Air pollution prediction

In [1]:

```
import numpy as np
import pandas as pd
```

In [2]:

```
import seaborn as sns
from matplotlib import pyplot as plt
%matplotlib inline
```

Retriving data FRom dataset

In [3]:

```
df = pd.read_csv('total-output-in-manufacturing-by-industry-annual.csv')
df.head()
```

Out[3]:

	year	level_1	level_2	value
0	1980	Total Manufacturing	Food, Beverage & Tobacco	2177.9
1	1980	Total Manufacturing	Textiles	432.7
2	1980	Total Manufacturing	Wearing Apparel	932.6
3	1980	Total Manufacturing	Leather, Leather Products & Footwear	115.3
4	1980	Total Manufacturing	Wood & Wood Products	806.4

In [4]:

```
df[['value']].groupby(df['level_2']).count()
```

Out[4]:

	value
level_2	
Basic Metal	37
Chemicals & Chemical Products	37
Computer, Electronic & Optical Products	37
Electrical Equipment	37
Fabricated Metal Products	37
Food, Beverage & Tobacco	37
Furniture	37
Leather, Leather Products & Footwear	37
Machinery & Equipment	37
Motor Vehicles, Trailers & Semi-trailers	37
Non-metallic Mineral Products	37
Other Manufacturing Industries	37
Other Transport Equipment	37
Paper & Paper Products	37
Pharmaceutical & Biological Products	37
Printing & Reproduction Of Recorded Media	37
Refined Petroleum Products	37
Rubber & Plastic Products	37
Textiles	37
Wearing Apparel	37
Wood & Wood Products	37

shape

In [5]:

```
df.shape
```

Out[5]:

```
(777, 4)
```

unique items in each columns

In [6]:

```
df['year'].unique()
```

Out[6]:

```
array([1980, 1981, 1982, 1983, 1984, 1985, 1986, 1987, 1988, 1989, 1990,
       1991, 1992, 1993, 1994, 1995, 1996, 1997, 1998, 1999, 2000, 2001,
       2002, 2003, 2004, 2005, 2006, 2007, 2008, 2009, 2010, 2011, 2012,
       2013, 2014, 2015, 2016], dtype=int64)
```

In [7]:

```
df['level_1'].unique()
```

Out[7]:

```
array(['Total Manufacturing'], dtype=object)
```

In [8]:

```
df['level_2'].unique()
```

Out[8]:

```
array(['Food, Beverage & Tobacco', 'Textiles', 'Wearing Apparel',
       'Leather, Leather Products & Footwear', 'Wood & Wood Products',
       'Paper & Paper Products',
       'Printing & Reproduction Of Recorded Media',
       'Refined Petroleum Products', 'Chemicals & Chemical Products',
       'Pharmaceutical & Biological Products',
       'Rubber & Plastic Products', 'Non-metallic Mineral Products',
       'Basic Metal', 'Fabricated Metal Products',
       'Computer, Electronic & Optical Products', 'Electrical Equipment',
       'Machinery & Equipment',
       'Motor Vehicles, Trailers & Semi-trailers',
       'Other Transport Equipment', 'Furniture',
       'Other Manufacturing Industries'], dtype=object)
```

describe manufaturing dataset

In [9]:

```
df.describe()
```

Out[9]:

	year	value
count	777.000000	777.000000
mean	1998.000000	7120.296396
std	10.683956	15443.333714
min	1980.000000	50.400000
25%	1989.000000	629.700000
50%	1998.000000	1741.200000
75%	2007.000000	5135.900000
max	2016.000000	101827.600000

In [10]:

```
df.columns
```

Out[10]:

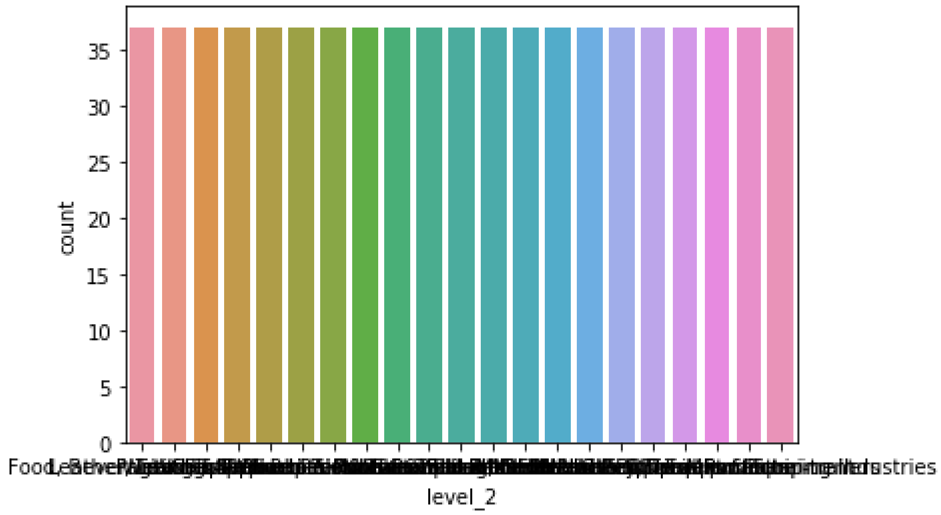
```
Index(['year', 'level_1', 'level_2', 'value'], dtype='object')
```

In [11]:

```
sns.countplot(x='level_2',data = df)
```

Out[11]:

```
<matplotlib.axes._subplots.AxesSubplot at 0x25006ba68d0>
```



In [12]:

```
df.corr()
```

Out[12]:

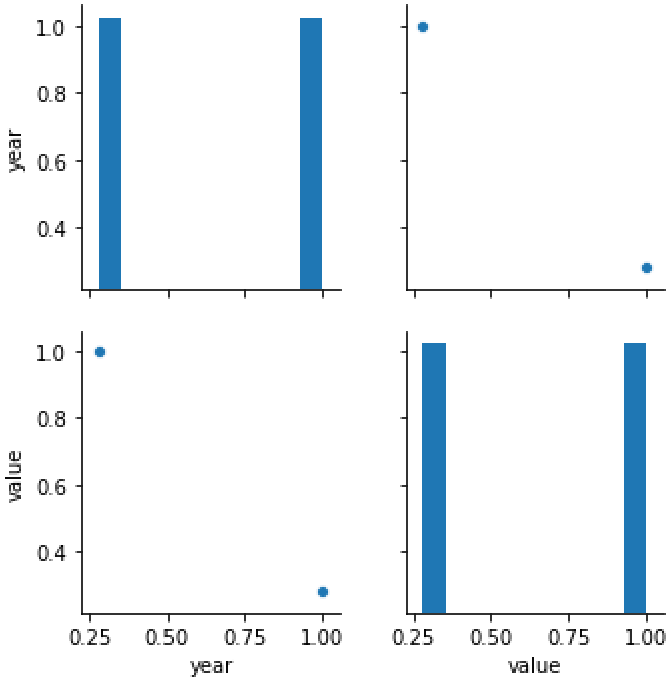
	year	value
year	1.000000	0.280772
value	0.280772	1.000000

In [13]:

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

Out[13]:

```
<seaborn.axisgrid.PairGrid at 0x25006f3ff98>
```

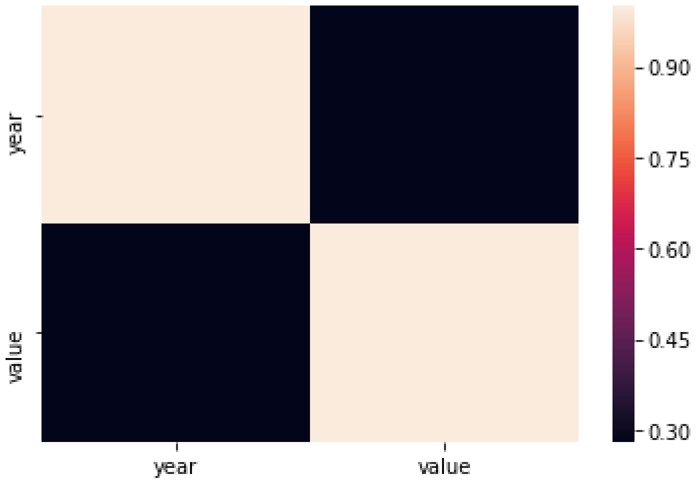


In [14]:

```
sns.heatmap(df.corr())
```

Out[14]:

```
<matplotlib.axes._subplots.AxesSubplot at 0x250071857f0>
```



In [15]:

```
df = df[['value']].groupby(df['year']).mean()
```

In [16]:

```
df
```

Out[16]:

	value
year	
1980	1546.623810
1981	1793.019048
1982	1778.638095
1983	1818.295238
1984	2003.004762
1985	1874.052381
1986	1820.347619
1987	2250.019048
1988	2754.585714
1989	3112.933333
1990	3490.933333
1991	3650.966667
1992	3779.295238
1993	4289.247619
1994	4934.671429
1995	5549.561905
1996	5864.538095
1997	6193.723810
1998	5963.195238
1999	6549.642857
2000	7827.557143
2001	6623.809524
2002	7051.261905
2003	7594.871429
2004	9167.819048
2005	10388.038095
2006	11385.985714
2007	12127.214286
2008	12619.338095
2009	10843.247619
2010	13066.823810
2011	14072.819048
2012	14359.557143

	value
year	
2013	14267.795238
2014	14601.995238
2015	13556.723810
2016	12878.814286

In [17]:

```
df.tail()
```

Out[17]:

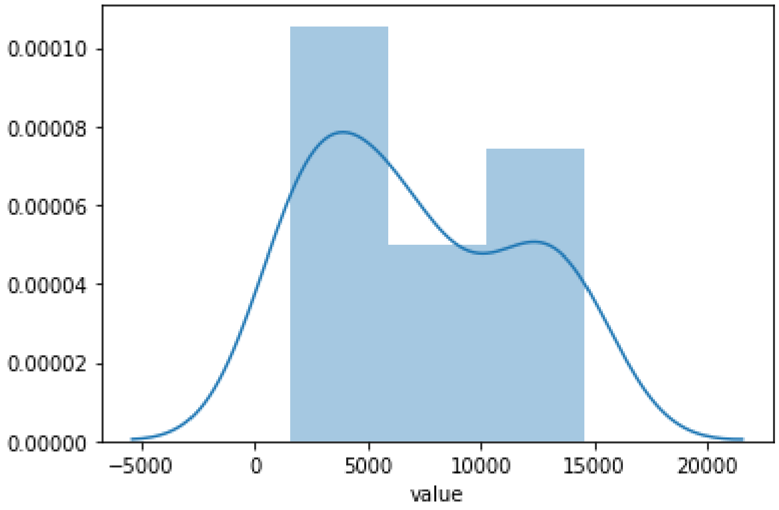
	value
year	
2012	14359.557143
2013	14267.795238
2014	14601.995238
2015	13556.723810
2016	12878.814286

In [18]:

```
sns.distplot(df['value'],kde=True)
```

Out[18]:

```
<matplotlib.axes._subplots.AxesSubplot at 0x2500726c828>
```



Retrive annual-motor-vehicle-population-by-vehicle-type data

In [19]:

```
df_motor_vehicle = pd.read_csv('annual-motor-vehicle-population-by-vehicle-type.csv')
df_motor_vehicle.head()
```

Out[19]:

	year	category	type	number
0	2005	Cars & Station-wagons	Private cars	401638
1	2006	Cars & Station-wagons	Private cars	421904
2	2007	Cars & Station-wagons	Private cars	451745
3	2008	Cars & Station-wagons	Private cars	476634
4	2009	Cars & Station-wagons	Private cars	497116

In [20]:

```
dfn = pd.pivot_table(df_motor_vehicle,values='number',index=['year','category'],margins=Tru
```

In [21]:

```
dfn.shape
```

Out[21]:

```
(79, 1)
```

In [22]:

```
dfn.head()
```

Out[22]:

		number
year	**category**	
2005	**Buses**	13220
	Cars & Station-wagons	438194
	Goods & Other Vehicles	128193
	Motorcycles	138588
	Tax Exempted Vehicles	14414

In [23]:

```
dfn.tail()
```

Out[23]:

		number
year	**category**	
2017	**Goods and Other Vehicles**	142857
	Motorcycles and Scooters	141304
	Tax Exempted Vehicles	23471
	Taxis	23140
year		11920069

In [24]:

```
dfn.drop(['year'],axis=0,inplace=True)
```

In [25]:

```
list_year = []
list_category = []
for i in range(dfn.shape[0]):
    list_year.append(dfn.index[i][0])
    list_category.append(dfn.index[i][1])
```

In [26]:

```
df_motor_vehicle = pd.DataFrame({'year':list_year,'category':list_category,'number':dfn['nu
df_motor_vehicle.index = range(0,dfn.shape[0])
df_motor_vehicle.head()
```

Out[26]:

	year	category	number
0	2005	Buses	13220
1	2005	Cars & Station-wagons	438194
2	2005	Goods & Other Vehicles	128193
3	2005	Motorcycles	138588
4	2005	Tax Exempted Vehicles	14414

In [27]:

```
df_motor_vehicle.tail()
```

Out[27]:

	year	category	number
73	2017	Cars and Station-wagons	612256
74	2017	Goods and Other Vehicles	142857
75	2017	Motorcycles and Scooters	141304
76	2017	Tax Exempted Vehicles	23471
77	2017	Taxis	23140

merging total-output-in-manufacturing-by-industry-annual and annual-motor-vehicle-population-by-vehicle-type

In [28]:

```
df = df.merge(df_motor_vehicle,on='year',how='outer')
```

Retrive air-polluant-lead data

In [29]:

```
df_lead = pd.read_csv('air-polluant-lead.csv')
df_lead
```

Out[29]:

	year	air_pollutant_lead_mean
0	2006	0.017
1	2007	0.019
2	2008	0.018
3	2009	0.010
4	2010	0.009
5	2011	0.011
6	2012	0.008
7	2013	0.009
8	2014	0.016

In [30]:

```
df_lead = df_lead.append({'year':2015,'air_pollutant_lead_mean':0.016},ignore_index=True)
df_lead = df_lead.append({'year':2016,'air_pollutant_lead_mean':0.019},ignore_index=True)
df_lead
```

Out[30]:

	year	air_pollutant_lead_mean
0	2006.0	0.017
1	2007.0	0.019
2	2008.0	0.018
3	2009.0	0.010
4	2010.0	0.009
5	2011.0	0.011
6	2012.0	0.008
7	2013.0	0.009
8	2014.0	0.016
9	2015.0	0.016
10	2016.0	0.019

In [31]:

```
df_lead['year']  = df_lead['year'].astype('int32')
df_lead
```

Out[31]:

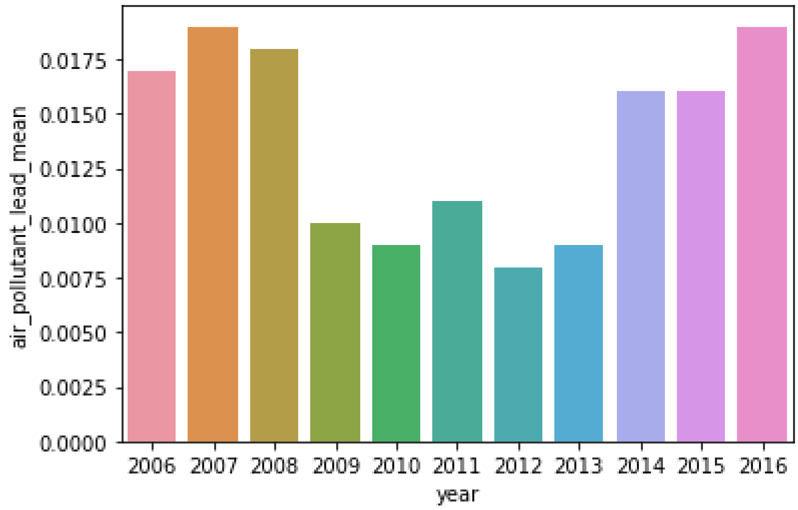
	year	air_pollutant_lead_mean
0	2006	0.017
1	2007	0.019
2	2008	0.018
3	2009	0.010
4	2010	0.009
5	2011	0.011
6	2012	0.008
7	2013	0.009
8	2014	0.016
9	2015	0.016
10	2016	0.019

In [32]:

```
sns.barplot(x='year',y='air_pollutant_lead_mean',data=df_lead,estimator=np.mean)
```

Out[32]:

```
<matplotlib.axes._subplots.AxesSubplot at 0x25007323208>
```



In [33]:

```
dict1 = {}
for i,j in zip(df_lead.iloc[:,0],df_lead.iloc[:,1]):
    dict1[i] = j
```

In [34]:

```
dict1
```

Out[34]:

```
{2006: 0.017,
 2007: 0.019,
 2008: 0.018000000000000002,
 2009: 0.01,
 2010: 0.009000000000000001,
 2011: 0.011000000000000001,
 2012: 0.008,
 2013: 0.009000000000000001,
 2014: 0.016,
 2015: 0.016,
 2016: 0.019}
```

In [35]:

```
obj = lambda x:dict1[x] if(x in dict1) else np.nan
```

In [36]:

```
df['air-polluant-lead']=df['year'].apply(obj)
```

In [37]:

```
df[df['year']>=2006].head()
```

Out[37]:

	year	value	category	number	air-polluant-lead
31	2006	11385.985714	Buses	13831.0	0.017
32	2006	11385.985714	Cars & Station-wagons	472308.0	0.017
33	2006	11385.985714	Goods & Other Vehicles	132841.0	0.017
34	2006	11385.985714	Motorcycles	141881.0	0.017
35	2006	11385.985714	Tax Exempted Vehicles	15178.0	0.017

In [38]:

```
df.loc[df['year']==2005,'air-polluant-lead']=0.017
df.loc[df['year']==2015,'air-polluant-lead']=0.016
df.loc[df['year']==2016,'air-polluant-lead']=0.019
```

retrive air-pollutant-carbon-monoxide-2nd-maximum-8-hour-mean data

In [39]:

```
df_carbon_monoxide = pd.read_csv('air-pollutant-carbon-monoxide-2nd-maximum-8-hour-mean.csv
```

In [40]:

```
df_carbon_monoxide
```

Out[40]:

	year	carbon_monoxide_2nd_maximum_8hourly_mean
0	1999	3.6
1	2000	3.7
2	2001	4.2
3	2002	2.8
4	2003	3.1
5	2004	2.8
6	2005	2.4
7	2006	2.6
8	2007	1.7
9	2008	1.5
10	2009	1.7
11	2010	2.2
12	2011	2.0
13	2012	1.9
14	2013	5.5
15	2014	1.8

In [41]:

```
dict2 = {}
for i,j in zip(df_carbon_monoxide.iloc[:,0],df_carbon_monoxide.iloc[:,1]):
    dict2[i] = j
dict2
```

Out[41]:

```
{1999: 3.6,
 2000: 3.7,
 2001: 4.2,
 2002: 2.8,
 2003: 3.1,
 2004: 2.8,
 2005: 2.4,
 2006: 2.6,
 2007: 1.7,
 2008: 1.5,
 2009: 1.7,
 2010: 2.2,
 2011: 2.0,
 2012: 1.9,
 2013: 5.5,
 2014: 1.8}
```

In [42]:

```
df_carbon_monoxide = pd.DataFrame({'year':list(dict2.keys()),'carbon_monoxide_2nd_maximum_8
df_carbon_monoxide = df_carbon_monoxide[(df_carbon_monoxide['year']>=2005) & (df_carbon_mon
df_carbon_monoxide
```

Out[42]:

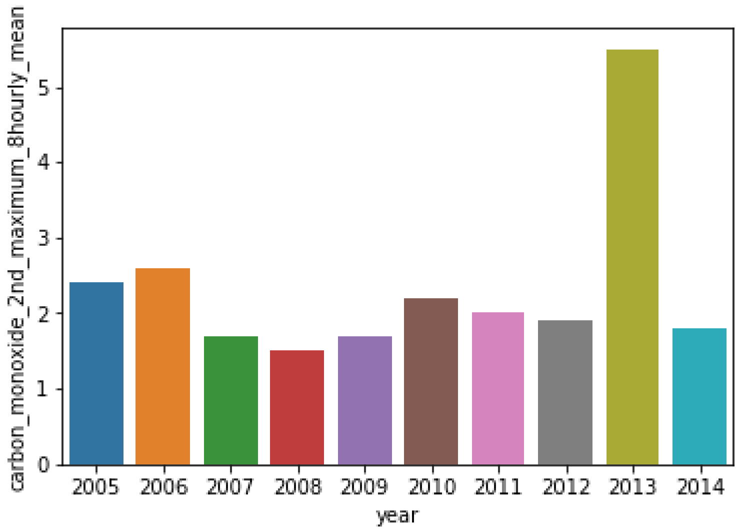
	year	carbon_monoxide_2nd_maximum_8hourly_mean
6	2005	2.4
7	2006	2.6
8	2007	1.7
9	2008	1.5
10	2009	1.7
11	2010	2.2
12	2011	2.0
13	2012	1.9
14	2013	5.5
15	2014	1.8

In [43]:

```
sns.barplot(x = 'year',y='carbon_monoxide_2nd_maximum_8hourly_mean',data=df_carbon_monoxide
```

Out[43]:

```
<matplotlib.axes._subplots.AxesSubplot at 0x250073d1908>
```



In [44]:

```
obj2 = lambda x:dict2[x] if x in dict2 else np.nan
```

In [45]:

```
df['air-pollutant-carbon-monoxide-2nd-maximum-8-hour-mean']=df['year'].apply(obj)
```

In [46]:

```
df.head()
```

Out[46]:

	year	value	category	number	air-polluant-lead	air-pollutant-carbon-monoxide-2nd-maximum-8-hour-mean
0	1980	1546.623810	NaN	NaN	NaN	NaN
1	1981	1793.019048	NaN	NaN	NaN	NaN
2	1982	1778.638095	NaN	NaN	NaN	NaN
3	1983	1818.295238	NaN	NaN	NaN	NaN
4	1984	2003.004762	NaN	NaN	NaN	NaN

In [47]:

```
df.loc[df['year']==2015,'air-pollutant-carbon-monoxide-2nd-maximum-8-hour-mean']=2.1
df.loc[df['year']==2016,'air-pollutant-carbon-monoxide-2nd-maximum-8-hour-mean']=2.5
```

In [48]:

```
df[df['year']>=2006].head()
```

Out[48]:

	year	value	category	number	air-polluant-lead	air-pollutant-carbon-monoxide-2nd-maximum-8-hour-mean
31	2006	11385.985714	Buses	13831.0	0.017	0.017
32	2006	11385.985714	Cars & Station-wagons	472308.0	0.017	0.017
33	2006	11385.985714	Goods & Other Vehicles	132841.0	0.017	0.017
34	2006	11385.985714	Motorcycles	141881.0	0.017	0.017
35	2006	11385.985714	Tax Exempted Vehicles	15178.0	0.017	0.017

Retriving air-pollutant-nitrogen-dioxide data

In [49]:

```
df_nitrogen_dioxide = pd.read_csv('air-pollutant-nitrogen-dioxide.csv')
df_nitrogen_dioxide
```

Out[49]:

	year	nitrogen_dioxide_mean
0	1999	36
1	2000	30
2	2001	26
3	2002	27
4	2003	24
5	2004	26
6	2005	25
7	2006	24
8	2007	22
9	2008	22
10	2009	22
11	2010	23
12	2011	25
13	2012	25
14	2013	25
15	2014	24

In [50]:

```
dict3 = {}
for i,j in zip(df_nitrogen_dioxide.iloc[:,0],df_nitrogen_dioxide.iloc[:,1]):
    dict3[i] = j
dict3[2015] = 25
dict3[2016] = 24
dict3
```

Out[50]:

```
{1999: 36,
 2000: 30,
 2001: 26,
 2002: 27,
 2003: 24,
 2004: 26,
 2005: 25,
 2006: 24,
 2007: 22,
 2008: 22,
 2009: 22,
 2010: 23,
 2011: 25,
 2012: 25,
 2013: 25,
 2014: 24,
 2015: 25,
 2016: 24}
```

In [51]:

```
df_nitrogen_dioxide = pd.DataFrame({'year':list(dict3.keys()),'nitrogen_dioxide_mean':list(
df_nitrogen_dioxide = df_nitrogen_dioxide[(df_nitrogen_dioxide['year']>=2005) & (df_nitroge
df_nitrogen_dioxide
```

Out[51]:

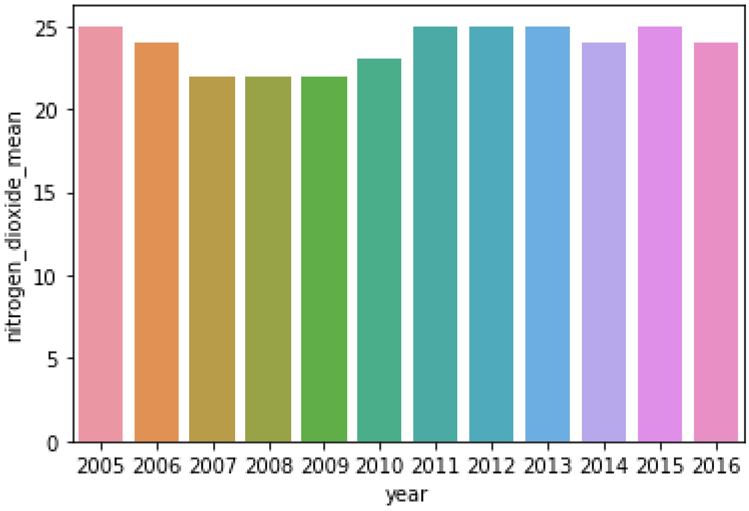
	year	nitrogen_dioxide_mean
6	2005	25
7	2006	24
8	2007	22
9	2008	22
10	2009	22
11	2010	23
12	2011	25
13	2012	25
14	2013	25
15	2014	24
16	2015	25
17	2016	24

In [52]:

```
sns.barplot(x = 'year',y='nitrogen_dioxide_mean',data=df_nitrogen_dioxide)
```

Out[52]:

```
<matplotlib.axes._subplots.AxesSubplot at 0x250072beb38>
```



In [53]:

```
obj3 = lambda x:dict3[x] if(x in dict3) else np.nan
```

In [54]:

```
df['air-pollutant-nitrogen-dioxide'] = df['year'].apply(obj3)
```

In [55]:

```
df.tail()
```

Out[55]:

	year	value	category	number	air-pollutant-lead	air-pollutant-carbon-monoxide-2nd-maximum-8-hour-mean	air-pollutant-nitrogen-dioxide
98	2017	NaN	Cars and Station-wagons	612256.0	NaN	NaN	NaN
99	2017	NaN	Goods and Other Vehicles	142857.0	NaN	NaN	NaN
100	2017	NaN	Motorcycles and Scooters	141304.0	NaN	NaN	NaN
101	2017	NaN	Tax Exempted Vehicles	23471.0	NaN	NaN	NaN
102	2017	NaN	Taxis	23140.0	NaN	NaN	NaN

Retriving air-pollutant-ozone data

In [56]:

```
df_ozone = pd.read_csv('air-pollutant-ozone.csv')
df_ozone
```

Out[56]:

	year	ozone_4th_maximum_8hourly_mean
0	1999	125
1	2000	108
2	2001	126
3	2002	114
4	2003	108
5	2004	143
6	2005	155
7	2006	127
8	2007	140
9	2008	103
10	2009	100
11	2010	129
12	2011	110
13	2012	122
14	2013	139
15	2014	135

In [57]:

```
dict4 = {}
for i,j in zip(df_ozone.iloc[:,0],df_ozone.iloc[:,1]):
    dict4[i] = j
dict4[2015] = 129
dict4[2016] = 131
dict4
```

Out[57]:

```
{1999: 125,
 2000: 108,
 2001: 126,
 2002: 114,
 2003: 108,
 2004: 143,
 2005: 155,
 2006: 127,
 2007: 140,
 2008: 103,
 2009: 100,
 2010: 129,
 2011: 110,
 2012: 122,
 2013: 139,
 2014: 135,
 2015: 129,
 2016: 131}
```

In [58]:

```
df_ozone = pd.DataFrame({'year':list(dict4.keys()),'ozone_4th_maximum_8hourly_mean':list(di
df_ozone = df_ozone[(df_ozone['year']>=2005) & (df_ozone['year']<=2016)]
df_ozone
```

Out[58]:

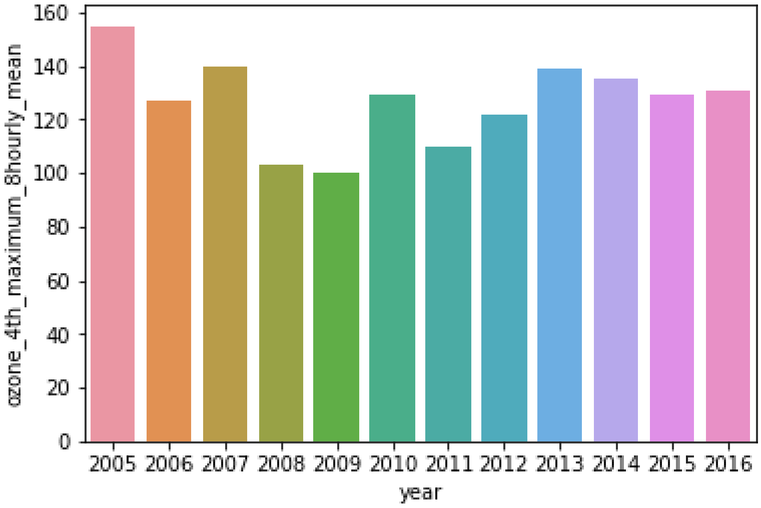
	year	ozone_4th_maximum_8hourly_mean
6	2005	155
7	2006	127
8	2007	140
9	2008	103
10	2009	100
11	2010	129
12	2011	110
13	2012	122
14	2013	139
15	2014	135
16	2015	129
17	2016	131

In [59]:

```
sns.barplot(x = 'year',y='ozone_4th_maximum_8hourly_mean',data=df_ozone)
```

Out[59]:

```
<matplotlib.axes._subplots.AxesSubplot at 0x250075374e0>
```



In [60]:

```
obj4 = lambda x:dict4[x] if(x in dict4) else np.nan
```

In [61]:

```
df['air-pollutant-ozone'] = df['year'].apply(obj4)
```

In [62]:

```
df.tail()
```

Out[62]:

	year	value	category	number	air-polluant-lead	air-pollutant-carbon-monoxide-2nd-maximum-8-hour-mean	air-pollutant-nitrogen-dioxide	air-pollutant-ozone
98	2017	NaN	Cars and Station-wagons	612256.0	NaN	NaN	NaN	NaN
99	2017	NaN	Goods and Other Vehicles	142857.0	NaN	NaN	NaN	NaN
100	2017	NaN	Motorcycles and Scooters	141304.0	NaN	NaN	NaN	NaN
101	2017	NaN	Tax Exempted Vehicles	23471.0	NaN	NaN	NaN	NaN
102	2017	NaN	Taxis	23140.0	NaN	NaN	NaN	NaN

Retrive air-pollutant-particulate-matter-pm2-5 data

In [63]:

```
df_pm2_5 = pd.read_csv('air-pollutant-particulate-matter-pm2-5.csv')
df_pm2_5
```

Out[63]:

	year	pm2.5_mean
0	2002	23
1	2003	19
2	2004	21
3	2005	21
4	2006	23
5	2007	19
6	2008	16
7	2009	19
8	2010	17
9	2011	17
10	2012	19
11	2013	20
12	2014	18

In [64]:

```
dict5 = {}
for i,j in zip(df_pm2_5.iloc[:,0],df_pm2_5.iloc[:,1]):
    dict5[i] = j
#dict5[2000]=19
#dict5[2001]=21
dict5[2015] = 17
dict5[2016] = 19
dict5
```

Out[64]:

```
{2002: 23,
 2003: 19,
 2004: 21,
 2005: 21,
 2006: 23,
 2007: 19,
 2008: 16,
 2009: 19,
 2010: 17,
 2011: 17,
 2012: 19,
 2013: 20,
 2014: 18,
 2015: 17,
 2016: 19}
```

In [65]:

```
df_pm2_5 = pd.DataFrame({'year':list(dict5.keys()),'pm2.5_mean':list(dict5.values())})
df_pm2_5 = df_pm2_5[(df_pm2_5['year']>=2005) & (df_pm2_5['year']<=2016)]
df_pm2_5
```

Out[65]:

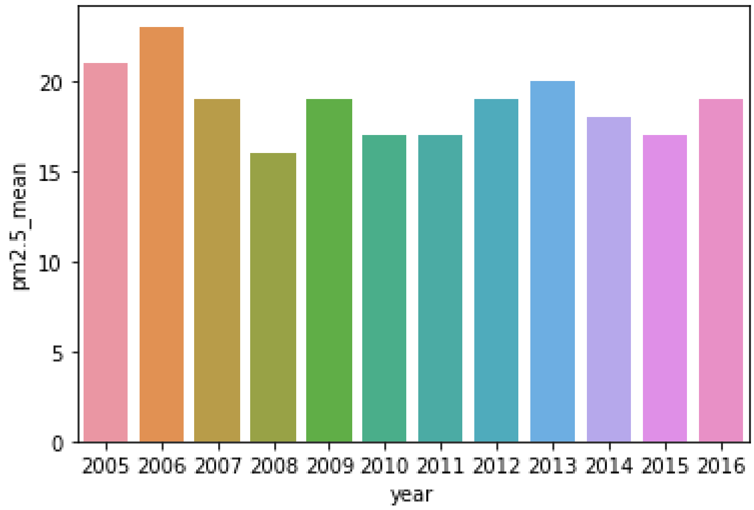
	year	pm2.5_mean
3	2005	21
4	2006	23
5	2007	19
6	2008	16
7	2009	19
8	2010	17
9	2011	17
10	2012	19
11	2013	20
12	2014	18
13	2015	17
14	2016	19

In [66]:

```
sns.barplot(x = 'year',y='pm2.5_mean',data=df_pm2_5)
```

Out[66]:

```
<matplotlib.axes._subplots.AxesSubplot at 0x25007598c50>
```



In [67]:

```
obj5 = lambda x:dict5[x] if(x in dict5) else np.nan
df['air-pollutant-particulate-matter-pm2-5'] = df['year'].apply(obj5)
df.tail()
```

Out[67]:

	year	value	category	number	air-pollutant-lead	air-pollutant-carbon-monoxide-2nd-maximum-8-hour-mean	air-pollutant-nitrogen-dioxide	air-pollutant-ozone	air-pollutant-particulate-matter-pm2-5
98	2017	NaN	Cars and Station-wagons	612256.0	NaN	NaN	NaN	NaN	NaN
99	2017	NaN	Goods and Other Vehicles	142857.0	NaN	NaN	NaN	NaN	NaN
100	2017	NaN	Motorcycles and Scooters	141304.0	NaN	NaN	NaN	NaN	NaN
101	2017	NaN	Tax Exempted Vehicles	23471.0	NaN	NaN	NaN	NaN	NaN
102	2017	NaN	Taxis	23140.0	NaN	NaN	NaN	NaN	NaN

Retrive air-pollutant-particulate-matter-pm10 data

In [68]:

```
df_pm10 = pd.read_csv('air-pollutant-particulate-matter-pm10.csv')
```

In [69]:

```
df_pm10
```

Out[69]:

	year	pm10_2nd_maximum_24hourly_mean
0	1999	139
1	2000	89
2	2001	80
3	2002	142
4	2003	83
5	2004	85
6	2005	101
7	2006	228
8	2007	69
9	2008	57
10	2009	77
11	2010	127
12	2011	55
13	2012	57
14	2013	215
15	2014	75

In [70]:

```
dict6 = {}
for i,j in zip(df_pm10.iloc[:,0],df_pm10.iloc[:,1]):
    dict6[i] = j
dict6[2015] = 80
dict6[2016] = 95
dict6
```

Out[70]:

```
{1999: 139,
 2000: 89,
 2001: 80,
 2002: 142,
 2003: 83,
 2004: 85,
 2005: 101,
 2006: 228,
 2007: 69,
 2008: 57,
 2009: 77,
 2010: 127,
 2011: 55,
 2012: 57,
 2013: 215,
 2014: 75,
 2015: 80,
 2016: 95}
```

In [71]:

```
df_pm10 = pd.DataFrame({'year':list(dict6.keys()),'pm10_2nd_maximum_24hourly_mean':list(dic
df_pm10 = df_pm10[(df_pm10['year']>=2005) & (df_pm10['year']<=2016)]
df_pm10
```

Out[71]:

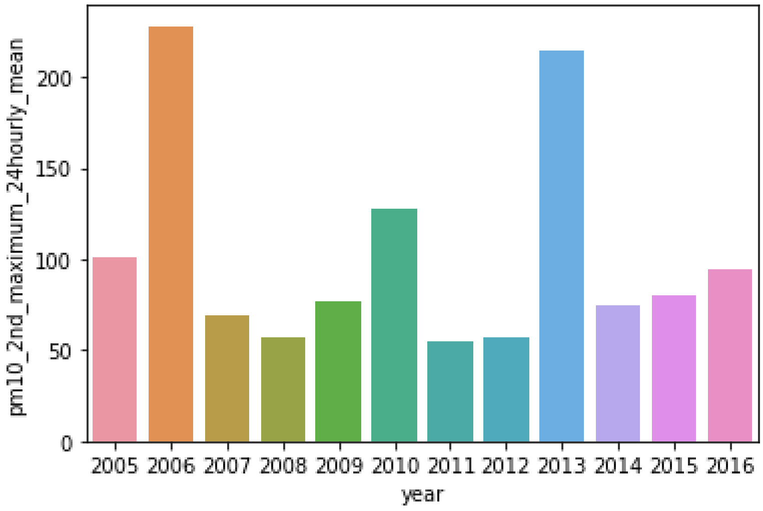
	year	pm10_2nd_maximum_24hourly_mean
6	2005	101
7	2006	228
8	2007	69
9	2008	57
10	2009	77
11	2010	127
12	2011	55
13	2012	57
14	2013	215
15	2014	75
16	2015	80
17	2016	95

In [72]:

```
sns.barplot(x = 'year',y='pm10_2nd_maximum_24hourly_mean',data=df_pm10)
```

Out[72]:

```
<matplotlib.axes._subplots.AxesSubplot at 0x250073c5198>
```



In [73]:

```
obj6 = lambda x:dict6[x] if(x in dict6) else np.nan
df['air-pollutant-particulate-matter-pm10'] = df['year'].apply(obj6)
df.tail()
```

Out[73]:

	year	value	category	number	air-pollutant-lead	air-pollutant-carbon-monoxide-2nd-maximum-8-hour-mean	air-pollutant-nitrogen-dioxide	air-pollutant-ozone	air-pollutant-particulate-matter-pm2-5
98	2017	NaN	Cars and Station-wagons	612256.0	NaN	NaN	NaN	NaN	NaN
99	2017	NaN	Goods and Other Vehicles	142857.0	NaN	NaN	NaN	NaN	NaN
100	2017	NaN	Motorcycles and Scooters	141304.0	NaN	NaN	NaN	NaN	NaN
101	2017	NaN	Tax Exempted Vehicles	23471.0	NaN	NaN	NaN	NaN	NaN
102	2017	NaN	Taxis	23140.0	NaN	NaN	NaN	NaN	NaN

Retriving air-pollutant-sulphur-dioxide data

In [74]:

```
df_sulphur_dioxide = pd.read_csv('air-pollutant-sulphur-dioxide.csv')
df_sulphur_dioxide
```

Out[74]:

	year	sulphur_dioxide_mean
0	1999	22
1	2000	22
2	2001	22
3	2002	18
4	2003	15
5	2004	14
6	2005	14
7	2006	11
8	2007	12
9	2008	11
10	2009	9
11	2010	11
12	2011	10
13	2012	13
14	2013	14
15	2014	12

In [75]:

```
dict7 = {}
for i,j in zip(df_sulphur_dioxide.iloc[:,0],df_sulphur_dioxide.iloc[:,1]):
    dict7[i] = j
dict7[2015] = 11
dict7[2016] = 13
dict7
```

Out[75]:

```
{1999: 22,
 2000: 22,
 2001: 22,
 2002: 18,
 2003: 15,
 2004: 14,
 2005: 14,
 2006: 11,
 2007: 12,
 2008: 11,
 2009: 9,
 2010: 11,
 2011: 10,
 2012: 13,
 2013: 14,
 2014: 12,
 2015: 11,
 2016: 13}
```

In [76]:

```
df_sulphur_dioxide = pd.DataFrame({'year':list(dict7.keys()),'sulphur_dioxide_mean':list(di
df_sulphur_dioxide = df_sulphur_dioxide[(df_pm10['year']>=2005) & (df_sulphur_dioxide['year
df_sulphur_dioxide
```

Out[76]:

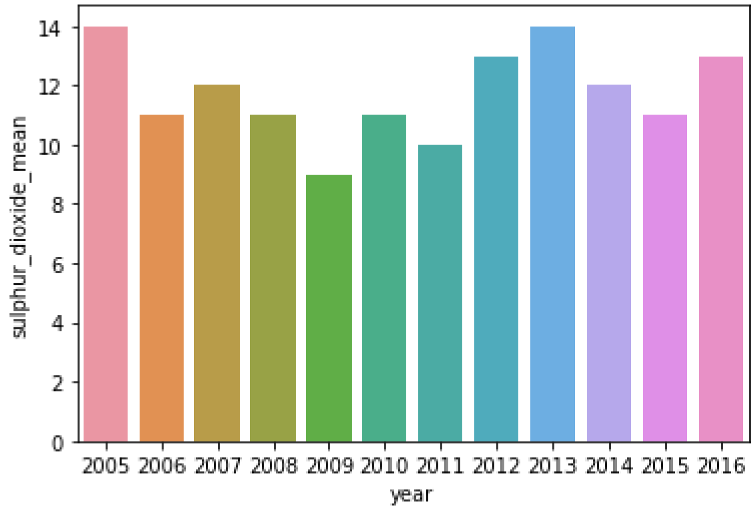
	year	sulphur_dioxide_mean
6	2005	14
7	2006	11
8	2007	12
9	2008	11
10	2009	9
11	2010	11
12	2011	10
13	2012	13
14	2013	14
15	2014	12
16	2015	11
17	2016	13

In [77]:

```
sns.barplot(x = 'year',y='sulphur_dioxide_mean',data=df_sulphur_dioxide)
```

Out[77]:

```
<matplotlib.axes._subplots.AxesSubplot at 0x25007713c50>
```



In [78]:

```
obj7 = lambda x:dict7[x] if(x in dict7) else np.nan
df['air-pollutant-sulphur-dioxide'] = df['year'].apply(obj7)
df.tail()
```

Out[78]:

	year	value	category	number	air-pollutant-lead	air-pollutant-carbon-monoxide-2nd-maximum-8-hour-mean	air-pollutant-nitrogen-dioxide	air-pollutant-ozone	air-pollutant-particulate-matter-pm2-5
98	2017	NaN	Cars and Station-wagons	612256.0	NaN	NaN	NaN	NaN	NaN
99	2017	NaN	Goods and Other Vehicles	142857.0	NaN	NaN	NaN	NaN	NaN
100	2017	NaN	Motorcycles and Scooters	141304.0	NaN	NaN	NaN	NaN	NaN
101	2017	NaN	Tax Exempted Vehicles	23471.0	NaN	NaN	NaN	NaN	NaN
102	2017	NaN	Taxis	23140.0	NaN	NaN	NaN	NaN	NaN

check columns has null value

In [79]:

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

Out[79]:

```
year                                                  0
value                                                 6
category                                             25
number                                               25
air-polluant-lead                                    31
air-pollutant-carbon-monoxide-2nd-maximum-8-hour-mean    37
air-pollutant-nitrogen-dioxide                       25
air-pollutant-ozone                                  25
air-pollutant-particulate-matter-pm2-5               28
air-pollutant-particulate-matter-pm10                25
air-pollutant-sulphur-dioxide                        25
dtype: int64
```

delete row which has null value

In [80]:

```
df.dropna(axis= 0,inplace = True)
```

In [81]:

```
df
```

Out[81]:

	year	value	category	number	air-polluant-lead	air-pollutant-carbon-monoxide-2nd-maximum-8-hour-mean	air-pollutant-nitrogen-dioxide	air-pollutant-ozone	air-pollutant-particulate-matter-pm2-5	poll partic m
31	2006	11385.985714	Buses	13831.0	0.017	0.017	24.0	127.0	23.0	
32	2006	11385.985714	Cars & Station-wagons	472308.0	0.017	0.017	24.0	127.0	23.0	
33	2006	11385.985714	Goods & Other Vehicles	132841.0	0.017	0.017	24.0	127.0	23.0	
34	2006	11385.985714	Motorcycles	[illegible]	0.017	0.017	24.0	127.0	23.0	

In [82]:

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

Out[82]:

```
year                                                      0
value                                                     0
category                                                  0
number                                                    0
air-polluant-lead                                         0
air-pollutant-carbon-monoxide-2nd-maximum-8-hour-mean     0
air-pollutant-nitrogen-dioxide                            0
air-pollutant-ozone                                       0
air-pollutant-particulate-matter-pm2-5                    0
air-pollutant-particulate-matter-pm10                     0
air-pollutant-sulphur-dioxide                             0
dtype: int64
```

In [83]:

```
df.shape
```

Out[83]:

```
(66, 11)
```

In [84]:

```
df
```

Out[84]:

	year	value	category	number	air-polluant-lead	air-pollutant-carbon-monoxide-2nd-maximum-8-hour-mean	air-pollutant-nitrogen-dioxide	air-pollutant-ozone	air-pollutant-particulate-matter-pm2-5	poll partic m
31	2006	11385.985714	Buses	13831.0	0.017	0.017	24.0	127.0	23.0	
32	2006	11385.985714	Cars & Station-wagons	472308.0	0.017	0.017	24.0	127.0	23.0	
33	2006	11385.985714	Goods & Other Vehicles	132841.0	0.017	0.017	24.0	127.0	23.0	
34	2006	11385.985714	Motorcycles	141881.0	0.017	0.017	24.0	127.0	23.0	

In [85]:

```
#df.drop(['level_1'],axis=1,inplace = True)
```

In [86]:

```
df.head(10)
```

Out[86]:

	year	value	category	number	air-pollutant-lead	air-pollutant-carbon-monoxide-2nd-maximum-8-hour-mean	air-pollutant-nitrogen-dioxide	air-pollutant-ozone	pol parti r
31	2006	11385.985714	Buses	13831.0	0.017	0.017	24.0	127.0	
32	2006	11385.985714	Cars & Station-wagons	472308.0	0.017	0.017	24.0	127.0	
33	2006	11385.985714	Goods & Other Vehicles	132841.0	0.017	0.017	24.0	127.0	
34	2006	11385.985714	Motorcycles	141881.0	0.017	0.017	24.0	127.0	
35	2006	11385.985714	Tax Exempted Vehicles	15178.0	0.017	0.017	24.0	127.0	
36	2006	11385.985714	Taxis	23334.0	0.017	0.017	24.0	127.0	
37	2007	12127.214286	Buses	14192.0	0.019	0.019	22.0	140.0	
38	2007	12127.214286	Cars & Station-wagons	514685.0	0.019	0.019	22.0	140.0	
39	2007	12127.214286	Goods & Other Vehicles	138604.0	0.019	0.019	22.0	140.0	
40	2007	12127.214286	Motorcycles	143482.0	0.019	0.019	22.0	140.0	

one hot encoding

In [87]:

```
df['category'].unique()
```

Out[87]:

```
array(['Buses', 'Cars & Station-wagons', 'Goods & Other Vehicles',
       'Motorcycles', 'Tax Exempted Vehicles', 'Taxis'], dtype=object)
```

In [88]:

```
#df['type'].unique()
```

In [89]:

```
df_category = pd.get_dummies(df['category'])
df_category
```

Out[89]:

	Buses	Cars & Station-wagons	Goods & Other Vehicles	Motorcycles	Tax Exempted Vehicles	Taxis
31	1	0	0	0	0	0
32	0	1	0	0	0	0
33	0	0	1	0	0	0
34	0	0	0	1	0	0
35	0	0	0	0	1	0
36	0	0	0	0	0	1
37	1	0	0	0	0	0
38	0	1	0	0	0	0
39	0	0	1	0	0	0
40	0	0	0	1	0	0
41	0	0	0	0	1	0
42	0	0	0	0	0	1
43	1	0	0	0	0	0
44	0	1	0	0	0	0
45	0	0	1	0	0	0
46	0	0	0	1	0	0
47	0	0	0	0	1	0
48	0	0	0	0	0	1
49	1	0	0	0	0	0
50	0	1	0	0	0	0
51	0	0	1	0	0	0
52	0	0	0	1	0	0
53	0	0	0	0	1	0
54	0	0	0	0	0	1
55	1	0	0	0	0	0
56	0	1	0	0	0	0
57	0	0	1	0	0	0
58	0	0	0	1	0	0
59	0	0	0	0	1	0
60	0	0	0	0	0	1
...	...	...	...	...	...	...
67	1	0	0	0	0	0
68	0	1	0	0	0	0

	Buses	Cars & Station-wagons	Goods & Other Vehicles	Motorcycles	Tax Exempted Vehicles	Taxis
69	0	0	1	0	0	0
70	0	0	0	1	0	0
71	0	0	0	0	1	0
72	0	0	0	0	0	1
73	1	0	0	0	0	0
74	0	1	0	0	0	0
75	0	0	1	0	0	0
76	0	0	0	1	0	0
77	0	0	0	0	1	0
78	0	0	0	0	0	1
79	1	0	0	0	0	0
80	0	1	0	0	0	0
81	0	0	1	0	0	0
82	0	0	0	1	0	0
83	0	0	0	0	1	0
84	0	0	0	0	0	1
85	1	0	0	0	0	0
86	0	1	0	0	0	0
87	0	0	1	0	0	0
88	0	0	0	1	0	0
89	0	0	0	0	1	0
90	0	0	0	0	0	1
91	1	0	0	0	0	0
92	0	1	0	0	0	0
93	0	0	1	0	0	0
94	0	0	0	1	0	0
95	0	0	0	0	1	0
96	0	0	0	0	0	1

66 rows × 6 columns

In [90]:

```
df.drop(['category'],axis=1,inplace=True)
```

conactenate category and cleaned data

In [91]:

```
df = pd.concat([df,df_category],axis=1)
df
```

Out[91]:

	year	value	number	air-polluant-lead	air-pollutant-carbon-monoxide-2nd-maximum-8-hour-mean	air-pollutant-nitrogen-dioxide	air-pollutant-ozone	air-pollutant-particulate-matter-pm2-5	p
31	2006	11385.985714	13831.0	0.017	0.017	24.0	127.0	23.0	
32	2006	11385.985714	472308.0	0.017	0.017	24.0	127.0	23.0	
33	2006	11385.985714	132841.0	0.017	0.017	24.0	127.0	23.0	
34	2006	11385.985714	141881.0	0.017	0.017	24.0	127.0	23.0	
35	2006	11385.985714	15178.0	0.017	0.017	24.0	127.0	23.0	
36	2006	11385.985714	23334.0	0.017	0.017	24.0	127.0	23.0	
37	2007	12127.214286	14192.0	0.019	0.019	22.0	140.0	19.0	
38	2007	12127.214286	514685.0	0.019	0.019	22.0	140.0	19.0	
39	2007	12127.214286	138604.0	0.019	0.019	22.0	140.0	19.0	
40	2007	12127.214286	143482.0	0.019	0.019	22.0	140.0	19.0	
41	2007	12127.214286	15927.0	0.019	0.019	22.0	140.0	19.0	
42	2007	12127.214286	24446.0	0.019	0.019	22.0	140.0	19.0	
43	2008	12619.338095	14976.0	0.018	0.018	22.0	103.0	16.0	
44	2008	12619.338095	550455.0	0.018	0.018	22.0	103.0	16.0	
45	2008	12619.338095	142966.0	0.018	0.018	22.0	103.0	16.0	
46	2008	12619.338095	145288.0	0.018	0.018	22.0	103.0	16.0	
47	2008	12619.338095	16697.0	0.018	0.018	22.0	103.0	16.0	
48	2008	12619.338095	24300.0	0.018	0.018	22.0	103.0	16.0	
49	2009	10843.247619	15659.0	0.010	0.010	22.0	100.0	19.0	
50	2009	10843.247619	576988.0	0.010	0.010	22.0	100.0	19.0	
51	2009	10843.247619	144802.0	0.010	0.010	22.0	100.0	19.0	
52	2009	10843.247619	146337.0	0.010	0.010	22.0	100.0	19.0	
53	2009	10843.247619	17030.0	0.010	0.010	22.0	100.0	19.0	
54	2009	10843.247619	24702.0	0.010	0.010	22.0	100.0	19.0	
55	2010	13066.823810	15936.0	0.009	0.009	23.0	129.0	17.0	
56	2010	13066.823810	595185.0	0.009	0.009	23.0	129.0	17.0	
57	2010	13066.823810	143613.0	0.009	0.009	23.0	129.0	17.0	
58	2010	13066.823810	147282.0	0.009	0.009	23.0	129.0	17.0	
59	2010	13066.823810	17740.0	0.009	0.009	23.0	129.0	17.0	
60	2010	13066.823810	26073.0	0.009	0.009	23.0	129.0	17.0	

	year	value	number	air-polluant-lead	air-pollutant-carbon-monoxide-2nd-maximum-8-hour-mean	air-pollutant-nitrogen-dioxide	air-pollutant-ozone	air-pollutant-particulate-matter-pm2-5	p
...	...	...	...	...	...	...	...	...	
67	2012	14359.557143	16768.0	0.008	0.008	25.0	122.0	19.0	
68	2012	14359.557143	617570.0	0.008	0.008	25.0	122.0	19.0	
69	2012	14359.557143	145046.0	0.008	0.008	25.0	122.0	19.0	
70	2012	14359.557143	143286.0	0.008	0.008	25.0	122.0	19.0	
71	2012	14359.557143	19030.0	0.008	0.008	25.0	122.0	19.0	
72	2012	14359.557143	28210.0	0.008	0.008	25.0	122.0	19.0	
73	2013	14267.795238	17065.0	0.009	0.009	25.0	139.0	20.0	
74	2013	14267.795238	621345.0	0.009	0.009	25.0	139.0	20.0	
75	2013	14267.795238	144202.0	0.009	0.009	25.0	139.0	20.0	
76	2013	14267.795238	144307.0	0.009	0.009	25.0	139.0	20.0	
77	2013	14267.795238	19556.0	0.009	0.009	25.0	139.0	20.0	
78	2013	14267.795238	27695.0	0.009	0.009	25.0	139.0	20.0	
79	2014	14601.995238	17109.0	0.016	0.016	24.0	135.0	18.0	
80	2014	14601.995238	616609.0	0.016	0.016	24.0	135.0	18.0	
81	2014	14601.995238	144507.0	0.016	0.016	24.0	135.0	18.0	
82	2014	14601.995238	144404.0	0.016	0.016	24.0	135.0	18.0	
83	2014	14601.995238	20672.0	0.016	0.016	24.0	135.0	18.0	
84	2014	14601.995238	28736.0	0.016	0.016	24.0	135.0	18.0	
85	2015	13556.723810	17740.0	0.016	2.100	25.0	129.0	17.0	
86	2015	13556.723810	602311.0	0.016	2.100	25.0	129.0	17.0	
87	2015	13556.723810	143972.0	0.016	2.100	25.0	129.0	17.0	
88	2015	13556.723810	143279.0	0.016	2.100	25.0	129.0	17.0	
89	2015	13556.723810	21685.0	0.016	2.100	25.0	129.0	17.0	
90	2015	13556.723810	28259.0	0.016	2.100	25.0	129.0	17.0	
91	2016	12878.814286	18338.0	0.019	2.500	24.0	131.0	19.0	
92	2016	12878.814286	601257.0	0.019	2.500	24.0	131.0	19.0	
93	2016	12878.814286	143966.0	0.019	2.500	24.0	131.0	19.0	
94	2016	12878.814286	142439.0	0.019	2.500	24.0	131.0	19.0	
95	2016	12878.814286	22896.0	0.019	2.500	24.0	131.0	19.0	
96	2016	12878.814286	27534.0	0.019	2.500	24.0	131.0	19.0	

66 rows × 16 columns

seperate dependent and independent dataset

In [92]:

```
X = df.drop(['air-pollutant-particulate-matter-pm2-5'],axis=1)
y = df['air-pollutant-particulate-matter-pm2-5']
```

In [93]:

```
X.shape
```

Out[93]:

```
(66, 15)
```

split dataset to training and testing

In [94]:

```
from sklearn.model_selection import train_test_split, cross_val_score
from sklearn.metrics import roc_auc_score , classification_report, mean_squared_error, r2_s
from sklearn.metrics import precision_score, recall_score, accuracy_score, classification_r
# split dataset to 60% training and 40% testing
X_train, X_test, y_train, y_test = train_test_split(X,y, test_size=0.4, random_state=0)
```

Apply Linear Regression algorithm to train model

In [95]:

```
from sklearn.linear_model import LinearRegression
lr = LinearRegression()
lr.fit(X_train.drop(['year'],axis=1), y_train)
y_train_pred = lr.predict(X_train.drop(['year'],axis=1))
y_test_pred = lr.predict(X_test.drop(['year'],axis=1))
```

Error

In [96]:

```
print('MSE train: %.3f, test: %.3f' % (
        mean_squared_error(y_train, y_train_pred),
        mean_squared_error(y_test, y_test_pred)))
print('R^2 train: %.3f, test: %.3f' % (
        r2_score(y_train, y_train_pred),
        r2_score(y_test, y_test_pred)))
```

```
MSE train: 0.230, test: 0.385
R^2 train: 0.936, test: 0.867
```

Accuracy Score

In [97]:

```
lr.score(X_test.drop(['year'],axis=1),y_test)
```

Out[97]:

```
0.8673040478396414
```

predict on unknown data

In [98]:

```
df_unkn = pd.read_csv('input.csv')
df_unkn.head(1)
```

Out[98]:

	year	value	number	air-pollutant-lead	air-pollutant-carbon-monoxide-2nd-maximum-8-hour-mean	air-pollutant-nitrogen-dioxide	air-pollutant-ozone	air-pollutant-particulate-matter-pm10	ai pollutan sulphu dioxid
0	2013	14267.79524	144307	0.009	0.009	25	139	215	1

In [99]:

```
df_pred = pd.DataFrame({'year':X_test['year'],'original_pm_2_5':y_test,'predicted_pm_2_5':y
df_pred = df_pred[['original_pm_2_5','predicted_pm_2_5']].groupby(df_pred['year']).mean()
df_pred
```

Out[99]:

	original_pm_2_5	predicted_pm_2_5
year		
2006	23.0	23.149206
2007	19.0	19.512665
2008	16.0	16.558319
2009	19.0	18.617594
2010	17.0	17.302050
2011	17.0	17.704665
2012	19.0	19.108745
2013	20.0	19.917517
2014	18.0	16.874872
2015	17.0	17.777308
2016	19.0	18.128126

In [100]:

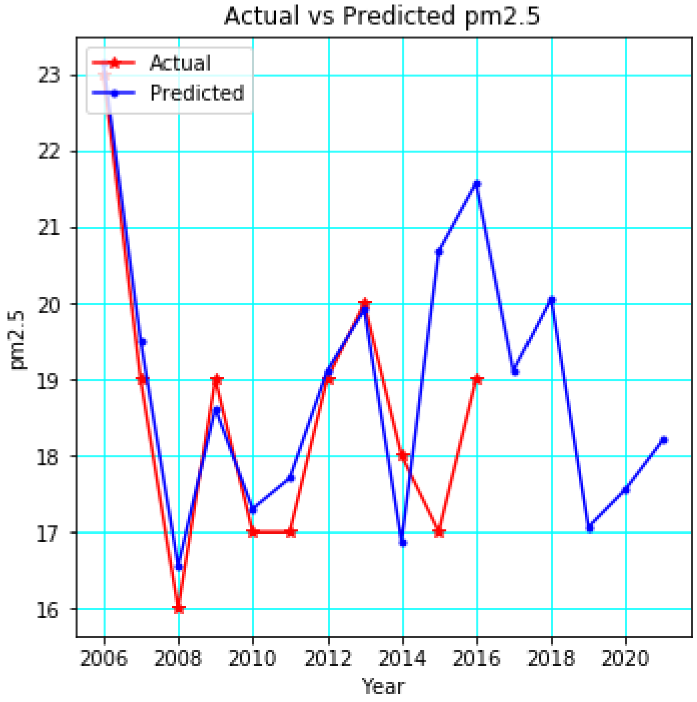
```
df_unkn_pred = lr.predict(df_unkn.drop(['year'],axis=1))
df_unkn_pred = pd.DataFrame({'year':df_unkn['year'],'predicted_pm_2_5':df_unkn_pred})
df_unkn_pred = df_unkn_pred[['predicted_pm_2_5']].groupby(df_unkn_pred['year']).mean()
df_unkn_pred
```

Out[100]:

	predicted_pm_2_5
year	
2006	23.149206
2007	19.512665
2008	16.558319
2009	18.617594
2010	17.302050
2011	17.704665
2012	19.108745
2013	19.917517
2014	16.874872
2015	20.683986
2016	21.583319
2017	19.114274
2018	20.060111
2019	17.062192
2020	17.558283
2021	18.203935

In [101]:

```
plt.figure(figsize=(5.5, 5.5))
plt.plot(df_pred.index,df_pred['original_pm_2_5'], linestyle='-', marker='*', color='r')
plt.plot(df_unkn_pred.index, df_unkn_pred['predicted_pm_2_5'], linestyle='-', marker='.', c
plt.legend(['Actual','Predicted'], loc=2)
plt.title('Actual vs Predicted pm2.5')
plt.ylabel('pm2.5')
plt.xlabel('Year')
plt.grid(color='cyan')
plt.show()
```



Apply Decision Tree algorithm to train the model

In [102]:

```
from sklearn.tree import DecisionTreeRegressor
tree = DecisionTreeRegressor(max_depth=3)
tree.fit(X_train.drop(['year'],axis=1), y_train)
y_train_pred = tree.predict(X_train.drop(['year'],axis=1))
y_test_pred = tree.predict(X_test.drop(['year'],axis=1))
```

Error

In [103]:

```
print('MSE train: %.3f, test: %.3f' % (
        mean_squared_error(y_train, y_train_pred),
        mean_squared_error(y_test, y_test_pred)))
print('R^2 train: %.3f, test: %.3f' % (
        r2_score(y_train, y_train_pred),
        r2_score(y_test, y_test_pred)))
```

```
MSE train: 0.148, test: 0.996
R^2 train: 0.959, test: 0.657
```

Accuracy score

In [104]:

```
tree.score(X_test.drop(['year'],axis=1),y_test)
```

Out[104]:

```
0.656665717374935
```

In [105]:

```
df_unkn = pd.read_csv('input.csv')
df_unkn.head(1)
```

Out[105]:

	year	value	number	air-pollutant-lead	air-pollutant-carbon-monoxide-2nd-maximum-8-hour-mean	air-pollutant-nitrogen-dioxide	air-pollutant-ozone	air-pollutant-particulate-matter-pm10	ai pollutan sulphu dioxid
0	2013	14267.79524	144307	0.009	0.009	25	139	215	1

In [106]:

```
df_pred = pd.DataFrame({'year':X_test['year'],'original_pm_2_5':y_test,'predicted_pm_2_5':y
df_pred = df_pred[['original_pm_2_5','predicted_pm_2_5']].groupby(df_pred['year']).mean()
df_pred
```

Out[106]:

	original_pm_2_5	predicted_pm_2_5
year		
2006	23.0	23.000000
2007	19.0	19.285714
2008	16.0	16.583333
2009	19.0	19.000000
2010	17.0	16.583333
2011	17.0	19.000000
2012	19.0	19.285714
2013	20.0	19.285714
2014	18.0	18.000000
2015	17.0	16.583333
2016	19.0	19.285714

In [107]:

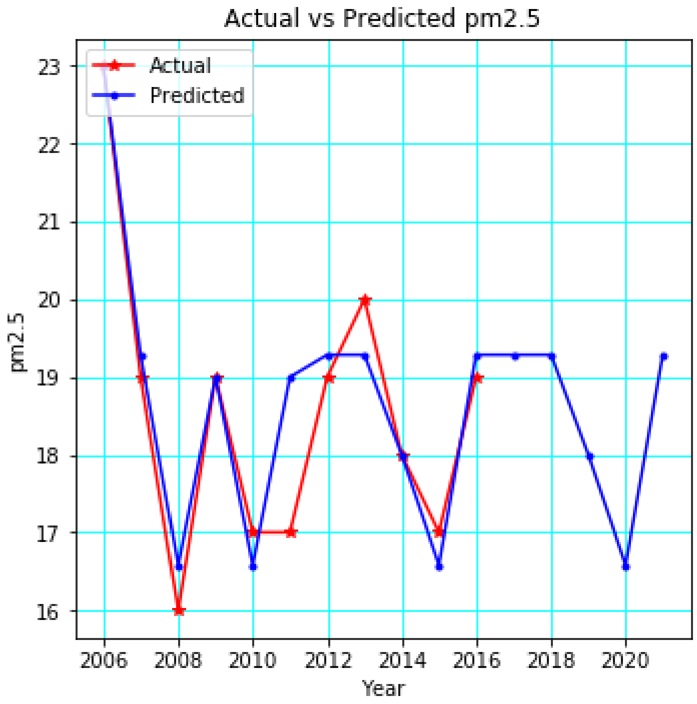
```
df_unkn_pred = tree.predict(df_unkn.drop(['year'],axis=1))
df_unkn_pred = pd.DataFrame({'year':df_unkn['year'],'predicted_pm_2_5':df_unkn_pred})
df_unkn_pred = df_unkn_pred[['predicted_pm_2_5']].groupby(df_unkn_pred['year']).mean()
df_unkn_pred
```

Out[107]:

	predicted_pm_2_5
year	
2006	23.000000
2007	19.285714
2008	16.583333
2009	19.000000
2010	16.583333
2011	19.000000
2012	19.285714
2013	19.285714
2014	18.000000
2015	16.583333
2016	19.285714
2017	19.285714
2018	19.285714
2019	18.000000
2020	16.583333
2021	19.285714

In [108]:

```
plt.figure(figsize=(5.5, 5.5))
plt.plot(df_pred.index,df_pred['original_pm_2_5'], linestyle='-', marker='*', color='r')
plt.plot(df_unkn_pred.index, df_unkn_pred['predicted_pm_2_5'], linestyle='-', marker='.', c
plt.legend(['Actual','Predicted'], loc=2)
plt.title('Actual vs Predicted pm2.5')
plt.ylabel('pm2.5')
plt.xlabel('Year')
plt.grid(color='cyan')
plt.show()
```



Apply Random Forest algorithm to train the model

In [109]:

```
from sklearn.ensemble import RandomForestRegressor

forest = RandomForestRegressor(n_estimators=1000,
                               criterion='mse',
                               random_state=1,
                               n_jobs=-1)
forest.fit(X_train.drop(['year'],axis=1), y_train)
y_train_pred = forest.predict(X_train.drop(['year'],axis=1))
y_test_pred = forest.predict(X_test.drop(['year'],axis=1))
```

Error

In [110]:

```
print('MSE train: %.3f, test: %.3f' % (
        mean_squared_error(y_train, y_train_pred),
        mean_squared_error(y_test, y_test_pred)))
print('R^2 train: %.3f, test: %.3f' % (
        r2_score(y_train, y_train_pred),
        r2_score(y_test, y_test_pred)))
```

```
MSE train: 0.005, test: 0.103
R^2 train: 0.998, test: 0.964
```

Acccuracy score

In [111]:

```
forest.score(X_test.drop(['year'],axis=1),y_test)
```

Out[111]:

```
0.9643659389782403
```

In [112]:

```
df_pred = pd.DataFrame({'year':X_test['year'],'original_pm_2_5':y_test,'predicted_pm_2_5':y
df_pred = df_pred[['original_pm_2_5','predicted_pm_2_5']].groupby(df_pred['year']).mean()
df_pred
```

Out[112]:

```
      original_pm_2_5  predicted_pm_2_5
year
2006             23.0         22.862500
2007             19.0         18.907333
2008             16.0         16.056000
2009             19.0         18.965000
2010             17.0         17.219000
2011             17.0         17.644167
2012             19.0         18.933000
2013             20.0         20.004000
2014             18.0         18.026000
2015             17.0         17.015000
2016             19.0         18.905000
```

In [113]:

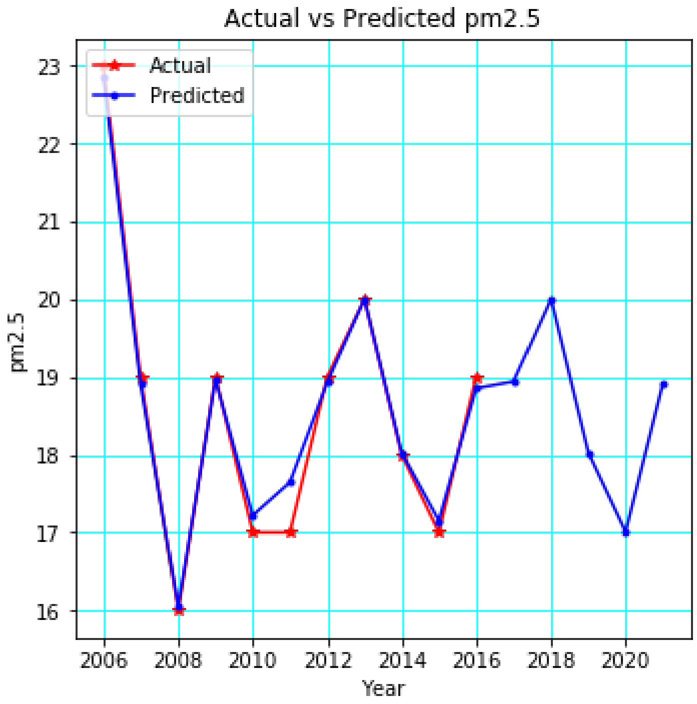
```
df_unkn_pred = forest.predict(df_unkn.drop(['year'],axis=1))
df_unkn_pred = pd.DataFrame({'year':df_unkn['year'],'predicted_pm_2_5':df_unkn_pred})
df_unkn_pred = df_unkn_pred[['predicted_pm_2_5']].groupby(df_unkn_pred['year']).mean()
df_unkn_pred
```

Out[113]:

	predicted_pm_2_5
year	
2006	22.862500
2007	18.907333
2008	16.056000
2009	18.965000
2010	17.219000
2011	17.644167
2012	18.933000
2013	20.004000
2014	18.026000
2015	17.151000
2016	18.855000
2017	18.941833
2018	20.002667
2019	18.025667
2020	17.007000
2021	18.911833

In [114]:

```
plt.figure(figsize=(5.5, 5.5))
plt.plot(df_pred.index,df_pred['original_pm_2_5'], linestyle='-', marker='*', color='r')
plt.plot(df_unkn_pred.index, df_unkn_pred['predicted_pm_2_5'], linestyle='-', marker='.', c
plt.legend(['Actual','Predicted'], loc=2)
plt.title('Actual vs Predicted pm2.5')
plt.ylabel('pm2.5')
plt.xlabel('Year')
plt.grid(color='cyan')
plt.show()
```



In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]: