Michael Kolonay, Anna Landi, Maureen O’Shea, David Vann
mhk9c, aol4h, mo2cr, dv6bq

CS 5010: Group Project Report

Introduction (Background and Data in Context):

For this project, we selected the following data set from the UCI Machine Learning Repository: “Beijing Multi-Site Air-Quality Data Data Set”. Overall, this data set contains information on the concentration levels of six major air pollutants (PM2.5, PM10, SO2, NO2, CO, and O3) and measurements other important meteorological variables (temperature, pressure, dew point temperature, precipitation, as well as wind direction and speed). Furthermore, the data are from twelve official air quality monitoring stations throughout Beijing (Aotizhongxin, Changping, Dingling, Dongsi, Guanyuan, Gucheng, Huairou, Nongzhanguan, Shunyi, Tiantan, Wanliu, and Wanshouxigong) over a four-year period (from March 1, 2013 to February 28, 2017) and totalling over 420,000 data points.

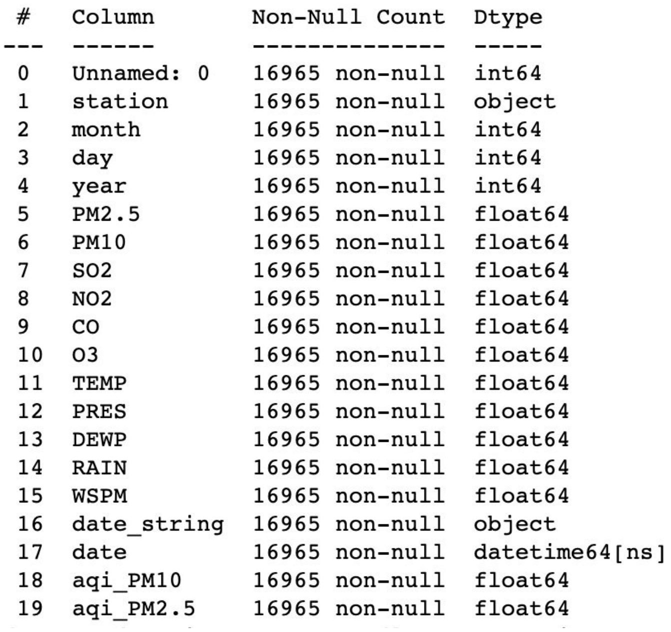
Therefore, this data set has multivariate components and is time-series in nature. Note: PM2.5 and PM10 are fine inhalable particulates that are 2.5 and 10 micrometers in size, respectively - these particulates are so small and so dangerous because they can “penetrate the lung barrier and enter the blood system...and increase the risk of heart and respiratory diseases, as well as lung cancer” (World Health Organization, 2020).

We found this data set to be particularly interesting because China has struggled with serious air quality issues in the past, which not only impact its locals, but also people in other parts of the world. Importantly, there are very serious, even life-threatening health consequences associated with poor air quality, including damage to one’s lungs, heart, and brain. China established “The Action Plan” back in 2012 to address air quality concerns - the initiative outlines the following goals: "by 2017 the urban concentration of PM10 must decrease by 10% compared with 2012, and as a result the annual number of days with fairly good air quality should gradually increase", "concentration of PM2.5 in the heavily polluted Beijing-Tianjin-Hebei, Yangtze River Delta, and Pearl River Delta regions must fall by around 25% and 15%, respectively", and "PM2.5 annual concentrations in Beijing must be controlled below 60 milligrams per cubic meter (mg/m^3)" (Library of Congress Law, 2020).

Data Exploration and Analyses:

After selecting this data set, we performed data pre-processing and cleaning, which included converting individual date fields (year, month, and day) into combined strings and timestamps, as well as removing duplicates and nulls (missing data) from the data. Additionally, the data were originally separated by station (total of twelve .csv files) and so we merged the data frames. Importantly, the data were collected multiple times a day (hourly) at each station, for plotting purposes we grouped the data by station and date, taking the maximum value for each of the attributes as the value for the day.

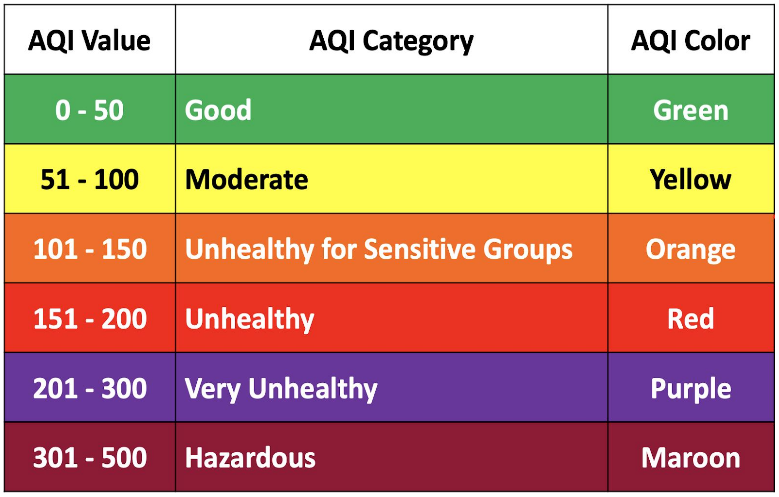
Figure 1: New, Cleaned Data



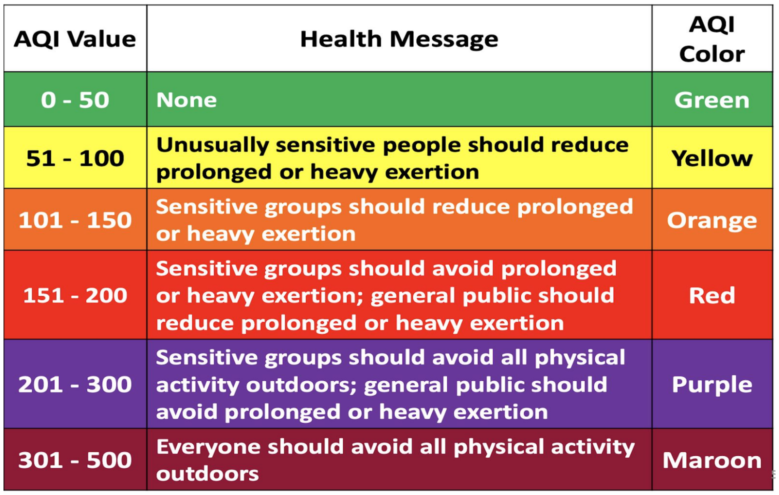
```
 #   Column       Non-Null Count  Dtype
---  ------       --------------  -----
 0   Unnamed: 0   16965 non-null  int64
 1   station      16965 non-null  object
 2   month        16965 non-null  int64
 3   day          16965 non-null  int64
 4   year         16965 non-null  int64
 5   PM2.5        16965 non-null  float64
 6   PM10         16965 non-null  float64
 7   SO2          16965 non-null  float64
 8   NO2          16965 non-null  float64
 9   CO           16965 non-null  float64
 10  O3           16965 non-null  float64
 11  TEMP         16965 non-null  float64
 12  PRES         16965 non-null  float64
 13  DEWP         16965 non-null  float64
 14  RAIN         16965 non-null  float64
 15  WSPM         16965 non-null  float64
 16  date_string  16965 non-null  object
 17  date         16965 non-null  datetime64[ns]
 18  aqi_PM10     16965 non-null  float64
 19  aqi_PM2.5    16965 non-null  float64
```

Next, we used a library called python-aqi 0.6.1, which converts between AQI values and pollutant concentrations. AQI stands for air quality index and it is a standard measure of air quality conditions and corresponding health risks. We chose to calculate the maximum (as opposed to the average) daily AQI values of PM2.5 and PM10 for each station. Note: readings greater than 500 are considered beyond the AQI, but our data set contains readings that are in the 800s and 900s (PM2.5 max. = 844 and PM10 max. = 999), so we set all readings above 500 to a default number of 501. Consequently, proceed with caution when interpreting AQI values, as a reading of 501 could really mean a value in the 800s or 900s.

Figures 2 & 3: AQI Categories and Health Messages



AQI Value	AQI Category	AQI Color
0 - 50	Good	Green
51 - 100	Moderate	Yellow
101 - 150	Unhealthy for Sensitive Groups	Orange
151 - 200	Unhealthy	Red
201 - 300	Very Unhealthy	Purple
301 - 500	Hazardous	Maroon



AQI Value	Health Message	AQI Color
0 - 50	None	Green
51 - 100	Unusually sensitive people should reduce prolonged or heavy exertion	Yellow
101 - 150	Sensitive groups should reduce prolonged or heavy exertion	Orange
151 - 200	Sensitive groups should avoid prolonged or heavy exertion; general public should reduce prolonged or heavy exertion	Red
201 - 300	Sensitive groups should avoid all physical activity outdoors; general public should avoid prolonged or heavy exertion	Purple
301 - 500	Everyone should avoid all physical activity outdoors	Maroon

We carried out basic exploratory data analyses to get a better sense of the data, which included plotting box plots, bar charts, and scatter plots. We also calculated percentiles and interquartile range for the different pollutants and meteorological variables.

Figure 4: box plot of pollutants and meteorological variables

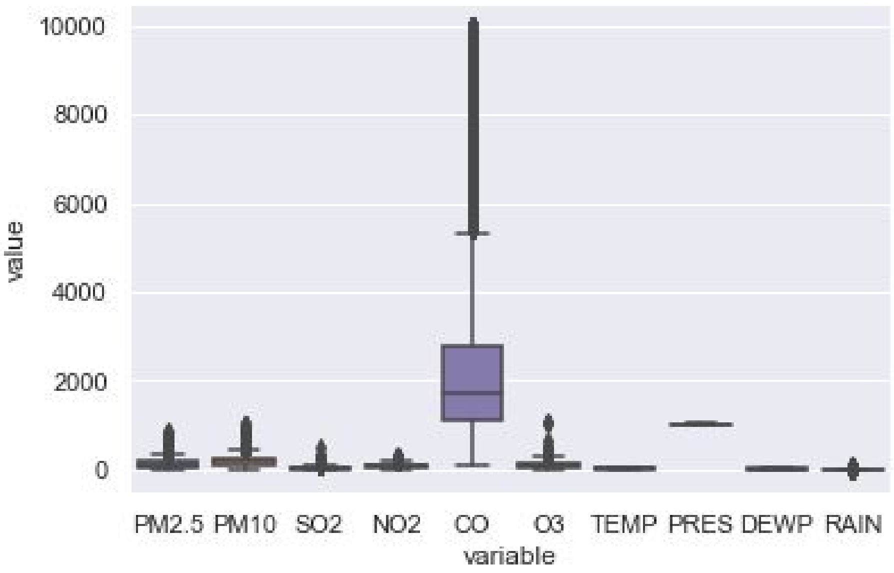


Figure 5: bar chart of PM10 by number of days

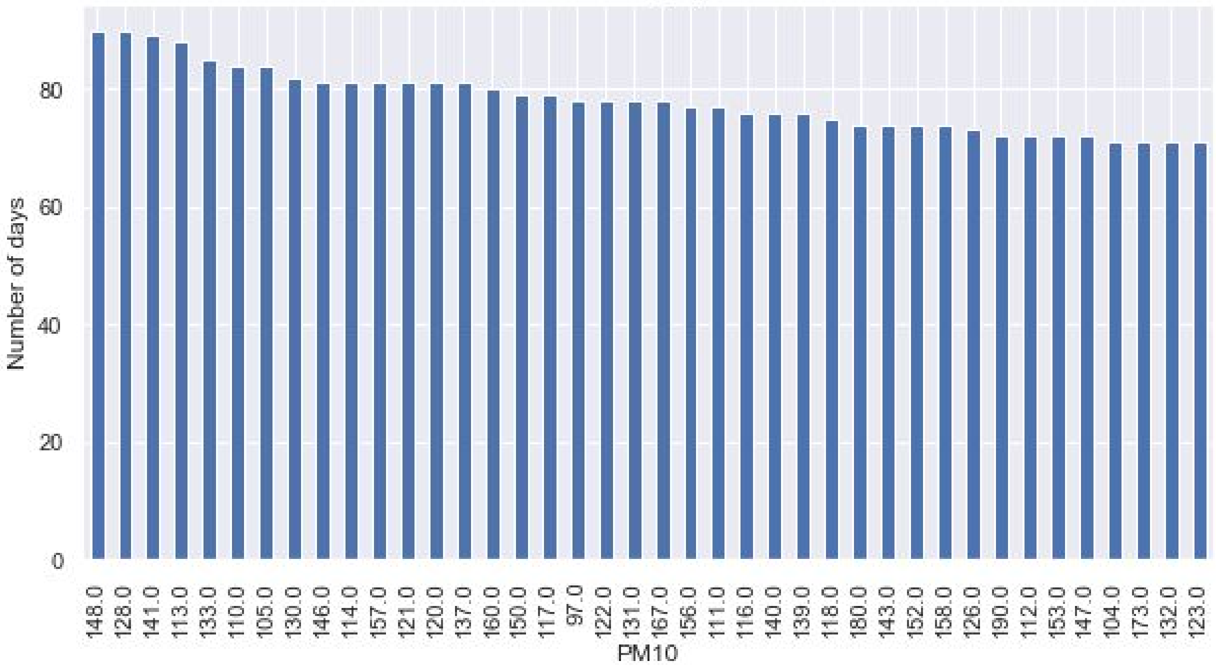
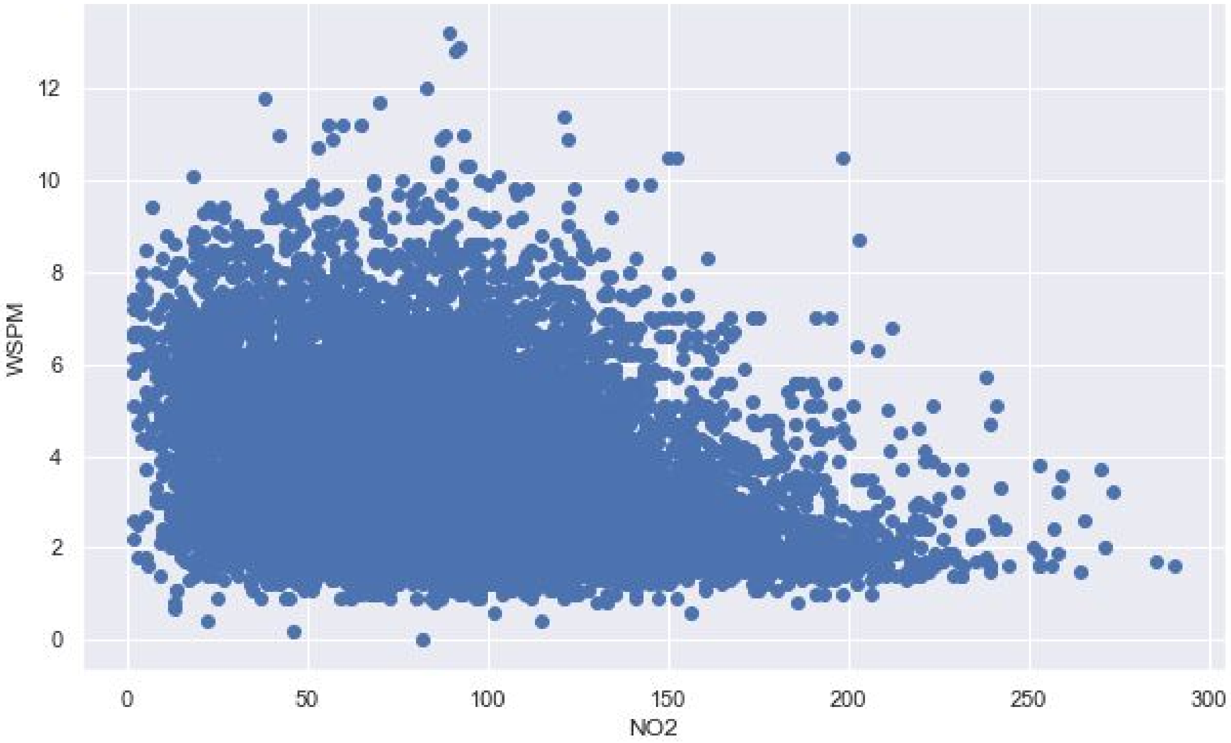
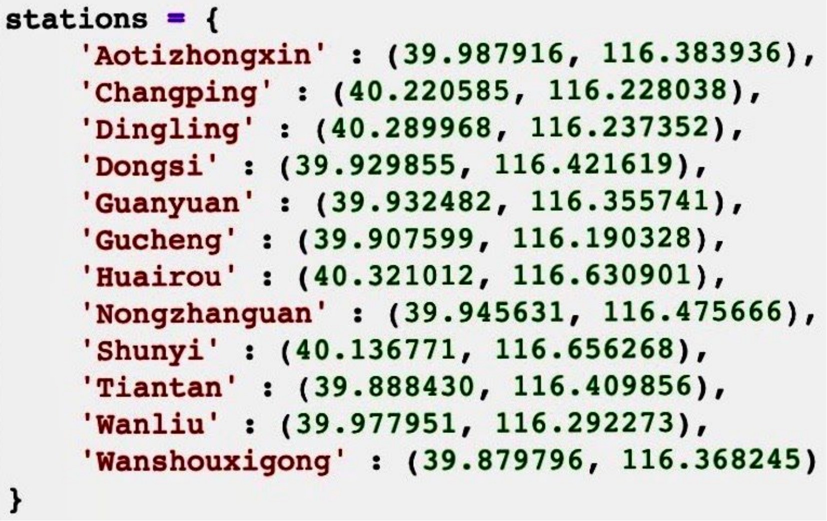


Figure 6: scatter plot of wind speed against NO2 concentration



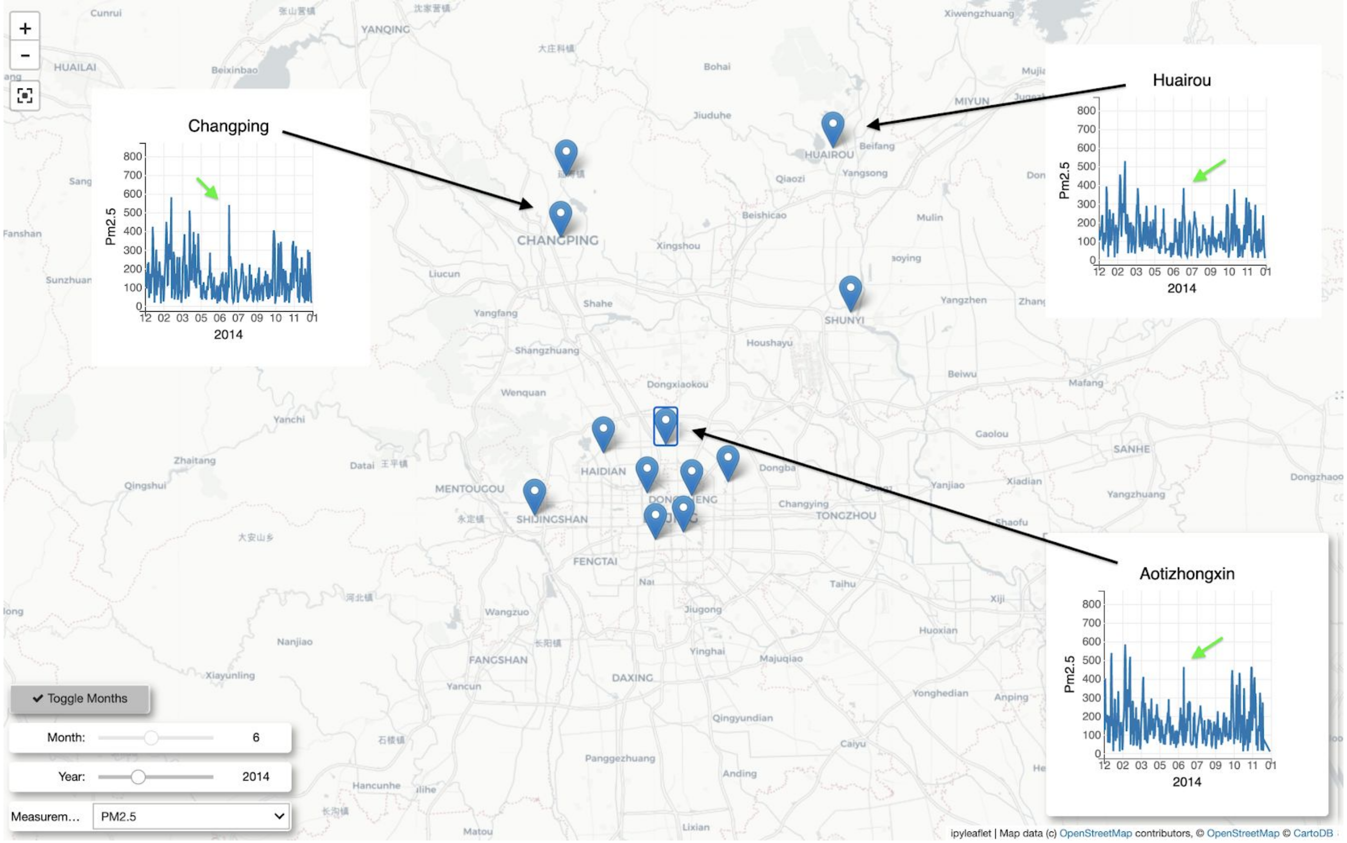
From here, it became clear that we really needed more than just static images to understand the data and hence we decided to create an interactive map with a hover over function and graphs that allow users to query the data against month, year, pollutant, meteorological variable, and station - our visual application has toggle and slider features for both month and year to allow the user to easily view data for a specific month/year, or for all 12 months/4 years. We also added a drop-down menu for users to select from the list of various pollutants and meteorological variables. Also, we utilized Google Maps to search the latitude and longitude of each weather station, which we then plugged into our interactive map. We used the available data from 2013 to 2017 to develop this prototype as proof of concept. Overall, our application is a great example of how we began this process with data, which we subsequently transformed into new knowledge to reveal interesting and useful information.

Figure 7: coordinates of each weather station



```
stations = {
    'Aotizhongxin' : (39.987916, 116.383936),
    'Changping' : (40.220585, 116.228038),
    'Dingling' : (40.289968, 116.237352),
    'Dongsi' : (39.929855, 116.421619),
    'Guanyuan' : (39.932482, 116.355741),
    'Gucheng' : (39.907599, 116.190328),
    'Huairou' : (40.321012, 116.630901),
    'Nongzhanguan' : (39.945631, 116.475666),
    'Shunyi' : (40.136771, 116.656268),
    'Tiantan' : (39.888430, 116.409856),
    'Wanliu' : (39.977951, 116.292273),
    'Wanshouxigong' : (39.879796, 116.368245)
}
```

Figure 8: map and graphs (PM2.5 concentrations at three stations in 2014)

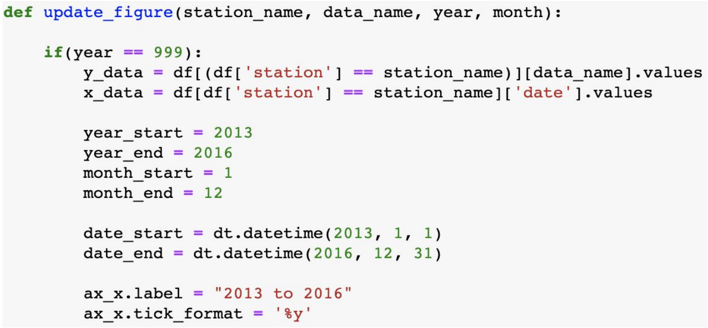


Month: 6

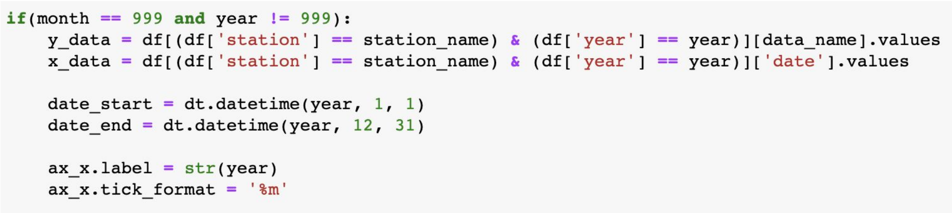
Year: 2014

Measurem... PM2.5

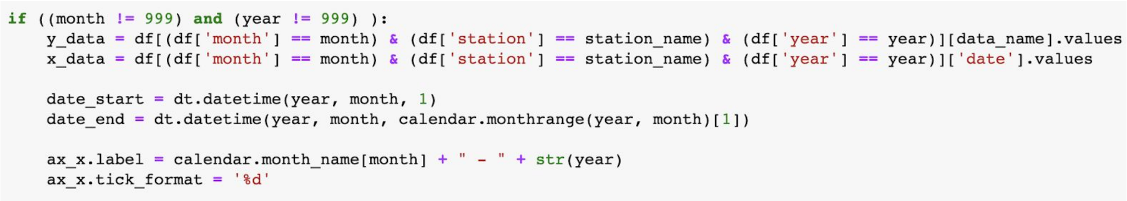
Examples of our parameterized queries:



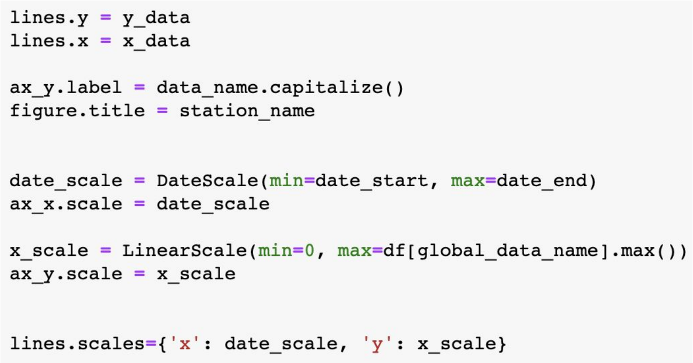
```
def update_figure(station_name, data_name, year, month):

    if(year == 999):
        y_data = df[(df['station'] == station_name)][data_name].values
        x_data = df[df['station'] == station_name]['date'].values

        year_start = 2013
        year_end = 2016
        month_start = 1
        month_end = 12

        date_start = dt.datetime(2013, 1, 1)
        date_end = dt.datetime(2016, 12, 31)

        ax_x.label = "2013 to 2016"
        ax_x.tick_format = '%y'
```



```
if(month == 999 and year != 999):
    y_data = df[(df['station'] == station_name) & (df['year'] == year)][data_name].values
    x_data = df[(df['station'] == station_name) & (df['year'] == year)]['date'].values

    date_start = dt.datetime(year, 1, 1)
    date_end = dt.datetime(year, 12, 31)

    ax_x.label = str(year)
    ax_x.tick_format = '%m'
```



```
if ((month != 999) and (year != 999) ):
    y_data = df[(df['month'] == month) & (df['station'] == station_name) & (df['year'] == year)][data_name].values
    x_data = df[(df['month'] == month) & (df['station'] == station_name) & (df['year'] == year)]['date'].values

    date_start = dt.datetime(year, month, 1)
    date_end = dt.datetime(year, month, calendar.monthrange(year, month)[1])

    ax_x.label = calendar.month_name[month] + " - " + str(year)
    ax_x.tick_format = '%d'
```



```
lines.y = y_data
lines.x = x_data

ax_y.label = data_name.capitalize()
figure.title = station_name


date_scale = DateScale(min=date_start, max=date_end)
ax_x.scale = date_scale

x_scale = LinearScale(min=0, max=df[global_data_name].max())
ax_y.scale = x_scale


lines.scales={'x': date_scale, 'y': x_scale}
```

Testing:

For testing, we selected a few stations to examine and ensure all fields appeared in the correct format. Additionally, besides running unit tests for some of the functions, we carried out visual examinations - for example, we used an online AQI calculator to cross-check our AQI values. Also, we compared our map of the different weather stations with the map from the journal article that this data originally comes from.

test_load_csv_data_correctly

Test if we have loaded the data into the dataframe from the csv files correctly.

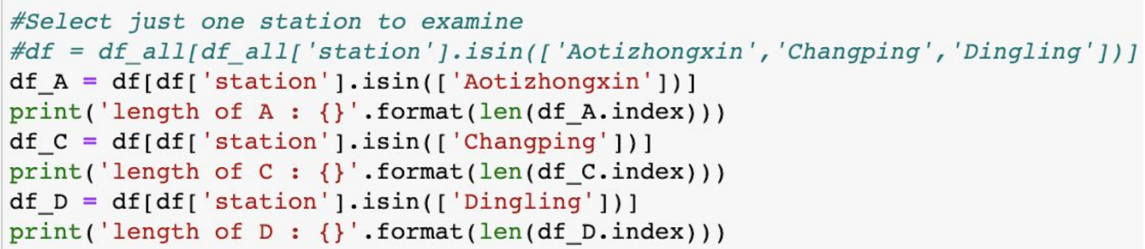
test_create_timestamp_correctly

test if we have converted the month, day, year column correctly into a date string and datetime.

test_create_aqis_correctly

Test if we have added the aqi column and if the aqi has been tested correctly.

Figure 9: example of testing data

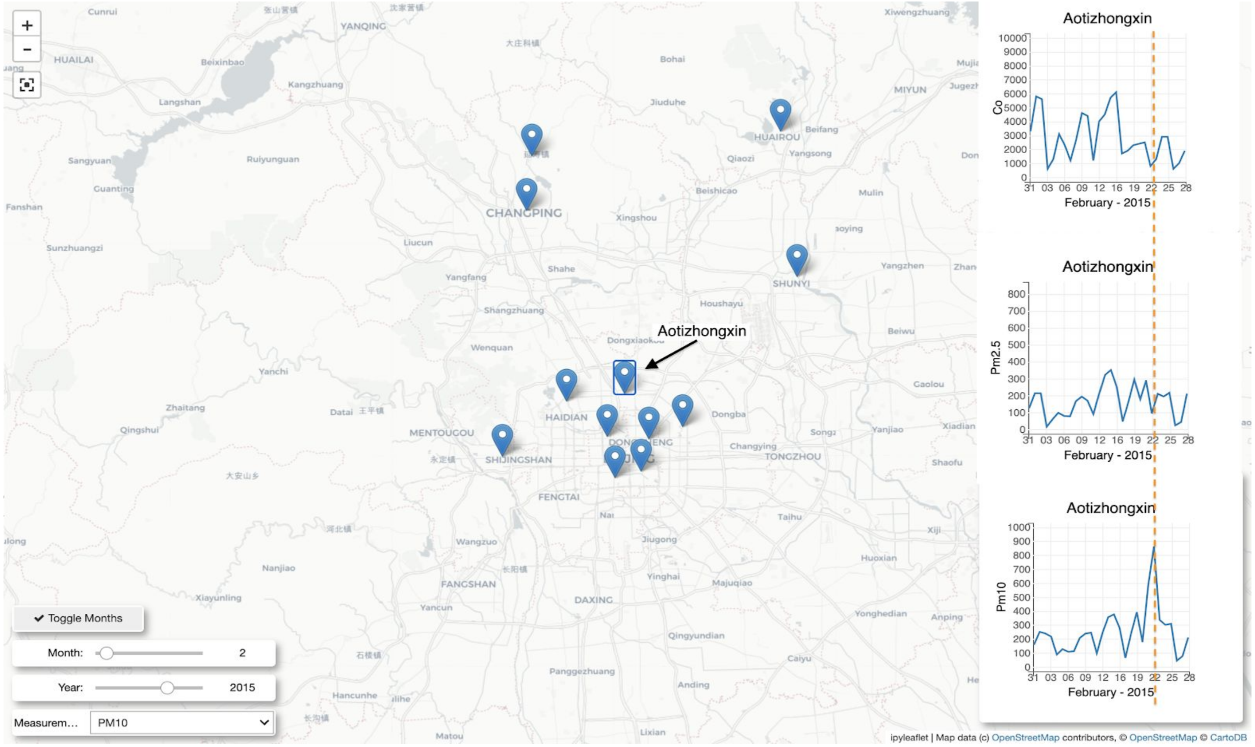


```
#Select just one station to examine
#df = df_all[df_all['station'].isin(['Aotizhongxin','Changping','Dingling'])]
df_A = df[df['station'].isin(['Aotizhongxin'])]
print('length of A : {}'.format(len(df_A.index)))
df_C = df[df['station'].isin(['Changping'])]
print('length of C : {}'.format(len(df_C.index)))
df_D = df[df['station'].isin(['Dingling'])]
print('length of D : {}'.format(len(df_D.index)))
```

Results and Conclusion:

Using the interactive map and graphs, we noticed that the AQI values for PM2.5 and PM10 appeared to peak around February 19, 2015 across all the weather stations. After doing some research, we realized that the Chinese New Year celebrations occurred on this exact date. CNY is one of the most important holidays in China and also the busiest time of year with increased travel, which could potentially explain the elevated AQI values for PM2.5 and PM10. Moreover, although China reported that it was successful in meeting its 2012 environmental action plan goals, the information derived from our map and graphs suggest otherwise. Overall, with our visual application, a user could ask a variety of different questions about the data, including “How has temperature changed over the four-year period?” and “How has the concentration of each pollutant changed from 2013 to 2017?”. Perhaps in the future, we could add more layers of information to the map, for example data on vehicular traffic and factories/industrial parks. We would also want to update the application with more recent air quality data to make it more relevant.

Figure 10: map and graphs (CO, PM2.5, and PM10 concentrations at Aotizhongxin in February 2015)



Toggle Months

Month: 2

Year: 2015

Measurem… PM10

References:

- Zhang, S., Guo, B., Dong, A., He, J., Xu, Z. and Chen, S.X. (2017) Cautionary Tales on Air-Quality Improvement in Beijing. Proceedings of the Royal Society A, Volume 473, No. 2205, Pages 20170457.
- https://archive.ics.uci.edu/ml/datasets/Beijing+Multi-Site+Air-Quality+Data
- https://www.who.int/airpollution/news-and-events/how-air-pollution-is-destroying-our-health
- https://www.loc.gov/law/help/air-pollution/china.php
- https://pypi.org/project/python-aqi/
- https://www.airnow.gov/aqi/aqi-calculator/
- https://www.epa.gov/sites/production/files/2014-05/documents/zell-aqi.pdf
- https://en.wikipedia.org/wiki/Chinese_New_Year
- https://chinadialogue.net/en/pollution/10711-china-releases-2-2-action-plan-for-air-pollution/