**PREDICTION OF**

**AIR POLLUTION**

**USING**

**RANDOM FOREST**

**CLASSIFIER**

**ABSTRACT**

Air pollution is a growing concern worldwide, and it has serious implications on human health, the environment, and the economy. We present the first comprehensive analysis of government air quality observations for PM10, PM2.5, SO2, NO2 and O3 from the Central Pollution Control Board (CPCB). We address inconsistencies and data gaps in datasets using a rigorous procedure to ensure data representativeness. In this project, we explore the prediction of Air Quality Index (AQI) using the Random Forest algorithm. AQI is a measure of air pollution that is used to communicate the health risks associated with breathing polluted air. We use historical data collected from various air quality monitoring stations in a city and apply the Random Forest algorithm to predict AQI. This study aims to predict the AQI using machine learning algorithms. The AQI is a crucial indicator of air quality, and accurate forecasting can help mitigate the negative effects of air pollution on human health and the environment. The study utilizes data from air quality monitoring stations and meteorological sensors to train and evaluate various machine learning models, including Random Forest, Support Vector Regression, Linear Regression, Logistic Regression and Navie bayes. The accuracy of the algorithm is measured using the root mean square error. The mean square error and the mean absolute error). The results indicate that the Random Forest algorithm performs well in predicting AQI and has the potential to be used as a tool to monitor air quality and help in making decisions to reduce air pollution. The findings of this study can be used by policy makers, city planners, and environmental agencies to design effective strategies to combat air pollution.

**INTRODUCTION**

In recent years, the issue of air pollution has garnered significant attention due to its detrimental effects on public health and the environment. According to the World Health Organization (WHO), air pollution is responsible for millions of premature deaths annually worldwide, making it a pressing global concern. Traditional methods of monitoring air quality rely on sparse networks of stationary sensors, which provide limited spatial coverage and temporal resolution. However, with the advent of machine learning (ML) techniques and the proliferation of IoT (Internet of Things) devices, there is a growing opportunity to improve the accuracy and efficiency of air pollution prediction.

Machine learning offers a promising approach to address the complexities involved in modeling and predicting air quality. By leveraging vast amounts of data from various sources such as meteorological conditions, geographical features, and historical pollution levels, ML algorithms can learn intricate patterns and relationships that influence air quality dynamics. This capability enables more precise forecasting of pollution levels across different spatial and temporal scales, thereby supporting proactive measures for pollution control and public health management.

This paper explores the application of machine learning techniques in predicting air pollution levels. It begins by reviewing existing methods and challenges in air quality monitoring and prediction. Subsequently, it discusses the potential benefits of ML-based approaches, including improved accuracy, real-time monitoring capabilities, and scalability. Furthermore, the paper examines several ML models commonly used for air quality prediction, highlighting their strengths and limitations.

By integrating advanced data analytics with environmental science, ML-based air pollution prediction represents a pivotal advancement towards sustainable urban development and public health improvement. This research contributes to the growing body of knowledge aimed at harnessing technology to mitigate the adverse impacts of air pollution and promote healthier living environments.

**LITERATURE SURVEY**

B.P. Pande came up with a very interesting study about the analysis air pollution and respiratory health. He analyzes the six years of air pollution data from Indian cities, focusing on twelve pollutants and the AQI. The dataset undergoes preprocessing and visualization to uncover patterns and data imbalance is addressed using resampling techniques. Five popular ML models are evaluated and compared using standard performance metrics. In this research XGBoost model achieved the highest accuracy among the Machine Learning models evaluated for predicting AQI values. The Support Vector Machine (SVM) model exhibited the lowest accuracy in comparison to other models in this research. While executing the dataset he knew that PM2.5 ad PM10 exhibit seasonal patterns with increased pollution with winter compared to summer. Additionally, SO2 levels have risen since 2018, while O3 levels remained variations in their concentration.

K. Kumar carried out a detailed literal survey on the impact of air pollution on human health and environment leading to premature deaths and substantial economic losses globally. It emphasizes the importance of ground monitoring and remote sensing technologies in understanding and addressing air pollution dynamicas. In this research he used cluster analysis. Cluseter1 focuses on applying ml models to identify specific signals in software and VOCs. ML models have been utilized for monitoring various pollutants such as VOCs, PM2.5, CO and NO2 with Artificial Neural Network (ANN) being the most commonly used algorithm. The text also mentions the challenges of single sensor detection and the introduction of ML algorithms in data processing modules for calibration and enhanced detection efficiency, enabling rapid identification of gases in gas mixture bases on signal values.

Aditya Kumar Agarwall and his team researched on the air pollutants in the laboratories of an Indian engineering institute. In their research they knew that the global concern of air pollution, particularly indoor air pollution in developing countries like India. It highlights the lack of guidelines for indoor Air Quality in India and emphasizes the need for continuous monitoring due to its negative impact on human health. This is focus on outdoor air pollution and the adverse effects of IAP on human health, economic growth and living habits. Here they used cluster analysis for indoor air pollutions include K-means, DBSCAN, Affinity Propagation and disadvantages with K-means being population due to its fast implementation. This research identify outlier concentration in PM and CO2 levels. It mentions the impact of surprise activities like renovation work and construction on air quality. Additionally, it discusses the correlation between different sized PM concentration in various laboratories, including distinct emission sources.

**FUTURE SCOPE**

The future scope of an air pollution prediction project is promising and multifaceted, with several areas of potential development and application:

1. **Enhanced Accuracy and Models**: Continuously improving predictive models using advanced machine learning algorithms (such as deep learning, ensemble methods) and incorporating more comprehensive datasets (including meteorological data, satellite imagery, and real-time sensor data) can enhance accuracy.
2. **Real-Time Prediction**: Developing capabilities for real-time or near-real-time prediction of air quality indices (AQI) can enable proactive measures to mitigate pollution impacts.
3. **Spatial Resolution**: Increasing the spatial resolution of predictions to provide localized forecasts can be beneficial for urban planning, public health interventions, and policy-making at a city or neighborhood level.
4. **Integration with IoT and Sensor Networks**: Integrating with Internet of Things (IoT) devices and sensor networks to gather continuous data and validate predictive models can improve reliability and responsiveness.
5. **Predictive Analytics for Health Impacts**: Linking air quality predictions with health data to forecast potential health impacts and provide early warnings to vulnerable populations.
6. **User-Friendly Interfaces**: Developing user-friendly interfaces (such as mobile apps or web dashboards) that deliver personalized air quality forecasts and recommendations to the public.
7. **Policy Support**: Supporting policymakers with predictive insights to design effective environmental policies and regulations aimed at reducing air pollution.
8. **Climate Change Adaptation**: Studying the interaction between air quality and climate change to anticipate future trends and adapt mitigation strategies accordingly.
9. **Cross-Domain Applications**: Exploring cross-domain applications, such as integrating air quality predictions with traffic management systems, urban planning, and renewable energy production, to optimize environmental sustainability.
10. **Global Collaboration**: Encouraging international collaboration and data sharing to address transboundary air pollution issues and develop globally applicable solutions.
11. **Ethical Considerations**: Addressing ethical considerations related to data privacy, equity in access to air quality information, and transparency in algorithmic decision-making.

Overall, the future of air pollution prediction projects lies in advancing technology, interdisciplinary collaboration, and leveraging data-driven insights to foster healthier and sustainable environments for communities worldwide.

Essemtions:

Delberg (2019) Air Pollution and its impact on business: the silent pandemic

<https://www.cleanairfund.org/wpcontent/uploads/2021/04/01042021_Busine%20ss-Cost-of-Air-Pollution_Long-Form-Report.pdf>

The survival of humanity is heavily reliant on air quality, which has been adversely affected by ongoing industrial, transportation and domestic activities. Monitoring and predicting air quality is curcial. This research analyzes six years of air pollution data from 23 Indian cities, utilizing machine learning methods. Exploratory data analysis reveals hidden patterns, with a notable decrease in pollutant levels during the pandemic year 2020.The Guassian Naïve Bayes model achieves the highest accuracy, while the Support Vector Machine model shows the lowest accuracy. The study underscores the importance of machine learning in air quality analysis and prediction.

A diesel Engine’s performance and exhaust emissions

<https://www.sciencedirect.com/science/article/abs/pii/S0306261904000376>

In this research paper they investigate the performance and exhaust emissions of a diesel engine using artificial neural networks(ANNs). It specifically examines how variables such as injection pressure, engine speed and throttle position effect engine performance and emissions. In this they are study on some points. Those are **Injection Pressure** which is focuses on a design injection pressure of 150 bar, with experiments conducted at four different pressures. **Throttle Positions** positions tested were 50%,75% and100%. **Performance Metrics** are used to measure various performance metrics including engine torque, power, brake mean, fuel flow and exhaust emission . **Neural Network Design**  is utilized a back-propagation learning algorithm with different configuration, including single and two hidden layers and employed Logistic sigmoid transfer function.

<https://www.sciencedirect.com/science/article/abs/pii/S1309104215301689?via%3Dihub>

This research paper aims that to provide a comprehensive review of Indoor Air Quality(IAQ) guidelines and standards set by international agencies like ASHRAE,HKEPD,WHO AND NHMRC. It summarizes key indoor air pollutants such as CO2,NO2,HCHO,CO,SO2,PM2.5 and PM10 along with factors like relative humidity, temperature and air movement that impact IAQ. The focus is on addressing Sick Building Syndrome caused by these pollutants and factors and reviewing guidelines to prevent SBS issued by regulatory bodies.

**METHODOLOGY**

The algorithms used to train the model are LinearRegression, Logistic Regression, Navie Bayes, SVM and Random Forest classifier

**LINEAR REGRESSION**

Linear regression is a type of supervised machine learning algorithm that computes the linear relationship between the dependent variable and one or more independent features by fitting a linear equation to observed data.

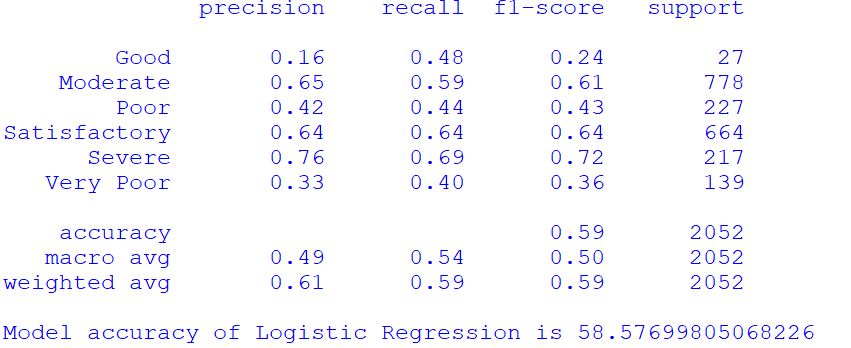
When there is only one independent feature, it is known as Simple Linear Regression, and when there are more than one feature, it is known as Multiple Linear Regression.

By applying this linear regression we got accuracy of **22.18916203256431**

**LOGISTIC REGRESSION**

Logistic regression is a data analysis technique that uses mathematics to find the relationships between two data factors. It then uses this relationship to predict the value of one of those factors based on the other. The prediction usually has a finite number of outcomes, like yes or no.

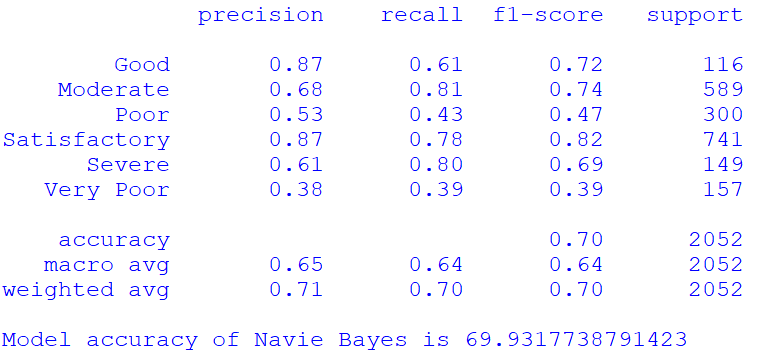
By applying this we get the accuracy of **58.57699805068226**



**NAVIE BAYES**

Naïve Bayes is part of a family of generative learning algorithms, meaning that it seeks to model the distribution of inputs of a given class or category. Unlike discriminative classifiers, like logistic regression, it does not learn which features are most important to differentiate between classes.

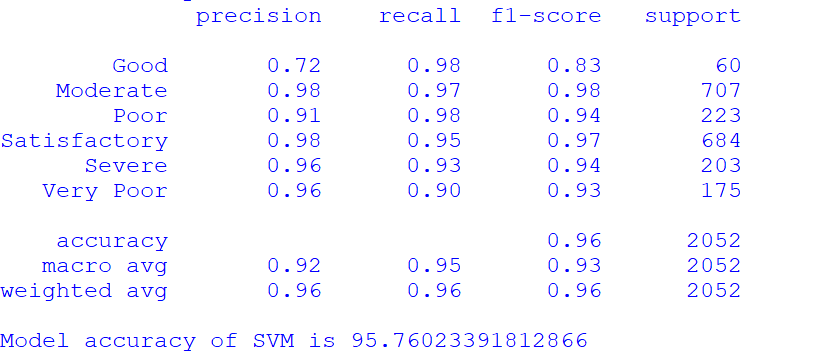
By applying Navie Bayes we get the accuracy of **69.9317738791423**



**SVM**

Support Vector Machine (SVM) is a powerful machine learning algorithm used for linear or nonlinear classification, regression, and even outlier detection tasks. SVMs can be used for a variety of tasks, such as text classification, image classification, spam detection, handwriting identification, gene expression analysis, face detection, and anomaly detection. SVMs are adaptable and efficient in a variety of applications because they can manage high-dimensional data and nonlinear relationships.

By applying SVM we get the accuracy of **95.76023391812866**

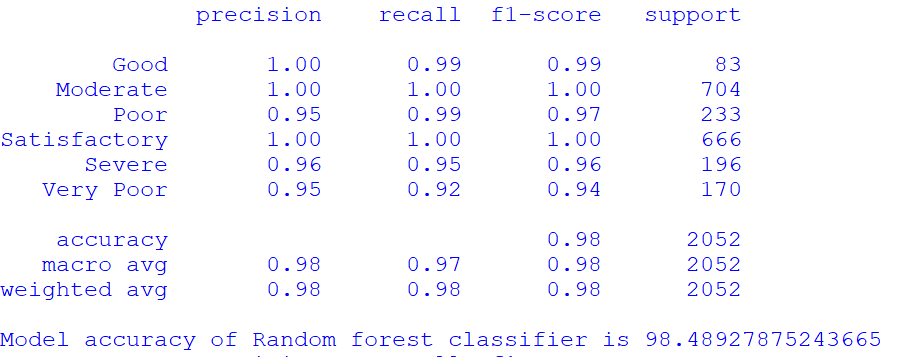


**RANDOM FOREST CLASSIFIER**

Random Forest is a popular machine learning algorithm that belongs to the supervised learning technique. It can be used for both Classification and Regression problems in ML. It is based on the concept of **ensemble learning,** which is a process of *combining multiple classifiers to solve a complex problem and to improve the performance of the model.*

***Random Forest is a classifier that contains a number of decision trees on various subsets of the given dataset and takes the average to improve the predictive accuracy of that dataset.***

By applying this we get the accuracy of **98.48927875243665**



Since the accuracy of the Random forest is greater than all other algorithms, we select the Random forest for training and building the model.

**CHAPTER 4**

**SOFTWARE ENVIRONMENT**

**4.1 INTRODUCTION TO PYTHON**

Python is a high-level, interpreted scripting language developed in the late 1980s by Guido van Rossum at the National Research Institute for Mathematics and Computer Science in the Netherlands. The initial version was published at the alt. Sources newsgroup in 1991, and version 1.0 was released in 1994.

Python 2.0 was released in 2000, and the 2.x versions were the prevalent releases until December 2008. At that time, the development team made the decision to release version 3.0, which contained a few relatively small but significant changes that were not backward compatible with the 2.x versions. Python 2 and 3 are very similar, and some features of Python 3 have been back ported to Python 2. But in general, they remain not quite compatible.

Both Python 2 and 3 have continued to be maintained and developed, with periodic release updates for both. As of this writing, the most recent versions available are 2.7.15 and 3.6.5. However, an official End of Life date of January 1, 2020 has been established for Python 2, after which time it will no longer be maintained. If you are a newcomer to Python, it is recommended that you focus on Python 3, as this tutorial will do.

Python is still maintained by a core development team at the Institute, and Guido is still in charge, having been given the title of BDFL (Benevolent Dictator For Life) by the Python community. The name Python, by the way, derives not from the snake, but from the British comedy troupe Monty Python’s Flying Circus, of which Guido was, and presumably still is, a fan. It is common to find references to Monty Python sketches and movies scattered throughout the Python documentation

**WHY CHOOSE PYTHON**

For some applications that are particularly computationally intensive like graphics processing or intense number, this can be limiting. In practice,however,for most programs,the difference execution speed is measured in milliseconds, or seconds at most, and not appreciably noticeable to a human user.The expediency of coding in an interpreted language is typically worth it for most applications.

**Python is Free**

The Python interpreter is developed under an OSI-approved opensource license making it free to install use and distribute even for commercial purposes. A version of the interpreter is available for virtually any platform there is, including all flavors of Unix, Windows ,macOS, smart phones and tablets, and probably anything else you ever heard. A version even exists for the half dozen people remaining who use OS/2.

**Python is Portable**

Because Python code is interpreted and not compiled into native machine instructions, code written for one platform will work on any other platform that has the Python interpreter installed. (This is true of any interpreted language, not just Python.)

**Python is Simple**

Python-3 has 33 keywords, and Python-2 has 31.By contrast, C++ has 62, Java has 53, and Visual Basic has more than 120, though these latter examples probably vary Somewhat by implementation or dialect. Python code has a simple and clean structure that is easy to learn and easy to read.In fact,as you will see,the language definition enforces code structure that is easy to read. But It’s Not That Simple for all its syntactical simplicity, Python supports most constructs that would be expected in a very high-level language,including complex dynamic data types, structured and functional programming, and object-oriented programming. Additionally, a very extensive library of classes and functions is available that provides capability well beyond what is built into the language,such as database manipulation. Python accomplishes what many programming languages don’t: the language itself is simply designed, but it is very versatile in terms of what you can accomplish with it.

**Conclusion**

This section gave an overview of the Python programming language, including:

A brief history of the development of Python Some reasons why you might select Python as your language of choice.

Python is a great option, whether you are a beginning programmer looking to learn the basics, an experienced programmer designing a large application, or anywhere in between. The basics of Python are easily grasped, and yet its capabilities are vast. Proceed to the next section to learn how to acquire and install Python on your computer.

Python is an open-source programming language that was made to be easy-to-read and powerful. A Dutch programmer named Guido van Rossum made Python in 1991. He named it after the television show Monty Python's Flying Circus. Many Python.

Python is an interpreted language. Interpreted languages do not need to be compiled to run. A program called an interpreter runs Python code on almost any kind of computer. This means that a programmer can change the code and quickly see the results. This also means Python is slower than a compiled language like C, because it is not running machine code directly.

Python is a good programming language for beginners. It is a highlevel language, which means a programmer can focus on what to do instead of how to do it.

Writing programs in Python takes less time than in some other languages. Python drew inspiration from other programming languages like C, C++, Java, Perl, and Lisp. 32

Python has a very easy-to-read syntax. Some of Python's syntax comes from C, because that is the language that Python was written in. But Python uses whitespace to delimit code: spaces or tabs are used to organize code into groups. This is different from C. In C, there is a semicolon at the end of each line and curly braces ({}) are used to group code. Using whitespace to delimit code makes Python a very easy-to-read language.

**Python use [change / change source]**

Python is used by hundreds of thousands of programmers and is used in many places. Sometimes only Python code is used for a program, but most of the time it is used to do simple jobs while another programming language is used to do more complicated tasks. Its standard library is made up of many functions that come with Python when it is installed. On the Internet there are many other libraries available that make it possible for the Python language to do more things. These libraries make it a powerful language; it can do many different things.

Some things that Python is often used for are:

• Web development

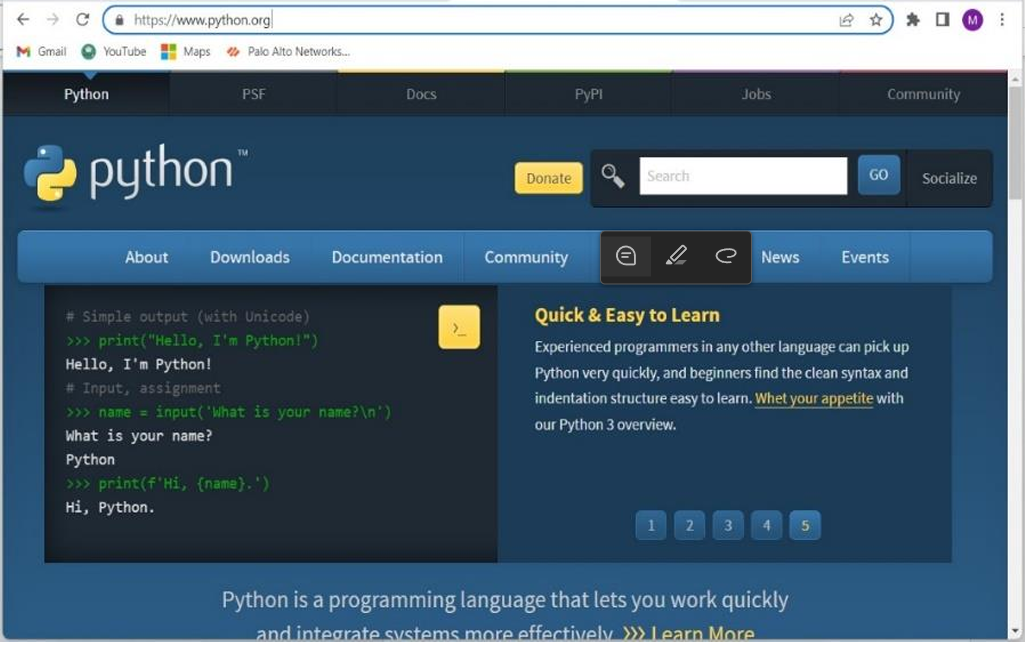
• Scientific programming

• Desktop GUIs • Network programming

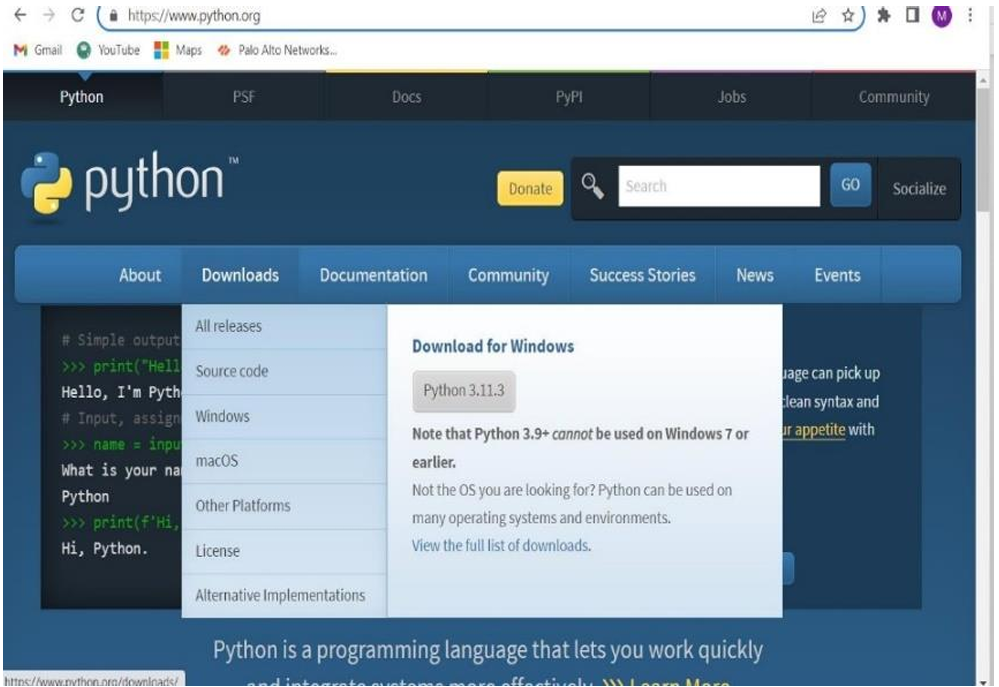
• Game programming

**4.3 STEPS TO INSTALL PYTHON**

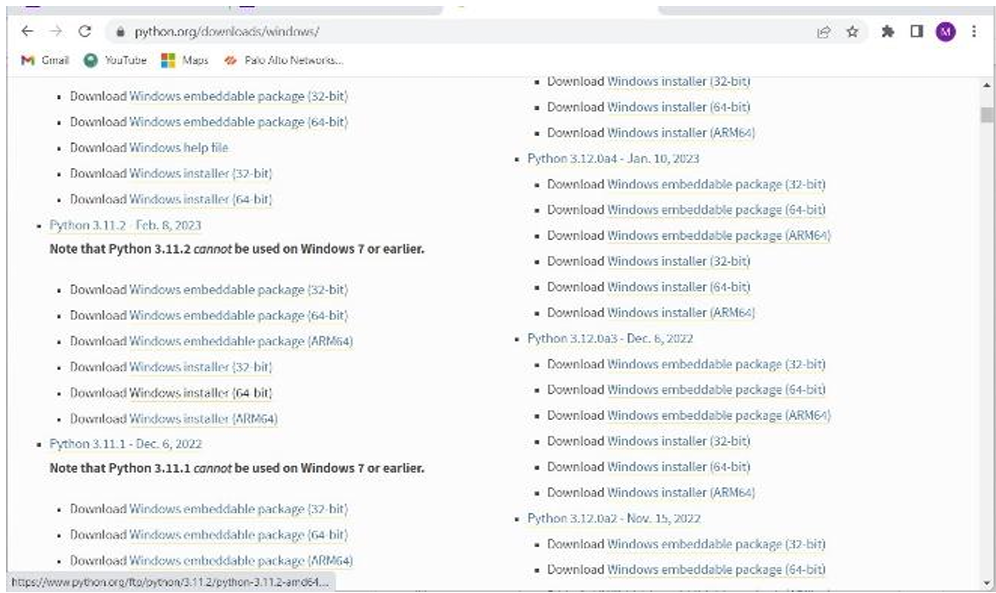
**Step 1:** Search python.org



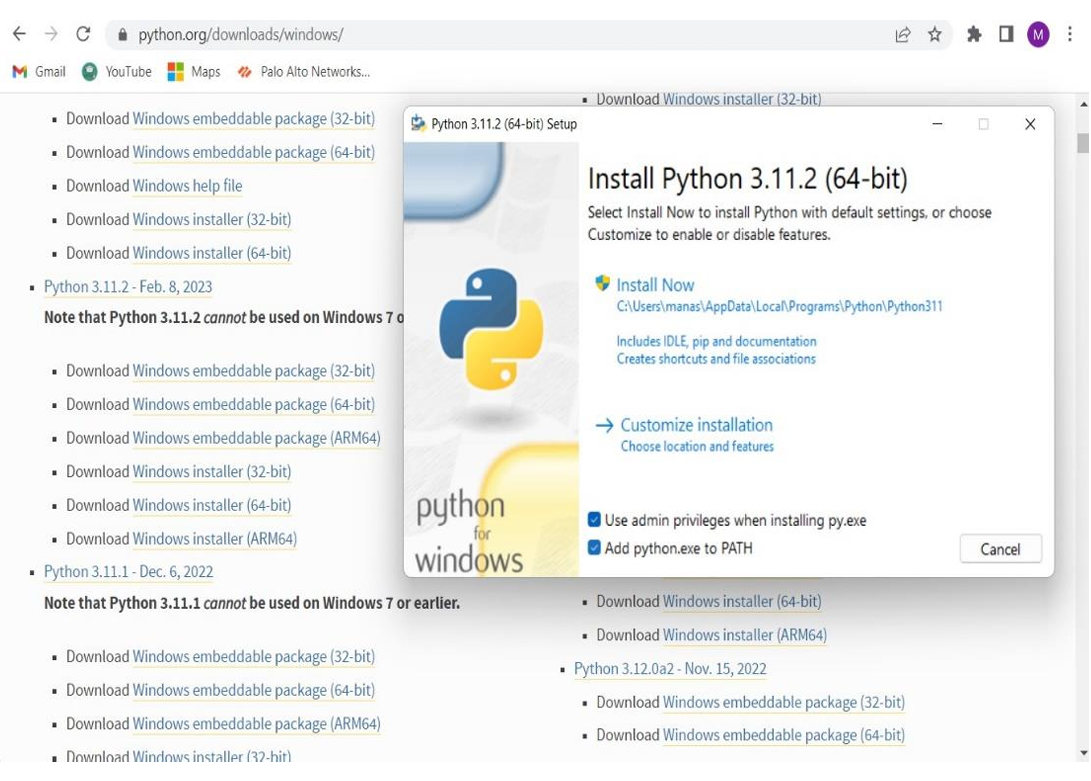
**Step 2:** Go to downloads and select windows



**Step 3:** Download Windows installer(64-bit)



**Ste 4:** Now select python.exe to path and install the IDLE



**Modules**

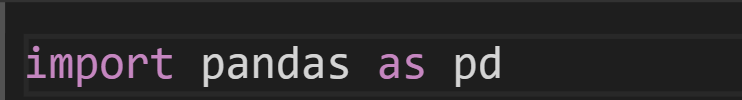
**Pandas :**

pandas is a Python package providing fast, flexible, and expressive data structures designed to make working with “relational” or “labeled” data both easy and intuitive. It aims to be the fundamental high-level building block for doing practical, real-world data analysis in Python. Additionally, it has the broader goal of becoming the most powerful and flexible open source data analysis/manipulation tool available in any language. It is already well on its way toward this goal.

The two primary data structures of pandas, Series (1-dimensional) and DataFrame (2-dimensional), handle the vast majority of typical use cases in finance, statistics, social science, and many areas of engineering. For R users, DataFrame provides everything that R’s data.frame provides and much more. pandas is built on top of NumPy and is intended to integrate well within a scientific computing environment with many other 3rd party libraries.

Many of these principles are here to address the shortcomings frequently experienced using other languages / scientific research environments. For data scientists, working with data is typically divided into multiple stages: munging and cleaning data, analyzing / modeling it, then organizing the results of the analysis into a form suitable for plotting or tabular display. pandas is the ideal tool for all of these tasks

Here is that how we import pandas from the python Libraries



**Numpy :**

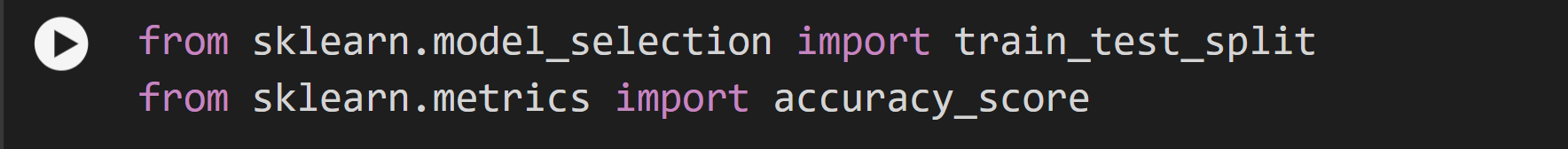
NumPy is a Python library used for working with arrays.It also has functions for working in domain of linear algebra, fourier transform, and matrices. NumPy stands for Numerical Python.

In Python we have lists that serve the purpose of arrays, but they are slow to process.NumPy aims to provide an array object that is up to 50x faster than traditional Python lists.The array object in NumPy is called ndarray, it provides a lot of supporting functions that make working with ndarray very easy.Arrays are very frequently used in data science, where speed and resources are very important.

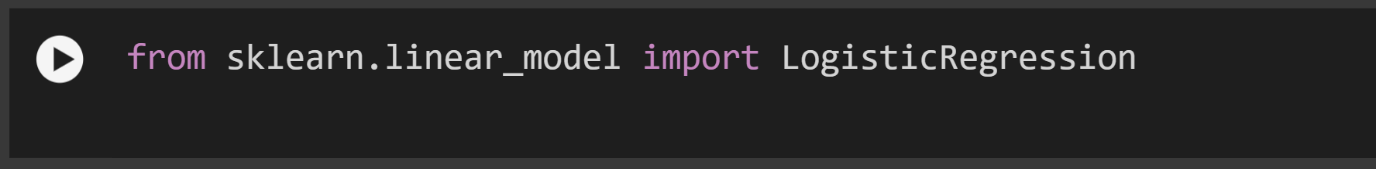
**Scikit-learn :**

scikit-learn (often abbreviated as sklearn) is a popular open-source machine learning library for Python. It provides simple and efficient tools for data mining and data analysis, and it is built on NumPy, SciPy, and Matplotlib. Scikit-learn is widely used for implementing and experimenting with a variety of machine learning models and algorithms.

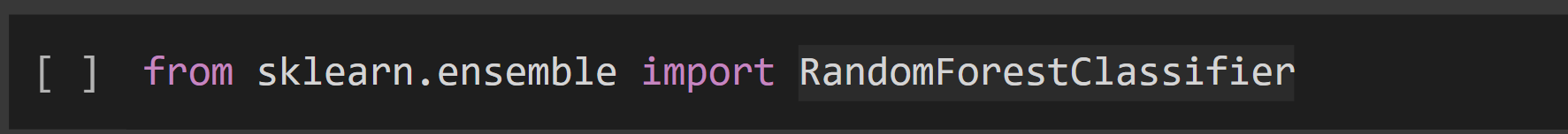
Here is that how we import sklearn from the python Libraries



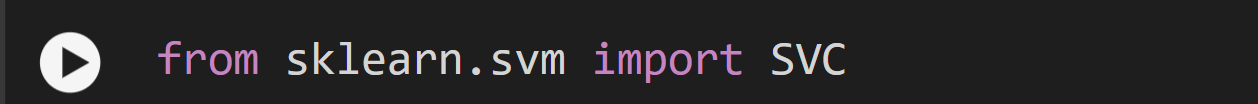
Here is that how we import sklearn for logistic regression



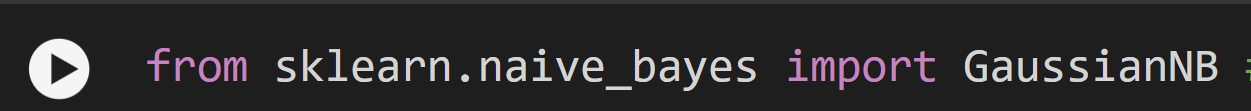
Here is that how we import sklearn for RandomForestClassifier



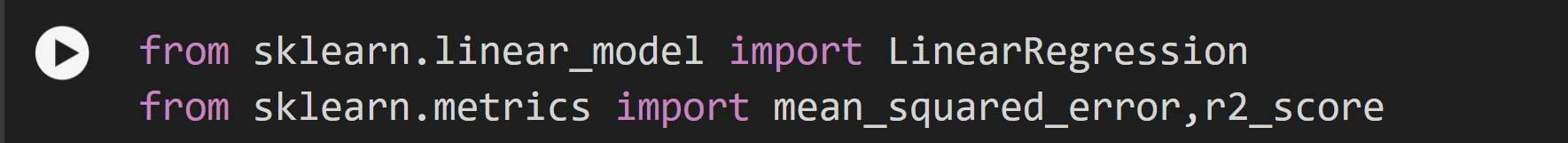
Here is that how we import sklearn for SVM(SupportVectorMachine)



Here is that how we import sklearn for naïve bayes



Here is that how we import sklearn for linear regression



Top of Form

**CHAPTER 7**

**PYTHON CODE**

**VIEW CODE**

import pandas as pd

import joblib

data=pd. read\_ csv('Air\_Pollution\_Datasets.csv')

print(data)

data\_ cleaned=data. drop (columns= ['MID', 'Unnamed: 17', 'Unnamed: 18', 'Unnamed: 19', 'Unnamed: 20'])

print (data\_ cleaned)

columns\_ to\_ fill = ['PM2.5', 'PM10', 'NO', 'NO2', 'NOx','NH3', 'CO', 'SO2', 'O3', 'Benzene', 'Toluene', 'Xylene', 'AQI']

data\_ cleaned [columns\_ to\_ fill] = data\_ cleaned [columns\_ to\_ fill]. fillna (data\_ cleaned [columns\_ to\_ fill]. mean ())

print (data\_ cleaned. columns)

print (data\_ cleaned. isnull(). sum())

from sklearn. model\_ selection import train\_test\_split

from sklearn. metrics import accuracy\_score,classification\_report

features = data\_ cleaned [['PM2.5', 'PM10', 'NO', 'NO2', 'NOx', 'NH3','CO', 'SO2', 'O3', 'Benzene', 'Toluene', 'Xylene', 'AQI']]

target = data\_ cleaned ['AQI\_ Bucket']

x\_train, x\_test, y\_train,y\_test=train\_test\_split(features, target, test\_size=0.2,random\_state=42)

from sklearn. linear\_model import LogisticRegression

model=LogisticRegression()

model.fit(x\_train ,y\_train)

y\_pred=model.predict (x\_test)

dd=classification\_report (y\_pred, y \_test)

print(dd)

print ("Model accuracy of Logistic Regression is", accuracy\_ score(y\_test,y\_pred)\*100)

from sklearn.ensemble import RandomForestClassifier

model1=RandomForestClassifier()

model1.fit(x\_train,y\_train)

test\_pred1=model1.predict(x\_test)

aa=classification\_report(test\_pred1,y\_test)

print(aa)

print ("Model accuracy of Random forest classifier is", accuracy\_score(y\_test,test\_pred1)\*100)

from sklearn.svm import SVC

model2=SVC()

model2.fit(x\_train,y\_train)

pred=model2.predict(x\_test)

cc=classification\_report(pred ,y\_test)

print(cc)

print ("Model accuracy of SVM is",accuracy\_score(y\_test,pred)\*100)

from sklearn.naive\_bayes import GaussianNB # Fixed the typo in module name

model3 = GaussianNB()

model3.fit(x\_train, y\_train)

pred1=model3.predict(x\_test)

bb=classification\_report(pred1,y\_test)

print(bb)

print("Model accuracy of Navie Bayes is",accuracy\_score(y\_test,pred1)\*100)

'''from sklearn.linear\_model import LinearRegression

from sklearn.metrics import mean\_squared\_error,r2\_score

from sklearn.preprocessing import LabelEncoder

label\_encoder = LabelEncoder()

y\_train\_encoded = label\_encoder.fit\_transform(y\_train)

y\_test\_encoded = label\_encoder.transform(y\_test)

model4=LinearRegression()

model4.fit(x\_train,y\_train\_encoded)

pr=model4.predict(x\_test)

r2=r2\_score(y\_test\_encoded, pr)

print(f"R-squared:{r2\*100}")

ee=classification\_report(pr,y\_test)

print(ee)'''

from sklearn.preprocessing import LabelEncoder

encoder=LabelEncoder()

data['AQI\_Bucket']=encoder.fit\_transform(data['AQI\_Bucket'])

data.head()

joblib.dump(model1,"my\_model.h5")

This code will train the model using the dataset and then it will predict the output of the data given.

**PREDICT**

import joblib

import pandas as pd

# Load the model

model = joblib.load("my\_model.h5")

# Input values for features

a = float(input("Enter the value of PM2.5: "))

b = float(input("Enter the value of PM10: "))

c = float(input("Enter the value of NO: "))

d = float(input("Enter the value of NO2: "))

e = float(input("Enter the value of NOx: "))

f = float(input("Enter the value of NH3: "))

g = float(input("Enter the value of CO: "))

h = float(input("Enter the value of SO2: "))

i = float(input("Enter the value of O3: "))

j = float(input("Enter the value of Benzene: "))

k = float(input("Enter the value of Toluene: "))

l = float(input("Enter the value of Xylene: "))

m = float(input("Enter the value of AQI: "))

# Create a dictionary with input values

d1 = {

"PM2.5": a, "PM10": b, "NO": c, "NO2": d, "NOx": e,

"NH3": f, "CO": g, "SO2": h, "O3": i, "Benzene": j,

"Toluene": k, "Xylene": l, "AQI": m

}

# Create a DataFrame from the dictionary

d2 = pd.DataFrame([d1])

# Assuming d2 already has the correct column names, proceed to predict

p = model.predict(d2)

print(p)

**INIT**

from flask import Flask, render\_template, request, redirect, url\_for

import os

import sys

import pandas as pd # type: ignore

import joblib

app = Flask(\_\_name\_\_)

if sys.stderr is None:

sys.stderr=open(os.devnull,'w')

@app.route('/')

def index():

return render\_template('index.html')

@app.route('/submit', methods=['POST'])

def submit():

try:

model=joblib.load("my\_model.h5")

a=float(request.form['PM2.5'])

b=float(request.form['PM10'])

c=float(request.form['NO'])

d=float(request.form['NO2'])

e=float(request.form['NOx'])

f1=float(request.form['NH3'])

g=float(request.form['CO'])

h=float(request.form['SO2'])

i=float(request.form['O3'])

j=float(request.form['Benzene'])

k=float(request.form['Toluene'])

l=float(request.form['Xylene'])

m=int(request.form['AQI'])

d1 = {'PM2.5': a, 'PM10': b, 'NO': c, 'NO2': d, 'NOx': e,'NH3': f1, 'CO': g, 'SO2': h, 'O3': i, 'Benzene': j,'Toluene': k, 'Xylene': l, 'AQI': m}

# Create a DataFrame from the dictionary

d2 = pd.DataFrame([d1])

# Assuming d2 already has the correct column names, proceed to predict

result = model.predict(d2)

except ValueError:

result = "Invalid input! Please enter numbers only."

return redirect(url\_for('result', result=result))

@app.route('/result')

def result():

result = request.args.get('result')

return render\_template('result.html', result=result)

if \_\_name\_\_ == '\_\_main\_\_':

app.run(debug=True,port=8080)

**CHAPTER-8**

**FINAL RESULT**

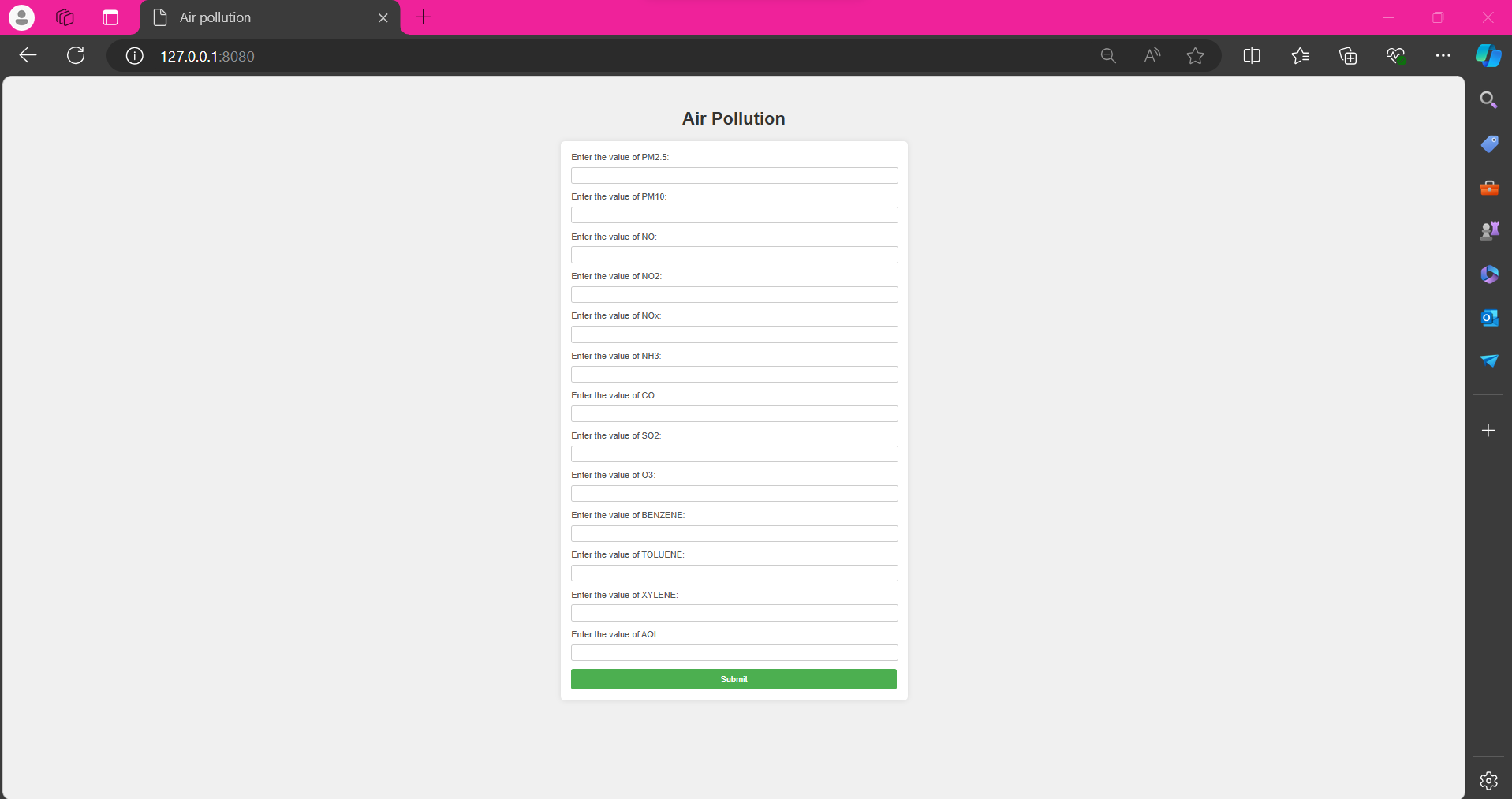
**OUTPUT**

**ACTIVATING SERVER**

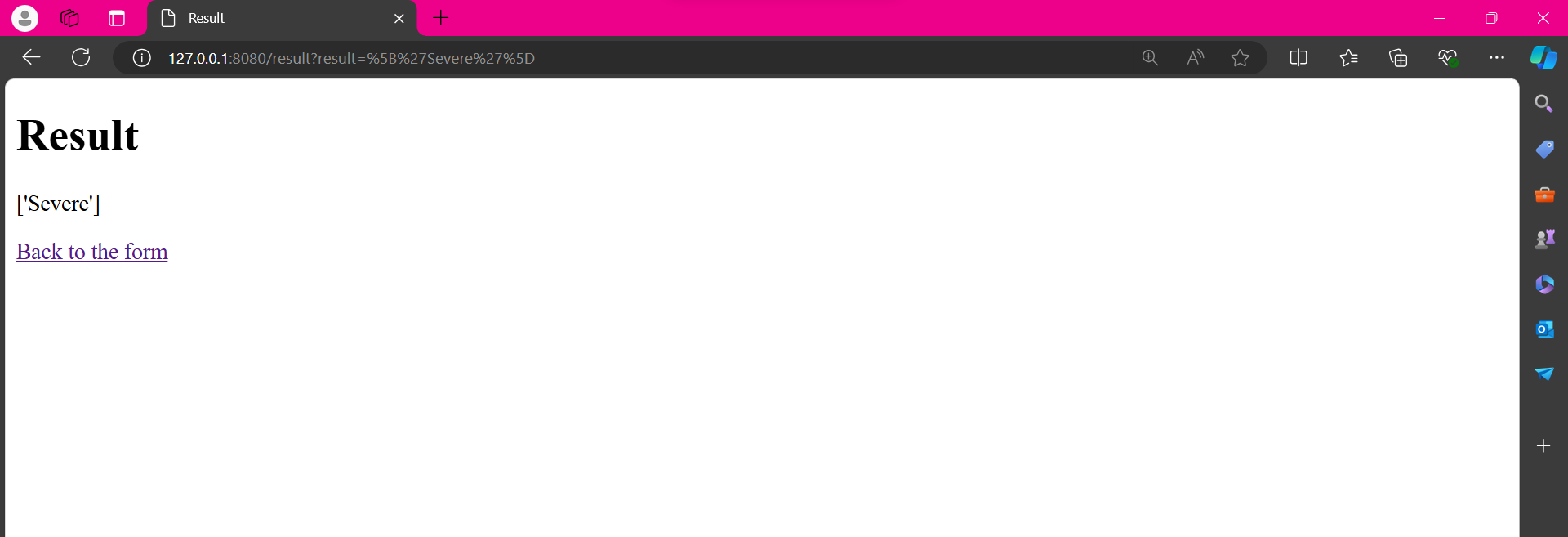


Open the main code and copy the IP address

**WEB PAGE**



After entering the values, the result that is the intensity of the air pollution will be predicted by the model. It will tell whether it is good, poor ,very poor,moderate or severe air pollution.



**CHAPTER-9**

**CONCLUSION**

In conclusion, random forest is a powerful machine learning algorithm that can be used for air quality index prediction. It is a popular method for its ability to handle complex, high-dimensional datasets and to identify important features for prediction. By using random forest to analyze various air quality parameters, such as temperature, humidity, and particulate matter concentrations, it is possible to accurately predict the air quality index at a given location and time. However, it is important to note that prediction accuracy can be affected by the quality and quantity of data used to train the model, as well as other external factors such as weather conditions and human activity.

Despite significant advancements in air pollution control technology and management techniques, the desired outcome remains a long way off. Hence, there is a need to strengthen available policies and technologies on the identified research gaps in India’s air pollution related aspects. Further, the identified gaps can be divided into three areas re- search, policies and economic scale. First, a research constraint has been identified that necessitates a calibrated management system based on data generated by a broad countrywide network to take legislative steps to procure development and growth in the arena of air quality management while addressing prevalent lacunas. Second, this review also identifies a policy constraint in which a country’s national air quality standards must be revisited in light of higher background concentrations of pollutants. Third, the vehicle sector contributes significantly to air pollution which needs to be manage through strategic phasing of reduction of tail pipe emission. The review highlights that the sector has all three identified gaps, indicating that more study is needed to establish region-specific vehicle emission factors and assess real-time vehicular emissions to reinforce control activities.

Bottom of Form