



# Power Prediction

A Time Series Analysis and Forecast: Can one's household electricity consumption be predicted with a reasonable accuracy?

*Presented by: Mai N. Nguyen*

# Introduction

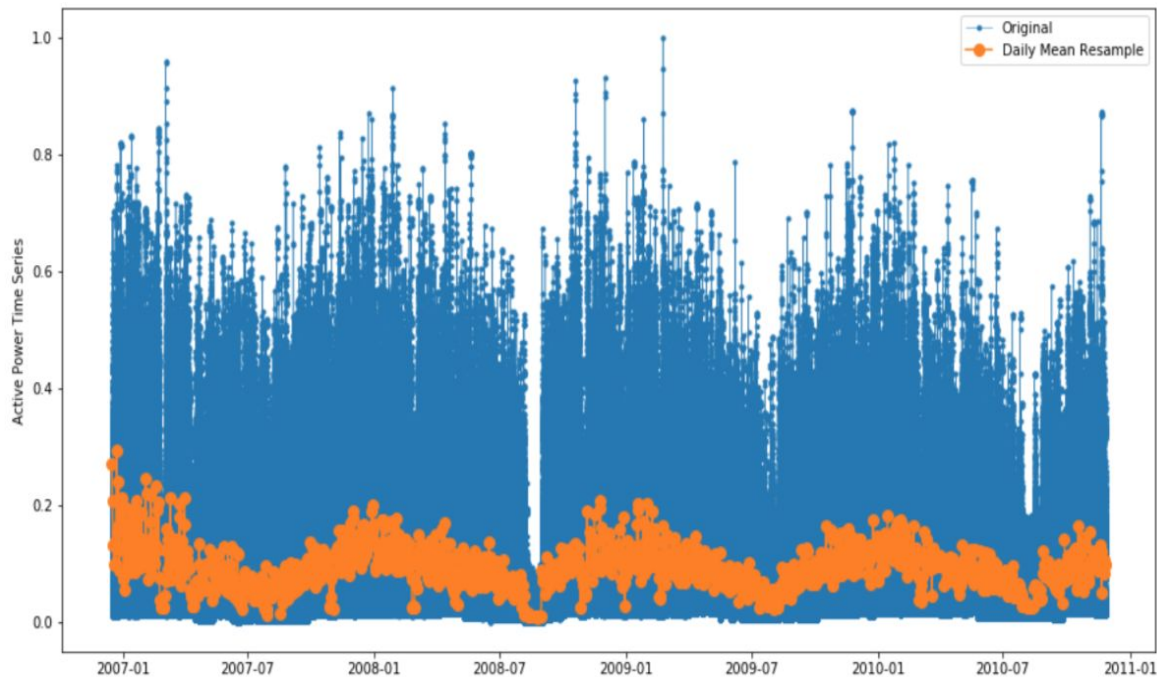


## Why predict electricity consumption?

The ability to forecast electricity consumption based on historical usage data is crucial to effective short-term power load allocation and better long-term infrastructure planning. In addition, prediction for electricity consumption could also play an essential role in enabling proactive energy system planning, reducing operations cost, and enforcing accurate billing.

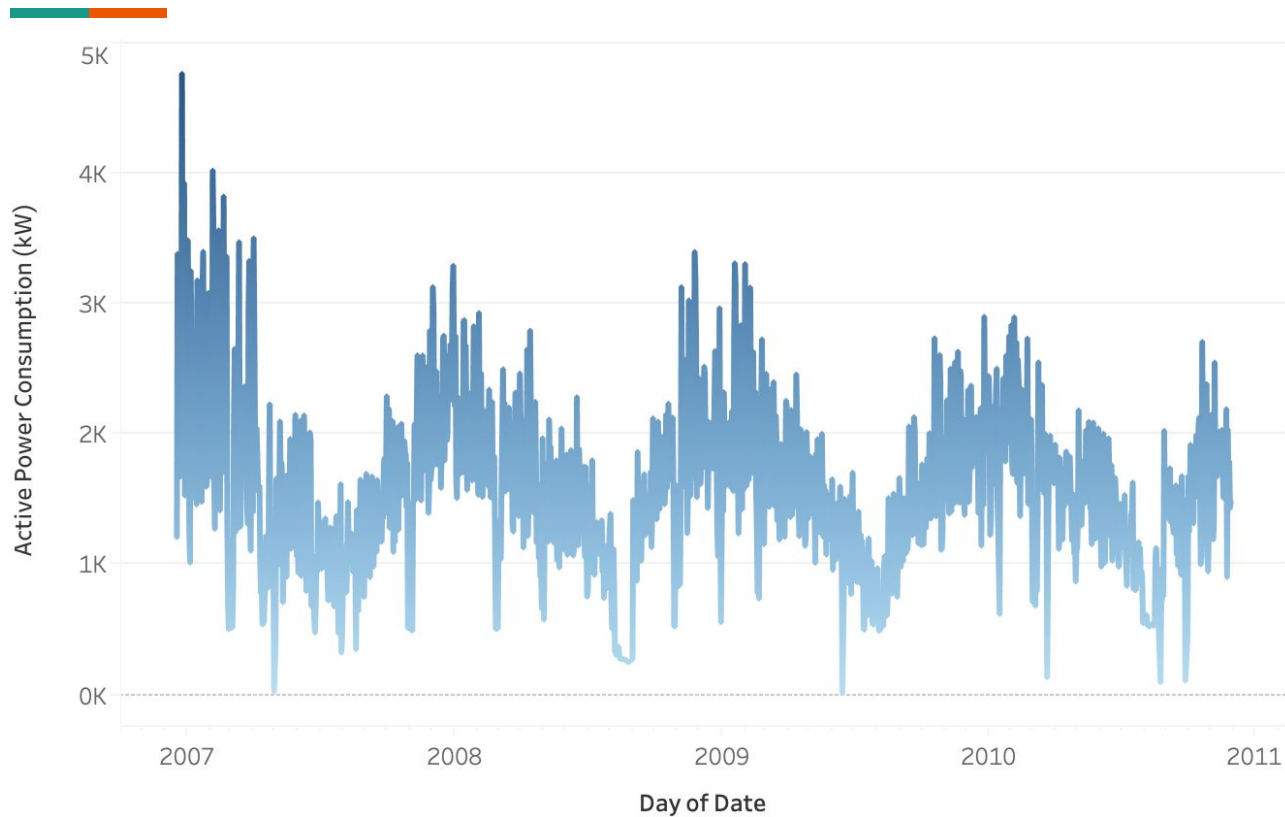
For this project, I'm using the [Household electricity power consumption](#) dataset from UCI Machine Learning Repository. Overall, my ultimate objective would be to answer the following question: **Can the household electricity consumption be predicted with reasonable accuracy?**

# About the dataset

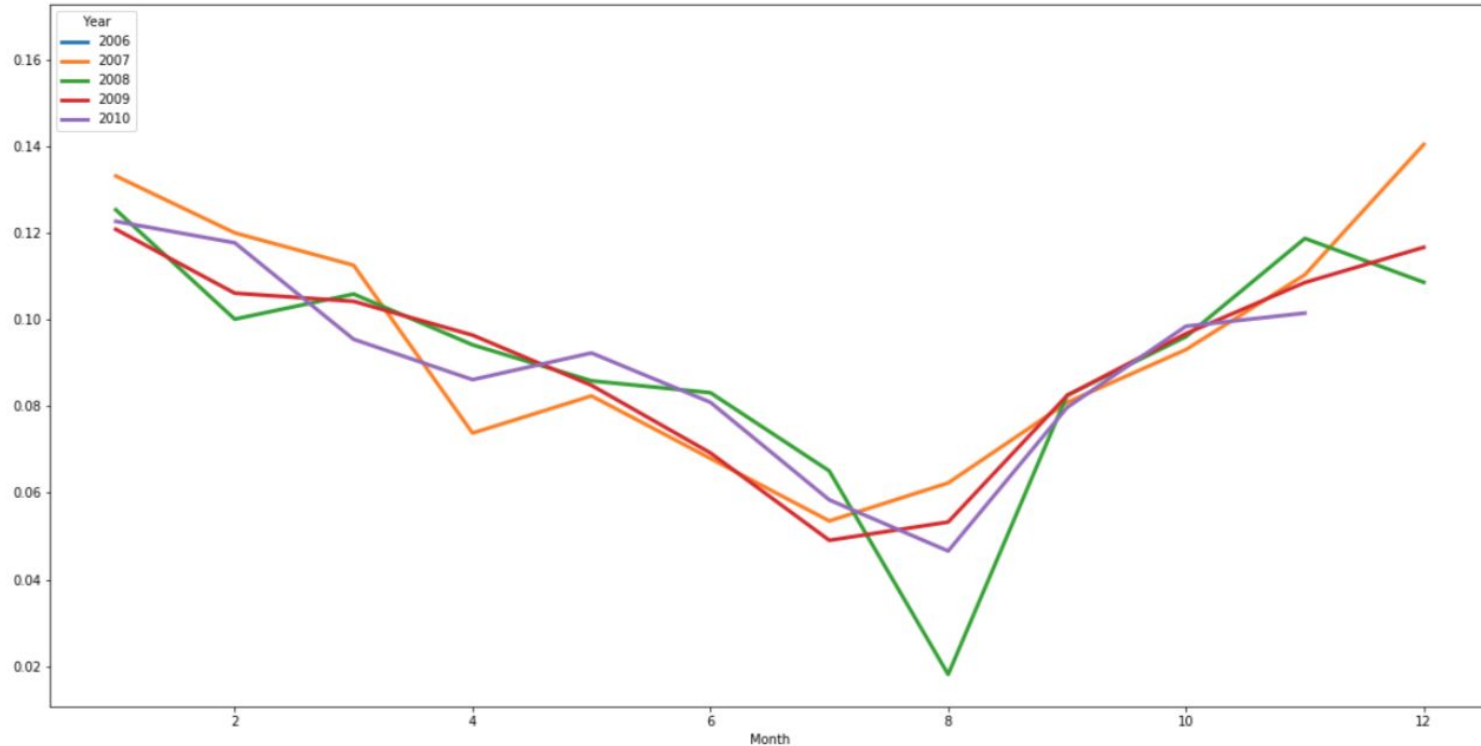


- **> 2 million data points** over 4 years
- Daily resampled: **1440 observations**
- **9 features.**

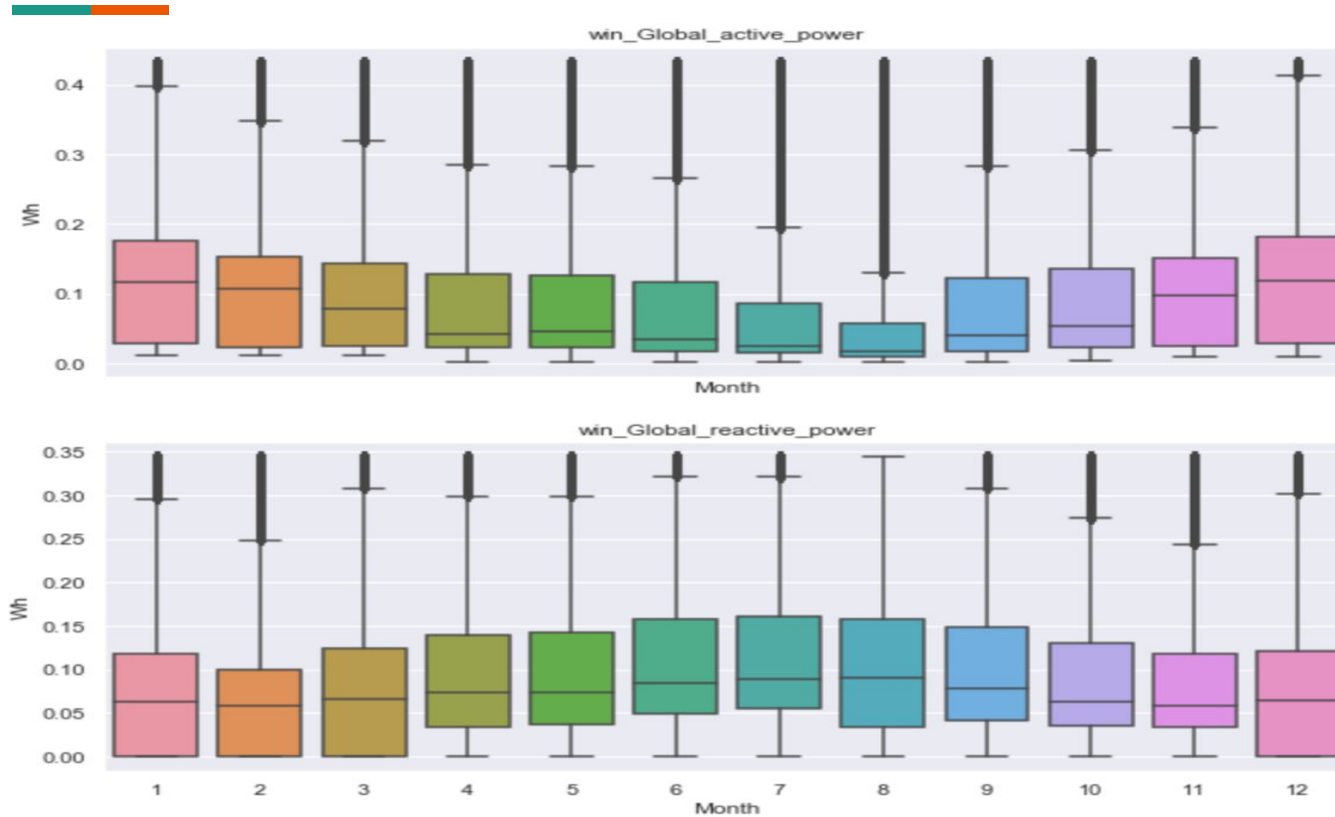
# EDA - The Target Variable



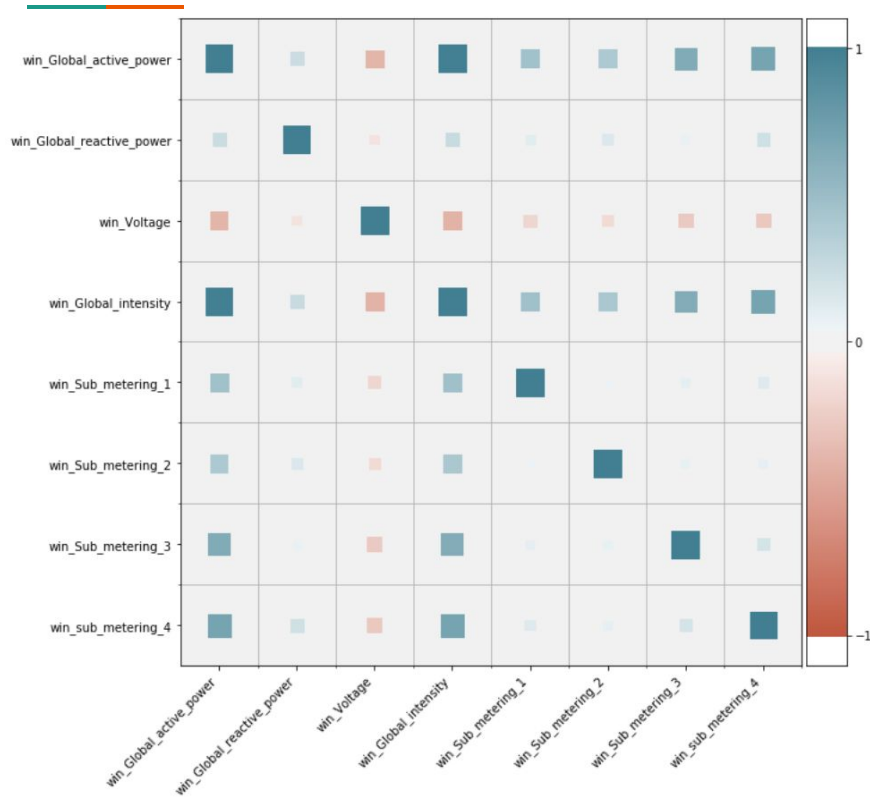
# EDA - Consumption by Year



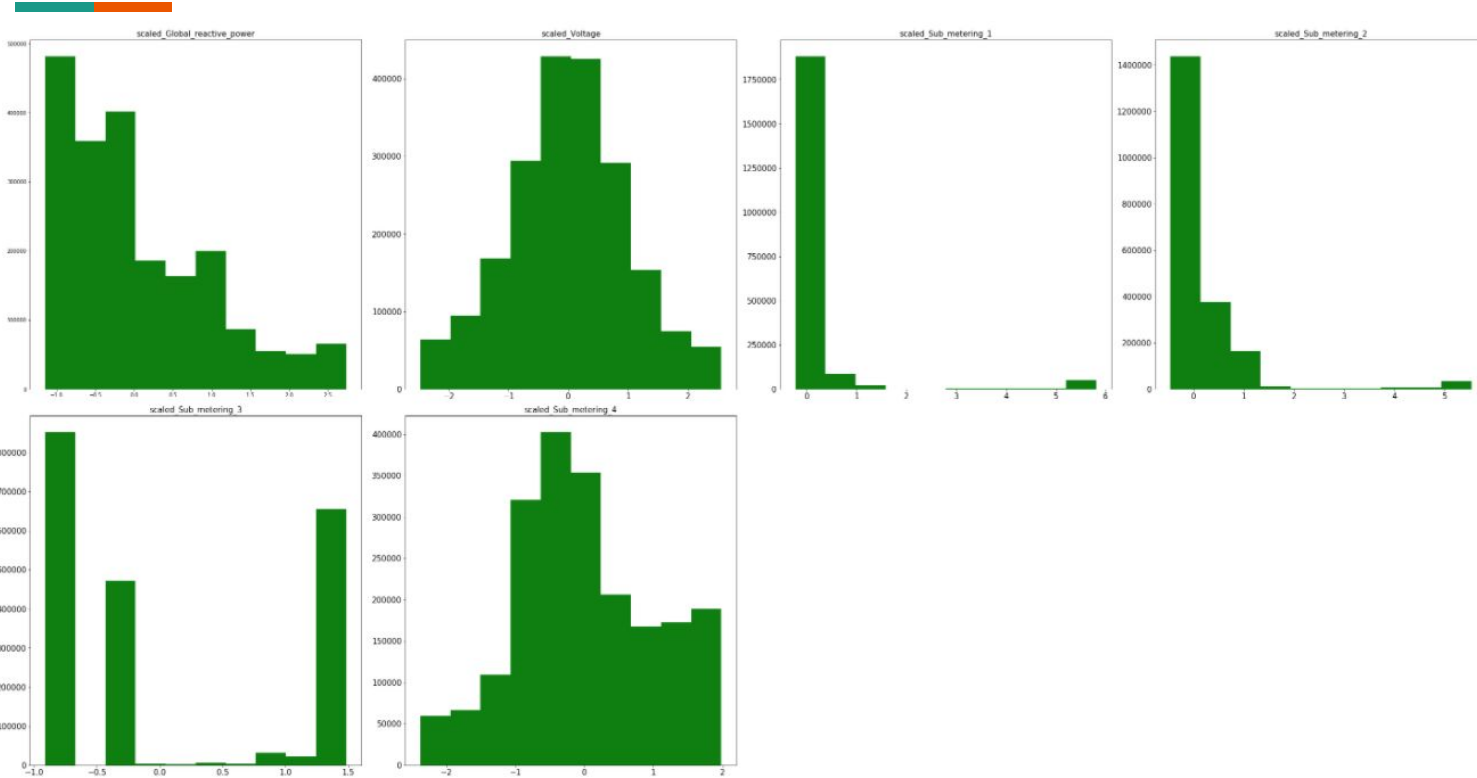
# EDA - Consumption by Year



# EDA - Correlation Matrix



# Feature Engineering - Normalization





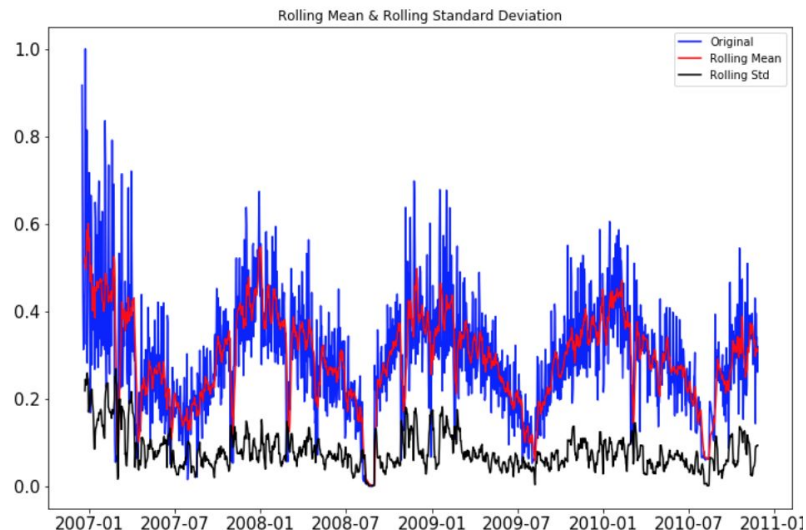
# Building Models



1. Base-line ARIMA
2. Tuned ARIMA
3. Tuned SARIMA
4. Tuned SARIMA vs. Fourier terms
5. Prophet

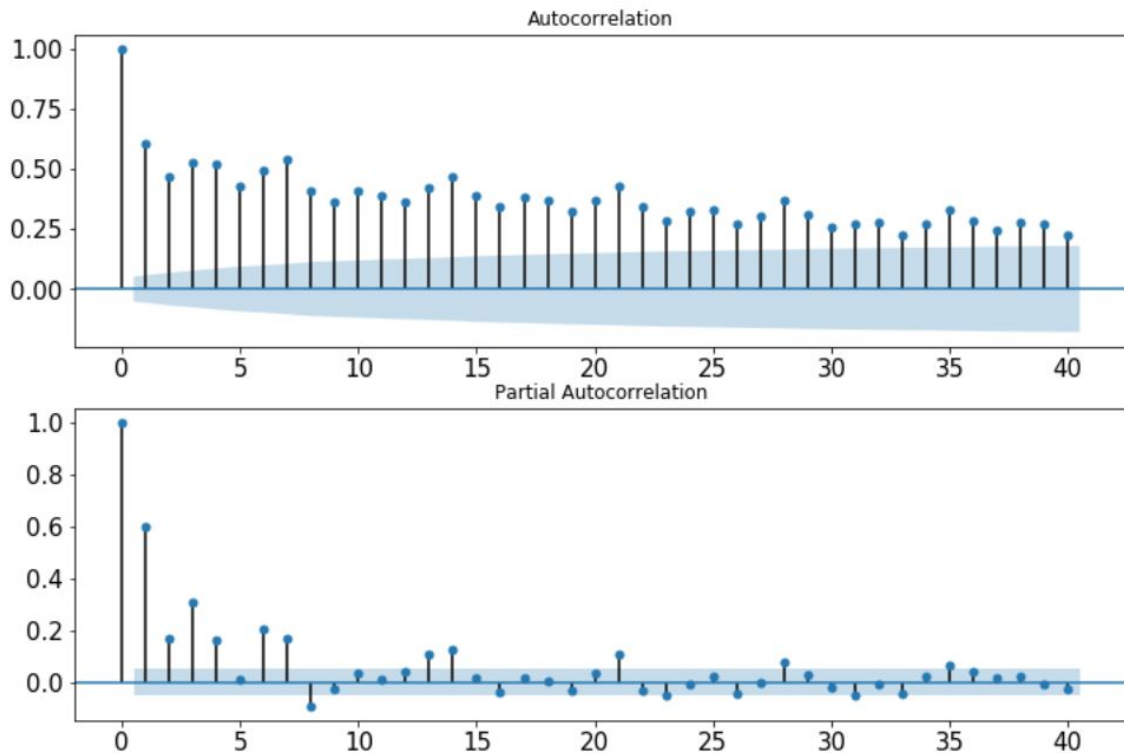
# Stationarity Check

```
test_stationarity(power_daily['daily_power'].dropna())
```

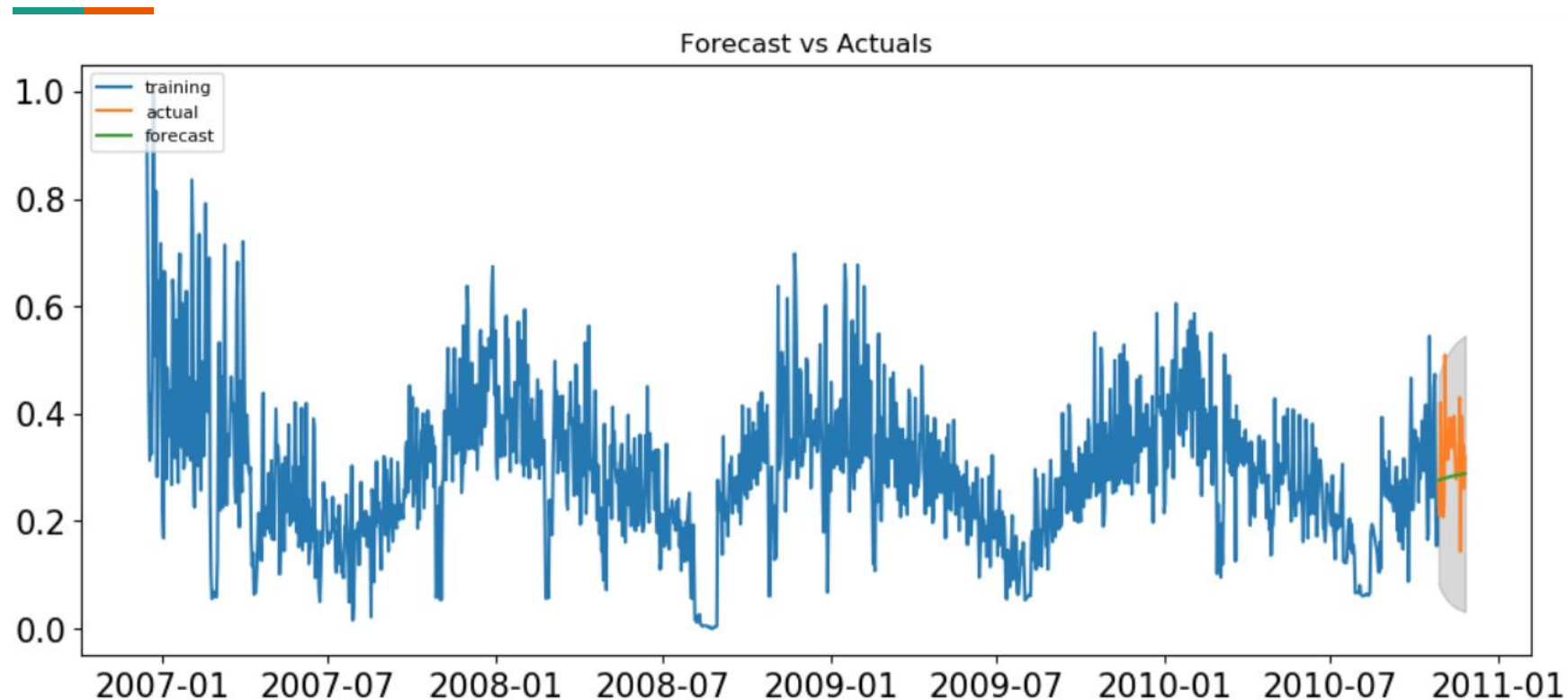


Results of Dickey-Fuller Test:  
p-value = 0.0048. The series is likely stationary.  
Test Statistic -3.653817  
p-value 0.004812  
#Lags Used 20.000000  
Number of Observations Used 1412.000000  
Critical Value (1%) -3.434990  
Critical Value (5%) -2.863589  
Critical Value (10%) -2.567861  
dtype: float64

# Baseline ARIMA (1, 0, 1)



# Baseline ARIMA Performance



# Grid-search ARIMA (3, 0, 5)

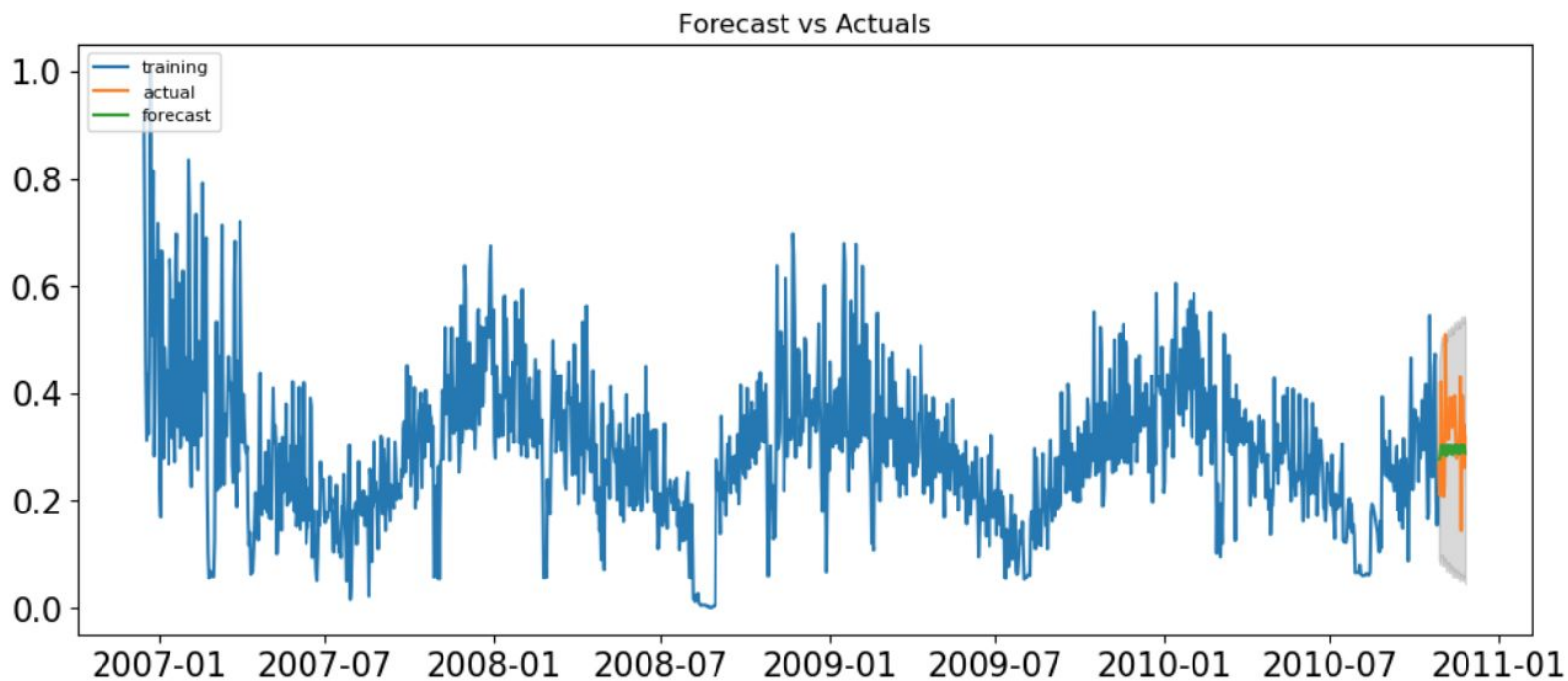
```
auto_model = pm.auto_arima(train_30, start_p=2, start_q=1,
                           test='adf',          # use adftest to find optimal 'd'
                           max_p=5, max_q=10,   # maximum p and q
                           m=1,                 # frequency of series
                           d=None,              # let model determine 'd'
                           seasonal=False,      # No Seasonality
                           start_P=0,
                           D=0,
                           trace=True,
                           error_action='ignore',
                           suppress_warnings=True,
                           stepwise=True)

print(auto_model.summary())
```

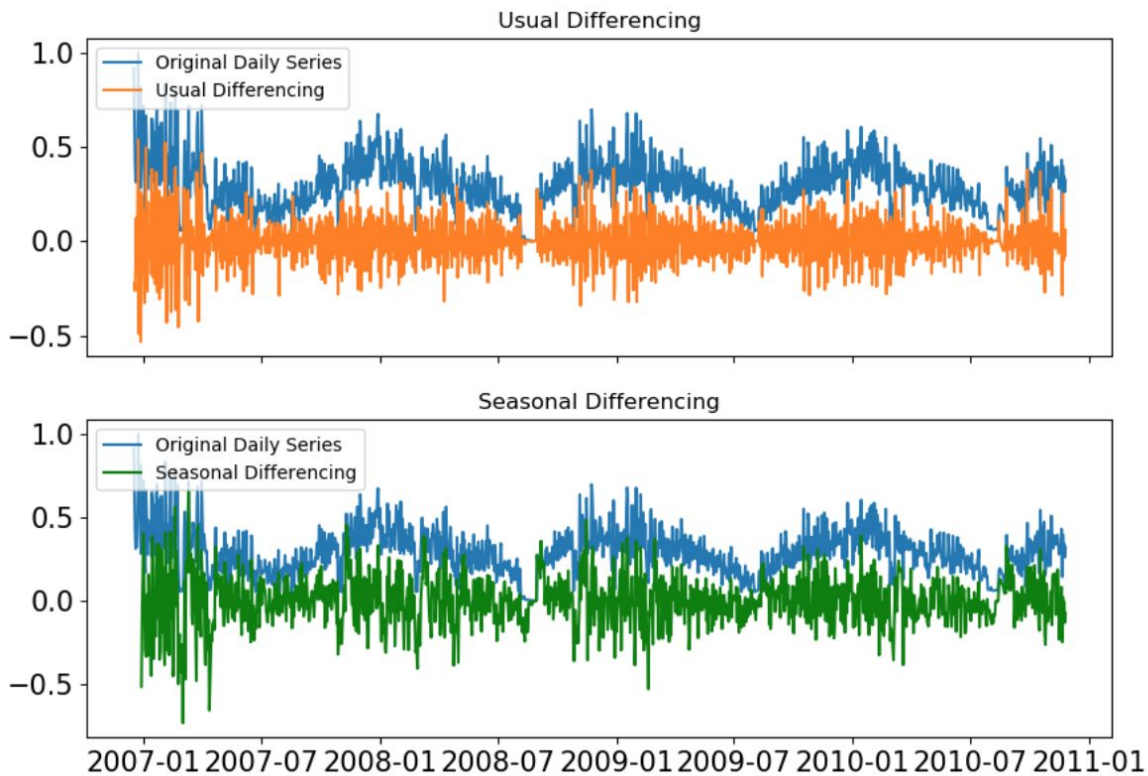
Performing stepwise search to minimize aic

```
Fit ARIMA: (2, 0, 1)x(0, 0, 0, 0) (constant=True); AIC=-2521.610, BIC=-2495.378, Time=1.982 seconds
Fit ARIMA: (0, 0, 0)x(0, 0, 0, 0) (constant=True); AIC=-1639.382, BIC=-1628.889, Time=0.207 seconds
Fit ARIMA: (1, 0, 0)x(0, 0, 0, 0) (constant=True); AIC=-2294.161, BIC=-2278.422, Time=0.366 seconds
Fit ARIMA: (0, 0, 1)x(0, 0, 0, 0) (constant=True); AIC=-2093.779, BIC=-2078.040, Time=0.409 seconds
Fit ARIMA: (0, 0, 0)x(0, 0, 0, 0) (constant=False); AIC=797.823, BIC=803.069, Time=0.086 seconds
Fit ARIMA: (1, 0, 1)x(0, 0, 0, 0) (constant=True); AIC=-2487.648, BIC=-2466.663, Time=1.749 seconds
```

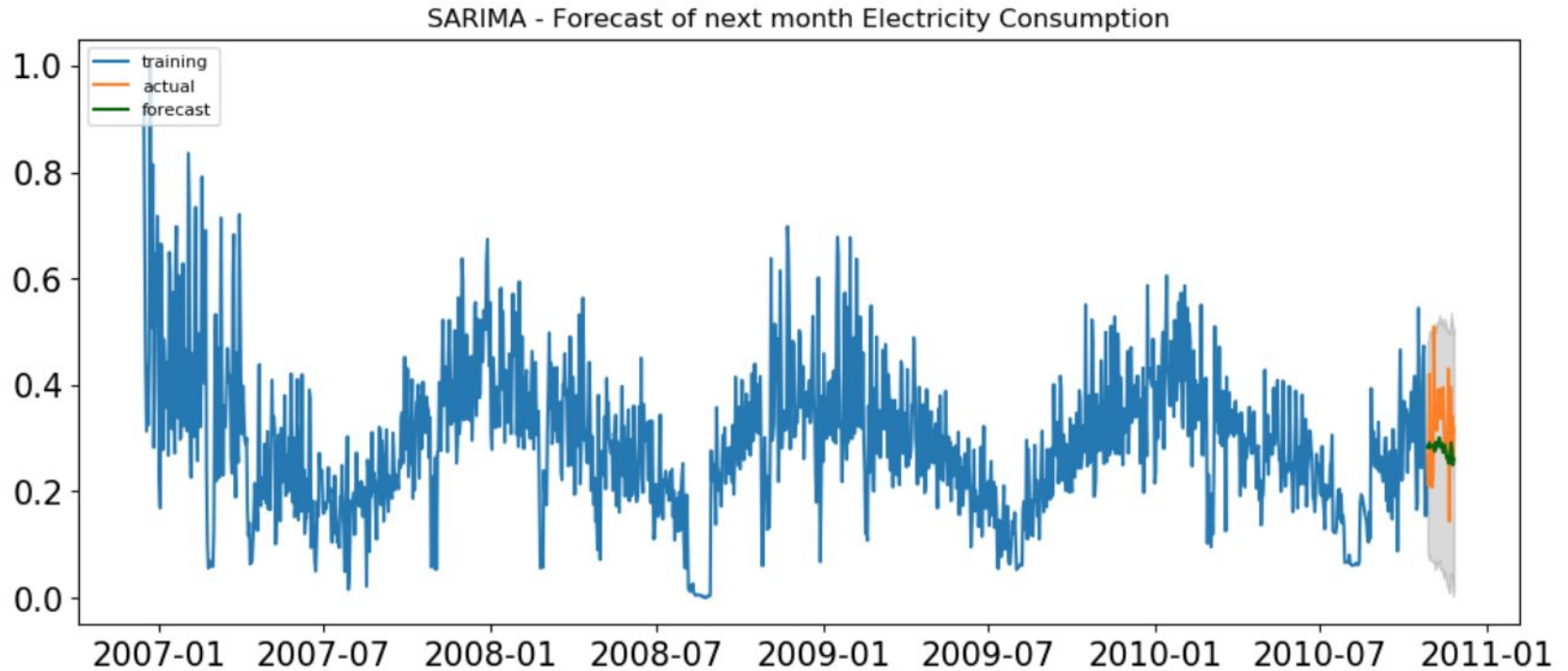
# Tuned ARIMA performance



# Tuned SARIMA



# Tuned SARIMA performance





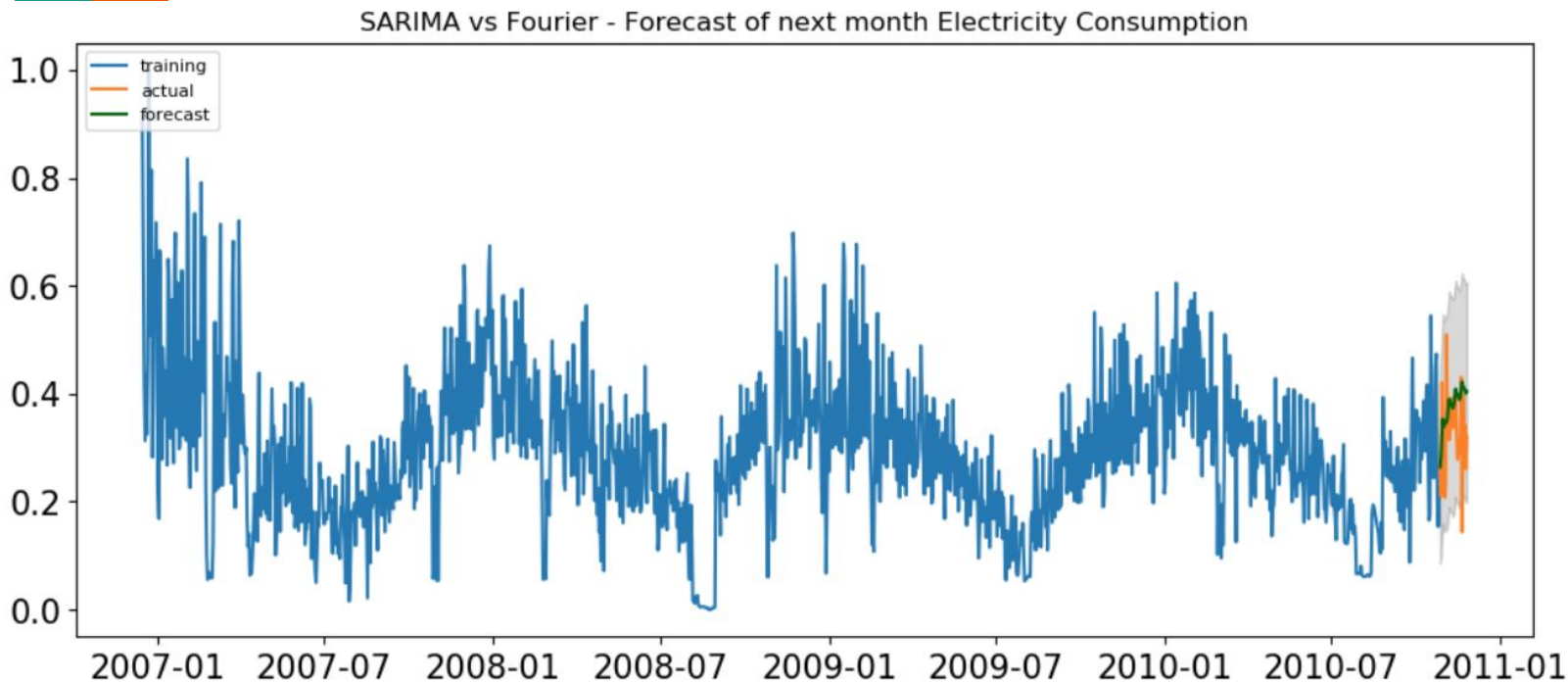
# Tuned SARIMA vs. Fourier terms



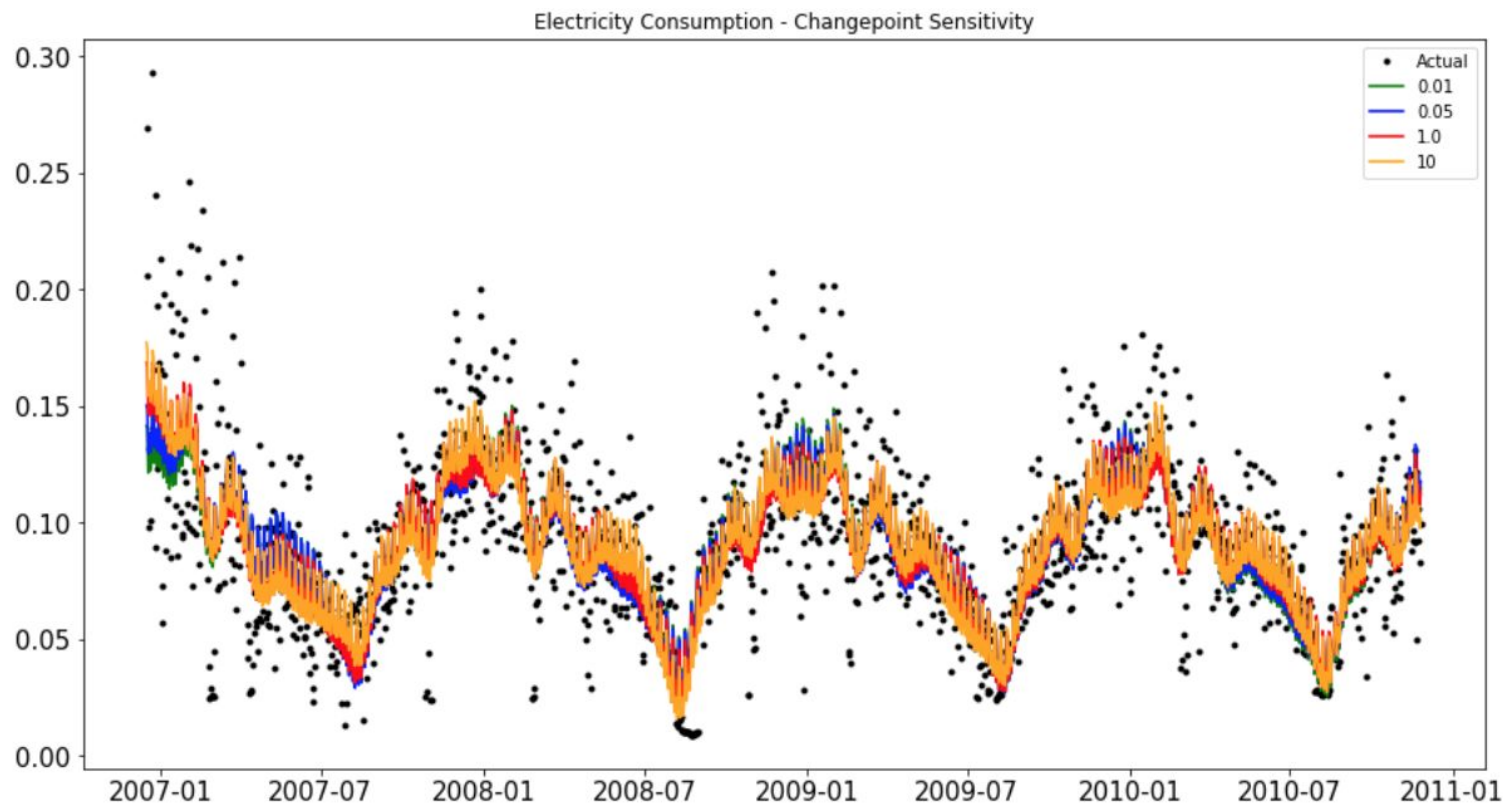
```
# prepare Fourier terms
exog = pd.DataFrame({'date': y.index})
exog = exog.set_index(pd.PeriodIndex(exog['date'], freq='D'))
exog['sin365'] = np.sin(2 * np.pi * exog.index.dayofyear / 365.25)
exog['cos365'] = np.cos(2 * np.pi * exog.index.dayofyear / 365.25)
exog['sin365_2'] = np.sin(4 * np.pi * exog.index.dayofyear / 365.25)
exog['cos365_2'] = np.cos(4 * np.pi * exog.index.dayofyear / 365.25)
exog = exog.drop(columns=['date'])
exog_to_train = exog.iloc[: (len(y)-30)]
exog_to_test = exog.iloc[(len(y)-30):]

# Fit model
sarima_m_model = pm.auto_arima(train_30, exogenous=exog_to_train, m=7, seasonality = True,
                               start_p=1, start_q=1, test='adf', max_p=3, max_q=10,
                               start_P=0, d=None, D=None, trace=True,
                               error_action='ignore', suppress_warnings=True, stepwise=True)
```

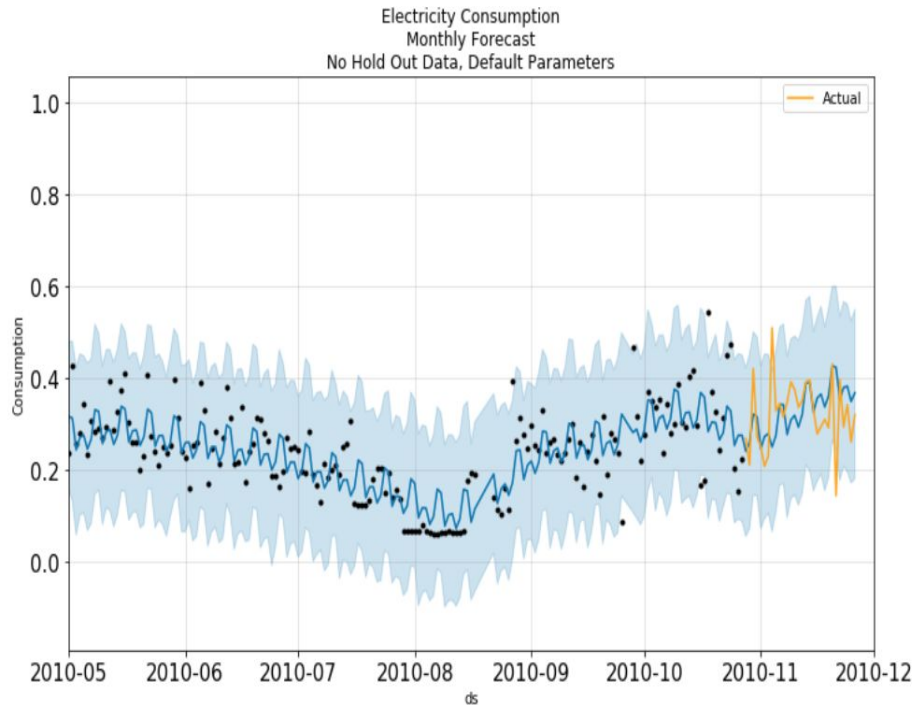
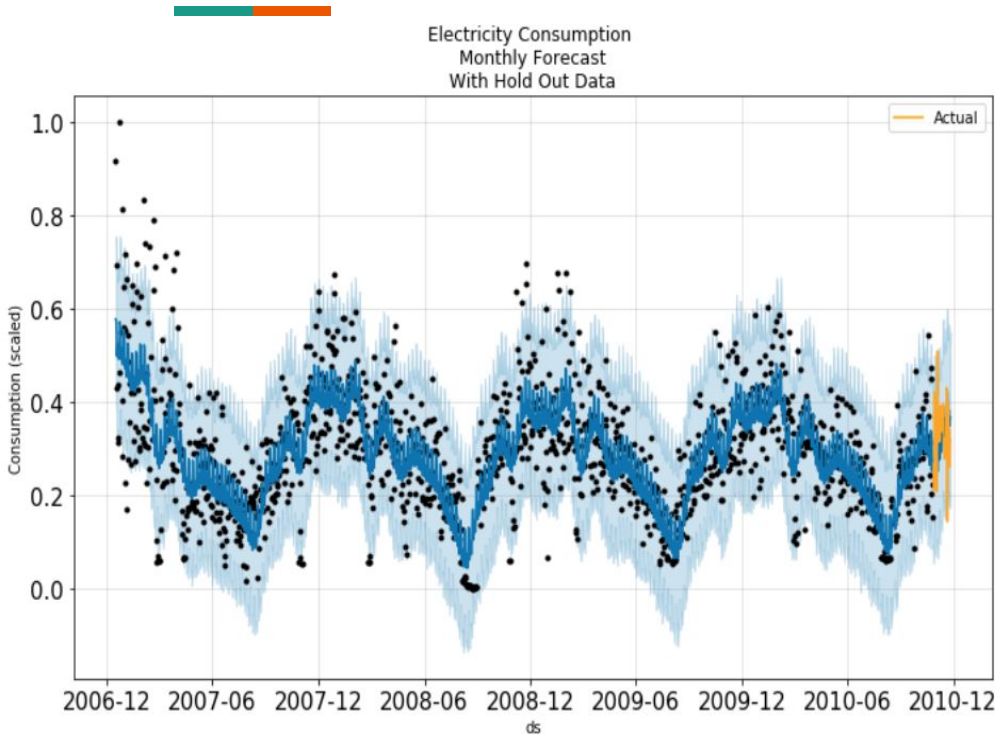
# SARIMA vs. Fourier performance



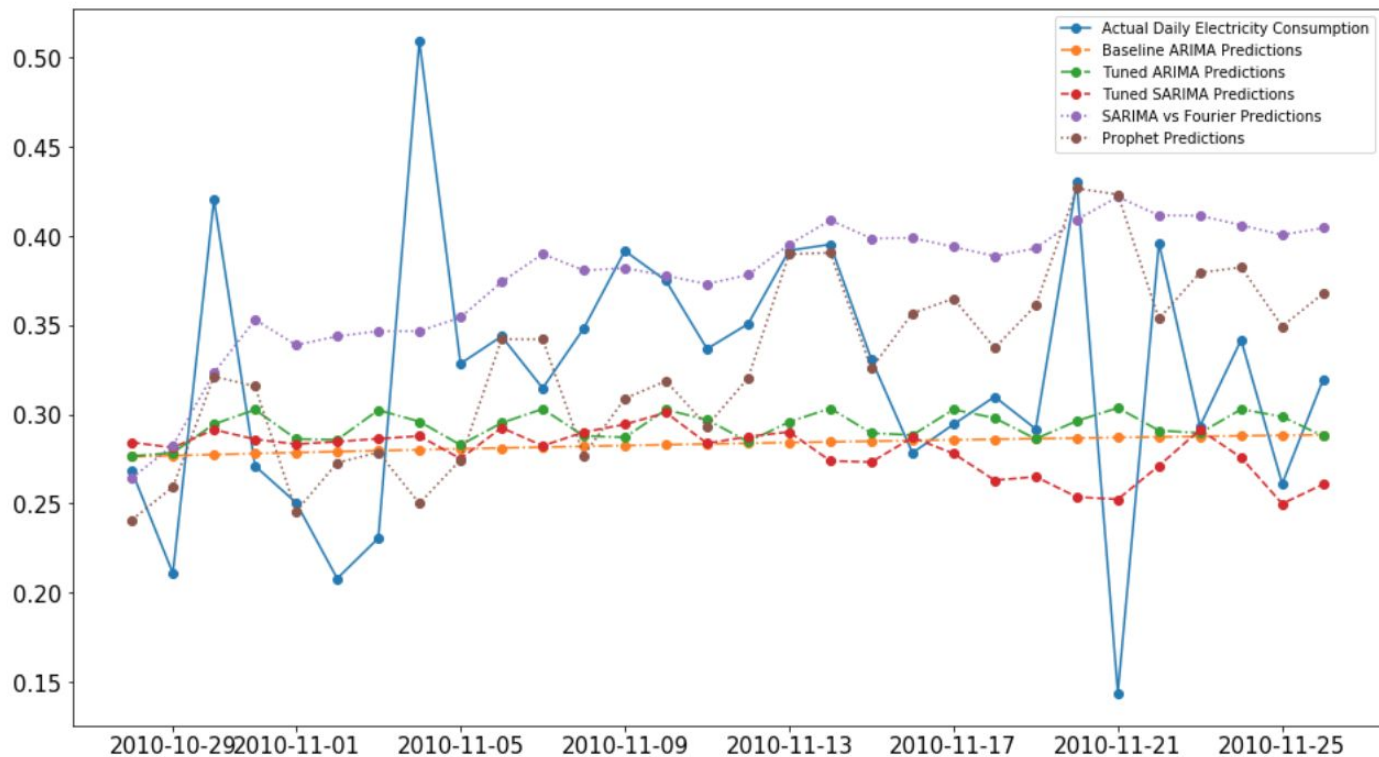
# Prophet



# Prophet Performance

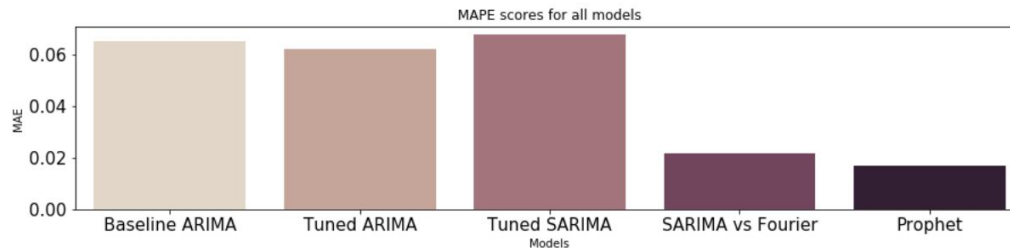
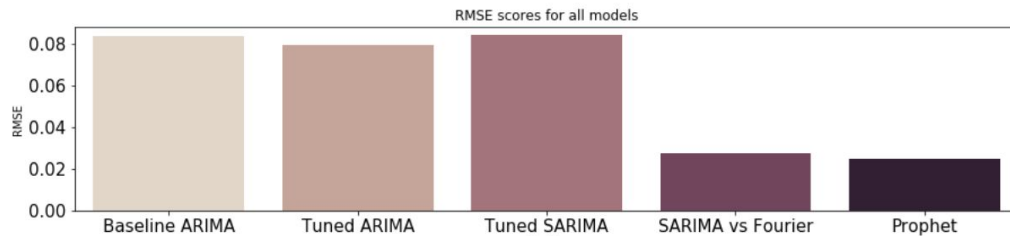
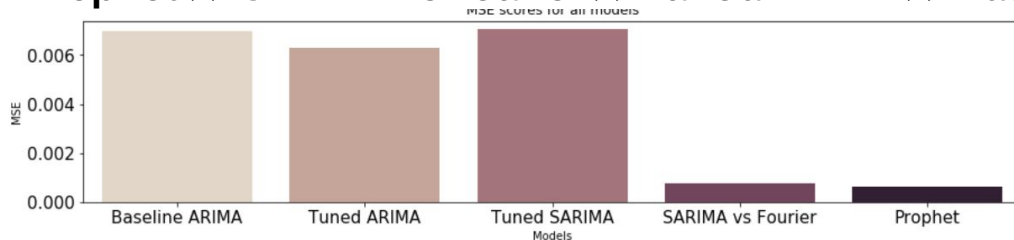


# Summary - Predictions Chart



# Summary - Evaluation Metrics

**Prophet >> SARIMA vs Fourier >> Tuned ARIMA >> Baseline ARIMA >> Tuned SARIMA**



# Final Words



**Prophet prevails.** It consistently has the best scores out of all models. Prophet not only provides us with the most accurate and robust model, its runtime is also the shortest.

**The next best option is SARIMA with an additional Fourier term as the exogenous variable.** Out of all the ARIMA models, this is the only model that was able to capture the yearly seasonality.

What's also surprising to me is how the **SARIMA model performs worst**, and worse than both of its simpler counterparts: the baseline ARIMA and tuned ARIMA. It doesn't seem like it's worth the trouble of going through the lengthy grid-search (since SARIMA also has more parameters than ARIMA, and its complexity increases as the m-variable increases). One possible explanation might be that we weren't able to use  $m=365$  to truly let SARIMA capture the yearly pattern; but then the extremely long time if we run the model with  $m=365$  also makes it not worth it.

It is also important to note that all evaluation metrics for **monthly predictions are better than those for quarterly predictions**. This is common in Time series forecasting, as most time series models predict from its previous prediction and stack up the errors from these prediction.

We've covered a lot of ground in this project. Ultimately, our original research question is **Can the household electricity consumption be predicted with reasonable accuracy?** An optimistic answer would be **Yes**. Although none of the model has perfect accuracy, I think our best performing model performance is definitely robust enough to have a good estimation of electricity consumption for the next month, either for effective short-term power load allocation and better long-term infrastructure planning.