Springboard Intermediate Data Science: Python

Analysis of Air Quality and Weather in the Baltimore Area

Faisal Mahmood

March 2019

I. Introduction

Air quality affects the livelihood of millions of people, especially those with respiratory issues, and at times it can affect the ability for people to spend time outdoors. In this project, I will attempt to build a model to predict ozone concentrations in the air, based on hourly weather observations. Ozone is one of six pollutants whose atmospheric concentration is regularly tracked for determining air quality. Although ozone in the stratosphere plays a crucial role in protecting humans from the harmful effects of ultraviolet radiation from the sun, ozone near the ground is harmful for public health, especially among individuals with asthma or other respiratory issues. According to the EPA, an ozone concentration of 0.075 parts per million (ppm) averaged over a period of 8 hours can affect sensitive groups[1].

For this project, the prospective stakeholders in this analysis would be anyone who has a vested interest in air quality as it relates to ozone concentration and its relationship to temperature and other weather observations. Such stakeholders may include health and wellness firms, companies that rely on summer tourism, and others who rely on a healthy air quality. If an effective model from this analysis can be used to reliably predict the ozone concentration, then the clients might benefit financially and potentially help people who suffer from the health problems associated with poor air quality.

II. Approach

A. Data Acquisition and Wrangling

Two sets of data were obtained for this project, each of them with data ranging from January 1, 2006 to December 31, 2017. One set, obtained from the Environmental Protection Agency (EPA), consisted of air quality data that had hourly ozone concentrations for a measuring station in Essex, MD (AQS site 840240053001)[2], measured in parts per million (ppm). The other set, obtained from the National Oceanic and Atmospheric Administration (NOAA)[3], consisted of hourly weather observations for Martin State Airport in Maryland (WBAN 93744), which is located 3 miles from Essex.

The objective of this data wrangling process was to ensure that for both datasets, observations for every hour would be available for the duration of the time period being analyzed. With those observations available, they could be merged into one dataframe containing both the measured ozone levels and the weather observations for each hour, over the same time window.

---

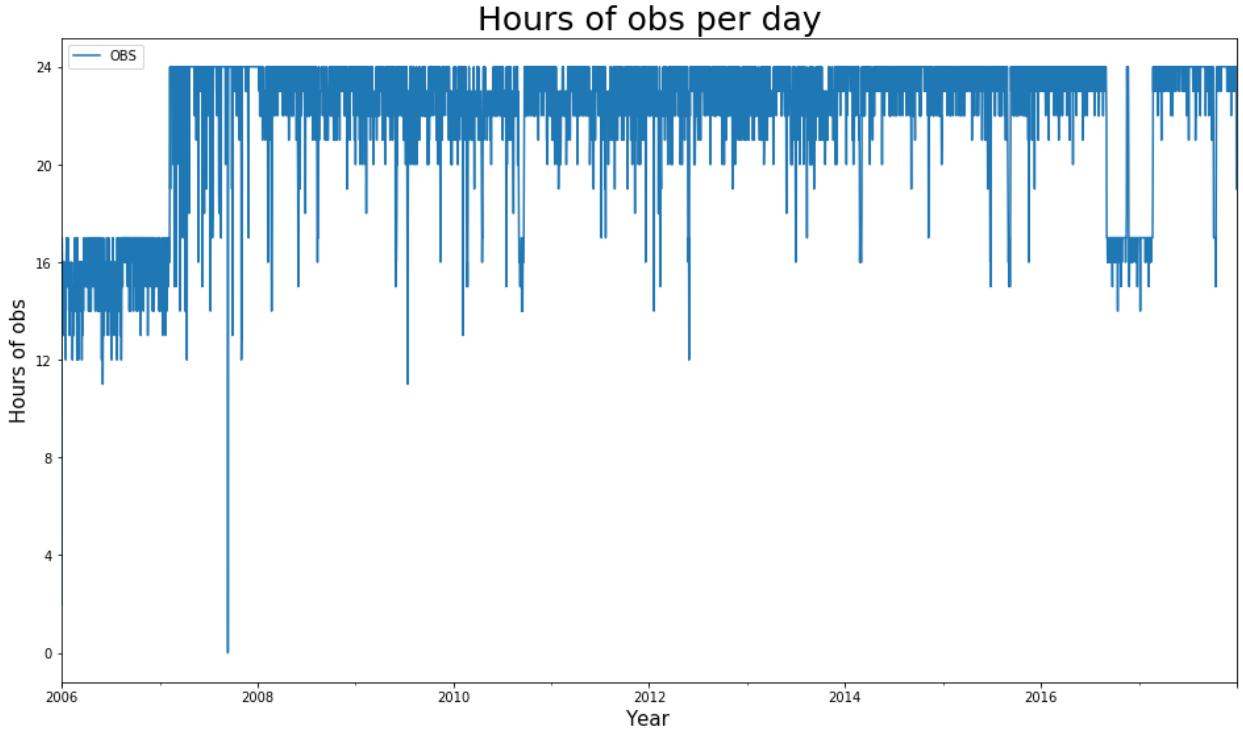
[1] "Air Quality Index - A Guide to Air Quality and Your Health." *AirNow*, Environmental Protection Agency, 27 July 2017, airnow.gov/index.cfm?action=aqi_brochure.index.

[2] "Air Data: Air Quality Data Collected at Outdoor Monitors Across the US." *EPA*, Environmental Protection Agency, 29 Oct. 2018, www.epa.gov/outdoor-air-quality-data.

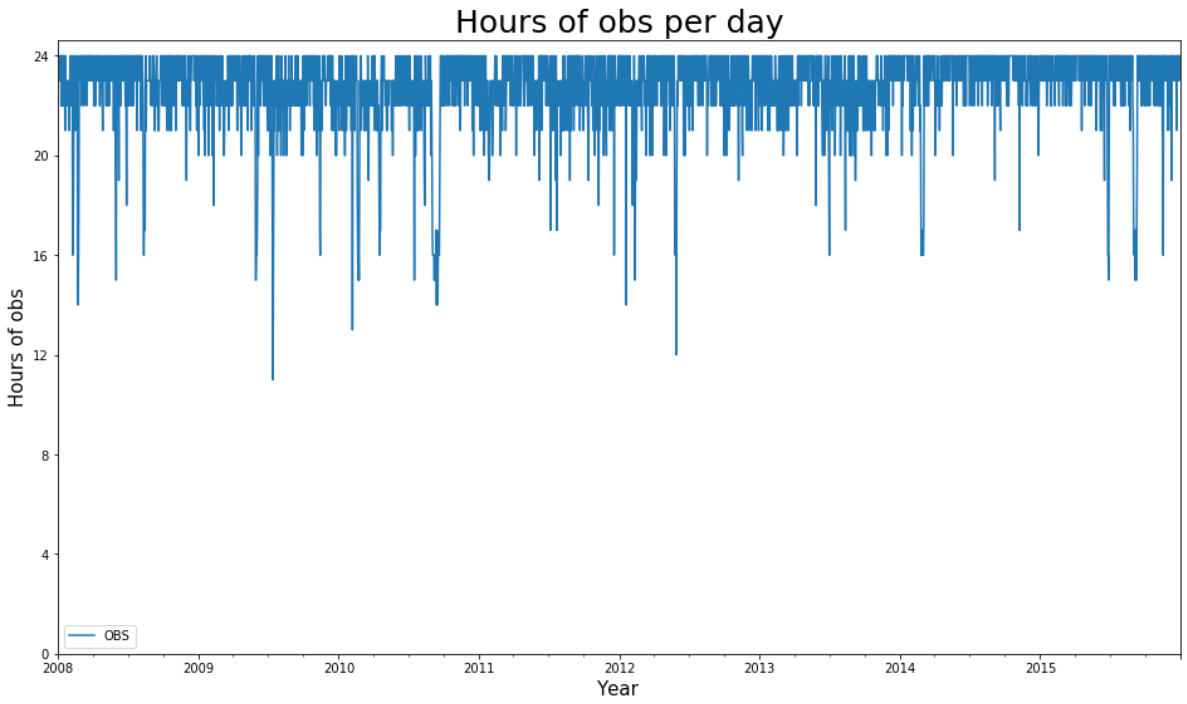
[3] *National Centers for Environmental Information*, National Oceanic and Atmospheric Administration, www.ncdc.noaa.gov/.

The weather dataframe was cleaned up first, and that one required more work. The dataframe was filtered so that only the datetime (hour and date, labeled **YR--MODAHRMN**, then renamed to **TIME**), wind speed (**SPD**, measured in miles per hour), visibility (**VSB**, measured in miles), temperature (**TEMP**, measured in degrees Fahrenheit), dew point (**DEWP**, also measured in degrees Fahrenheit), and surface air pressure (**STP**, measured in millibars).

Other than the **TIME** column, the columns appeared to show no clear datatype. All of the variables are numeric in nature, so they were converted using the pd.to_numeric() function, with the errors coerced into NaN values. Also, the **TIME** column was set as the DatetimeIndex for this dataframe. The null values were identified by converting all columns to float data types, and then the missing values were counted. In the datetime column, the times were initially given as Greenwich time, so they were converted to Eastern US time. Next, the observations were resampled into a new dataframe so they could be cleanly shown for every hour, and then they were counted hourly to see whether each hour had an observation. Each hour was shown, and there was a column in which the number of observations per hour were counted. Many hours had zero observations, and those zero values were converted to null values so that the number of hours with observations for each day could be counted. A plot was done, and it showed that for the earlier and later parts of the dataset, significant numbers of observations were consistently missing.



Therefore, the dataset was truncated to only include 2008 to 2015, inclusive. This time period had occasional missing values (as would most datasets), but no consistent period of missing hours was found. As a result, the main weather observation dataframe was similarly truncated.



Then, an interpolation was done. To interpolate this data, a new dataframe ‘wx2’ consisting of every hour from 2008 to 2015 was created, under the singular column **TIME**. The index of the initial ‘wx’ dataframe was reset, so that the datetimes could be used as a column. A for loop was used to replicate all the column names of the wx dataframe, but without any data. However, the data types of those columns were set to float, so that any data that gets appended would recognized as numeric.

Afterwords, a new column 'stamped' in wx2 was created, with the value being set to 1, so that every row of this original wx2 dataframe would be recognized once the data from the original wx dataframe is appended. Another dataframe, wx_int, was then created to include all the rows and data from both the original wx and the newly created wx2. The 'stamped' column indicated which dataframe the row originated from, with values of 1 associated with the wx2 dataframe, and values of 0 associated with the original wxdataframe. The rows were then sorted by time.

With all the rows of the wx and wx2 dataframes sorted in chronological order, with the rows from wx2 having null values but clean hourly timeframes, the dataframe was finally ready for interpolation.

The interpolation was done in both directions, and all the hourly rows of the wx2 dataframe were successfully filled in with values based on the data from the original wx dataframe. The values from wx were deleted, and then wx_int was made to be the new weather dataframe.

The values in this interpolated dataframe were all floats, and every column needed for its values to either be rounded or expressed as integers. Once that was done, and once every column was confirmed to have no missing values, the data cleaning of the weather dataframe was finally complete.

The ozone dataframe was then cleaned. The ozone dataset was uploaded as a dataframe. Although it was a plain text file, it had commas separating each value, so no delimiter needed to be specified. The date format was odd, so that column needed to be cleaned before setting it as an index. The initialized dataframe contained 20 columns and 102,249 rows. The only columns of importance were the 'datetime' column and the 'value' column, the latter of which contains the ozone concentration measured in parts per million (ppm). Fortunately, there were no missing values for those two columns, with the exception of the final row, which is void of any values.

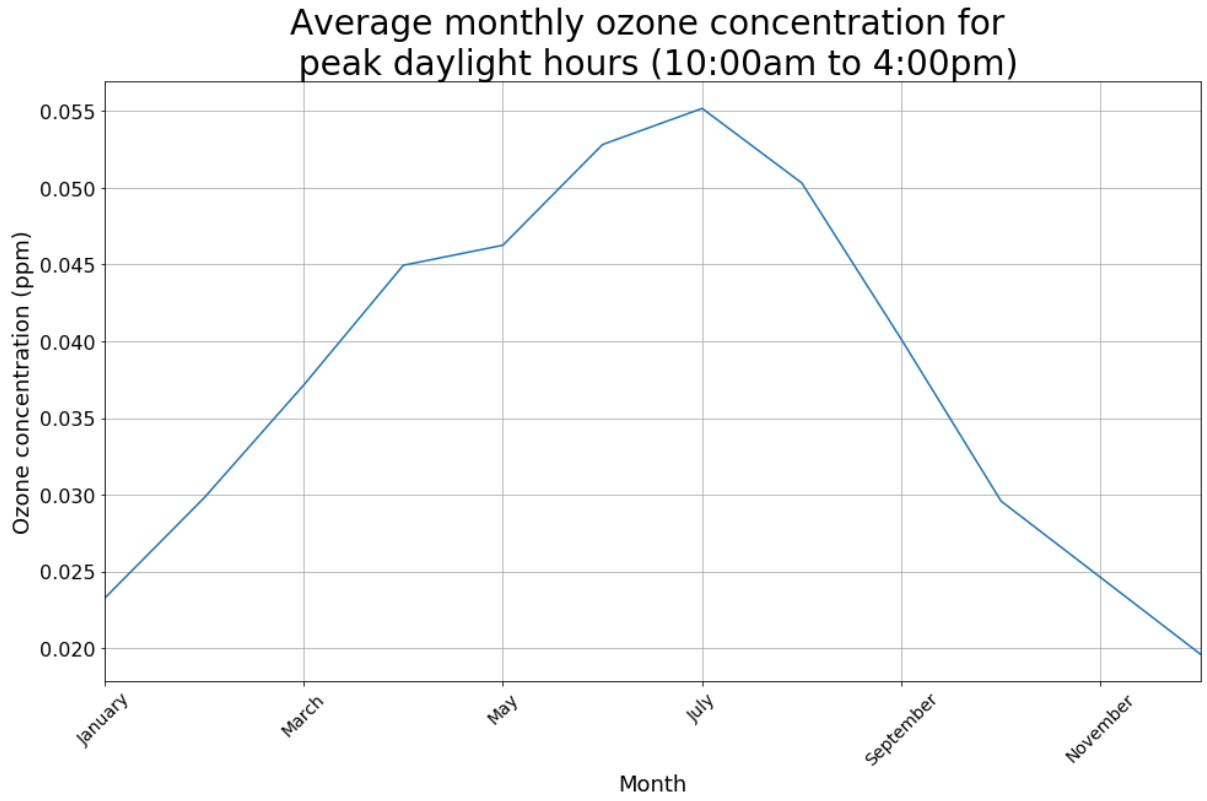
After a brief inspection, the dataframe was filtered so that only the 'datetime' and 'value' columns were selected. The 'datetime' column still needed to be cleaned. The reported times were local, so no time zone needed to be assigned. After the 'datetime' column was cleaned, it was ready to be converted to a datetime object. The datetime values were then arranged in chronological order, and the index was set to that column. The value column was renamed to ozone_ppm, so that the column name could be more descriptive of the data.

After the dataframe was confirmed to have no missing values, the ozone data was merged with the weather data into the dataframe df. After the merge, the new dataframe df was confirmed to have no missing values, and was pickled so it could easily be loaded again in another notebook. The data wrangling was finally complete.

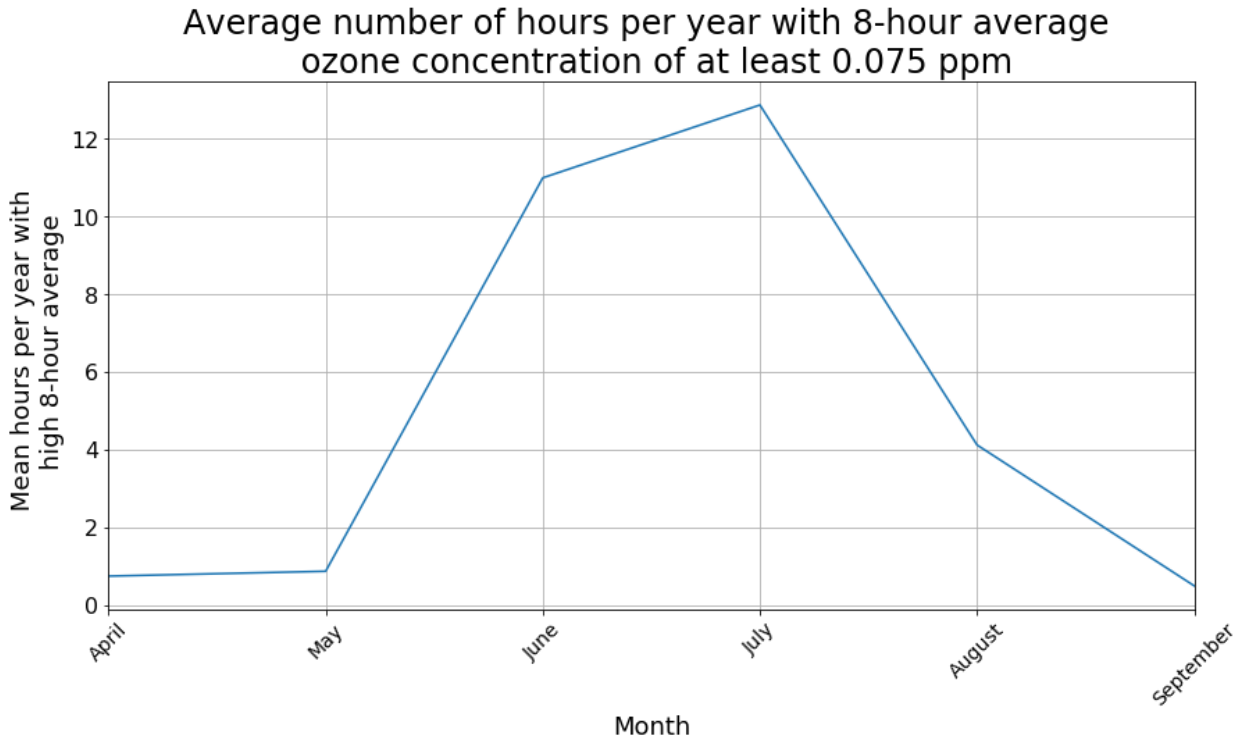
B. Storytelling and Inferential Statistics

The first step was to import all the relevant packages and upload the dataframe from the pickle. Afterwards, there was a great deal of possibilities to explore with the data, to see how it appears and what insights can be gathered from it. Since ozone concentration (ppm) is the target variable, the first step was to simply visualize the entire scope of the data and see how it appears. This dataframe included 68,231 rows, and it consisted of every hour of observation from 2008 to 2015, inclusive.

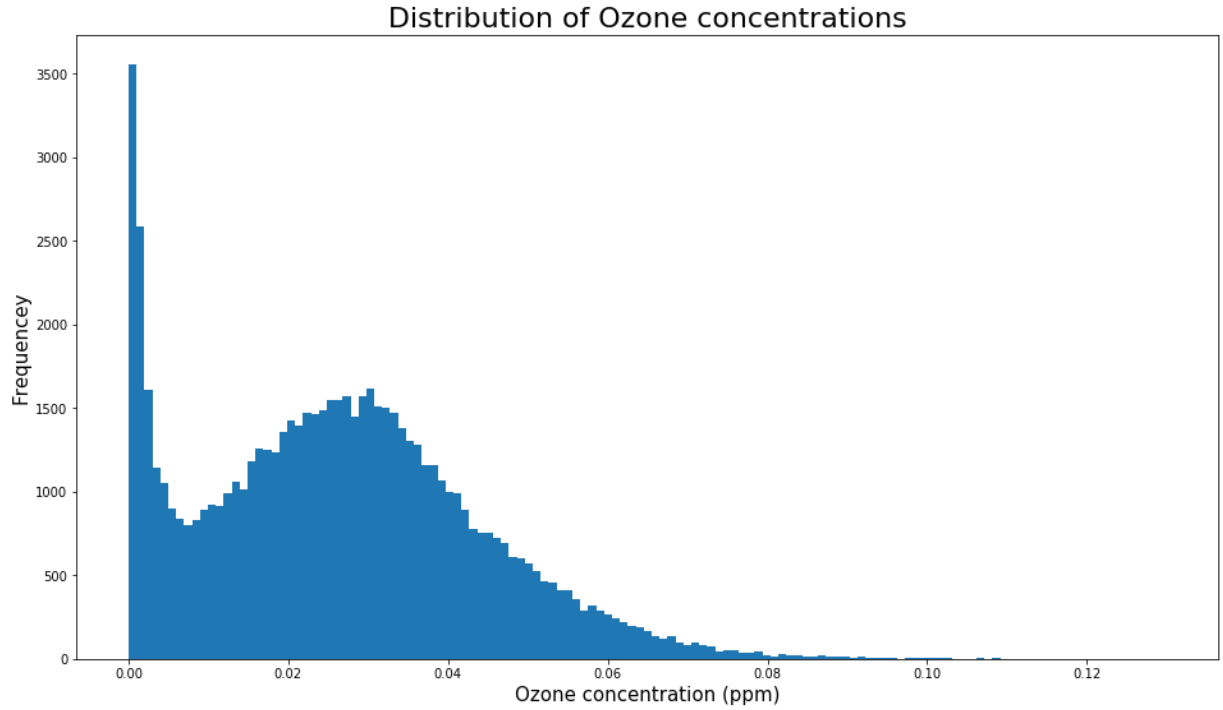
A time series plot showing ozone concentrations throughout the whole time period was displayed, but was later zoomed in to 2011 and then specifically July 2011, for a better view of the cyclical nature of the data. Then, for a cleaner, more simple visualization of the seasonal differences in ozone levels, the monthly mean of ozone levels during peak daylight hours was plotted for every month. The dataframe was filtered to only include data for the hours from 10:00am to 4:00pm, and then the average monthly levels were taken. Average ozone levels during those hours ranged from under 0.02 ppm in December, to over 0.05 ppm in the summer months.



Ozone concentrations of 0.075 ppm, averaged over a period of 8 hours, are high enough to affect sensitive groups, so the monthly frequency of ozone levels reaching that threshold was plotted, as shown below.

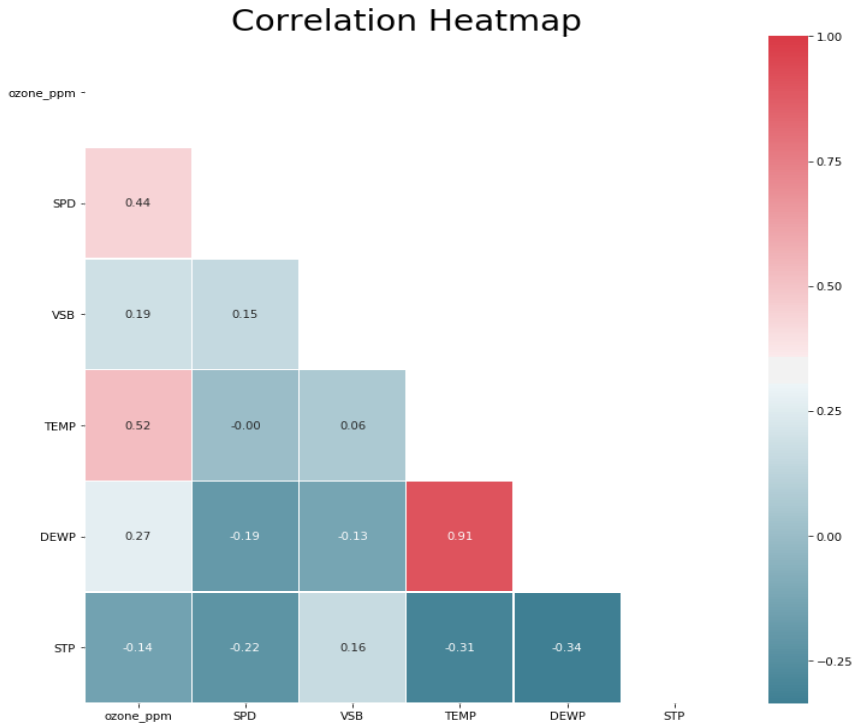


The total frequency of ozone concentrations was then plotted in a histogram to show how the overall distribution appeared. With 131 bins chosen, the plot appeared as a smooth curve, as shown below.



Next, the weather variables were evaluated to see how much they correlate with ozone levels.

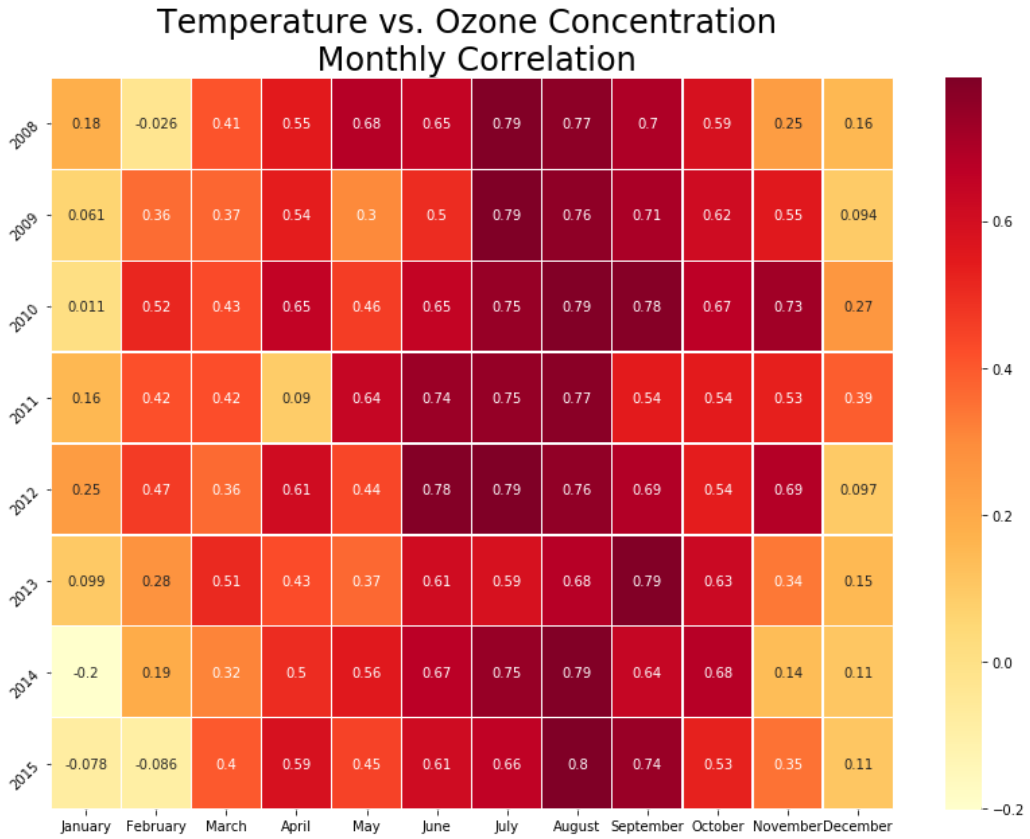
The correlation heatmap shows the pairwise correlations between all of the variables of the dataframe. The dew point is regarded as a proxy for humidity, and not surprisingly, the dew point and temperature were very highly correlated. However, since ozone concentration was the target variable, any correlations of the other variables with ozone levels were be of particular interest. The far left column shows the correlations of each of the feature variables with ozone concentration, with temperature and wind speed having the strongest correlation with the ozone level. The dew point has a weaker relationship with ozone levels, but even that variable was chosen for more exploration.



Since the ozone time series plots showed seasonal variation in the ozone concentration, a breakdown of monthly correlations with ozone was computed for temperature, dew point, and wind speed, which showed the highest overall correlation with ozone levels, with heatmaps being shown for all three feature variables.

As seen below, the temperature vs. ozone heatmap shows a dramatic seasonal difference in the relationship between temperature and ozone level. That relationship is much stronger in the warmer months, especially from June to September, than it is in the winter months, when correlations near zero are common. Since the summer months are the most likely time for heat

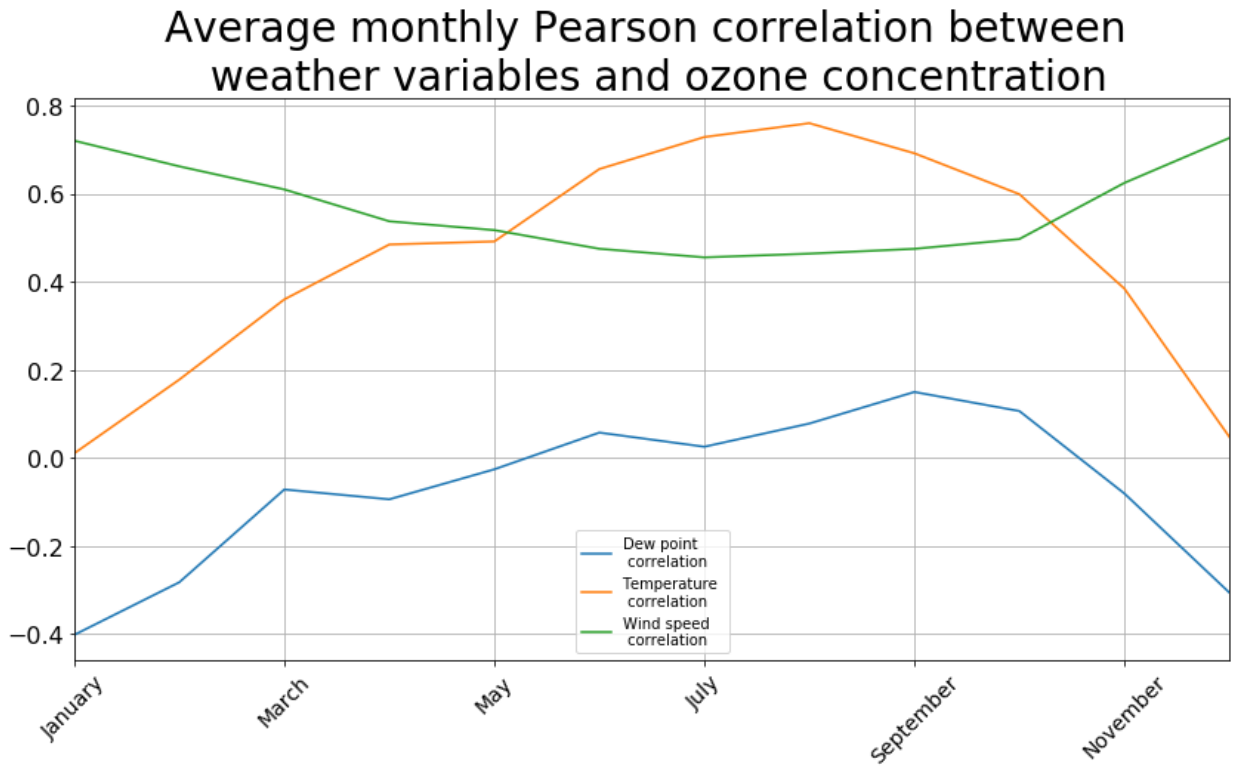
waves that affect air quality, the temperature is a powerful predictor of ozone levels when they are the most relevant.



The next variable whose correlation with ozone was evaluated was dew point. Although temperature and dew point are highly correlated, dew point does not correlate with ozone levels as well as the temperature does. In fact, even in the summer months, average monthly correlations near zero were found to be common. However, in the winter months, dew points had a moderate negative correlation with ozone levels. In other words, higher dew points in the winter are moderately associated with lower ozone concentrations, which is a somewhat surprising finding.

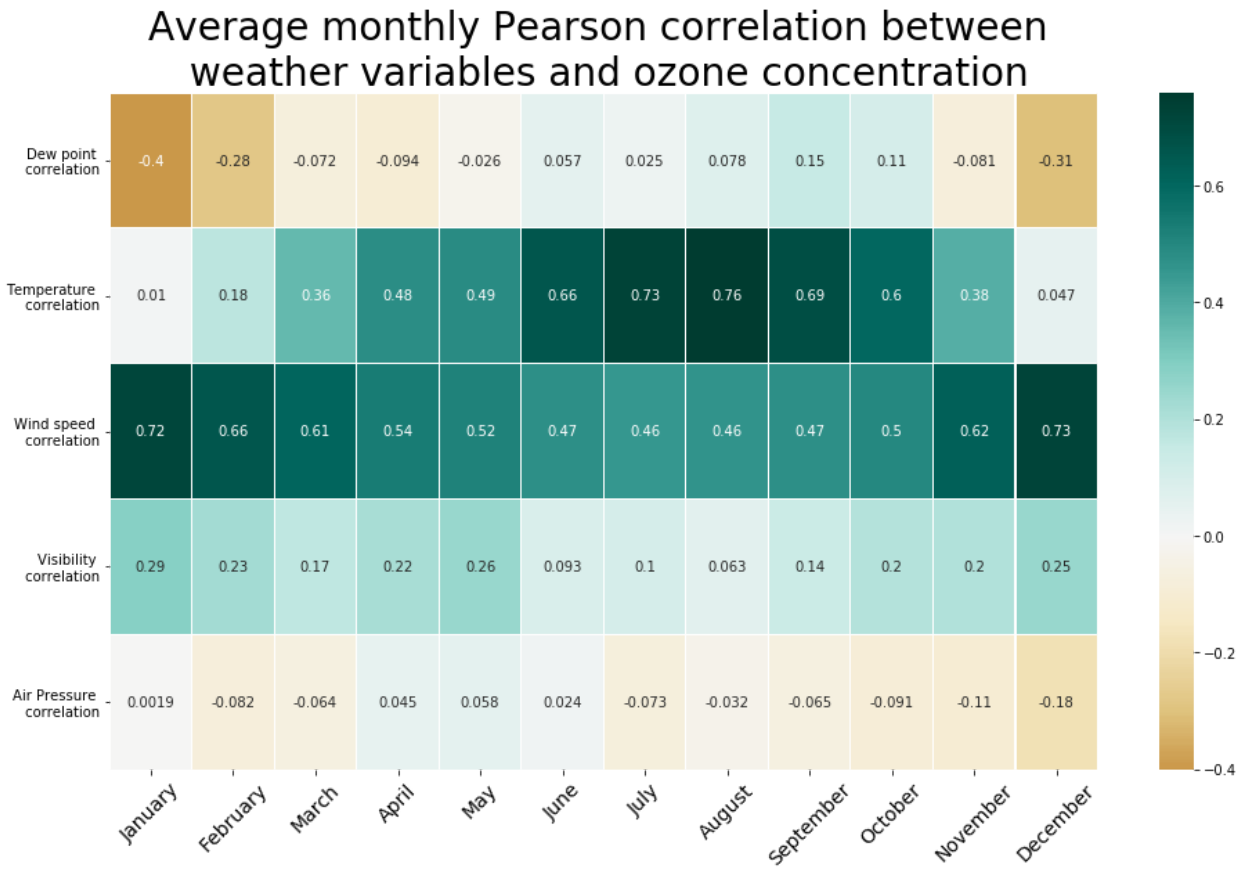
Finally, the wind speed correlation with ozone was evaluated. Wind speed had an overall correlation of 0.44 with ozone level, making it the second most correlated variable after temperature. This was a fascinating yet unexpected finding, exactly the type of discovery that is worthy of being further explored. The monthly breakdown of this data shows that wind speed correlates strongly with ozone levels in the winter months, but has a moderate association with ozone even in the summer months. Although ozone levels in the winter are too low to have a significant impact on human health, the strong association with wind speed is still an interesting and very persistent finding.

After the individual monthly plots, the monthly pearson correlations were plotted (displayed below) for a direct comparison, and the seasonal differences can clearly be seen. Wind speed has the strongest relationship with ozone levels in the winter months, while temperature is the strongest predictor in the summer.



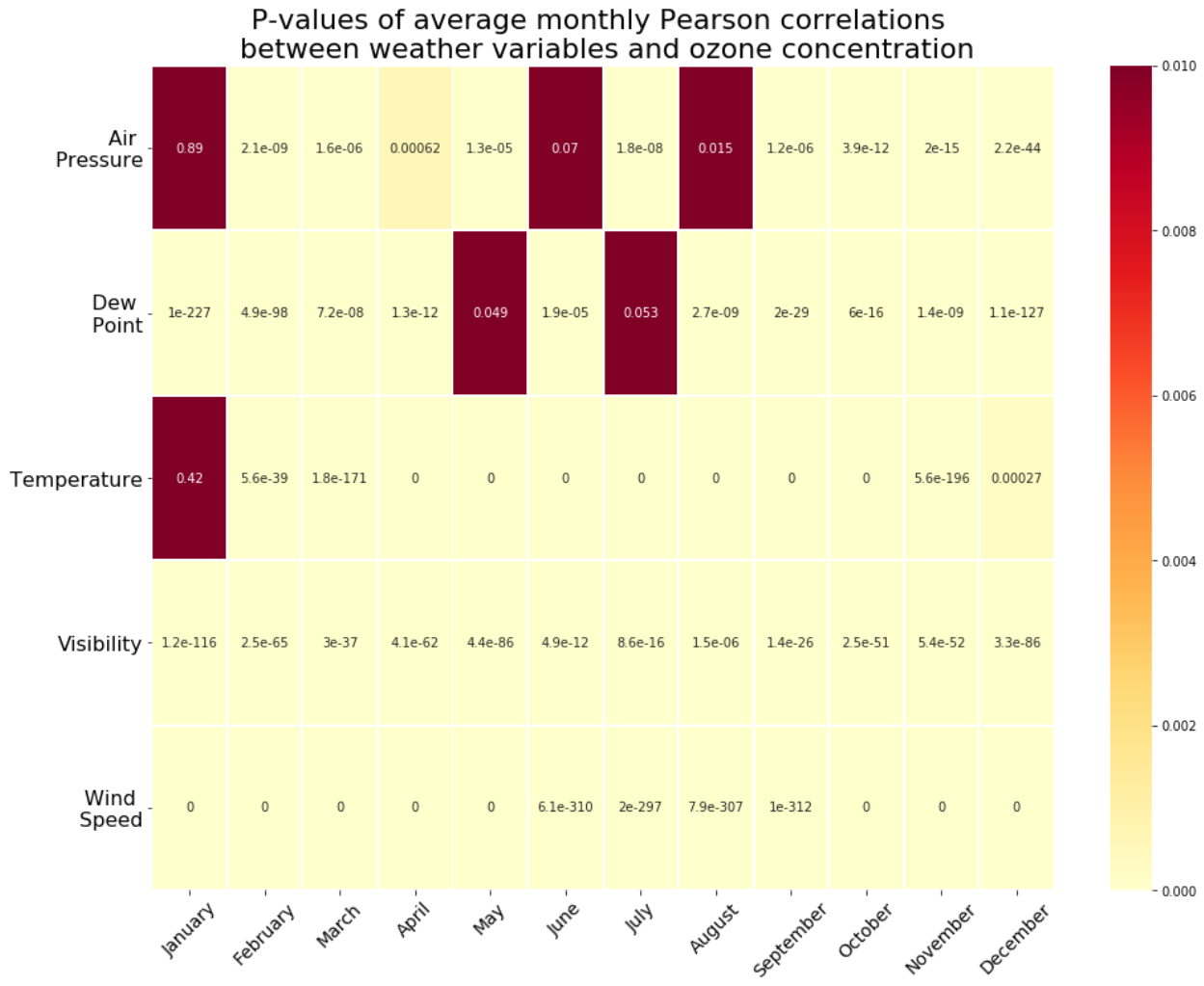
The dataframe with which the above plot was created was then given correlation columns for air pressure and visibility, for the sake of the inferential statistical analysis. Then the dataframe was pickled, so that it could be downloaded in the inferential statistics notebook.

Using the dataframe that was pickled at the end of the storytelling portion, a full monthly Pearson correlation heatmap with all of the variables was created, with temperature and wind speed having the highest correlation with ozone concentration, as was also shown in the storytelling portion. The visibility and air pressure were added to the comparison, after being disregarded in the storytelling portion due to their weak correlations. They both have very little relationship with ozone levels during the summer months, while visibility has a moderate relationship with ozone levels in the winter months. Air pressure is a weak variable throughout the year.



While the weather variables had various levels of correlation with the ozone concentration, they were evaluated on a monthly basis to determine the statistical significance of that correlation. The null hypothesis assumed no relationship, and the value of alpha was be set to 0.01, which means that the null hypothesis could be rejected for p-values that are under 0.01.

The p-values for the Pearson correlations of each of the variables with ozone concentration can be seen in the heatmap below. For all the values under 0.01, the null hypothesis can be rejected, and a statistically significant relationship is likely to exist between the relevant variable and ozone concentration for the given month.



For the vast majority of values, a statistically significant relationship exists with the ozone levels, since the null hypothesis can be rejected. This includes values that are associated with very low correlations, which illustrates the difference between statistical significance and *practical* significance. Only the months shaded in a dark maroon color have a p-value above 0.01 for which the null hypothesis cannot be rejected.

One possible reason for the discrepancy between statistical significance and practical significance is the large size of this dataset. With tens of thousands of values, a statistically significant model can be constructed even if it has very low predictability.

C. Baseline Modeling

To construct the baseline model, the first step was to load the dataframe through the pickled file. Then, the dataframe was split to a training and test sets. The training set consisted of all the data

from January 2008 to November 2015, while the test set consisted of the data for December 2015. The training set was then split into an X_train dataframe that represented values of the weather-related variables in the training set, while the y_train dataframe represented the ozone concentration values in the training set.

Using the LinearRegression() constructor method, a linear regression model was created and then fit to the training set, represented by both the X_train and y_train dataframes. That fitted regression model was then used to compute the predicted y-values for the training set using the X_train data. The data was stored in the dataframe y_pred_train.

With the predicted y-values of the training set, performance scores were calculated, namely: R^2 , the root mean square error (RMSE), and the mean absolute error (MAE). The value of R^2 was around 0.58, which indicated a moderate to strong correlation. The RMSE was around 0.011, and the MAE was near 0.009.

The next step was to evaluate this model on the test set. The test set data from December 2015 was split into X_test and y_test dataframes, similar to how the training set was split. Then the regression model that was trained on the training set was used to predict the target on the test set. A dataset of predicted y-values that corresponded to the X_test data was created, and stored in the dataframe y_pred_test. Then the performance scores were calculated. The RMSE and MAE reported comparable values, of 0.009 and 0.007 respectively. However, the R^2 was far lower, with a value of only 0.04.

With the baseline model showing a vast disparity for R^2 values between the training and test sets, further analysis was done to see if a better model could be built.

Baseline Model Accuracy Scores			
Score Metric	R^2 (correlation)	Root Mean Squared Error (RMSE)	Mean Absolute Error (MAE)
Training Set Score	0.582197679433	0.0113865390885	0.00903829888978
Test Set Score	0.0432025252226	0.00903532790804	0.00742406213061

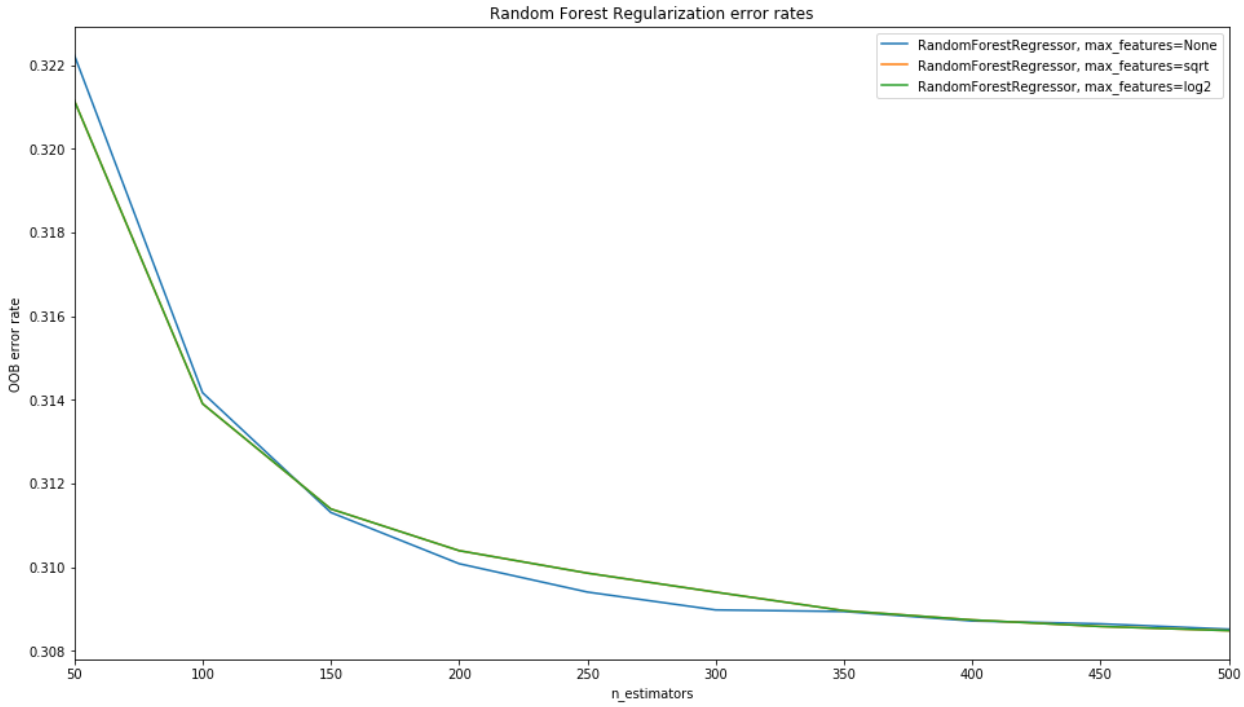
D. Extended Analysis

Lasso and Ridge regularization were both used on the baseline model in an attempt to see if the R^2 could be diminished. Both of those regularization models were computed according to a common procedure.

For the Lasso regularization, a list of possible alpha values ranging from 10^{-15} to 100 was chosen, and a for loop was written to determine the value in which the highest R^2 score can be found. In the for loop, a Lasso regressor was created for each value of alpha in the list, and then that regressor was fit onto the training set, similar to how the baseline model was fit. Then for each regression model, the R^2 value was revealed. None of the possible scores were higher than the baseline score, and the smaller the value of alpha, the higher the score. Therefore, the smallest value of alpha in the list was chosen for the test set. A dataset of predicted y-values from that model was created, and stored into the dataframe y_pred_lasso. The test scores ended up being nearly identical to the scores from the baseline model.

A similar process was followed with the Ridge regularization. A for loop was run for the same list of possible alpha values, with a Ridge regressor created for each value of alpha, and then each regressor being fit onto the training set. As with the Lasso regularization, no model was found to have a higher R^2 score than the baseline model, and the model with the smallest value of alpha had the highest score. That model was chosen for the test set, a dataset of predicted y-values was stored in the dataframe y_pred_ridge, and the test scores were nearly identical to the scores from the baseline and Lasso models.

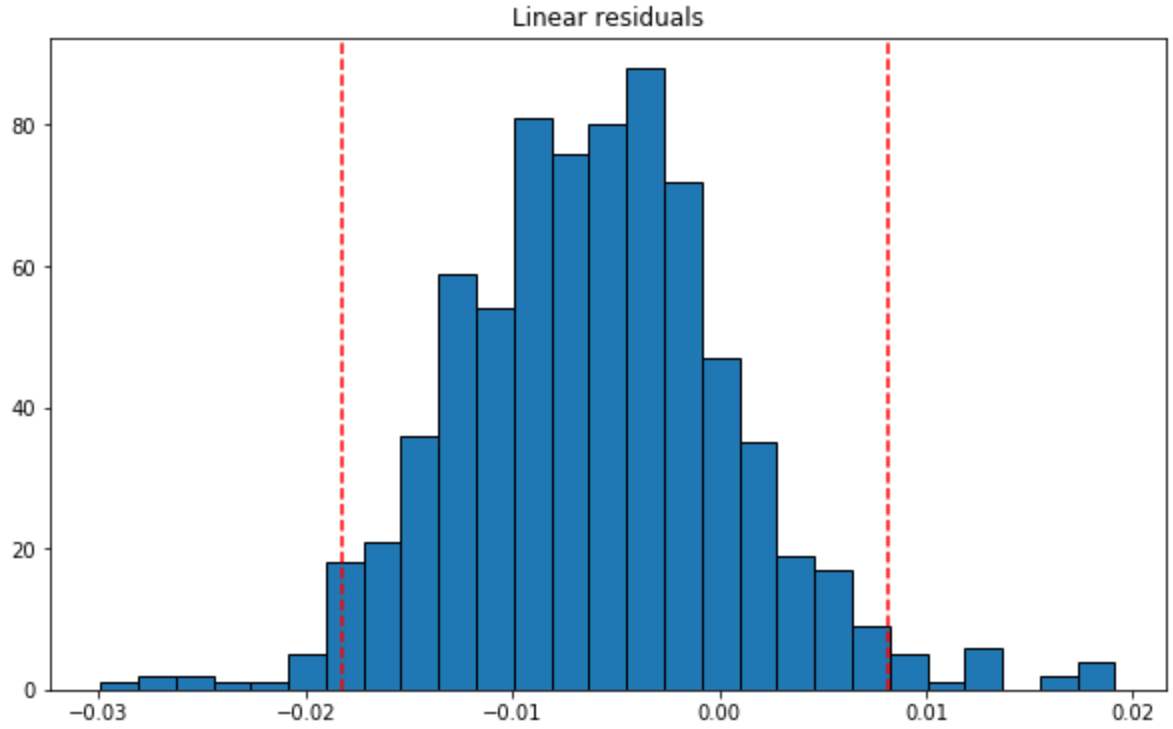
Random Forest regressors were also investigated. Three different max_features variations, and different numbers of trees were explored, from ranging from 50 to 500, with intervals of 50. Each model was fit to the training set, and the OOB error rate was plotted as shown below to minimize the error.



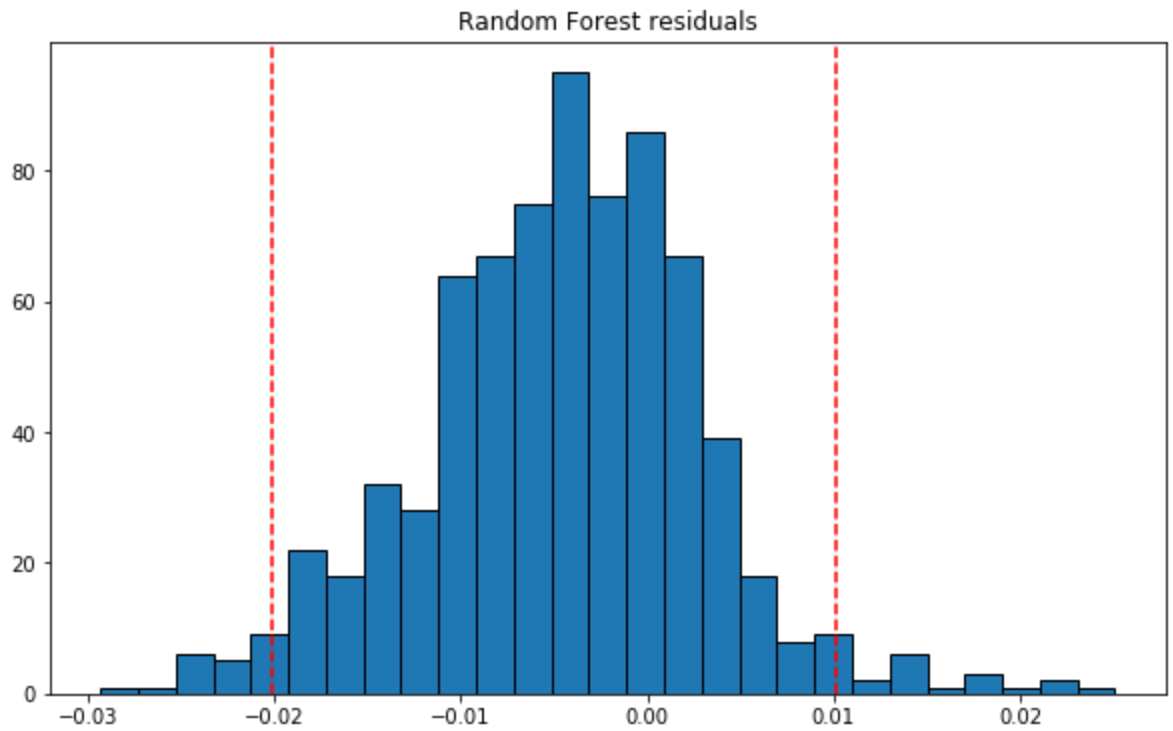
The model with the lowest error had 500 trees, with the square root and log2 max_features being a tie. Therefore, the model with 500 trees and the square root max features was fit onto the test set. The R^2 test score was slightly higher for the random forest model, near 0.09. The RMSE and MAE were slightly lower, so the random forest model was a slight improvement over the baseline, lasso, and ridge models.

Baseline and Regularized Model Accuracy Scores			
Score Metric	R^2 (correlation)	Root Mean Squared Error (RMSE)	Mean Absolute Error (MAE)
Baseline Training Score	0.582197679433	0.0113865390885	0.00903829888978
Baseline Test Score	0.0432025252226	0.00903532790804	0.00742406213061
Lasso Test Score	0.0432025252194	0.00903532790806	0.00742406213071
Ridge Test Score	0.0432025252226	0.00903532790804	0.00742406213061
Random Forest Test Score	0.089999847083	0.00881159729491	0.00678782734193

After the regression models were built, the residual scores were computed by plotting a histogram of the differences between the actual y-values of the test set and the predicted y-values of the relevant model. For the linear regression model, as well as the lasso and ridge models, 95% of the residual values ranged from -0.018 to +0.008 ppm from their predicted values. The Linear regression residuals can be seen below.



For the Random Forest residuals, 95% of the values ranged from -0.020 to 0.010 ppm, so the Random Forest residuals had slightly more spread at the 95% range, as seen below.



Since the test set for this analysis only consisted of data for a single month of December, this analysis was not representative of the full range of weather that occurs at this site, nor was there any predictive analysis focused on the summer months, which is when ozone levels are much more likely to be high enough to affect human health.

Therefore, a similar analysis was done with a focus exclusively on the summer, which for the purpose of this analysis included the months of of June through September, which was also the period in which the correlation between temperature and ozone concentration exceeded 0.6. The dataset was filtered to only include those months for all of the years of the set. The summers of 2008 through 2014 were chosen for the training set, while the summer of 2015 was chosen for the test set.

Afterwards, a baseline model was built using the same methods as the original analysis. The R^2 scores for both the training and test sets were near 0.62, with similar values for the RMSE and MAE. Therefore, there was no overfitting, and the Lasso and Ridge regularizations showed no significant changes in the values. For the Random Forest model, the decline of error rate continued after 500, so 750 trees were chosen. However, in spite of the model, the R^2 value ended up slightly lower compared to the baseline model, with slightly higher RMSE and MAE values.

Furthermore, for the summer data, the baseline model had a lower 95% residual spread than the Random forest model, with a baseline spread of -0.019 to 0.023 ppm, compared to the Random forest spread of -0.021 to 0.024 ppm. Therefore, for the evaluation of the summer months, the baseline model is the most effective model, which is to be expected when overfitting is not found.

Baseline and Regularized Model Accuracy Scores for Summer Period			
Score Metric	R^2 (correlation)	Root Mean Squared Error (RMSE)	Mean Absolute Error (MAE)
Baseline Training Score	0.61954291927	0.0120153737564	0.0093943006521
Baseline Test Score	0.616979835465	0.0108932691041	0.00867100523515
Lasso Test Score	0.616979835465	0.0108932691041	0.00867100523515
Ridge Test Score	0.616979835465	0.0108932691041	0.00867100523515
Random Forest Test Score	0.594303997616	0.0112110884086	0.00875663544491

III. Conclusions and Future Work

In conclusion, the random forest model is slightly more effective than the baseline model when building a model containing data from throughout the year.

While the summer model had similar RMSE and MAE scores to the original model, the R^2 score was far higher. With the best summer-only model having a correlation of 0.62, compared to the R^2 score of the best original model at merely 0.09, splitting the data by seasons is much more effective than creating a model incorporating data from throughout the year. Also, this model can predict the ozone level within 0.025 ppm of the actual value more than 95% of the time (for the used test set).

Ideas for future work include the following:

- Expand the number of trees in the Random Forest method. Choosing 500 or 750 trees requires significant computational power, but the data showed that the error rate trendline was still declining afterwards. However, given the limit of computational power available for this analysis, the 500 to 750 range was sufficient.

- Create a similar model for other cities, especially where air quality is a more significant issue.
- Evaluate the concentration of pollutants other than ground-level ozone, and create a similar model linking them to hourly weather data.
- Fully analyze all pollutants tracked by the EPA to create a classification model that predicts the general AQI (air quality index) by color code[4].

IV. Recommendations for Client

The clients are recommended to use the baseline model in the case of predicting ozone values in the summer months, since that is when air quality is most likely to be hazardous to human health. The high correlation and relatively low error rate can help the clients use weather forecasts to devise a more effective way to predict air quality levels. If the client believes a general margin of error of 0.02 ppm is effective enough, then this model would be highly recommended. All of the above analysis shows that as a result of the seasonal changes in both the weather and the ozone levels, a model that incorporates the data from the entire year will not be effective. Furthermore, the fact that ozone levels never reached potentially hazardous levels from October to March is another reason to disregard the colder months.

[4] "Air Quality Index - A Guide to Air Quality and Your Health." *AirNow*, Environmental Protection Agency, 27 July 2017, airnow.gov/index.cfm?action=aqi_brochure.index.