Air Quality Index of India

Data Preprocessing Project Report

Submitted by-Kshitij Kumar

Name-Kshitij Kumar

Roll no.-201020426

Branch-DSAI

Session-2020-2024

INTRODUCTION

This report is based on Data Preprocessing Techniques implemented on India's Air Quality Index. This air quality index is monitored from different monitoring stations across the India. The pollutants measured are Sulphur Dioxide (SO_2), Nitrogen Dioxide (NO_2), Particulate Matter (PM10 and PM2.5), Carbon Monoxide (CO), Ozone(O_3).

Dataset URL

https://www.kaggle.com/chitwanmanchanda/indias-air-quality-index

Overview of Data

In [2]:
```
df = pd.read_csv('C:/Users/Anand Kumar Sahu/OneDrive/Desktop/ml resources/Air_Quality.csv')
df
```

Out[2]:

	id	country	state	city	station	pollutant_id	last_update	pollutant_min	pollutant_max	pollutant_avg
0	1	India	Andhra_Pradesh	Amaravati	Secretariat, Amaravati - APPCB	PM2.5	21-10-2021 01:00	69.0	109.0	86.0
1	2	India	Andhra_Pradesh	Amaravati	Secretariat, Amaravati - APPCB	PM10	21-10-2021 01:00	82.0	138.0	105.0
2	3	India	Andhra_Pradesh	Amaravati	Secretariat, Amaravati - APPCB	NO2	21-10-2021 01:00	10.0	42.0	19.0
3	4	India	Andhra_Pradesh	Amaravati	Secretariat, Amaravati - APPCB	NH3	21-10-2021 01:00	4.0	5.0	4.0
4	5	India	Andhra_Pradesh	Amaravati	Secretariat, Amaravati - APPCB	SO2	21-10-2021 01:00	NaN	NaN	NaN
...	...	...	...	...	...	...	...	...	...	...
1831	1832	India	West_Bengal	Kolkata	Victoria, Kolkata - WBPCB	NO2	21-10-2021 01:00	10.0	22.0	15.0
1832	1833	India	West_Bengal	Kolkata	Victoria, Kolkata - WBPCB	NH3	21-10-2021 01:00	1.0	3.0	2.0
1833	1834	India	West_Bengal	Kolkata	Victoria, Kolkata - WBPCB	SO2	21-10-2021 01:00	6.0	28.0	10.0
1834	1835	India	West_Bengal	Kolkata	Victoria, Kolkata - WBPCB	CO	21-10-2021 01:00	34.0	92.0	41.0
1835	1836	India	West_Bengal	Kolkata	Victoria, Kolkata - WBPCB	OZONE	21-10-2021 01:00	10.0	116.0	43.0

Contents of Columns

1. ID: Unique Identifier for each Data Point.
2. Country: Name of the country (India in this case)
3. State: Name of the state in the country
4. City: Name of the City
5. Station: Name of the Air Quality Monitoring Station
6. Pollutant ID: ID of the Pollutant
7. Last Update: Time when information was last updated (Date-Time values)
8. Pollutant Min: Minimum units of the Pollutant measured
9. Pollutant Max*:* Maximum units of the Pollutant measured
10. Pollutant Avg: Average units of the pollutant measured

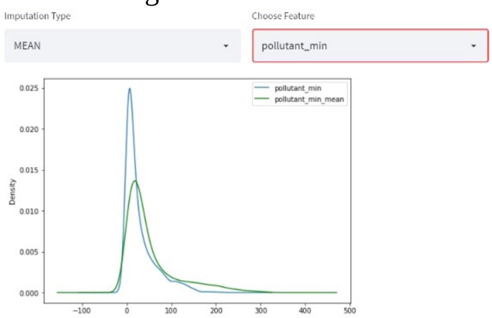
2.Data imputation

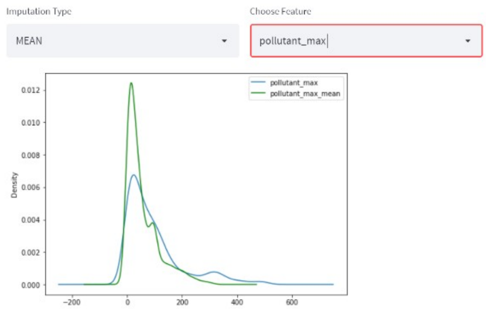
The dataset contains dirtiness in the form of missing values. The columns ‘pollutant_min’, ‘pollutant_max’ and ‘pollutant_avg’ contains 5.77% ,5.78% and 5.82% of missing values respectively.
Missing Values:

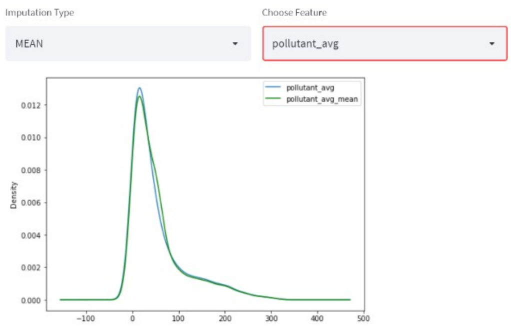
```
In [3]: df.isnull().mean()

Out[3]: id               0.000000
        country          0.000000
        state            0.000000
        city             0.000000
        station          0.000000
        pollutant_id     0.000000
        last_update      0.000000
        pollutant_min    0.057734
        pollutant_max    0.057190
        pollutant_avg    0.058279
        dtype: float64
```

These missing values can be removed using various Data Imputation Techniques:

1. **By Mean**
   Missing values of the columns are replaced by mean of their column.



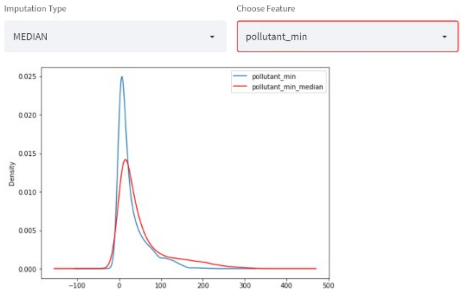


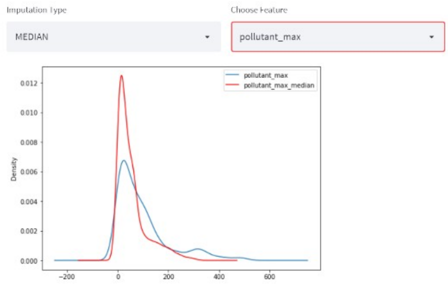


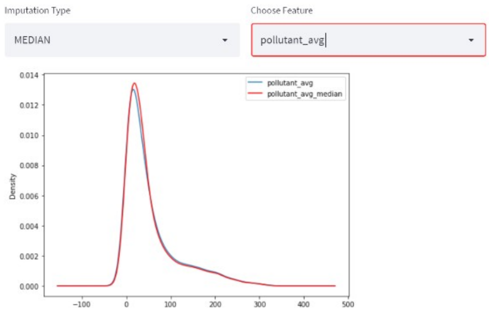
   On applying and comparing the mean method, we can clearly notice that it has performed well in ‘pollutant_min’ and ‘pollutant_avg’ columns and failed to perform on ‘pollutant_max’ column.

2. **By Median**

Missing values are replaced by Median of their Column.

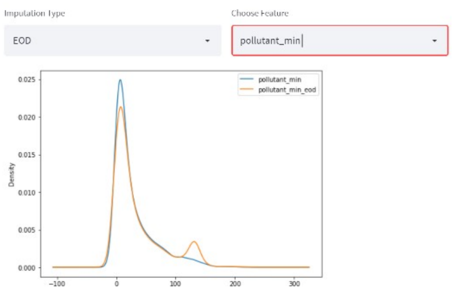


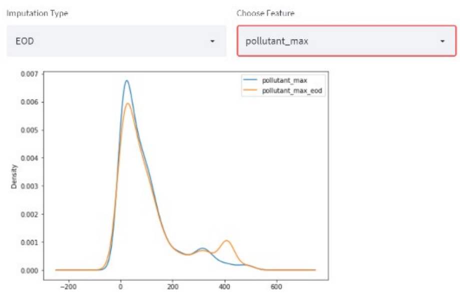


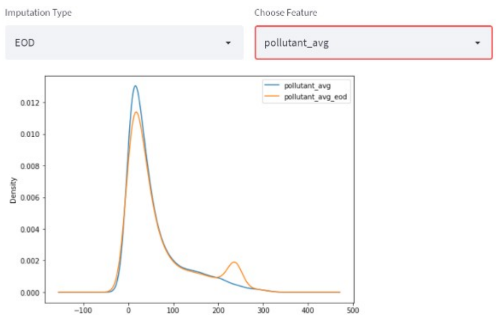


The results are very similar to previous case. Both ‘pollutant_min’ and ‘pollutant_avg’ column have performed well while ‘pollutant_max’ has failed to yield good results.

3. **End Of Distribution**
   In this method the missing values have been replaced by [mean + 3*standard deviation] of their respective columns.

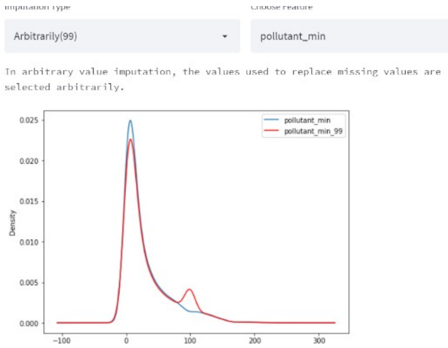


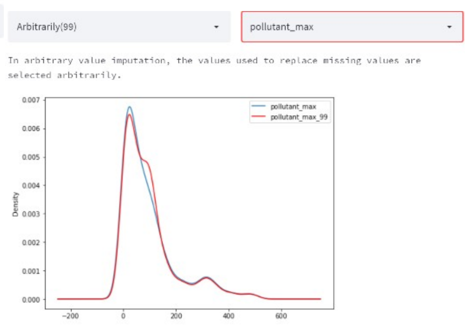




It has performed well in all three columns but we can notice a bump between value 200-300. It means the EOD value is quite larger than expected.

4. **Arbitrarily (99)**
   In this method an arbitrarily value is passed into the missing cells. Here the value is 99.



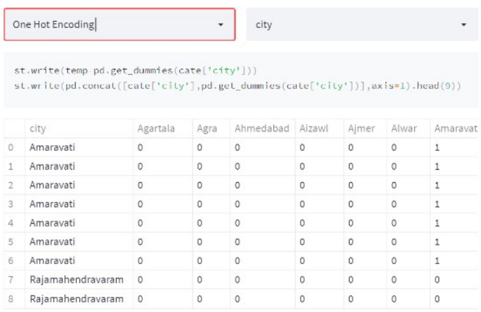


This method yield good results in ‘pollutant_max’ column while we can notice an unexpected bump on another column.

2.Categorical Encoding

This technique is used to convert categorical columns to numeric so that data can be used to build statistical model.

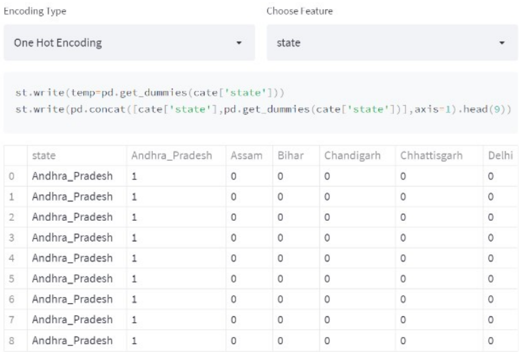
1. One Hot Encoding



One Hot Encoding | city

```
st.write(temp pd.get_dummies(cate['city']))
st.write(pd.concat([cate['city'],pd.get_dummies(cate['city'])],axis=1).head(9))
```

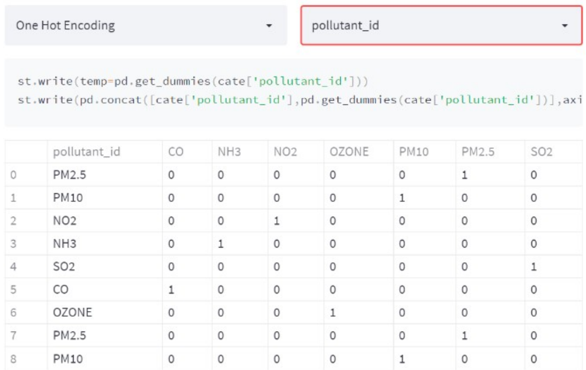
	city	Agartala	Agra	Ahmedabad	Aizawl	Ajmer	Alwar	Amaravat
0	Amaravati	0	0	0	0	0	0	1
1	Amaravati	0	0	0	0	0	0	1
2	Amaravati	0	0	0	0	0	0	1
3	Amaravati	0	0	0	0	0	0	1
4	Amaravati	0	0	0	0	0	0	1
5	Amaravati	0	0	0	0	0	0	1
6	Amaravati	0	0	0	0	0	0	1
7	Rajamahendravaram	0	0	0	0	0	0	0
8	Rajamahendravaram	0	0	0	0	0	0	0



Encoding Type: One Hot Encoding | Choose Feature: state

```
st.write(temp=pd.get_dummies(cate['state']))
st.write(pd.concat([cate['state'],pd.get_dummies(cate['state'])],axis=1).head(9))
```

	state	Andhra_Pradesh	Assam	Bihar	Chandigarh	Chhattisgarh	Delhi
0	Andhra_Pradesh	1	0	0	0	0	0
1	Andhra_Pradesh	1	0	0	0	0	0
2	Andhra_Pradesh	1	0	0	0	0	0
3	Andhra_Pradesh	1	0	0	0	0	0
4	Andhra_Pradesh	1	0	0	0	0	0
5	Andhra_Pradesh	1	0	0	0	0	0
6	Andhra_Pradesh	1	0	0	0	0	0
7	Andhra_Pradesh	1	0	0	0	0	0
8	Andhra_Pradesh	1	0	0	0	0	0



One Hot Encoding | pollutant_id

```
st.write(temp=pd.get_dummies(cate['pollutant_id']))
st.write(pd.concat([cate['pollutant_id'],pd.get_dummies(cate['pollutant_id'])],axi
```

	pollutant_id	CO	NH3	NO2	OZONE	PM10	PM2.5	SO2
0	PM2.5	0	0	0	0	0	1	0
1	PM10	0	0	0	0	1	0	0
2	NO2	0	0	1	0	0	0	0
3	NH3	0	1	0	0	0	0	0
4	SO2	0	0	0	0	0	0	1
5	CO	1	0	0	0	0	0	0
6	OZONE	0	0	0	1	0	0	0
7	PM2.5	0	0	0	0	0	1	0
8	PM10	0	0	0	0	1	0	0

2. Label Encoding

Label Encoding | pollutant_id

```
le.fit(cate['pollutant_id'])
cate['le_pollutant_id']=le.transform(cate['pollutant_id'])
st.write(cate)
```

	state	city	pollutant_id	le_pollutant_id
0	Andhra_Pradesh	Amaravati	PM2.5	5
1	Andhra_Pradesh	Amaravati	PM10	4
2	Andhra_Pradesh	Amaravati	NO2	2
3	Andhra_Pradesh	Amaravati	NH3	1
4	Andhra_Pradesh	Amaravati	SO2	6
5	Andhra_Pradesh	Amaravati	CO	0
6	Andhra_Pradesh	Amaravati	OZONE	3
7	Andhra_Pradesh	Rajamahendravar...	PM2.5	5
8	Andhra_Pradesh	Rajamahendravar...	PM10	4
9	Andhra_Pradesh	Rajamahendravar...	NO2	2

Label Encoding | state

```
le.fit(cate['state'])
cate['le_state']=le.transform(cate['state'])
st.write(cate)
```

	state	city	pollutant_id	le_state
39	Assam	Guwahati	CO	1
40	Assam	Guwahati	OZONE	1
41	Bihar	Gaya	PM2.5	2
42	Bihar	Gaya	NO2	2
43	Bihar	Gaya	SO2	2
44	Bihar	Gaya	CO	2
45	Bihar	Gaya	OZONE	2
46	Bihar	Gaya	PM2.5	2
47	Bihar	Gaya	PM10	2
48	Bihar	Gaya	NO2	2

Label Encoding | city

```
le.fit(cate['city'])
cate['le_city']=le.transform(cate['city'])
st.write(cate)
```

	state	city	pollutant_id	le_city
0	Andhra_Pradesh	Amaravati	PM2.5	6
1	Andhra_Pradesh	Amaravati	PM10	6
2	Andhra_Pradesh	Amaravati	NO2	6
3	Andhra_Pradesh	Amaravati	NH3	6
4	Andhra_Pradesh	Amaravati	SO2	6
5	Andhra_Pradesh	Amaravati	CO	6
6	Andhra_Pradesh	Amaravati	OZONE	6
7	Andhra_Pradesh	Rajamahendravaram	PM2.5	113
8	Andhra_Pradesh	Rajamahendravaram	PM10	113
9	Andhra_Pradesh	Rajamahendravaram	NO2	113

3. Frequency Encoding

Frequency | city

```
value_counts_city=cate['city'].value_counts().to_dict()
cate['city']=cate['city'].map(value_counts_city)
st.write(cate)
```

	state	city	pollutant_id
0	Andhra_Pradesh	7	PM2.5
1	Andhra_Pradesh	7	PM10
2	Andhra_Pradesh	7	NO2
3	Andhra_Pradesh	7	NH3
4	Andhra_Pradesh	7	SO2
5	Andhra_Pradesh	7	CO
6	Andhra_Pradesh	7	OZONE
7	Andhra_Pradesh	7	PM2.5
8	Andhra_Pradesh	7	PM10
9	Andhra_Pradesh	7	NO2

Frequency | state

```
value_counts_state=cate['state'].value_counts().to_dict()
cate['state']=cate['state'].map(value_counts_state)
st.write(cate)
```

	state	city	pollutant_id
0	28	Amaravati	PM2.5
1	28	Amaravati	PM10
2	28	Amaravati	NO2
3	28	Amaravati	NH3
4	28	Amaravati	SO2
5	28	Amaravati	CO
6	28	Amaravati	OZONE
7	28	Rajamahendravaram	PM2.5
8	28	Rajamahendravaram	PM10
9	28	Rajamahendravaram	NO2

Frequency | pollutant_id

```
value_counts_pollutant=cate['pollutant_id'].value_counts().to_dict()
cate['pollutant_id']=cate['pollutant_id'].map(value_counts_pollutant)
st.write(cate)
```

	state	city	pollutant_id
0	Andhra_Pradesh	Amaravati	272
1	Andhra_Pradesh	Amaravati	267
2	Andhra_Pradesh	Amaravati	271
3	Andhra_Pradesh	Amaravati	235
4	Andhra_Pradesh	Amaravati	260
5	Andhra_Pradesh	Amaravati	273
6	Andhra_Pradesh	Amaravati	258
7	Andhra_Pradesh	Rajamahendravaram	272
8	Andhra_Pradesh	Rajamahendravaram	267
9	Andhra_Pradesh	Rajamahendravaram	271

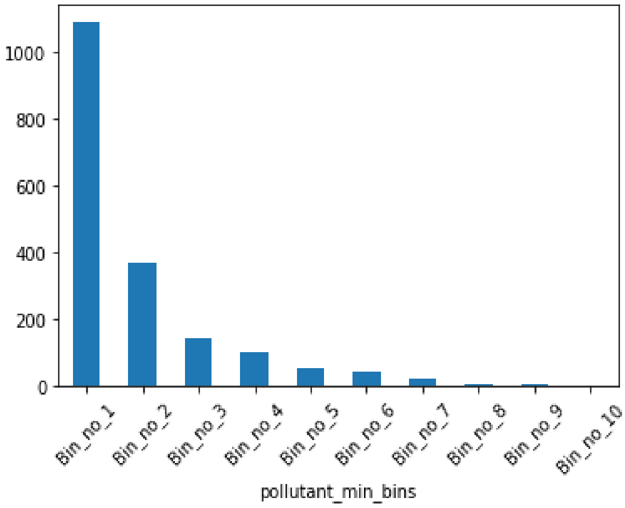
1. Ordinal Encoding

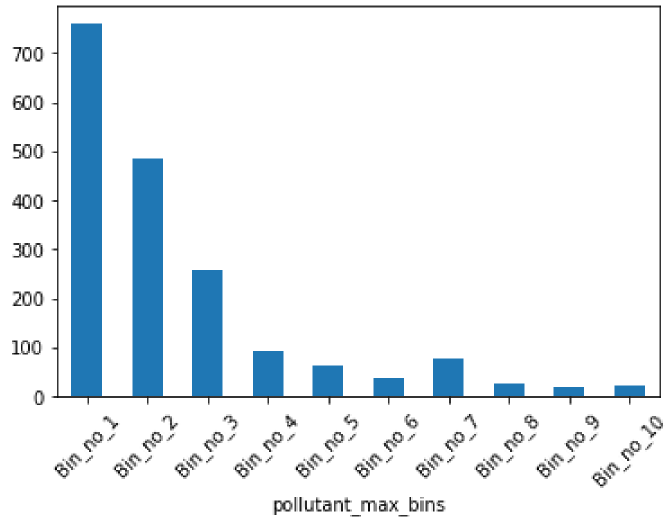
3.Discretization

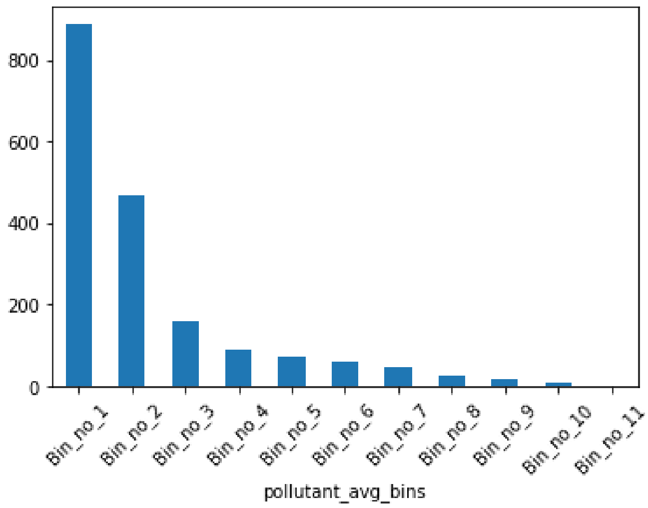
The process of converting continuous numeric values into discrete intervals is called discretization or binning.

1.Equal width discretization

The width or the size of all the intervals remains the same. An interval is also called a bin.



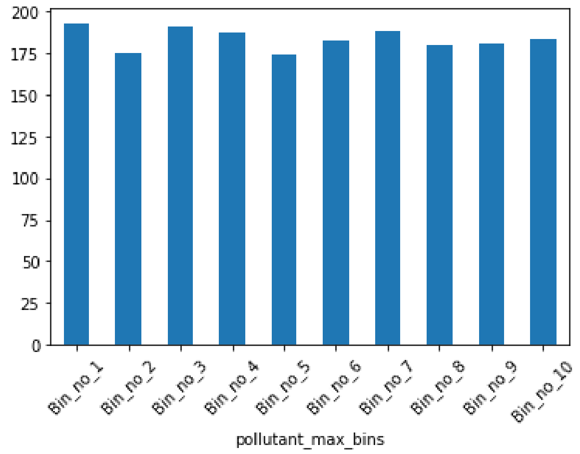


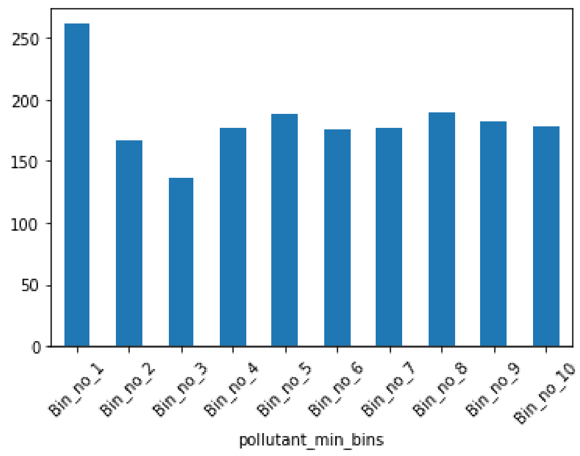


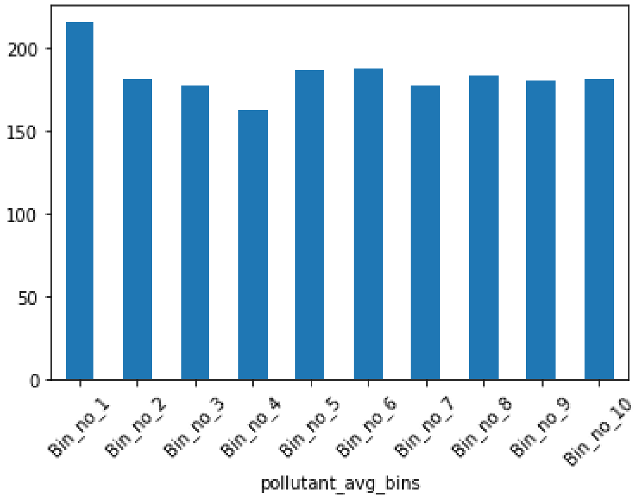
Here data of these columns are discretized into 10 bins. The intervals are 22, 50 and 31 for columns 'pollutant_min', 'pollutant_max' and 'pollutant_avg' respectively.
On observing the kde plots of these columns, initial bins contain max number of data points since data looks positively skewed. And 'pollutant_max' column have highest outliers.

2.Equal Frequency Discretization

In equal frequency discretization, the bin width is adjusted automatically in such a way that each bin contains exactly the same number of records or has the same frequency.



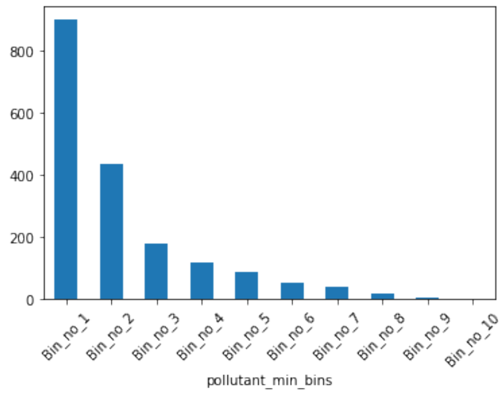


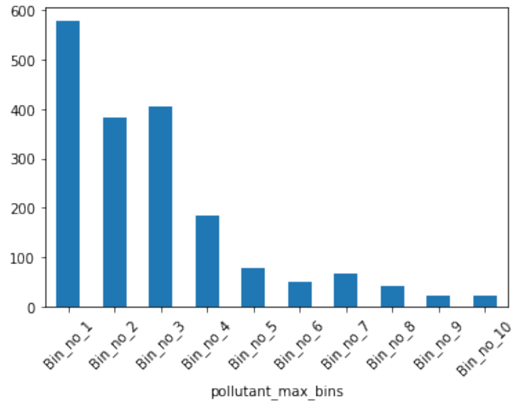


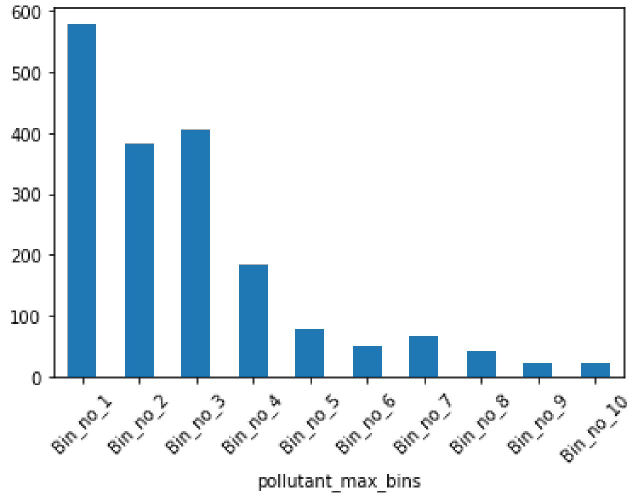
10 bins have been created for all three continuous numerical columns with varied width. ‘pollutant_max’ column yielded best results while ‘pollutant_avg’ is moderately and ‘pollutant_min’ is poorly. Since bin no 3 of pollutant_min contains very less data points compared to others in the column.

3.K-Means Discretization

K-means discretization is another unsupervised discretization technique based on the K-means algorithm.
A brief description of the K-Means algorithm is given below:
1. In the beginning, K random clusters of data points are created, where K is the number of bins or intervals.
2. Each data point is linked to the closest cluster centre.
3. The centres of all the clusters are updated based on the associated data points.





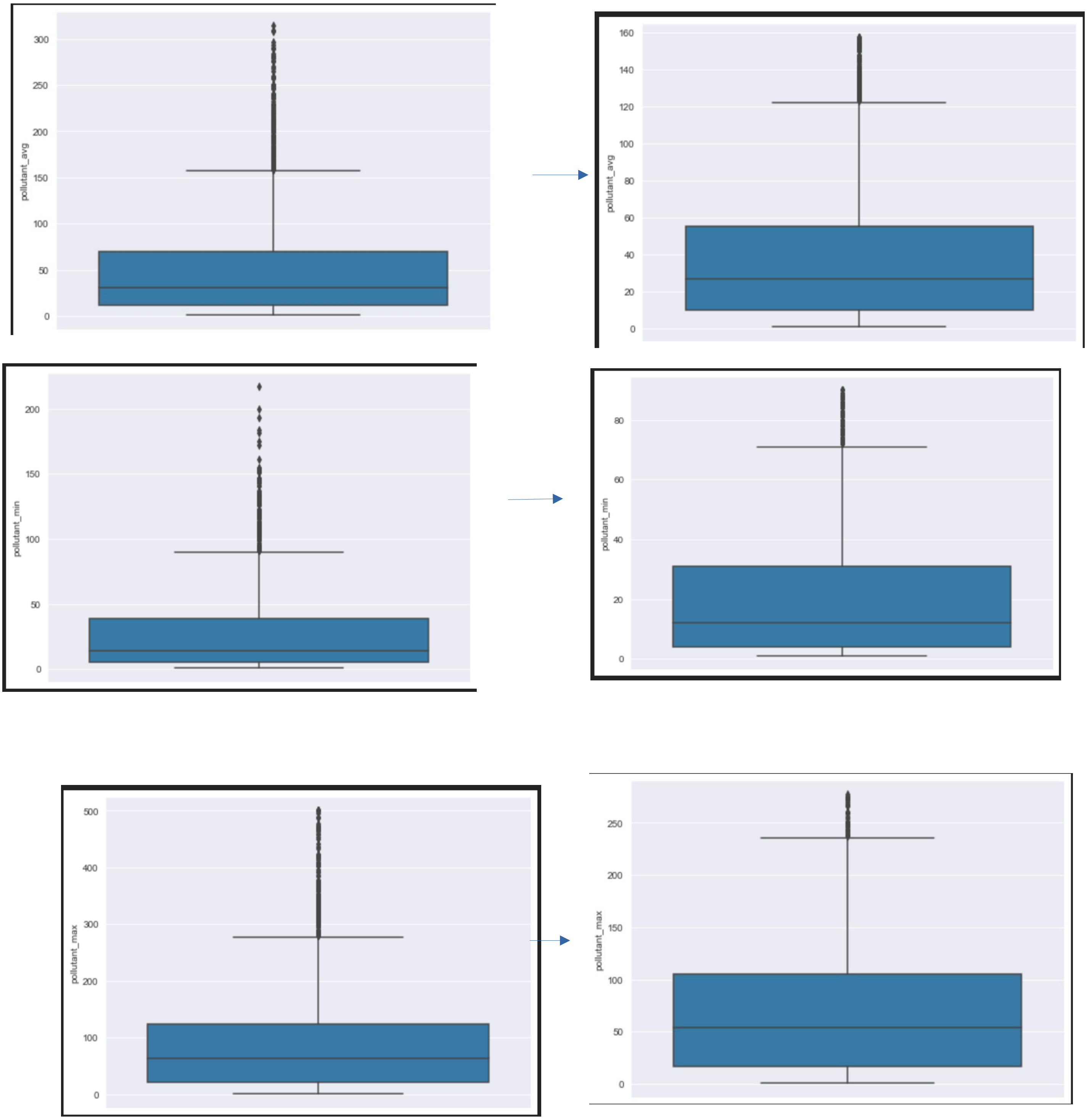


10 bin or clusters have been created with centers associated of all the three continuous numerical columns. Each bar represents a cluster and the number of data points closely associated to its center. The bin no 3 of column 'pollutant_max' have greater data points closer to its center than the bin no 2. Moreover all the bar graphs represents positively skewed data points in this data set.

6.Outlier handling

a. Outlier Trimming

Outlier trimming refers to simply removing the outliers beyond a certain threshold value.



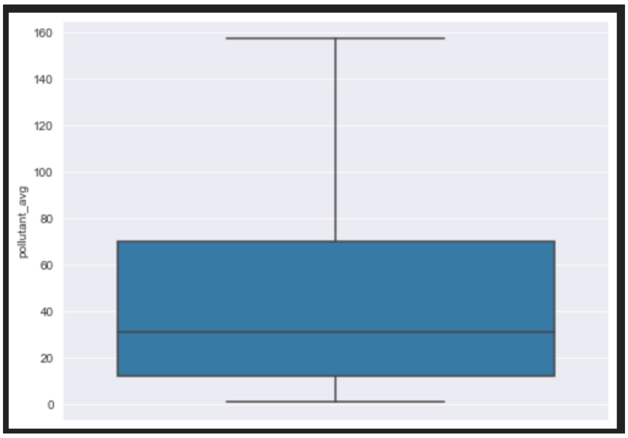
In this outlier trimming is done for all the three columns.Graph in left side is before trimming and in right side is after trimming.

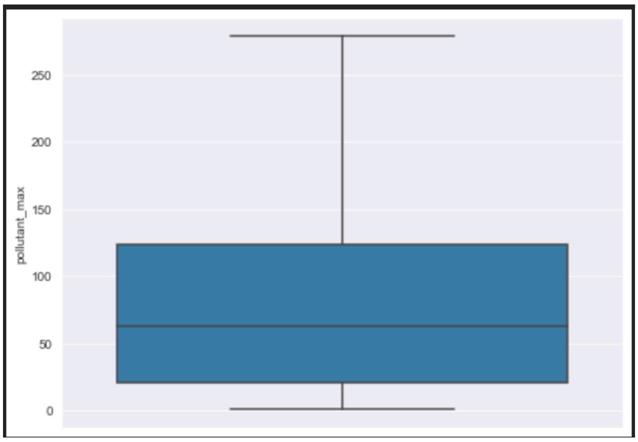
Observing the changes we could see that many outliers are removed which are very far away from others.

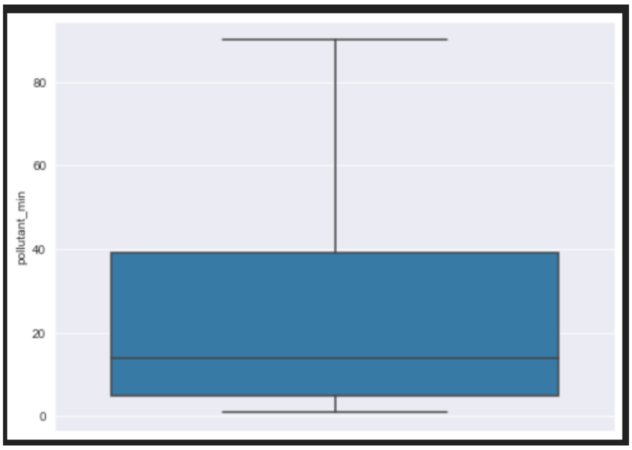
b.Outlier Capping

1.Using IQR-

IQR is the range between the first and the third quartiles namely Q1 and Q3: *IQR = Q3 – Q1*. The data points which fall below *Q1 – 1.5 IQR* or above *Q3 + 1.5 IQR* are outliers.



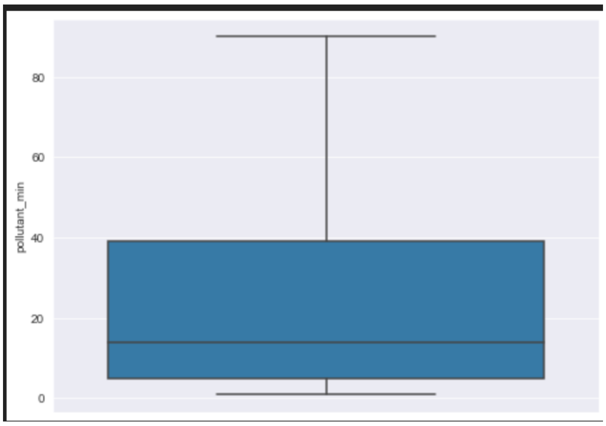


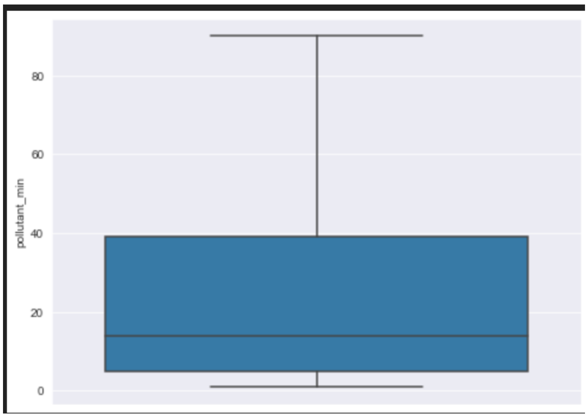


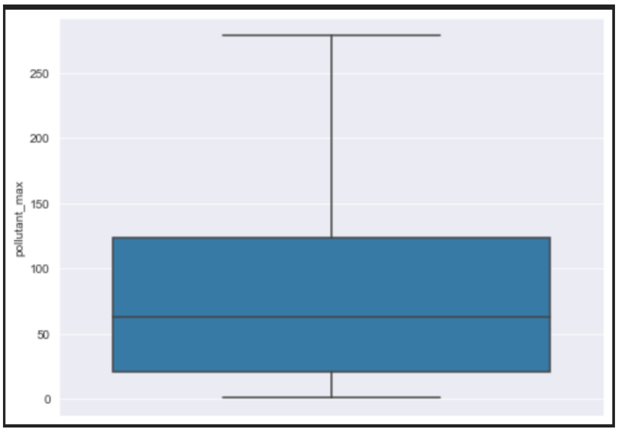
In this outlier is removed with Inter quantile limits.

2.Using mean and standard deviation

Instead of using the IQR method, the upper and lower thresholds for outliers can be calculated via the mean and standard deviation method.

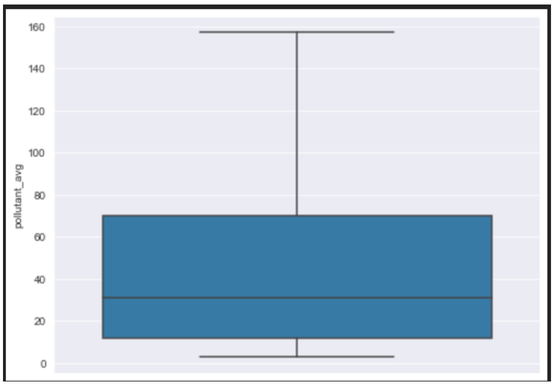


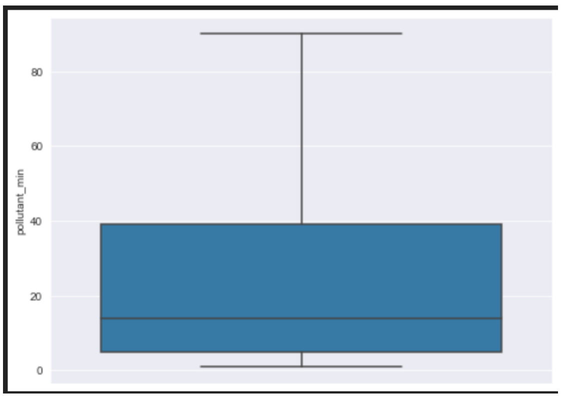


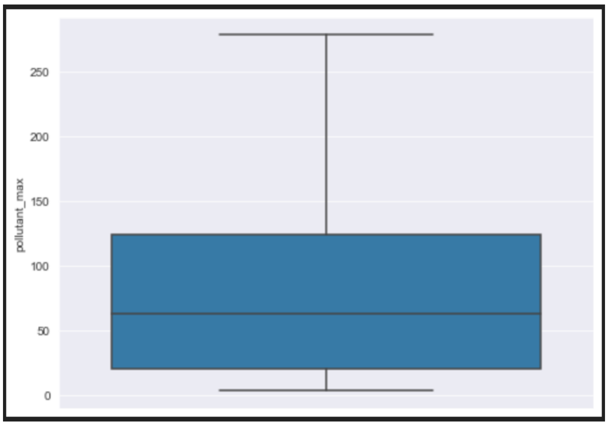


3.Using Quantiles

In this quantile information is used to set the lower and upper limits to find outliers.

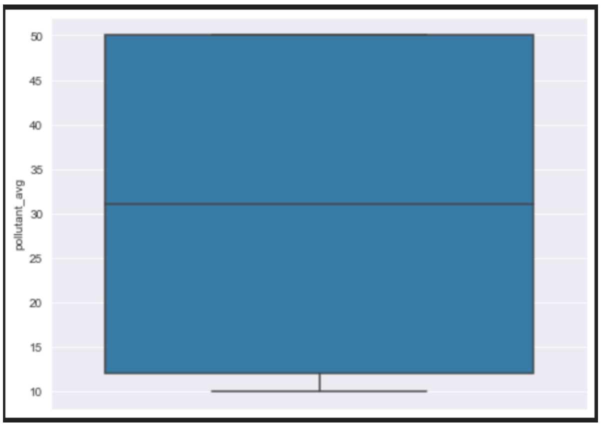


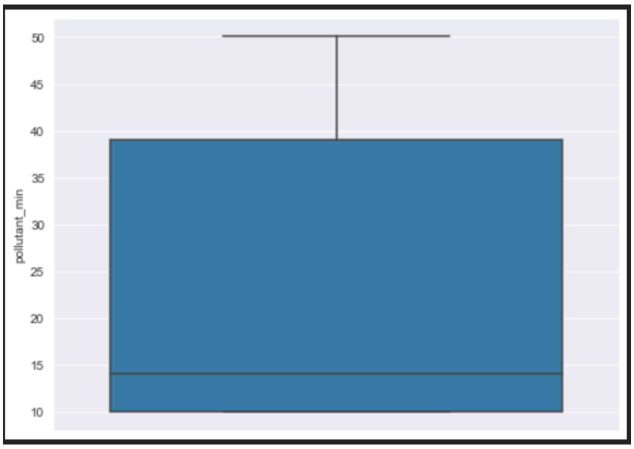


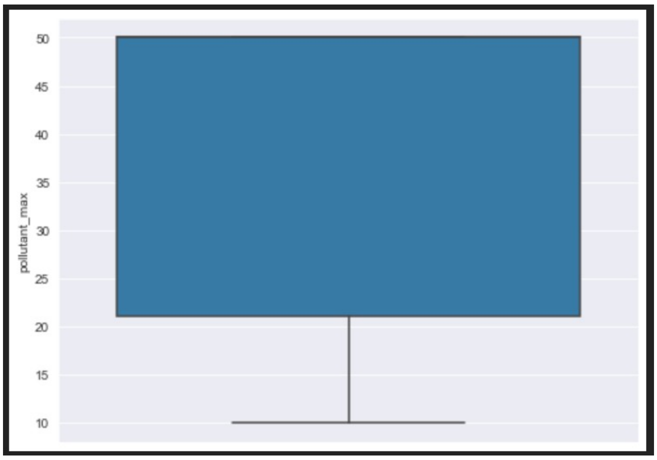


4.Outlier capping using custom values

In this we set the custom values for lower and upper limits to find the outliers.



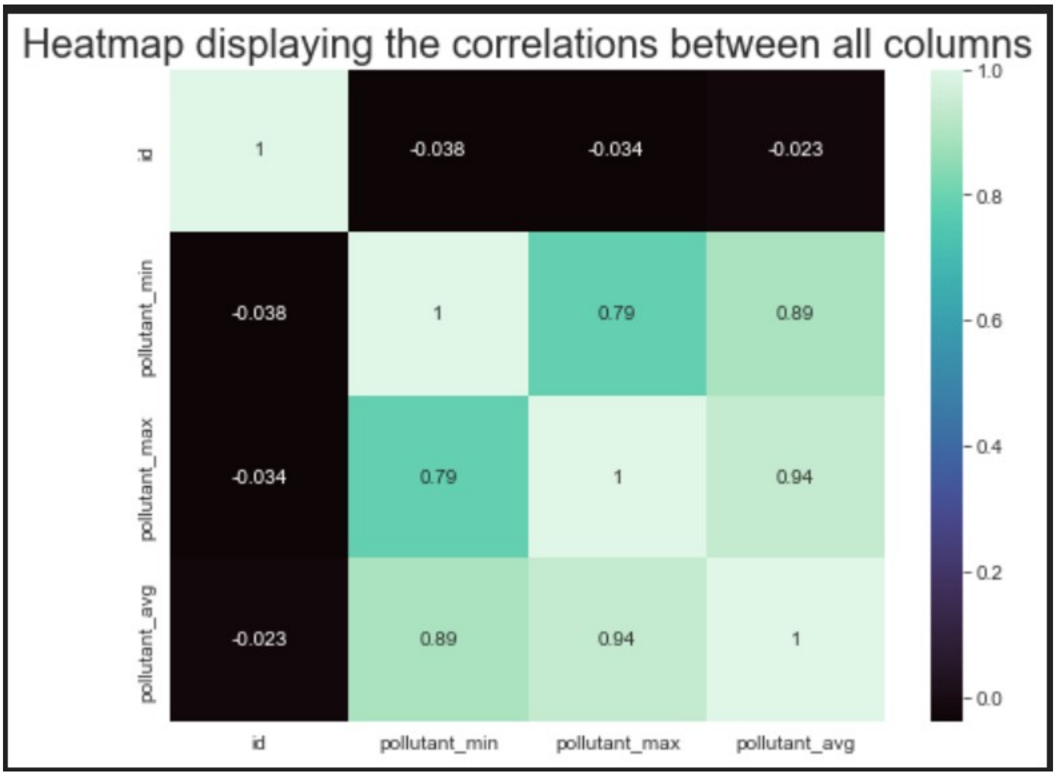




We can observe that all the outlier are removed by setting the custom values.

7.Correlation matrix with Heat map

Correlation states how the features are related to each other or the target variable.
Heat map makes it easy to identify which features are most related to the target variable.

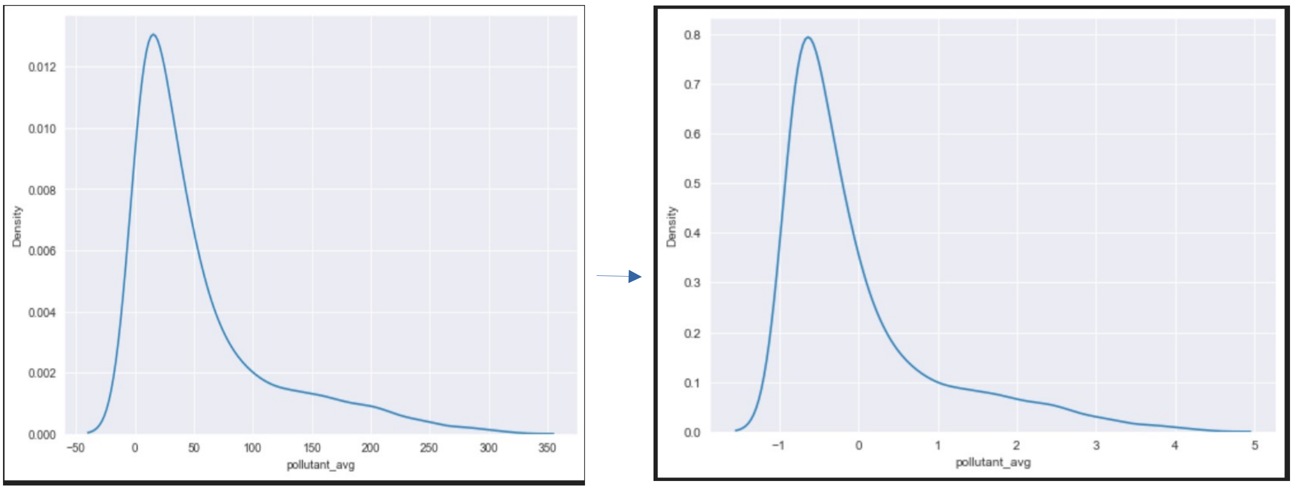


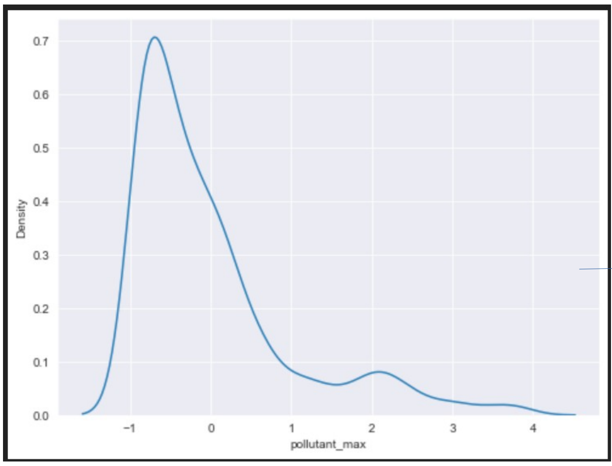
In this we can observe the correlation between all the columns of datasets in which some are negatively correlated and some positively correlated.

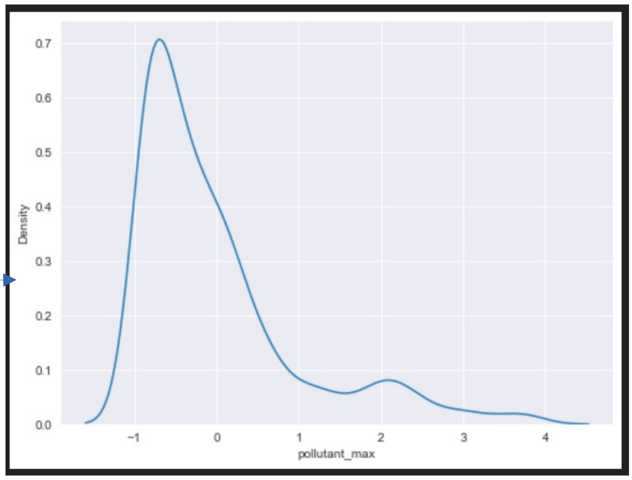
7.Transformation

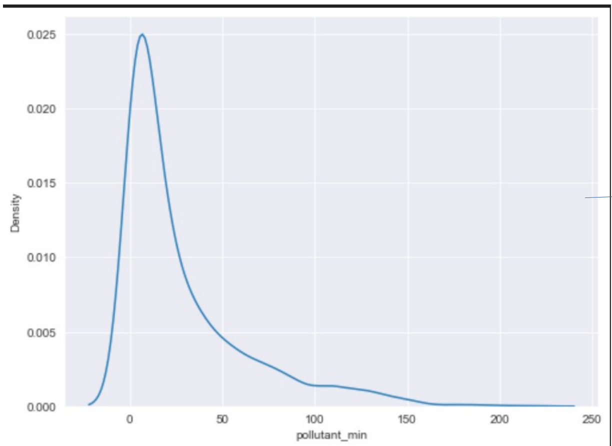
Standardisation(Z score normalisation)

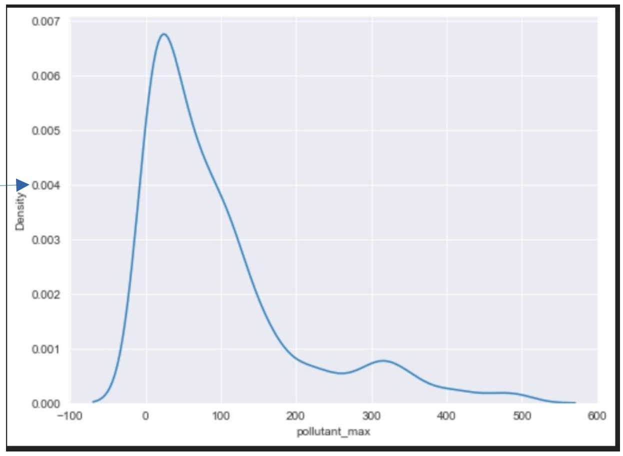
Standardization is the processing of centering the variable at zero and standardizing the data variance to 1.
To standardize the dataset, you simply have to subtract each data point from the mean of the datapoint and divide the result by the standard deviation of the data

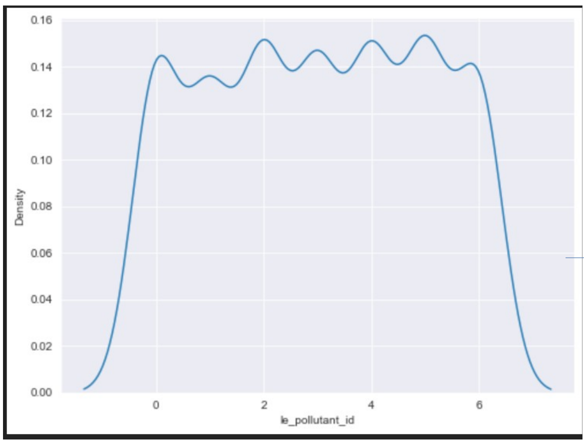


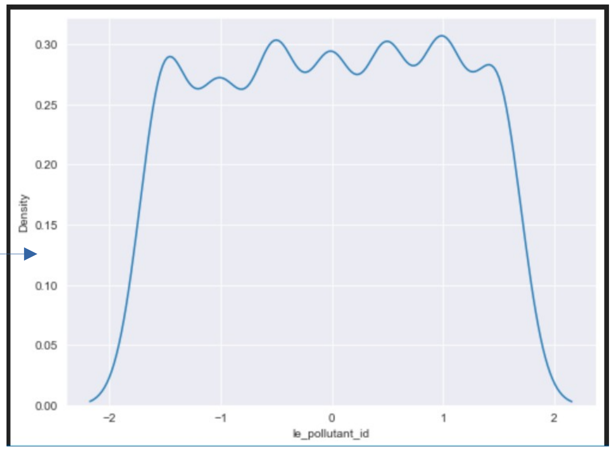

Density
pollutant_max


Density
pollutant_max


Density
pollutant_min


Density
pollutant_max


Density
le_pollutant_id


Density
le_pollutant_id

In this,data are transformed into normalised form as the data does not follow normalisation.In this mean is 0 and standard deviation is 1.