**Data Science 420**

**Capstone Project Report**

**KDD Beijing and London Air Quality Prediction**

**Team - DS420两条Malax**

**Michael Li & Christo Palaskas**

**Problem Description:**

We were given a historical data set containing the air pollutant values from 35 stations in Beijing and 13 stations in London since January 1st, 2017, on an hourly basis. We were also given daily the hourly updates from the same locations. Those values were PM2.5, PM10, O3 (which were the target of the predictions, except O3 for London only), and NO2, CO, SO2.

We were also given the meteorological data from the same period in the form of a grid that superimposed the aforementioned air quality stations. The given data covered the same period of time, and included: temperature, pressure, humidity, wind direction and wind speed.

Finally, we were given the coordinates of the grid points of the meteorological report as well as the coordinates of the air quality stations, in Beijing and in London.

We were asked to predict the pollutant levels of PM2.5, PM10 and O3 for Beijing, and PM2.5 and PM10 for London for the next 48 hours starting at 00:00 UTC every day.

**Data Description:**

We used the historical data set provided and the meteorological data set provided to create models for the predictions. We also used an external API to obtain the meteorological data for the next 48 hours needed for the predictions. The external API used was:

<https://openweathermap.org/forecast5>

which returned a 5-day forecast with a 3-hour interval.

**Data exploration and analyzing:**

The first thing we noticed was that some stations did not contain data at some hours. So we had to use Kalman Filters to extrapolate for the missing data. And since Kalman filters could return negative numbers depending on the trend, we had to set every negative to zero.

Also we found duplicate records (multiple data from exact same hour) in the dataset. After we applied Kalman imputing the missing hours, the duplicate hours were then removed from the dataset.

**Feature Engineering:**

We created a feature for the prediction of each pollutant by taking the value of the same hour but one week ago, thinking that pollutant producing entities operate on a weekly cycle, such as factories and cars. We also created a feature using the data from 2 days ago, to accommodate for the weather being responsible for pushing away masses of air from a region.

**Feature Selection:**

Besides the features we created we also used the time as a feature, believing that the time of the day plays an important role to the pollutant levels, as does the day of the week, the month and the season. We also added the year.

We also used the meteorological data provided. We used the temperature, wind speed, wind direction (which was transformed to 8 categories, N, NE, E, SE, S, SW, W, NW), pressure and humidity.

Using the importance function of the Random Forest library we discarded the values of other pollutants as not so important features.

**Model Selection**

Before any feature engineering we created a simple time series prediction model using ETS. We also tried ARIMA but ETS returned better values. That gave us the first lower-than-2 score on the leaderboard, just as a placeholder and submitting mechanism validation. To our surprise it worked pretty well, giving us a score of 0.665. That gave us the encouragement to start working more on data preparation and feature engineering so we could create a model.

The first model we created was a Random Forest model. We experimented with different number of trees, from 50 to 500, different mtry and nodesize. We learned that mtry sets the number of random features selected to evaluate at each split and that it is best set close to one third of the number of total features we were using. We also learned that changing the nodesize changes the size of the tree, the largest the nodesize the smaller the trees created.

The next model we tried was the Support Vector Regression which produced very strong results. At first Polynomial kernel function was used. Later we switched to Radial Basis Function to do regression and tried different values of the epsilon, cost, and gamma hyper-parameters. Higher gamma values yielded flatter lines of predictions, so at the end 0.02 gamma was used to produce the more likely variation that we needed.

We also trained a model using only data from the end of March 2018 because it included one more feature we could use: the weather description, as Sunny, rainy, cloudy etc. We mapped the return from the weather API to match the data we had and trained one more SVR model with only a few months of data instead of the whole period from January 2017.

**Model Performance**

We introduced the validation mechanism in our algorithm. There were 2 modes of training the

model: for submission and for validation. When we trained for validation we removed the last 48 hours of the latest data we had and used it to cross validate the predictions from our model, using the KDD scoring method (Symmetric Mean Absolute Percent Error). This method would prove handy later on, when we would find the optimal weights for the different models. Of course when we trained for submission we used all the data and predicted for the 48 hours after the last hour of the current day. To achieve this we converted all times to UTC using the Posix format.

Every model performed very differently during the competition. There were days that ETS performed really well, and days that SVR performed better. But as a rule of thumb most days we had the best results when we included all 4 models equally (0.25 ETS / 0.25 RF / 0.25 SVR / 0.25 SVR with weather description). We used our intuition for the final weighting of the models to submit our predictions but as an indicator we ran the validation mode on all 4 models and then used 4 nested loops that we tried all weights with a sum of 0 to 1.5 and found the combination with the smallest SMAPE.

**Leaderboard**

Our team is top of the class with public score of 0.4865 over the entire month-long competition. Our final weighted score is 0.4625, ranks 63rd in the leaderboard.

**Distribution of Work**

Most of the work was done physically together, because we work at the same place. We would discuss each and every step and write code together, after agreeing on our next move. One person would code and the other would research syntax and tactics. But we also did some of the work separately as follows:

Michael: SVR models, Neural Networks, Kalman filters

Christo: Model validation, Meteorological Data feature extraction

**Retrospective**

Firstly we would like to emphasize on our team size, which was the smallest possible team (one would be individual, not team). The pros were that we did all the steps and had hands on experience to every aspect of the process, and that we didn’t have much overhead for managing. (We could just email each other instead of setting up Github repository etc). The cons were that we were stretched thin and didn’t manage to do everything we planned to.

As we kept on working we realized that we didn’t have unit testing. We kept building on previous steps without being able to validate if everything was still correct. For example, we divided the historical data into 48 files, one for every air quality station. Then we added features as columns to that file, one by one, without validating that we didn’t mismatch an hour or a day and everything has an offset.

Another aspect we lacked was plenty of visualization. We used Kalman filters to extrapolate for missing values or missing days but we didn’t plot all the stations’ data to see what exactly did the kalman filters do. Sometimes the filters returned negative numbers which we had to truncate to zero.

We also should have spent more time on feature engineering and selections. When we checked the correlation between the pollutants we found out that for instance NO2 was very strongly correlated to PMs, we could have included these as features but that required getting third party forecast values to do our prediction. Another step that we did not have time and resource to do properly was hyper-parameter tuning for our Random Forest and Support Vector Regression models. Using our personal laptops and the specs costed us a significant amount of time in preparing and computing the data we needed to train during the first phase of the competition. If doing this again we would plan early and use more tools from web services.

**Conclusions**

Overall this was a wonderful experience and a great opportunity to participate in a data science competition. It was an opportunity to put together everything we had learned over the past 8 months in theory and make something practical out of it. Even the final prize incentive played a small factor to our work, just as an idea in the back of our minds. Small, because we knew that we couldn’t compete with people that work full time on it. But still trying to be at least the first team in our class was a strong enough motive to make us work a few weekends and spend a couple hours after work every day to the Capstone Project.