SKILL UP

IBM

AIR QUALITY ANALYSIS IN TAMIL NADU

DAC_Phase4

Team Members:

1. **Nithyasri V G**
2. **Sakshi Rajesh Bhavsar**
3. **Mathew P**
4. **Sarfraj Ansari**

AIR QUALITY ANALYSIS IN TAMIL NADU

Project Definition:

The primary objective of this project is to conduct a comprehensive assessment and visualization of air quality data collected from monitoring stations situated across Tamil Nadu. Through this project, we aim to extract valuable insights into the prevailing patterns of air pollution, identify regions exhibiting elevated levels of pollution, and establish a predictive model capable of estimating RSPM/PM10 levels. This analysis will be based on the concentrations of SO2 and NO2.

Phase 4: Development Part 2

-Perform the air quality analysis and create visualizations.

-Calculate average SO2, NO2, and RSPM/PM10 levels across different monitoring stations, cities, or areas.

-Identify pollution trends and areas with high pollution levels.

-Create visualizations using data visualization libraries

Dataset Link: **https://tn.data.gov.in/resource/location-wise-daily-ambient-air-quality-tamil-nadu-year-2014**

Importing the initial data:

```
import numpy as np # linear algebra
import pandas as pd # data processing, CSV file I/O
import matplotlib.pyplot as plt
%matplotlib inline
import seaborn as sns
```

Updating the initial data:

```
import pandas as pd
new_df = pd.read_csv("air_quality_data.csv")
new_df
```

[118] Python

	Stn Code	Sampling Date	State	City/Town/Village/Area	Location of Monitoring Station	Agency	Type of Location	SO2	NO2	RSPM/PM10	PM 2.5
0	38	01-02-2014	Tamil Nadu	Chennai	Kathivakkam, Municipal Kalyana Mandapam, Chennai	Tamilnadu State Pollution Control Board	Industrial Area	11.0	17.0	55.0	NaN
1	38	01-07-2014	Tamil Nadu	Chennai	Kathivakkam, Municipal Kalyana Mandapam, Chennai	Tamilnadu State Pollution Control Board	Industrial Area	13.0	17.0	45.0	NaN
2	38	21-01-2014	Tamil Nadu	Chennai	Kathivakkam, Municipal Kalyana Mandapam, Chennai	Tamilnadu State Pollution Control Board	Industrial Area	12.0	18.0	50.0	NaN
3	38	23-01-2014	Tamil Nadu	Chennai	Kathivakkam, Municipal Kalyana Mandapam, Chennai	Tamilnadu State Pollution Control Board	Industrial Area	15.0	16.0	46.0	NaN
4	38	28-01-2014	Tamil Nadu	Chennai	Kathivakkam, Municipal Kalyana Mandapam, Chennai	Tamilnadu State Pollution Control Board	Industrial Area	13.0	14.0	42.0	NaN
...	...	...	...	...	...	...	...	...	...	...	...
2874	773	12-03-2014	Tamil Nadu	Trichy	Central Bus Stand, Trichy	Tamilnadu State Pollution Control Board	Residential, Rural and other Areas	15.0	18.0	102.0	NaN
2875	773	12-10-2014	Tamil Nadu	Trichy	Central Bus Stand, Trichy	Tamilnadu State Pollution Control Board	Residential, Rural and other Areas	12.0	14.0	91.0	NaN
2876	773	17-12-2014	Tamil Nadu	Trichy	Central Bus Stand, Trichy	Tamilnadu State Pollution Control Board	Residential, Rural and other Areas	19.0	22.0	100.0	NaN
2877	773	24-12-2014	Tamil Nadu	Trichy	Central Bus Stand, Trichy	Tamilnadu State Pollution Control Board	Residential, Rural and other Areas	15.0	17.0	95.0	NaN
2878	773	31-12-2014	Tamil Nadu	Trichy	Central Bus Stand, Trichy	Tamilnadu State Pollution Control Board	Residential, Rural and other Areas	14.0	16.0	94.0	NaN

2879 rows × 11 columns

Feature Engineering:

```
Feature Engineering
```

markdown

```
Feature engineering is a subset of data preprocessing in the context of data analysis and machine learning. -Removing unnecessary datas -Data Cleaning -Data Transformation -Data Reduction -Dealing with Imbalanced Data
```

markdown

```
import pandas as pd
df = pd.read_csv("air_quality_data.csv")
df.drop(labels=['Stn Code','Location of Monitoring Station','Agency'], axis = 1, inplace = True)
new_df=df.drop(labels=['PM 2.5'], axis = 1, inplace = True)
df['SO2'].fillna(0, inplace=True)
df['NO2'].fillna(0, inplace=True)
df['RSPM/PM10'].fillna(0, inplace=True)
# Convert the column to integer
df['SO2'] = df['SO2'].astype(int)
df['NO2'] = df['NO2'].astype(int)
df['RSPM/PM10'] =df['RSPM/PM10'].astype(int)
df.to_csv("air_quality_data.csv", index=False)
df.info()
```

[119]

Python

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 2879 entries, 0 to 2878
Data columns (total 7 columns):
 #   Column                  Non-Null Count  Dtype 
---  ------                  --------------  ----- 
 0   Sampling Date           2879 non-null   object
 1   State                   2879 non-null   object
 2   City/Town/Village/Area  2879 non-null   object
 3   Type of Location        2879 non-null   object
 4   SO2                     2879 non-null   int32 
 5   NO2                     2879 non-null   int32 
 6   RSPM/PM10               2879 non-null   int32 
dtypes: int32(3), object(4)
memory usage: 123.8+ KB
```

```
df.isnull().sum()
```

[24]

```
Sampling Date             0
State                     0
City/Town/Village/Area    0
Type of Location          0
SO2                       0
NO2                       0
RSPM/PM10                 0
dtype: int64
```

```
df = pd.read_csv("air_quality_data.csv")
df['SO2'].fillna(0, inplace=True)
df['NO2'].fillna(0, inplace=True)
df['RSPM/PM10'].fillna(0, inplace=True)
```

[83]

```
new_df = pd.read_csv("air_quality_data.csv")
column_name = 'SO2'
column_name = 'NO2'
column_name = 'RSPM/PM10'
# Count NaN values
nan_count = df[column_name].isna().sum()
# Get indices of NaN values
nan_indices = df.index[df[column_name].isna()].tolist()

print(f"Number of NaN values in column '{column_name}': {nan_count}")
print(f"Indices of NaN values: {nan_indices}")
```

[84]

```
Number of NaN values in column 'RSPM/PM10': 0
Indices of NaN values: []
```

```
[85]
df['SO2'].unique()
```

```
array([11, 13, 12, 15, 14, 10, 16, 19,  9, 20, 17, 18, 25, 21, 23, 26, 24,
       32, 27, 30, 22,  0,  8, 31, 28, 29,  6, 49,  3,  7,  5,  2,  4, 39],
      dtype=int64)
```

```
[28]
df['NO2'].unique()
```

```
array([17, 18, 16, 14, 19, 15, 13, 20, 12, 10, 11, 23, 30, 29, 25, 26, 27,
       34, 35, 32, 22, 24, 21, 28, 31, 33,  0, 38, 41, 47, 36, 42,  9, 44,
        8, 50, 46, 37, 43, 39, 49, 40, 45,  7, 48, 55, 53, 52,  6,  5, 54,
       51, 71, 69], dtype=int64)
```

```
[29]
df['RSPM/PM10'].unique()
```

```
array([ 55,  45,  50,  46,  42,  43,  51,  48,  32,  29,  17,  44,  25,
        41,  54,  62,  66,  40,  56,  49,  63, 119,  61,  52,  53,  57,
        39,  47,  35,  58,  74,  34,  60,  38, 104,  65,  33,  68,  59,
        64, 105,  36,  28,  26,  37,  27,  31,  30,  71,  24, 100, 142,
       115,  83,  96,  82,  84, 122, 107,  92,  90, 102,  81,  89, 120,
        99,  67, 103,  95, 106, 124,  91,  98, 146, 111, 117,  93, 163,
       118,  79,  77, 128, 147, 153, 121, 114, 109, 101, 148, 131, 125,
       108, 116, 110, 129, 211, 202, 143,  76,  94,  69,  86,  72,  75,
        87, 157,  88,  78, 130, 138,  73,  70,  80,  85,  97, 112, 133,
       150, 123,  22, 149, 113, 139, 175, 166, 181,  20,  21,  15,  14,
        16,  12,  13,  19,  18,  23, 164, 155,   0, 199, 152, 160, 182,
       132, 127, 225, 134, 233, 176, 170, 141, 197, 206, 126, 269, 159,
       140, 136, 204, 174, 154, 198, 180, 135, 145, 165, 151, 240, 262,
       238], dtype=int64)
```

Data Visualization:

```
Data Visualization
```

```
df
```

[86]

	Sampling Date	State	City/Town/Village/Area	Type of Location	SO2	NO2	RSPM/PM10
0	01-02-2014	Tamil Nadu	Chennai	Industrial Area	11	17	55
1	01-07-2014	Tamil Nadu	Chennai	Industrial Area	13	17	45
2	21-01-2014	Tamil Nadu	Chennai	Industrial Area	12	18	50
3	23-01-2014	Tamil Nadu	Chennai	Industrial Area	15	16	46
4	28-01-2014	Tamil Nadu	Chennai	Industrial Area	13	14	42
...	...	...	...	...	...	...	...
2874	12-03-2014	Tamil Nadu	Trichy	Residential, Rural and other Areas	15	18	102
2875	12-10-2014	Tamil Nadu	Trichy	Residential, Rural and other Areas	12	14	91
2876	17-12-2014	Tamil Nadu	Trichy	Residential, Rural and other Areas	19	22	100
2877	24-12-2014	Tamil Nadu	Trichy	Residential, Rural and other Areas	15	17	95
2878	31-12-2014	Tamil Nadu	Trichy	Residential, Rural and other Areas	14	16	94

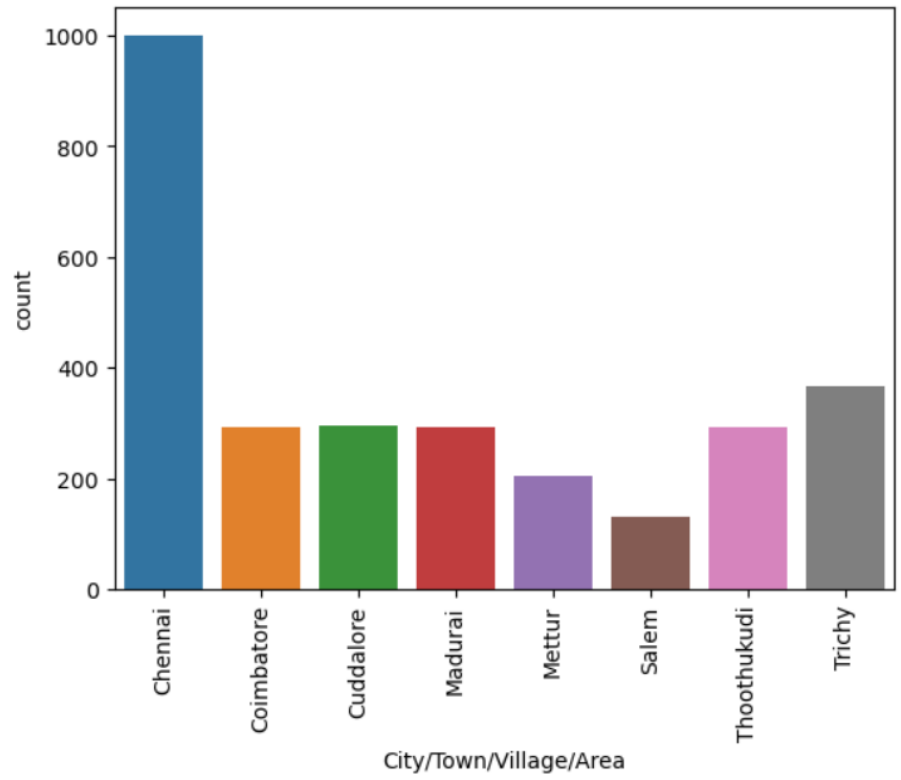
2879 rows × 7 columns

```
print(df["City/Town/Village/Area"])
```

[31]

```
0       Chennai
1       Chennai
2       Chennai
3       Chennai
4       Chennai
        ...
2874     Trichy
2875     Trichy
2876     Trichy
2877     Trichy
2878     Trichy
Name: City/Town/Village/Area, Length: 2879, dtype: object
```

```
datacount =sns.countplot(x ="City/Town/Village/Area",data = df);
datacount.set_xticklabels(datacount.get_xticklabels(), rotation=90);
```



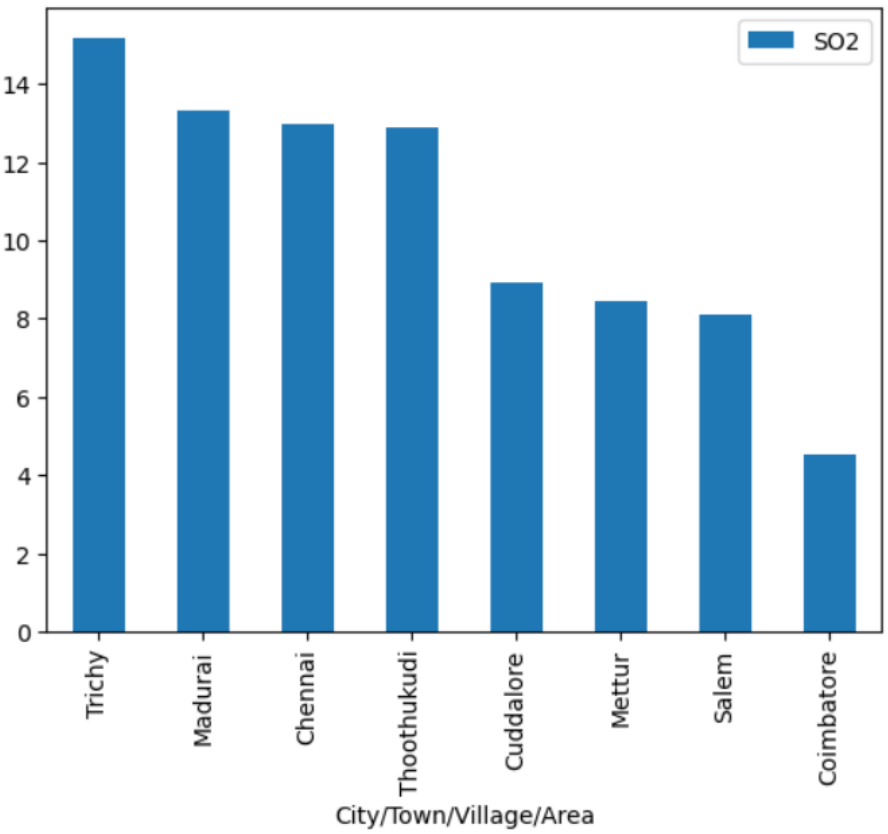

```
loc = pd.pivot_table(df, values=['SO2','NO2','RSPM/PM10'],index='City/Town/Village/Area')
# Aggfunc: default-np.mean()
loc
```

City/Town/Village/Area	NO2	RSPM/PM10	SO2
Chennai	21.978000	58.998000	12.975000
Coimbatore	25.238908	48.713311	4.525597
Cuddalore	19.577703	61.881757	8.905405
Madurai	25.768707	45.724490	13.319728
Mettur	23.185366	52.721951	8.429268
Salem	28.664122	62.954198	8.114504
Thoothukudi	18.385666	83.174061	12.901024
Trichy	18.542234	85.054496	15.168937

```
maxso2 = loc.sort_values(by='SO2',ascending=False)
maxso2.loc[:,['SO2']].head(10).plot(kind='bar')
# Based on average values
```

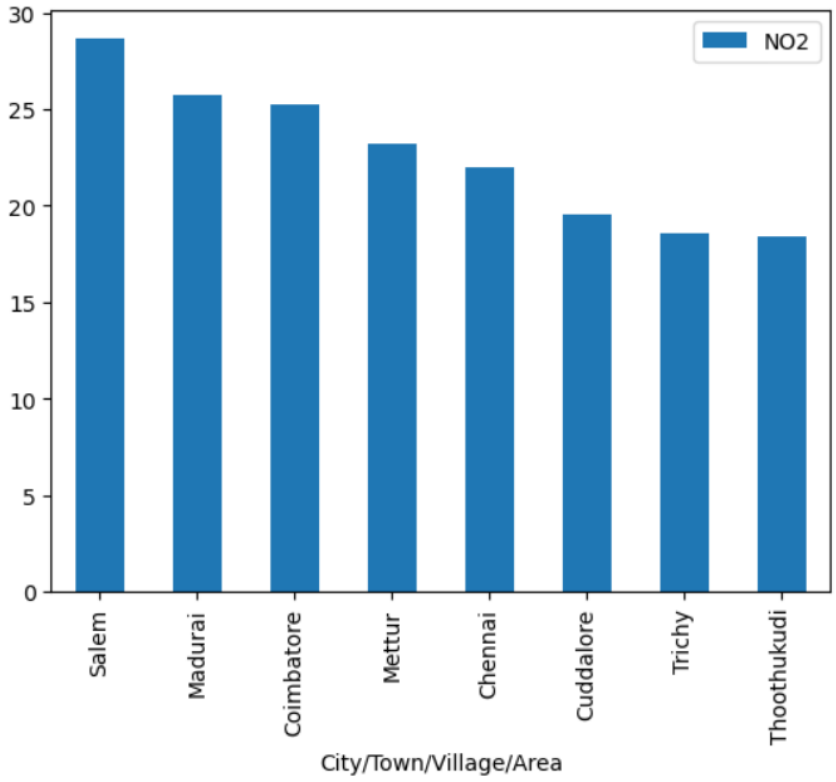
[88]

```
<Axes: xlabel='City/Town/Village/Area'>
```



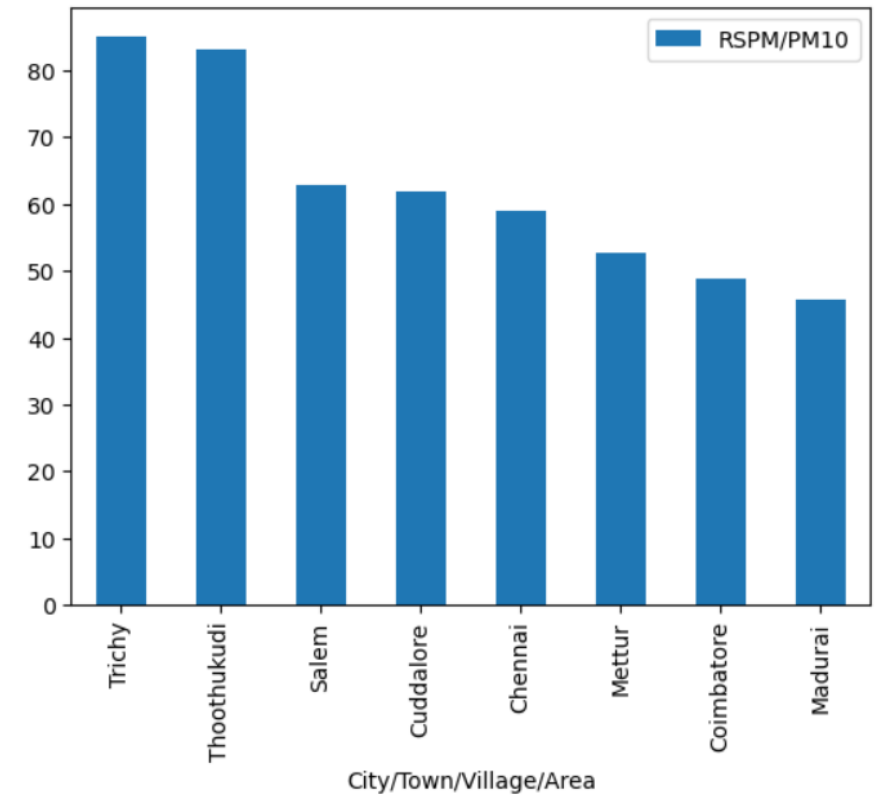
```
maxno2 = loc.sort_values(by='NO2',ascending=False)
maxno2.loc[:,['NO2']].head(10).plot(kind='bar')
```

```
<Axes: xlabel='City/Town/Village/Area'>
```



```
maxrspm_pm10 = loc.sort_values(by='RSPM/PM10',ascending=False)
maxrspm_pm10.loc[:,['RSPM/PM10']].head(10).plot(kind='bar')
```

```
<Axes: xlabel='City/Town/Village/Area'>
```



Calculating AQI-Air Quality Index:

```
# Sulfur Dioxide (SO2):
# A pungent gas released by volcanic eruptions and industrial processes.
def calculate_si(SO2):
    si=0
    if (SO2<=40):
     si= "s1"
    if (SO2>40 and SO2<=80):
     si= "s2"
    if (SO2>80 and SO2<=380):
     si= "s3"
    if (SO2>380 and SO2<=800):
     si= "s4"
    if (SO2>800 and SO2<=1600):
     si= "s5"
    if (SO2>1600):
     si= "s6"
    return si
df['si']=df['SO2'].apply(calculate_si)
ds= df[['SO2','si']]
ds.tail()
```

[120]

	SO2	si
2874	15	s1
2875	12	s1
2876	19	s1
2877	15	s1
2878	14	s1

```
# Nitrogen Dioxide (NO2): A reddish-brown gas that is a byproduct of burning fossil fuels.
def calculate_ni(NO2):
    ni=0
    if (NO2<=40):
     ni= "n1"
    if (NO2>40 and NO2<=80):
     ni= "n2"
    if (NO2>80 and NO2<=180):
     ni= "n3"
    if (NO2>180 and NO2<=280):
     ni= "n4"
    if (NO2>280 and NO2<=400):
     ni= "n5"
    if (NO2>400):
     ni= "n6"
    return ni
df['ni']=df['NO2'].apply(calculate_ni)
dn= df[['NO2','ni']]
dn.tail()
```

[121]

	NO2	ni
2874	18	n1
2875	14	n1
2876	22	n1
2877	17	n1
2878	16	n1

```
# RSPM (Respirable Suspended Particulate Matter)
# PM10 (Particulate Matter with a diameter of 10 micrometers or less)
def calculate_spi(rspm_pm10):
    spi=0
    if (rspm_pm10<=40):
      spi= "sp1"
    if (rspm_pm10>40 and rspm_pm10<=80):
      spi= "sp2"
    if (rspm_pm10>80 and rspm_pm10<=180):
      spi= "sp3"
    if (rspm_pm10>180 and rspm_pm10<=280):
      spi= "sp4"
    if (rspm_pm10>280 and rspm_pm10<=400):
      spi= "sp5"
    if (rspm_pm10>400):
      spi= "sp6"
    return  spi
df['spi']=df['RSPM/PM10'].apply(calculate_spi)
dsp= df[['RSPM/PM10','spi']]
dsp.tail()
```

[122]

	RSPM/PM10	spi
2874	102	sp3
2875	91	sp3
2876	100	sp3
2877	95	sp3
2878	94	sp3

```
df.head()
```

[35]

	Sampling Date	State	City/Town/Village/Area	Type of Location	SO2	NO2	RSPM/PM10	si	ni	spi
0	01-02-2014	Tamil Nadu	Chennai	Industrial Area	11	17	55	s1	n1	sp2
1	01-07-2014	Tamil Nadu	Chennai	Industrial Area	13	17	45	s1	n1	sp2
2	21-01-2014	Tamil Nadu	Chennai	Industrial Area	12	18	50	s1	n1	sp2
3	23-01-2014	Tamil Nadu	Chennai	Industrial Area	15	16	46	s1	n1	sp2
4	28-01-2014	Tamil Nadu	Chennai	Industrial Area	13	14	42	s1	n1	sp2

```
# AQI
def calculate_aqi(si,ni,spi):
    aqi=0
    if(si>ni and si>spi):
     aqi=si
    if (spi>ni and spi>si):
     aqi=spi
    if(ni>si and ni>spi):
     aqi= ni
    return aqi
df['AQI']=df.apply(lambda x:calculate_aqi(x['SO2'],x['NO2'],x['RSPM/PM10']),axis=1)
```

5]

```
df.head()
```

4]

	Sampling Date	State	City/Town/Village/Area	Type of Location	SO2	NO2	RSPM/PM10	si	ni	spi	AQI
0	01-02-2014	Tamil Nadu	Chennai	Industrial Area	11	17	55	s1	n1	sp2	55
1	01-07-2014	Tamil Nadu	Chennai	Industrial Area	13	17	45	s1	n1	sp2	45
2	21-01-2014	Tamil Nadu	Chennai	Industrial Area	12	18	50	s1	n1	sp2	50
3	23-01-2014	Tamil Nadu	Chennai	Industrial Area	15	16	46	s1	n1	sp2	46
4	28-01-2014	Tamil Nadu	Chennai	Industrial Area	13	14	42	s1	n1	sp2	42

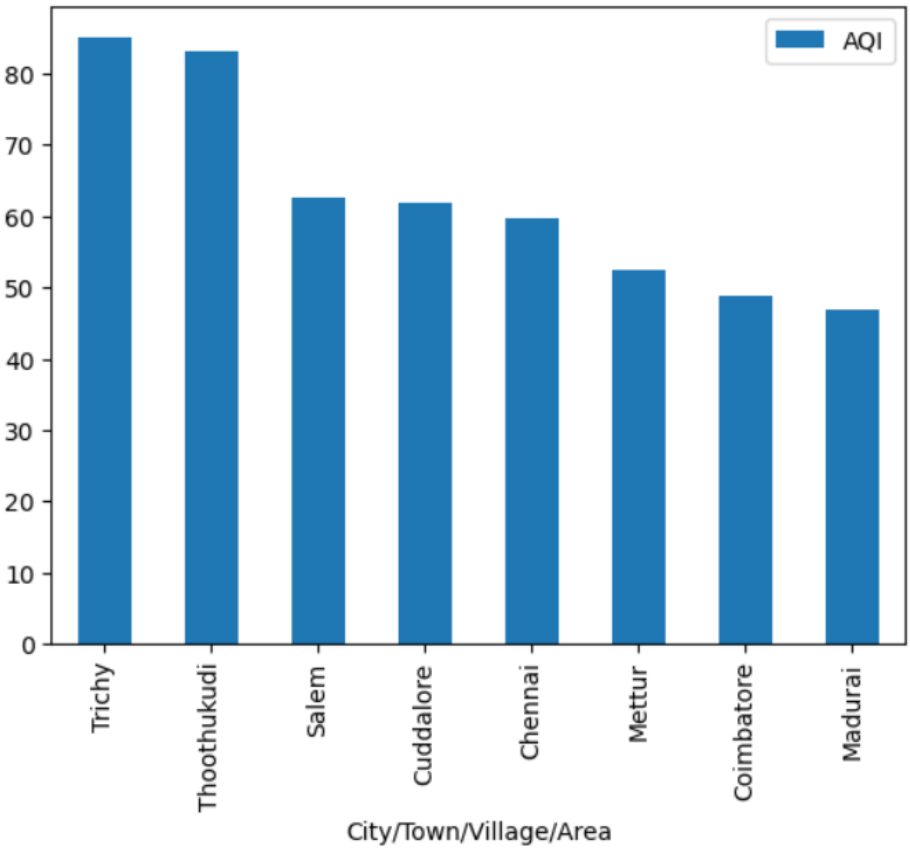
```
aq_wise = pd.pivot_table(df, values=['AQI'],index='City/Town/Village/Area')
aq_wise
```

[125]

City/Town/Village/Area	AQI
Chennai	59.691000
Coimbatore	48.914676
Cuddalore	61.885135
Madurai	46.945578
Mettur	52.541463
Salem	62.541985
Thoothukudi	83.252560
Trichy	85.054496

```
[126]
maxaqi = aq_wise.sort_values(by='AQI',ascending=False)
maxaqi.loc[:,['AQI']].head(37).plot(kind='bar')
```

```
<Axes: xlabel='City/Town/Village/Area'>
```



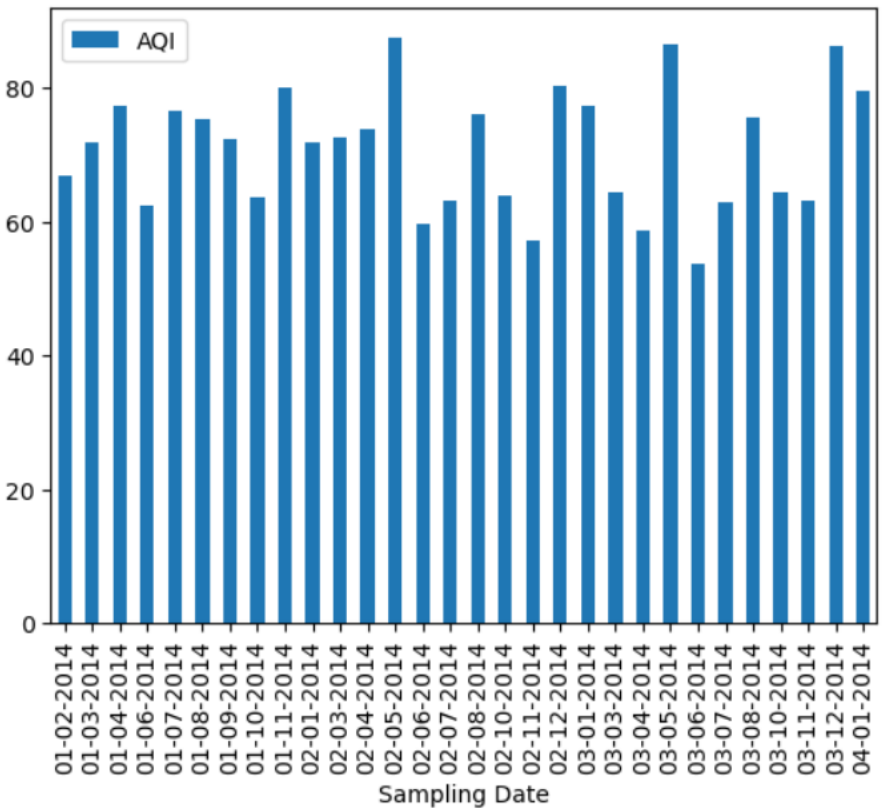
```
[127]
date_wise = pd.pivot_table(df, values=['AQI'],index='Sampling Date')
date_wise
```

Sampling Date	AQI
01-02-2014	66.818182
01-03-2014	71.769231
01-04-2014	77.250000
01-06-2014	62.454545
01-07-2014	76.461538
...	...
31-03-2014	62.000000
31-05-2014	50.000000
31-07-2014	58.076923
31-10-2014	59.700000
31-12-2014	57.300000

302 rows × 1 columns

```
[128]  date_wise.loc[:,['AQI']].head(30).plot(kind='bar')
```

```
<Axes: xlabel='Sampling Date'>
```

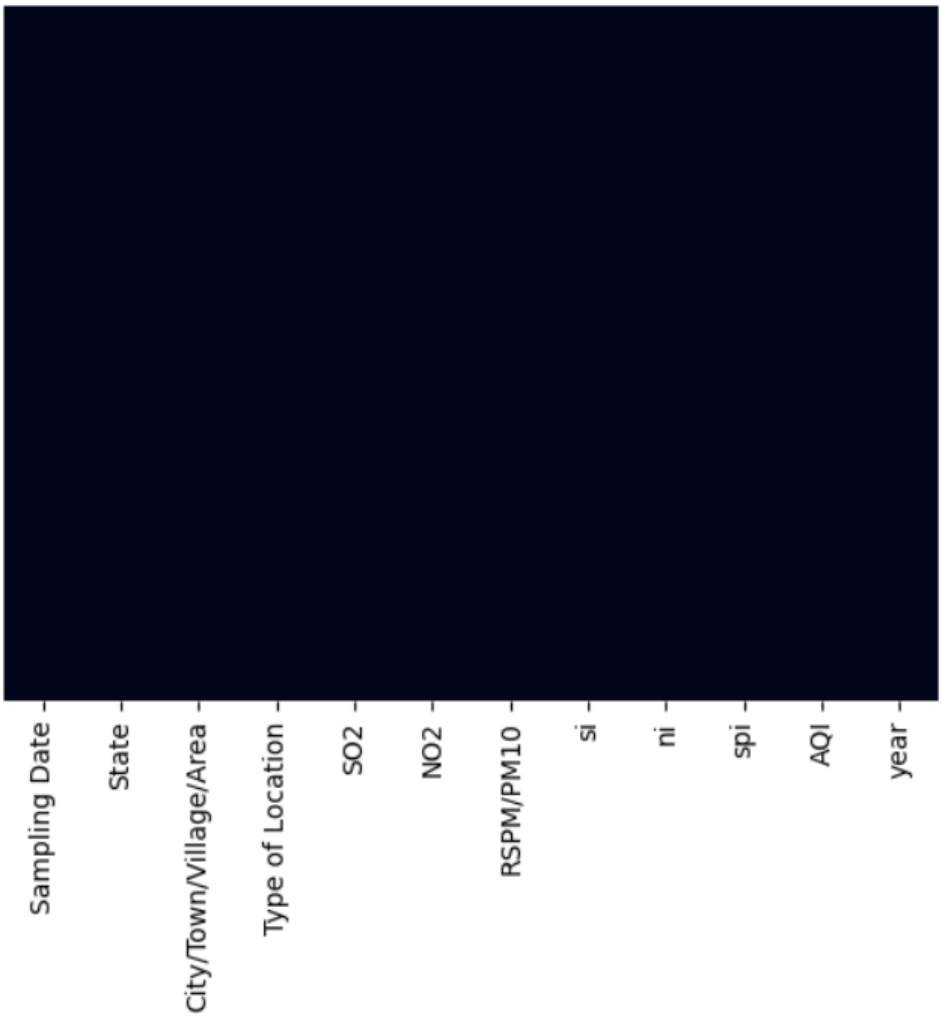


Training Data:

```
dum1 = pd.get_dummies(df['Type of Location'])
dum2 = pd.get_dummies(df['City/Town/Village/Area'])
df['year'] = df['Sampling Date']
```

```
[132]  sns.heatmap(df.isna(), yticklabels=False, cbar=False)
```

```
<Axes: >
```



```
df = pd.concat([df, dum1, dum2], axis = 1)
df.head()
```

[98]

Python

	Sampling Date	State	City/Town/Village/Area	Type of Location	SO2	NO2	RSPM/PM10	si	ni	spi	...	Industrial Area	Residential, Rural and other Areas	Chennai	Coimbatore	Cuddalore	Madurai	Mettur	Salem	Thoothukudi
0	01-02-2014	Tamil Nadu	Chennai	Industrial Area	11	17	55	s1	n1	sp2	...	True	False	True	False	False	False	False	False	False
1	01-07-2014	Tamil Nadu	Chennai	Industrial Area	13	17	45	s1	n1	sp2	...	True	False	True	False	False	False	False	False	False
2	21-01-2014	Tamil Nadu	Chennai	Industrial Area	12	18	50	s1	n1	sp2	...	True	False	True	False	False	False	False	False	False
3	23-01-2014	Tamil Nadu	Chennai	Industrial Area	15	16	46	s1	n1	sp2	...	True	False	True	False	False	False	False	False	False
4	28-01-2014	Tamil Nadu	Chennai	Industrial Area	13	14	42	s1	n1	sp2	...	True	False	True	False	False	False	False	False	False

5 rows × 22 columns

```
df.info()
```

[130]

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 2879 entries, 0 to 2878
Data columns (total 11 columns):
 #   Column                 Non-Null Count  Dtype
---  ------                 --------------  -----
 0   Sampling Date          2879 non-null   object
 1   State                  2879 non-null   object
 2   City/Town/Village/Area 2879 non-null   object
 3   Type of Location       2879 non-null   object
 4   SO2                    2879 non-null   int32
 5   NO2                    2879 non-null   int32
 6   RSPM/PM10              2879 non-null   int32
 7   si                     2879 non-null   object
 8   ni                     2879 non-null   object
 9   spi                    2879 non-null   object
 10  AQI                    2879 non-null   int64
dtypes: int32(3), int64(1), object(7)
memory usage: 213.8+ KB
```

```
df.corr()
```

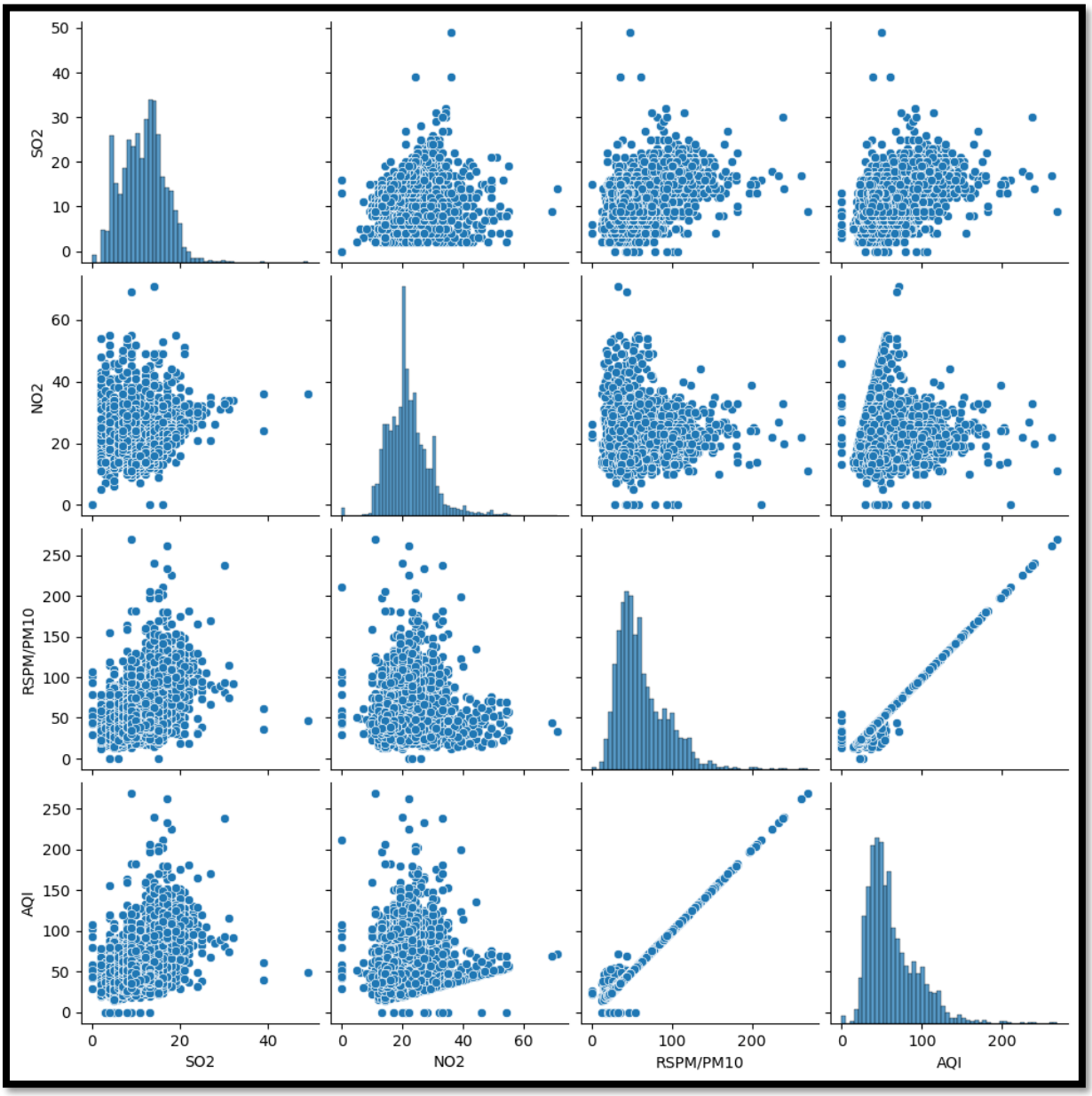
[164]

	SO2	NO2	RSPM/PM10	AQI
SO2	1.000000	0.100779	0.440141	0.441655
NO2	0.100779	1.000000	0.060369	0.093855
RSPM/PM10	0.440141	0.060369	1.000000	0.993675
AQI	0.441655	0.093855	0.993675	1.000000

```
X = df.iloc[:, :-1]
y = df.iloc[:, -1]
```

```
sns.pairplot(df)
```

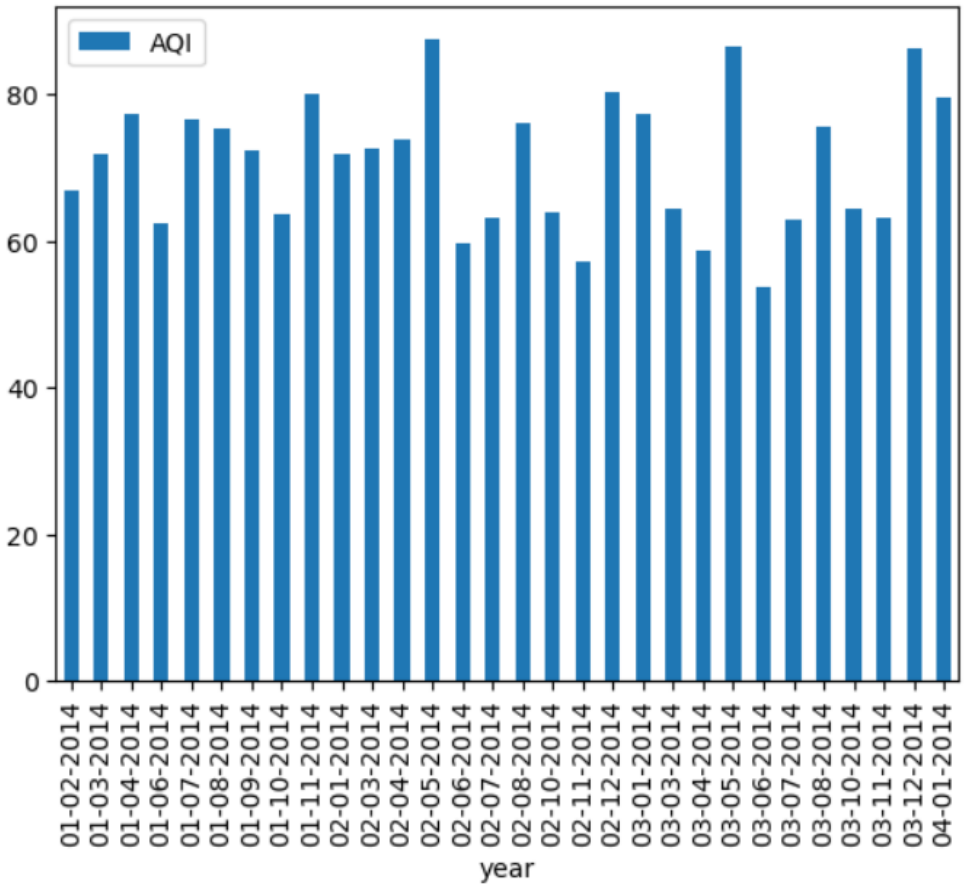
```
c:\ProgramData\anaconda3\Lib\site-packages\seaborn\axisgrid.py:118: UserWarning: The figure layout has changed to tight
  self._figure.tight_layout(*args, **kwargs)

<seaborn.axisgrid.PairGrid at 0x181191f7d10>
```



```
yr_wise = pd.pivot_table(df, values=['AQI'],index='year')
yr_wise.loc[:,['AQI']].head(30).plot(kind='bar')
```

[33]

```
<Axes: xlabel='year'>
```




```
X_train,X_test,y_train,y_test= train_test_split(X,y,test_size=0.30,random_state=25)
```

Training Data:

Model fittings

Simple Linear Regression

```
[161]
from sklearn.linear_model import LinearRegression
```

```
[167]
from sklearn.model_selection import train_test_split
X=df[['SO2','NO2','RSPM/PM10']]
y=df[['AQI']]
x_train,x_test,y_train,y_test = train_test_split(X,y,test_size=0.30,random_state=25)
```

```
[173]
from sklearn.linear_model import LinearRegression
lin_mod = LinearRegression()
lin_mod.fit(x_train, y_train)
lin_mod.score(x_train, y_train )
```

```
0.9887589817873087
```