance of ensemble learning. For example, in the event of insufficient training data to establish a best fit model, it may be the case that several different learners provide an equally accurate but vastly different fit to the training data. Making a future prediction with only one of these models can be risky, due to the high variance of the predictions. However, an ensemble can reduce this prediction variance and thus reduce risk with a simple majority vote or average over all predictions.

It can also be the case that finding the best model to fit to a training data set can be too computationally expensive. This is especially true with gradient based methods, as a learner may become trapped at a local minimum and fail to reach the global minimum. In such a scenario, an ensemble reduces the risk of choosing the wrong minimum.

Finally, it is possible that no single learner can accurately model a given data. Creating an ensemble allows one to explore more functional relationships between between data and model, which comprise what Dietterich calls the representation space. One of these previously untested members of the representation space may perform better than any of its component learners.

In each of the three explanations provided by Dietterich, the advantages attributed to ensemble learning directly relate to the diversity of its member learners (Oliveira 2014). That is, the predictions made by the member learners are relatively uncorrelated. In bagging, uncorrelated learners are created with bootstrap sampling of the dataset. The random forest, in addition to bootstrap sampling, uses a random subset of predictor variables to construct uncorrelated classification and regression trees (Breiman 2001).

Application to Time Series

How, then, can an ensemble of uncorrelated time series models be constructed? The predictions from two reasonable ARIMA models fit to the same data will likely have a correlation coefficient close to one. An ensemble created from these two models will likely look very similar to both of the original forecasts.

In time series forecasting, a natural way to create diverse forecasts is to combine the forecasts of different classes of time series models, such as the ARIMA, Theta, Exponential Smoothing, and Bayesian Structural Time Series discussed in the previous section. Because forecasts created these different classes of models will naturally vary, an ensemble of these learners should enjoy the same benefits, namely a increase in prediction accuracy from a reduction in bias and variance and a robustness to training noise and outliers, as those ensemble methods mentioned above.

Expand on:

Talk about Random Forest methods (Breiman 2001): better predictive performance than a single CART

Therefore ensembles of time series models should enjoy the same benefits as other ensemble methods, such as Random Forest, Adaboost, XGBoost, etc.

The key to this result lies in making sure predictors arent highly correlated. Random Forest works because the CARTs in the model are not correlated. Result is a reduction in bias and variance. Robust to outliers and noise. Implies better predictive performance. For example, two reasonable ARIMA models with similar (but not the same) orders fit to a dataset will likely have a correlation coefficient near one. An ensemble created from these two models will likely look similar very similar to both of the originals. So we must create predictors that arent highly correlated.

In using four distinct classes of time series models (neural network, ARIMA, Exponential Smoothing, Theta method), we solve the problem of highly correlated predictors. So ensembles of these models should work better than any single one of them.

Creating the Ensembles

We propose the following ensemble methods:

Naive Ensemble

Perhaps the simplest way to create an ensemble of multiple predictions is to take the average of all the predictors for each time point on the prediction horizon. Specifically, let X be a h by p matrix, where h is the prediction horizon and p is the number of models from which predictions were made. In this design, each column of the forecast matrix X corresponds to a prediction of horizon h made by one of the p predictions. Let W be a 1 by h weight matrix with every element given by 1/p. Then a final forecast is given by XW.

We are specifically interested in Naive Ensembles made with the four classes of automatic time series models discussed in the Chapter 2. We will refer to this model as the BEAT (Bsts, Ets, Auto Arima, Theta) Ensemble. Also of interest are the four Naive Ensembles made using three of the four component models, called BEA, EAT, BAT, and BET, following the same naming convention.

Median Ensembles

The Naive Ensembles proposed in the previous section simply take the average of component forecasts at each time point in the forecast horizon. If one of these component forecasts predicts extreme or unreasonable values, they can have a huge impact on the accuracy of the Naive Ensemble on the whole.

One solution to combat the influence of an extreme forecast is to take the median, rather than the mean, of the p predictors at each time point in the forecast horizon. If X is the same h by p forecast matrix, let M = m1, , mh be the final forecast, where each mi is the median ith row of X, i = 1,,h. We will refer to this method as the Median Ensemble.

Methodology

Each ensemble method, in addition to the automatic component learners (auto.arima, thetaf, bsts, and ets), will be evaluated on the M3 Competition Dataset (Makridakis and Hibon, 2000). This dataset contains 3003 time series, each of which is divided into a training and test set. Of the 3,003 series, 645 are yearly data, 756 are quarterly, 1428 are monthly, and 174 have frequencies that are not yearly, quarterly, or monthly. To ensure that enough data is available to make a reasonable forecast, each yearly series has at least 14 observations, quarterly series have at least 16 observations, monthly data have at least 48 observations, and series with frequencies that are not yearly, quarterly, or monthly have at least 60 observations.

Each method will be fit to the training set of each series, and will create a forecast on the same time interval as the test set. In this way, forecasts can be evaluated against the ground truth for each of the 3,003 series. Metrics: RMSE, MAPE, MAE for each series. Problem: RMSE and MAE are scale dependent. Does not make sense to compare RMSE and MAE of a model on multiple time series. Solve this problem by normalization. Calculate mean of training set. Divide RMSE and MAE by this mean to get normalized RMSE (nRMSE) and normalized MAE (nMAE). This normalization process allows us to compare the performance of the models on the 3,003 series in the M3 Competition Dataset. MAPE, mean absolute percent error, is already scale independent.

In evaluating the nRMSE, nMAE, and MAPE for each method on each of the time series in the competition dataset