Predicting Ride Duration and Ride Demand of NYC Taxi Cabs

Problem Statement

- Given various data about geolocation, trip duration, and the weather, it would be beneficial for both the taxi driver and the passengers to know when and where the taxi demand would be high, and also to know how long a trip can be expected to take.
- By predicting the appropriate amount of time a ride can be expected to take, both the customers and the taxi drivers can optimally plan the use of their time spent in transit.

The Data

- The primary dataset was acquired from Kaggle : https://www.kaggle.com/c/nyc-taxi-trip-duration/data the dataset contains more than 1.4 million entries recording taxi trips detailing columns such as the following: unique taxi vendor id, pick-up and drop-off geolocation coordinates, trip duration, passenger count, and more.

- Second, complementary data set: https://www.kaggle.com/mathijs/weather-data-in-new-york-city-2016 contains the entries regarding data gathered from a weather station in Central Park for the first six months of 2016. For each day, there are relevant columns pertaining to the minimum temperature, maximum temperature, average temperature, precipitation, new snow-fall, and current snow depth.

Data Wrangling

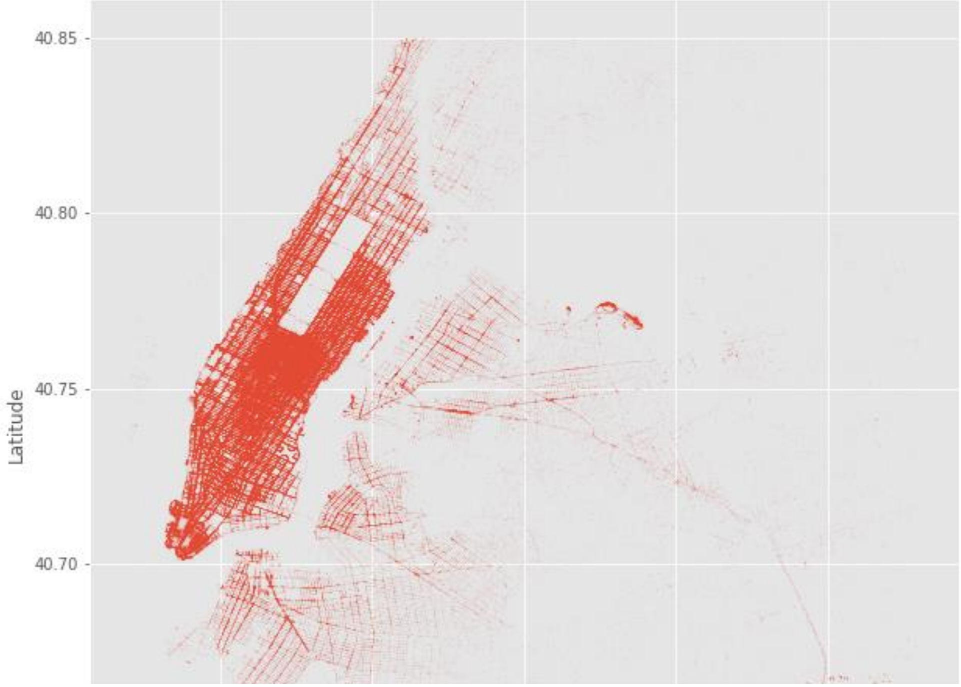
1. The columns of “pickup_datetime”, “dropoff_datetine” and “pickup_date” were all converted to datetime modules using pandas.
2. There were some obviously erroneous trips, where the recorded trip duration was more than 979 hours long. In order to account for a more “normal” data, all trips deviating further than 2 stand deviations away from the mean of the trip duration were disregarded.
3. Other datetime related variables were made from the pick up time such as month, day of week, and hour.
4. The weather data was cleaned such that precipitation, snow-fall, and snow depth was obtained. There was a value ‘T’ for trace amounts, which was converted to 0. After these values were converted to float types, the weather data was merged by date to the NYC taxi data along with the minimum temperature of each day.
5. The longitude and latitude borders of New York City was found through a simple internet search (an array of -74.03, 40.63, -73.77, 40.85) and data points were further filtered to be within these city limits. 1.43 million observations remained in total.

Feature Engineering

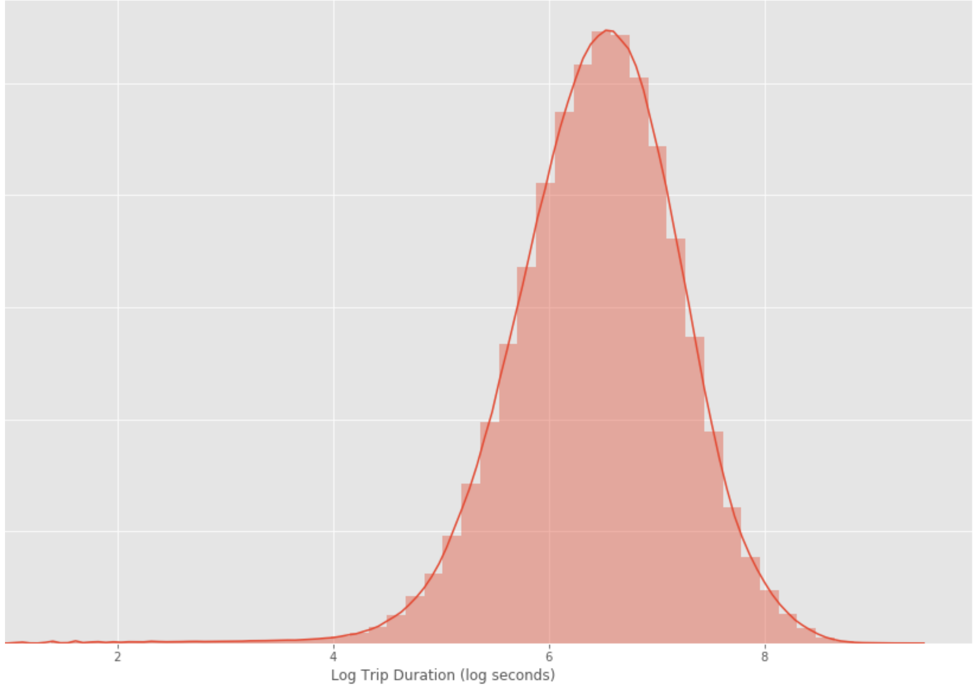
- Distance metrics such as the haversine distance, Manhattan distance, and distance bearing were found by applying the appropriate equations to the given pickup/dropoff longitudes and latitudes.
  - Link to relevant equations: http://www.movable-type.co.uk/scripts/latlong.html

Insights and Analysis

- We can see that a high density of rides originating within the island of Manhattan as expected.
- Looking at our main predictor variable of log trip duration, we see that it is normally distributed with a mean of around 6.5 log-seconds.



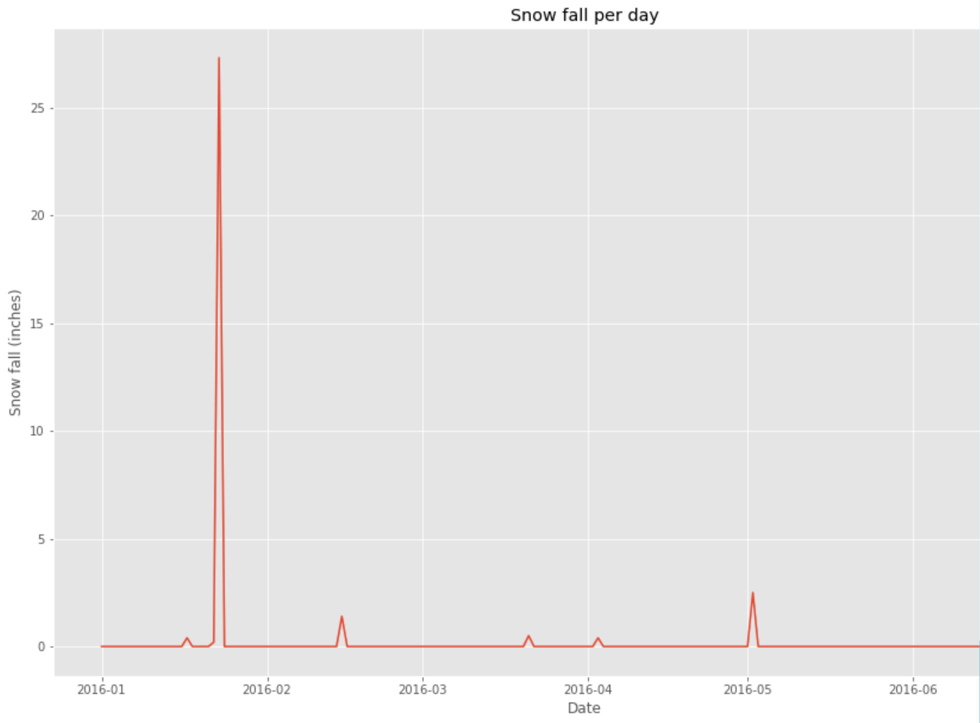
Scatterplot of Rides

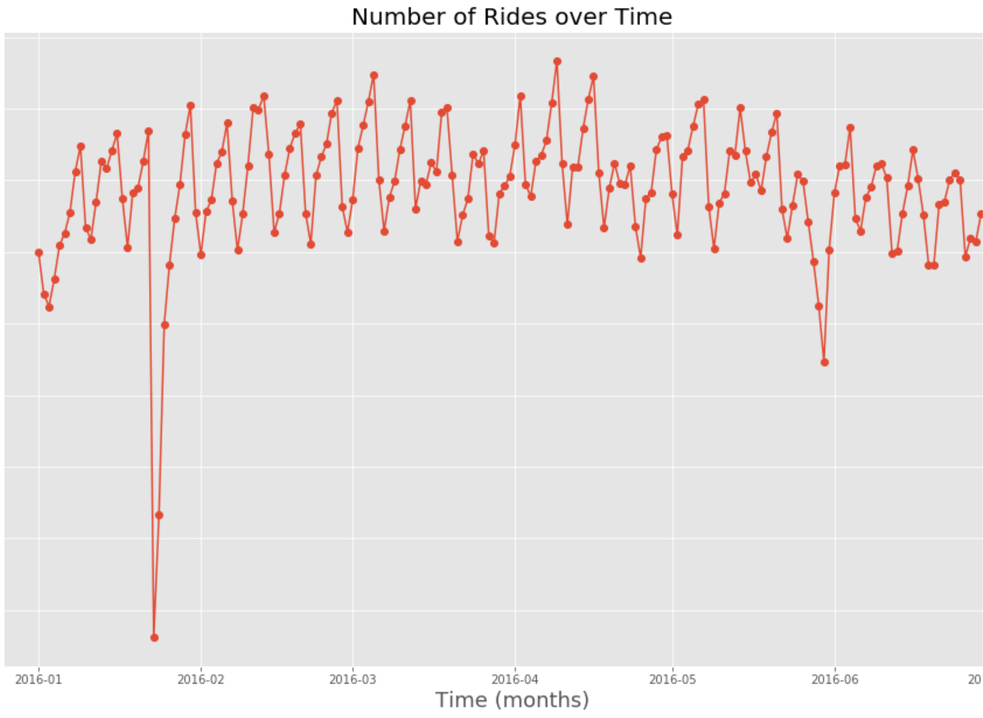


Distribution of ride durations

Insights and Analysis

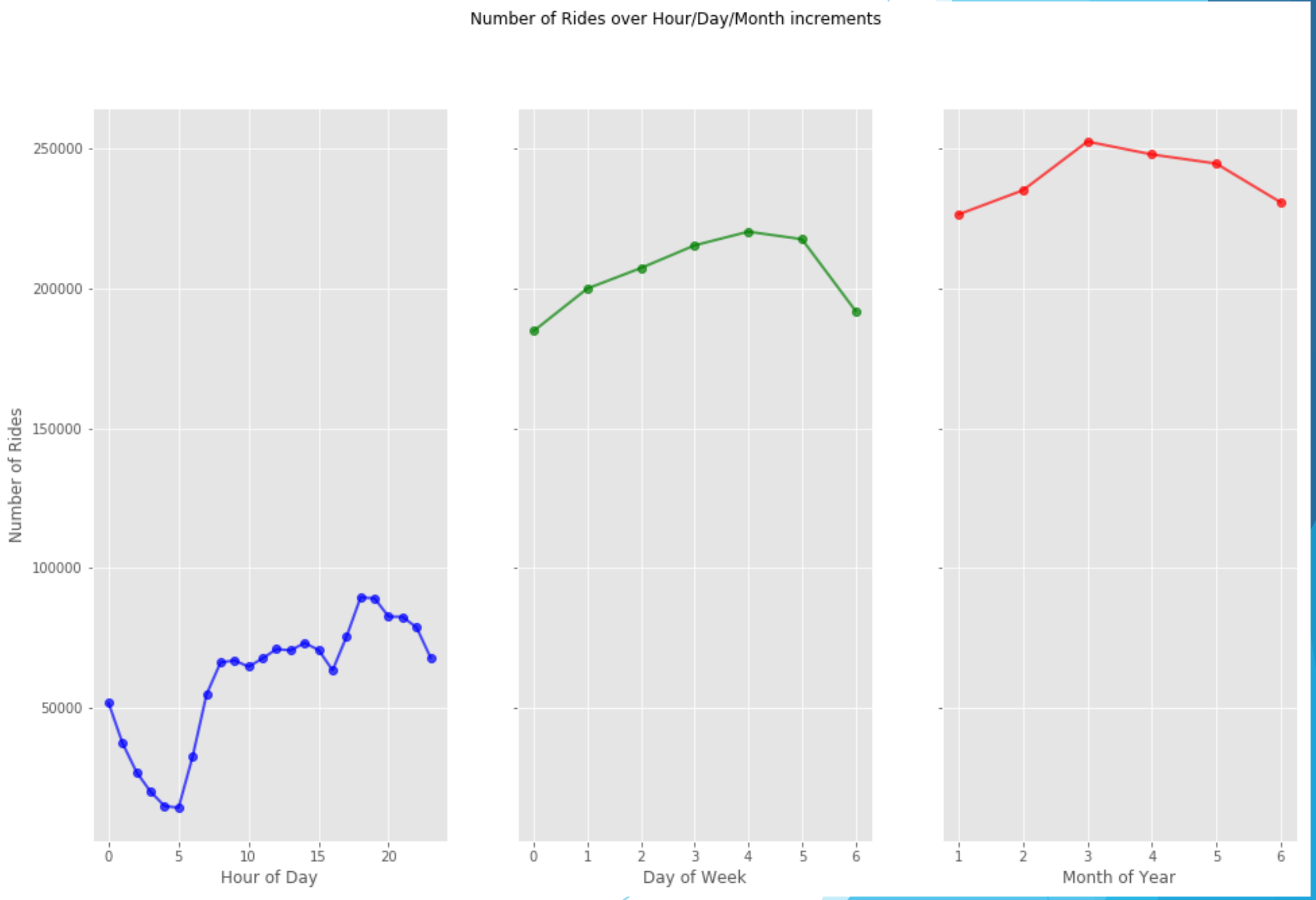
- As for the dip in the number of rides in late January, we must think of outside factors that may have caused this significant dip in the number of rides.
- Since January is in the middle of Winter, we may expect there to be extremely inclement snow related weather conditions that may have resulted in a city-wide driving ban.
- Looking at the average snowfall per day plot, we can confirm that this is indeed the case, with there being 27 inches of snow fall on that same day.





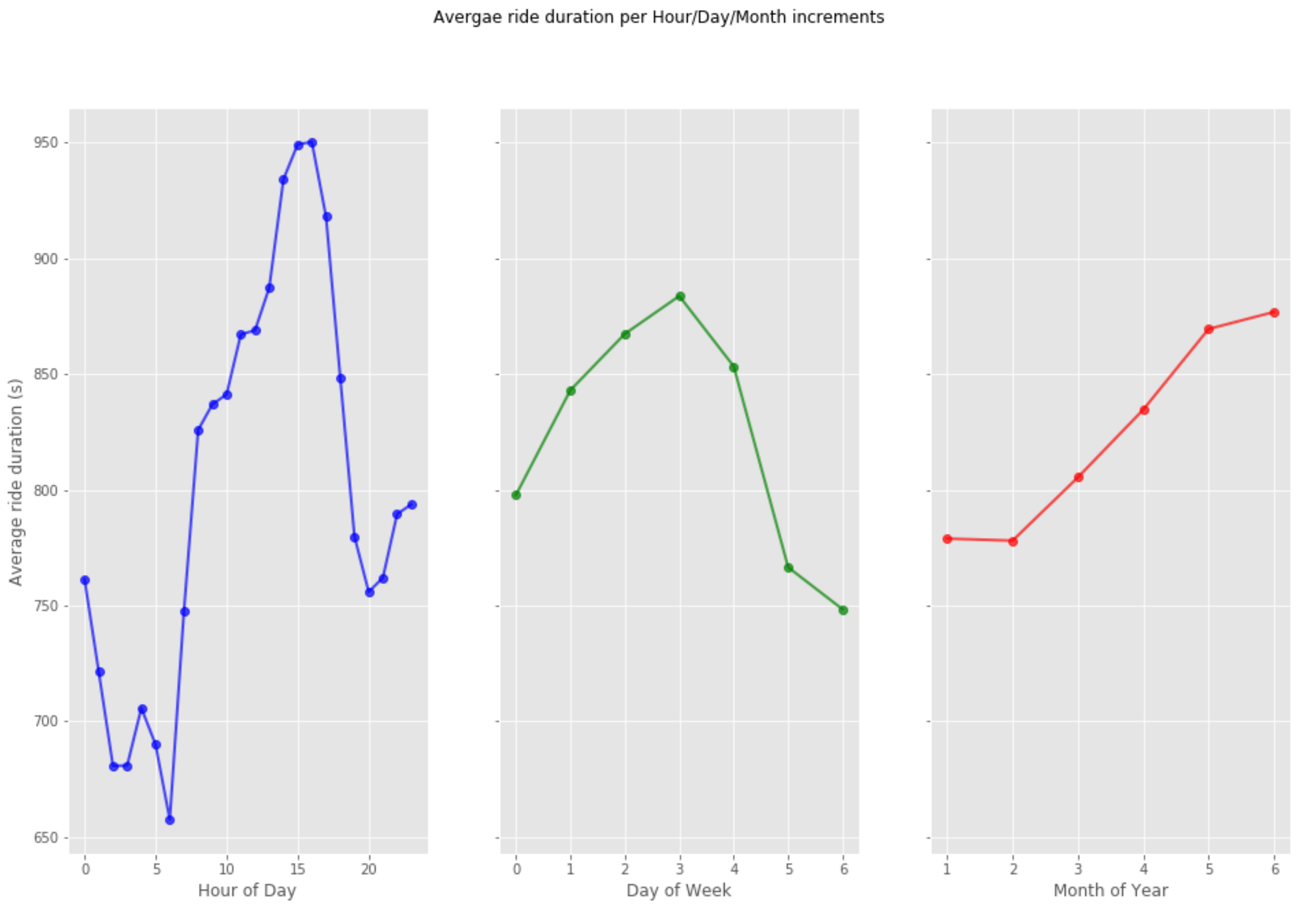
Average Number of Ride Trends

- The trends of the number of rides over different increments of time may be important in determining when a customer or taxi driver should plan to be active.
- The number of rides taken over the day hits a minimum at 5am and a maximum at around 6pm.
- The number of rides taken over the days of the week is at a minimum on Monday (0 on the x axis) and gradually rises to peak on Friday (4) before dropping sharply on Sunday.
- Over the months of the first half of the year, January has the least number of rides, while March has the most.
- However, this January count of rides was significantly affected by the lack of rides on the 23rd, and the following days also serviced quite a bit below the average number of rides.



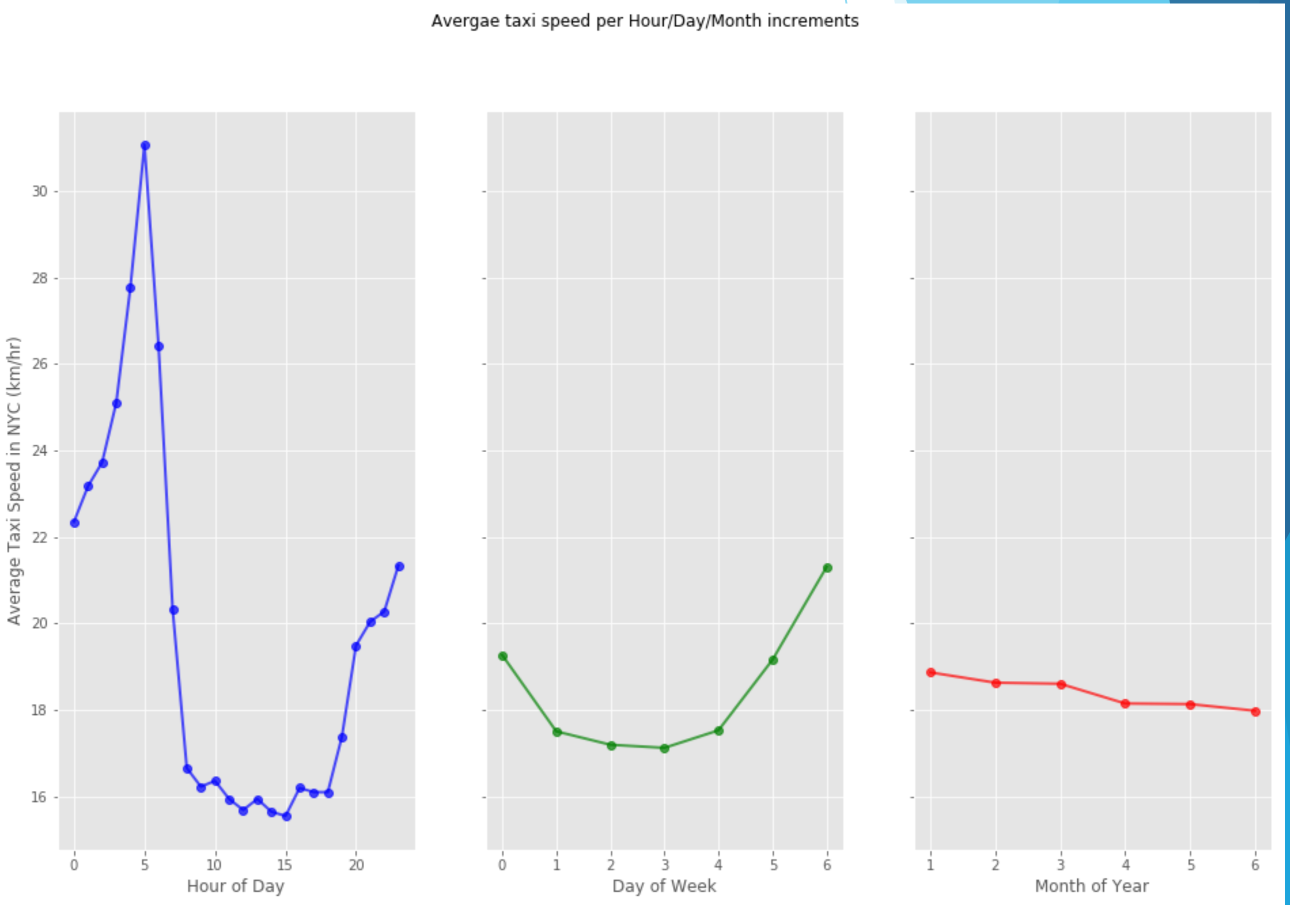
Average Ride Duration Trends

- During the day, the shortest trips also occur around 5am, when there are the fewest number of trips. This may be due to the lack of traffic at the hour, and the lack of need for people to go "far" in a cab at that hour.
- The average ride duration seems to peak at around 3pm each day, and continues to decrease in duration until around 8pm. Surprisingly enough, the longest average ride durations did not occur at the customary "rush hour" times of 5-6pm.
- During the week, Sunday has the lowest average ride duration followed by Saturday. Somewhat intuitively, more people have to work and have a pressing need for cabs during the weekdays, and that may be attributed to the higher taxi ride durations that gradually peak on Thursdays.
- On average, people are in taxis for less amount of times on weekends. Finally, over the course of the first 6 months of the year, the average ride duration is the shortest during January-February, and grows almost linearly until June (when the average ride duration is at highest).



Average Taxi Speed Trend

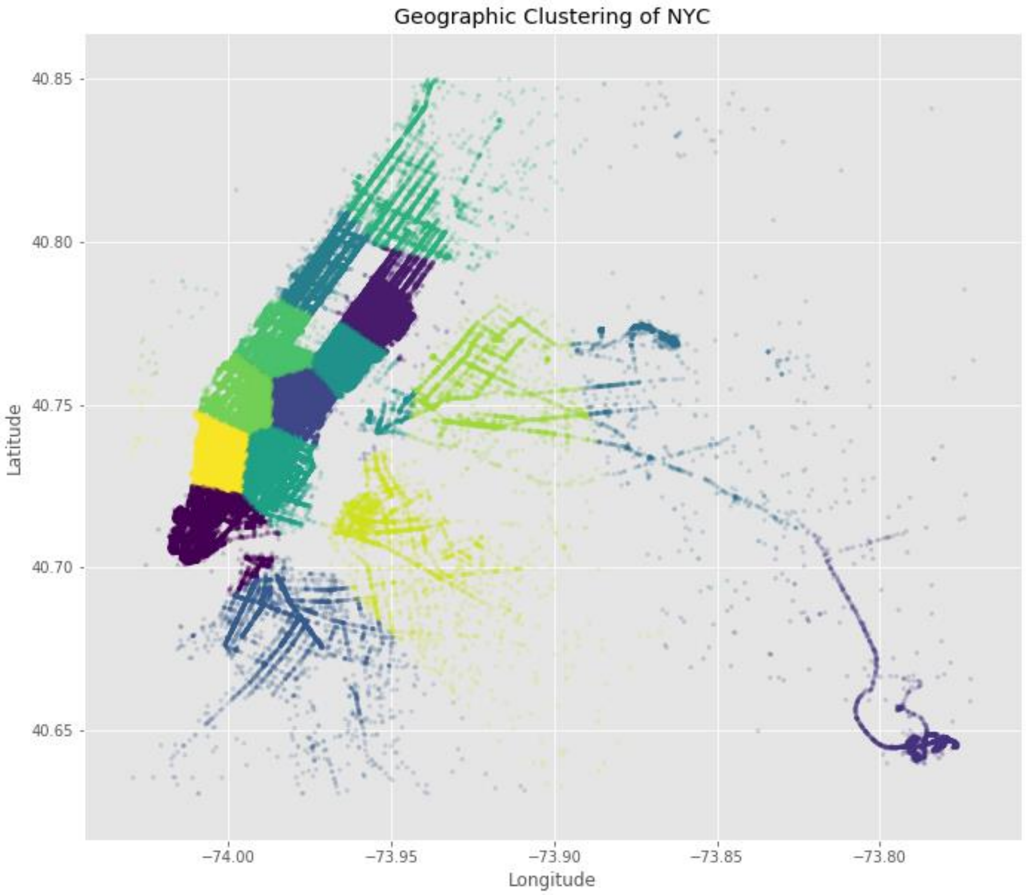
- Here we can see that during the day, the taxis are cruising at their fastest average speeds of around 31 km/hr at 5 am, which we know to be a time of day where there are the fewest number of cars around.
- Otherwise during the day, starting at around 8am, the cars crawl around 16 km/hr until 6pm, where the speed gradually climbs back up until 5 am the next day.
- During the week, Sundays have the fastest taxi speeds on average. This is again the day of the week that we know to have the fewest number of cars around.
- As for the month, the highest taxi speeds are in January, and it gradually declines until the average taxi speeds hit their minimum in June. This seems somewhat counter-intuitive, since one would think that taxi speeds would be lower during the colder months where there are snow and other adverse weather conditions on the road.

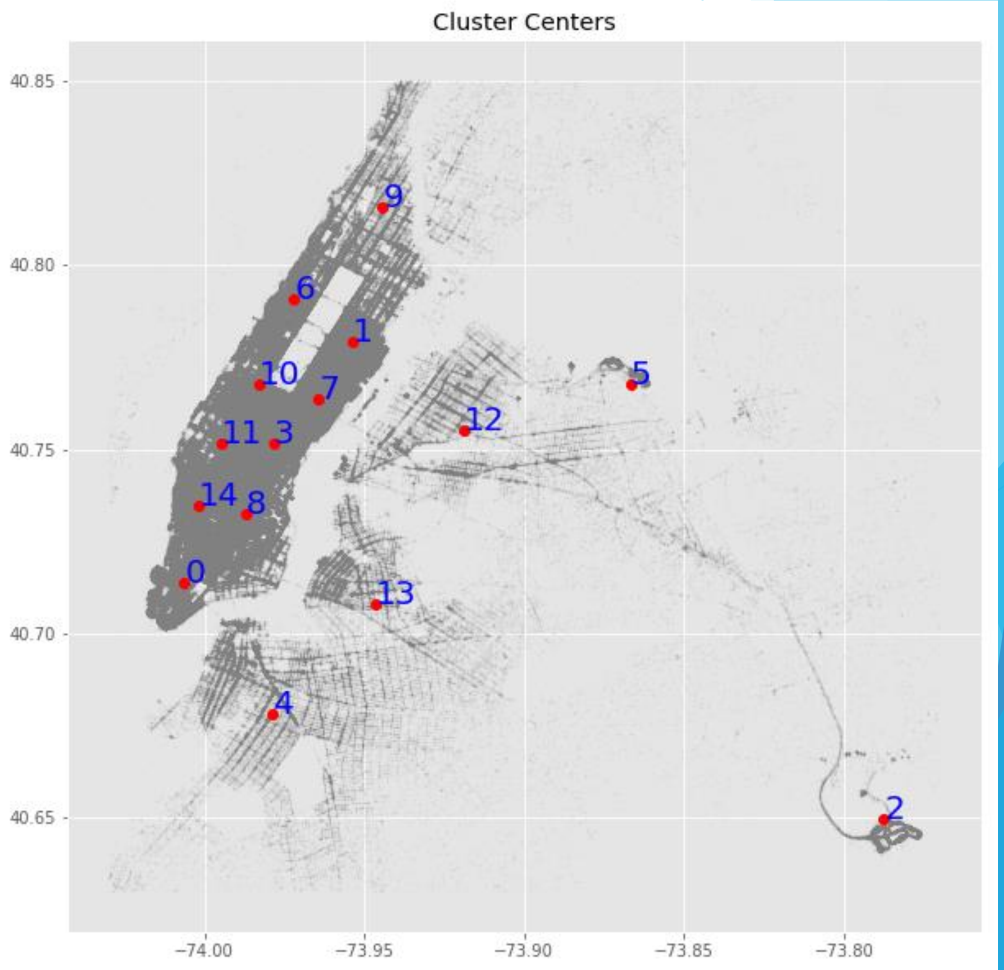


Avergae taxi speed per Hour/Day/Month increments

KMeans Clustering

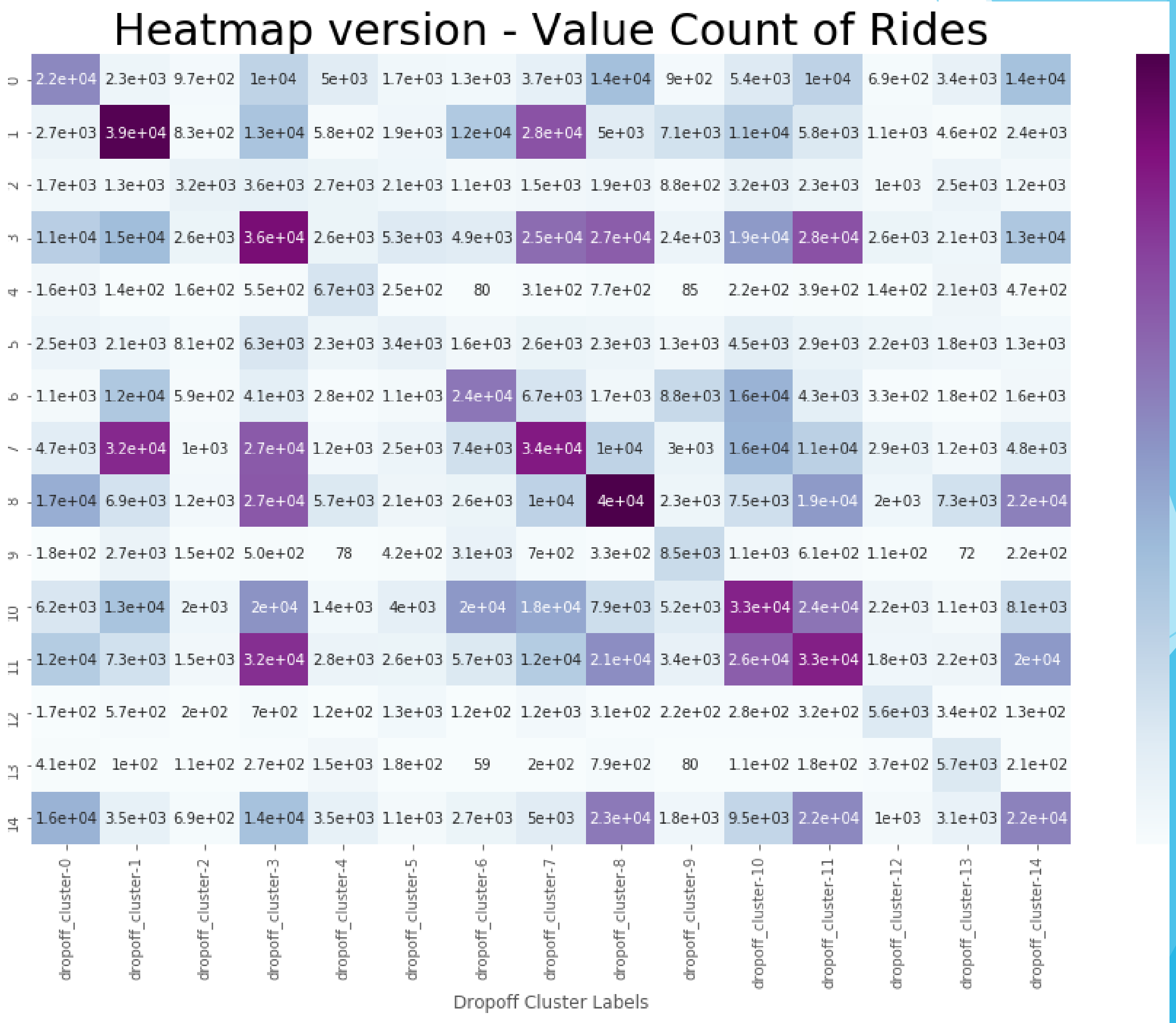
- KMeans Clustering algorithm was used to cluster the taxi rides taken within NYC based on geolocation coordinates into 15 clusters.





Number of Rides To and From each Cluster

- The heatmap can be interpreted as following: the row labels indicate the pickup_cluster labels. the column labels indicate the corresponding dropoff_cluster label. The values are the value counts of the taxi ride instances that the pickup_cluster label resulted in the corresponding dropoff_cluster.
- For example, 21673 rides were picked up at cluster 0 and dropped off at cluster 0 (top left corner of the table). Then 2734 rides were picked up at cluster 1 and dropped off at cluster 0 (the cell immediately below the top right corner of the table). The table follows such format etc.



Heatmap version - Value Count of Rides

	dropoff_cluster-0	dropoff_cluster-1	dropoff_cluster-2	dropoff_cluster-3	dropoff_cluster-4	dropoff_cluster-5	dropoff_cluster-6	dropoff_cluster-7	dropoff_cluster-8	dropoff_cluster-9	dropoff_cluster-10	dropoff_cluster-11	dropoff_cluster-12	dropoff_cluster-13	dropoff_cluster-14
0	2.2e+04	2.3e+03	9.7e+02	1e+04	5e+03	1.7e+03	1.3e+03	3.7e+03	1.4e+04	9e+02	5.4e+03	1e+04	6.9e+02	3.4e+03	1.4e+04
1	2.7e+03	3.9e+04	8.3e+02	1.3e+04	5.8e+02	1.9e+03	1.2e+04	2.8e+04	5e+03	7.1e+03	1.1e+04	5.8e+03	1.1e+03	4.6e+02	2.4e+03
2	1.7e+03	1.3e+03	3.2e+03	3.6e+03	2.7e+03	2.1e+03	1.1e+03	1.5e+03	1.9e+03	8.8e+02	3.2e+03	2.3e+03	1e+03	2.5e+03	1.2e+03
3	1.1e+04	1.5e+04	2.6e+03	3.6e+04	2.6e+03	5.3e+03	4.9e+03	2.5e+04	2.7e+04	2.4e+03	1.9e+04	2.8e+04	2.6e+03	2.1e+03	1.3e+04
4	1.6e+03	1.4e+02	1.6e+02	5.5e+02	6.7e+03	2.5e+02	80	3.1e+02	7.7e+02	85	2.2e+02	3.9e+02	1.4e+02	2.1e+03	4.7e+02
5	2.5e+03	2.1e+03	8.1e+02	6.3e+03	2.3e+03	3.4e+03	1.6e+03	2.6e+03	2.3e+03	1.3e+03	4.5e+03	2.9e+03	2.2e+03	1.8e+03	1.3e+03
6	1.1e+03	1.2e+04	5.9e+02	4.1e+03	2.8e+02	1.1e+03	2.4e+04	6.7e+03	1.7e+03	8.8e+03	1.6e+04	4.3e+03	3.3e+02	1.8e+02	1.6e+03
7	4.7e+03	3.2e+04	1e+03	2.7e+04	1.2e+03	2.5e+03	7.4e+03	3.4e+04	1e+04	3e+03	1.6e+04	1.1e+04	2.9e+03	1.2e+03	4.8e+03
8	1.7e+04	6.9e+03	1.2e+03	2.7e+04	5.7e+03	2.1e+03	2.6e+03	1e+04	4e+04	2.3e+03	7.5e+03	1.9e+04	2e+03	7.3e+03	2.2e+04
9	1.8e+02	2.7e+03	1.5e+02	5.0e+02	78	4.2e+02	3.1e+03	7e+02	3.3e+02	8.5e+03	1.1e+03	6.1e+02	1.1e+02	72	2.2e+02
10	6.2e+03	1.3e+04	2e+03	2e+04	1.4e+03	4e+03	2e+04	1.8e+04	7.9e+03	5.2e+03	3.3e+04	2.4e+04	2.2e+03	1.1e+03	8.1e+03
11	1.2e+04	7.3e+03	1.5e+03	3.2e+04	2.8e+03	2.6e+03	5.7e+03	1.2e+04	2.1e+04	3.4e+03	2.6e+04	3.3e+04	1.8e+03	2.2e+03	2e+04
12	1.7e+02	5.7e+02	2e+02	7e+02	1.2e+02	1.3e+03	1.2e+02	1.2e+03	3.1e+02	2.2e+02	2.8e+02	3.2e+02	5.6e+03	3.4e+02	1.3e+02
13	4.1e+02	1e+02	1.1e+02	2.7e+02	1.5e+03	1.8e+02	59	2e+02	7.9e+02	80	1.1e+02	1.8e+02	3.7e+02	5.7e+03	2.1e+02
14	1.6e+04	3.5e+03	6.9e+02	1.4e+04	3.5e+03	1.1e+03	2.7e+03	5e+03	2.3e+04	1.8e+03	9.5e+03	2.2e+04	1e+03	3.1e+03	2.2e+04

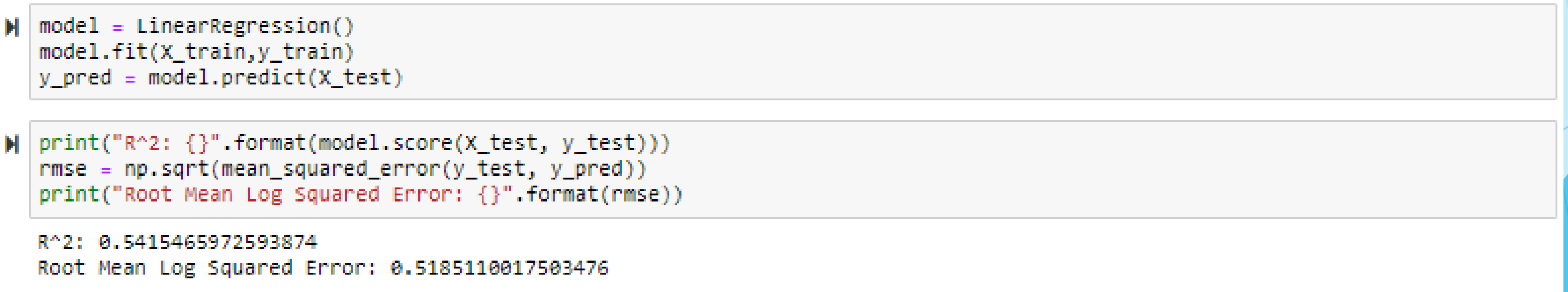
Models: Predicting Log Trip Duration

- Since we have a variety of data types within the dataset, we must wrangle them accordingly to get them ready for modelling.

1. Get rid of irrelevant columns such as "id“
2. Only use log_trip_duration instead of both the former and trip_duration
3. Use one hot encoding (pandas get_dummies) for categorical variables such as "vendor_id", "passenger_count", various date columns (day, week, month), and cluster labels
4. Randomly split the data, assign predictor and target variables and train!

Models: Predicting Log Trip Duration

- Linear Regression
- For the simple linear regression model, the Root Mean Log Squared Error is still quite high at approximately 0.5185. The highest positive valued coefficients of the linear regression model is the vendor ids, and several of the dropoff cluster labels. Similarly, on the other end of the spectrum, the most negative valued coefficients of the linear regression model were some of the passenger count dummy variables, followed by the month dummy variables.



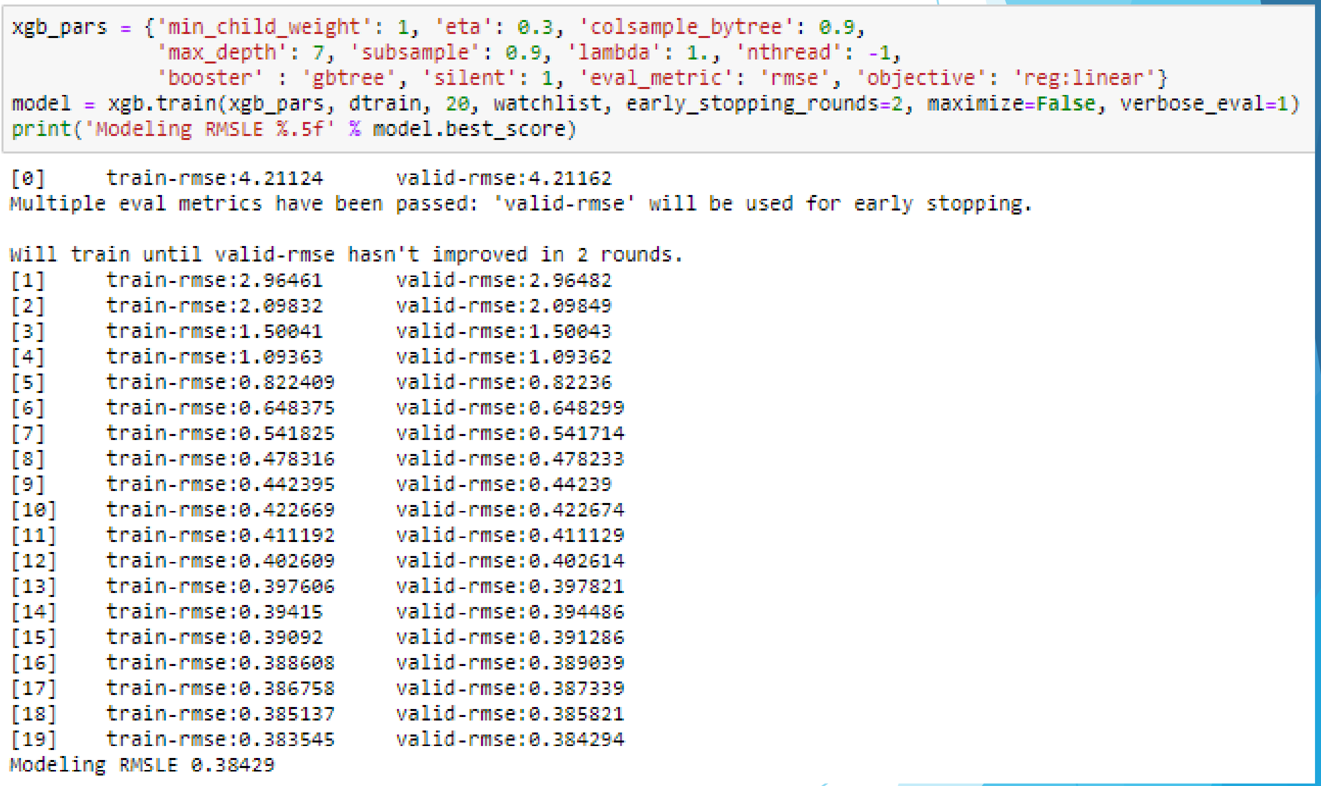
```
model = LinearRegression()
model.fit(X_train,y_train)
y_pred = model.predict(X_test)
```

```
print("R^2: {}".format(model.score(X_test, y_test)))
rmse = np.sqrt(mean_squared_error(y_test, y_pred))
print("Root Mean Log Squared Error: {}".format(rmse))
```

```
R^2: 0.5415465972593874
Root Mean Log Squared Error: 0.5185110017503476
```

Models: Predicting Log Trip Duration

- XGBoost: Gradient Boosted Trees for Regression
- The XGBoost model was able to get the Root Mean Log Squared Error of the test set down to 0.384, a considerable improvement from the simple linear regression model. We can see that the model performs as well as it did on the training set as it did on the test set (randomly split set, same as above for the lin-reg model).

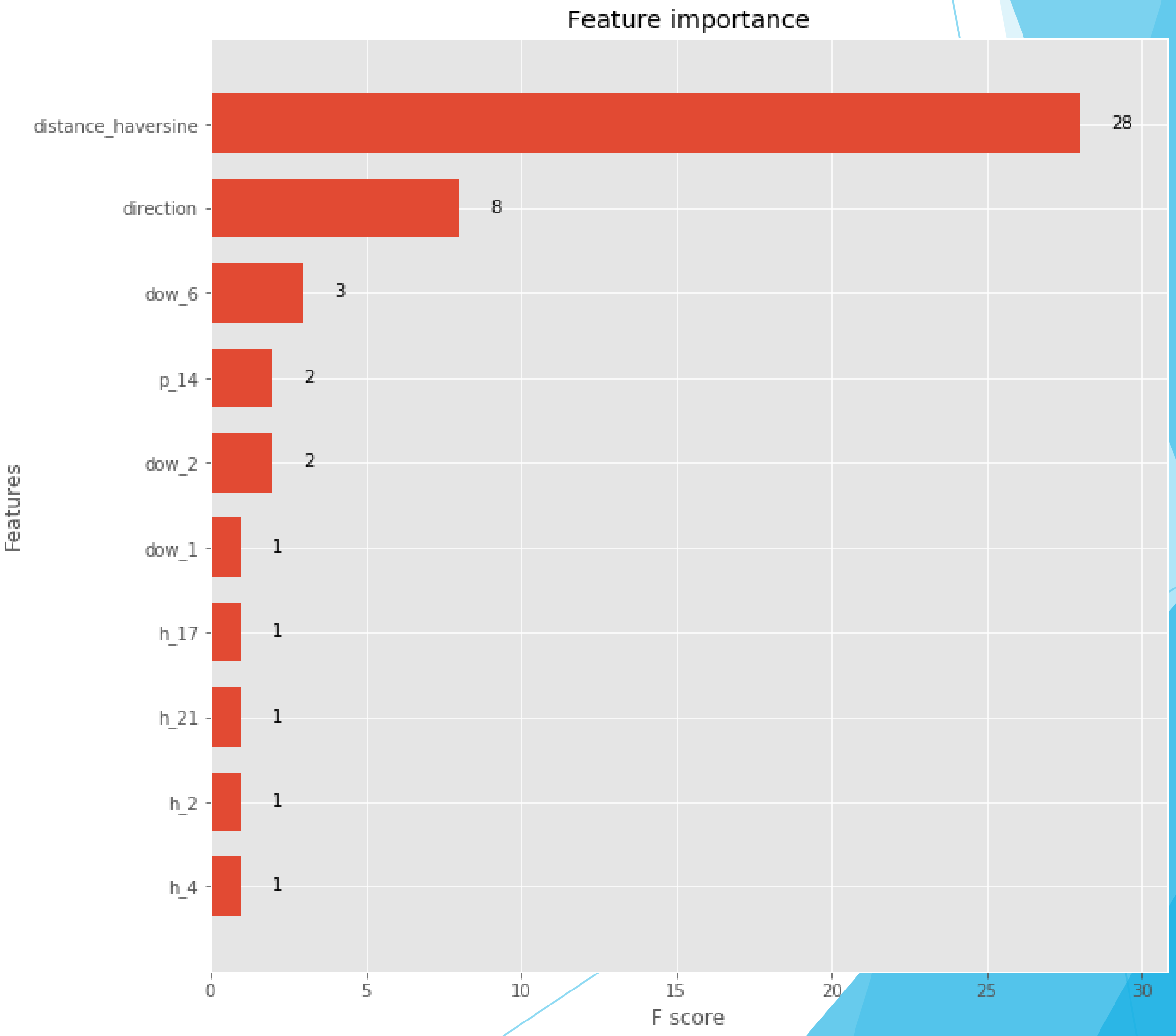


```
xgb_pars = {'min_child_weight': 1, 'eta': 0.3, 'colsample_bytree': 0.9,
            'max_depth': 7, 'subsample': 0.9, 'lambda': 1., 'nthread': -1,
            'booster' : 'gbtree', 'silent': 1, 'eval_metric': 'rmse', 'objective': 'reg:linear'}
model = xgb.train(xgb_pars, dtrain, 20, watchlist, early_stopping_rounds=2, maximize=False, verbose_eval=1)
print('Modeling RMSLE %.5f' % model.best_score)
```

```
[0]	train-rmse:4.21124	valid-rmse:4.21162
Multiple eval metrics have been passed: 'valid-rmse' will be used for early stopping.

Will train until valid-rmse hasn't improved in 2 rounds.
[1]	train-rmse:2.96461	valid-rmse:2.96482
[2]	train-rmse:2.09832	valid-rmse:2.09849
[3]	train-rmse:1.50041	valid-rmse:1.50043
[4]	train-rmse:1.09363	valid-rmse:1.09362
[5]	train-rmse:0.822409	valid-rmse:0.82236
[6]	train-rmse:0.648375	valid-rmse:0.648299
[7]	train-rmse:0.541825	valid-rmse:0.541714
[8]	train-rmse:0.478316	valid-rmse:0.478233
[9]	train-rmse:0.442395	valid-rmse:0.44239
[10]	train-rmse:0.422669	valid-rmse:0.422674
[11]	train-rmse:0.411192	valid-rmse:0.411129
[12]	train-rmse:0.402609	valid-rmse:0.402614
[13]	train-rmse:0.397606	valid-rmse:0.397821
[14]	train-rmse:0.39415	valid-rmse:0.394486
[15]	train-rmse:0.39092	valid-rmse:0.391286
[16]	train-rmse:0.388608	valid-rmse:0.389039
[17]	train-rmse:0.386758	valid-rmse:0.387339
[18]	train-rmse:0.385137	valid-rmse:0.385821
[19]	train-rmse:0.383545	valid-rmse:0.384294
Modeling RMSLE 0.38429
```

XGBoost: Gradient Boosted Trees

- The feature importance plot shows which features have the greatest effect on trip duration by plotting the number of times each feature is split on across all boosting rounds in the model, illustrated as a horizontal bar plot.
- Since there were 88 features, it would be difficult to see all of the feature importance, so the above plot shows the top 10 features (arbitrarily determined number).
- Furthermore, we see that direction, distance, day of week variables, and pickup cluster variables are the biggest factors in determining trip duration for taxis in NYC.



Time Series Forecasting
ARIMA : AR, I , MA

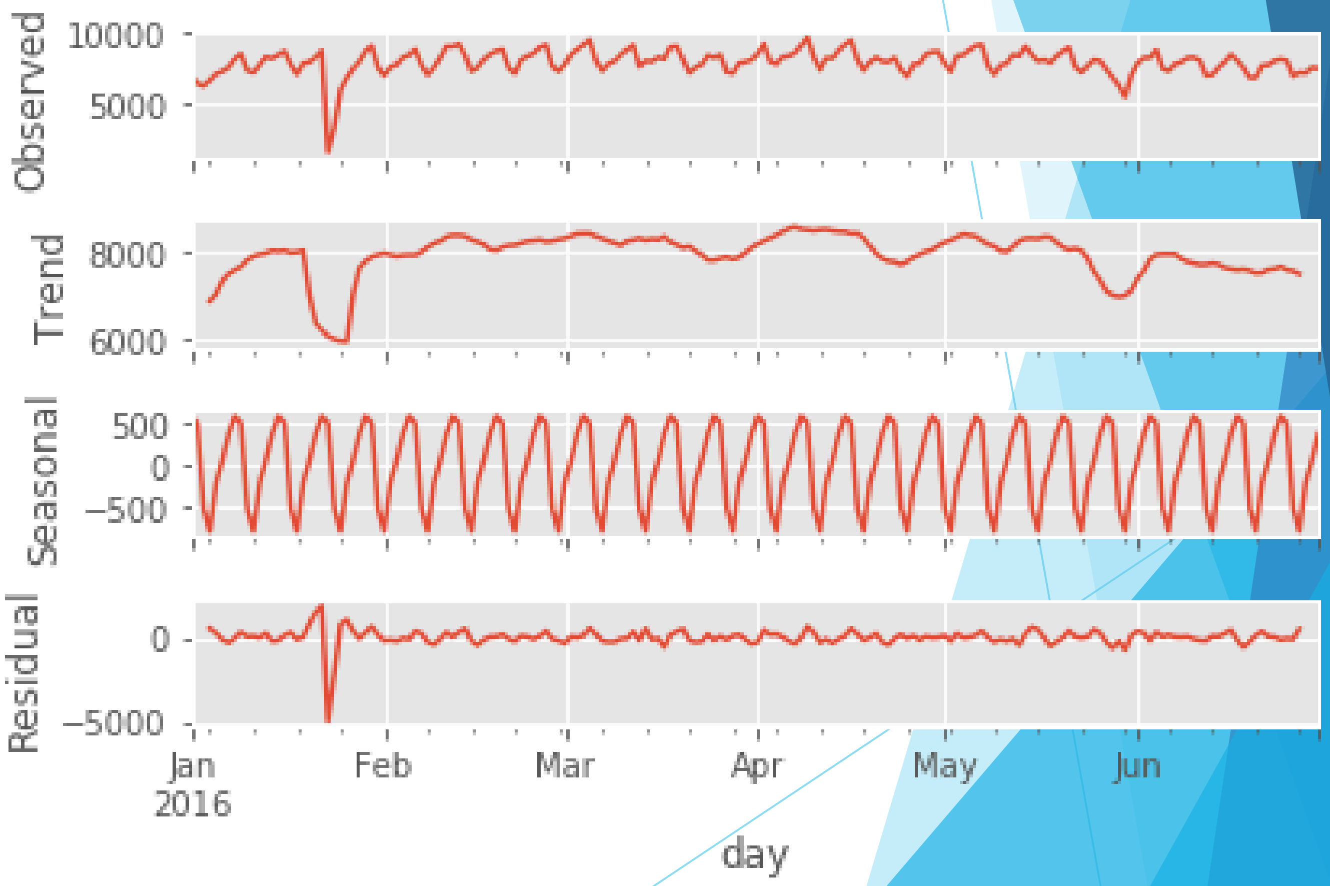
AR or Autoregression (p): A regression model that uses the dependent relationship between the current observation and observations over a previous period

I or Integrated (d): Differencing of observations (subtracting an observation from an observation at the previous time step) to make the time series stationary

MA or Moving Average (q): Using past errors to predict the current value. To put it simply if the series is stationary then it will have a constant mean, however at any point in time it may not actually be at the mean. The distance from the mean is called the error

Time Series Analysis

- Other than the anomalous dip in the number of rides on January 23rd (which we know from EDA to be caused by inclement weather conditions) we can see a strong weekly seasonal component existing in the time series.
- We can use the statsmodel.tsa.seasonal modules seasonal_decompose to look at the error-trend-seasonality graph to further validate this claim.

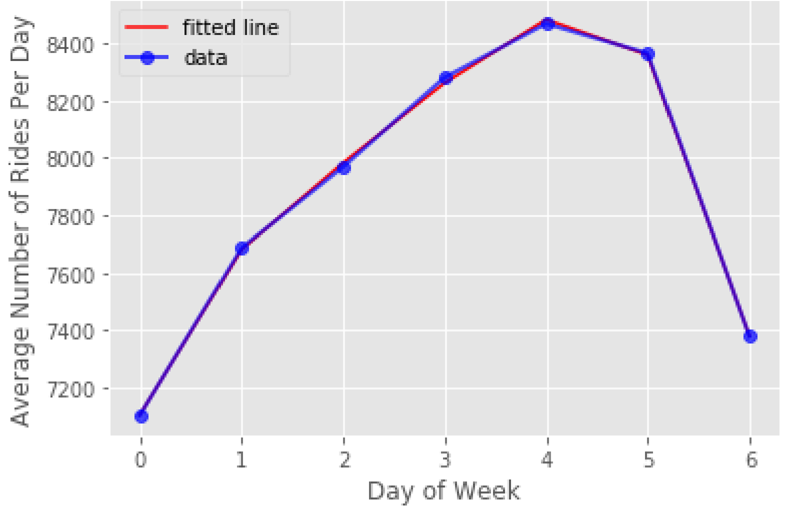


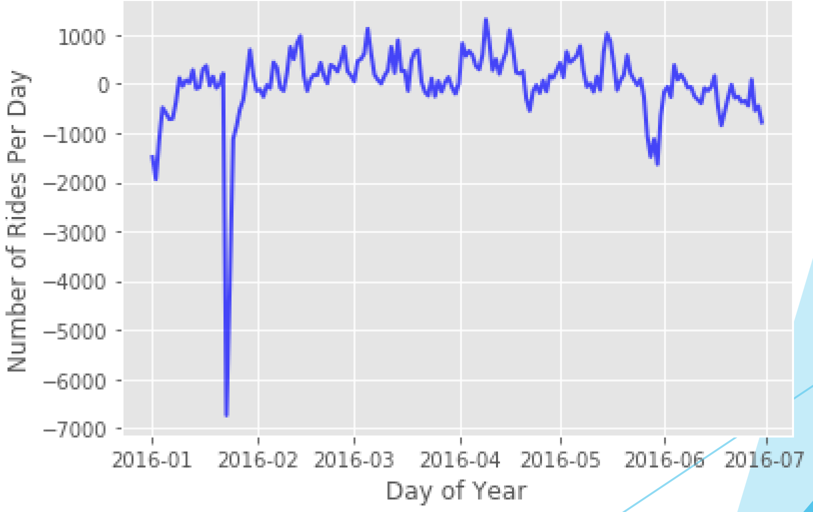
Time Series Analysis

- Since the weekly seasonality of our time series has been identified, it can be modeled.
- A time series where the seasonal component has been removed is called seasonal stationary.
- One of the assumptions of doing ARIMA modelling is that the time series is stationary - that it has constant mean, variance, and autocorrelation over time.
- We can model the seasonal component directly as a sine wave over a generally fixed period and amplitude. We can choose a single week/month of data, or all of the data. We could also smooth the observations using a moving average centered on each value. When using np.polyfit(), a consistent sine wave can usually be modeled by using order 4 or 5.

Seasonal Adjustment

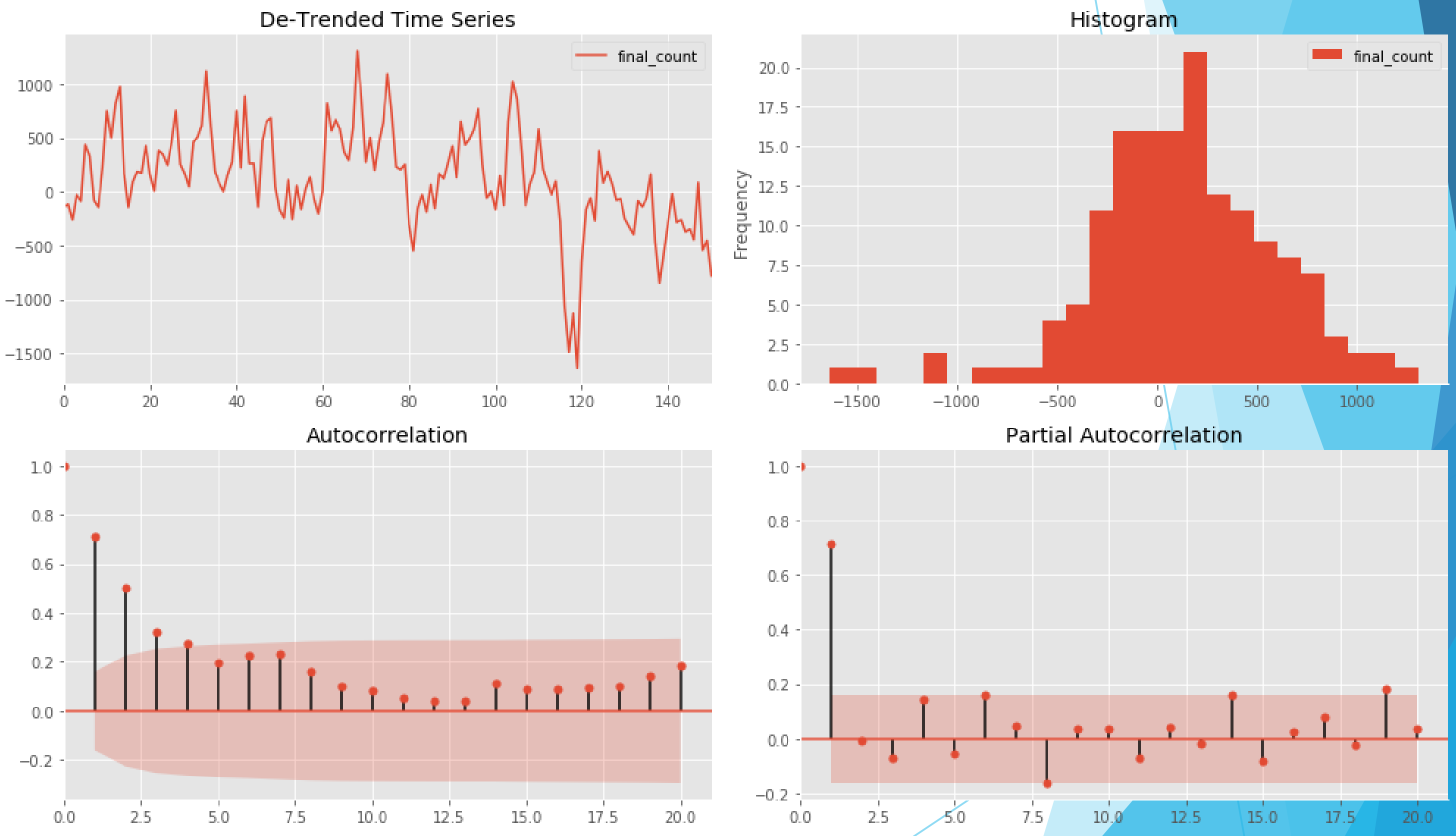
- We can take these fitted counts of the number of rides predicted by the polynomial fit, and create a dataframe that consists of the days of the week and the polynomial fit predicted values for the number of rides. We can then merge it into the original dataframe, obtaining what is a weekly trend subtracted, seasonal stationary time series


Weekly-Trend Polynomial-RegressionFitted Number of Rides


Weekly-Trend Polynomial-Regression subtracted, De-Seasonalized Number of Rides

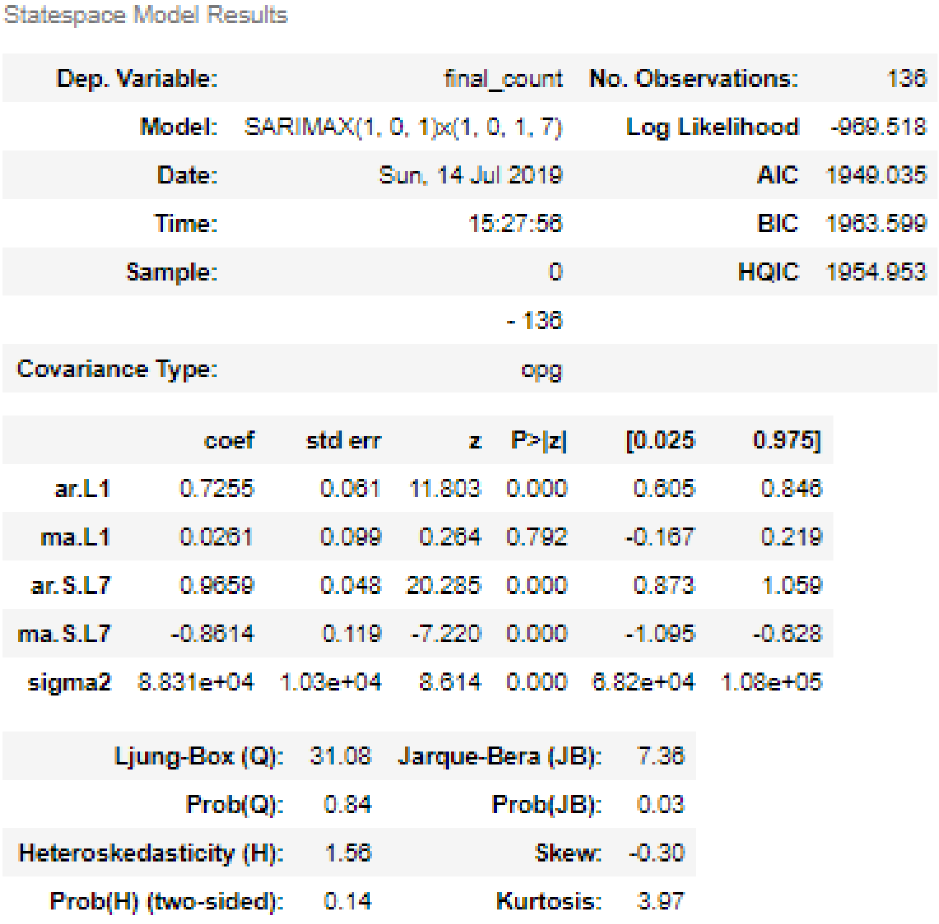
ARIMA Model Diagnostics

- The ACF and PACF are good diagnostic tools to obtain some more general information about the de-trended time series as well.
- You can visually check to see if your time series is displaying stationarity.



Fitting a SARIMAX Model

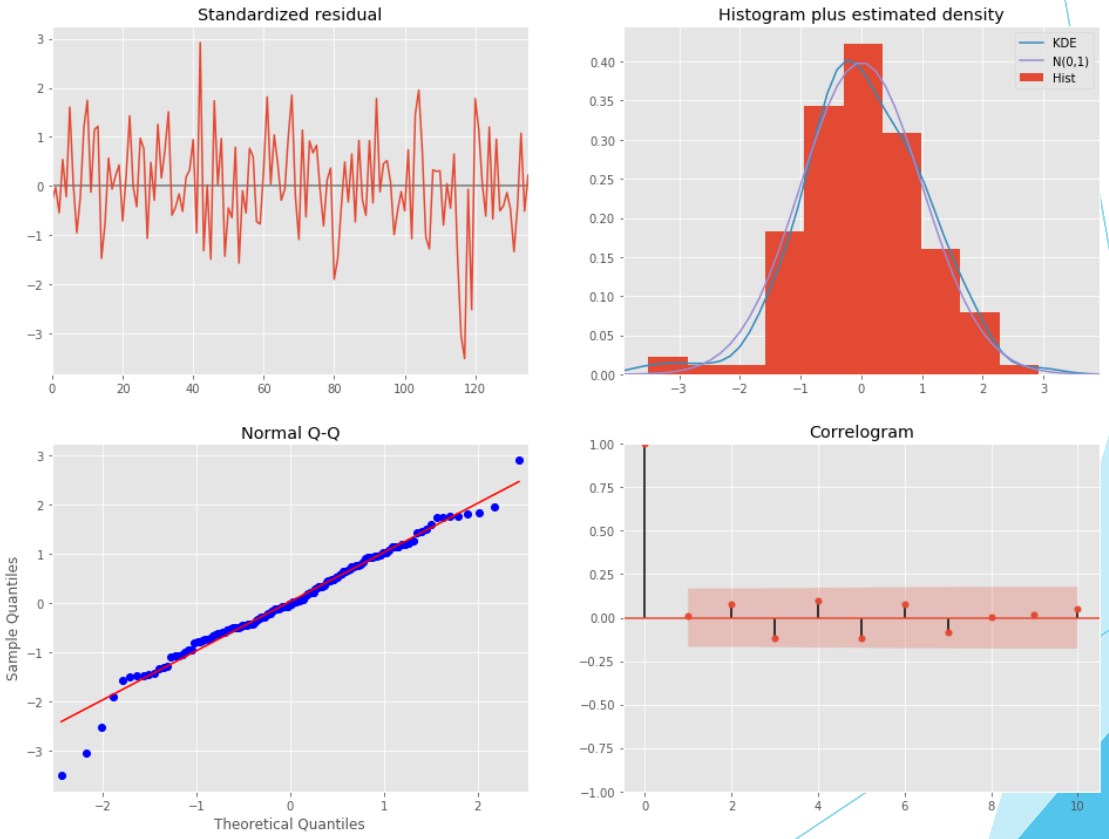
- We can use the statsmodel's SARIMAX function to model a seasonal ARIMA model on our dataset by prescribing a set of p,d,q,s variables.
- Calling the summary method on the fitted model then gives information about the fitted model including information criteria such as AIC and BIC.



```
Statespace Model Results

   Dep. Variable:                  final_count  No. Observations:       136
          Model:  SARIMAX(1, 0, 1)x(1, 0, 1, 7)    Log Likelihood  -969.518
           Date:                Sun, 14 Jul 2019               AIC  1949.035
           Time:                        15:27:56               BIC  1963.599
         Sample:                               0              HQIC  1954.953
                                           - 136
Covariance Type:                             opg

              coef    std err        z   P>|z|    [0.025     0.975]
   ar.L1    0.7255      0.061   11.803   0.000     0.605      0.846
   ma.L1    0.0261      0.099    0.264   0.792    -0.167      0.219
 ar.S.L7    0.9659      0.048   20.285   0.000     0.873      1.059
 ma.S.L7   -0.8614      0.119   -7.220   0.000    -1.095     -0.628
  sigma2 8.831e+04   1.03e+04    8.614   0.000  6.82e+04   1.08e+05

         Ljung-Box (Q):  31.08  Jarque-Bera (JB):   7.36
               Prob(Q):   0.84          Prob(JB):   0.03
Heteroskedasticity (H):   1.56              Skew:  -0.30
  Prob(H) (two-sided):    0.14          Kurtosis:   3.97
```

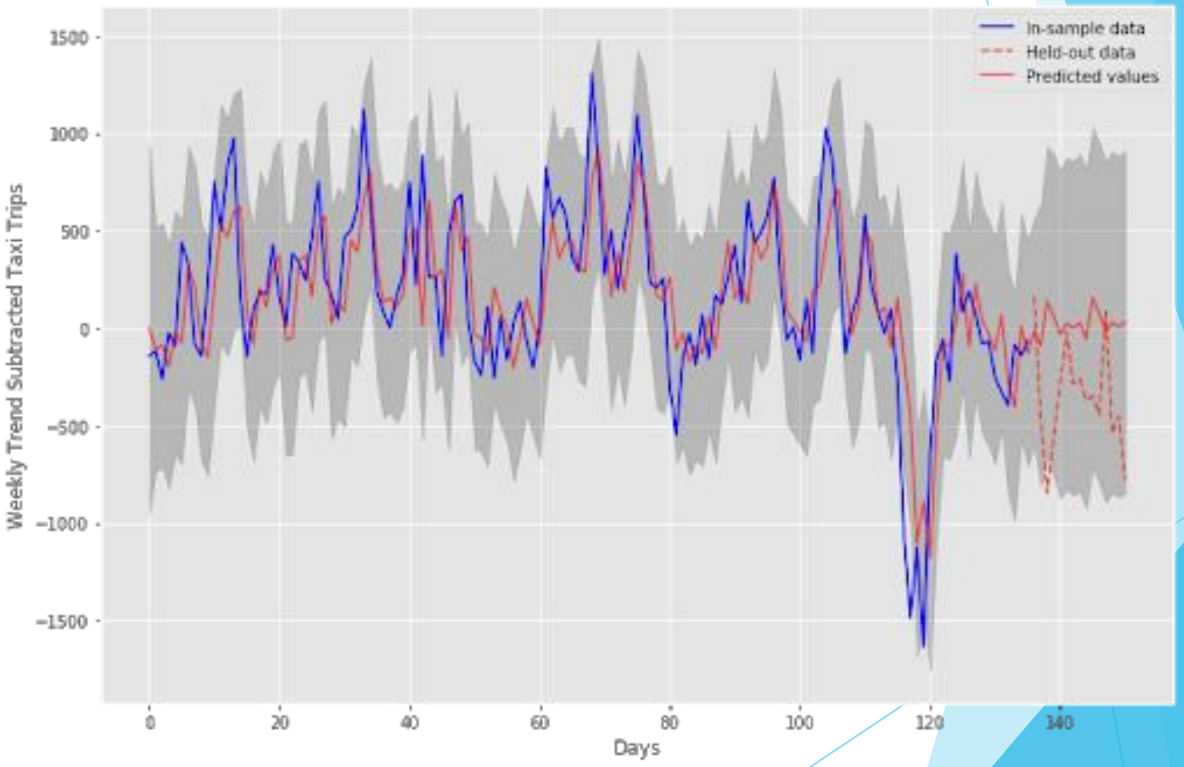
SARIMAX results

- Using the plot_diagnostic method on the model defined with AR 1 and MA 1 order components, we can see features such as residuals, histogram, normality, and correlogram.
- The plot distributions of the residuals from the SARIMAX model shows us that the residuals look scattered around zero, and are roughly normally distributed when looking at the histogram and the linear fit



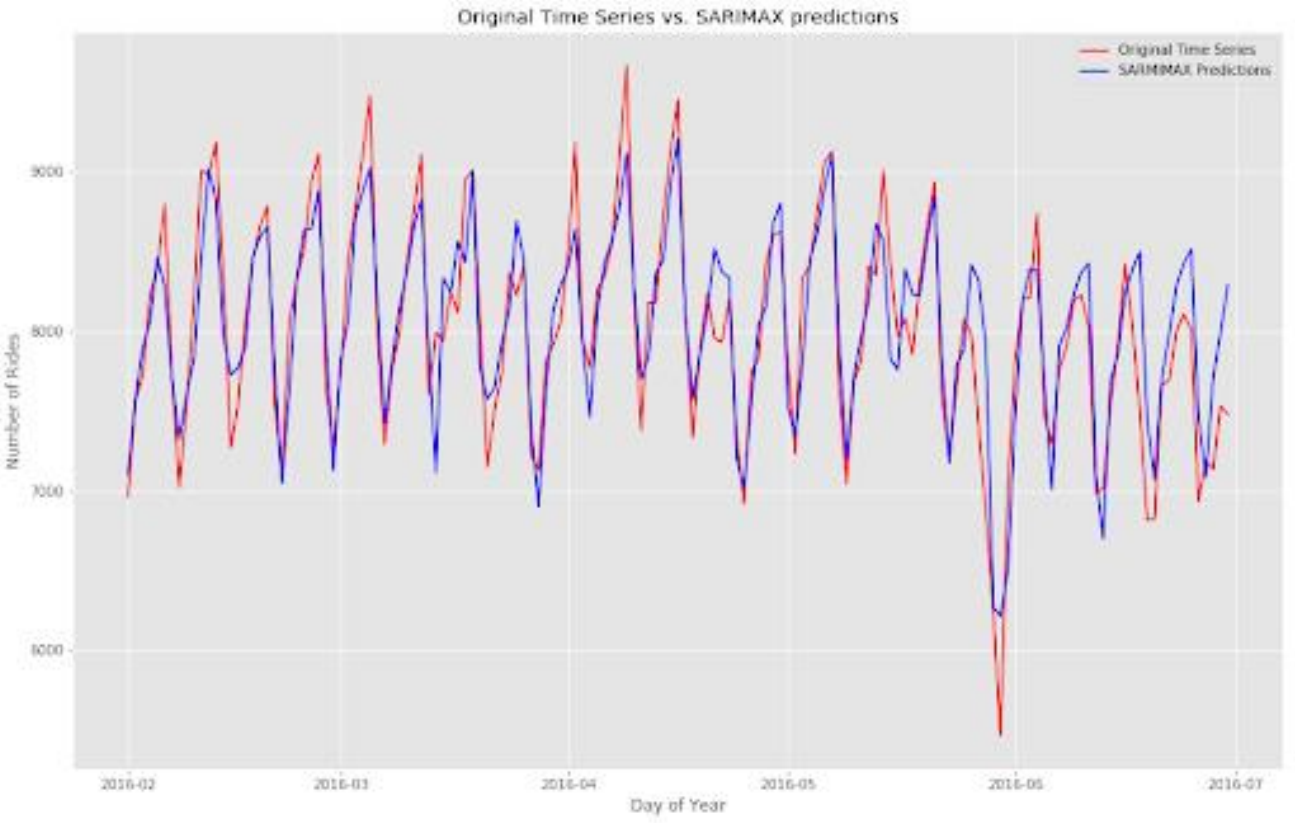
SARIMAX results

- With our SARIMAX model, we achieve an in sample RMSE error value of 300, while we achieve an out of sample forecast RMSE of 490.
- While our forecast model was only 15 data samples, we get a promising error metric, when considering that there are more than 7 thousand trips on average given any day in NYC.



SARIMAX results

- Now, by merging back into the original dataframe, we can reapply the weekly trends, and get the final transformed predictions on top of the original time series data. The re-transformed SARIMAX predictions show how the RMSE error of around 300 cars per day was not a bad measure at all.
- Our SARMIX model was able to accurately predict the model and its weekly seasonality.

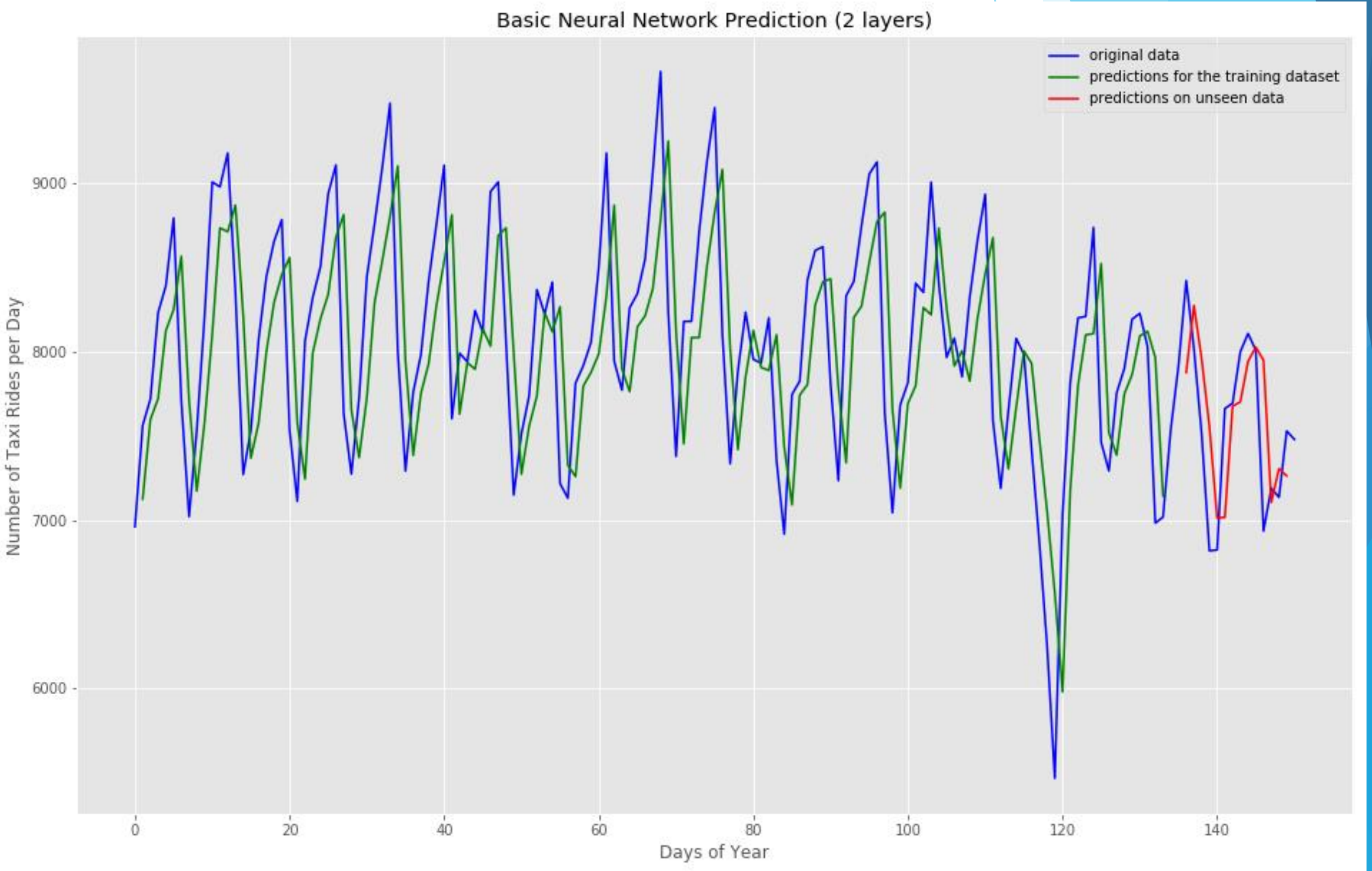


Neural Network Predictions

- In order to use neural networks as a regression predictor for time series datasets, we first had to preprocess the data such that there were two columns, first with the data, and second with the shifted “lag” data.
- The neural networks would be using the past values as a predictor for the future values.
- Furthermore, since the activation functions are usually sigmoid or hyperbolic tangent functions, the scale of the input data was also handed with MinMaxScaler to be in the range between 0 to 1.

Neural Network Predictions
Basic 2 Layer Neural Network

- Without doing any further detrending or removing any sort of seasonality, we will use the neural networks to see how they perform in forecasting the time series.
- The following neural network had 2 layers with 50 neurons each, with the activation function of ReLu.
- Out of sample test RMSE = 448
- Our Best Model!



Neural Network Predictions
Long Short Term Memory Network

- Long Short Term Memory networks are a type of recurrent neural network that is trained through back propagation and overcomes the vanishing gradient problem. The recurrent neural network makes use of memory cells and the output of the previous time period values to make predictions of the future.
- Instead of neurons LSTM networks have memory blocks that are connected through layers. A block contains gates that manage the block's state and output.

Neural Network Predictions
Long Short Term Memory Network

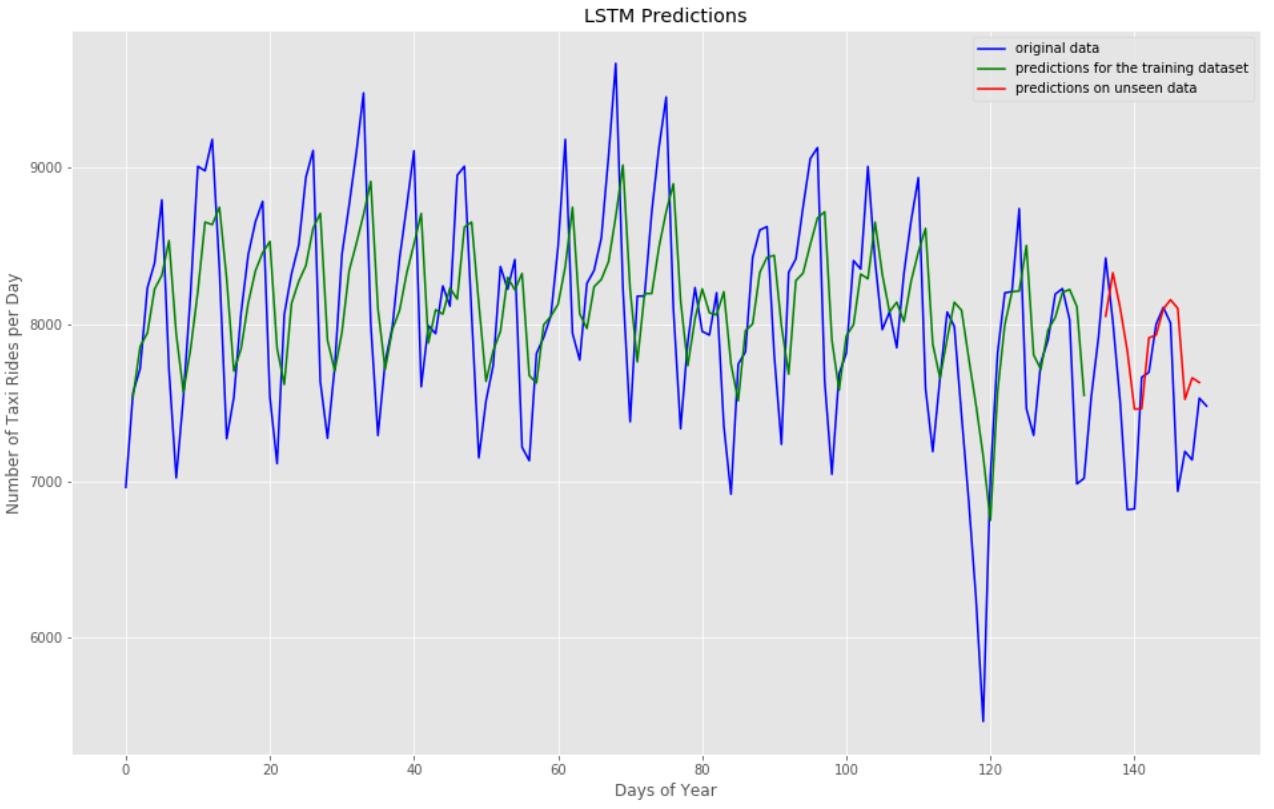
- There are three types of gates in each unit:

1. Forget Gate: decides what information to throw away from the block
2. Input Gate: decides which values from the input to update the memory state
3. Output Gate: decides what to output based on input and the memory of the block

Each unit is like a mini state machine where the gates of the units have weights that are learned during the training procedure. Phrasing the data frame as a regression problem, we can phrase the question as given the number of rides these past couple months, what will be the number of taxi rides in the next couple of days?

Neural Network Predictions
Long Short Term Memory Network

- The network has a visible layer with 1 input, a hidden layer with 50 LSTM blocks or neurons, and an output layer that makes a single value prediction.
- The default sigmoid activation function is used for the LSTM blocks.
- The network is trained for 40 epochs and a batch size of 1 is used.
- out of sample (test) RMSE = 527



Results + Insights + Tips

- Distance, direction, and day of the week matters most when predicting taxi trip duration.
- Basic 2 layer neural network has the best RMSE value when forecasting future taxi demand.
- Detrending based on observed trends is important in achieving stationarity before modeling the time series with traditional ARIMA models.
- Most Taxi trips are taken between the same clusters : Taxi trips are usually taken for shorter distances
- Longest trips are taken in the morning, and they are usually to the airports
- Vendor 2 seems to consistently take longer trips, although this is anonymized, try to avoid Vendor 2 if you’re in a hurry!