IBM NAAN MUDHALVAN PHASE-5

DOMAIN : DATA SCIENCE

AIR QUALITY ANALYSIS &PREDICTION

USING DATA SCIENCE

**Introduction :**

* Data science is the study of the data to extract meaningful insights for business.it is a fields of mathematics , statistics ,artificial intelligence and computer engineering .

**Steps :**

* + - Problem identification
    - Data collection
    - Data preparation
    - Data model
    - Data analysis
    - Model evaluation

**Air Quality analysis & prediction :**

* Air quality analysis and prediction involve assessing the current state of the air quality in a specific area, identifying pollutants, and forecasting air quality conditions in the future. This process is crucial for public health, environmental protection, and urban planning.
* Examples are to use less toxic raw materials or fuels, use a less-polluting industrial process, and to improve the efficiency of the process. The Clean Air Technology Center serves as a resource on air pollution prevention and control technologies, including their use, effectiveness and cost.

**Types of air quality models:**

* + - **Statistical Models**
    - **Chemical Transport Models (CTMs)**
    - **Machine Learning Models**
    - **Epidemiological Models**
    - **Dispersion Models**
    - **Hybrid Models**

**Problem Statement:**

Air pollution poses a significant threat to public health, environmental sustainability, and overall quality of life in urban areas. The difficulty lies in creating reliable predictive models that make use of both historical and current data, such as pollutant concentrations, weather conditions, traffic patterns, and geographic characteristics, to estimate air quality levels in particular places.

**Advantages of air quality analysis and prediction:**

• Early warning,

• Informing decisions,

• Identifying sources of pollution,

• Targeted intervention,

• Research and analysis.

**Tools We are Using for this Project:**

Of course, we're using Python to build our project

Air quality analysis and prediction using data science involves various tools and technologies to collect, process, analyze, and visualize data.

1.Jupyter environment (Jupyter Lab or Jupyter notebook) – for experimenting with our project.

2.Pandas – for loading data as a dataframe and wrangling the data.

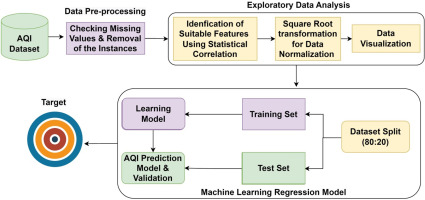
3.NumPy and SciPy - For performing some data manipulation and analysis.

4. Scikit-Learn, TensorFlow, and PyTorch - The are used for building machine learning models to predict air quality based on historical data.

5. Seaborn, Matplotlib and Plotly Express – It used for creating static, interactive, and animated visualizations of air quality data.

6. Git- Git is used for version control, allowing collaborative work on code and data analysis projects.

The features of the dataset are:



**Exploratory Data Analysis (EDA):**

As you might know, EDA is the key to performing well as a data analyst or data scientist. It gives you first-hand information about the whole dataset, and it helps you understand all the relationships between the features in our dataset.

We are performing the three phases of EDA here which are:

1. Univariate Analysis.

2. Bivariate Analysis.

3. Multivariate Analysis

Firstly we are imporrting all the necessary libraries that we are using in this project. We also need to load the dataset into a dataframe so we can see all the features that are present in it.

import pandas as pd

import seaborn as sns

import matplotlib. pyplot as plt

import plotly.express as px

import numpy as np

from scipy.stats import iqr

from sklearn.preprocessing import StandardScaler

from sklearn.cluster import KMeans

df = pd.read\_csv('../input/air-quality-data-set/AirQuality.csv',sep=';')

df.head()

First, since the air quality is based on the total amount prediction have spent, we'll add the amount spent on the product:

df.info()

df.drop('NMHC(GT)', axis=1, inplace=True)

df[ar.columns[2:13]][(lower\_outliers | upper\_outliers)].info()

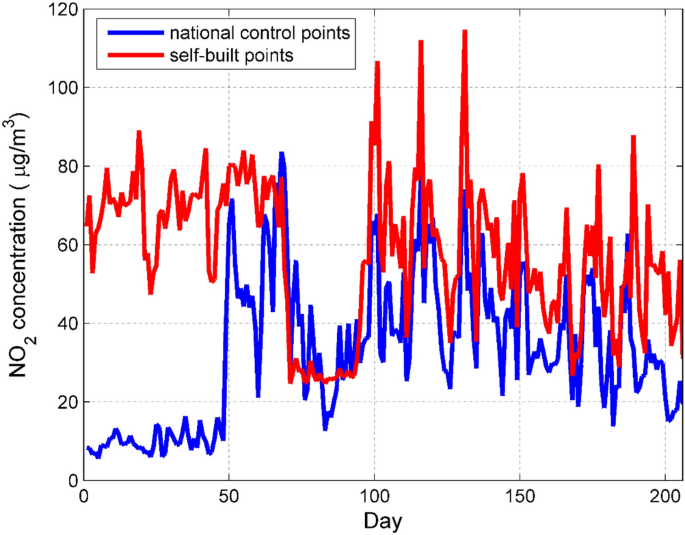
df\_filt.info()

**Univariate analysis:**

Univariate analysis entails evaluating a single feature in order to get insights about it. So, the initial step in performing EDA is to undertake univariate analysis, which includes evaluating descriptive or summary statistics about the feature.

sns.histplot(data=df, x="Day", bins = list(range(10, 150, 10)))

plt.title("National control points")



**Bivariate Analysis:**

The next step is to perform bivariate analysis. This involves comparing two attributes at the same time.

Bivariate analysis entails determining the correlation between two features, for example.

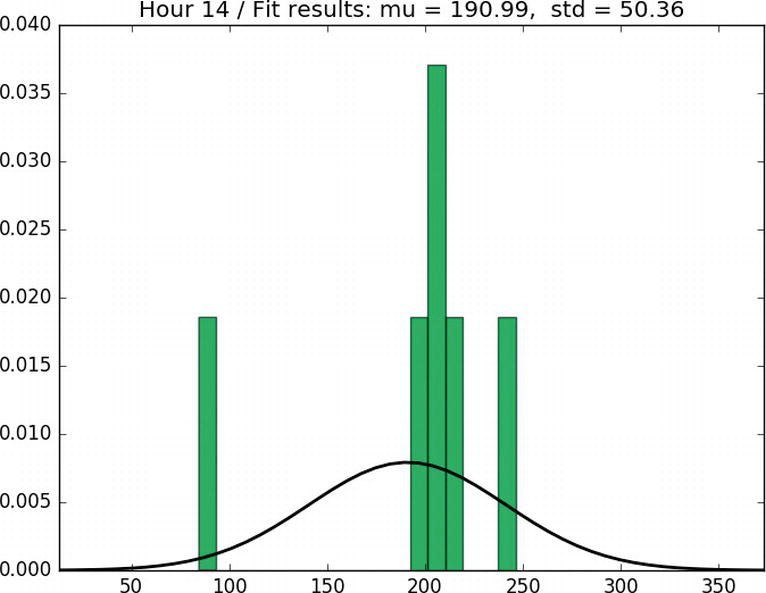
In our case, some of the bivariate analysis we'll perform in the project include observing the average total spent across different client age groups, determining a correlation between customer income and total amount spent, and so on, as shown below.

for i **in** df.columns[0:0.40]:

sns.boxplot(x=df[i])

plt.title('Boxplot of the sensors data')

plt.show()



**Multivariate Analysis:**

After you've completed univariate (analysis of single feature) and bivariate (analysis of two features) analysis, the last phase of EDA is to perform Multivariate Analysis. Multivariate Analysis consists of understanding the relationship between two or more variables.

fig = px.scatter(

data\_frame=df\_cut,

x = "Dataset Type",

y= "Number of publications",

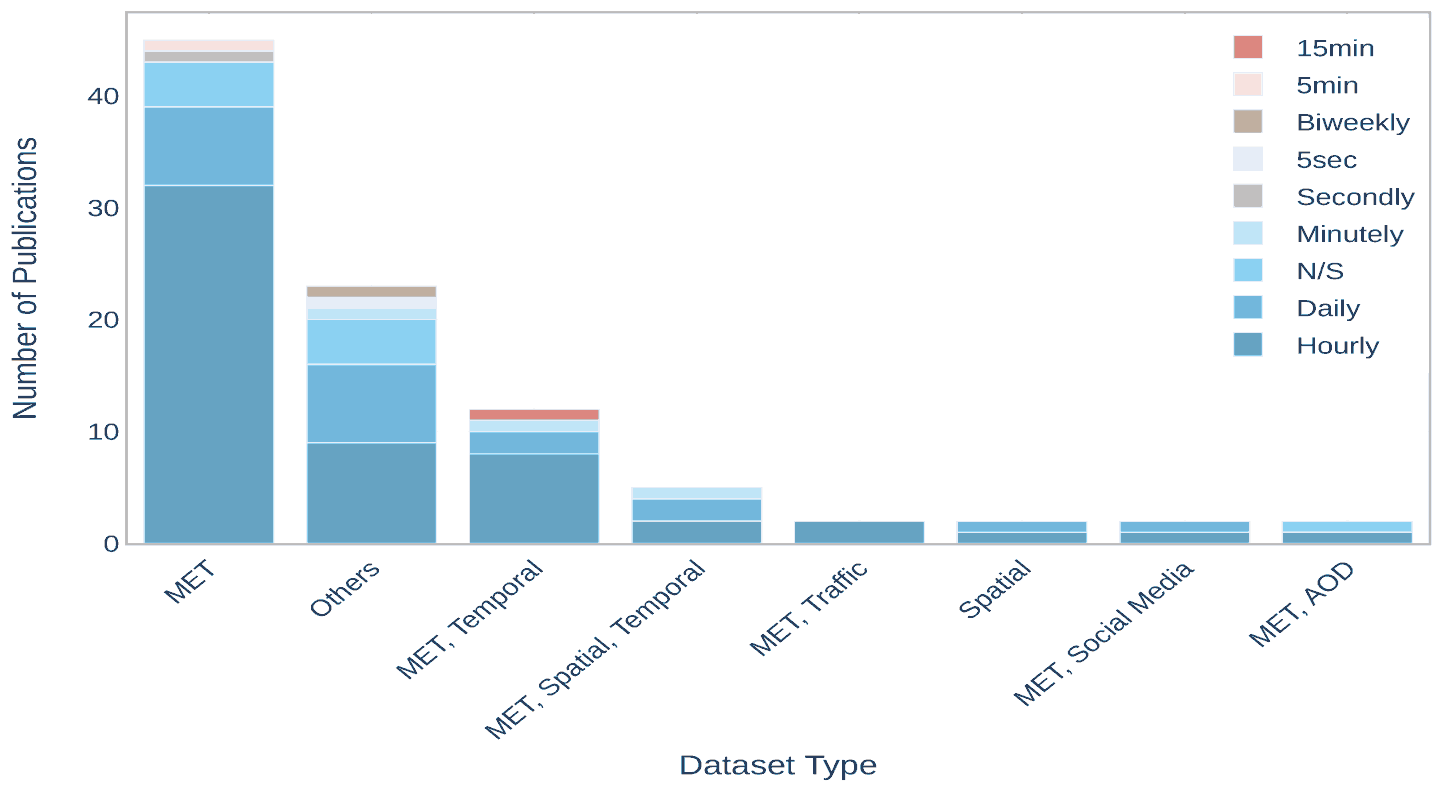
title = "Relationship between Dataset TypeVS Number of publications",

color = "publication",

height=40

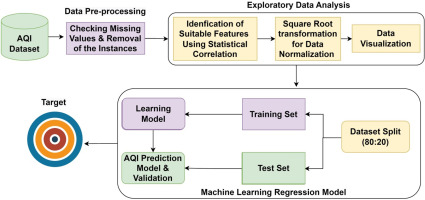
)

fig.show()



**Feature Engineering:**

Sensors can track all the key pollution markers like particulate matter (PM1, PM2. 5, PM10), NO2, O3, SO2, H2S, NO, and CO gases. They also provide reliable data about key weather parameters such as temperature, humidity, air pressure, and wind from our sensors locations.



**PROGRAM :**

**#import**

import numpy as np *# linear algebra*

import pandas as pd *# data processing, CSV file I/O (e.g. pd.read\_csv)*

import matplotlib.pyplot as plt

import seaborn as sns

%matplotlib inline

plt.rcParams['figure.figsize'] = (10, 7)

*# Warnings*

import warnings

warnings.filterwarnings('ignore')

*# Input data files are available in the "../input/" directory.*

*# For example, running this (by clicking run or pressing Shift+Enter) will list the files in the input directory*

import os

print(os.listdir("../input"))

*# Any results you write to the current directory are saved as output.*

data=pd.read\_csv('../input/india-air-quality-data/data.csv',encoding="ISO-8859-1")

data.fillna(0, inplace=True)

data.head()

from mpl\_toolkits.basemap import Basemap

%matplotlib inline

import warnings

warnings.filterwarnings('ignore')

%config InlineBackend.figure\_format = 'retina'

m = Basemap(projection='mill',llcrnrlat=5,urcrnrlat=40, llcrnrlon=60,urcrnrlon=110,lat\_ts=20,resolution='c')

longitudes = dff["lon"].tolist()

latitudes = dff["lat"].tolist()

*#m = Basemap(width=12000000,height=9000000,projection='lcc',*

*#resolution=None,lat\_1=80.,lat\_2=55,lat\_0=80,lon\_0=-107.)*

x,y = m(longitudes,latitudes)

fig = plt.figure(figsize=(12,10))

plt.title("All affected areas")

m.plot(x, y, "o", markersize = 3, color = 'blue')

m.drawcoastlines()

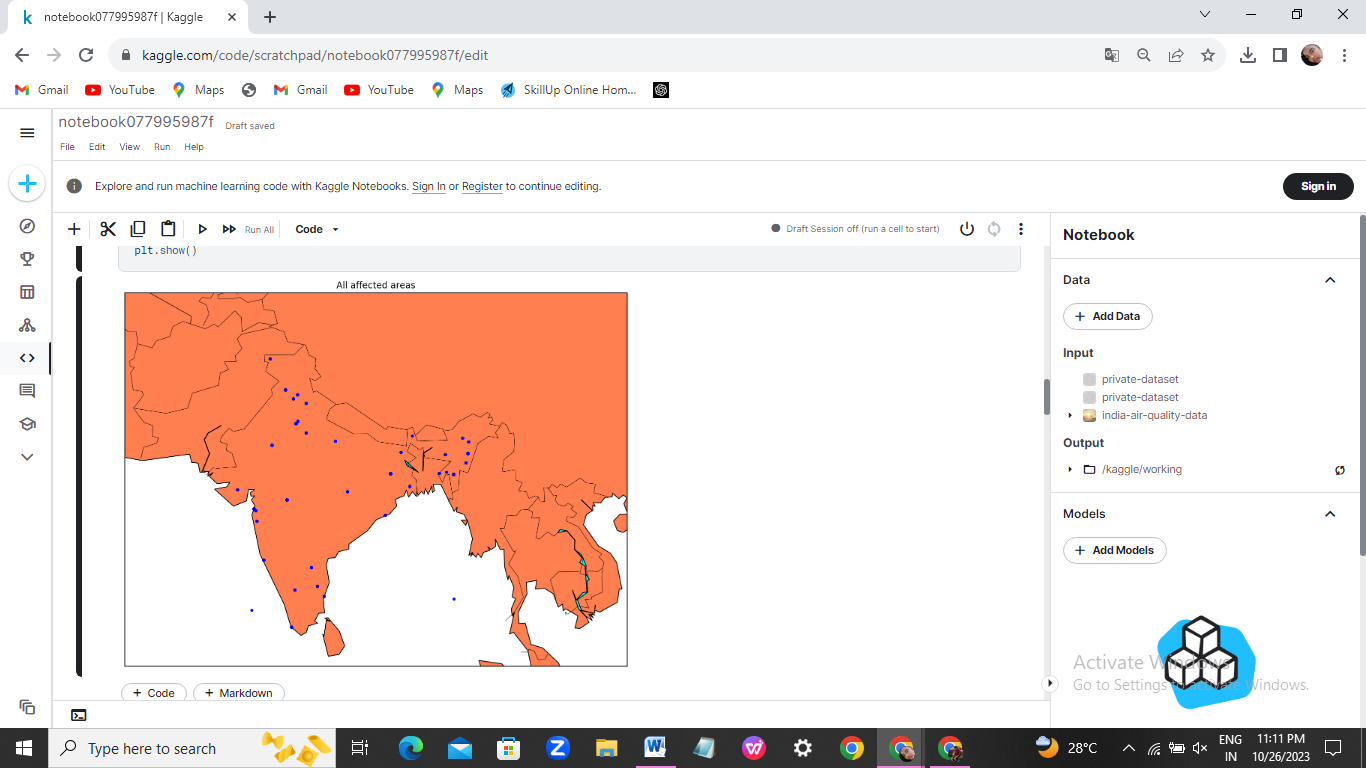
m.fillcontinents(color='coral',lake\_color='aqua')

m.drawmapboundary()

m.drawcountries()

plt.show()

**OUTPUT:**



*#Visualization of AQI across india*

data['date'] = pd.to\_datetime(data['date'],format='%Y-%m-**%d**') *# date parse*

data['year'] = data['date'].dt.year *# year*

data['year'] = data['year'].fillna(0.0).astype(int)

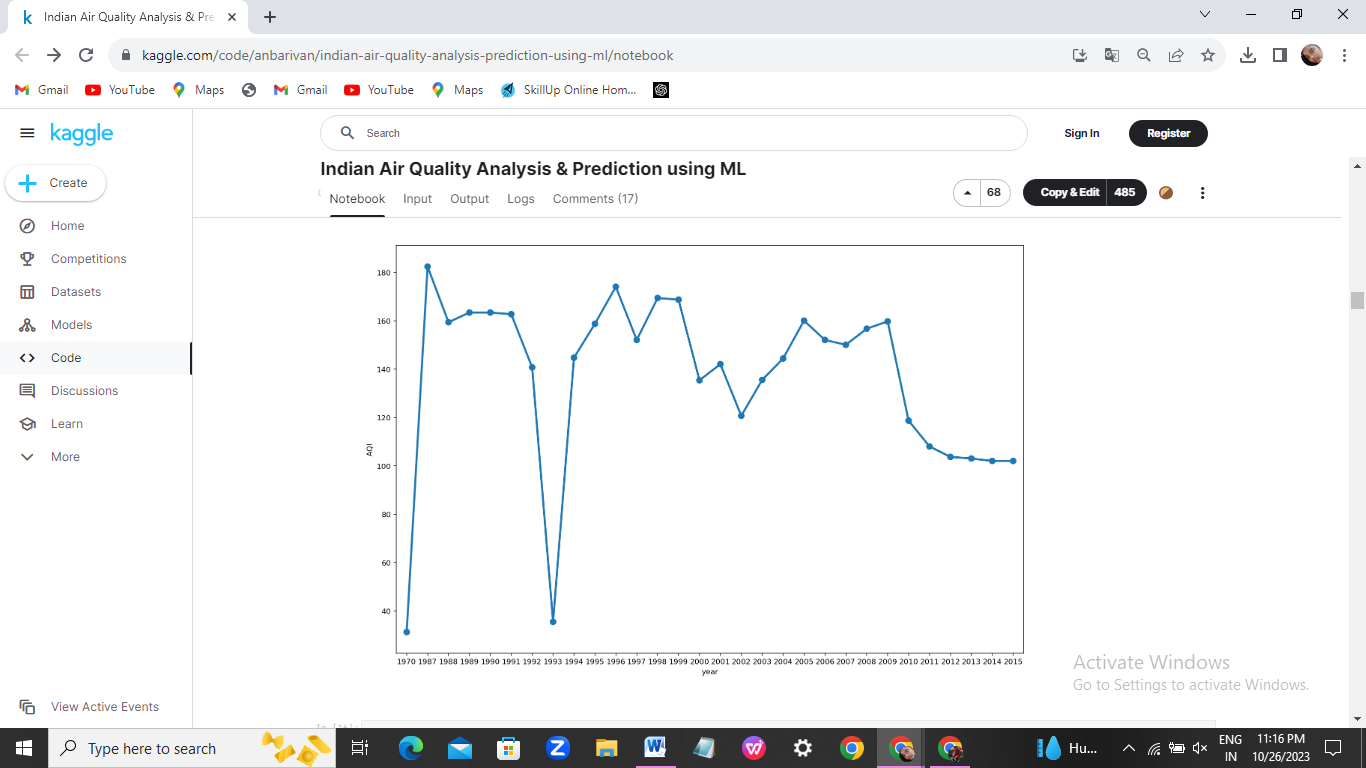
data = data[(data['year']>0)]

df = data[['AQI','year','state']].groupby(["year"]).median().reset\_index().sort\_values(by='year',ascending=False)

f,ax=plt.subplots(figsize=(15,10))

sns.pointplot(x='year', y='AQI', data=df)

**OUTPUT:**



*#exctracting knowledge about data*

*#spliting dataframes into test and train*

n = df.shape[0]

train\_size = 0.65

features\_dataframe = df.sort\_values('date')

train = df.iloc[:int(n \* train\_size)]

test = df.iloc[int(n \* train\_size):]

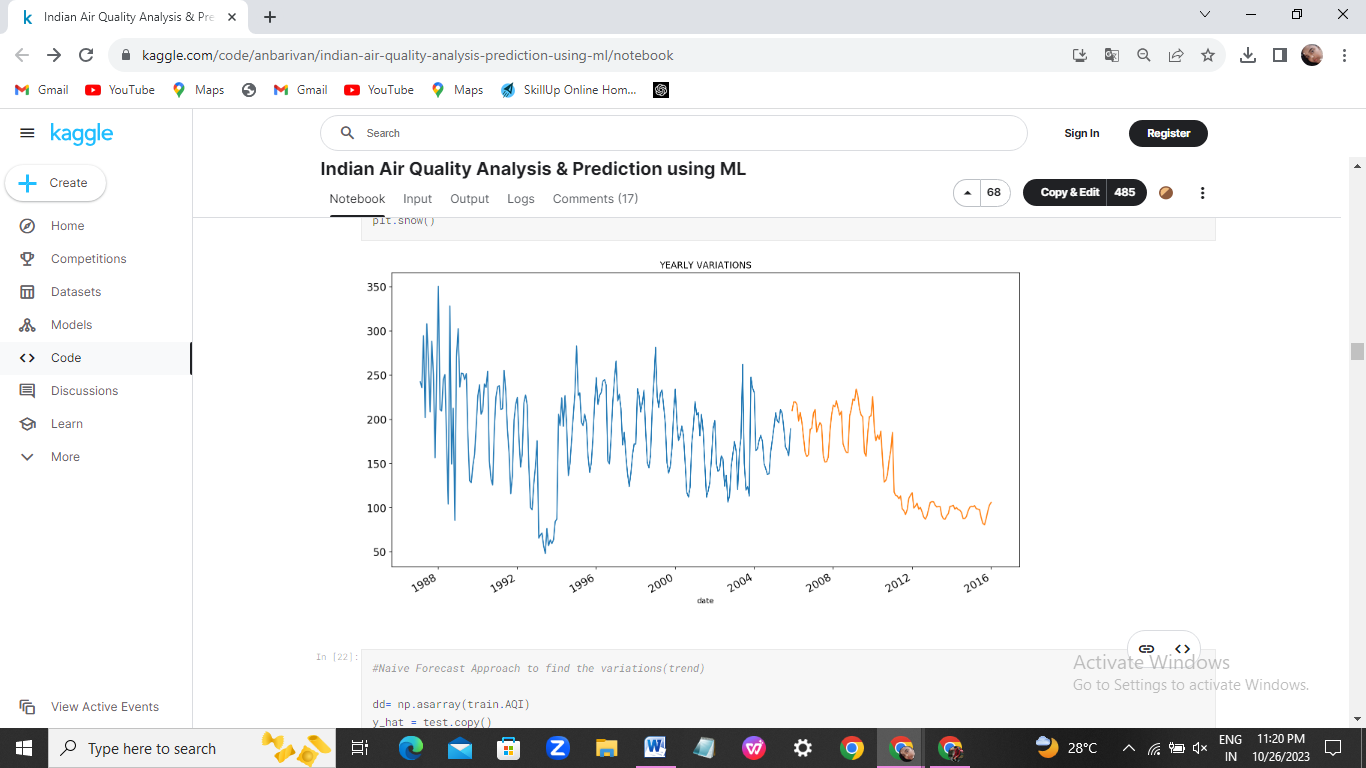
*#plotting the yearly variations of AQI*

train.AQI.plot(figsize=(15,8), title= 'YEARLY VARIATIONS', fontsize=14)

test.AQI.plot(figsize=(15,8), title= 'YEARLY VARIATIONS', fontsize=14)

plt.show()

**Output:**



import statsmodels.api as sm

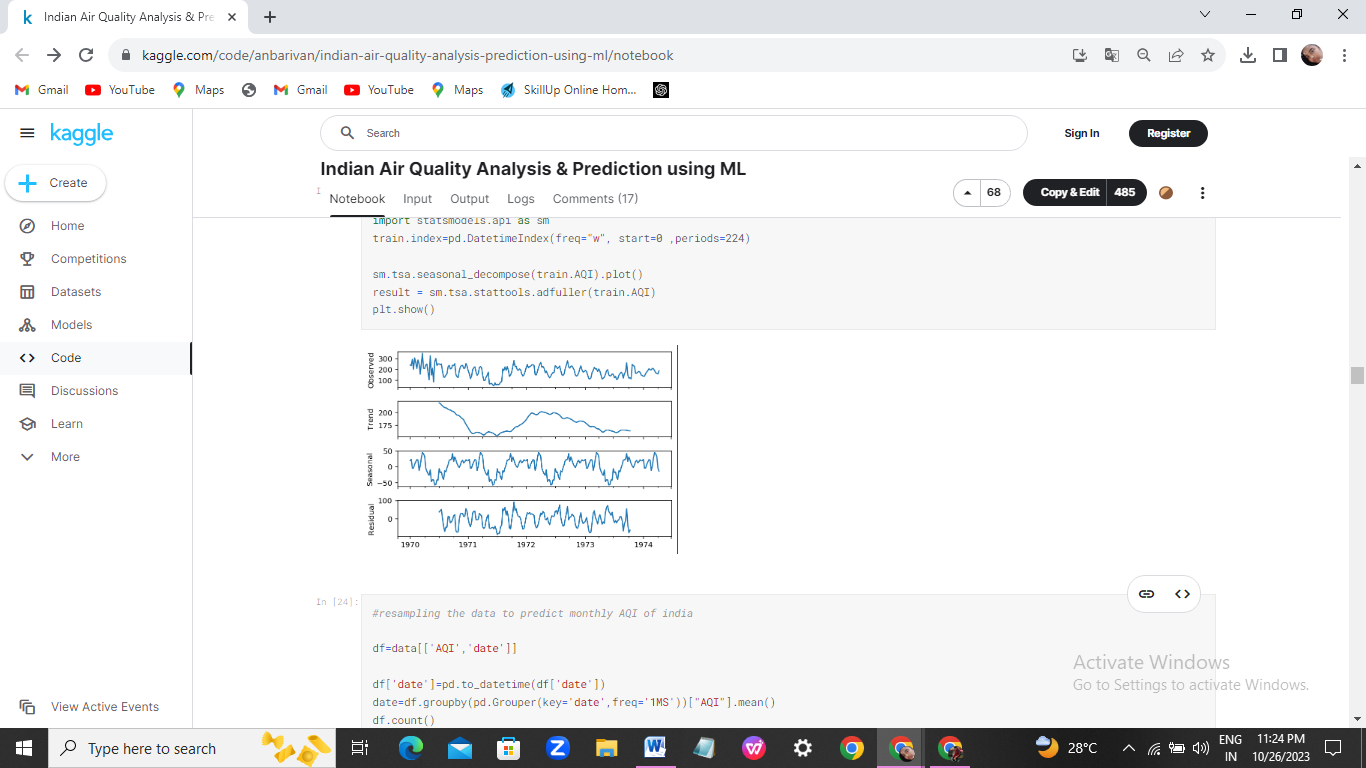
train.index=pd.DatetimeIndex(freq="w", start=0 ,periods=224)

sm.tsa.seasonal\_decompose(train.AQI).plot()

result = sm.tsa.stattools.adfuller(train.AQI)

plt.show()

**OUTPUT:**



*#resampling the data to predict monthly AQI of india*

df=data[['AQI','date']]

df['date']=pd.to\_datetime(df['date'])

date=df.groupby(pd.Grouper(key='date',freq='1MS'))["AQI"].mean()

df.count()

**Output:**

AQI 346

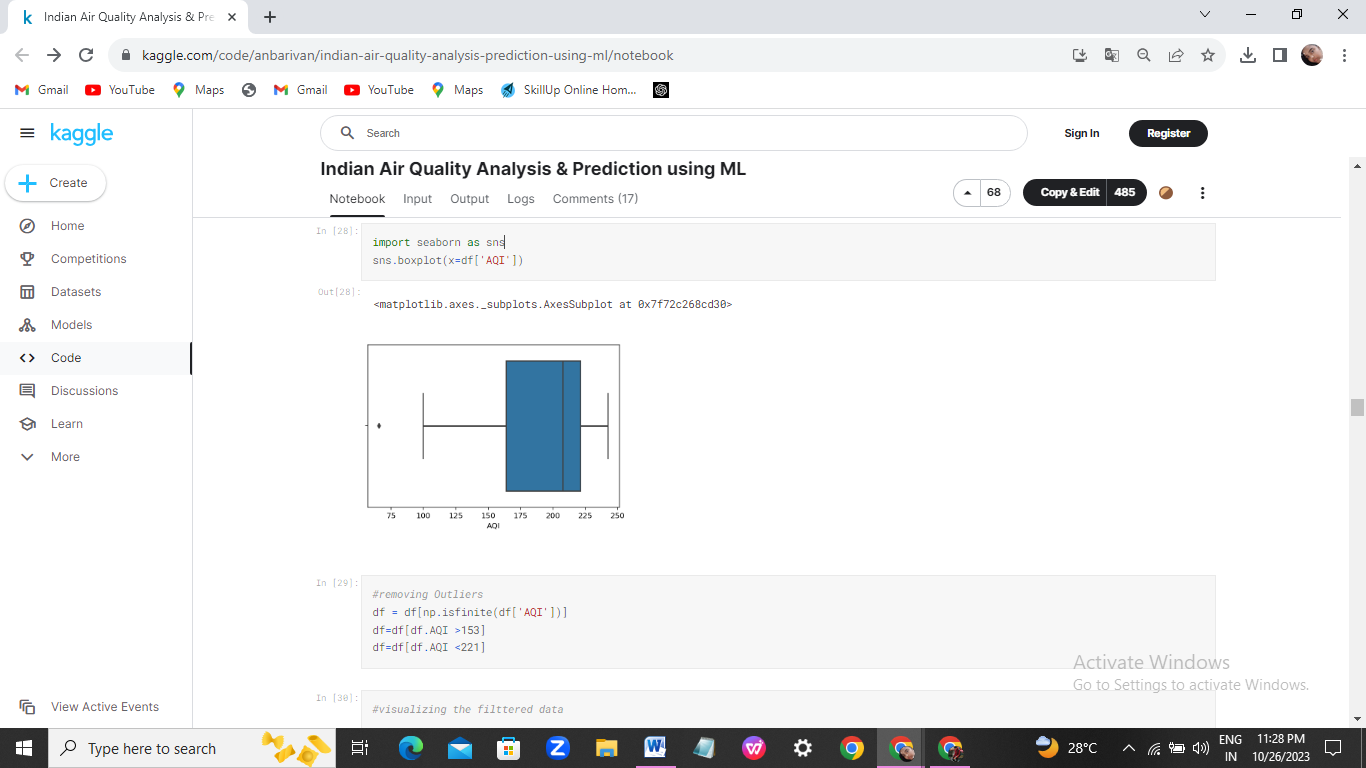
date 346

dtype: int64

import seaborn as sns

sns.boxplot(x=df['AQI'])

**Output:**



*#removing Outliers*

df = df[np.isfinite(df['AQI'])]

df=df[df.AQI >153]

df=df[df.AQI <221]

*#scatter plot of data points*

cols =['year']

y = df['AQI']

x=df[cols]

plt.scatter(x,y)

plt.show()

*#visualizing the filttered data*

year=df['year'].values

AQI=df['AQI'].values

df['AQI']=pd.to\_numeric(df['AQI'],errors='coerce')

df['year']=pd.to\_numeric(df['year'],errors='coerce')

import matplotlib.pyplot as plt

plt.rcParams['figure.figsize'] = (20.0, 10.0)

from mpl\_toolkits.mplot3d import Axes3D

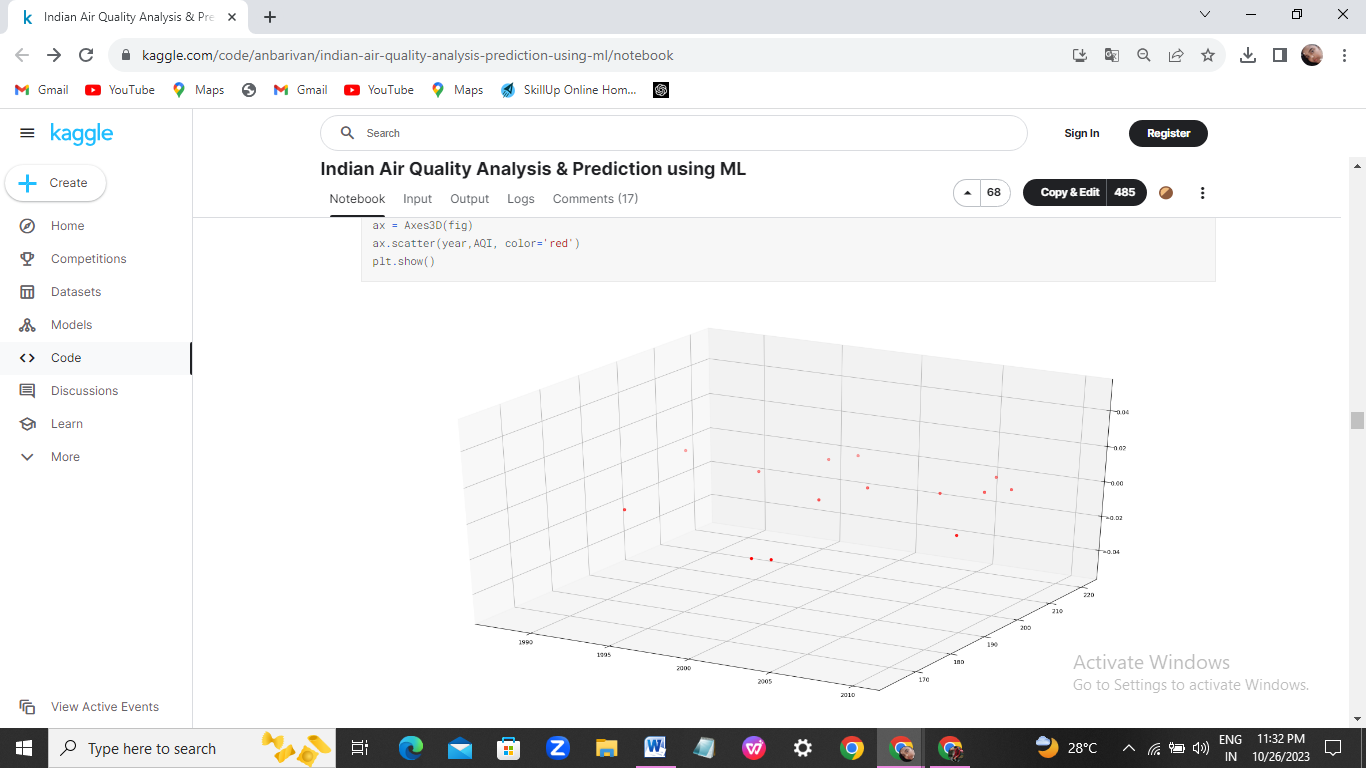
fig = plt.figure()

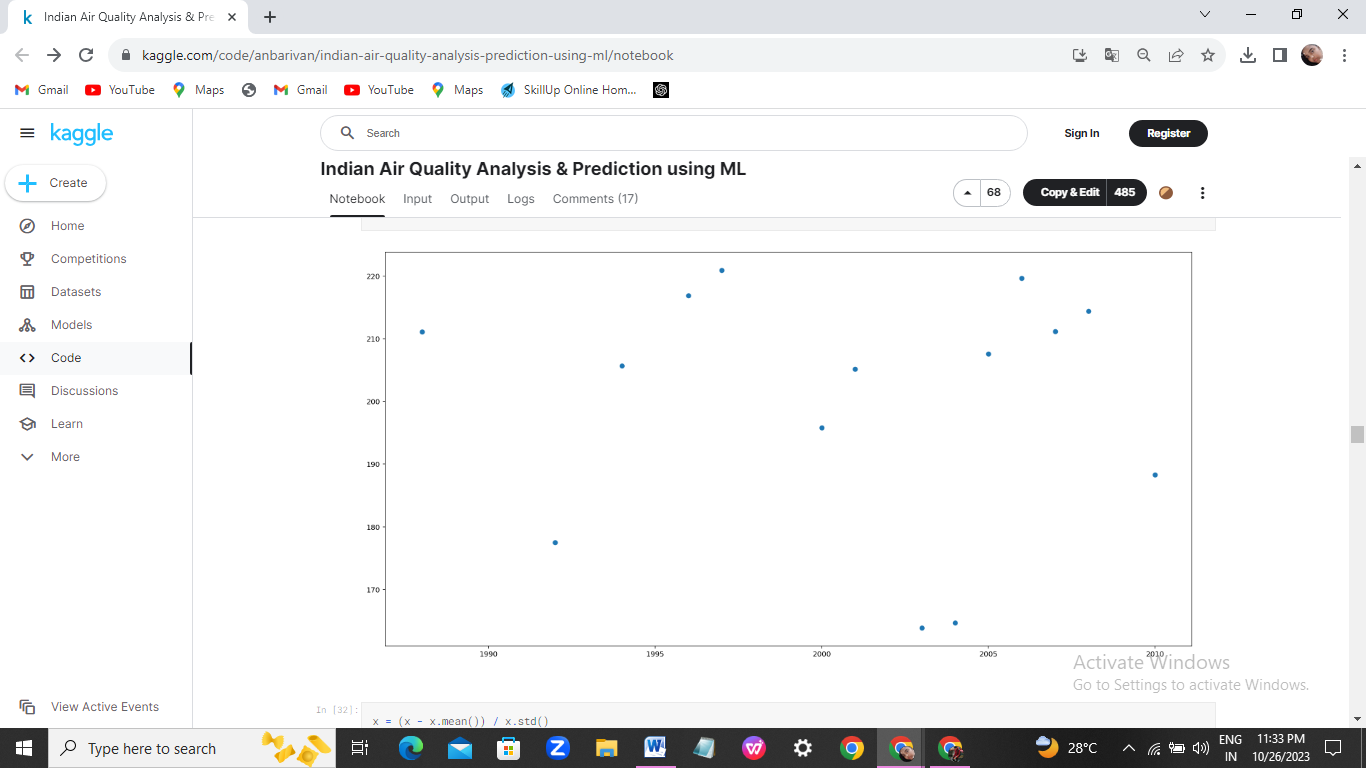
ax = Axes3D(fig)

ax.scatter(year,AQI, color='red')

plt.show()

**OUTPUT:**





x\_axis=x.year

y\_axis=x.Actual

y1\_axis=x.Predicted

plt.plot(x\_axis,y\_axis)

plt.plot(x\_axis,y1\_axis)

plt.title("Actual vs Predicted",fontsize=20)

plt.legend(["actual ","predicted"])

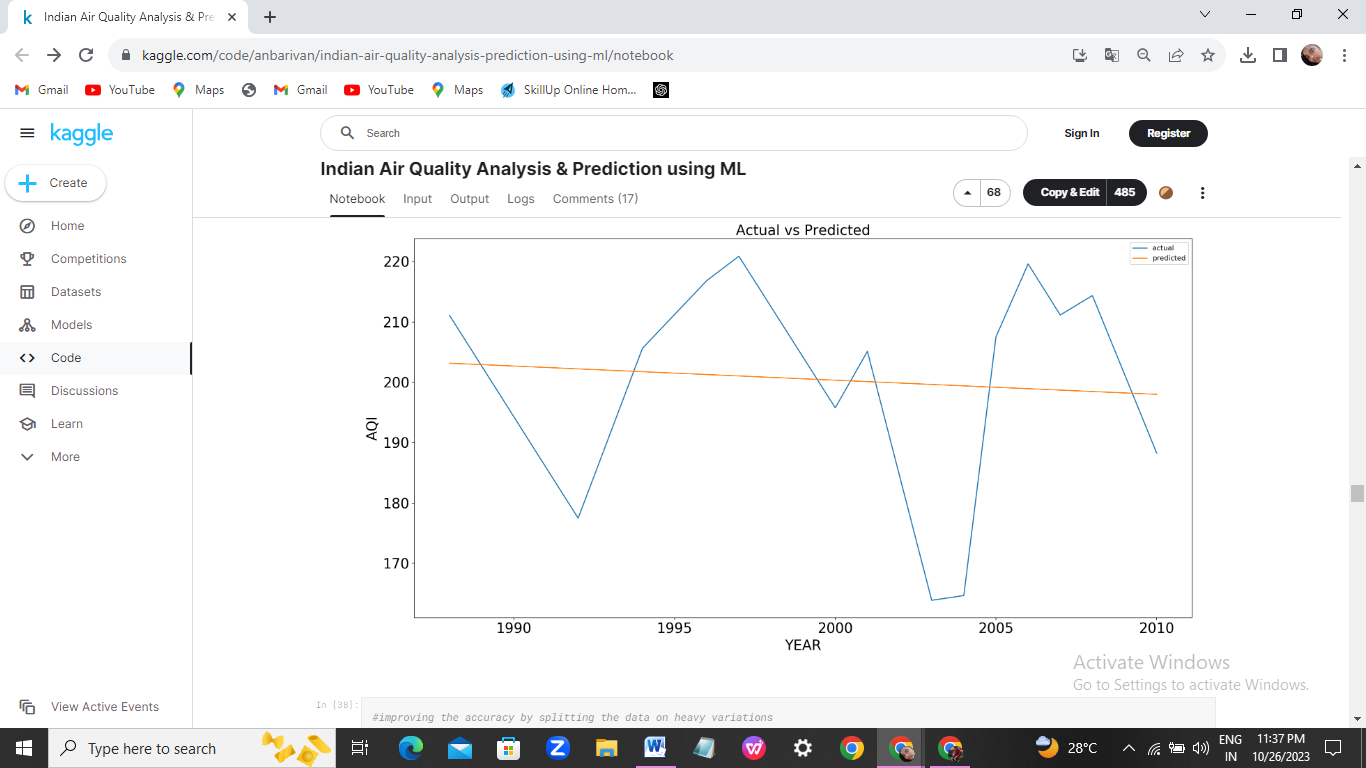
plt.xlabel("YEAR",fontsize=20)

plt.ylabel("AQI",fontsize=20)

plt.tick\_params(labelsize=20)

plt.show()

**Output:**



*#plotting the actual and predicted results*

x\_axis=x.year

y\_axis=x.Actual

y1\_axis=x.Predicted

plt.plot(x\_axis,y\_axis)

plt.plot(x\_axis,y1\_axis)

plt.title("Actual vs Predicted",fontsize=20)

plt.legend(["actual ","predicted"])

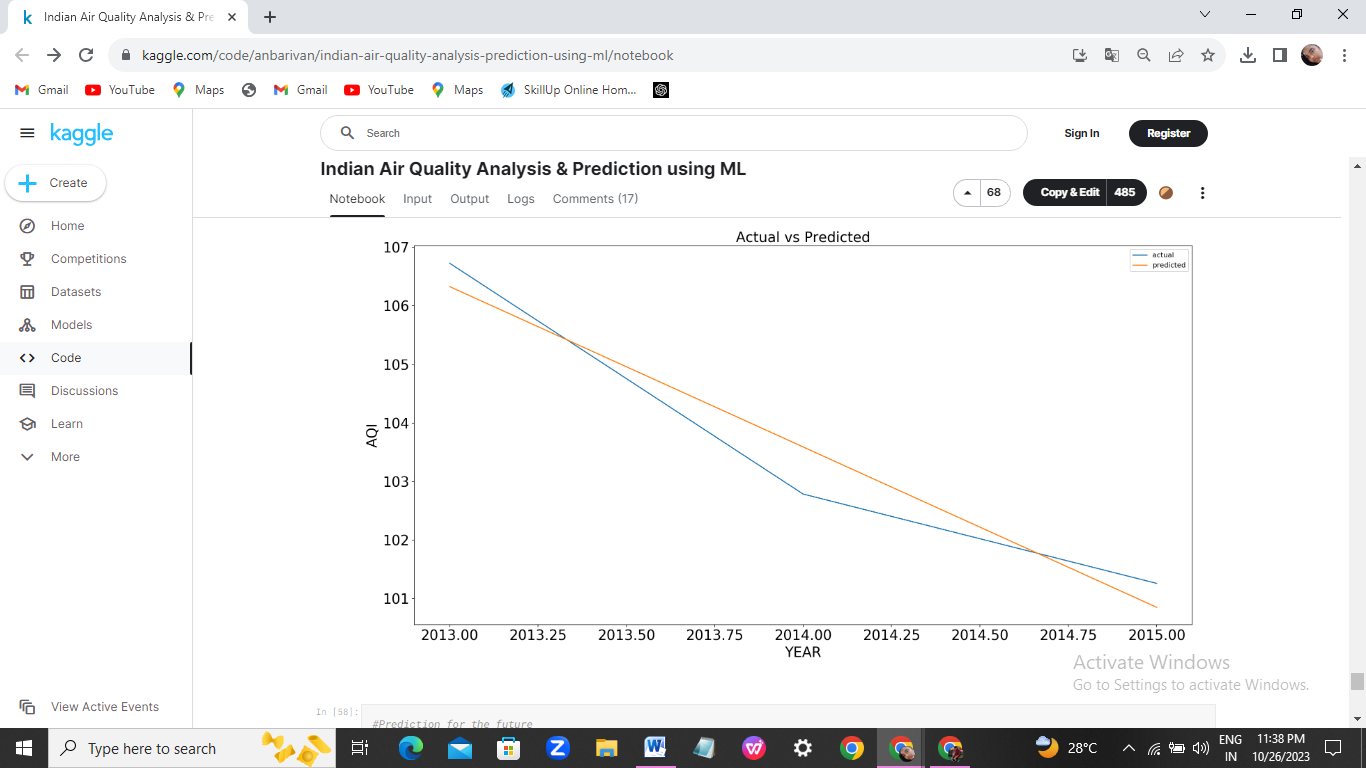
plt.xlabel("YEAR",fontsize=20)

plt.ylabel("AQI",fontsize=20)

plt.tick\_params(labelsize=20)

plt.show()

**Output:**



*#Plotting the cost function...*

plt.title('Cost Function J')

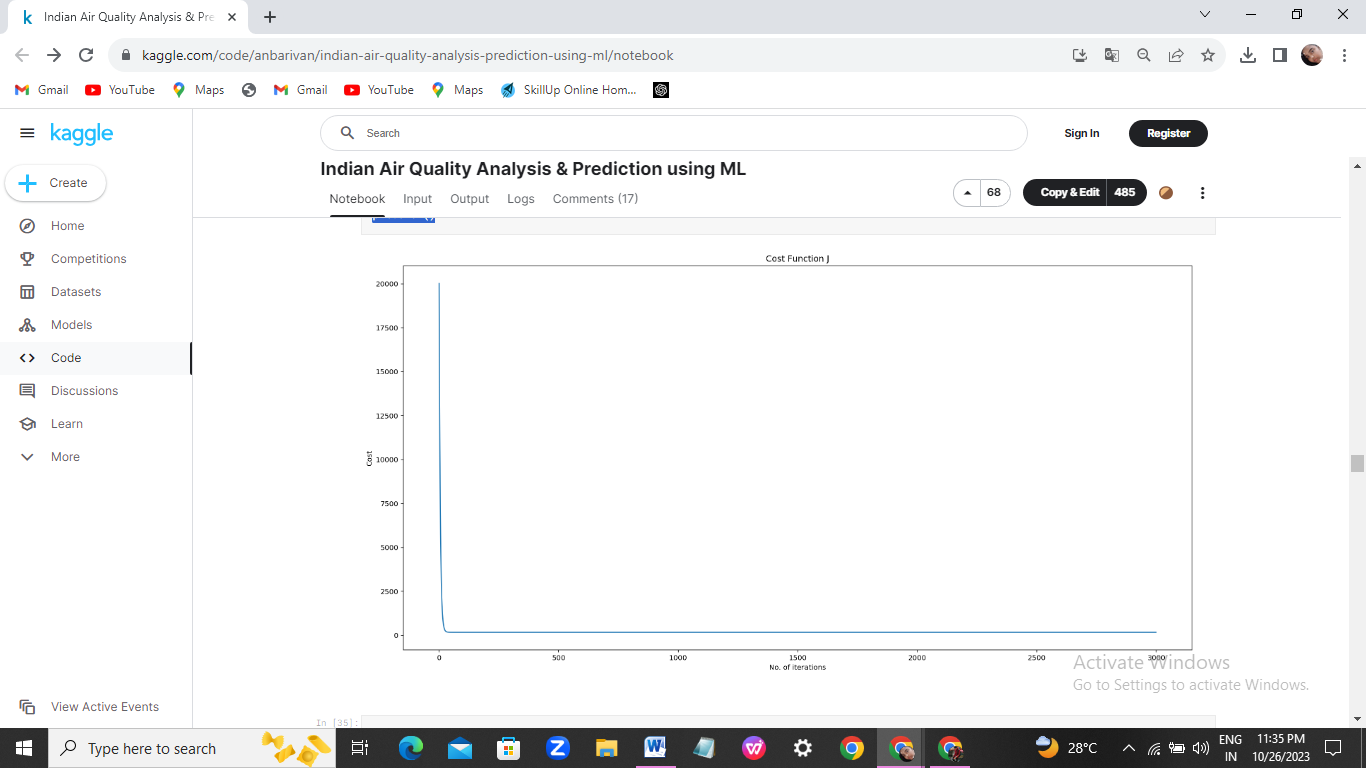
plt.xlabel('No. of iterations')

plt.ylabel('Cost')

plt.plot(past\_costs)

plt.show()

**Output:**



**Interpretation:**

The higher the AQI value, the greater the level of air pollution and the greater the health concern. For example, an AQI value of 50 or below represents good air quality, while an AQI value over 300 represents hazardous air quality.

**The goal of Air quality analysis & prediction :**

Prediction of air pollution index may help in traffic routing and identifying serious pollutants. Modeling of the complex relationships between these variables by sophisticated methods in machine learning is a promising field.

**Rules of Air quality analysis:**

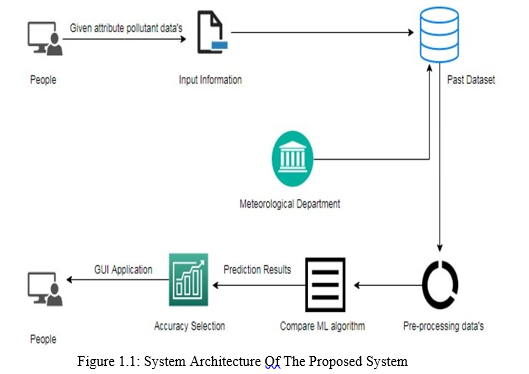
This revised national standard aims to provide uniform air quality for all, irrespective of land use pattern. the provisions of the Air (Prevention & Control of Pollution) Act, 1981, the CPCB has notified fourth version of National Ambient Air Quality Standards (NAAQS) in 2009.

**The life cycle of Air quality analysis:**

To test the importance of incorporating full life cycle supply chain information when performing air quality impact assessment, we perform a sensitivity analysis that considers only emissions from the single phase of each life cycle most frequently associated with its environmental impacts.

**DATASET LINK:**

<https://www.kaggle.com/datasets/india/air-quality>



**CONCLUSION:**

In conclusion, air quality analysis and prediction using advanced data science techniques are crucial for addressing the pressing challenges posed by air pollution. The methods employed, ranging from statistical analyses to machine learning algorithms, allow us to transform complex environmental data into actionable insights