ISyE 6402 Project Report
Forecasting Hourly Electricity Demand in the state of Georgia

Vidyut Rao

[1]MS in Industrial Engineering, GT ID: 903511348, vrao67@gatech.edu

1 Introduction

In the domain of statistical methods for forecasting, few use cases see more utility and impact on the bottom line than that of demand forecasting. This field of predictive analytics tries to understand and predict consumer demand in order to optimize supply decisions. Predicting the demand for electricity is crucial to inform governments and private energy providers on how much energy is to be generated for the near future. Improved data and understanding of not only local, but also global energy consumption may reveal systemic trends and patterns, which could help frame current energy issues and encourage movement towards collectively useful solutions.

In this project I study the characteristics of hourly electricity demand in the state of Georgia over the course of 4 years from Jan 2016 to Dec 2019. This data is available on the *U.S Energy Information Administration* website, where the data collected on the hourly grid monitors of the Southern Company Services (SOCO) is what we use for our needs. The EIA website provides a centralized and comprehensive source for hourly operating data about the high-voltage bulk electric power grid in the Lower 48 states. Their data is collected from the electricity balancing authorities (BAs) that operate the grid.

I begin with some data cleaning and wrangling to obtain the time series in a format appropriate for analysis. After some exploratory data analysis, I study the trend and seasonality of the time series and proceed to remove those elements. Next, an ARIMA model is fit to the time series and the characteristics of it's residuals are studied to see if a GARCH model is required. The eventual forecasts are evaluated against a validation set and the metric (Root Mean Squared Error) is compared against the results of the xgboost model and a model than only predicts the mean of the training data.

2 Dataset

While the US EIA website has hourly energy demand to the present day (April 2020), the current COVID-19 pandemic has made the demand for electricity during the year of 2020 highly erratic. The data during this time doe not follow the trend during the years that preceded it. This makes forecasting the energy demand near impossible. To ease this analysis, I have only used the data available until December 31st, 2019. This data has 35064 datapoints. The last 1000 data points, since 2019/11/20 09:00:00, have been used as the validation set. We plot the demand against time below:

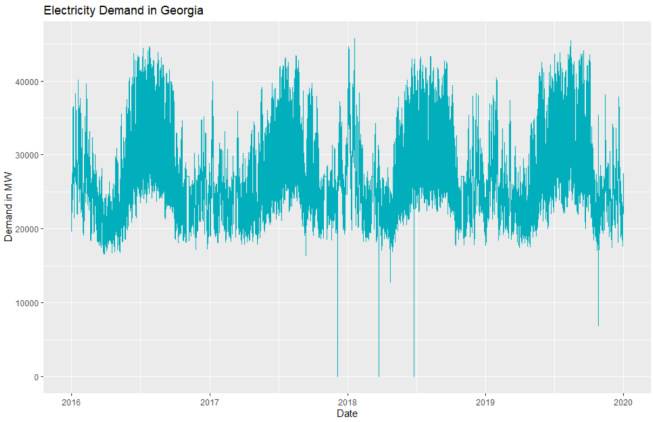

Figure 1: Electricity Demand over time

It is clear from the plot above that the demand for electricity sees some seasonality. We see a surge in demand during the summer and winter months, most likely due to an increased utilization of cooling and heating devices amongst other things, and a lull during the spring and fall months. Apart from the yearly trend, seasonality of smaller scales (hourly/daily/weekly) might also be present, but this is hard to perceive here.

As a final step before starting the forecasting process, I augment the data with more information about the time such as hour of day (x1), month of the year(x2), day of the week(x3) and week of the year(x4). In order to gauge its usefulness, I calculate the feature importances using the *randomForest()* function in R. We see that the hour of the day and month of the year are very useful predictors.

```
> importance(PredImp, type = 1)
          %IncMSE
x1      899.87756
x2      214.07532
x3       88.91557
x4       42.96391
time.pts 71.83938
```

Figure 2: Feature Importance

3 Methodology and Analysis

3.1 Trend and Seasonality Estimation

The first step in the forecasting process is to remove the trend and seasonality of the time series. With the predictors built above, I test a host of methods to estimate the trend and seasonality. These methods include: linear regression, non- parametric regression, generalized additive models, splines and the sine-cosine model. Below is the fit of the GAM model using the original data, in purple color:

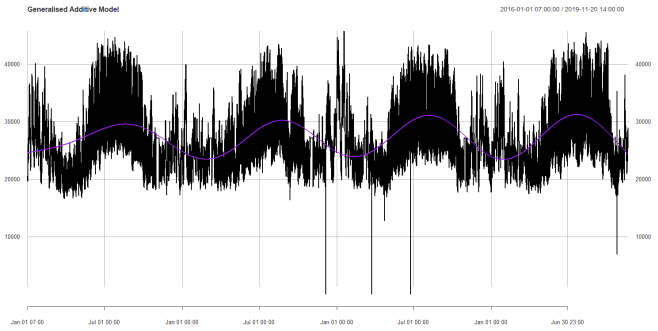

Figure 3: Using Generalized Additive Models to fit the seasonality and trend

Running the *ndiff()* function in R tells us that this data requires a differencing of order 1 to be more tractable. This also eliminates the need for a 'd' component in the ARIMA model, allowing us to easily feed the ARMA parameters into the GARCH model later on. Amongst the models tested, the best fit was obtained by the Local Polynomial Regression(LOESS) model, trained on the features x1 and x2. The fit on the differenced data is shown below:

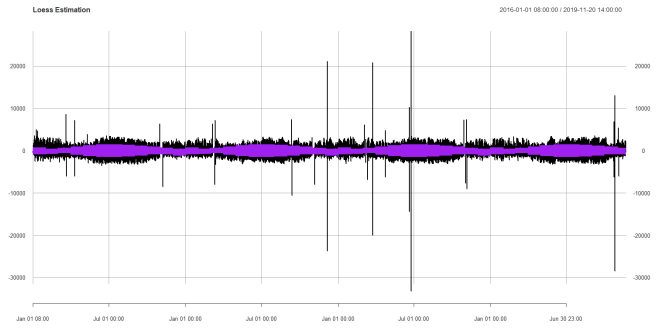

Figure 4: LOESS estimation of the seasonality and trend on the differenced series

3.2 ARIMA

ARIMA, which is short for *Autoregressive Integrated Moving Average* is a model that can be applied to a time series that has been detrended and deseasonalized. Ideally, this series should also be stationary. If not, the differencing aspect of the ARIMA model or future GARCH treatment can take care of the non-stationarity.

The Autoregressive section of the model indicates that random variable is regressed on its own lagged values. The Moving Average part indicates that the regression error is actually a linear combination of error terms whose values occurred contemporaneously and at various times in the past. The I (for "integrated") indicates that the data values have been replaced with the difference between their values and the previous values (and this differencing process may have been performed more than once). The purpose of each of these features is to make the model fit the data as well as possible.

To find the parameters of the model that best fit the data, I use the *auto.arima()* model in R. This function automatically searches for the best AR (p), differencing (d) and MA (q) components of the model that yields the lowest *Akaike Information Criterion*. This function also gives us the best seasonal components of the ARIMA model (P,D,Q), but these turn up to be 0.

The final components of the ARIMA model obtained are (6,0,4). More details about the summarized ARIMA model can be found in the R code. For further analysis, we study the Autocorrelation Function(ACF) and Partial Autocorrelation Function(PACF) plots of the residuals of the fitted ARIMA model.

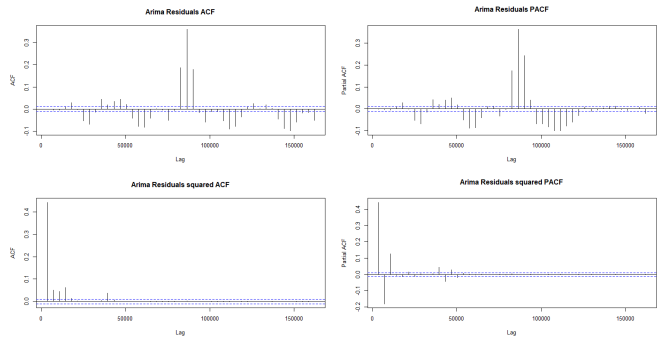

Figure 5: ACF and PACF plots of the ARIMA residuals and their squared values

The ACF and PACF plots of the ARIMA residuals show random spikes at very large lags that don't really give us much information. The squared residuals however, tell us that the first few lags exhibit significant correlation indicating the variance is autocorrelated and non-stationary. Since the ACF and PACF are geometrically decreasing, it indicates that an ARMA model is appropriate to model the variance, which is the foundation of the GARCH(p,q) model.

The *Box-Pierce* tests below also confirm the results. While the residuals at p value greater than 0.05 seem stationary (the Null hypothesis of the Box-Pierce test is that the residuals are independent), the squared residuals have a significant p value (much less than 0.05) implying dependence.

```
> Box.test(resids, type = "Box-Pierce")

        Box-Pierce test

data:  resids
X-squared = 0.022206, df = 1, p-value = 0.8815

> Box.test(resids^2, type = "Box-Pierce")

        Box-Pierce test

data:  resids^2
X-squared = 6680.5, df = 1, p-value < 2.2e-16
```

Figure 6: Box-Pierce tests of the ARIMA residuals and their squared values

3.3 GARCH

Generalized Autoregressive Conditional Heteroskedasticity (GARCH) models are appropriate when the error variance in a time series follows an autoregressive moving average (ARMA) model. It's clear from the ACF and PACF plots we see earlier of the squared residual series that a GARCH model is necessary to estimate the volatility.

Our object is now to determine the parameters p and q of the GARCH(p,q) model. Using the *rugarch* library in R, we set up a simple grid search that runs through a multiple combinations of p and q to find the combination that has the lowest *Bayesian Information Criterion*. The best model was found to be GARCH(1,3). Below is the summary of the GARCH model:

```
*---------------------------------*
*          GARCH Model Fit        *
*---------------------------------*

Conditional Variance Dynamics
-----------------------------------
GARCH Model      : sGARCH(0,3)
Mean Model       : ARFIMA(6,0,4)
Distribution     : std

Optimal Parameters
------------------------------------
        Estimate  Std. Error     t value Pr(>|t|)
mu    -98.019360    6.009794 -1.6310e+01 0.000000
ar1     1.763781    0.000118  1.4918e+04 0.000000
ar2    -0.458480    0.000167 -2.7428e+03 0.000000
ar3    -1.324405    0.000104 -1.2747e+04 0.000000
ar4     1.262182    0.000114  1.1075e+04 0.000000
ar5    -0.504829    0.000236 -2.1435e+03 0.000000
ar6     0.104377    0.000289  3.6169e+02 0.000000
ma1    -0.891269    0.000002 -5.5112e+05 0.000000
ma2    -0.606536    0.000014 -4.2719e+04 0.000000
ma3     1.101017    0.000010  1.1030e+05 0.000000
ma4    -0.166530    0.000007 -2.3445e+04 0.000000
omega 2230.155802   39.515013  5.6438e+01 0.000000
beta1    0.000005    0.001688  2.8880e-03 0.997696
beta2    0.005837    0.001688  3.4577e+00 0.000545
beta3    0.986878    0.000002  4.3450e+05 0.000000
shape    3.000514    0.042044  7.1366e+01 0.000000
```

Figure 7: Summary of the GARCH(1,3) model. All coefficients are significant except for the first MA coefficient of the GARCH model.

3.4 Forecasting

With all our models in place, we forecast the last 1000 data points from 2019/11/20 09:00:00 in steps of 50 i.e. we forecast 50 points into the future with the training data and then add these forecasted values to the training data and forecast the next 50 points and so on. This requires 20 GARCH models to be fit over 1000 datapoints. With better computational power, one could fit these models in steps of 1, forecasting each point individually, thus getting much better forecasts. We also add the seasonal and trend components back to these forecasts and undifference the data in order to evaluate the forecasts against the original validation set. The resulting forecasts are plotted below:

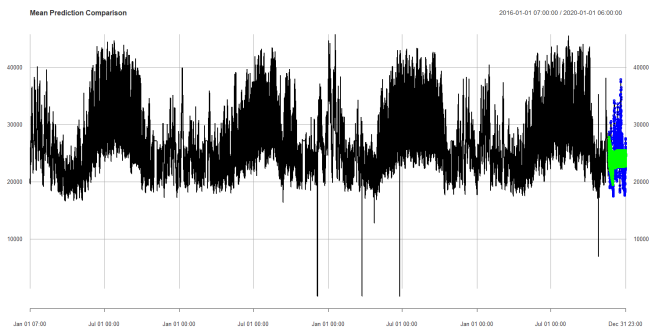

Figure 8: Green: Forecasted Values, Blue: Actual Values

The Root Mean Squared Error of our model is 3650.836

4 Summary and Conclusion

In order to gauge the relative magnitude of the RMSE obtained, I also built a model using the popular *xgboost()* function in R with the predictors defined earlier, as well as another model which only predicts the mean of the training data. I have summarized my findings below:

Model	**RMSE**
ARIMA + GARCH	3650.836
XGBoost	4472.886
Mean Model	4511.758

Table 1: Results of the different models

We see that the ARIMA/GARCH model we built has a **19.08%** improvement over the mean model and a **18.38%** improvement over the xgboost model. While this performance is great, there is much room for improvement. With better computational power we can calculate the test data in steps of 1 as opposed to the step of 50 I used, and use each forecasted point to predict the next point. This will give us far more accurate forecasts.

Additionally, we can also build vector autoregressive models by studying the interaction of the energy demand in Georgia with that of neighboring states or even the weather (since temperature has a high correlation with the utilization of electricity consuming devices)