In [23]:
```
#importing necessary libraries
import pandas as pd
import numpy as np
import seaborn as sns
import matplotlib.pyplot as plt
import warnings
warnings.filterwarnings("ignore")

from sklearn.preprocessing import LabelEncoder
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LinearRegression
from sklearn.tree import DecisionTreeRegressor
from sklearn.ensemble import RandomForestRegressor
from sklearn import metrics
from sklearn.metrics import mean_absolute_error,mean_squared_error,r2_score
from sklearn.metrics import accuracy_score,confusion_matrix
```

In [24]:
```
df=pd.read_csv("D:\\air.csv")
df
```

Out[24]:

	Stn Code	Sampling Date	State	City/Town/Village/Area	M_Station	Agency	Type of Location	SO2
0	38	01-02-14	Tamil Nadu	Chennai	Kathivakkam, Municipal Kalyana Mandapam, Chennai	Tamilnadu State Pollution Control Board	Industrial Area	11.(
1	38	01-07-14	Tamil Nadu	Chennai	Kathivakkam, Municipal Kalyana Mandapam, Chennai	Tamilnadu State Pollution Control Board	Industrial Area	13.(
2	38	21-01-14	Tamil Nadu	Chennai	Kathivakkam, Municipal Kalyana Mandapam, Chennai	Tamilnadu State Pollution Control Board	Industrial Area	12.(
3	38	23-01-14	Tamil Nadu	Chennai	Kathivakkam, Municipal Kalyana Mandapam, Chennai	Tamilnadu State Pollution Control Board	Industrial Area	15.(
4	38	28-01-14	Tamil Nadu	Chennai	Kathivakkam, Municipal Kalyana Mandapam, Chennai	Tamilnadu State Pollution Control Board	Industrial Area	13.(
...	...	...	...	...	...	...	...	..
2874	773	12-03-14	Tamil Nadu	Trichy	Central Bus Stand, Trichy	Tamilnadu State Pollution Control Board	Residential, Rural and other Areas	15.(
2875	773	12-10-14	Tamil Nadu	Trichy	Central Bus Stand, Trichy	Tamilnadu State Pollution Control Board	Residential, Rural and other Areas	12.(
2876	773	17-12-14	Tamil Nadu	Trichy	Central Bus Stand, Trichy	Tamilnadu State Pollution Control Board	Residential, Rural and other Areas	19.(
2877	773	24-12-14	Tamil Nadu	Trichy	Central Bus Stand, Trichy	Tamilnadu State Pollution Control Board	Residential, Rural and other Areas	15.(
2878	773	31-12-14	Tamil Nadu	Trichy	Central Bus Stand, Trichy	Tamilnadu State Pollution Control Board	Residential, Rural and other Areas	14.(

2879 rows × 11 columns

In [25]:
```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 2879 entries, 0 to 2878
Data columns (total 11 columns):
 #   Column                  Non-Null Count  Dtype  
---  ------                  --------------  -----  
 0   Stn Code                2879 non-null   int64  
 1   Sampling Date           2879 non-null   object 
 2   State                   2879 non-null   object 
 3   City/Town/Village/Area  2879 non-null   object 
 4   M_Station               2879 non-null   object 
 5   Agency                  2879 non-null   object 
 6   Type of Location        2879 non-null   object 
 7   SO2                     2868 non-null   float64
 8   NO2                     2866 non-null   float64
 9   RSPM                    2875 non-null   float64
 10  PM                      0 non-null      float64
dtypes: float64(4), int64(1), object(6)
memory usage: 247.5+ KB
```

In [26]:
```
df.shape
```

Out[26]: (2879, 11)

In [27]:
```
df.head()
```

Out[27]:

	Stn Code	Sampling Date	State	City/Town/Village/Area	M_Station	Agency	Type of Location	SO2	NO
0	38	01-02-14	Tamil Nadu	Chennai	Kathivakkam, Municipal Kalyana Mandapam, Chennai	Tamilnadu State Pollution Control Board	Industrial Area	11.0	17.
1	38	01-07-14	Tamil Nadu	Chennai	Kathivakkam, Municipal Kalyana Mandapam, Chennai	Tamilnadu State Pollution Control Board	Industrial Area	13.0	17.
2	38	21-01-14	Tamil Nadu	Chennai	Kathivakkam, Municipal Kalyana Mandapam, Chennai	Tamilnadu State Pollution Control Board	Industrial Area	12.0	18.
3	38	23-01-14	Tamil Nadu	Chennai	Kathivakkam, Municipal Kalyana Mandapam, Chennai	Tamilnadu State Pollution Control Board	Industrial Area	15.0	16.
4	38	28-01-14	Tamil Nadu	Chennai	Kathivakkam, Municipal Kalyana Mandapam, Chennai	Tamilnadu State Pollution Control Board	Industrial Area	13.0	14.

In [28]:
```
df.describe()
```

Out[28]:

	Stn Code	SO2	NO2	RSPM	PM
count	2879.000000	2868.000000	2866.000000	2875.000000	0.0
mean	475.750261	11.503138	22.136776	62.494261	NaN
std	277.675577	5.051702	7.128694	31.368745	NaN
min	38.000000	2.000000	5.000000	12.000000	NaN
25%	238.000000	8.000000	17.000000	41.000000	NaN
50%	366.000000	12.000000	22.000000	55.000000	NaN
75%	764.000000	15.000000	25.000000	78.000000	NaN
max	773.000000	49.000000	71.000000	269.000000	NaN

In [29]:
```
df.nunique()
```

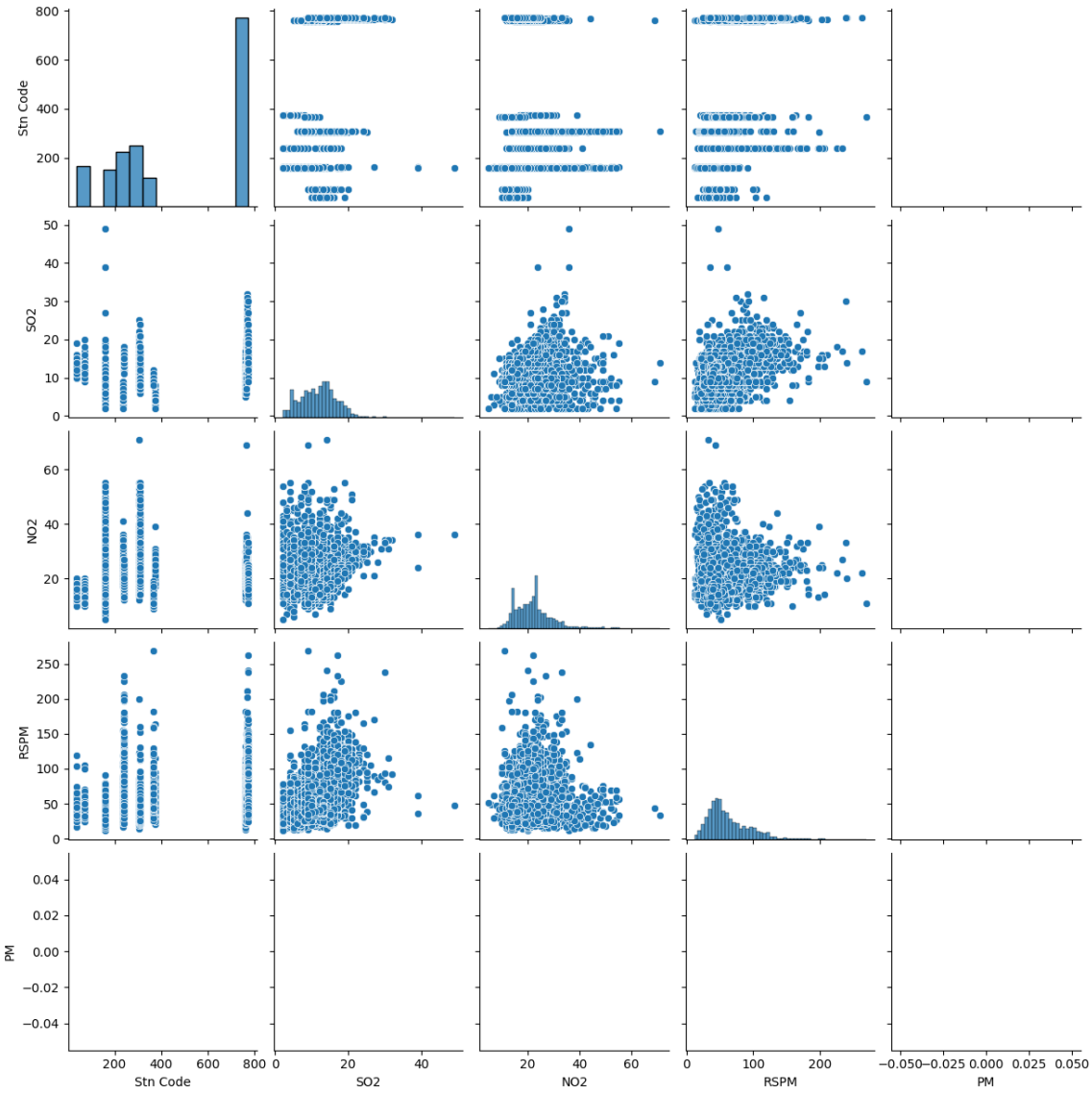
Out[29]:
```
Stn Code                   30
Sampling Date             302
State                       1
City/Town/Village/Area      8
M_Station                  30
Agency                      2
Type of Location            2
SO2                        33
NO2                        53
RSPM                      169
PM                          0
dtype: int64
```

In [30]:
```
df.columns
```

Out[30]:
```
Index(['Stn Code', 'Sampling Date', 'State', 'City/Town/Village/Area',
       'M_Station', 'Agency', 'Type of Location', 'SO2', 'NO2', 'RSPM', 'P
M'],
      dtype='object')
```

In [31]:
```
sns.pairplot(data=df)
```

Out[31]:
```
<seaborn.axisgrid.PairGrid at 0x21f54684890>
```



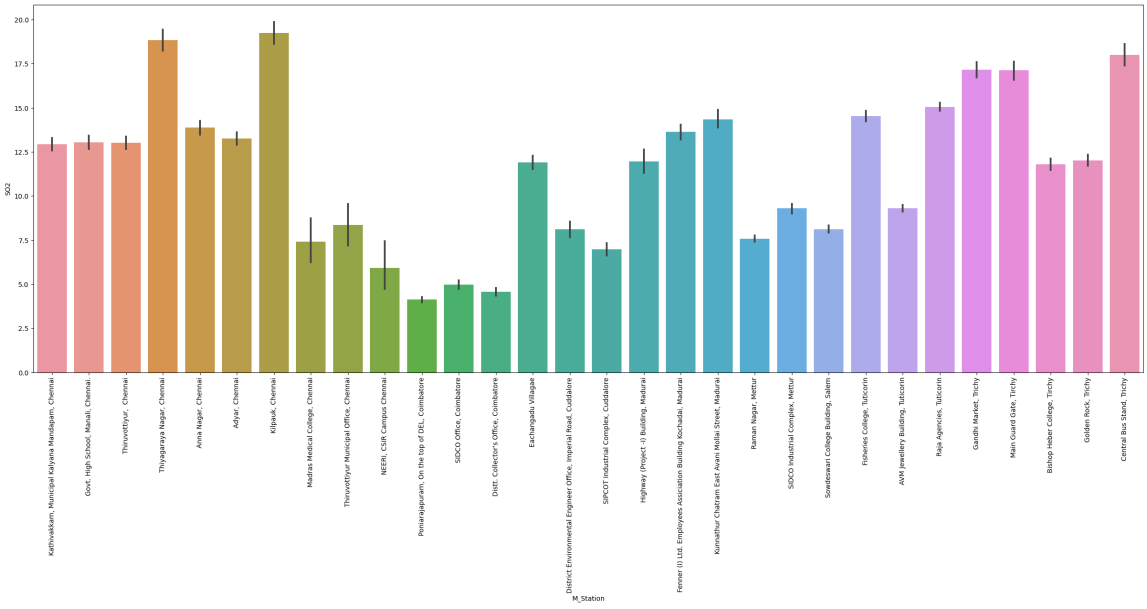
In [32]:
```
df['M_Station'].value_counts()
```

Out[32]:
```
Sowdeswari College Building, Salem                                  131
Adyar, Chennai                                                      116
Kilpauk, Chennai                                                    116
Thiyagaraya Nagar, Chennai                                          113
Anna Nagar, Chennai                                                 112
Raman Nagar, Mettur                                                 103
Poniarajapuram, On the top of DEL, Coimbatore                       103
Raja Agencies, Tuticorin                                            102
SIDCO Industrial Complex, Mettur                                    102
Fenner (I) Ltd. Employees Assiciation Building Kochadai, Madurai    101
SIPCOT Industrial Complex, Cuddalore                                 99
District Environmental Engineer Office, Imperial Road, Cuddalore     99
Eachangadu Villagae                                                  98
Kunnathur Chatram East Avani Mollai Street, Madurai                  97
SIDCO Office, Coimbatore                                             97
AVM Jewellery Building, Tuticorin                                    97
Highway (Project -I) Building, Madurai                               96
Thiruvottiyur,  Chennai                                              96
Fisheries College, Tuticorin                                         94
Kathivakkam, Municipal Kalyana Mandapam, Chennai                     94
Govt. High School, Manali, Chennai.                                  93
Distt. Collector's Office, Coimbatore                                93
NEERI, CSIR Campus Chennai                                           87
Thiruvottiyur Municipal Office, Chennai                              87
Madras Medical College, Chennai                                      86
Main Guard Gate, Tirchy                                              75
Central Bus Stand, Trichy                                            75
Gandhi Market, Trichy                                                74
Golden Rock, Trichy                                                  72
Bishop Heber College, Tirchy                                         71
Name: M_Station, dtype: int64
```

In [12]:
```
df['SO2'].value_counts()
```

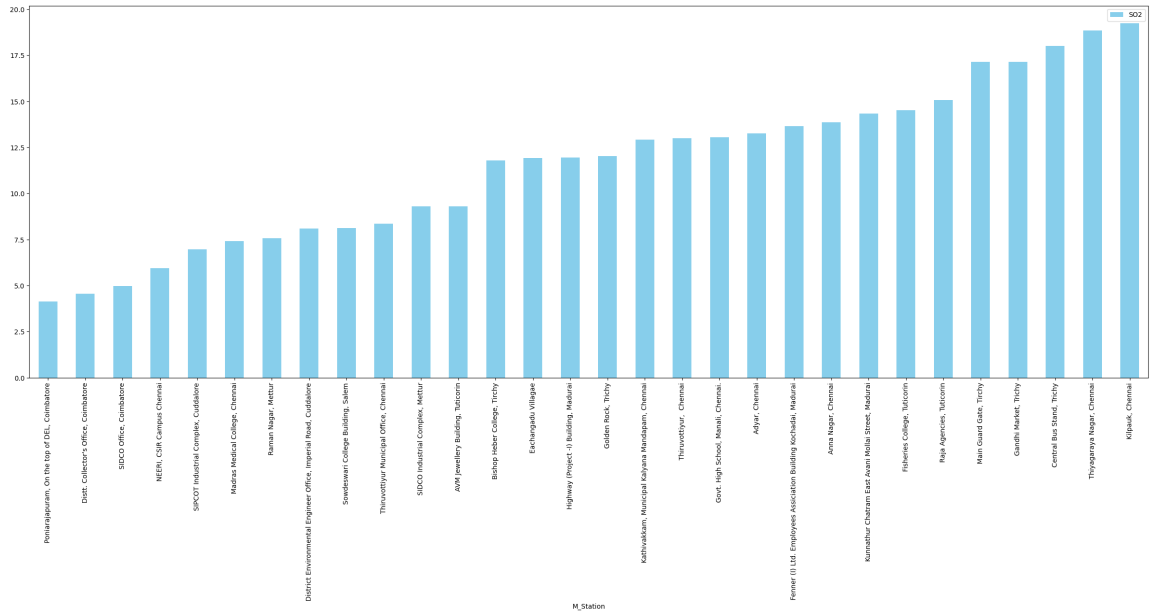
Out[12]:
```
13.0    249
14.0    248
12.0    220
10.0    198
15.0    196
4.0     194
8.0     187
9.0     177
11.0    160
7.0     145
16.0    131
5.0     121
17.0    115
18.0    110
6.0     104
19.0     80
20.0     60
2.0      49
3.0      48
21.0     23
22.0     17
24.0      7
23.0      7
25.0      6
30.0      3
27.0      3
31.0      2
49.0      2
39.0      2
28.0      1
29.0      1
32.0      1
26.0      1
Name: SO2, dtype: int64
```

In [33]:
```
plt.figure(figsize=(30, 10))
plt.xticks(rotation=90)
sns.barplot(x='M_Station',y='SO2',data=df);
```

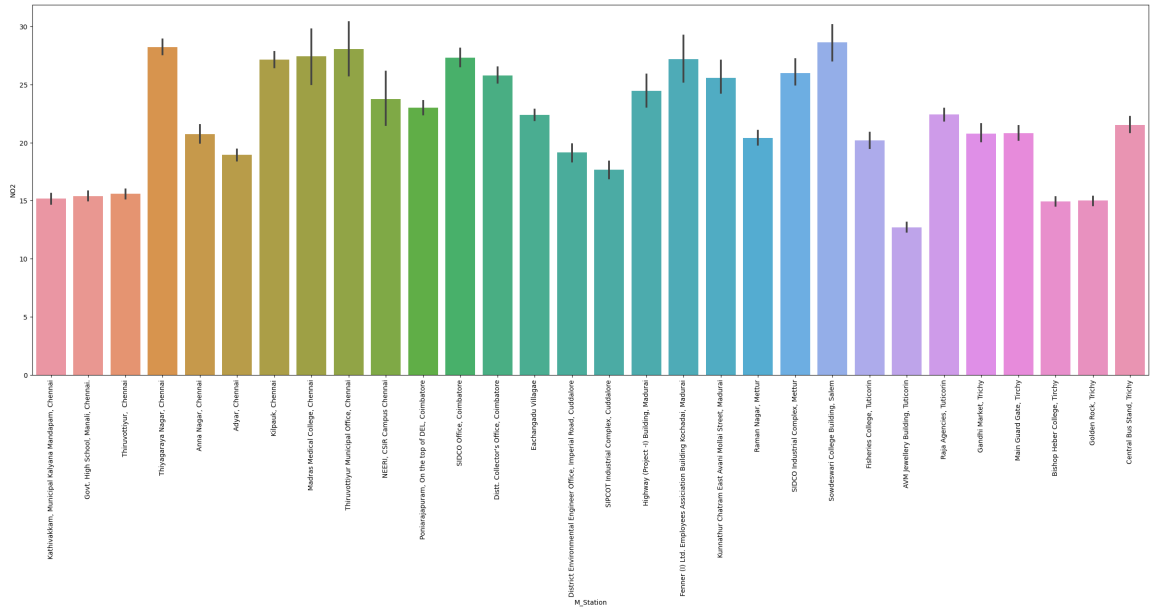


In [34]:
```
plt.rcParams['figure.figsize']=(30,10)
```

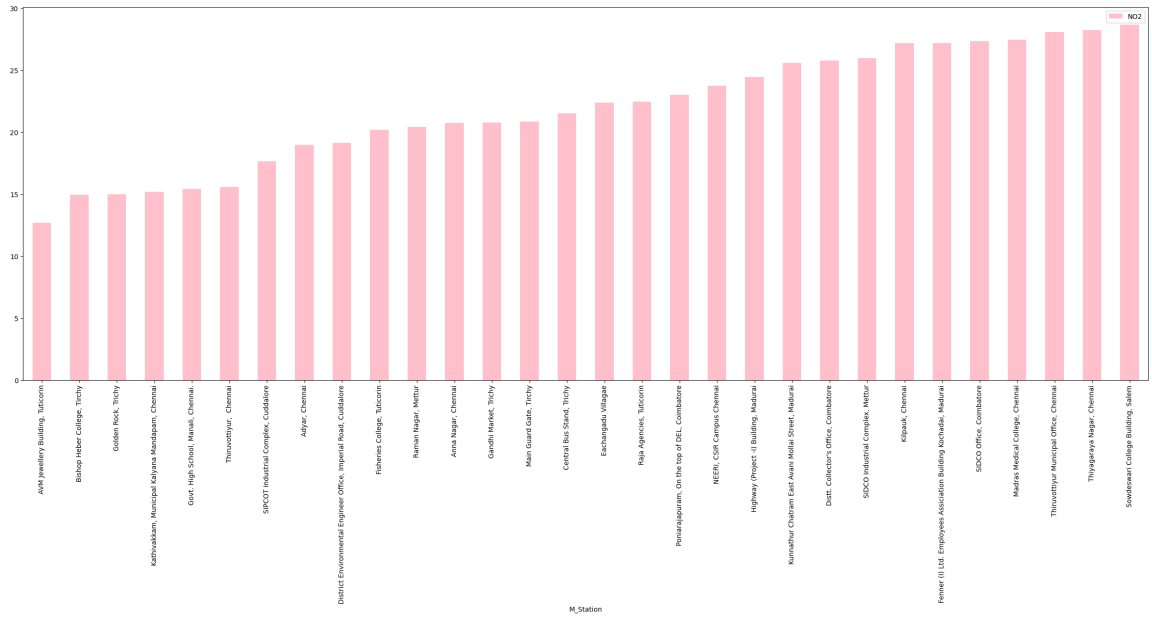
In [35]:
```
df[['SO2','M_Station']].groupby(["M_Station"]).mean().sort_values(by='SO2')
plt.show()
```



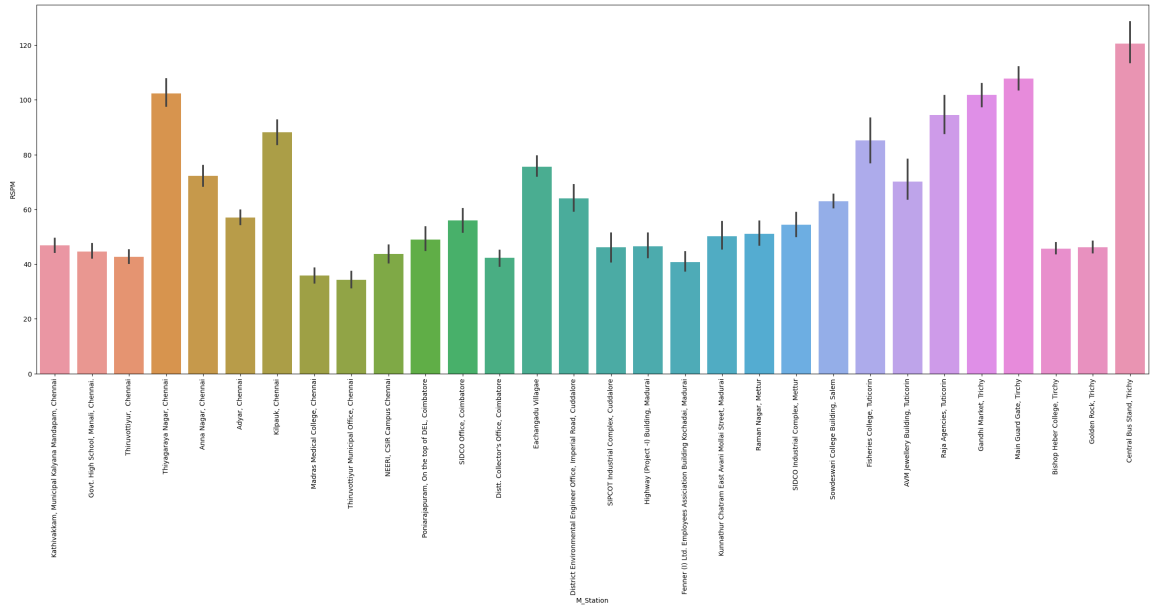
In [36]:
```
plt.figure(figsize=(30, 10))
plt.xticks(rotation=90)
sns.barplot(x='M_Station',y='NO2',data=df);
```



In [37]:
```
df[['NO2','M_Station']].groupby(["M_Station"]).mean().sort_values(by='NO2')
plt.show()
```



In [39]:
```
plt.figure(figsize=(30, 10))
plt.xticks(rotation=90)
sns.barplot(x='M_Station',y='RSPM',data=df);
```



In [40]:
```
df
```

Out[40]:

	Stn Code	Sampling Date	State	City/Town/Village/Area	M_Station	Agency	Type of Location	SO2
0	38	01-02-14	Tamil Nadu	Chennai	Kathivakkam, Municipal Kalyana Mandapam, Chennai	Tamilnadu State Pollution Control Board	Industrial Area	11.(
1	38	01-07-14	Tamil Nadu	Chennai	Kathivakkam, Municipal Kalyana Mandapam, Chennai	Tamilnadu State Pollution Control Board	Industrial Area	13.(
2	38	21-01-14	Tamil Nadu	Chennai	Kathivakkam, Municipal Kalyana Mandapam, Chennai	Tamilnadu State Pollution Control Board	Industrial Area	12.(
3	38	23-01-14	Tamil Nadu	Chennai	Kathivakkam, Municipal Kalyana Mandapam, Chennai	Tamilnadu State Pollution Control Board	Industrial Area	15.(
4	38	28-01-14	Tamil Nadu	Chennai	Kathivakkam, Municipal Kalyana Mandapam, Chennai	Tamilnadu State Pollution Control Board	Industrial Area	13.(
...	...	...	...	...	...	...	...	..
2874	773	12-03-14	Tamil Nadu	Trichy	Central Bus Stand, Trichy	Tamilnadu State Pollution Control Board	Residential, Rural and other Areas	15.(
2875	773	12-10-14	Tamil Nadu	Trichy	Central Bus Stand, Trichy	Tamilnadu State Pollution Control Board	Residential, Rural and other Areas	12.(
2876	773	17-12-14	Tamil Nadu	Trichy	Central Bus Stand, Trichy	Tamilnadu State Pollution Control Board	Residential, Rural and other Areas	19.(
2877	773	24-12-14	Tamil Nadu	Trichy	Central Bus Stand, Trichy	Tamilnadu State Pollution Control Board	Residential, Rural and other Areas	15.(
2878	773	31-12-14	Tamil Nadu	Trichy	Central Bus Stand, Trichy	Tamilnadu State Pollution Control Board	Residential, Rural and other Areas	14.(

2879 rows × 11 columns

In [41]:
```
#this column contains nan values
b=df.drop("PM",axis=1)
b
```

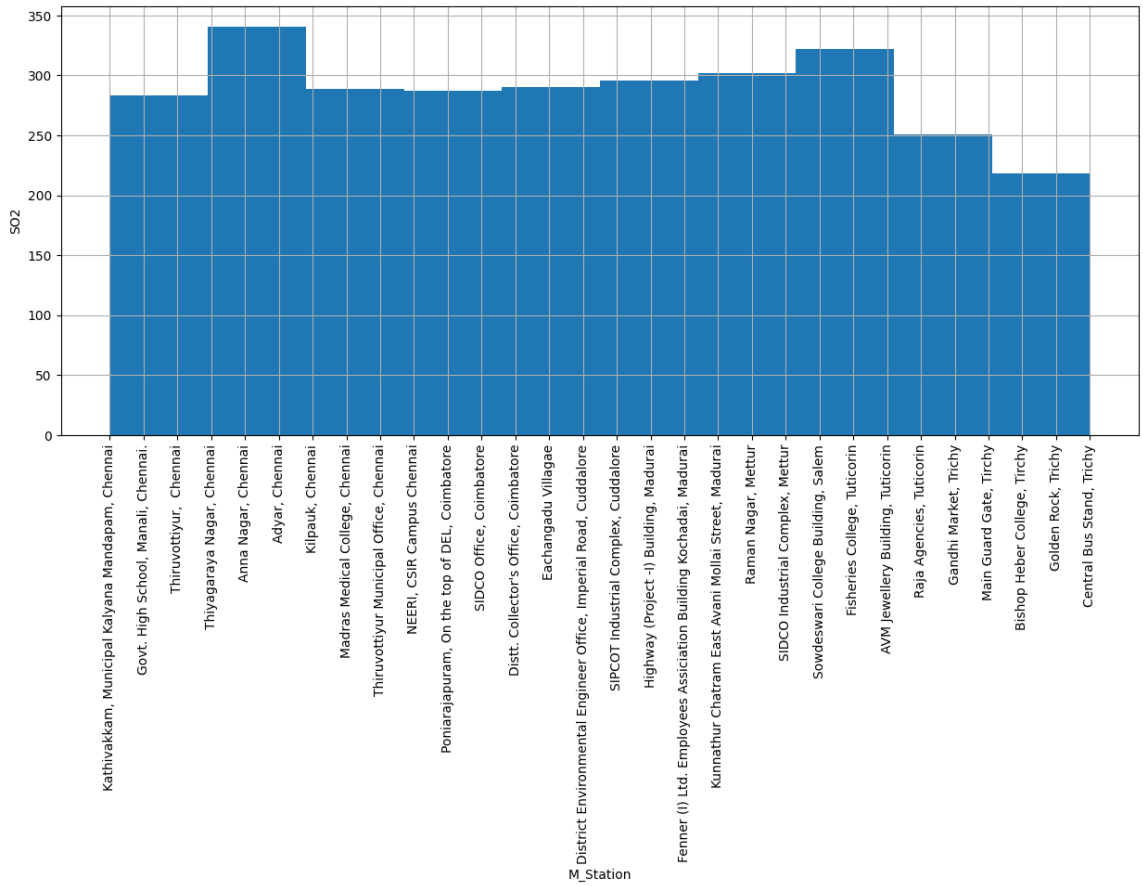
Out[41]:

	Stn Code	Sampling Date	State	City/Town/Village/Area	M_Station	Agency	Type of Location	SO2
0	38	01-02-14	Tamil Nadu	Chennai	Kathivakkam, Municipal Kalyana Mandapam, Chennai	Tamilnadu State Pollution Control Board	Industrial Area	11.(
1	38	01-07-14	Tamil Nadu	Chennai	Kathivakkam, Municipal Kalyana Mandapam, Chennai	Tamilnadu State Pollution Control Board	Industrial Area	13.(
2	38	21-01-14	Tamil Nadu	Chennai	Kathivakkam, Municipal Kalyana Mandapam, Chennai	Tamilnadu State Pollution Control Board	Industrial Area	12.(
3	38	23-01-14	Tamil Nadu	Chennai	Kathivakkam, Municipal Kalyana Mandapam, Chennai	Tamilnadu State Pollution Control Board	Industrial Area	15.(
4	38	28-01-14	Tamil Nadu	Chennai	Kathivakkam, Municipal Kalyana Mandapam, Chennai	Tamilnadu State Pollution Control Board	Industrial Area	13.(
...	...	...	...	...	...	...	...	..
2874	773	12-03-14	Tamil Nadu	Trichy	Central Bus Stand, Trichy	Tamilnadu State Pollution Control Board	Residential, Rural and other Areas	15.(
2875	773	12-10-14	Tamil Nadu	Trichy	Central Bus Stand, Trichy	Tamilnadu State Pollution Control Board	Residential, Rural and other Areas	12.(
2876	773	17-12-14	Tamil Nadu	Trichy	Central Bus Stand, Trichy	Tamilnadu State Pollution Control Board	Residential, Rural and other Areas	19.(
2877	773	24-12-14	Tamil Nadu	Trichy	Central Bus Stand, Trichy	Tamilnadu State Pollution Control Board	Residential, Rural and other Areas	15.(
2878	773	31-12-14	Tamil Nadu	Trichy	Central Bus Stand, Trichy	Tamilnadu State Pollution Control Board	Residential, Rural and other Areas	14.(

2879 rows × 10 columns

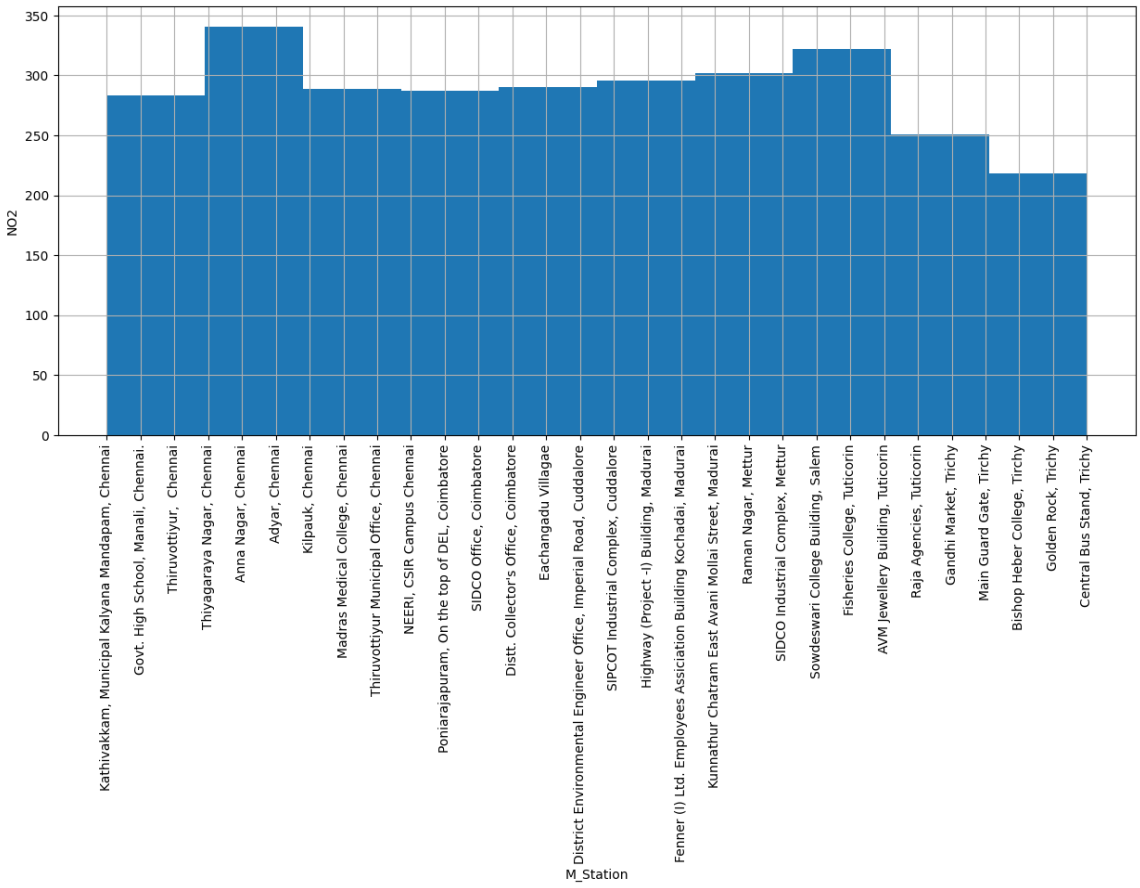
In [42]:
```
plt.figure(figsize=(15, 6))
plt.xticks(rotation=90)
df.M_Station.hist()
plt.xlabel('M_Station')
plt.ylabel('SO2')
plt.plot()
```

Out[42]: []

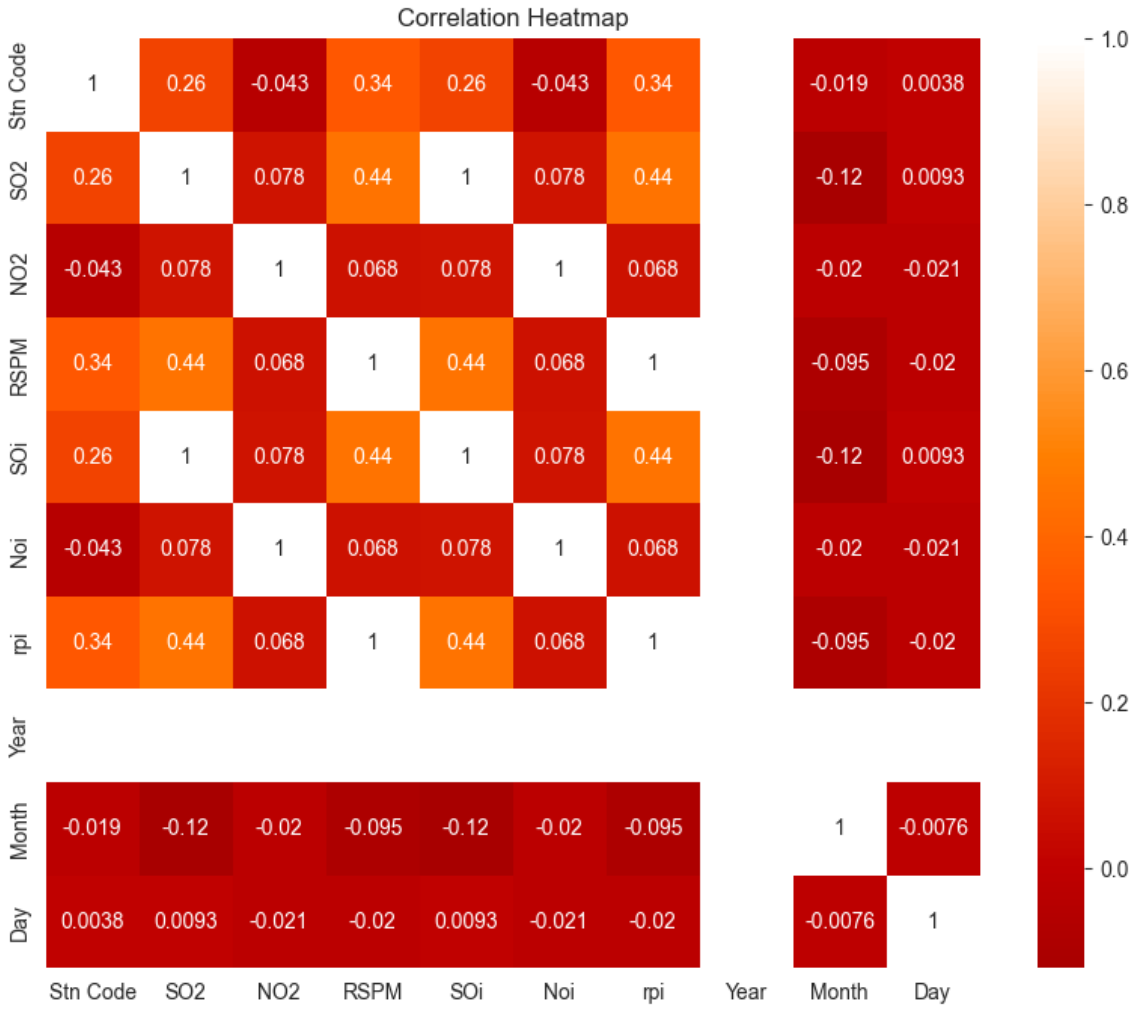


In [43]:
```
plt.figure(figsize=(15, 6))
plt.xticks(rotation=90)
df.M_Station.hist()
plt.xlabel('M_Station')
plt.ylabel('NO2')
plt.plot()
```

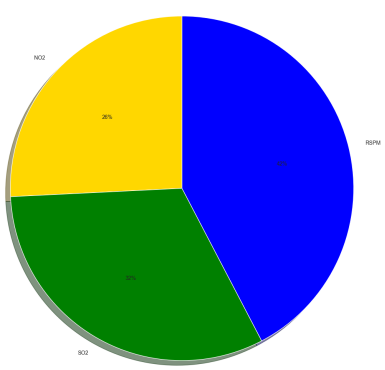
Out[43]: []



In [109]:
```
correlation_matrix = b.corr()
plt.figure(figsize=(10, 8))
sns.heatmap(correlation_matrix, annot=True, cmap="gist_heat", center=0)
plt.title("Correlation Heatmap")
plt.show()
```

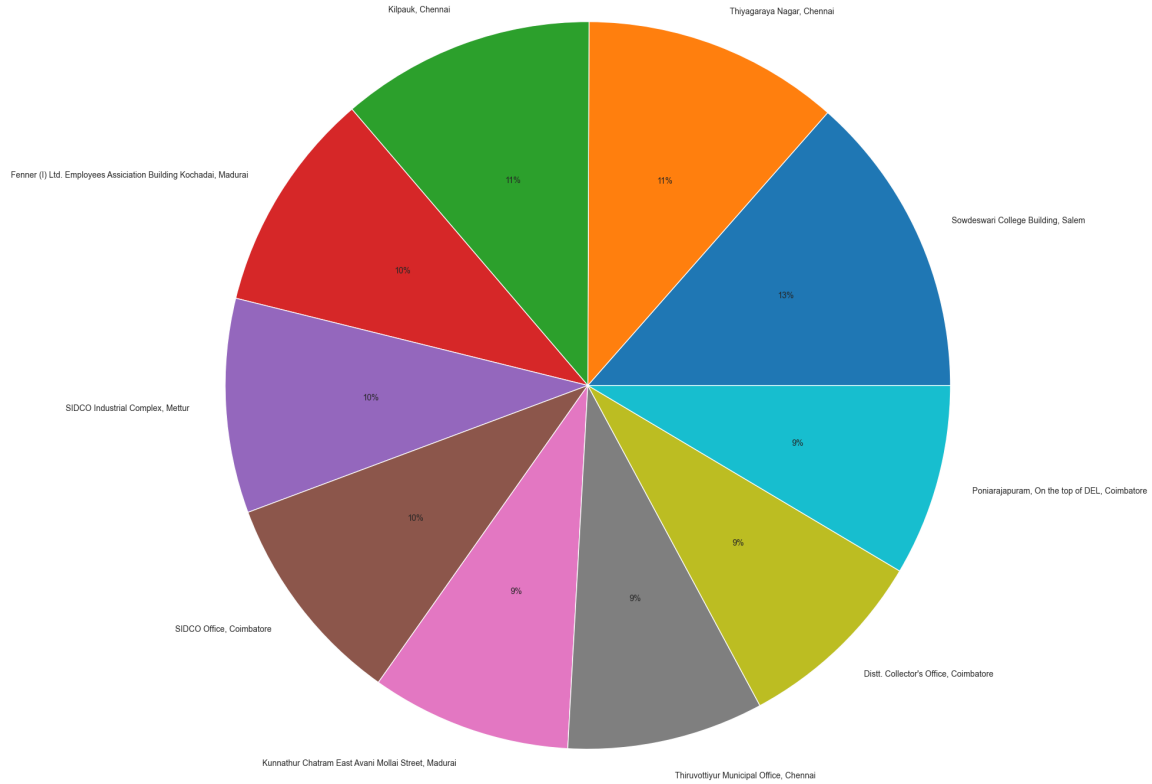


In [112]:
```
labels='NO2','SO2','RSPM'
sizes=[307.0,380.0,504.0]
colors=['gold','green','blue']
explode=(0,0,0)

plt.pie(sizes,labels=labels, colors=colors,radius=1,autopct='%2.f%%', shadow
plt.legend( labels, loc="best")
plt.axis('equal')
plt.tight_layout()
plt.show()
```



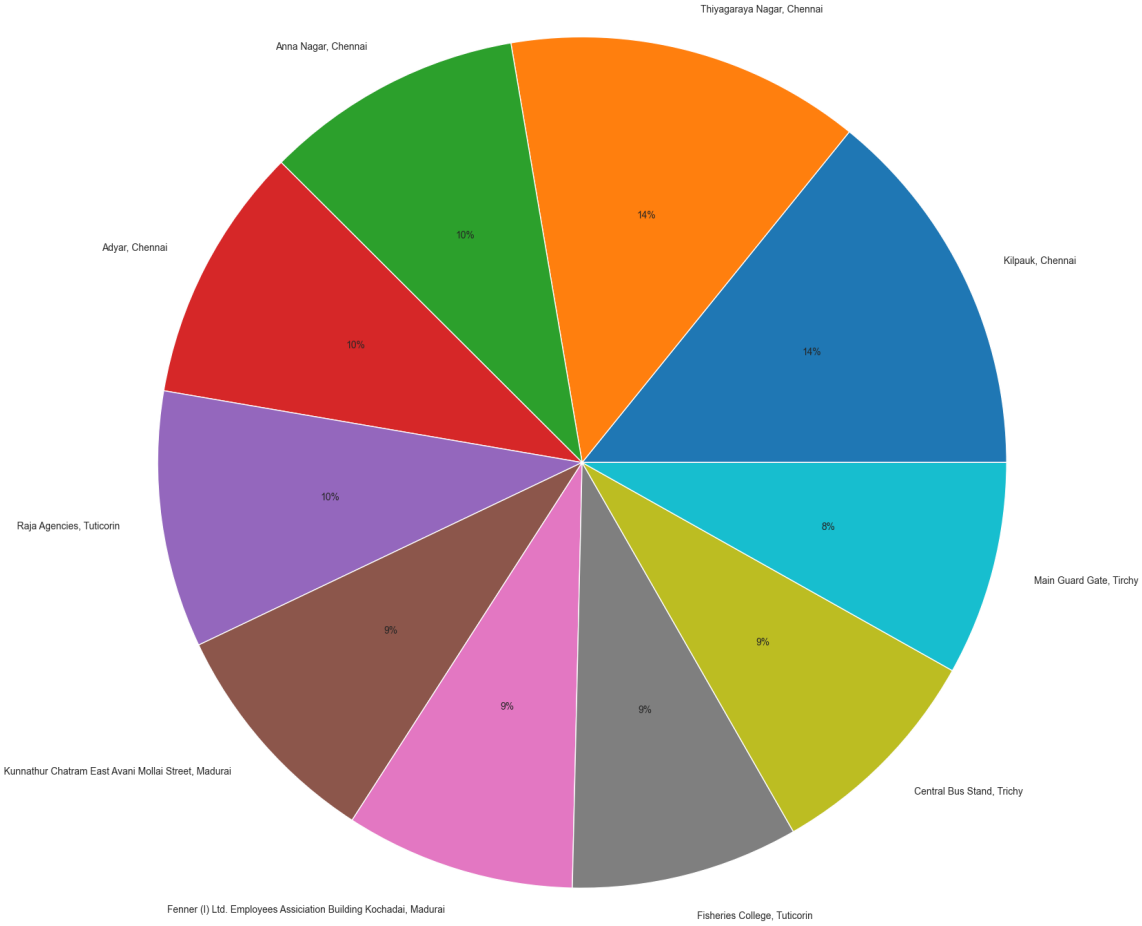
In [113]:
```
d=b.groupby(['M_Station'])['NO2'].sum().sort_values(kind='mergesort',ascend
d.plot(kind='pie',autopct='%.0f%%',radius=2,subplots=True)
```

Out[113]:
```
array([<AxesSubplot: ylabel='NO2'>], dtype=object)
```



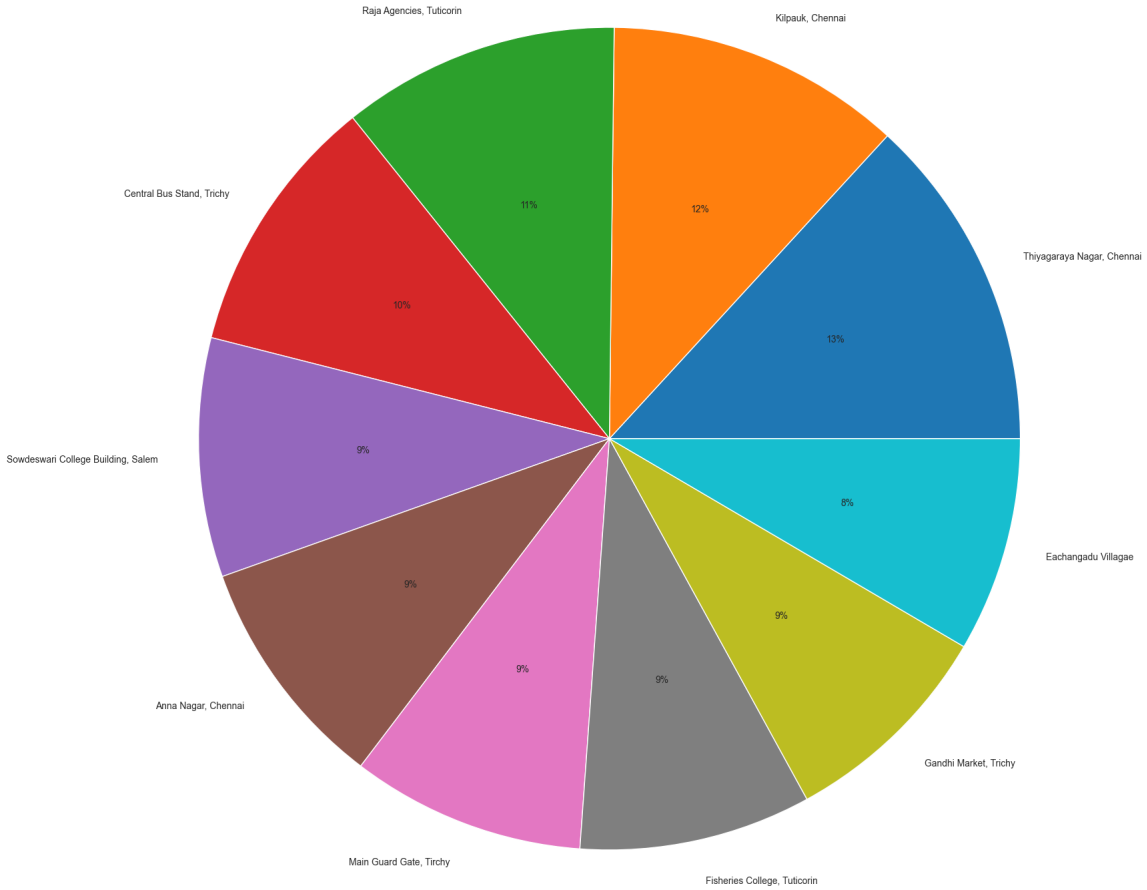
In [114]:
```
d=b.groupby(['M_Station'])['SO2'].sum().sort_values(kind='mergesort',ascend
d.plot(kind='pie',autopct='%.0f%%',radius=2,subplots=True)
```

Out[114]:
```
array([<AxesSubplot: ylabel='SO2'>], dtype=object)
```

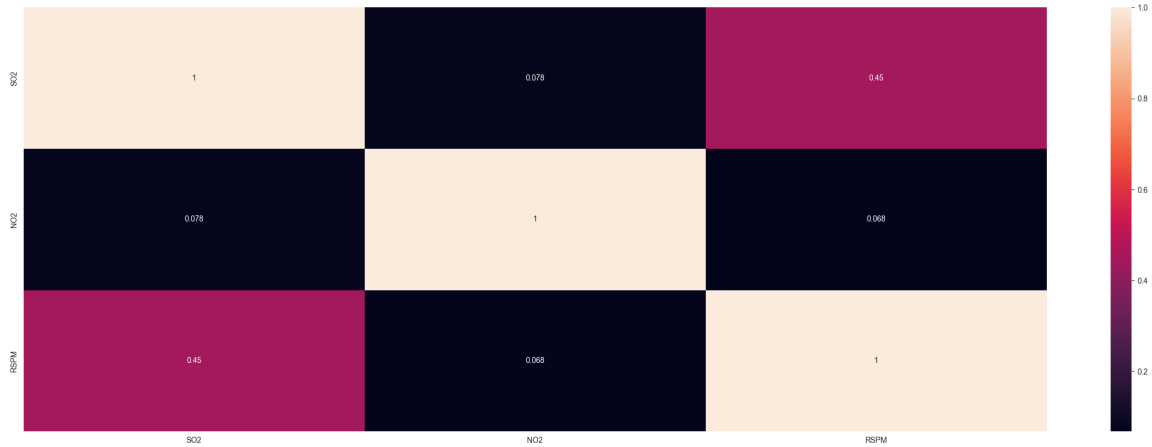


In [116]:
```
d=b.groupby(['M_Station'])['RSPM'].sum().sort_values(kind='mergesort',ascen
d.plot(kind='pie',autopct='%.0f%%',radius=2,subplots=True)
```

Out[116]:
```
array([<AxesSubplot: ylabel='RSPM'>], dtype=object)
```



In [117]:
```
viz=sns.heatmap(df[['SO2','NO2','RSPM']].corr(),annot=True)
```



In [44]:
```
#handing the null values
nullvalues=b.isnull().sum().sort_values(ascending=False)
```

In [45]:
```
nullvalues
```

Out[45]:
```
NO2                       13
SO2                       11
RSPM                       4
Stn Code                   0
Sampling Date              0
State                      0
City/Town/Village/Area     0
M_Station                  0
Agency                     0
Type of Location           0
dtype: int64
```

In [46]:
```
null_values_percentage = (b.isnull().sum()/b.isnull().count()*100).sort_valu
```

In [47]:
```
missing_data_with_percentage = pd.concat([nullvalues, null_values_percentage
```

In [48]:
```
missing_data_with_percentage
```

Out[48]:

	Total	Percent
NO2	13	0.451546
SO2	11	0.382077
RSPM	4	0.138937
Stn Code	0	0.000000
Sampling Date	0	0.000000
State	0	0.000000
City/Town/Village/Area	0	0.000000
M_Station	0	0.000000
Agency	0	0.000000
Type of Location	0	0.000000

In [58]:
```
b['SO2']=b['SO2'].fillna(b['SO2'].mode()[0])
b['NO2']=b['NO2'].fillna(b['NO2'].mode()[0])
b['RSPM']=b['RSPM'].fillna(b['RSPM'].mode()[0])
```

In [59]:
```
b.isnull().sum()
```

Out[59]:
```
Stn Code                  0
Sampling Date             0
State                     0
City/Town/Village/Area    0
M_Station                 0
Agency                    0
Type of Location          0
SO2                       0
NO2                       0
RSPM                      0
SOi                       0
Noi                       0
rpi                       0
RPI_Range                 0
Year                      0
Month                     0
Day                       0
dtype: int64
```

In [60]:
```
def cal_SOi(SO2):
    si=0
    if (SO2<=40):
     si= SO2*(50/40)
    elif (SO2>40 and SO2<=80):
     si= 50+(SO2-40)*(50/40)
    elif (SO2>80 and SO2<=380):
     si= 100+(SO2-80)*(100/300)
    elif (SO2>380 and SO2<=800):
     si= 200+(so2-380)*(100/420)
    elif (SO2>800 and SO2<=1600):
     si= 300+(SO2-800)*(100/800)
    elif (SO2>1600):
     si= 400+(SO2-1600)*(100/800)
    return si
b['SOi']=b['SO2'].apply(cal_SOi)
data= b[['SO2','SOi']]
data.head()
# calculating the individual pollutant index for so2(sulphur dioxide)
```

Out[60]:

	SO2	SOi
0	11.0	13.75
1	13.0	16.25
2	12.0	15.00
3	15.0	18.75
4	13.0	16.25

In [61]:
```
def cal_Noi(NO2):
    ni=0
    if(NO2<=40):
     ni= NO2*50/40
    elif(NO2>40 and NO2<=80):
     ni= 50+(NO2-40)*(50/40)
    elif(NO2>80 and NO2<=180):
     ni= 100+(NO2-80)*(100/100)
    elif(NO2>180 and no2<=280):
     ni= 200+(NO2-180)*(100/100)
    elif(NO2>280 and NO2<=400):
     ni= 300+(NO2-280)*(100/120)
    else:
     ni= 400+(NO2-400)*(100/120)
    return ni
b['Noi']=b['NO2'].apply(cal_Noi)
data= b[['NO2','Noi']]
data.head()
# calculating the individual pollutant index for no2(nitrogen dioxide)
```

Out[61]:

	NO2	Noi
0	17.0	21.25
1	17.0	21.25
2	18.0	22.50
3	16.0	20.00
4	14.0	17.50

In [62]:
```
def cal_RSPMI(RSPM):
    rpi=0
    if(RSPM<=30):
     rpi=RSPM*50/30
    elif(RSPM>30 and rpi<=60):
     rpi=50+(RSPM-30)*50/30
    elif(RSPM>60 and rpi<=90):
     rpi=100+(RSPM-60)*100/30
    elif(RSPM>90 and rpi<=120):
     rpi=200+(RSPM-90)*100/30
    elif(RSPM>120 and rpi<=250):
     rpi=300+(RSPM-120)*(100/130)
    else:
     rpi=400+(RSPM-250)*(100/130)
    return RSPM
b['rpi']=b['RSPM'].apply(cal_RSPMI)
data= b[['RSPM','rpi']]
data.head()
# calculating the individual pollutant index for rspm(respirable suspended p
```

Out[62]:

	RSPM	rpi
0	55.0	55.0
1	45.0	45.0
2	50.0	50.0
3	46.0	46.0
4	42.0	42.0

In [63]:
```
def RPI_Range(x):
    if x<=50:
        return "Good"
    elif x>50 and x<=100:
        return "Moderate"
    elif x>100 and x<=200:
        return "Poor"
    elif x>200 and x<=300:
        return "Unhealthy"
    elif x>300 and x<=400:
        return "Very unhealthy"
    elif x>400:
        return "Hazardous"

b['RPI_Range'] = b['RSPM'] .apply(RPI_Range)
b.head()
# Using threshold values to classify a particular values as good, moderate,
```

Out[63]:

	Stn Code	Sampling Date	State	City/Town/Village/Area	M_Station	Agency	Type of Location	SO2	NO
0	38	01-02-14	Tamil Nadu	Chennai	Kathivakkam, Municipal Kalyana Mandapam, Chennai	Tamilnadu State Pollution Control Board	Industrial Area	11.0	17.
1	38	01-07-14	Tamil Nadu	Chennai	Kathivakkam, Municipal Kalyana Mandapam, Chennai	Tamilnadu State Pollution Control Board	Industrial Area	13.0	17.
2	38	21-01-14	Tamil Nadu	Chennai	Kathivakkam, Municipal Kalyana Mandapam, Chennai	Tamilnadu State Pollution Control Board	Industrial Area	12.0	18.
3	38	23-01-14	Tamil Nadu	Chennai	Kathivakkam, Municipal Kalyana Mandapam, Chennai	Tamilnadu State Pollution Control Board	Industrial Area	15.0	16.
4	38	28-01-14	Tamil Nadu	Chennai	Kathivakkam, Municipal Kalyana Mandapam, Chennai	Tamilnadu State Pollution Control Board	Industrial Area	13.0	14.

In [64]:
```
b['RPI_Range'].value_counts()
# These are the counts of values present in the AQI_Range column.
```

Out[64]:
```
Good          1264
Moderate      1246
Poor           359
Unhealthy       10
Name: RPI_Range, dtype: int64
```

In [65]:
```
date_col=(pd.DatetimeIndex(b['Sampling Date']))
b['Year']=date_col.year
b['Month']=date_col.month
b['Day']=date_col.day
```

In [66]:
```
b
```

Out[66]:

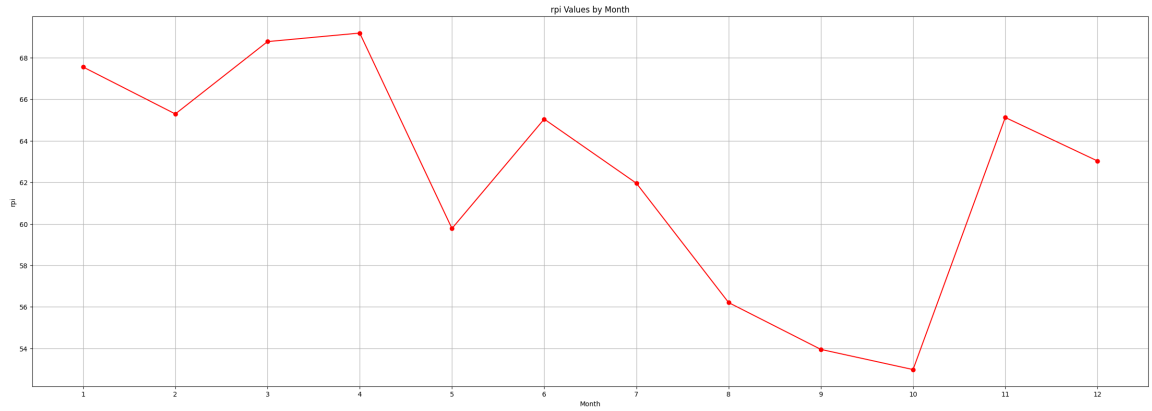
	Stn Code	Sampling Date	State	City/Town/Village/Area	M_Station	Agency	Type of Location	SO2
0	38	01-02-14	Tamil Nadu	Chennai	Kathivakkam, Municipal Kalyana Mandapam, Chennai	Tamilnadu State Pollution Control Board	Industrial Area	11.(
1	38	01-07-14	Tamil Nadu	Chennai	Kathivakkam, Municipal Kalyana Mandapam, Chennai	Tamilnadu State Pollution Control Board	Industrial Area	13.(
2	38	21-01-14	Tamil Nadu	Chennai	Kathivakkam, Municipal Kalyana Mandapam, Chennai	Tamilnadu State Pollution Control Board	Industrial Area	12.(
3	38	23-01-14	Tamil Nadu	Chennai	Kathivakkam, Municipal Kalyana Mandapam, Chennai	Tamilnadu State Pollution Control Board	Industrial Area	15.(
4	38	28-01-14	Tamil Nadu	Chennai	Kathivakkam, Municipal Kalyana Mandapam, Chennai	Tamilnadu State Pollution Control Board	Industrial Area	13.(
...	...	...	...	...	...	...	...	..
2874	773	12-03-14	Tamil Nadu	Trichy	Central Bus Stand, Trichy	Tamilnadu State Pollution Control Board	Residential, Rural and other Areas	15.(
2875	773	12-10-14	Tamil Nadu	Trichy	Central Bus Stand, Trichy	Tamilnadu State Pollution Control Board	Residential, Rural and other Areas	12.(
2876	773	17-12-14	Tamil Nadu	Trichy	Central Bus Stand, Trichy	Tamilnadu State Pollution Control Board	Residential, Rural and other Areas	19.(
2877	773	24-12-14	Tamil Nadu	Trichy	Central Bus Stand, Trichy	Tamilnadu State Pollution Control Board	Residential, Rural and other Areas	15.(
2878	773	31-12-14	Tamil Nadu	Trichy	Central Bus Stand, Trichy	Tamilnadu State Pollution Control Board	Residential, Rural and other Areas	14.(

2879 rows × 17 columns

In [67]:
```
grouped = b.groupby('Month')['rpi'].mean()
grouped
```

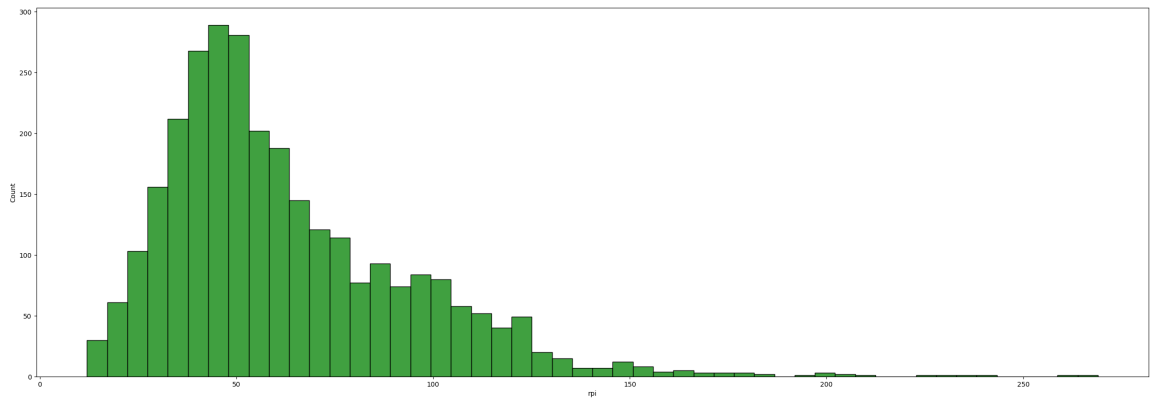
Out[67]:
```
Month
1     67.547826
2     65.287449
3     68.770751
4     69.180672
5     59.774436
6     65.041322
7     61.952569
8     56.208511
9     53.955224
10    52.984043
11    65.118182
12    63.025105
Name: rpi, dtype: float64
```

In [69]:
```
months = grouped.index
rpi = grouped.values
# Creating a line plot with markers
#plt.figure(figsize=(10, 6))
plt.plot(months, rpi, marker='o', color='red', linestyle='-')
plt.title('rpi Values by Month')
plt.xlabel('Month')
plt.ylabel('rpi')
plt.xticks(months)  # Setting x-axis ticks to be the months
plt.grid(True)
plt.show()
```



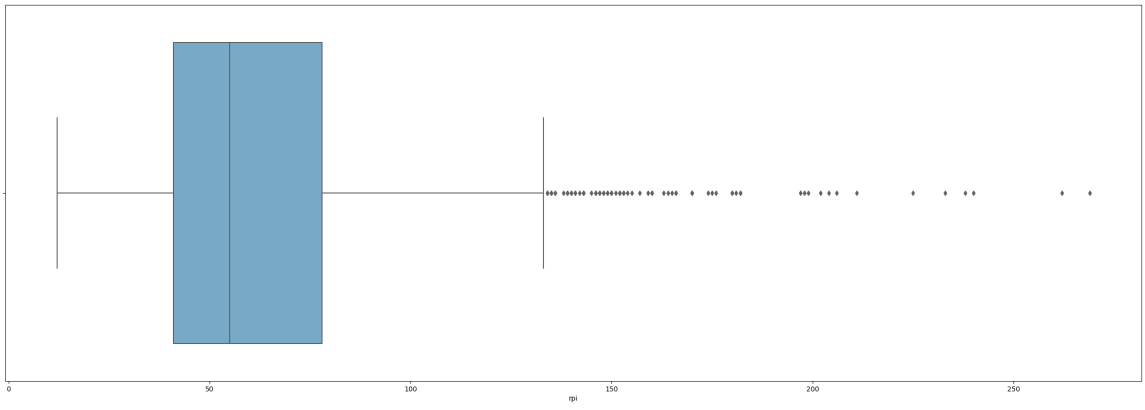
In [70]:
```
sns.histplot(b,x='rpi',bins=50,color='g')
```

Out[70]:
```
<AxesSubplot: xlabel='rpi', ylabel='Count'>
```



In [71]:
```
sns.boxplot(b,x='rpi',palette='Blues')
```

Out[71]:
```
<AxesSubplot: xlabel='rpi'>
```



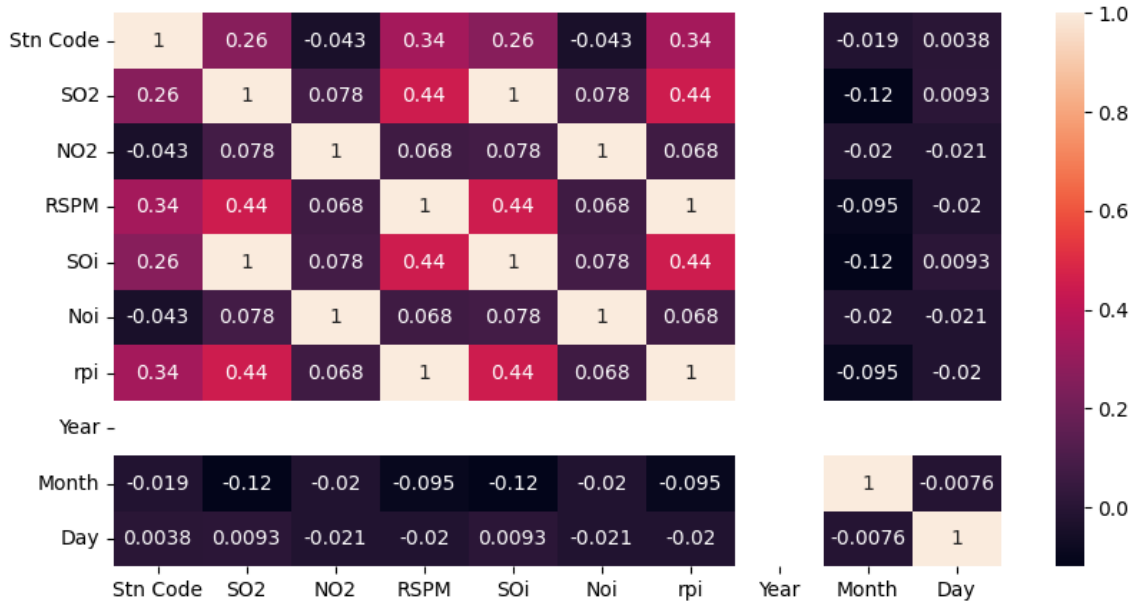
In [72]:
```
b.corr(numeric_only=True)
```

Out[72]:

	Stn Code	SO2	NO2	RSPM	SOi	Noi	rpi	Year	M
Stn Code	1.000000	0.263429	-0.043202	0.336549	0.263429	-0.043202	0.336549	NaN	-0.01
SO2	0.263429	1.000000	0.078191	0.444668	1.000000	0.078191	0.444668	NaN	-0.12
NO2	-0.043202	0.078191	1.000000	0.067767	0.078191	1.000000	0.067767	NaN	-0.01
RSPM	0.336549	0.444668	0.067767	1.000000	0.444668	0.067767	1.000000	NaN	-0.09
SOi	0.263429	1.000000	0.078191	0.444668	1.000000	0.078191	0.444668	NaN	-0.12
Noi	-0.043202	0.078191	1.000000	0.067767	0.078191	1.000000	0.067767	NaN	-0.01
rpi	0.336549	0.444668	0.067767	1.000000	0.444668	0.067767	1.000000	NaN	-0.09
Year	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	
Month	-0.019100	-0.120004	-0.019605	-0.095041	-0.120004	-0.019605	-0.095041	NaN	1.00
Day	0.003780	0.009256	-0.021214	-0.020039	0.009256	-0.021214	-0.020039	NaN	-0.00

In [73]:
```
plt.figure(figsize=(10,5))
sns.heatmap(b.corr(numeric_only=True),annot=True)
```

Out[73]: <AxesSubplot: >



In [74]:
```
# Descriptive statistics for specific columns
mean_values = b.mean()
median_values = b.median()
std_deviation = b.std()
min_values = b.min()
max_values = b.max()

# Display the computed statistics
print("Mean Values:")
print(mean_values)
```

```
Mean Values:
Stn Code     475.750261
SO2           11.508857
NO2           22.136158
RSPM          62.472734
SOi           14.386072
Noi           27.670198
rpi           62.472734
Year        2014.000000
Month          6.407780
Day           15.806183
dtype: float64
```

In [75]:
```
print("\nMedian Values:")
print(median_values)
```

```
Median Values:
Stn Code     366.0
SO2           12.0
NO2           22.0
RSPM          55.0
SOi           15.0
Noi           27.5
rpi           55.0
Year        2014.0
Month          6.0
Day           16.0
dtype: float64
```

In [76]:
```
print("\nStandard Deviations:")
print(std_deviation)
```

```
Standard Deviations:
Stn Code    277.675577
SO2           5.042885
NO2           7.112582
RSPM         31.352252
SOi           6.303606
Noi           8.890727
rpi          31.352252
Year          0.000000
Month         3.408451
Day           8.716400
dtype: float64
```

In [77]:
```
print("\nMinimum Values:")
print(min_values)
```

```
Minimum Values:
Stn Code                                                                3
8
Sampling Date                                                    01-02-1
4
State                                                          Tamil Nad
u
City/Town/Village/Area                                           Chenna
i
M_Station                                  AVM Jewellery Building, Tuticori
n
Agency                       National Environmental Engineering Research I
n...
Type of Location                                        Industrial Are
a
SO2                                                                  2.
0
NO2                                                                  5.
0
RSPM                                                                12.
0
SOi                                                                  2.
5
Noi                                                                 6.2
5
rpi                                                                 12.
0
RPI_Range                                                         Goo
d
Year                                                              201
4
Month
1
Day
1
dtype: object
```

In [78]:
```
print("\nMaximum Values:")
print(max_values)
```

```
Maximum Values:
Stn Code                                                    773
Sampling Date                                         31-12-14
State                                                Tamil Nadu
City/Town/Village/Area                                   Trichy
M_Station                              Thiyagaraya Nagar, Chennai
Agency                    Tamilnadu State Pollution Control Board
Type of Location               Residential, Rural and other Areas
SO2                                                        49.0
NO2                                                        71.0
RSPM                                                      269.0
SOi                                                       61.25
Noi                                                       88.75
rpi                                                       269.0
RPI_Range                                             Unhealthy
Year                                                       2014
Month                                                        12
Day                                                          31
dtype: object
```

In [79]:
```
X=b[['SOi','Noi']]
Y=b['rpi']
X.head()
# we only select columns like soi, noi, rpi
```

Out[79]:

	SOi	Noi
0	13.75	21.25
1	16.25	21.25
2	15.00	22.50
3	18.75	20.00
4	16.25	17.50

In [80]:
```
Y.head()
# the AQI column is the target column
```

Out[80]:
```
0    55.0
1    45.0
2    50.0
3    46.0
4    42.0
Name: rpi, dtype: float64
```

In [81]:
```
X_train,X_test,Y_train,Y_test=train_test_split(X,Y,test_size=0.2,random_sta
print(X_train.shape,X_test.shape,Y_train.shape,Y_test.shape)
# splitting the data into training and testing data
```

```
(2303, 2) (576, 2) (2303,) (576,)
```

In [83]:
```
model=LinearRegression()
model.fit(X_train,Y_train)
```

Out[83]:
```
LinearRegression()
```

In a Jupyter environment, please rerun this cell to show the HTML representation or trust the notebook.
On GitHub, the HTML representation is unable to render, please try loading this page with nbviewer.org.

In [84]:
```
#predicting train
train_pred=model.predict(X_train)
#predicting on test
test_pred=model.predict(X_test)
```

In [85]:
```
RMSE_train=(np.sqrt(metrics.mean_squared_error(Y_train,train_pred)))
RMSE_test=(np.sqrt(metrics.mean_squared_error(Y_test,test_pred)))
print("RMSE TrainingData = ",str(RMSE_train))
print("RMSE TestData = ",str(RMSE_test))
print('-'*50)
print('RSquared value on train:',model.score(X_train, Y_train))
print('RSquared value on test:',model.score(X_test, Y_test))
```

```
RMSE TrainingData =  28.20370232315062
RMSE TestData =  27.470003898413395
--------------------------------------------------
RSquared value on train: 0.1991238486260516
RSquared value on test: 0.19630364013815804
```

In [86]:
```
DT=DecisionTreeRegressor()
DT.fit(X_train,Y_train)
```

Out[86]:
```
DecisionTreeRegressor()
```

In a Jupyter environment, please rerun this cell to show the HTML representation or trust the notebook.
On GitHub, the HTML representation is unable to render, please try loading this page with nbviewer.org.

In [87]:
```
#predicting train
train_preds=DT.predict(X_train)
#predicting on test
test_preds=DT.predict(X_test)
```

In [88]:
```
RMSE_train=(np.sqrt(metrics.mean_squared_error(Y_train,train_preds)))
RMSE_test=(np.sqrt(metrics.mean_squared_error(Y_test,test_preds)))
print("RMSE TrainingData = ",str(RMSE_train))
print("RMSE TestData = ",str(RMSE_test))
print('-'*50)
print('RSquared value on train:',DT.score(X_train, Y_train))
print('RSquared value on test:',DT.score(X_test, Y_test))
```

```
RMSE TrainingData =  21.934741520752567
RMSE TestData =  27.743316976028918
--------------------------------------------------
RSquared value on train: 0.5155843572539975
RSquared value on test: 0.18023131013864246
```

In [89]:
```
RF=RandomForestRegressor().fit(X_train,Y_train)
```

In [90]:
```
#predicting train
train_preds1=RF.predict(X_train)
#predicting on test
test_preds1=RF.predict(X_test)
```

In [91]:
```
RMSE_train=(np.sqrt(metrics.mean_squared_error(Y_train,train_preds1)))
RMSE_test=(np.sqrt(metrics.mean_squared_error(Y_test,test_preds1)))
print("RMSE TrainingData = ",str(RMSE_train))
print("RMSE TestData = ",str(RMSE_test))
print('-'*50)
print('RSquared value on train:',RF.score(X_train, Y_train))
print('RSquared value on test:',RF.score(X_test, Y_test))
```

```
RMSE TrainingData =  22.232709157936906
RMSE TestData =  26.98010877992365
--------------------------------------------------
RSquared value on train: 0.5023340927840427
RSquared value on test: 0.22471398163277145
```

In [92]:
```
from sklearn.linear_model import LogisticRegression
from sklearn.tree import DecisionTreeClassifier
from sklearn.ensemble import RandomForestClassifier
from sklearn.neighbors import KNeighborsClassifier
```

In [93]:
```
X2 = b[['SOi','Noi']]
Y2 = b['rpi']
# Splitting the data into independent and dependent columns for classificat
```

In [94]:
```
X_train2, X_test2, Y_train2, Y_test2 = train_test_split(X2, Y2, test_size=0
```

In [95]:
```
#fit the model on train data
log_reg = LogisticRegression().fit(X_train2, Y_train2)

#predict on train
train_preds2 = log_reg.predict(X_train2)
#accuracy on train
print("Model accuracy on train is: ", accuracy_score(Y_train2, train_preds2

#predict on test
test_preds2 = log_reg.predict(X_test2)
#accuracy on test
print("Model accuracy on test is: ", accuracy_score(Y_test2, test_preds2))
print('-'*50)

# Kappa Score.
print('KappaScore is: ', metrics.cohen_kappa_score(Y_test2,test_preds2))
```

```
Model accuracy on train is:  0.029564315352697094
Model accuracy on test is:  0.022082018927444796
--------------------------------------------------
KappaScore is:  0.004585235406931121
```

In [96]:
```
log_reg.predict([[727,327.55]])
```

Out[96]:
```
array([145.])
```

In [97]:
```
log_reg.predict([[2.7,45]])
```

Out[97]:
```
array([59.])
```

In [98]:
```
log_reg.predict([[2,45.8]])
```

Out[98]:
```
array([12.])
```

In [99]:
```
#fit the model on train data
DT2 = DecisionTreeClassifier().fit(X_train2,Y_train2)

#predict on train
train_preds3 = DT2.predict(X_train2)
#accuracy on train
print("Model accuracy on train is: ", accuracy_score(Y_train2, train_preds3

#predict on test
test_preds3 = DT2.predict(X_test2)
#accuracy on test
print("Model accuracy on test is: ", accuracy_score(Y_test2, test_preds3))
print('-'*50)

# Kappa Score
print('KappaScore is: ', metrics.cohen_kappa_score(Y_test2,test_preds3))
```

```
Model accuracy on train is:  0.2966804979253112
Model accuracy on test is:  0.014721345951629864
--------------------------------------------------
KappaScore is:  0.0019499836474660137
```

In [100]:
```
#fit the model on train data
RF=RandomForestClassifier().fit(X_train2,Y_train2)
#predict on train
train_preds4 = RF.predict(X_train2)
#accuracy on train
print("Model accuracy on train is: ", accuracy_score(Y_train2, train_preds4

#predict on test
test_preds4 = RF.predict(X_test2)
#accuracy on test
print("Model accuracy on test is: ", accuracy_score(Y_test2, test_preds4))
print('-'*50)

# Kappa Score
print('KappaScore is: ', metrics.cohen_kappa_score(Y_test2,test_preds4))
```

```
Model accuracy on train is:  0.2966804979253112
Model accuracy on test is:  0.01892744479495268
--------------------------------------------------
KappaScore is:  0.00732688021910155
```

In [101]:
```
#fit the model on train data
KNN = KNeighborsClassifier().fit(X_train2,Y_train2)
#predict on train
train_preds5 = KNN.predict(X_train2)
#accuracy on train
print("Model accuracy on train is: ", accuracy_score(Y_train2, train_preds5

#predict on test
test_preds5 = KNN.predict(X_test2)
#accuracy on test
print("Model accuracy on test is: ", accuracy_score(Y_test2, test_preds5))
print('-'*50)

# Kappa Score
print('KappaScore is: ', metrics.cohen_kappa_score(Y_test2,test_preds5))
```

```
Model accuracy on train is:  0.17271784232365145
Model accuracy on test is:  0.016824395373291272
--------------------------------------------------
KappaScore is:  0.004926213116350109
```

In [102]:
```
# training the model on training set
from sklearn.naive_bayes import GaussianNB
gnb = GaussianNB()
gnb.fit(X_train2, Y_train2)

# making predictions on the testing set
y_pred = gnb.predict(X_test2)

# comparing actual response values (y_test) with predicted response values
from sklearn import metrics
print("Gaussian Naive Bayes model accuracy(in %):", metrics.accuracy_score(
```

```
Gaussian Naive Bayes model accuracy(in %): 1.4721345951629863
```

In [103]:
```
from sklearn.linear_model import Perceptron, LogisticRegression
from sklearn.svm import SVC
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import StandardScaler
from sklearn import datasets
from sklearn import metrics
```

In [104]:
```
sc = StandardScaler()
sc.fit(X_train2)
X_train2_std = sc.transform(X_train2)
X_test2_std = sc.transform(X_test2)
```

In [105]:
```
# Instantiate the Support Vector Classifier (SVC)
svc = SVC(C=1.0, random_state=1, kernel='linear')

# Fit the model
svc.fit(X_train2_std, Y_train2)
```

Out[105]:
```
SVC(kernel='linear', random_state=1)
```
In a Jupyter environment, please rerun this cell to show the HTML representation or trust the notebook.
On GitHub, the HTML representation is unable to render, please try loading this page with nbviewer.org.

In [106]:
```
# Make the predictions
y_predict = svc.predict(X_test2_std)

# Measure the performance
print("Accuracy score %.3f" %metrics.accuracy_score(Y_test2, y_predict))
```

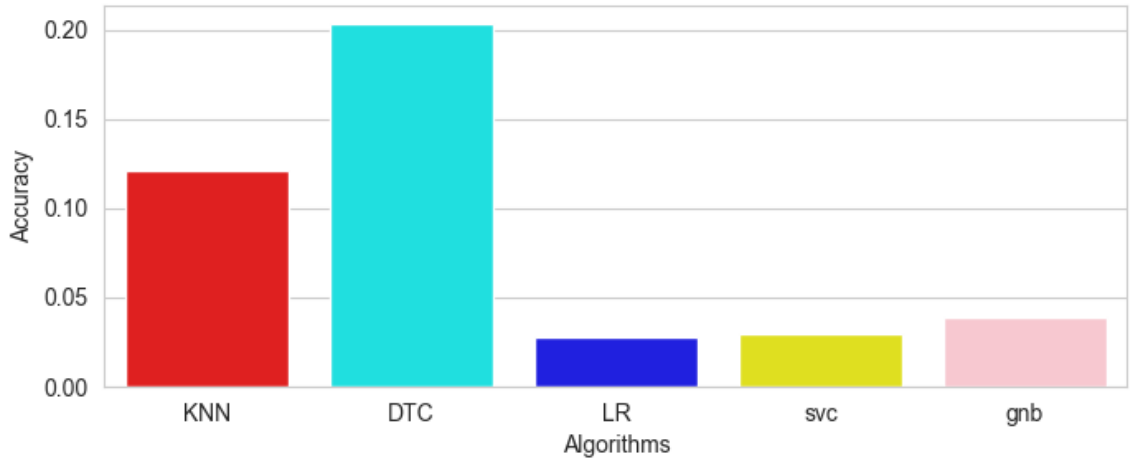
```
Accuracy score 0.019
```

In [107]:
```
decisiontree = DecisionTreeClassifier()
logisticregression = LogisticRegression()
knearestclassifier = KNeighborsClassifier()
svm_classifier = SVC()
naivebayesclassifier=GaussianNB()

knearestclassifier.fit(X_train2,Y_train2)
decisiontree.fit(X_train2, Y_train2)
logisticregression.fit(X_train2, Y_train2)
svc.fit(X_train2, Y_train2)
gnb.fit(X_train2,Y_train2)

data1 = {"Classification Algorithms": ["KNN", "DTC",
                                       "LR","svc","gnb"],
      "Score": [knearestclassifier.score(X2,Y2), decisiontree.score(X2, Y2)
                logisticregression.score(X2, Y2),svc.score(X2,Y2),gnb.score
score = pd.DataFrame(data1)
score
```

Out[107]:

	Classification Algorithms	Score
0	KNN	0.121223
1	DTC	0.203543
2	LR	0.027093
3	svc	0.029524
4	gnb	0.038208

In [108]:
```
colors= ["red","cyan","blue","yellow","pink"]
sns.set_style("whitegrid")
plt.figure(figsize=(8,3))
sns.barplot(x=data1['Classification Algorithms'],y=data1['Score'], palette =
plt.ylabel("Accuracy")
plt.xlabel("Algorithms")
plt.show()
```



In [63]:
```
b
```

Out[63]:

	Stn Code	Sampling Date	State	City/Town/Village/Area	M_Station	Agency	Type of Location	SO2
0	38	01-02-14	Tamil Nadu	Chennai	Kathivakkam, Municipal Kalyana Mandapam, Chennai	Tamilnadu State Pollution Control Board	Industrial Area	11.0
1	38	01-07-14	Tamil Nadu	Chennai	Kathivakkam, Municipal Kalyana Mandapam, Chennai	Tamilnadu State Pollution Control Board	Industrial Area	13.0
2	38	21-01-14	Tamil Nadu	Chennai	Kathivakkam, Municipal Kalyana Mandapam, Chennai	Tamilnadu State Pollution Control Board	Industrial Area	12.0
3	38	23-01-14	Tamil Nadu	Chennai	Kathivakkam, Municipal Kalyana Mandapam, Chennai	Tamilnadu State Pollution Control Board	Industrial Area	15.0
4	38	28-01-14	Tamil Nadu	Chennai	Kathivakkam, Municipal Kalyana Mandapam, Chennai	Tamilnadu State Pollution Control Board	Industrial Area	13.0
...	...	...	...	...	...	...	...	...
2874	773	12-03-14	Tamil Nadu	Trichy	Central Bus Stand, Trichy	Tamilnadu State Pollution Control Board	Residential, Rural and other Areas	15.0
2875	773	12-10-14	Tamil Nadu	Trichy	Central Bus Stand, Trichy	Tamilnadu State Pollution Control Board	Residential, Rural and other Areas	12.0
2876	773	17-12-14	Tamil Nadu	Trichy	Central Bus Stand, Trichy	Tamilnadu State Pollution Control Board	Residential, Rural and other Areas	19.0
2877	773	24-12-14	Tamil Nadu	Trichy	Central Bus Stand, Trichy	Tamilnadu State Pollution Control Board	Residential, Rural and other Areas	15.0
2878	773	31-12-14	Tamil Nadu	Trichy	Central Bus Stand, Trichy	Tamilnadu State Pollution Control Board	Residential, Rural and other Areas	14.0

2879 rows × 17 columns

In [ ]: