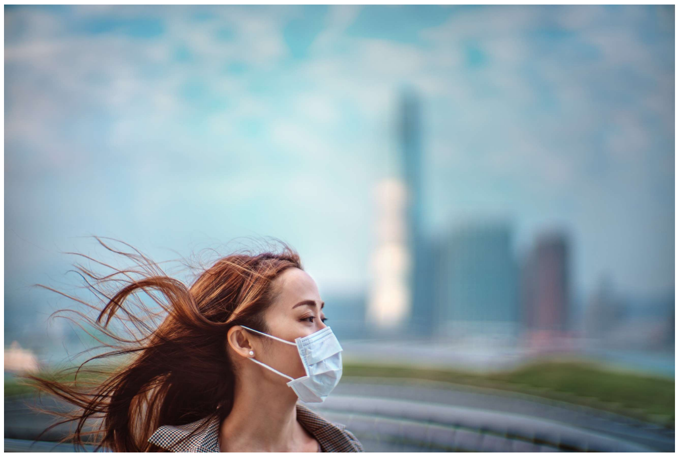
Northeastern University, Seattle, WA



PROJECT REPORT

Air Quality Pollutant Regression Analysis

IE 7300

STATISTICAL LEARNING FOR ENGINEERING

Fall 2022

Jessica Tanumihardja

Percentage of Effort Contributed by Student:_100%_

Signature of Student: ______________________

Submission Date:__12/17/22____________________

Project Overview

This project is an individual semester-long project for IE 7300 Statistical Learning for Engineering class. The project objective is to design, implement, evaluate, and validate machine learning models using python programming. The models are based on classification, regression, dimensionality reduction, and/or clustering algorithms depending on the chosen dataset. The business problem to be addressed is training the model and predicting the target variable values.

Problem Statement

The objective of this dataset is to perform statistical analysis on this dataset to classify the type of the dataset distribution, find a regression equation to predict the fit, and conclude which regression equation is the best fit for each parameter.

Project Dataset

To utilize the environmental engineering experience of the author, the environmental-related dataset was searched for this project. The chosen dataset is "The Beijing Air Quality Data" from the UCI Machine Learning Repository. The dataset can be found here: https://archive-beta.ics.uci.edu/ml/datasets/beijing+multi+site+air+quality+data.

The chosen dataset consists of 12 different CSV files corresponding to 12 station/region names where the air quality data is measured within Beijing, China. Each file consists of 18 columns as listed below. The total air quality data contains 420,768 rows and 14 parameters when the year, month, date, and hour columns are combined into one cell. This dataset fulfills the project requirement of at least 10,000 rows and 14 parameters.

Background

Air quality monitoring and assessment is a crucial part of public health and the environment. Seattleites might feel the worsening air quality in the past few years due to increasing wildfire and industrial activities. It also happens in China for a slightly different reason. China has been dealing with air pollution problems in the past decades due to large economic development and population growth (Ye, 2022). The main energy source for the country, including how the electricity is generated, is coal-burning activities. Coal produces a lot of by-products when burned, such as fine particulates and smog. Additional pollutants are also produced by the high use of motor vehicles in big cities like the nation's capital, Beijing.

The topography of the city also contributes to low air quality as wind direction greatly impacts the distribution of pollutants. Hence, meteorological data is an important subset to be included when analyzing the impact of pollutants in Beijing. The dataset includes several common air pollutants and meteorological data.

Dataset details

The dataset is used by Zhang, et.al. for the “Cautionary Tales on Air-Quality Improvement in Beijing” paper (Zhang, et al., 2017). The paper was verifying the claim that Beijing air quality data, measured by the annual fine particulate matter concentration, were improved by 9.9% in 2016. The researchers assembled hourly air pollutants data (6 main air pollutants and 6 relevant meteorological variables) from 12 air quality monitoring sites in Beijing, China. The period of the dataset is from March 1st, 2013, to February 28th, 2017. The air pollutants concentrations were supplied by Beijing Municipal Environmental Monitoring Center. The list of pollutants is:

- PM2.5 (fine particulate matter that is less than 2.5 microns in width)
- PM10 (particulate matter, inhalable, that is less than 10 microns in width)
- SO_2 (sulfur dioxide)
- NO_2 (nitrogen dioxide)
- CO (carbon monoxide)
- O_3 (ozone)

All the concentrations are measured in $\mu g/m^3$ (or parts per million, ppm).

The meteorological variables were sourced from China Meteorological Administration from the closest weather station to the air quality monitoring sites. The variables (and their unit) listed are:

- temperature (Celsius)
- pressure (hPa)
- dew point temperature (Celsius)
- precipitation (mm)
- wind direction (16 different directions)
- wind speed (m/s)

Data Analysis and Preprocessing Approach

Python was used to achieve the objective of this project. Data cleaning was done first since there are missing values (indicated by NA) in the dataset. After the clean-up and no duplicate was confirmed, the central tendency was calculated to see the features of the data. Since only NumPy, Pandas, Keras, pytorch, and Tensorflow libraries can be used, custom functions were defined and coded for each model.

Dataset definition or data dictionary

The dataset consists of 12 files representing each air-quality station data. The file names are given below:

- 'PRSA_Data_Aotizhongxin_20130301-20170228.csv',
- 'PRSA_Data_Changping_20130301-20170228.csv',
- 'PRSA_Data_Dingling_20130301-20170228.csv',

- 'PRSA_Data_Dongsi_20130301-20170228.csv',
- 'PRSA_Data_Guanyuan_20130301-20170228.csv',
- 'PRSA_Data_Gucheng_20130301-20170228.csv',
- 'PRSA_Data_Huairou_20130301-20170228.csv',
- 'PRSA_Data_Nongzhanguan_20130301-20170228.csv',
- 'PRSA_Data_Shunyi_20130301-20170228.csv',
- 'PRSA_Data_Tiantan_20130301-20170228.csv',
- 'PRSA_Data_Wanliu_20130301-20170228.csv',
- 'PRSA_Data_Wanshouxigong_20130301-20170228.csv'

Each file contains a table of the listed 18 data columns as elaborated below.

Index #	Column Name	Variable	Units	Data Type	Additional Description	Allowed/ Included values	Accepts null value?
0	No	Index	-	int64	-	no limit	No
1	year	Year	-	int64	-	integers in range [2013, 2017]	No
2	month	The month of the year	-	int64	-	integers in range [0, 12]	No
3	day	Day of the month	-	int64	-	integers in range [0, 31]	No
4	hour	Hour of the day	-	int64	-	integers in range [0, 24]	No
5	PM2.5	Particulate Matter 2.5	$\mu g/m^3$	float64	measurement of fine particulate matter in the air that is less than 2.5 microns in width (primary pollutant)	no limit (typically < 1000)	Yes
6	PM10	Particulate Matter 10	$\mu g/m^3$	float64	measurement of particulate matter, inhalable, that is less than 10 microns in width (primary pollutant)	no limit (typically < 1000)	Yes
7	SO2	Sulfur Dioxide	$\mu g/m^3$	float64	Generated by most hydrocarbon and suspended particles (primary pollutant)	no limit (typically < 400)	Yes
8	NO2	Nitrogen Oxide	$\mu g/m^3$	float64	Generated by the reaction of primary pollutant with	no limit (typically < 300)	Yes

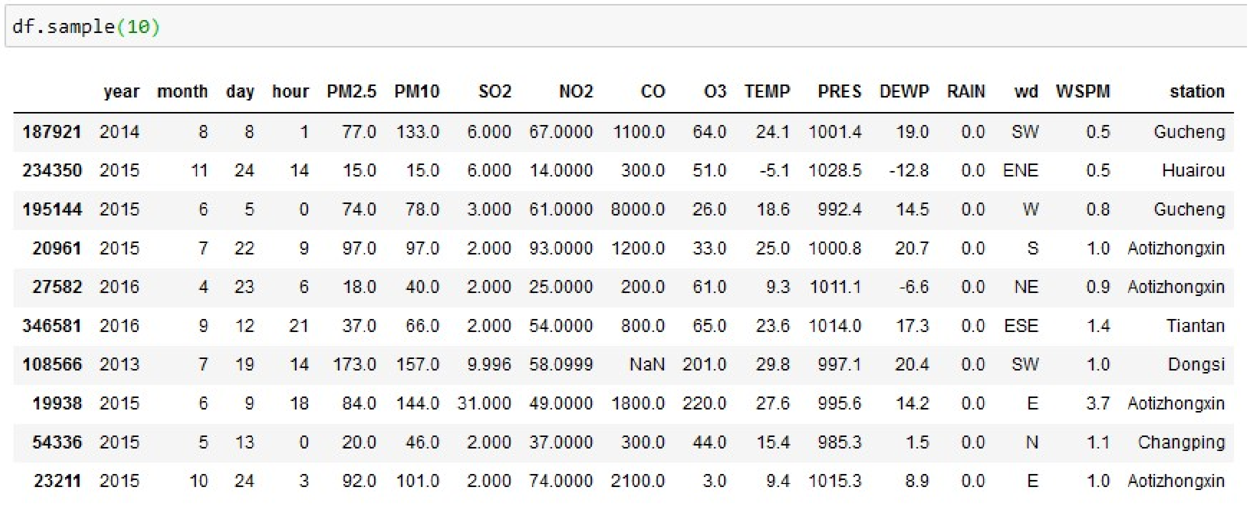
					ambient air (secondary pollutant)		
9	CO	Carbon Monoxide	µg/m^3	float64	Mainly generated from combustion of hydrocarbon (primary pollutant)	no limit (typically < 10000)	Yes
10	O3	Ozone	µg/m^3	float64	Also mainly known as photochemical smog	no limit (typically < 500)	Yes
11	TEMP	Temperature	^{o}C	float64	Ambient air	no limit	Yes
12	PRES	Pressure	hPa	float64	Station pressure	no limit	Yes
13	DEWP	Dew Point	^{o}C	float64	Ambient air	no limit	Yes
14	RAIN	Precipitation	mm	float64	At the station	no limit	Yes
15	wd	Wind Direction	-	object	Cardinal, Ordinal, and secondary intercardinal directions of the wind speed	16 directions: N, NNE, NE, ENE, E, ESE, SE, SSE, S, SSW, SW, WSW, W, NWN, NW, and NNW	Yes
16	WSPM	Wind Speed	m/s	float64	At the station	no limit	Yes
17	station	Station (City) Name	-	object	Station identifier	-	No

Exploratory Data Analysis (EDA)

Exploratory Data Analysis (EDA) for the dataset is conducted in Python. The pdf file of the python code is attached in Appendix. The EDA is mostly conducted using Pandas' function (pd) except for the plots. The EDA includes compiling all CSV files, checking data info, data types, duplicate values, unique values, missing values, descriptive statistics, visualization, and correlations.

Data compilation

As previously mentioned, the original dataset is separated into 12 CSV according to the city where the data was measured. Since the city is one of the independent variables, all 12 files are combined into one file using the "OS" and "Glob" libraries. This resulted in 420768 rows and 17 columns data frame (shape). The column, "No", is the same as the Python index, and values are repeated when 12 CSV is compiled. This column is dropped from the data frame. Values are checked with df.sample(10) as shown below.



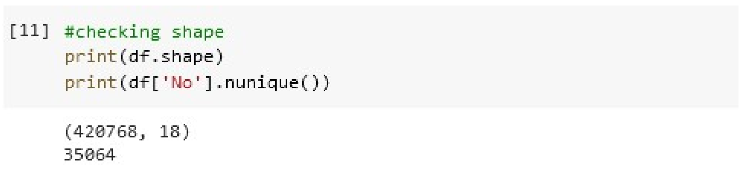
```
df.sample(10)
```

	year	month	day	hour	PM2.5	PM10	SO2	NO2	CO	O3	TEMP	PRES	DEWP	RAIN	wd	WSPM	station
187921	2014	8	8	1	77.0	133.0	6.000	67.0000	1100.0	64.0	24.1	1001.4	19.0	0.0	SW	0.5	Gucheng
234350	2015	11	24	14	15.0	15.0	6.000	14.0000	300.0	51.0	-5.1	1028.5	-12.8	0.0	ENE	0.5	Huairou
195144	2015	6	5	0	74.0	78.0	3.000	61.0000	8000.0	26.0	18.6	992.4	14.5	0.0	W	0.8	Gucheng
20961	2015	7	22	9	97.0	97.0	2.000	93.0000	1200.0	33.0	25.0	1000.8	20.7	0.0	S	1.0	Aotizhongxin
27582	2016	4	23	6	18.0	40.0	2.000	25.0000	200.0	61.0	9.3	1011.1	-6.6	0.0	NE	0.9	Aotizhongxin
346581	2016	9	12	21	37.0	66.0	2.000	54.0000	800.0	65.0	23.6	1014.0	17.3	0.0	ESE	1.4	Tiantan
108566	2013	7	19	14	173.0	157.0	9.996	58.0999	NaN	201.0	29.8	997.1	20.4	0.0	SW	1.0	Dongsi
19938	2015	6	9	18	84.0	144.0	31.000	49.0000	1800.0	220.0	27.6	995.6	14.2	0.0	E	3.7	Aotizhongxin
54336	2015	5	13	0	20.0	46.0	2.000	37.0000	300.0	44.0	15.4	985.3	1.5	0.0	N	1.1	Changping
23211	2015	10	24	3	92.0	101.0	2.000	74.0000	2100.0	3.0	9.4	1015.3	8.9	0.0	E	1.0	Aotizhongxin

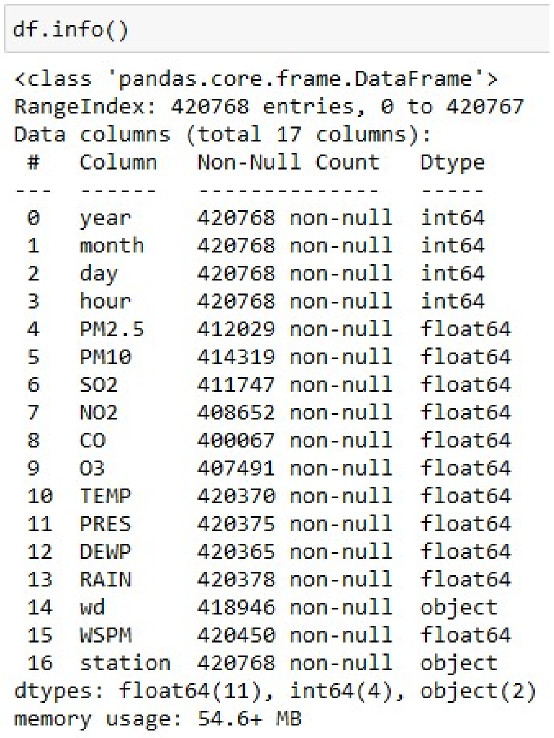
The combined data frame is saved into a new CSV file for future use.

Data information

The null count and data types can be checked using pd.info() function. The result is shown on the next page. There are two columns with "object" datatype; these were converted into categorical numeric values prior to regression analysis using pd.replace() (similar to homework problems).



```
[11] #checking shape
     print(df.shape)
     print(df['No'].nunique())

     (420768, 18)
     35064
```

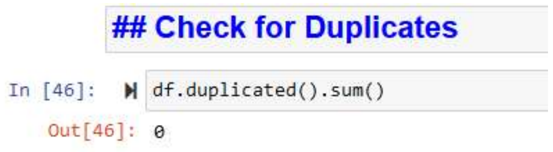


```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 420768 entries, 0 to 420767
Data columns (total 17 columns):
 #   Column   Non-Null Count   Dtype  
---  ------   --------------   -----  
 0   year     420768 non-null  int64  
 1   month    420768 non-null  int64  
 2   day      420768 non-null  int64  
 3   hour     420768 non-null  int64  
 4   PM2.5    412029 non-null  float64
 5   PM10     414319 non-null  float64
 6   SO2      411747 non-null  float64
 7   NO2      408652 non-null  float64
 8   CO       400067 non-null  float64
 9   O3       407491 non-null  float64
 10  TEMP     420370 non-null  float64
 11  PRES     420375 non-null  float64
 12  DEWP     420365 non-null  float64
 13  RAIN     420378 non-null  float64
 14  wd       418946 non-null  object 
 15  WSPM     420450 non-null  float64
 16  station  420768 non-null  object 
dtypes: float64(11), int64(4), object(2)
memory usage: 54.6+ MB
```

Duplicate values

There are no duplicate values in the data frame.

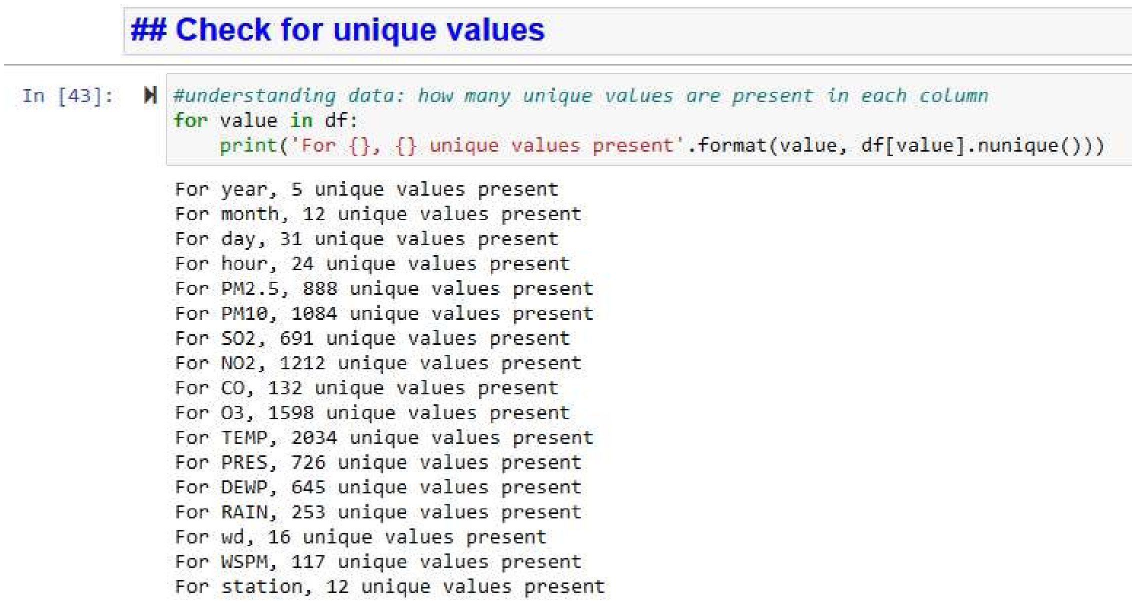


```
## Check for Duplicates

In [46]:  df.duplicated().sum()

Out[46]: 0
```

Unique values

To understand data further, unique values in each column are obtained. The values are important to check especially for time columns and the column with object datatype.

The data is for the year 2013 -2017 so it only has 5 unique year values. As described above, the wind direction ("wd") column has 16 unique values. It is using the Cardinal, Ordinal, and secondary intercardinal directions of the wind speed. There are 12 stations measured.



Check for unique values

```
In [43]:  #understanding data: how many unique values are present in each column
          for value in df:
              print('For {}, {} unique values present'.format(value, df[value].nunique()))
```

```
For year, 5 unique values present
For month, 12 unique values present
For day, 31 unique values present
For hour, 24 unique values present
For PM2.5, 888 unique values present
For PM10, 1084 unique values present
For SO2, 691 unique values present
For NO2, 1212 unique values present
For CO, 132 unique values present
For O3, 1598 unique values present
For TEMP, 2034 unique values present
For PRES, 726 unique values present
For DEWP, 645 unique values present
For RAIN, 253 unique values present
For wd, 16 unique values present
For WSPM, 117 unique values present
For station, 12 unique values present
```

Missing values

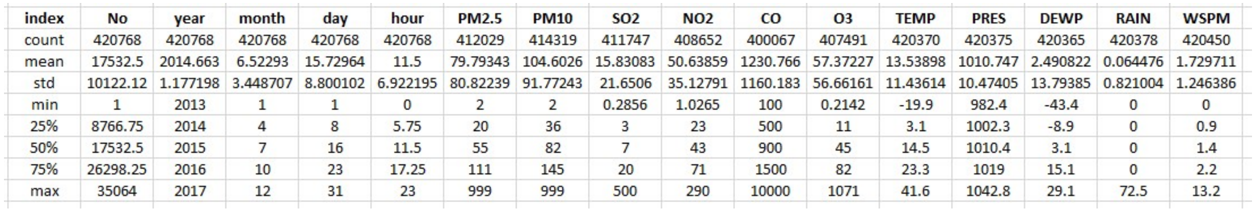
As shown in the data types section above, columns ['PM2.5', 'PM10', 'SO2', 'NO2', 'CO', 'O3', 'wd', 'WSPM'] have missing (NULL) values. Those column statistics are checked to identify if there is a large variation in the dataset and if any outlier is present. There are two options to handle missing values:

- Option 1: Since the amount of the NULL is not that many, remove the NULL values instead of estimating the data to provide accurate analysis. The central limit theorem should hold since we saw that the data was mainly normally distributed (skewed).
- Option 2: Filling the value (pd.fillna()) with the median of the data, forward-fill (ffill), backward-fill (bfill), or interpolate (pd.interpolate) can be done if we don't want to remove NULL values.

It is decided that the missing values are removed before further processing the data.

Descriptive statistics

Preliminary analysis in python produces these dataset statistics using df.describe() function:



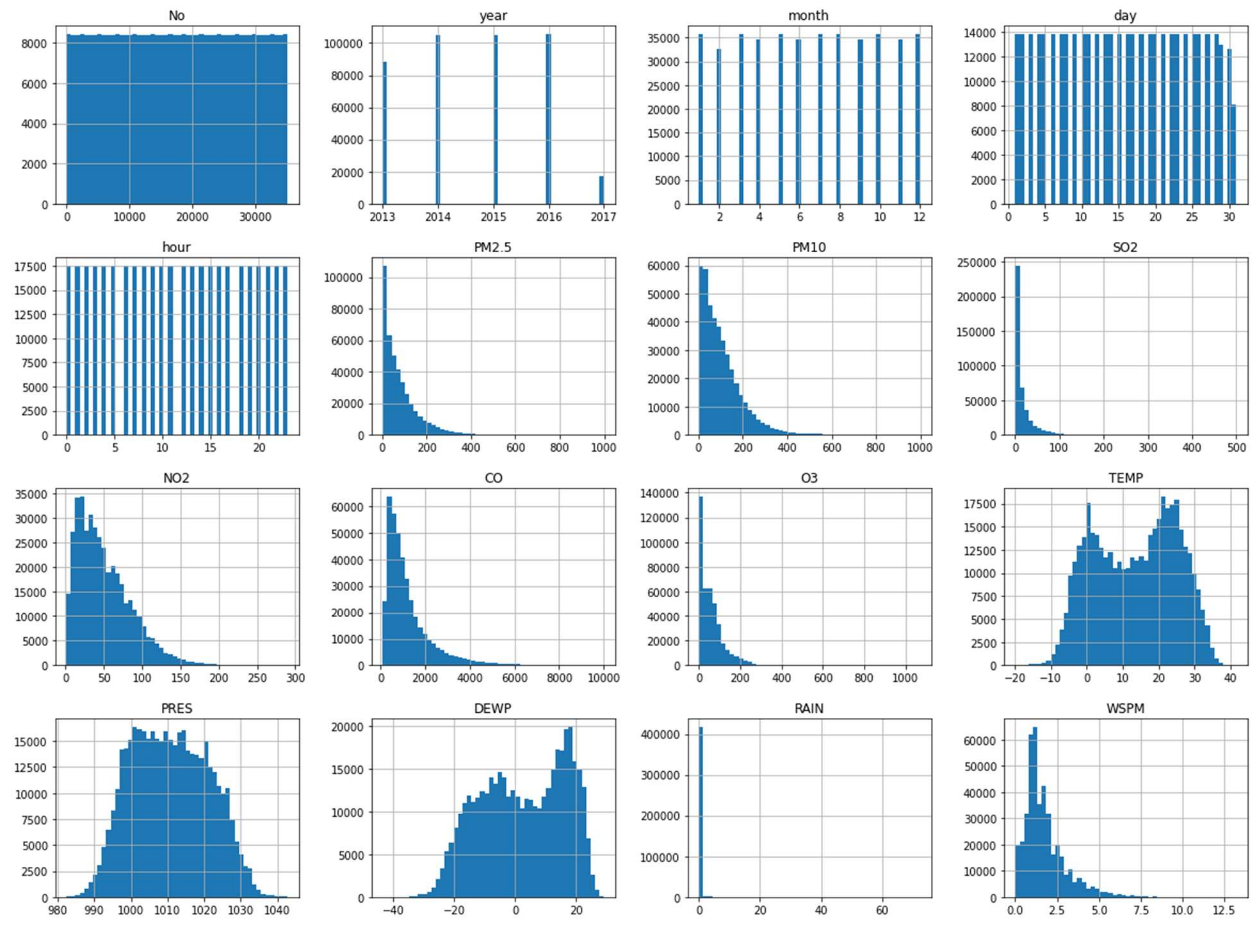
index	No	year	month	day	hour	PM2.5	PM10	SO2	NO2	CO	O3	TEMP	PRES	DEWP	RAIN	WSPM
count	420768	420768	420768	420768	420768	412029	414319	411747	408652	400067	407491	420370	420375	420365	420378	420450
mean	17532.5	2014.663	6.52293	15.72964	11.5	79.79343	104.6026	15.83083	50.63859	1230.766	57.37227	13.53898	1010.747	2.490822	0.064476	1.729711
std	10122.12	1.177198	3.448707	8.800102	6.922195	80.82239	91.77243	21.6506	35.12791	1160.183	56.66161	11.43614	10.47405	13.79385	0.821004	1.246386
min	1	2013	1	1	0	2	2	0.2856	1.0265	100	0.2142	-19.9	982.4	-43.4	0	0
25%	8766.75	2014	4	8	5.75	20	36	3	23	500	11	3.1	1002.3	-8.9	0	0.9
50%	17532.5	2015	7	16	11.5	55	82	7	43	900	45	14.5	1010.4	3.1	0	1.4
75%	26298.25	2016	10	23	17.25	111	145	20	71	1500	82	23.3	1019	15.1	0	2.2
max	35064	2017	12	31	23	999	999	500	290	10000	1071	41.6	1042.8	29.1	72.5	13.2

From the descriptive statistics data, the standard deviations are quite high for some variables, especially for variables with missing values listed above. The largest standard maximum value can be found in the "CO" (carbon monoxide) column.

Since most of these columns are measured using some type of probe, the minimum and maximum value are likely to be the low and high limit of the probe, which has high possibility to be outliers. For example, if the probe is plugged and not cleaned/addressed within an hour, it will read the maximum or minimum value of the probe and it is stored in the dataset (or NULL values, depending on the probe setting). This is clearly an outlier. Data visualization will help to analyze this further.

Data Visualization

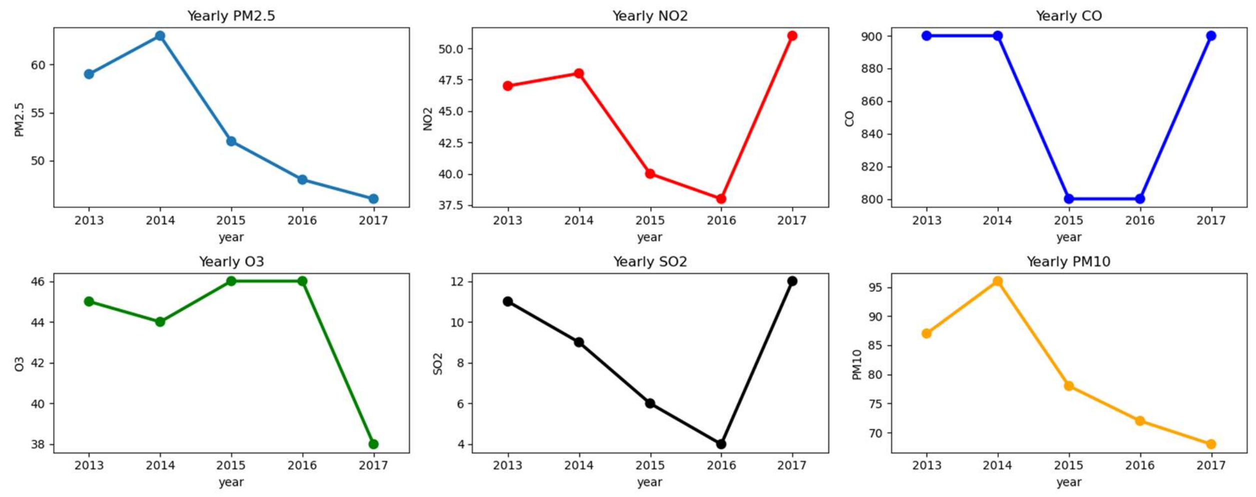
The histogram of each numeric variable is shown below.



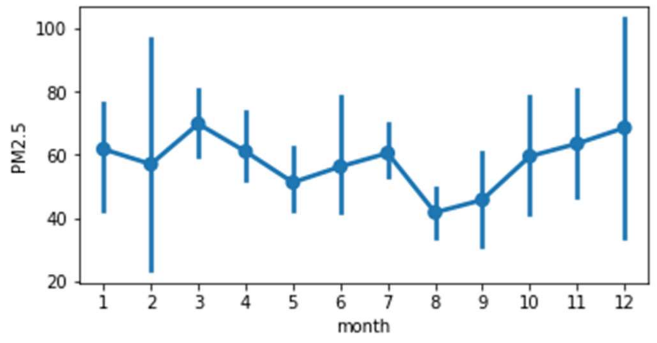
From the histogram, it can be concluded that:

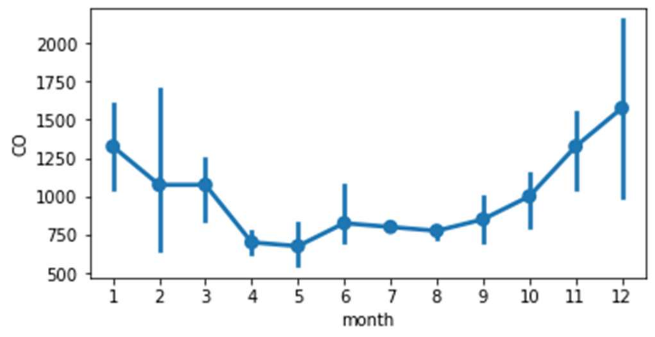
- The 6 atmospheric variables follow an exponential reduction pattern
- Temperature, Pressure, and Dewpoint follow a normal distribution
- WSPM (wind speed (m/s)) follows a skewed normal distribution.
- Rain (rainfall intensity) distribution is unclear.
- It is best to standardize the data prior to regression analysis as they all have a different scale.

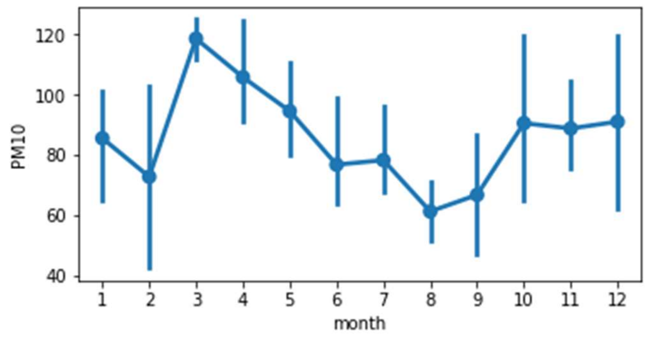
The dataset is grouped per year to obtain the annual trend. The plot shows that the annual PM2.5, PM10, and O3 are trending downward. However, the annual CO, SO2, and NO2 are trending up. Data for other pollutants in 2017 needs to be analyzed as it is exceptionally high than other years opposing the trend.

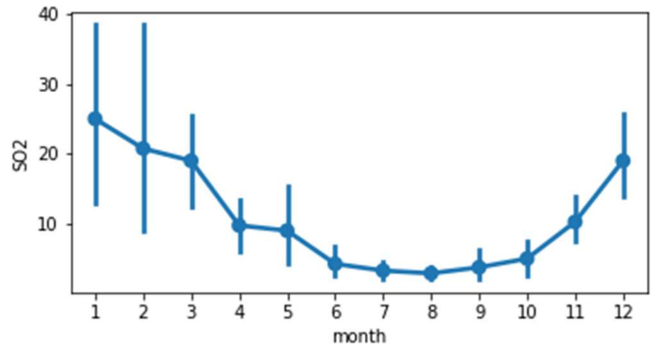


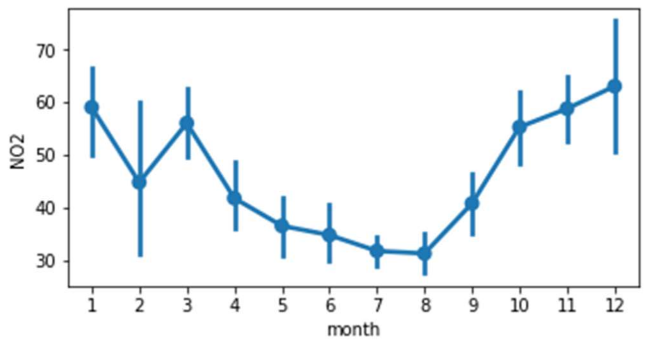
The monthly analysis per pollutant is shown below. The marker indicates the median value (ppm) over 2013-2017. The error bar indicates the minimum and maximum range of the respective values.

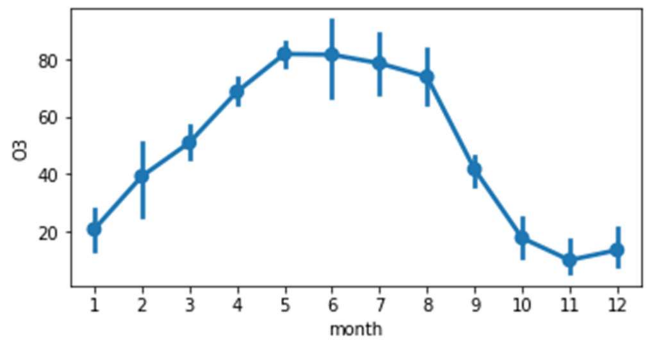






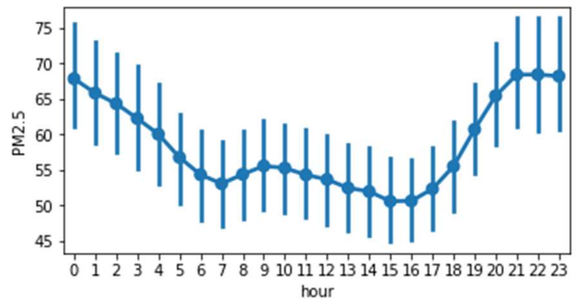


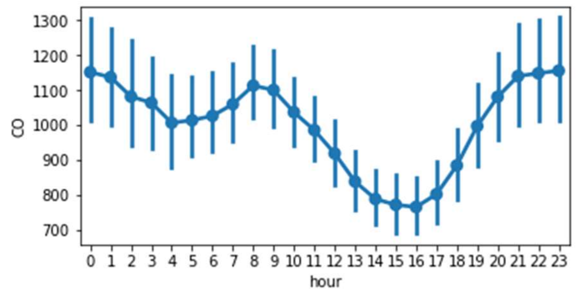


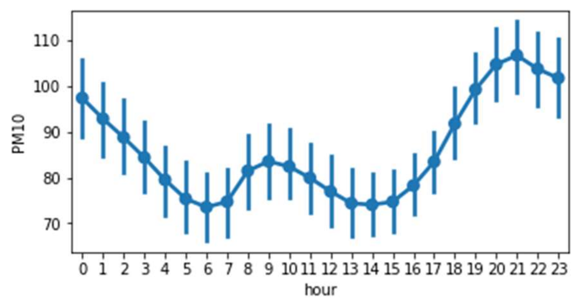


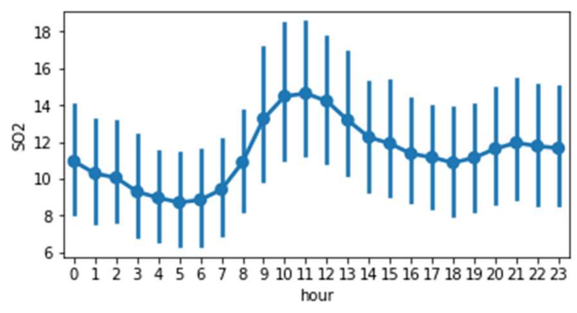
From the monthly analysis above, PM2.5 are stable throughout the different season. The NO2, CO, and SO2 median concentrations are lower during warmer months (May (5) to September (9)). O3 has the opposite trend with other pollutants with higher values during warmer months. PM10 has sharper peaks and valleys compared to PM2.5 even though the yearly analysis is almost identical.

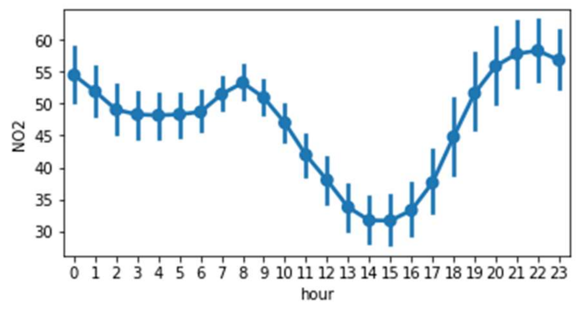
The hourly analysis per pollutant (median concentration, ppm) is shown below. The diurnal variation can be observed for all pollutants. The trend is like the monthly analysis above. The highest PM2.5, PM10, NO2, and CO concentrations are during night hours. The highest SO2 value is during midday while the highest O3 concentration is mid-afternoon.

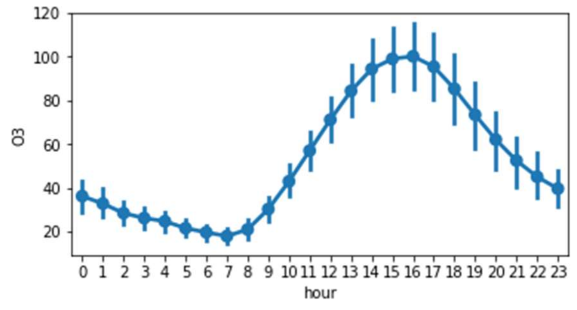








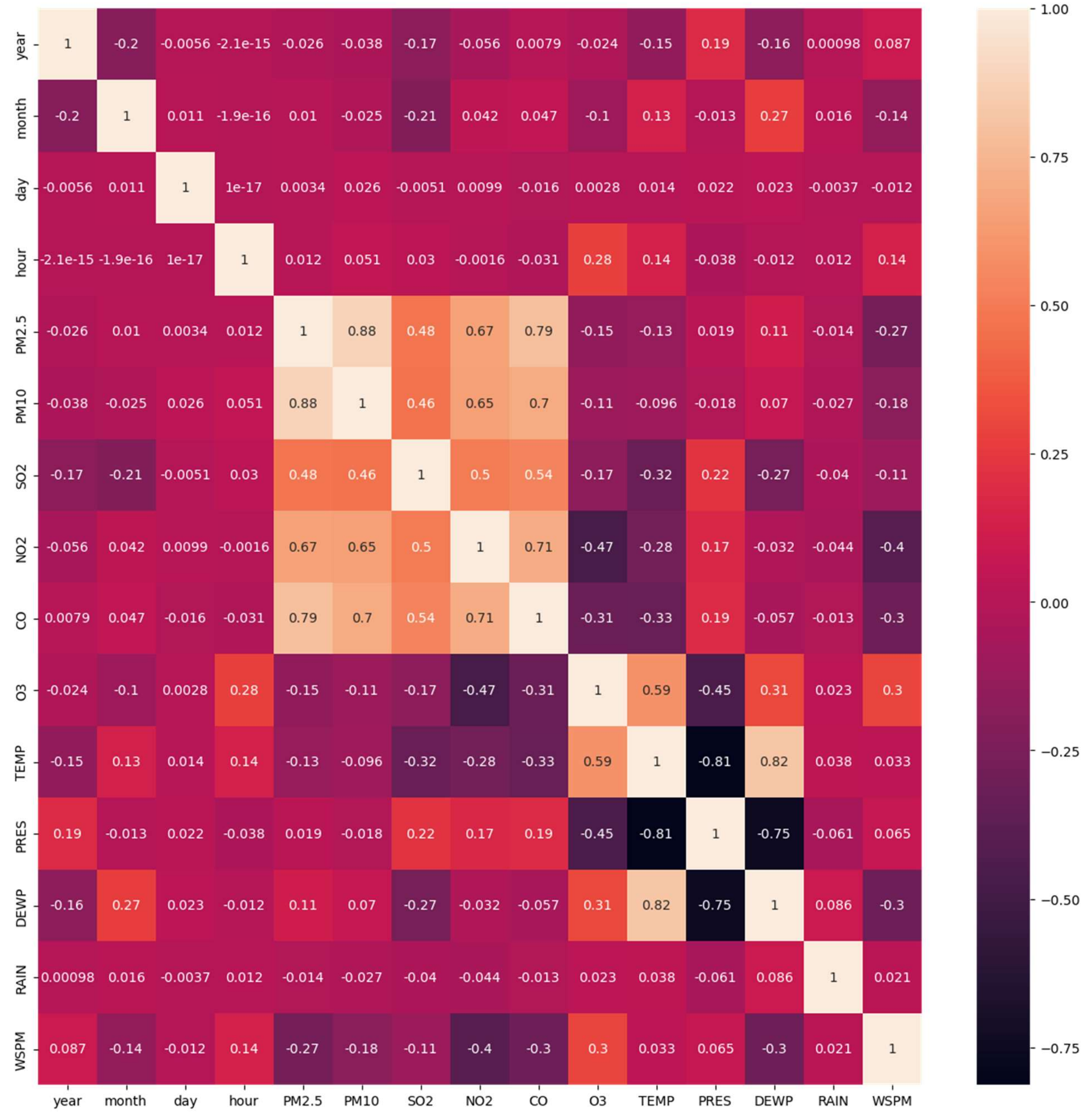




The correlation heatmap is produced to find the highest correlation between variables as shown on the next page. The summary of the observed correlation is:

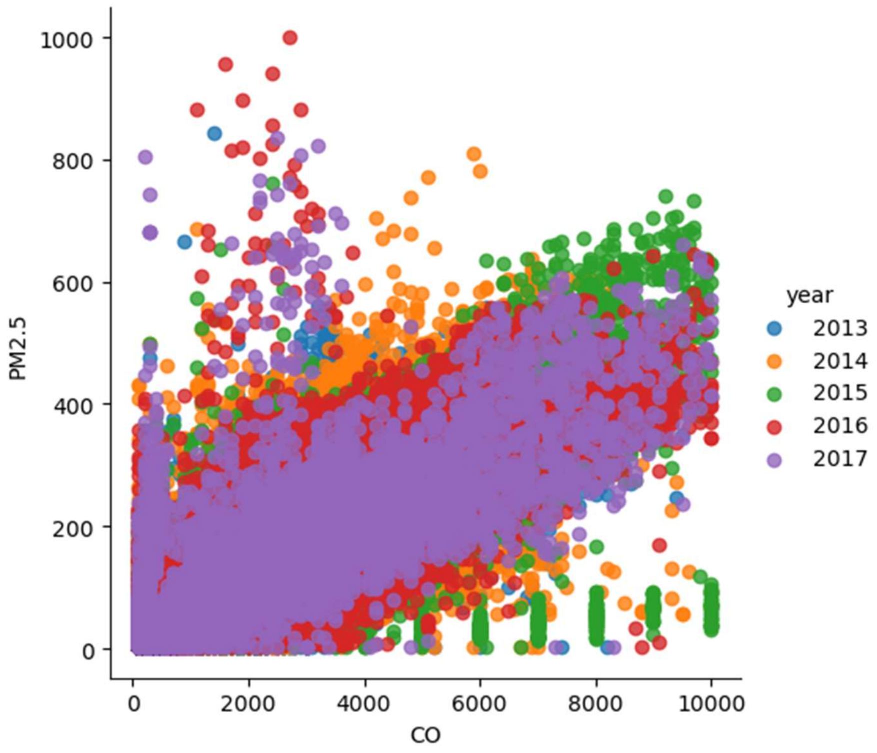
- There is strong correlation between PM2.5 and PM10 as PM2.5 is part of PM10. PM10 variable might need to be removed before regression analysis to truly predict PM2.5 only from other independent variables.

- There is a moderate positive correlation between PM2.5 and CO as well as NO. There is a weak positive correlation between PM2.5 and SO2. There is a very low correlation between PM2.5 and atmospheric data.
- There is also a moderate positive correlation between NO and CO. These two values might be repetitive in predicting PM2.5 values.
- There is a strong negative correlation between air temperature and pressure (as expected), and a strong positive correlation between temperature and dew point (as expected).



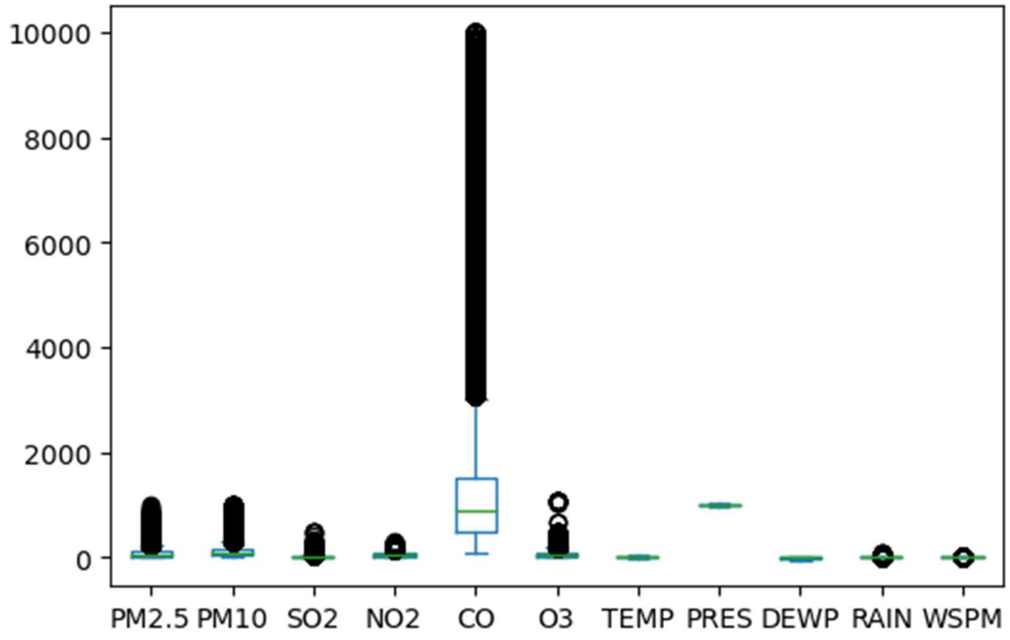
	year	month	day	hour	PM2.5	PM10	SO2	NO2	CO	O3	TEMP	PRES	DEWP	RAIN	WSPM
year	1	-0.2	-0.0056	-2.1e-15	-0.026	-0.038	-0.17	-0.056	0.0079	-0.024	-0.15	0.19	-0.16	0.00098	0.087
month	-0.2	1	0.011	-1.9e-16	0.01	-0.025	-0.21	0.042	0.047	-0.1	0.13	-0.013	0.27	0.016	-0.14
day	-0.0056	0.011	1	1e-17	0.0034	0.026	-0.0051	0.0099	-0.016	0.0028	0.014	0.022	0.023	-0.0037	-0.012
hour	-2.1e-15	-1.9e-16	1e-17	1	0.012	0.051	0.03	-0.0016	-0.031	0.28	0.14	-0.038	-0.012	0.012	0.14
PM2.5	-0.026	0.01	0.0034	0.012	1	0.88	0.48	0.67	0.79	-0.15	-0.13	0.019	0.11	-0.014	-0.27
PM10	-0.038	-0.025	0.026	0.051	0.88	1	0.46	0.65	0.7	-0.11	-0.096	-0.018	0.07	-0.027	-0.18
SO2	-0.17	-0.21	-0.0051	0.03	0.48	0.46	1	0.5	0.54	-0.17	-0.32	0.22	-0.27	-0.04	-0.11
NO2	-0.056	0.042	0.0099	-0.0016	0.67	0.65	0.5	1	0.71	-0.47	-0.28	0.17	-0.032	-0.044	-0.4
CO	0.0079	0.047	-0.016	-0.031	0.79	0.7	0.54	0.71	1	-0.31	-0.33	0.19	-0.057	-0.013	-0.3
O3	-0.024	-0.1	0.0028	0.28	-0.15	-0.11	-0.17	-0.47	-0.31	1	0.59	-0.45	0.31	0.023	0.3
TEMP	-0.15	0.13	0.014	0.14	-0.13	-0.096	-0.32	-0.28	-0.33	0.59	1	-0.81	0.82	0.038	0.033
PRES	0.19	-0.013	0.022	-0.038	0.019	-0.018	0.22	0.17	0.19	-0.45	-0.81	1	-0.75	-0.061	0.065
DEWP	-0.16	0.27	0.023	-0.012	0.11	0.07	-0.27	-0.032	-0.057	0.31	0.82	-0.75	1	0.086	-0.3
RAIN	0.00098	0.016	-0.0037	0.012	-0.014	-0.027	-0.04	-0.044	-0.013	0.023	0.038	-0.061	0.086	1	0.021
WSPM	0.087	-0.14	-0.012	0.14	-0.27	-0.18	-0.11	-0.4	-0.3	0.3	0.033	0.065	-0.3	0.021	1

Preliminary plotting of PM2.5 with the highest correlated variable CO is also provided below.

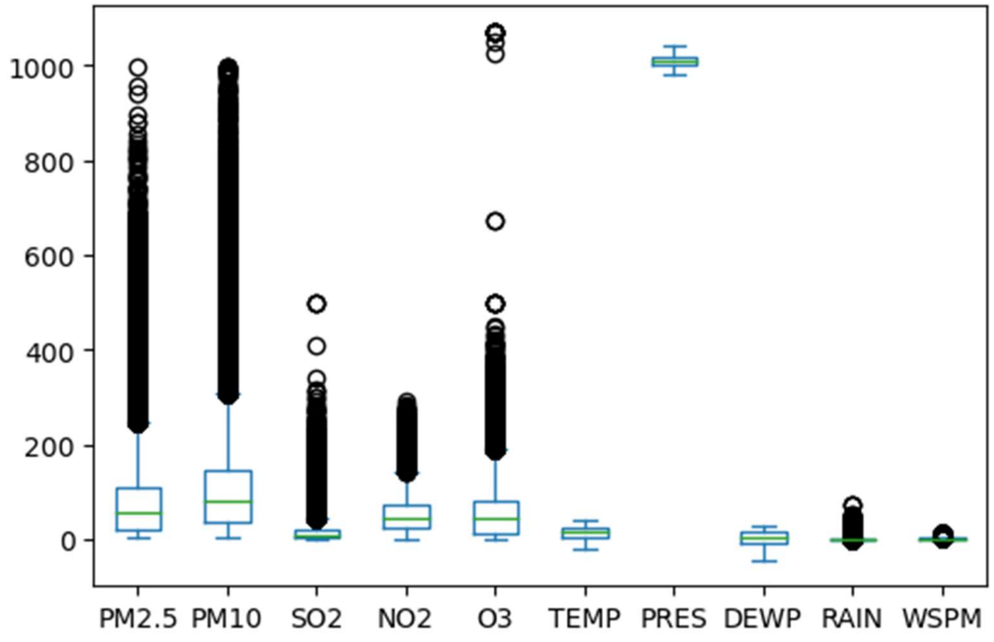


Outlier Analysis

For outlier analysis, the boxplot of each numerical variable (excluding time) is constructed. All of the pollutant variables have data outside of the top line of the boxplot, i.e. outliers are present here.



Reobtain the graph without the ‘CO’ column to see the other variables clearly:



Since there are quite a bit of outlier present in the pollutant concentration variables, outlier removal might need to be done before the regression analysis model. Performance metrics will determine if the removal is necessary.

Data Transformation

As described above, we have more sample numbers than required so the null values were removed instead of imputing values to avoid skew. Another alternative is (not used) filling missing data (fillna) according to the month's mean/median for numerical type, according to station location for object data (wind direction), etc.

According to the data visualization section above, all the pollutant variables have some degree of outliers, especially CO (carbon monoxide). The outlier is not removed in this model at first to get the original data performance metrics. If needed, outlier removal might be performed (drop samples in quantiles 10 and 90 of each feature).

There are two object-type variables in this dataset, which are the wind direction (wd) and station (city) features. Those categories were encoded to integers using pd.replace function. Encoding using pd.dummies is also possible but it will increase the number of columns significantly, hence not been chosen for this dataset.

The new label for wind direction is: {'NNW':0, 'N':1, 'NW':2, 'NNE':3, 'ENE':4, 'E':5, 'NE':6, 'W':7, 'SSW':8, 'WSW':9, 'SE':10, 'WNW':11, 'SSE':12, 'ESE':13, 'S':14, 'SW':15}.
The new label for station names is: {'Aotizhongxin':0, 'Changping':1, 'Dingling':2, 'Dongsi':3, 'Guanyuan':4, 'Gucheng':5, 'Huairou':6, 'Nongzhanguan':7, 'Shunyi':8, 'Tiantan':9, 'Wanliu':10, 'Wanshouxigong':11}.

For a more complex model, a subset of the data frame containing 10% of the total sample was used to compare results since central tendency holds high computational time for a decision tree-based model.

Algorithm and Methodology

Algorithm and Models

The chosen dataset contains a continuous value of the air pollutants, ranging from 3 to 999 ppm. Hence, a regression analysis is conducted for this dataset to find the relationship between the independent variables (features) and dependent (target) variables. The target variable is the main pollutant in question for the paper, PM2.5 pollutant. Since this dataset has labels, supervised machine-learning algorithms were tested. The other pollutants, time, and meteorological variables are the independent or explanatory variables. The importance of those independent variables in predicting the target variable can be obtained from each model (feature importance). That way, the researchers can reduce the time and cost of gathering data by only selecting what features matter the most.

Throughout the whole semester, we learned to make custom classification and regression models. The general overview of the supervised machine learning models is shown in Figure 1 below.

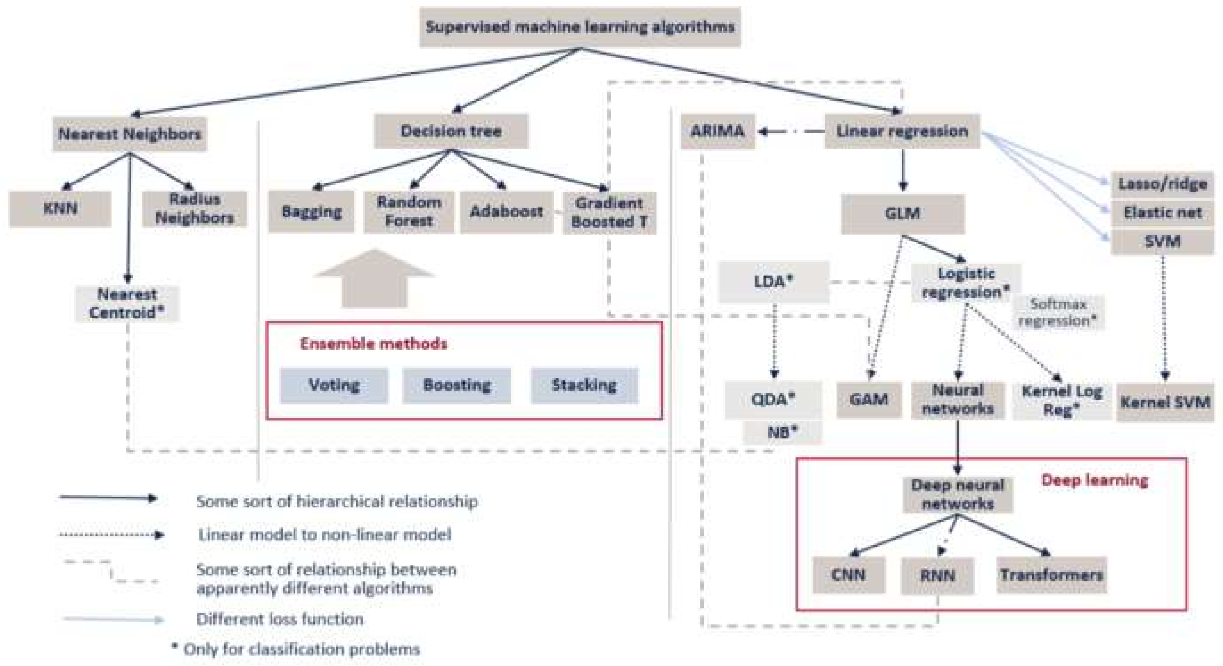


Figure 1. Types of Supervised Machine Learning Models (Shi, 2022)

Some of those models are chosen to be applied to this dataset. A total of 7 custom regression models are tested to predict the PM2.5 for this dataset, ranked from simplest to most complicated, are:

- Linear Regression
- Lasso Regression (L1 regularization)
- Decision Tree
- Random Forest

- Gradient Boosting
- K-Nearest Neighbor (KNN)
- Neural network

As shown in Figure 1, those models can be categorized into 3 general algorithms: linear regression, decision tree, and nearest neighbor algorithm.

Linear regression algorithm utilizes mathematical functions to the dataset as well as minimizes the cost (or loss or objective) function. It works by fitting a linear line (called the best-fit line) into the given data such that the error or bias is minimized. The general formula is y = **w x** + **b**, where **w** = weights (slope) of each independent variable and b = intercepts. The project dataset is univariate (the target variable is only PM2.5), so we are finding the best fit line to best represent the relationship between independent (x-axis) and dependent (y-axis) variables. Gradient descent is used to find the minimum cost function. The advantage of linear regression is it is simple to apply and has a minimal computational cost.

Lasso (least absolute shrinkage and selection operator) regression is a special application of linear regression where a penalty (lambda) is introduced to the model's slope to reduce variance. The advantage over another regularization (ridge regression) is that it can be used to do the feature selection. Lasso regression penalized the weight (slope) of each feature when it has less importance in predicting the target variable. The lower the importance of a feature is, the higher the penalty is. Hence, the lower its weight will be. It is also called L1 regularization as it introduced an L1 penalty in the model. The lambda value can be between 0 to infinity. Lasso regression is especially useful for high collinearity between the independent variable since it can filter the unimportant features.

A decision tree is a rule-based algorithm to group values into different categories. Similar categories are lumped together. Just like the name, the data is split according to chosen feature values (node) for each layer of the tree, up to a predetermined number of layers (tree depth) and restriction (minimum number of samples per split). The model is optimized by maximizing information gain for each layer. For regression application, it is calculating variance reduction. A decision tree model has various advantages such as insensitivity to missing values, insensitivity to errors in the training data (low bias), and flexible applications (linear or non-linear, regression or classification). This model is also easier to understand due to the ability to visualize and get feature importance (Shobha & Rangaswamy, 2018). However, this model is more complex than linear regression and requires high computational costs as the number of variables increases.

Two ensemble methods based on the decision tree model were tested in this dataset: random forest (bagging) and gradient boost (boosting). Both models utilized and combined multiple models and get the best one (aggregate). Random forest runs a predetermined number of models simultaneously (parallel) while the gradient boost creates a small decision tree model, learns from it, and creates another one (sequential). Both models are expected to improve the decision tree model by reducing overfitting and hence, creating a more robust model. The disadvantage is the same as a decision tree with higher computational cost as the number of trees is made.

K-Nearest Neighbor (KNN) is the third type of machine learning algorithm that calculates the distance of two data points so that similar ones can be grouped together. The hyperparameters are the k value. The disadvantage of KNN is sensitivity to outliers and data imbalance. KNN also requires high computational cost because it needs to create a database of all the data point's neighbors (Shi, 2022).

Neural Network algorithm attempts to imitate human brain function by processing data and relationship with nodes (neurons). It is a specialized and optimized linear regression model (with weights, bias, and cost functions) with the addition of activation functions to add the non-linearity (Baheti, 2022). Therefore, the neural network can be used to explain broader use cases than other machine learning models. For the scope of this project, only a simple neural network (one hidden layer) is used for the dataset.

Another notable model to mention is the Support Vector Machine (SVM), which also belongs to the linear regression category. It is insensitive to outliers and can be optimized for each dataset by determining a suitable kernel. This model is not tested in this project but can be tested for future accuracy improvement.

Methodology

As mentioned in the previous section, most of the models that can be used for regression are tested in this project. Linear and Lasso regressions were trained and tested first for all the samples and features to get the base results. One can also select a few top important features from lasso regression for the following more complex models, but a different approach was used in this project. To reduce the computational cost, only 10% of random samples were taken from the original >300,000 samples. The details will be discussed in the model outcome section.

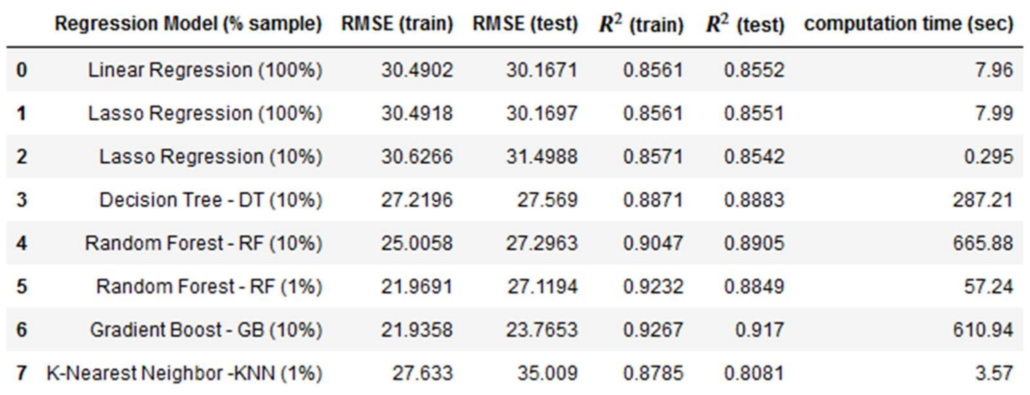
Decision tree based models (decision tree, random forest, and gradient boost) were utilized in this project to determine whether the rule-based models fit with the dataset. KNN was also tested but it requires a higher computational cost so only 1% of the total samples were used for this. Lastly, the result from Homework 10 Neural Network is also included here.

To avoid overfitting, data was split into a train (80%) and test (20%) dataset. The models were trained using 80% of the available data from all stations (randomized). Model fit parameters will also be calculated (R squared, RMSE, etc.). Then, the models were tested with a test data set (the remaining 20% of the available data) to determine how close is the regression to the actual dataset. The custom codes are posted on GitHub (https://github.com/jtanumihardja/IE7300.git).

Standardization with a standard scaler was also utilized for distance-based models as the pollutants and meteorological data has different units and scale.

Model Outcome and Performance Metrics

Each model parameter, hyperparameter, and feature importance are discussed in this section. R-squared (accuracy) and root mean squared error (RMSE) are the metrics used to compare the performance of the models. The summary of the performance metrics for each model is listed below.



	Regression Model (% sample)	RMSE (train)	RMSE (test)	R^2 (train)	R^2 (test)	computation time (sec)
0	Linear Regression (100%)	30.4902	30.1671	0.8561	0.8552	7.96
1	Lasso Regression (100%)	30.4918	30.1697	0.8561	0.8551	7.99
2	Lasso Regression (10%)	30.6266	31.4988	0.8571	0.8542	0.295
3	Decision Tree - DT (10%)	27.2196	27.569	0.8871	0.8883	287.21
4	Random Forest - RF (10%)	25.0058	27.2963	0.9047	0.8905	665.88
5	Random Forest - RF (1%)	21.9691	27.1194	0.9232	0.8849	57.24
6	Gradient Boost - GB (10%)	21.9358	23.7653	0.9267	0.917	610.94
7	K-Nearest Neighbor -KNN (1%)	27.633	35.009	0.8785	0.8081	3.57

Figure 2. Performance metrics for Air Quality dataset to predict PM2.5 WITH PM10 feature

Linear and Lasso Regressions

The linear regression utilized gradient descent as the optimizer of the cost function. The optimum parameters for this dataset are 1000 iterations and a learning rate of 0.05. The optimized lasso regression parameters are a learning rate of 0.05 and an alpha of 0.1. Both the linear and lasso regressions are trained and tested using the 100% samples available (>300,000 rows) and 10% sample to be compared with the other models with 10% samples only.

The regular custom linear regression (from statmodel folder, code pushed to Github) with gradient batch descent successfully trained and predicted the full dataset. The R^2 for both the train and test datasets are good at ~0.85 (85%). It is an indicator that the models are not overfitted (have relatively low bias and low variance). The RMSE is also similar for the train and test data (~30 ppm). This value is well below the standard deviation and acceptable RMSE for the model. This linear model result will be used as a baseline for comparing other models.

The feature importance list is obtained from the list of coefficients of the features. The feature 'year', 'SO2', and 'O3' are the top important features (highest linear model coefficient). This finding does not support the visualization found above though. Will be compared with the lasso regression result next.

For the lasso regression, the R^2 value for the train lasso is also good at 0.8557. The test dataset R^2 value is very close at 0.8565. There is no overfitting in this model. Both the R^2 and RMSE values are very similar to the linear model above. The lasso regularization does not really improve the linear model performance, though it shows a better feature importance list that matched other models well. Other learning rates and lambda values were trialed, but it doesn't change the accuracy to be better.

	Features	Importance
0	year	79.431143
5	SO2	49.414644
8	O3	23.410423
12	RAIN	19.995995
9	TEMP	5.784761
11	DEWP	4.088955
6	NO2	2.748844
7	CO	2.714977
1	month	0.655909
14	WSPM	-0.027460
4	PM10	-0.229029
13	wd	-0.451773
15	station	-0.537803
3	hour	-0.908143
2	day	-1.635631
10	PRES	-12.602704

	Features	Importance
4	PM10	49.426140
7	CO	23.518849
11	DEWP	19.437018
8	O3	5.515030
10	PRES	3.927786
5	SO2	2.699635
6	NO2	2.596638
0	year	0.620371
13	wd	0.002453
3	hour	-0.179867
12	RAIN	-0.381462
14	WSPM	-0.587031
15	station	-0.775199
2	day	-0.846990
1	month	-1.557143
9	TEMP	-12.108225

Figure 3 Linear (left) and lasso (right) regression feature importance list

Decision Tree

The decision tree model with max depth = 4 and minimum sample per split =3 was modeled. The chosen features (5 out of 16 features) for the decision tree are listed below. The root level is the "PM10" and it is still "PM10" at the 1st level. This shows that PM10 is doing a great job to split the data initially as it is chosen to be the root node (even utilized at levels 3 and 4, too). This is in line with the discussion in the visualization section above. The PM10 column is removed from the features list and the results are shown in the next section. The variance reduction is relatively high for the nodes as the value range is high.

CO (carbon monoxide) is chosen at tree level 2, which is also predicted since it has a higher correlation with PM2.5. It is interesting that NO2 (nitrogen dioxide) is not chosen at all, maybe because it has a high inter-correlation with CO.

The atmospheric measurement only enters the equation at the 4th level, and it is only the DEWP (dew point) variable, which is the same as the Lasso Feature importance. The other chosen features are the year and month features. The unchosen features for the DT among Lasso regressions top 8 are O3, Pressure, NO2, and SO2.

	featurename	treelevel
30	PM10	0
14	PM10	1
6	PM10	2
13	CO	2
2	CO	3
12	PM10	3
0	PM10	4
3	CO	4
7	DEWP	4
15	year	4
23	month	4

Figure 4. Feature importance of Decision Tree Regression Model

The RMSE of the decision tree model is still well below the standard deviation of the data (~80 ppm) so it is an acceptable value. The R^2 of the training set is 0.8991 while the test dataset is only 0.05 below that. This shows that the model is not overfitting. With values close to 90%, this model explained 90% of the variance in the PM2.5 value. Random Forest or Gradient Boosting is expected to improve these metrics.

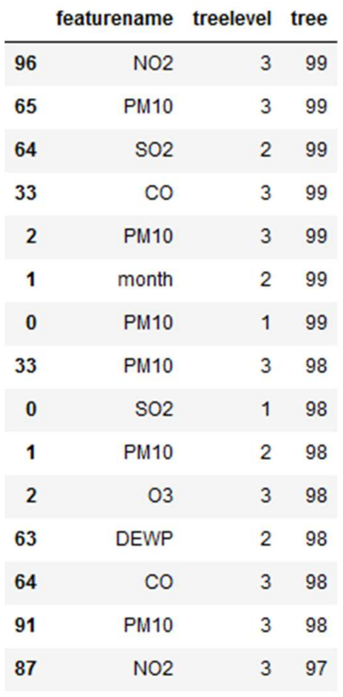
Random Forest

The random forest (RF) model is conducted with a 10% sample to minimize computation time and memory. The computational time of each model is also shown in Figure 2.

The RF model was first initialized with the same hyperparameter as DT, max_depth = 4 and number of trees = 100. However, the R^2 value is 0.7956 for the train dataset & 0.7718 for the test dataset. This metric is even lower than the regular linear regression. The max_depth was increased to 8 (double) and significant performance improvements were obtained. The train R^2 is 0.9211 and the test's is 0.8616. Both are great numbers with minimal overfitting. We can try to increase the max_depth again but we might encounter overfitting. Therefore, the final model max_depth is 8.

The RMSE score for the trained model is the best so far. The RMSE for the test model is higher but it is still well below the standard deviation of PM2.5.

The top features for the random forest mode are listed below. The features are sorted from the most used. The tree level and number of trees that the features used at the specified tree level are also listed.



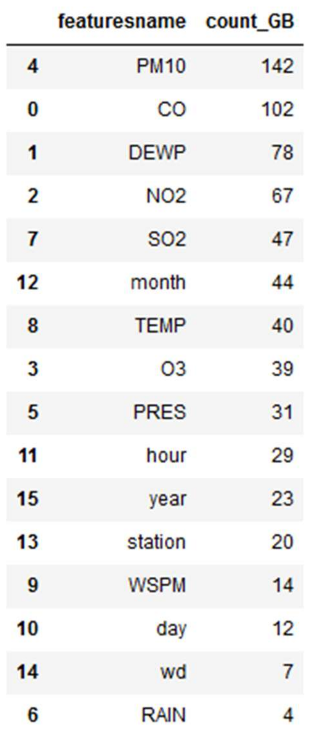
	featurename	treelevel	tree
96	NO2	3	99
65	PM10	3	99
64	SO2	2	99
33	CO	3	99
2	PM10	3	99
1	month	2	99
0	PM10	1	99
33	PM10	3	98
0	SO2	1	98
1	PM10	2	98
2	O3	3	98
63	DEWP	2	98
64	CO	3	98
91	PM10	3	98
87	NO2	3	97

Figure 5. Feature importance of Random Forest Regression Model

Gradient Boost

The gradient boost (GB) model is also conducted with a 10% sample to minimize computation time and memory. The hyperparameters set for the modes are: number of trees = 100 and a learning rate of 0.1. The performance metrics of the train and test datasets are the highest among the rest of the models. The training dataset has an accuracy of 0.9267 and an RMSE of 21.96. The test dataset has an accuracy of 0.917 and an RMSE of 23.76. Although the train data accuracy is very close to the random forest model, the test data accuracy is higher. However, it requires the highest computational cost to run this model.

The feature importance is listed below. The number of times that the features were used in the gradient boosting weak learner is listed in the second column.



	featuresname	count_GB
4	PM10	142
0	CO	102
1	DEWP	78
2	NO2	67
7	SO2	47
12	month	44
8	TEMP	40
3	O3	39
5	PRES	31
11	hour	29
15	year	23
13	station	20
9	WSPM	14
10	day	12
14	wd	7
6	RAIN	4

Figure 6. Feature Importance for Gradient Boost Regression Model

K-Nearest Neighbor (KNN)

The KNN model successfully fit and predicted the target variable (PM2.5) but it requires high computational storage. Therefore, only 1% of the standardized data were used for this model as there was a memory error message.

The train and test RMSE values are lower than the standard deviation of PM2.5 so it is a good model. The R^2 of the train is 0.8785 while the test is 0.8081. It doesn't have an overfitting issue because the test data R^2 is only 7% lower than the train. Hence, it has low bias and low variance.

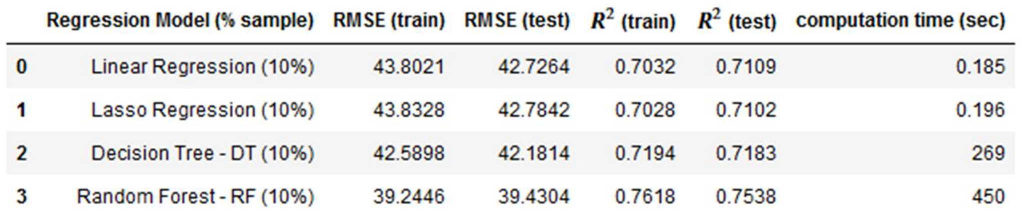
Neural Network

A simple neural network with one hidden layer with 8 nodes was tested for the 1% sample of the dataset. The optimal hyperparameters are learning rate = 0.001 and number of iterations = 100. The cost function was set to be a mean-squared error. ReLU (rectified Linear Unit) was used for the activation functions. The batch gradient descent was used to optimize the cost function.

The accuracy for the training data is 0.8849 with RMSE = 31.75. The accuracy is surprisingly lower than the best model above (gradient boosting), though it is higher than the linear regression base model. Increasing the number of hidden layers, testing with a 10% dataset, and/or increasing the number of iterations might increase the accuracy significantly. The test data metrics are R^2 = 0.8465 and RMSE = 28.45. The model does not overfit as it has a lower bias and lower variance.

Model Outcome & Performance Metrics without PM10

Model training and evaluation without the PM10 feature is conducted as PM2.5 has a strong correlation with PM10 (PM2.5 is part of PM10). To simplify the comparison, only 10% of the sample data is used for this section. Here are the performance metrics for the updated dataset (15 features instead of 16). KNN is not conducted as it requires 1% of data (anomaly). Gradient boost was not tested due to time constraints.



	Regression Model (% sample)	RMSE (train)	RMSE (test)	R^2 (train)	R^2 (test)	computation time (sec)
0	Linear Regression (10%)	43.8021	42.7264	0.7032	0.7109	0.185
1	Lasso Regression (10%)	43.8328	42.7842	0.7028	0.7102	0.196
2	Decision Tree - DT (10%)	42.5898	42.1814	0.7194	0.7183	269
3	Random Forest - RF (10%)	39.2446	39.4304	0.7618	0.7538	450

Figure 7. Summary of Performance Metrics WITHOUT PM10 feature

The performance metrics of the models without the PM10 column are lower than before, as suspected. R^2 are still ~70% so they still explained >70% of the variance. The accuracy might increase if we remove the outlier in the pollutant's column (future work). Similar to the previous result, random forest is the best regression model to predict PM2.5 among the 4 models.

Model Selection

The best regression model based on the performance metrics is the gradient boost model. It has the highest train and test accuracy and the lowest RMSE. Therefore, the gradient boost model should be used if the most accurate model is the objective of the use case. For a quick and easy result, lasso regression has comparable accuracy (~8% lower) and RMSE (8 units lower).

Conclusion

The model performance metrics above summarize 7 different models to predict the dependent variable, PM2.5, in various Beijing cities using the air pollutants dataset as well as atmospheric weather station data. The regression models successfully predict the train and test data (80-20 split) with relatively high accuracy. Most of the models can explain >= 85% of the variance in the dataset, even with a smaller fraction of the dataset.

The linear and lasso regressions were conducted using 100% of the samples (382,168 rows). Since the requirement is only 10,000 rows and most of the numerical features are in standard/skewed normal distribution, by central tendency the model should be scalable. Hence, only 10% of the dataset was used in the more complex model like DT and RF. Lasso regression was modeled using 100% and 10% (38,217 rows) data and the RMSE & R^2 values are very similar with <10% of computational time.

The chosen model for this dataset is either Lasso Regression or Gradient Boost depending on the use case and performance target preference. Gradient Boost produces the highest R^2 and lowest RMSE for training and test data but with higher computation time. Lasso Regression also produces comparable performance metrics with fast calculation and the ability to extract feature importance.

For future works, a 10% dataset for KNN and SVM shall be tested. Also, all the models can be tested with 100% of the dataset for a more accurate number.

Appendix

References

Baheti, P. (2022, October 21). *Activation Functions in Neural Networks [12 Types & Use Cases]*. Retrieved from V7 Labs: https://www.v7labs.com/blog/neural-networks-activation-functions

Shi, A. (2022, January 18). *Overview of Supervised Machine Learning Algorithms*. Retrieved from Towards Data Science: https://towardsdatascience.com/overview-of-supervised-machine-learning-algorithms-a5107d036296

Shobha, G., & Rangaswamy, S. (2018). Chapter 8 - Machine Learning. In V. N. Gudivada, & C. Rao, *Computational Analysis and Understanding of Natural Languages: Principles, Methods, and Applications. Handbook of Statistics, Vol. 38* (pp. 197-228). Bengaluru, India: Elsevier.

Ye, M. F. (2022). 4.2 Causes and Consequences of Air Pollution in Beijing, China. In T. R. Edited by Kylienne A. Clark, *Environmental ScienceBites* (pp. 123-129). Colombus, Ohio: PRESSBOOKS. Retrieved from PRESSBOOKS: https://ohiostate.pressbooks.pub/sciencebites/part/pollution/

Zhang, S., Guo, B., Dong, A., He, J., Xu, Z., & Xi Chen, S. (2017). Cautionary tales on air-quality improvement in Beijing. *The Royal Society Collection*.

Additional References

Zhang, Shuyi; Guo, Bin; Dong, Anlan; He, Jing; Xu, Ziping; Xi Chen, Song (2017): Supplementary material from "Cautionary tales on air-quality improvement in Beijing". The Royal Society. Collection. https://doi.org/10.6084/m9.figshare.c.3865483.v2

Mason F. Ye. Causes and Consequences of Air Pollution in Beijing, China. https://ohiostate.pressbooks.pub/sciencebites/chapter/causes-and-consequences-of-air-pollution-in-beijing-china/

Beijing Air Pollution: Real-time Air Quality Index (AQI). https://aqicn.org/city/beijing/

Ambient Air Quality. https://www.paho.org/en/topics/air-quality-and-health/ambient-air-quality

AirNow (EPA): Seattle Air Quality. https://www.airnow.gov/?city=Seattle&state=WA&country=USA

Puget Sound Data Summary. https://pscleanair.gov/615/Data-Summary

King County Open Data. https://data.kingcounty.gov/browse?q=air%20quality&sortBy=relevance