An Algorithm for Forecasting Time Series From the M4 Competition Dataset

David Veitch
University of Toronto
david.veitch@mail.utoronto.ca

August 2019

1 Introduction

The M4 competition, conducted in 2018, is the latest iteration of the Makridakis competitions, which seek to further the understanding of time series forecasting techniques [Makridakis et al., 2018]. The competition's dataset is a collection of 100,000 time series from a variety of domains such as economics, demographics, and tourism, taken at a variety of frequencies (e.g. yearly, monthly, hourly).

What separated the M4 competition from previous versions of the competition was the size of the dataset (in comparison the M3 competition only included 3003 time series), the inclusion of some machine learning techniques in the benchmark models (MLPs and RNNs), and using prediction intervals as a performance metric (previous competitions only used point forecasts).

In this paper I describe the forecasting approach I used to forecast a portion of the M4 competition dataset. Specifically I focus on computing forecasts, and their corresponding prediction intervals, for the time series with monthly frequencies found in the M4 competition dataset. The models which I used, and ultimately combined, were purely statistical methods. I also attempt to understand what features of time series lead certain models to outperform others in their forecasting ability.

2 Unique Challenges of the Competition

Creating an 'automatic model selection' algorithm for this dataset posed two specific challenges:

2.1 'Blind Forecasting'

Many approaches to model selection for time series data include some form of ad-hoc, or visual determination, of what is an appropriate model. For example, when using ARIMA models the presence of a slowly decaying positive autocorrelation function points at the need to difference the data. Another example is the need for a variable transformation is usually determined by visual inspection of the residuals.

In building a forecasting algorithm for the M4 competition dataset, one cannot rely on ad-hoc methods, and must instead create a systematic approach for dealing with issues such as those mentioned above.

2.2 The Size of the Dataset

Even when only considering time series with monthly frequencies, the total number of time series is 48,000. One challenge that arises is that some of these time series exhibit very erratic behaviour, and the forecasting method used must be robust against such behaviour.

The other, and more significant challenge with the size of the dataset is the computational cost of fitting several models to each time series. Even assuming each time series took one second to fit several models to, this would take in total thirteen hours to model every time series. This high computational cost limited the opportunity of taking an iterative approach to model development. Further discussion of how the computation was conducted can be found in Section 6.

3 Time Series Models Used

Four linear time series models from the ARIMA family were used to model the data. Of these, two were decidedly ‘naïve’ random walk models, and were included because for time series demonstrating erratic behaviours, a naïve model is much less likely to overfit the data.

For model selection I rely on AICC as a selection criteria, a bias corrected version of the information criterion of Akaike [Hurvich and Tsai, 1989].

$$\begin{aligned} AICC &= AIC + \frac{2(m+1)(m+2)}{n-m-2} \\ AIC &= -2\log\hat{L} + 2m \end{aligned}$$

- $\hat{L}$ the likelihood of the data under the model
- n the number of observations
- m the number of paramaters in the model, $m = p + q + P + Q + k$ for a seasonal ARIMA model
- $k = 1$ if the model contains drift, $k = 0$ otherwise

3.1 Model 1 - Seasonal ARIMA Model with Drift

The first model fit to each time series was a Seasonal ARIMA Model with Drift with the following paramaterization [Brockwell and Davis, 2016]:

$$\begin{aligned} \phi(B)\Phi(B^s)Y_t &= \theta(B)\Theta(B^s)Z_t \\ Y_t &= (1-B)^d(1-B^s)^D X_t \\ Z_t &\sim \text{WN}(0,\sigma^2) \end{aligned}$$

The model incorporates drift by allowing $\mu_y \neq 0$ for $d = 1$. The best ARIMA model was selected using the auto.arima() function in the `forecast` package in R. In this function the models were chosen based on the best AICC via a stepwise selection method.

3.2 Model 2 - Holt Winters’ Seasonal Method

Two Holt-Winters’ models, additive and multiplicative, were fit to each time series and the model with the highest AICC was selected. The paramaterization [Hyndman, 2018] is one which α, β^*,and γ are smoothing parameters for the level, trend, and seasonal component respectively. This method can be seen as a special case of an ARIMA model.

3.2.1 Holt Winters’ Additive Method

$$\begin{aligned} \hat{y}_{t+h|t} &= \ell_t + hb_t + s_{t+h-m(k+1)} \\ \ell_t &= \alpha(y_t - s_{t-m}) + (1-\alpha)(\ell_{t-1} + b_{t-1}) \\ b_t &= \beta^*(\ell_t - \ell_{t-1}) + (1-\beta^*)b_{t-1} \\ s_t &= \gamma(y_t - \ell_{t-1} - b_{t-1}) + (1-\gamma)s_{t-m} \end{aligned}$$

3.2.2 Holt Winters’ Multiplicative Method

$$\begin{aligned} \hat{y}_{t+h|t} &= (\ell_t + hb_t)s_{t+h-m(k+1)} \\ \ell_t &= \alpha\frac{y_t}{s_{t-m}} + (1-\alpha)(\ell_{t-1} + b_{t-1}) \\ b_t &= \beta^*(\ell_t - \ell_{t-1}) + (1-\beta^*)b_{t-1} \\ s_t &= \gamma\frac{y_t}{\ell_{t-1} + b_{t-1}} + (1-\gamma)s_{t-m} \end{aligned}$$

3.3 Model 3 - Random Walk

The third model fit, and the most naïve of all models was the Random Walk model:

$$\begin{aligned} X_t - X_{t-1} &= Z_t \\ Z_t &\sim \text{IID}(0, \sigma^2) \end{aligned}$$

3.4 Model 4 - Random Walk with Drift

The final model, a slightly less naïve model, is the Random Walk with Drift model:

$$\begin{aligned} X_t - X_{t-1} &= \mu + Z_t \\ Z_t &\sim \text{IID}(0, \sigma^2) \end{aligned}$$

- μ the average change between periods

4 Performance Measures

For the purposes of the M4 competition two point forecast performance measures were used (sMAPE and MASE) and one prediction interval performance measure (MSIS). The averages of sMAPE and MASE are to be computed and then divided by the average sMAPE and MASE for the Naive2 method (from the M4 competition), and the average of these two ratios is then reported as Overall Weighted Average (OWA). The benchmark sMAPE and MASE for the Naive2 model for the 48,000 monthly series are 14.427% and 1.063 respectively.

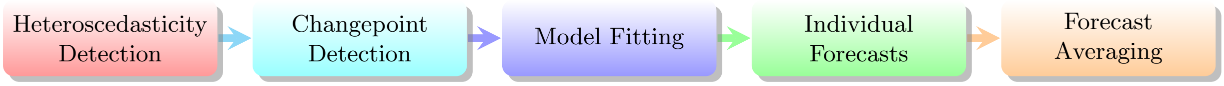
$$\begin{aligned} \text{sMAPE} &= \frac{1}{h}\sum_{i=1}^{h}\frac{2|Y_i - \hat{Y}_i|}{|Y_i| + |\hat{Y}_i|} \\ \text{MASE} &= \frac{1}{h}\frac{\sum_{i=1}^{h}|Y_i - \hat{Y}_i|}{\frac{1}{n-m}\sum_{i=m+1}^{n}|Y_i - Y_{i-m}|} \end{aligned}$$

- Y_t an observation at time t
- $\hat{Y}_t$ the predicted value at time t
- m the frequency of the data (12 for monthly)

For our purposes we will assess performance of prediction intervals based on what proportion of future observations fall within our 95% prediction intervals.

5 Modelling Process

A multi-step modelling process was used to produce forecasts for each time series.



5.1 Heteroscedasticity Detection

A key assumption of the above time series models is that the variance of the white noise sequence is constant. A common problem in time series, particularly those in economics, is that the white noise variance increases with the level of the series.

To test for this we use a process inspired by the Breusch-Pagan test. First an ARIMA model is fit to a time series. Next a Breusch-Pagan test is conducted with the fitted values of the model as the explanatory variables, and the squared residuals as the dependent variables in a linear regression ($\hat{u}^2 = \beta_0 + \beta_1 FittedValues + \epsilon$). A linear model is fit to test the statistical significance of β_1. If β_1 is insignificant than no transformation is applied to the data. However, if β_1 is significant a square-root transformation is applied. The test is then conducted once again and if β_1 is still significant a more powerful transform, a log transform, is applied to the original series.

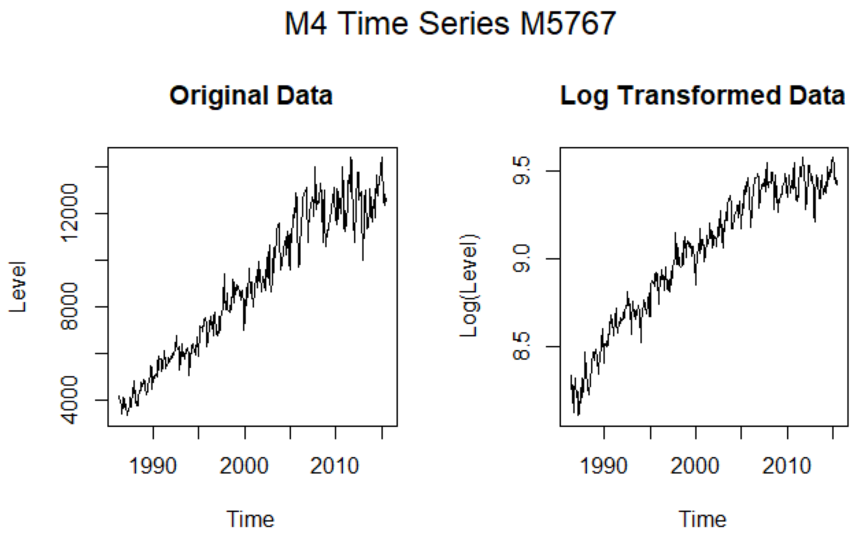


Figure 1: Example Detection of Heteroscedasticity

5.2 Changepoint Detection

Many of the time series in the M4 competition dataset exhibit dramatic changes in behaviour, be it changes in the mean or variance of the process.

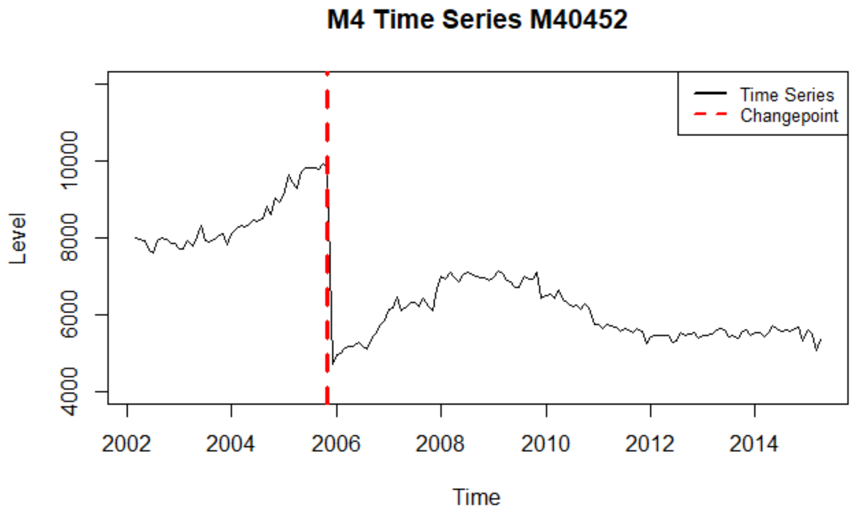


Figure 2: Example Detection of a Changepoint

To account for the fact the time series may be behaving differently near the forecast window relative to how it was previously, a changepoint detection algorithm (from the R library `changepoint`) [Killick and Eckley, 2014] detects changepoints in the mean and covariance structure which occurred sixty or more months before the last observation. A cutoff of sixty is used to ensure there is enough data to fit a reasonable model to. If such a cutoff exists, the algorithm fits a model during the model fitting step to the data occurring after the changepoint, and uses this for forecasting.

The At Most One Changepoint (AMOC) method is used and the data generating process is assumed to be normal. At a high level this test uses a hypothesis test where $H_0 : m = 0$, $H_1 : m = 1$, where m is the number of changepoints in the time series. This is conducted via maximum likelihood where the changepoint comes at τ_1 ($\tau_1 \in \{1, \dots, n\}$), where τ_1 is computed from:

$$\tau_1 = \max_{\tau} \left[\log p(y_{1:\tau}|\hat{\theta}_1) + \log p(y_{(\tau+1):n}|\hat{\theta}_2) \right]$$

A test statistic is then computed of the form:

$$\lambda = 2\Big[\big(\log p(y_{1:\tau_1}|\hat{\theta}_1) + \log p(y_{(\tau_1+1):n}|\hat{\theta}_2)\big) - \log p(y_{1:n}|\hat{\theta})\Big]$$

5.3 Model Fitting

After heteroscedasticity and changepoint detection has occurred the model fitting step occurs. Each of the models in Section 3 are fit by minimizing the AICC of each model.

5.4 Individual Forecasts

From each model, forecasts of the next 18 observations, and their 95% prediction intervals are produced. These forecasts seek to minimize the squared forecast errors. It should be noted that any forecasts or confidence intervals which produced negative values were floored at zero.

5.5 Forecast Averaging

To combine our forecasts we utilize a decision tree algorithm. This method is particularly attractive for its interpretability. We train this algorithm to predict the probability each of the four time series models will produce forecasts with the lowest error based on features of a time series.

The features of each time series are computed using the `tsfeatures` package in R. They include:

1. x_acf1, diff1_acf1, diff2_acf1: the first autocorrelation coefficient of the original time series, its first difference, and its second difference

2. x_acf10, diff1_acf10, diff2_acf10: the sum of the first ten squared autocorrelation coefficients of the original time series, its first difference, and its second difference

3. seas_acf: the autocorrelation coefficient at the first seasonal lag

4. entropy: the spectral entropy of a time series (a measure of how chaotic a time series is). Where $\hat{f}(\lambda)$ is the estimated spectral density of the data, it is calculated as:

$$\text{Entropy} = -\int_{-\pi}^{\pi} \hat{f}(\lambda) \log \hat{f}(\lambda) d\lambda$$

The probabilities produced by this model are then used as the weights to average each models' forecasts.

The decision tree is initially trained on training set of 24,000 time series. From there another 12,000 time series are used as a validation set to tune the optimal depth and splitting criteria of the tree.

6 Computation

As mentioned above one of the unique challenges of this dataset was its sheer size. This computational challenge was overcome via parallel computing, specifically via R's `foreach` and `doParallel` packages. This was made possible by the fact that fitting models to each time series only depended on the values of the specific time series in question. Using this functionality, in combination with 20 cores on a server, the model fitting and forecasting step could be run in under eight hours.

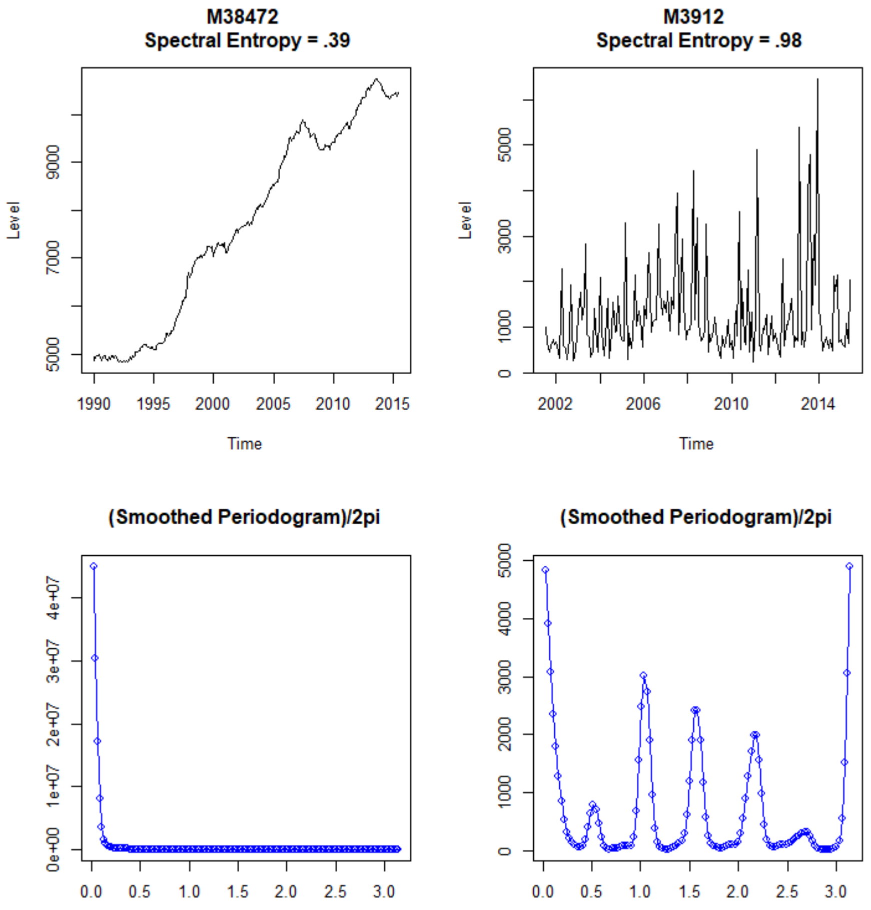


Figure 3: Example of Time Series with High and Low Spectral Entropy

7 Results

7.1 Key Performance Measures

7.1.1 Point Forecasts

	MASE (all series)	MASE (test set)	sMAPE (all series)	sMAPE (test set)	OWA (all series)	OWA (test set)*
Naive2 Benchmark	1.063		14.427%		1.000	1.000
Equal Weight	1.097	1.147	14.475%	14.567%	1.017	1.045
Optimal Weight	1.046	1.087	13.752%	13.826%	0.968	0.991
ARIMA	1.078	1.091	14.414%	14.468%	1.007	1.015
Holt Winters'	1.212	1.376	16.295%	16.317%	1.135	1.213
Random Walk	1.182	1.191	15.655%	15.725%	1.098	1.105
Random Walk w/Drift	1.158	1.166	15.496%	15.489%	1.082	1.085
* For test set OWA Naive2 Benchmark for all series was used for calculation purposes						

As can be seen from Figure 4 the performance of my algorithm falls somewhere near the middle of the performance reported for submissions' to the competition performance on time series with monthly frequencies. I would expect the 'true performance' to land somewhere between that of the Optimal Weightings on the test set and of all series. Given the fact that the naïve method reports worse performance on the test set than all series, this leads me to believe the test set contained a few 'harder' time series to forecast. It is notable that using an optimal weighting scheme leads OWA to fall by approximately 5%, pointing to the value in weighting forecasts based on their expected usefulness.

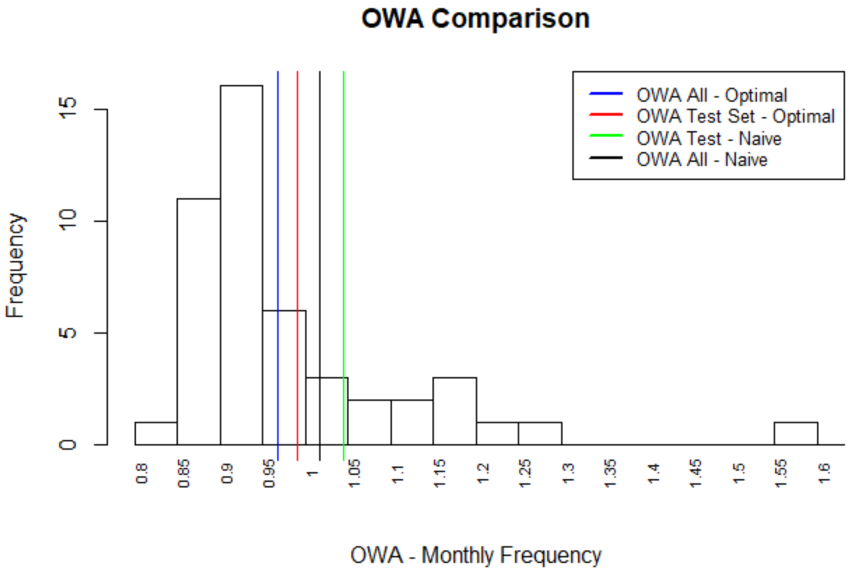


Figure 4: Comparison of OWA with M4 Submissions' Monthly OWA

This seems like a reasonable performance given the relatively limited number of models I considered compared to some other entries to the competition.

7.1.2 Prediction Intervals

	Percentage of Out-of-Sample Values Within 95% Prediction Intervals
Naive Weights (test set)	95.30%
Optimal Weights (test set)	95.21%
Naive Weights (all series)	95.45%
Optimal Weights (all series)	95.37%

Table 1: Prediction Interval Coverage

Amazingly the prediction intervals produced by averaging the prediction intervals for both weighting methods are remarkably closer to 95%, as shown in Table 1.

7.2 Model Averaging

We see in Figure 4 the optimal decision tree used for model averaging. This tree is a maximum of two layers deep and uses the gini criteria for splitting (parameters tuned on the validation set). A few things stand out.

First entropy and seas_acf1 (the autocorrelation at the first seasonal lag, which is 12 for monthly series) appear to be the most important variables for model selection. This intuitively makes sense that time series with higher spectral entropy require more complex models (i.e. ARIMA as opposed to random walk models), and also that time series with low entropy but high seasonal autocorrelations would be best suited for an ARIMA model which captures seasonality.

Next, it is notable that the Holt-Winters model does not appear to be the best model in any of the nodes of the decision tree. Potentially this can be explained by the fact that since it is just a special case of an ARIMA model, on average ARIMA models will outperform it.

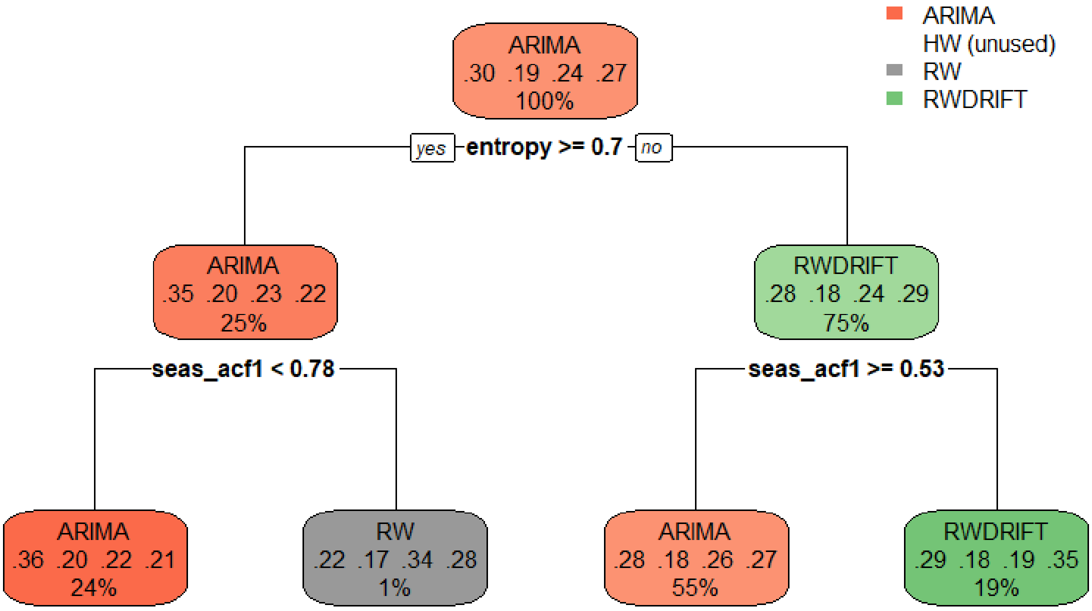


Figure 5: Decision Tree Used for Model Averaging. Each node lists method which is most frequently the best method, proportions showing how often each method is the best method, and a percentage referencing the total number of time series in the training set that fall into the node.

Finally, it is interesting how frequently random walk, and random walk with drift models are the top performing models. Approximately 51% of the top performing models in the test set are these relatively naïve models. This is a somewhat surprising result, which speaks to the inherent unpredictability of real-world time series.

8 Future Research Directions

With the M5 competition slated to begin in 2020 there are a number of promising avenues to pursue with respect to improvements to this modelling approach.

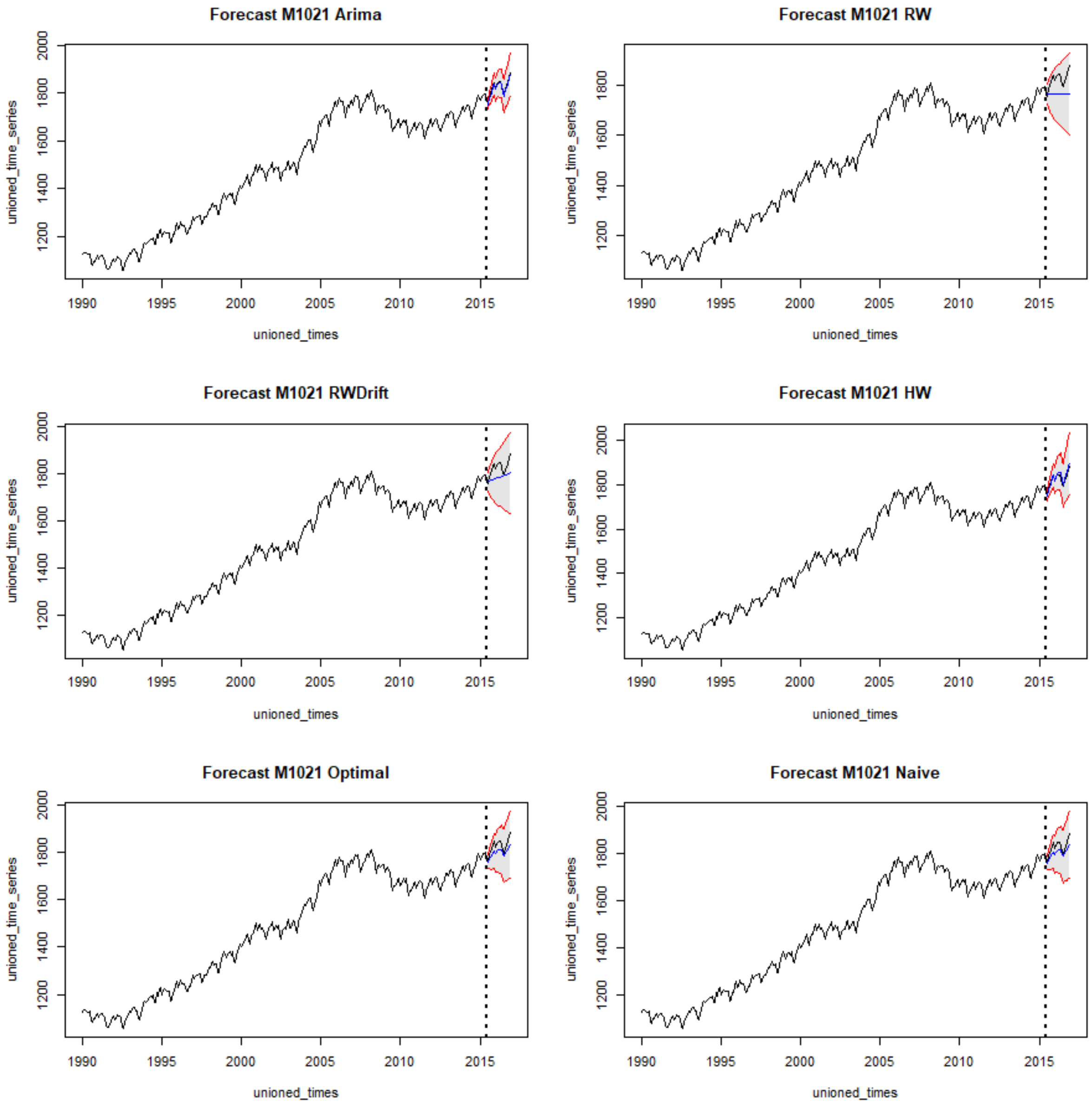
The first, and most obvious being to increase the number of forecasting methods used to include new methods, and modifications of existing methods. For example, it would be worthwhile considering ARIMA models that do, and do not incorporate changepoints; or those that use BIC instead of AICC as a model selection criteria.

Another potential avenue is to include more ‘black-box’ forecasting methods. Notably absent from my analysis was a ‘pure’ machine learning method. And there is some evidence from the winners of the M4 competition that using machine learning is a key to success [Smyl, 2018].

Another potential direction is coming up with meaningful new features to describe time series. Clearly certain models describe certain types of time series better than others, and knowing the features that dictate which models do well are key to improving forecast performance.

Appendices

A Example Time Series Forecast



Forecast M1021 Arima

Forecast M1021 RW

Forecast M1021 RWDrift

Forecast M1021 HW

Forecast M1021 Optimal

Forecast M1021 Naive

Figure 6: Example forecast, and prediction intervals, made on time series M1021 in the M4 dataset by each of the four models used, and the optimal and naive weightings.

References

Peter J. Brockwell and Richard A. Davis. *Introduction to Time Series and Forecasting*. Springer, 2016.

Clifford M. Hurvich and Chih-Ling Tsai. Regression and Time Series Model Selection in Small Samples. *Biometrika*, 76(2):297–307, 1989. ISSN 00063444. URL `http://www.jstor.org/stable/2336663`.

Rob J. Hyndman. *Forecasting: Principles and Practice*. OTexts, 2018. URL `OTexts.com/fpp2`.

Rebecca Killick and Idris Eckley. changepoint: An R Package for Changepoint Analysis. *Journal of Statistical Software, Articles*, 58(3):1–19, 2014. ISSN 1548-7660. doi: 10.18637/jss.v058.i03. URL `https://www.jstatsoft.org/v058/i03`.

Spyros Makridakis, Evangelos Spiliotis, and Vassilios Assimakopoulos. The M4 Competition: Results, Findings, Conclusion and Way Forward. *International Journal of Forecasting*, 34(4):802–808, 2018. doi: 10.1016/j.ijforecast.2018. URL `https://ideas.repec.org/a/eee/intfor/v34y2018i4p802-808.html`.

Slawek Smyl. My M4 Competition Models , Dec 2018. URL `http://www.mcompetitions.unic.ac.cy/wp-content/uploads/2018/12/Smyl-M4-models.pdf`.