Team: DS420Factoria

Capstone Project

06/06/2018

**KSS Cup of Fresh Air**

*Section 01: Introduction/Problem Description*

Breathing; It’s the most essential part of life. As populations increase and more vehicles enter the roadways (especially in large, dense cities), a focus for maintaining air quality is critical. Studies show the health repercussions for breathing in certain air pollutants is extremely high. Particulate Matters (PM) of 2.5µm or less are said to be one of the deadliest forms because, if inhaled, can penetrate deep into the lungs and even enter the blood stream. This can go on to cause DNA mutations, neural system damage, cancer, and unfortunately premature death. Obviously, these health risks are enough to create a lot of concern in the general population of big, condensed cities. The governments of these cities (we’ll be focusing on Beijing and London) are extremely interested in finding out if any data that is already captured can be analyzed in order to identify patterns that would lead to air quality improvements. Also, in the immediate future, these predictions could help identify certain locations in the city that will have high air pollutants, leading to governments sending out warnings that people should stay indoors.

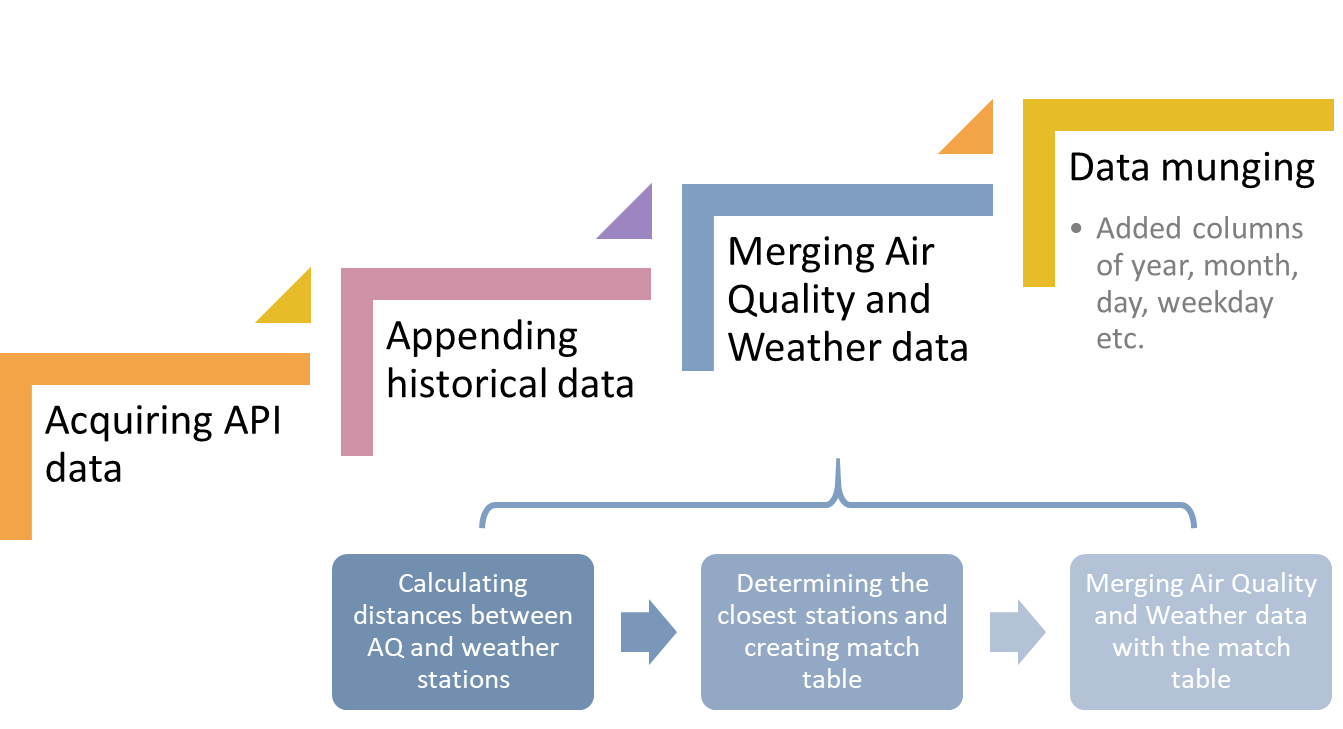
The main pollutants that we focused on in our predictions were the ones that can cause the most damage to the human population. Those are PM2.5, PM10, and 03. The challenge was to be able to predict the levels of these pollutants at every hour for 48 hours. Beijing has 35 air quality station locations, and London has 13 locations that we were required to provide predictions for. Doing the math, we had predictions for 48 stations, and each station would have 48 hours of predictions for each of the 3 pollutants (48\*48 = 2,304 rows of data and 3 columns for each pollutant).

The major downside of this project was not being able to see the results right away. Yes, we could break up the data into a training set and testing set (which we did), and see how our predictions compared to actuals, but the results weren’t as understandable. This is because the KDD Cup created a complex calculation they used to evaluate our submissions (The Symmetric mean absolute percentage error – “SMAPE”). For example, even if our R-squared result was awful when testing against our regression training model, it might surprisingly perform relatively well when submitted for evaluation. That was what caused some confusion around this project; We had to wait 3 days after each submission, and then we were able to see how our predictions of these 3 values matched the actual air quality data using the SMAPE evaluation. We had to make sure we documented what we did for each submission, so that we could understand how our different models and model parameters behaved. There was also the fact that some things are unpredictable; one day the air quality could be due to something that we don’t collect data on, and our model performs poorly. The next day, the same model could perform extremely well because that unknown factor is no longer present. In this way, sometimes it was difficult to understand why we received a poor score (or stellar score!).

The data that we did have available to us was the hourly 2017 historical data for each of the air quality stations in which we had to predict. Along with this, we had up to the hour API data that we could bring in at any time which we were able to take advantage of. Before our submissions, we could bring in the latest which at least helped in our ARIMA time-series model. We also were able to extrapolate more features with the features that were given to us. For example, with the date/time column, we could obtain the hour, the month, whether it was a weekday or not, holiday, etc. (We’ll expand on this feature engineering topic later in this paper). Along with air quality station data, for each city location (Beijing and London), we had weather station data which was collected in a grid fashion (imagine a square grid placed over the city). These stations collected data on weather (ex. Rain, Partly cloudy, sun, etc.), temperature, pressure, humidity, wind direction, and wind speed. We ended up using historical weather data and future forecast data in our linear regression model during the last couple of days in the competition. For our other models and rest of the days, we ended up using only historical weather data. This could have been helpful in improving our model, but we didn’t find the time or have the necessary knowledge to add it into our models until late in the competition. We will talk more about how we utilized this weather data later in this paper. Other than what was provided on the KDD cup website, we did not use outside resources as it was against the rules of the competition.

As far as submitting each of our models, it would have been helpful to set up something that automatically collected the API data, ran our model, and submitted on a daily-basis. Since we are amateurs in this field and did not get help in setting something up like this, we had to make sure that we were available to submit before UTC midnight (5pm Pacific time) which sometimes proved to be difficult. That being said, let’s get into the details of our project and what we were able to accomplish. In this paper we will discuss how we analyzed our data and how we created features with the data that was given to us. Furthermore, we will describe how we selected the models we used, and how we changed different model parameters to improve and obtain the best results.

*Section 02: Data Description*



*Section 03: Data Exploration*

The primary tool that was used for data exploration was the Interactive Data Exploration, Analysis and Reporting (IDEAR) that is available as part of Data Science Utilities for TDSP from Microsoft. The below summary of the data exploration step will focus mainly on an example of one air quality station (London BLO) and hourly weather data from the closest weather station (Camden – Bloomsbury). The methodology for mapping air quality data to weather data is covered in the next section.

The first step in the data exploration process involved plotting the hourly air pollution levels over several different time periods. As expected, there are clear seasonal patterns and trends in the air pollution levels. The below chart shows a clear daily pattern in air pollution levels, presumably related to the number of cars on the road during peak traffic periods. This suggested that the seasonal patterns should be explored further using autocorrelation analysis, time series modeling or regression analysis with time segments (Month, Day of Week, Hour of Day, Weekend, etc.) as features.

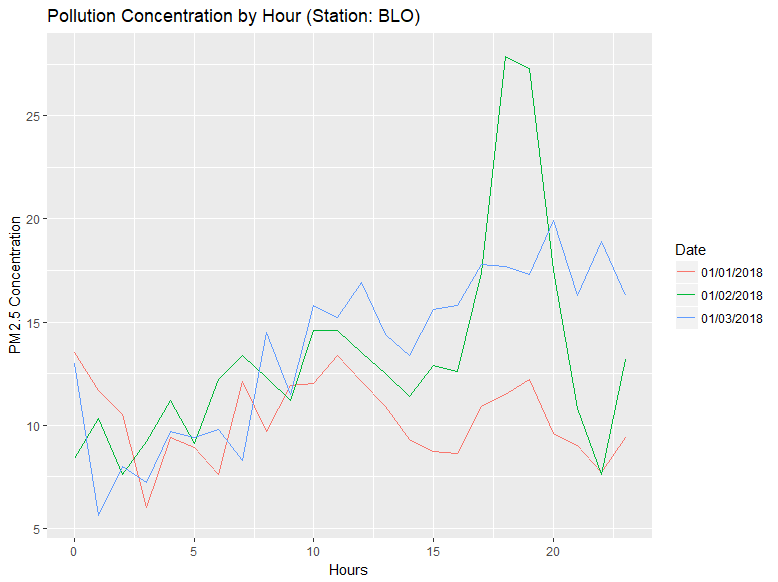


Chart: Pollution Concentration by Hour for three days in Jan-2018 (Station: BLO; PM2.5 Pollution)

There are several important qualities of the target variable (PM2.5) that are visible in the below charts. The distribution is non-normal with some right skewness. The test of for normality is failed at the 95% confidence level. The boxplot indicates there are many outliers present which suggest that the outliers could have a significant impact on the forecast and should be explored further during feature engineering. Finally, a summary of missing values indicates that 10% of the values for PM25 are missing. The missing values are concentrated toward the newer observations. Methods for dealing with missing values should be applied and tested for during the modeling phase.

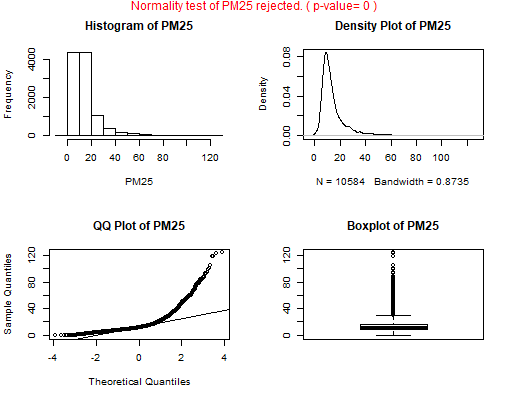


Chart: Visual of the PM25 variable

Creating a correlation matrix provided some important insights into the variable relationships. As expected, there are a several variables that have strong correlations. The three pollution species variables have a strong positive correlation, especially PM10 and PM25. This makes sense given that pollution levels should have similar sources. In this case the drivers are likely traffic levels, industry pollution and weather but this should be confirmed later. The time variables (Month, Day of Week, Hour of Day, Weekend, etc.) are also strongly correlated which is not surprising. Interestingly, wind direction and wind speed are the two weather variables that are most highly corrected to pollution levels. Given the high possibility of multicollinearity among the variables, feature selection is an crucial step in the process.

*Section 04: Feature Engineering*

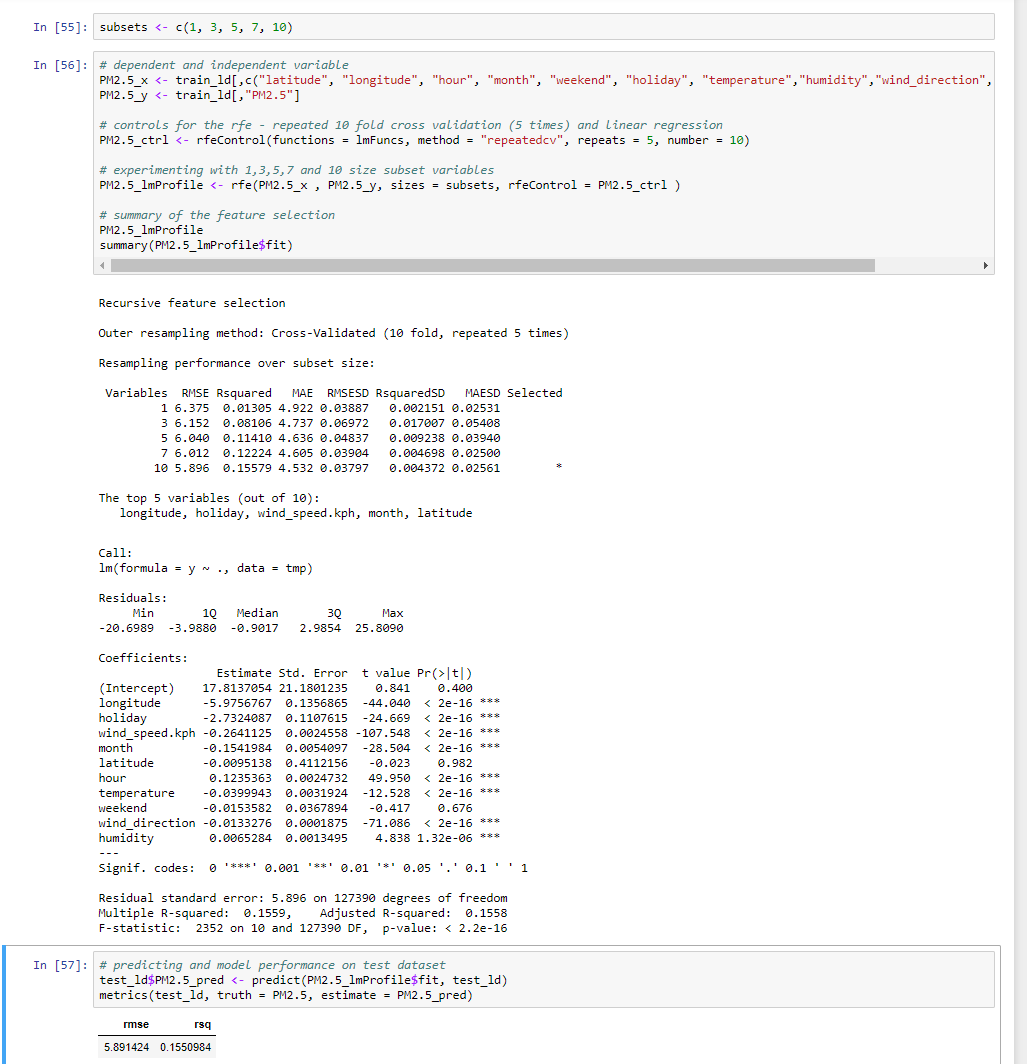
Our approach to feature engineering was representing variables differently for experimenting with different model algorithms – turning the inputs in the dataset into things that the algorithm can understand.

* So, we used the datetime variable in the dataset (utc\_time) to generate three components of the variable – ***hour, month and date***.
  + Hour variable seemed to have significant relation with the air quality variables – Understanding: time of the day matters due to factory/manufacturing industries open times and traffic patterns
  + Month variables also mattered to predict air quality variables – Understanding: Weather patterns do have impact on the air quality
* Additionally we added couple of variables – ***weekend and holiday*** based on a basic understanding of what could affect air quality around the stations.
  + If it was a weekend, there could be high traffic which affects air quality
  + If it was a holiday, closures in manufacturing industries/holiday traffic patterns affect air quality
    - Used an external data source suggested in ‘Discussion board’ to identify dates considered as holidays - <https://www.timeanddate.com/holidays/china/2017> , <https://www.timeanddate.com/holidays/china/2018> , <https://www.gov.uk/bank-holidays>

*Section 05: Feature and Model Selection*

Linear Regression Model

Post converting the time variable into various component features, for the 48 air quality stations in Beijing and London and each of air quality variables – PM2.5, PM10 and O3, we built a linear regression model (3 models each for Beijing and London) using caret package in R.

* **Train/Test data:** The Beijing/London dataset was partitioned into training and testing data set with training dataset containing 80% of observations/rows and testing data set containing 20% of observations/rows
  + Beijing dataset: Approx. 320,000 rows for training set and 80,000 for testing set
  + London dataset: Approx. 120,000 rows for training set and 32,000 for testing set
  + Used sample() function for random sampling
* **Features:** Each of the datasets had the following features apart from the air quality dependent variables – station name, lat and long data, corresponding nearest grid station, utc time at which the observation was recorded, hour, month, date, temperature, pressure, humidity, wind direction, wind speed in kph, whether weekend/not, whether the day was a public holiday or not.
* **Methods for improving model accuracy:** In order to better evaluate the model performance on unseen data (other than partition data as training and testing dataset), we used repeated 10-fold cross validation (repeat = 5 times).
* **Feature selection:** Recursive feature elimination, rfe(), using caret package was used to identify the variables used in the model. We used wrapper function such as RFE to optimize model performance effectively and automatically as there was high variability in the data available on a daily basis. We also experimented with variable subset size in the rfe package.
* Apart from using repeated k**-**fold cross validation and recursive feature elimination to automatically enhance the model, we were on the lookout to validate that the rmse doesn’t differ much for training set and the testing set to ensure that the model is not over fit.
* PM2.5 model was built using time and weather variables; For PM10, additionally predicted PM2.5 variable was used and for O3, predicted PM2.5 and PM10 variables were used.
* **Observations:** 
  + The model averaged 0.56 SMAPE over 27 submissions with minimum of 0.39 and maximum of 0.67
  + Before doing missing value imputation and outlier elimination, for Beijing the PM2.5 and PM10 model had rmse = ~65 whereas post the treatments the rmse reduced to ~36.
  + Recursive feature selection seemed to be working well even though it was computationally intensive as it reduced the rmse by atleast 5 for the Beijing PM2.5, PM10 and O3 models compared to a model without feature selection.
  + In case of Weather data for the future 2 days, we used caiyunapp’s data. There were some missing values in this data as it was real time. In order to impute for the missing values, we used the previous day’s data for the stationId, hour to match the predictor variables as closely possible.
* **Sample code & output :** 

ARIMA Model

We decided to try an ARIMA model with this data since it is a time-series forecasting based model. Unlike a regression model where you get to choose what features to use in the model, and there’s a whole process to figure out which features would be best to use, ARIMA is simply based on time and the feature you are trying to predict. For example, to predict PM2.5, all we needed to put in the model was the timestamp and the history of PM2.5 up until the latest hour. We chose the past month of API data so that the model would be able to hopefully deduce some seasonality and trends. Another task to make sure to accomplish before running the model was to take care of the rows of data that had NAs. As you can imagine, NAs can be especially detrimental in time-series modeling due to the gaps of data causing the predictions to be skewed. Therefore, it was imperative to take care of the NAs. To accomplish this, we used the last hour at that station to fill in any future hours of data. We were assuming that a station would not be unavailable for long gaps of time. We were hoping to be able to figure out how to take the average of the last hour recorded and the next hour recorded, and smooth the NAs between, but it proved to be more complex than we thought. We settled with just carrying over the past hour recorded over the future NAs using na.locf in the zoo package.

Once we took care of NAs, it was time to decompose our data. The goal of this step was to be determine if there were seasonal components and/or trend components. Overall, most stations in the past month only had seasonal components based on time, and weekend/weekday. Therefore, each station seemed to have a seasonal component, and were relatively stationary. We used auto.arima to build the model with ‘seasonal = TRUE’. We had to loop each station through the set of code because each was unique. Then we had to duplicate that loop 5 times (once for each pollutant: PM2.5, PM10, and O3 for Beijing, and PM2.5, PM10 for London). Writing the R code and getting it set up in the correct format for submission was difficult since it was all new to us, but we eventually figured it out, and can chalk it up to a good learning experience. The results that we got back after waiting a few days was decent and performed just slightly better than the regression model used at that time. We would have liked to test other settings and parameters with ARIMA, and possibly ensemble it together with our regression model since that is known to be a worthwhile technique.

*Section 06: Model Performance*

*Section 07: Collaboration*

*Section 08: What would you have done differently?*

In general, we are satisfied with the outcome of the project given the deadline. With additional time, there are several improvements that could have been made. The first would have been to submit earlier. The relatively small number of submissions hurt our overall score on the leaderboard. It also would have been beneficial to create an ensemble model that combined the linear regression and ARIMA models, instead of submitting results separately. Finally, we made considerable progress toward an LSTM model that showed a significant improvement in performance over the other models. Unfortunately, with the time constraints we were not able to use the model for our forecast submissions.