```
import pandas as pd
import numpy as np
import seaborn as sns
import matplotlib.pyplot as plt
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
```

```
from google.colab import drive
drive.mount("/content/drive")
```

```
Drive already mounted at /content/drive; to attempt to forcibly remount, call drive.mount("/content/drive", force_remount=True).
```

```
df=pd.read_csv("/content/drive/MyDrive/DS&A_Project/data.csv",encoding='latin-1')
```

```
<ipython-input-47-d67a7fd7b8d5>:1: DtypeWarning: Columns (0) have mixed types. Specify dtype option on import or set low_memory=False.
  df=pd.read_csv("/content/drive/MyDrive/DS&A_Project/data.csv",encoding='latin-1')
```

```
df
```

	stn_code	sampling_date	state	location	agency	type	so2	no2	rspm	spm	location_monitoring_station	pm2_5
0	150.0	February - M021990	Andhra Pradesh	Hyderabad	NaN	Residential, Rural and other Areas	4.8	17.4	NaN	NaN	NaN	NaN
1	151.0	February - M021990	Andhra Pradesh	Hyderabad	NaN	Industrial Area	3.1	7.0	NaN	NaN	NaN	NaN
2	152.0	February - M021990	Andhra Pradesh	Hyderabad	NaN	Residential, Rural and other Areas	6.2	28.5	NaN	NaN	NaN	NaN
3	150.0	March - M031990	Andhra Pradesh	Hyderabad	NaN	Residential, Rural and other Areas	6.3	14.7	NaN	NaN	NaN	NaN
4	151.0	March - M031990	Andhra Pradesh	Hyderabad	NaN	Industrial Area	4.7	7.5	NaN	NaN	NaN	NaN
...	...	...	...	...	...	...	...	...	...	...	...	...
435737	SAMP	24-12-15	West Bengal	ULUBERIA	West Bengal State Pollution Control Board	RIRUO	22.0	50.0	143.0	NaN	Inside Rampal Industries,ULUBERIA	NaN
435738	SAMP	29-12-15	West Bengal	ULUBERIA	West Bengal State	RIRUO	20.0	46.0	171.0	NaN	Inside Rampal	NaN

```
df.shape
```

```
(435742, 13)
```

```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 435742 entries, 0 to 435741
Data columns (total 13 columns):
 #   Column                       Non-Null Count   Dtype
---  ------                       --------------   -----
 0   stn_code                     291665 non-null  object
 1   sampling_date                435739 non-null  object
 2   state                        435742 non-null  object
 3   location                     435739 non-null  object
 4   agency                       286261 non-null  object
 5   type                         430349 non-null  object
 6   so2                          401096 non-null  float64
 7   no2                          419509 non-null  float64
 8   rspm                         395520 non-null  float64
 9   spm                          198355 non-null  float64
 10  location_monitoring_station  408251 non-null  object
 11  pm2_5                        9314 non-null    float64
 12  date                         435735 non-null  object
dtypes: float64(5), object(8)
memory usage: 43.2+ MB
```

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

	0
stn_code	144077
sampling_date	3
state	0
location	3
agency	149481
type	5393
so2	34646
no2	16233
rspm	40222
spm	237387
location_monitoring_station	27491
pm2_5	426428
date	7

dtype: int64

```
df.describe()
```

	so2	no2	rspm	spm	pm2_5
count	401096.000000	419509.000000	395520.000000	198355.000000	9314.000000
mean	10.829414	25.809623	108.832784	220.783480	40.791467
std	11.177187	18.503086	74.872430	151.395457	30.832525
min	0.000000	0.000000	0.000000	0.000000	3.000000
25%	5.000000	14.000000	56.000000	111.000000	24.000000
50%	8.000000	22.000000	90.000000	187.000000	32.000000
75%	13.700000	32.200000	142.000000	296.000000	46.000000
max	909.000000	876.000000	6307.033333	3380.000000	504.000000

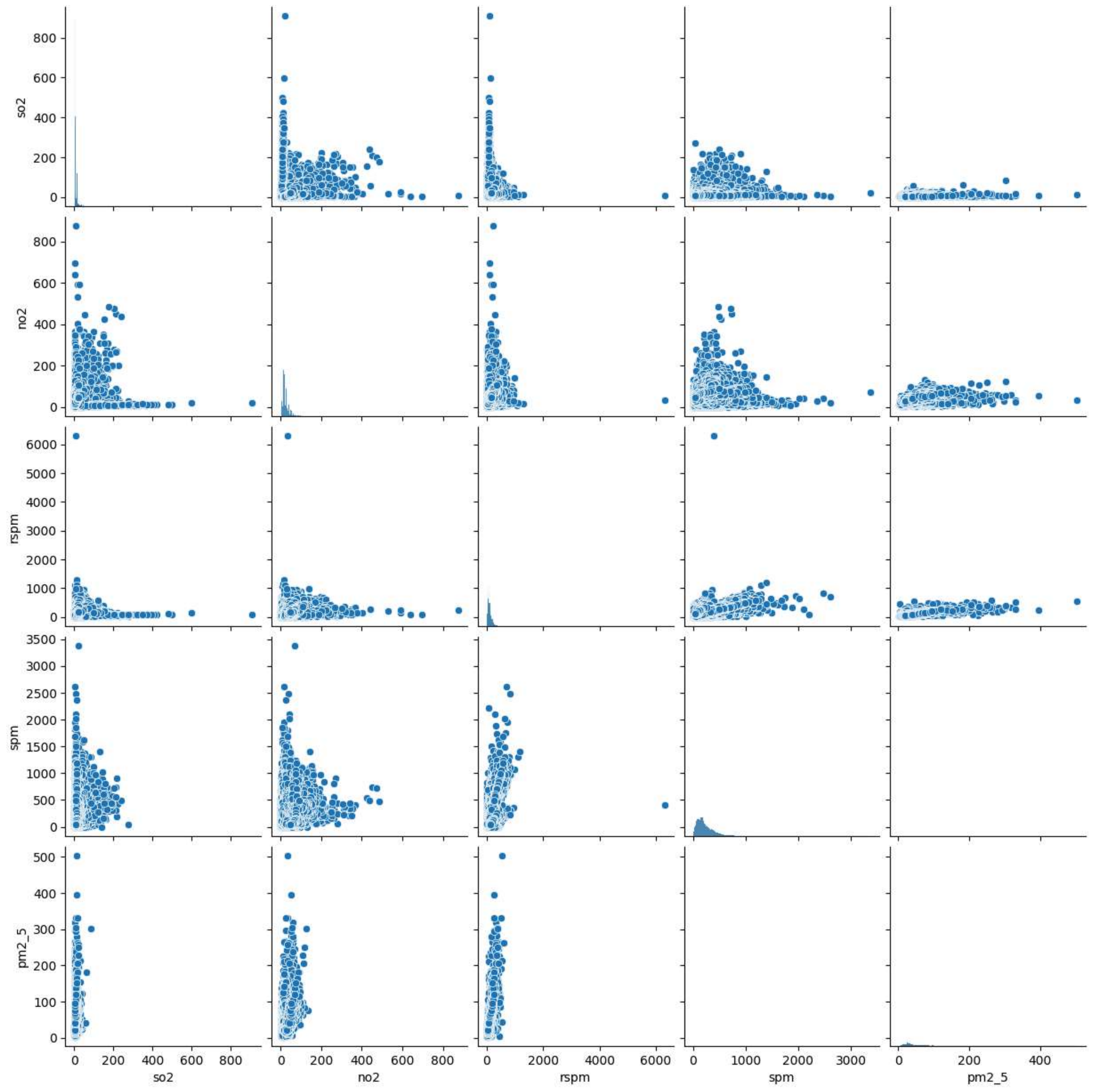
```
df.nunique()
```

	0
stn_code	803
sampling_date	5485
state	37
location	304
agency	64
type	10
so2	4197
no2	6864
rspm	6065
spm	6668
location_monitoring_station	991
pm2_5	433
date	5067

dtype: int64

```
sns.pairplot(data=df)
```

```
<seaborn.axisgrid.PairGrid at 0x790582974b10>
```



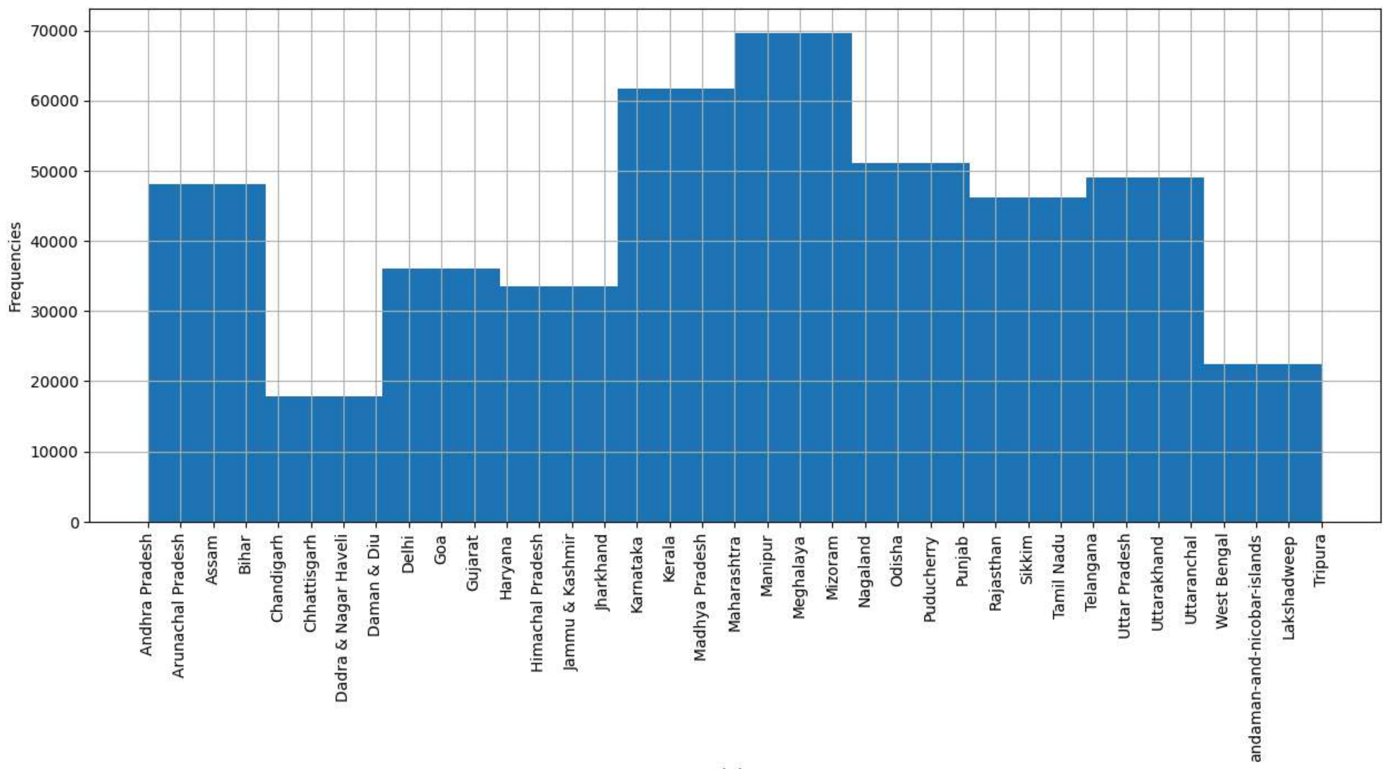
```
df['state'].value_counts()
```

```
                                count
                        state
              Maharashtra       60384
             Uttar Pradesh      42816
            Andhra Pradesh      26368
                  Punjab        25634
                Rajasthan       25589
                  Kerala        24728
          Himachal Pradesh      22896
               West Bengal      22463
                  Gujarat       21279
                Tamil Nadu      20597
            Madhya Pradesh      19920
                  Assam         19361
                  Odisha        19279
                Karnataka       17119
                  Delhi          8551
               Chandigarh        8520
              Chhattisgarh       7831
                   Goa           6206
                Jharkhand        5968
                 Mizoram         5338
                Telangana        3978
                Meghalaya        3853
               Puducherry        3785
                 Haryana         3420
                 Nagaland        2463
                  Bihar          2275
               Uttarakhand       1961
            Jammu & Kashmir      1289
              Daman & Diu         782
         Dadra & Nagar Haveli     634
               Uttaranchal        285
           Arunachal Pradesh       90
                 Manipur           76
                  Sikkim            1
      andaman-and-nicobar-islands   1
               Lakshadweep          1
                 Tripura            1

dtype: int64
```

```
plt.figure(figsize=(15, 6))
plt.xticks(rotation=90)
df.state.hist()
plt.xlabel('state')
plt.ylabel('Frequencies')
plt.plot()
```

```
[]
```



```
df['agency'].value_counts()
```

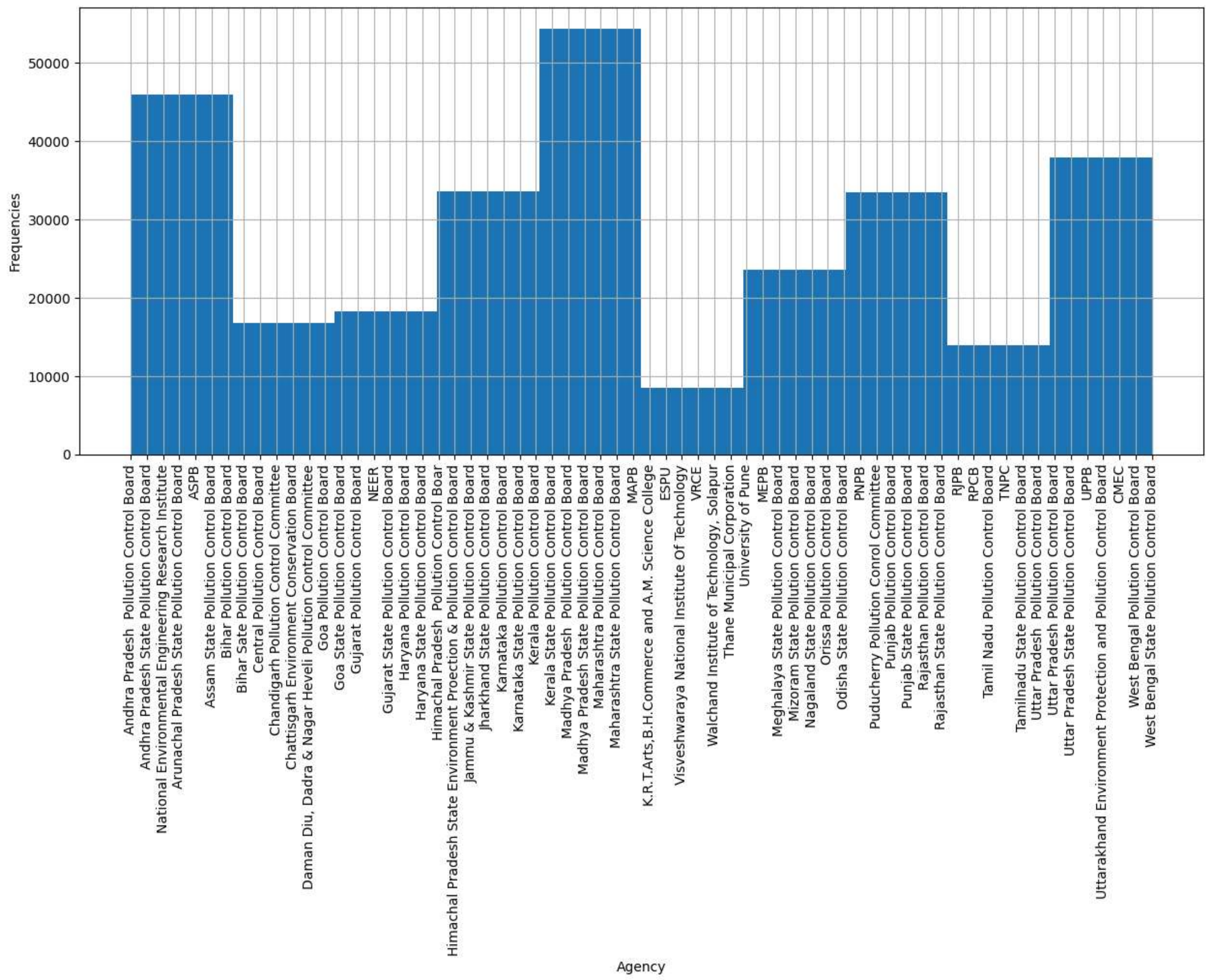
agency	count
Maharashtra State Pollution Control Board	27857
Uttar Pradesh State Pollution Control Board	22686
Andhra Pradesh State Pollution Control Board	19139
Himachal Pradesh State Environment Proection & Pollution Control Board	15287
Punjab State Pollution Control Board	15232
...	...
Arunachal Pradesh State Pollution Control Board	90
TNPC	82
RPCB	63
VRCE	61
RJPB	53

64 rows × 1 columns

dtype: int64

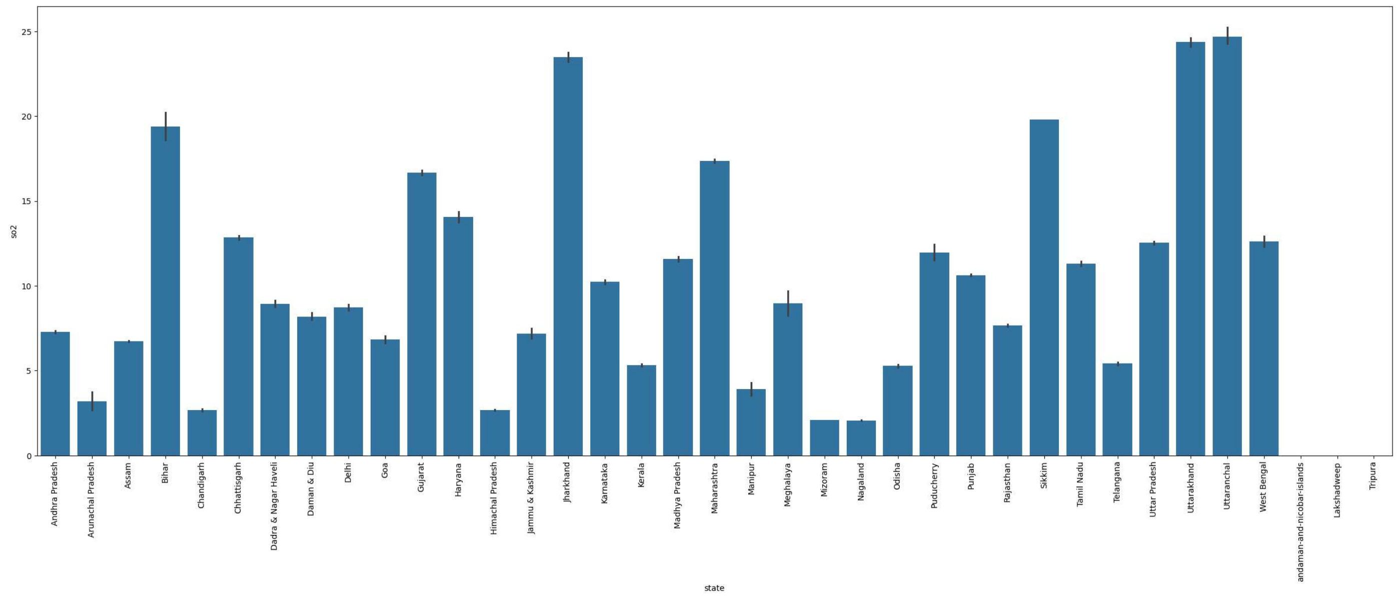
```
plt.figure(figsize=(15, 6))
plt.xticks(rotation=90)
df.agency.hist()
plt.xlabel('Agency')
plt.ylabel('Frequencies')
plt.plot()
```

[]



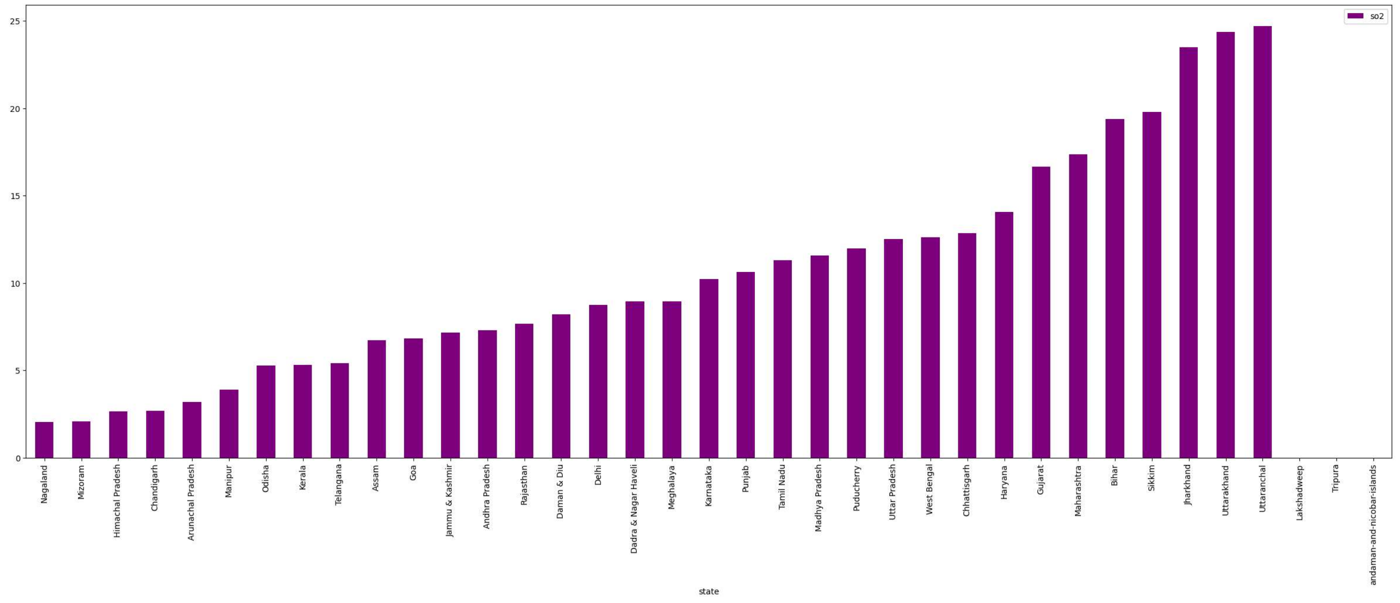
```
plt.figure(figsize=(30, 10))
plt.xticks(rotation=90)
sns.barplot(x='state',y='so2',data=df)
```

```
<Axes: xlabel='state', ylabel='so2'>
```

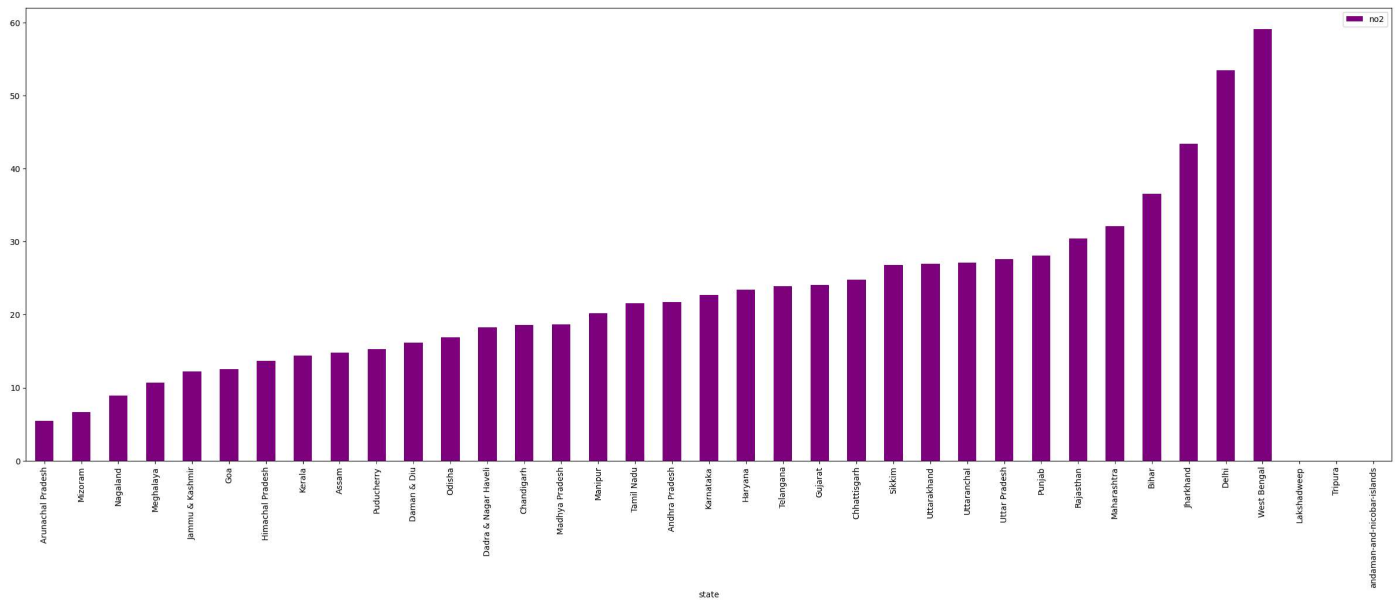


```
plt.rcParams['figure.figsize']=(30,10)
```

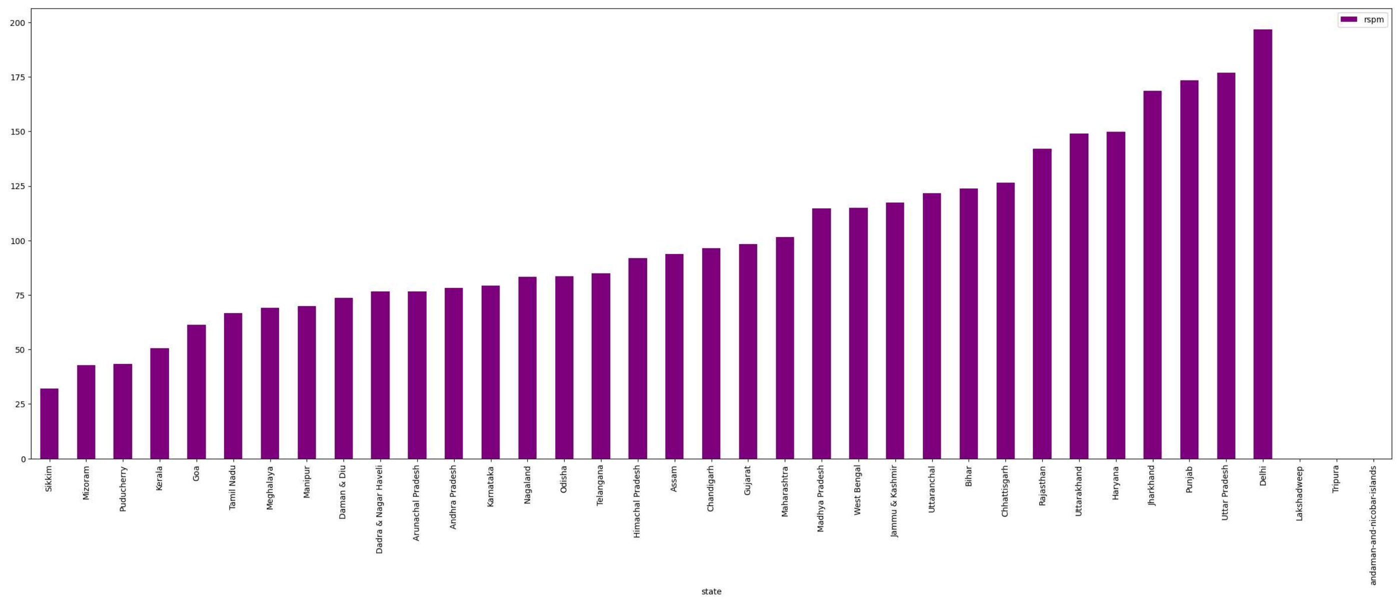
```
df[['so2','state']].groupby(["state"]).mean().sort_values(by='so2').plot.bar(color='purple')
plt.show()
```



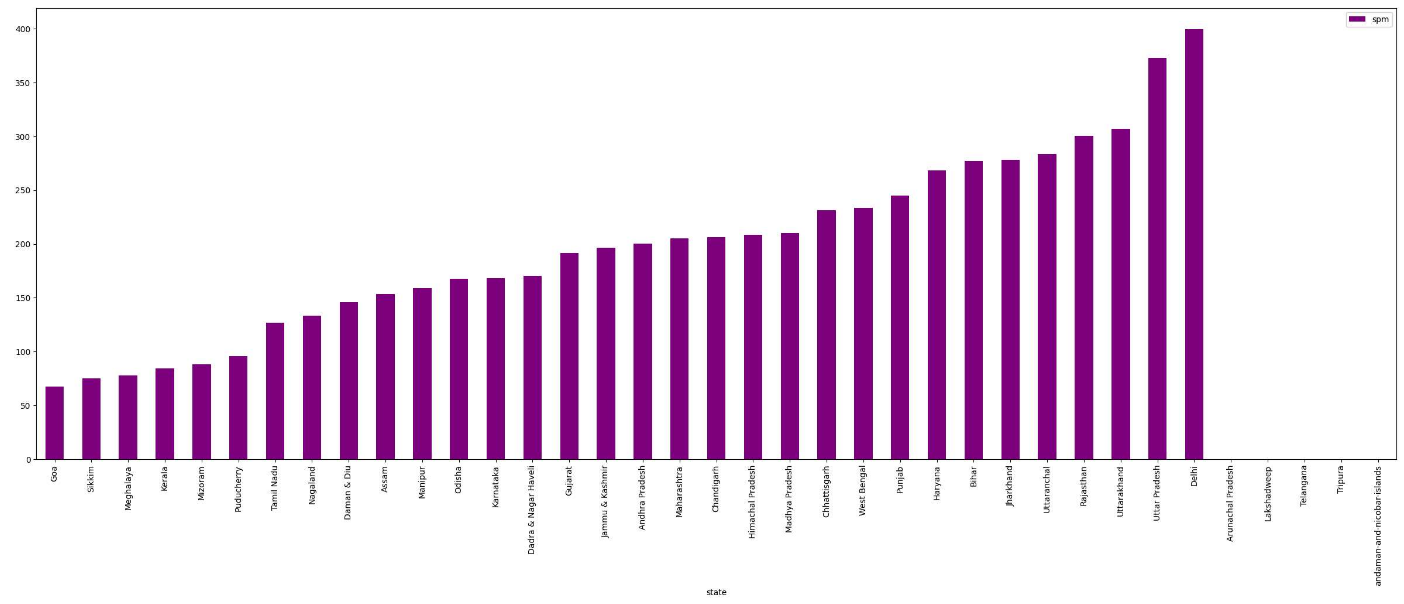
```
df[['no2','state']].groupby(["state"]).mean().sort_values(by='no2').plot.bar(color='purple')
plt.show()
```



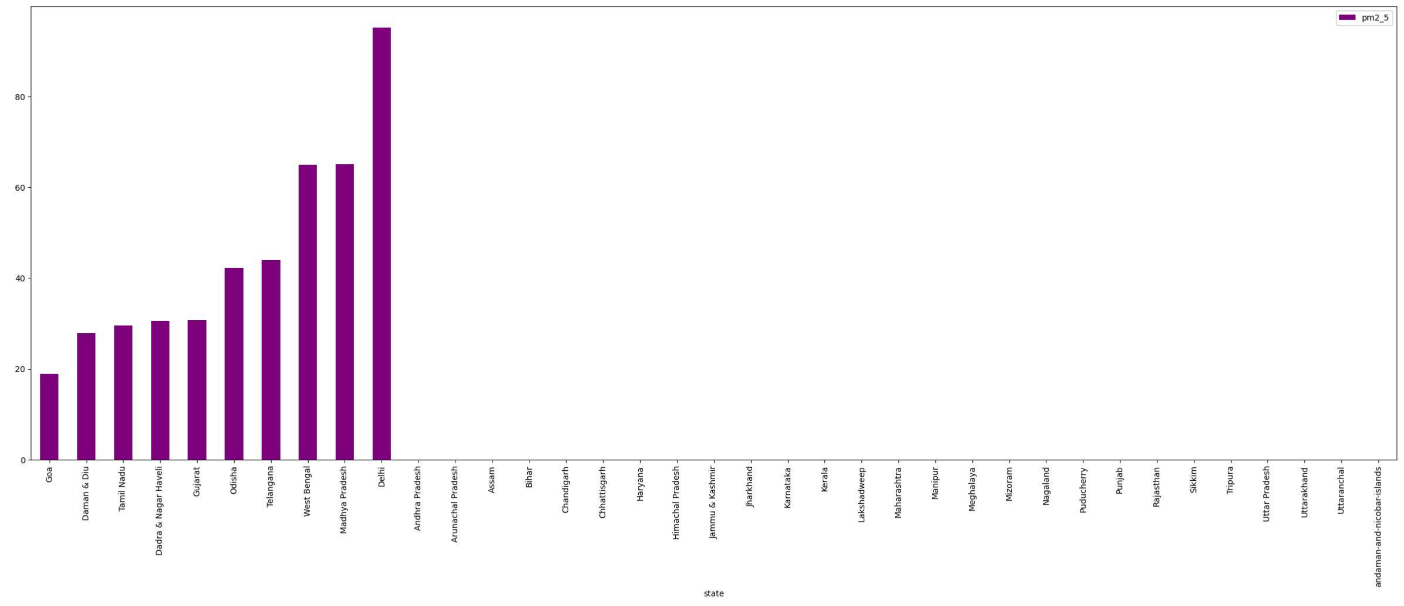
```
df[['rspm','state']].groupby(["state"]).mean().sort_values(by='rspm').plot.bar(color='purple')
plt.show()
```



```
df[['spm','state']].groupby(["state"]).mean().sort_values(by='spm').plot.bar(color='purple')
plt.show()
```



```
df[['pm2_5','state']].groupby(["state"]).mean().sort_values(by='pm2_5').plot.bar(color='purple')
plt.show()
```



```
nullvalues = df.isnull().sum().sort_values(ascending=False)
df.drop(['agency'],axis=1,inplace=True)
null_values_percentage = (df.isnull().sum()/df.isnull().count()*100).sort_values(ascending=False)
missing_data_with_percentage = pd.concat([nullvalues, null_values_percentage], axis=1, keys=['Total', 'Percent'])
missing_data_with_percentage
missing_data_with_percentage
```

```
                              Total    Percent
                     pm2_5   426428  97.862497
                       spm   237387  54.478797
                    agency   149481        NaN
                  stn_code   144077  33.064749
                      rspm    40222   9.230692
                       so2    34646   7.951035
location_monitoring_station   27491   6.309009
                       no2    16233   3.725370
                      type     5393   1.237659
                      date        7   0.001606
             sampling_date        3   0.000688
                  location        3   0.000688
                     state        0   0.000000
```

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

```
                                  0
                  stn_code   144077
             sampling_date        3
                     state        0
                  location        3
                      type     5393
                       so2    34646
                       no2    16233
                      rspm    40222
                       spm   237387
location_monitoring_station   27491
                     pm2_5   426428
                      date        7

dtype: int64
```

```
df['location']=df['location'].fillna(df['location'].mode()[0])
df['type']=df['type'].fillna(df['type'].mode()[0])
```

```
df.fillna(0, inplace=True)
```

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

	0
stn_code	0
sampling_date	0
state	0
location	0
type	0
so2	0
no2	0
rspm	0
spm	0
location_monitoring_station	0
pm2_5	0
date	0

dtype: int64

```
df
```

	stn_code	sampling_date	state	location	type	so2	no2	rspm	spm	location_monitoring_station	pm2_5	date
0	150.0	February - M021990	Andhra Pradesh	Hyderabad	Residential, Rural and other Areas	4.8	17.4	0.0	0.0	0	0.0	1990-02-01
1	151.0	February - M021990	Andhra Pradesh	Hyderabad	Industrial Area	3.1	7.0	0.0	0.0	0	0.0	1990-02-01
2	152.0	February - M021990	Andhra Pradesh	Hyderabad	Residential, Rural and other Areas	6.2	28.5	0.0	0.0	0	0.0	1990-02-01
3	150.0	March - M031990	Andhra Pradesh	Hyderabad	Residential, Rural and other Areas	6.3	14.7	0.0	0.0	0	0.0	1990-03-01
4	151.0	March - M031990	Andhra Pradesh	Hyderabad	Industrial Area	4.7	7.5	0.0	0.0	0	0.0	1990-03-01
...	...	...	...	...	...	...	...	...	...	...	...	...
435737	SAMP	24-12-15	West Bengal	ULUBERIA	RIRUO	22.0	50.0	143.0	0.0	Inside Rampal Industries,ULUBERIA	0.0	2015-12-24

```
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
df['SOi']=df['so2'].apply(cal_SOi)
data= df[['so2','SOi']]
data.head()
```

	so2	SOi
0	4.8	6.000
1	3.1	3.875
2	6.2	7.750
3	6.3	7.875
4	4.7	5.875

```
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
df['Noi']=df['no2'].apply(cal_Noi)
data= df[['no2','Noi']]
data.head()
```

	no2	Noi
0	17.4	21.750
1	7.0	8.750
2	28.5	35.625
3	14.7	18.375
4	7.5	9.375

```
def cal_RSPMI(rspm):
    rpi=0
    if(rpi<=30):
     rpi=rpi*50/30
    elif(rpi>30 and rpi<=60):
     rpi=50+(rpi-30)*50/30
    elif(rpi>60 and rpi<=90):
     rpi=100+(rpi-60)*100/30
    elif(rpi>90 and rpi<=120):
     rpi=200+(rpi-90)*100/30
    elif(rpi>120 and rpi<=250):
     rpi=300+(rpi-120)*(100/130)
    else:
     rpi=400+(rpi-250)*(100/130)
    return rpi
df['Rpi']=df['rspm'].apply(cal_RSPMI)
data= df[['rspm','Rpi']]
data.head()
```

	rspm	Rpi
0	0.0	0.0
1	0.0	0.0
2	0.0	0.0
3	0.0	0.0
4	0.0	0.0

```
def cal_SPMi(spm):
    spi=0
    if(spm<=50):
     spi=spm*50/50
    elif(spm>50 and spm<=100):
     spi=50+(spm-50)*(50/50)
    elif(spm>100 and spm<=250):
     spi= 100+(spm-100)*(100/150)
    elif(spm>250 and spm<=350):
     spi=200+(spm-250)*(100/100)
    elif(spm>350 and spm<=430):
     spi=300+(spm-350)*(100/80)
    else:
     spi=400+(spm-430)*(100/430)
    return spi

df['SPMi']=df['spm'].apply(cal_SPMi)
data= df[['spm','SPMi']]
data.head()
```

	spm	SPMi
0	0.0	0.0
1	0.0	0.0
2	0.0	0.0
3	0.0	0.0
4	0.0	0.0

```
def cal_aqi(si,ni,rspmi,spmi):
    aqi=0
    if(si>ni and si>rspmi and si>spmi):
     aqi=si
    if(ni>si and ni>rspmi and ni>spmi):
     aqi=ni
    if(rspmi>si and rspmi>ni and rspmi>spmi):
     aqi=rspmi
    if(spmi>si and spmi>ni and spmi>rspmi):
     aqi=spmi
    return aqi

df['AQI']=df.apply(lambda x:cal_aqi(x['SOi'],x['Noi'],x['Rpi'],x['SPMi']),axis=1)
data= df[['state','SOi','Noi','Rpi','SPMi','AQI']]
data.head()
```

	state	SOi	Noi	Rpi	SPMi	AQI
0	Andhra Pradesh	6.000	21.750	0.0	0.0	21.750
1	Andhra Pradesh	3.875	8.750	0.0	0.0	8.750
2	Andhra Pradesh	7.750	35.625	0.0	0.0	35.625
3	Andhra Pradesh	7.875	18.375	0.0	0.0	18.375
4	Andhra Pradesh	5.875	9.375	0.0	0.0	9.375

```
def AQI_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"

df['AQI_Range'] = df['AQI'] .apply(AQI_Range)
df.head()
```