A Machine Learning Model for Air Quality Prediction in Smart Cities

Manogna Devalla, M.Pushpa Pavani, Divija Bandari, Himaja Luri
Department of ECM, Sreenidhi Institute of Science and Technology

ABSTRACT

Air pollution is one of the major environmental issues facing our planet and contributing to its global warming. As defined by the world health organization (WHO), air pollution denotes the presence of harmful substances that cause detrimental changes to air quality. Numerous studies have shown that long-term exposure to such toxic and poisonous air components has several adverse effects on human health and the living ecosystem in general. Furthermore, statistical studies have shown that pollution is tightly associated with mortality and morbidity. This is because it can cause a variety of diseases that range from cough and asthma to heart attacks and cancers.

The most common air pollutants can be broadly classified into three categories: solid particles (e.g.mercury, lead), gaseous pollutants (Nitrogen oxide, NO2, Carbon monoxide, CO, sulphur dioxide,SO2), and a mixture of liquid droplets and small particles called particulate matters .The concentrations of such components define the quality of the air. At high air pollutants' concentrations, the atmospheric composition experiences severe changes, and is thus subject to a higher damage. Therefore, pollution levels can be evaluated by measuring the concentration of such elements in the air. These measurements are generally performed by air monitoring stations on hourly, daily, and monthly basis.

INTRODUCTION

PROJECT INTRODUCTION

Air pollution is one of the major environmental issues facing our planet and contributing to its global warming. As defined by the world health organization (WHO), air pollution denotes the presence of harmful substances that cause detrimental changes to air quality. Numerous studies have shown that long-term exposure to such toxic and poisonous air components has several adverse effects on human health and the living ecosystem in general. Furthermore, statistical studies have shown that pollution is tightly associated with mortality and morbidity. This is because it can cause a variety of diseases that range from cough and asthma to heart attacks and cancers.

The most common air pollutants can be broadly classified into three categories: solid particles (e.g.mercury, lead), gaseous pollutants (Nitrogen oxide, NO2, Carbon monoxide, CO, sulphur dioxide,SO2), and a mixture of liquid droplets and small particles called particulate matters .The concentrations of such components define the quality of the air. At high air pollutants' concentrations, the atmospheric composition experiences severe changes, and is thus subject to a higher damage. Therefore, pollution levels can be evaluated by

measuring the concentration of such elements in the air. These measurements are generally performed by air monitoring stations on hourly, daily, and monthly basis.

Problem Definition

The goal is to develop a machine learning model for real-time air quality forecasting, to potentiallyreplace the updatable supervised machine learning classification models by predicting results in the form of best accuracy by comparing supervised algorithm. Problem Description/ Problem Statements: Monitoring and preserving air quality has become one of the most essential activities in many industrial and urban areas today. The quality of air is adversely affected due to various forms of pollution caused by transportation, electricity, fuel uses etc. The deposition of harmful gases is creating a serious threat for the quality of life in smart cities. With increasing air pollution, we need to implement efficient air quality monitoring models which collect information about theconcentration of air pollutants and provide assessment of air pollution in each area.

Objective of Project

Investigate a dataset of air pollutants records for India meteorological sector using machine learning technique. To identifying air quality is more difficult. We try to reduce this risk factor behind predicting from Air Quality Index (AQI) of India to safe human to save lots of meteorological efforts and assets and to predict whether assigning the air quality is bad or good.

Existing System

The best statistical method for predicting time series data is the autoregressive integrated moving average model (ARIMA). It has several advantages in terms of its statistical properties, potential for a wide range of applications and extendibility. It was demonstrated to reach accuracies around 85% for forecasting AQI monthly values when compared to the performance of ARIMA with a Holt exponential smoothing model and proved the superiority of the ARIMA model for forecasting AQI daily values. However, this method requires extensive manual intervention in terms of selecting the data fed into the system, as it has a low tolerance towards outliers.

Proposed System

In the proposed system, the air quality dataset is downloaded, which is available in Excel format or we can use that in CSV Format. The comma-separated value data format can easily be processed and analyzed fast using a computer and the data utilized for various purposes. It is imported to the project by using apanda packages. The dataset contains unique important attributes that help in air quality prediction. Initially, the dataset is preprocessed with suitable techniques to remove the inconsistent and missing valued data, and the needed features from the dataset are elected for better results. Then the dataset is split off into training and test dataset in order to evaluate the performance of the model. The processed data sets are analyzed through

different regression analysis techniques for accurate results. Regression analysis is the form of a predictive modeling technique that investigates the relationship between a dependent and independent variable. This technique is used for forecasting or predicting, time series modeling, and finding the causal effect relationship between the variables. Regression analysis is a method of analyzing and modeling data. There are different kinds of regression techniques available to make predictions namely Linear regression, Support vector regression, Decision tree regression and Lasso regression.

1.SYSTEM ANALYSIS

3.1 FUNCTIONAL REQUIREMENTS

Functional Requirements are also called functional specifications. In software and system engineering, a functional requirement can range from the high-level abstract statement of the sender necessity to detailed mathematical functional requirement specification. Collecting the system functional and technical requirements in one place will give you clarity and create a requirement document to start the software selection process. Additionally, this document will serve as a constant point of reference for you and your team during software selection and implementation. All the standard libraries like NumPy, pandas, matplotlib and seaborn are imported in this step. We use numpy for linear algebra operations, pandas for using data frames, matplotlib and seaborn for plotting graphs.

3.2 PERFORMANCE REQUIREMENTS

Performance is measured in terms of the output provided by the application. Requirement specification plays an important part in the analysis of a system. Only when the requirement specifications are properly given,it is possible to design a system, which will fit into required environment. It rests largely with the users of the existing system to give the requirement specifications because they are the people who finally use the system. This is because the requirements must be known during the initial stages so that the system can be designed according to those requirements. It is very difficult to change the system once it has been designed and on the other hand designing a system, which does not cater to the requirements of the user, isof no use. The requirement specification for any system can be broadly stated as given below:

• The systemshould be able to interface with the existing system

• The system should be accurate

• The system should be better than the existing system. The existing system is completely dependent on the user to perform all the duties

3.2 Software Requirements

Operating system: Windows XP, 7 and aboveTool: Google Colabs

Language: Python 3.7 and above

Python NumPy, Pandas and Sklearn Libraries

3.3 Hardware Requirements

Hard Disk: min 20 GBRAM: min 2 GB

Processor: core 2 duo and above

SYSTEM DESIGN

Systems design is the process of defining the architecture, components, modules, interfaces, and data for a system to satisfy specified requirements. One could see it as the application of systems theory to product development. Object-oriented analysis and design methods are becoming the most widely used methods for computer systems design

4.1 SYSTEM ARCHITECTURE

A system architecture is the conceptual model that defines the structure, behavior, and more views of a system. An architecture description is a formal description and representation of a system, organized in a way that supports reasoning about the structures and behaviors of the system.

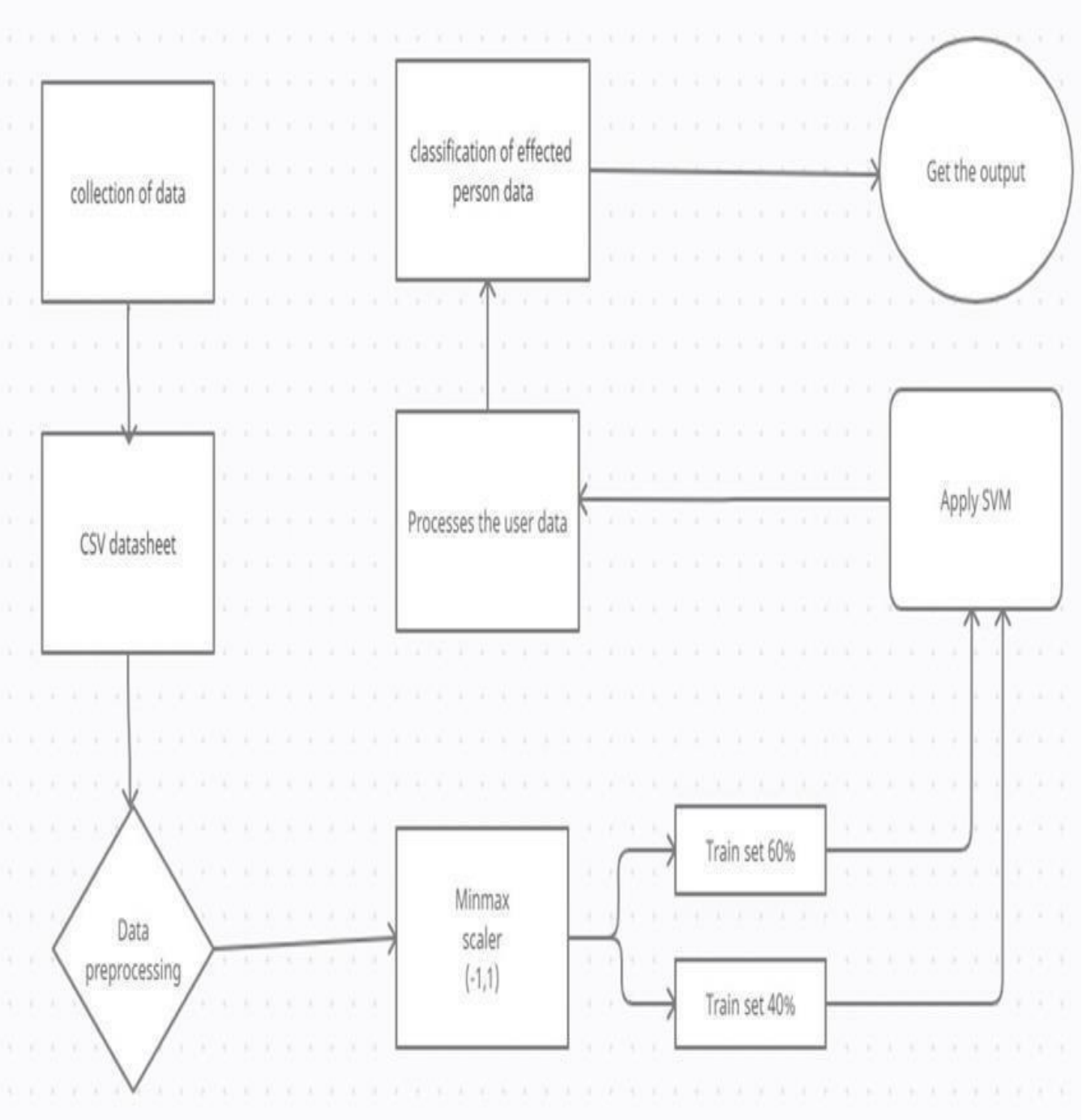


Figure 4.1 Architecture of Proposed system

4.2 Data Flow Diagram

A data-flow diagram is a way of representing a flow of data through a process or a system (usually an information system). The DFD also provides information about the outputs and inputs of each entity and the process itself. A data-flow diagram has no control flow — there are no decision rules and no loops

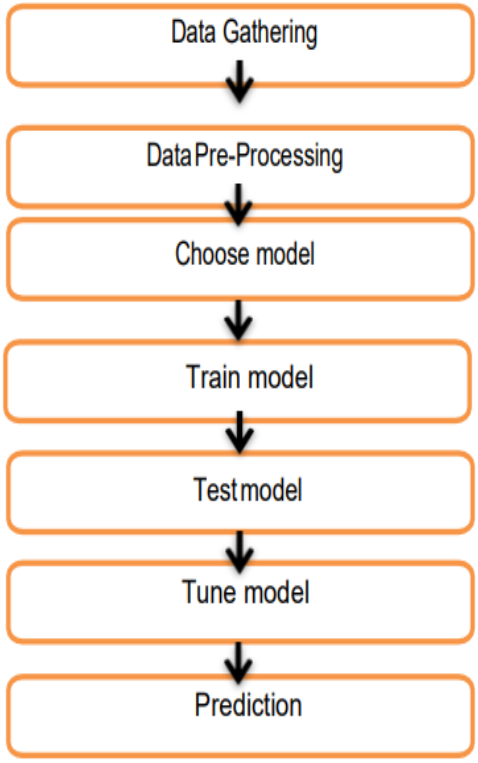


Fig 4.2 Data Flow Diagram

4.3 UML Diagrams

4.3.1 Use Case Diagram

A utilization case chart is a sort of conduct diagram characterized by and created from a Use- case examination in the Unified Modeling Language (UML). It will probably get a graphical portrayal of cutting-edge materials regarding entertainers, targets (addressed as use cases), and any incongruencies between those utilization cases. A utilization case chart's primary objective is to show which framework exercises are led for which entertainer. The jobs of the framework's entertainers can be highlighted.

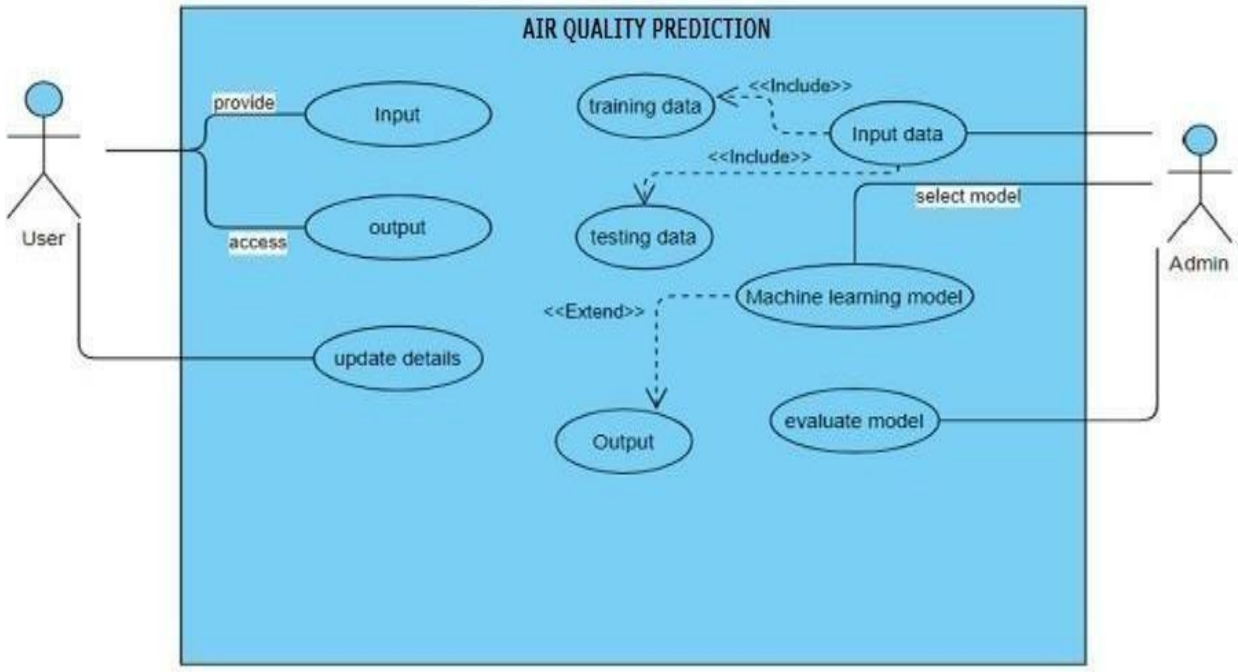


Figure 4.3.1 use case diagram

4.3.2 Class Diagram

A class figure in the Unified Modeling Language (UML) is a context diagram in software development that depicts the system architecture by displaying the system's classes, qualities, procedures (or methods), and interactions among the categories. It clarifies which categorycontains data.

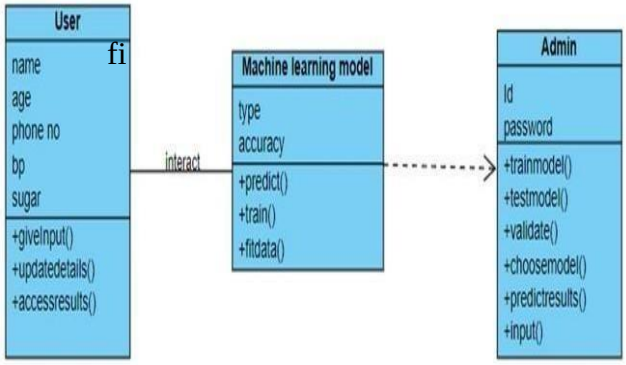


Figure 4.3.2 Class Diagram

4.3.3 Sequence Diagram

In the Unified Modeling Language (UML), a sequential figure is a sort of activity plan that depicts how presided with each other and from where order. It's a Message Sequence Chart construct. Event diagrams, context diagrams, and scheduling diagrams are all terms used to describesequence diagrams.

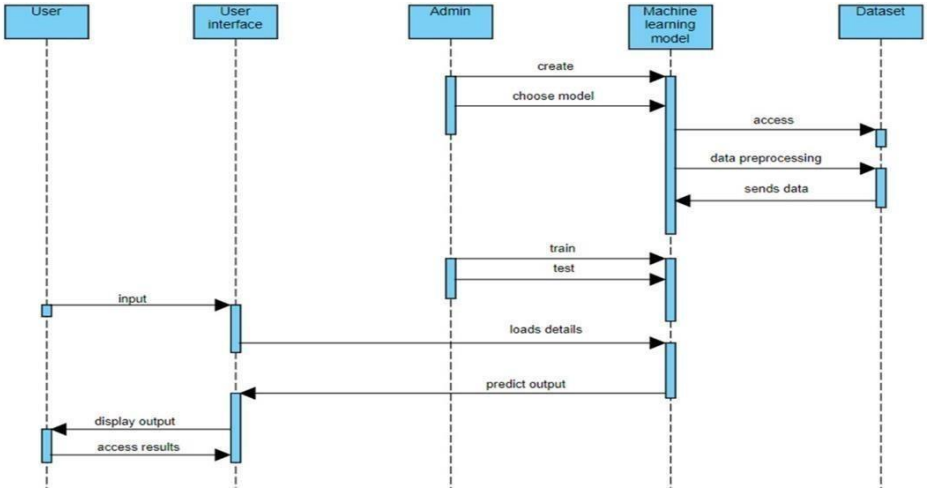


Figure 4.3.3 Sequence Diagram

4.3.4 Activity Diagram

Motion charts illustrate step-by-step physical activities that aid in decision-making, emphasis, and simultaneity. Action graphs can be used in the Unified Modeling Language to represent the technology and business bit-by-bit working practices of portions in a framework. A movement graph depicts the development of control.

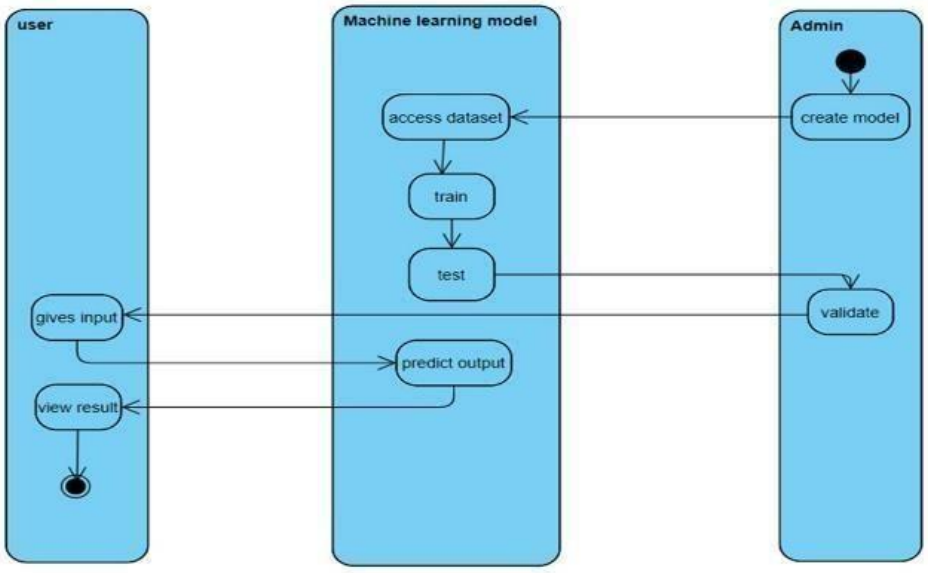


Figure 4.3.4 Activity Diagram

5. IMPLEMENTATION AND RESULTS

The implementation stage of any project is a true display of the defining moments that make a project a success or a failure. The implementation stage is defined as the system or system modifications being installed and made operational in a production environment. The phase is initiated after the system has been tested and accepted by the user. This phase continues until the system is operating in production in accordance with the defined user requirements.

5.1 Algorithm

Step-1: Importing the Libraries.Step-2: Load the Dataset.

Step-3: Check the Missing values.

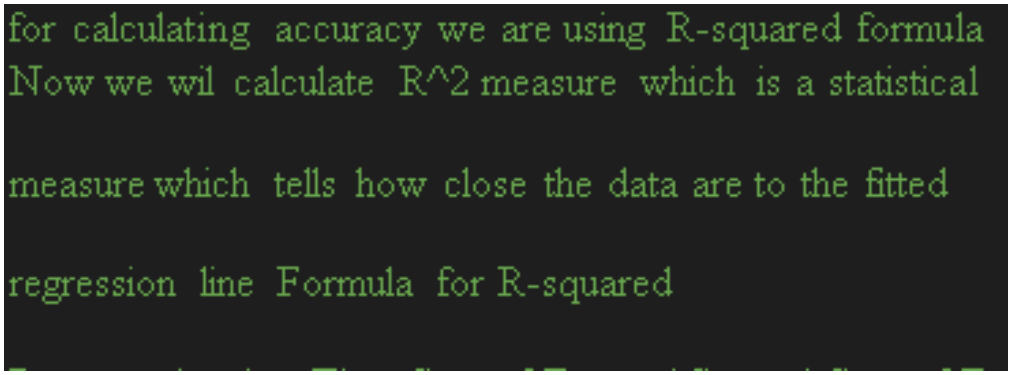
Step-4: Splitting Data into Training and Testing set.Step-5: Linear Regression and Decision Tree.

5.2 Performance Metrics

Several metrics can be used while evaluating how well a model is performing. It is necessary to understand how each metric measures to select the evaluation metric to better assess the model. This thesis main objective was to compare the performance of Machine Learning techniques by evaluating all of these performance metrics such as Accuracy score, Mean Absolute Error, and Max error.

Accuracy score

Accuracy is known as the ratio of a several correct predictions (both true positives andtrue negatives) to the total number of data points.



By using regressor we can calculate accuracy directlyAccuracy=regressor.score(X_test,y_test)

5.3 Sample Code

Steps to be followed

1. Importing the Libraries

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
#pyplot is mainly intended for interactive plots and simple cases
of programmatic plot generation
%matplotlib inline
#%matplotlib inline to enable the inline plotting
import seaborn as snc
```

Pandas

Pandas is an open source, BSD-authorized library giving superior execution, simple to-utilize information designs and information examination instruments for the Python programming language. Pandas is a Number FOCUS supported project. This will assistwith guaranteeing the progress of improvement of pandas as a top notch open-source venture, and makes it conceivable to give to the undertaking.

NumPy

NumPy is the central bundle for logical figuring with Python. It contains another thing suchas: A powerful N-dimensional array object.

Matplotlib

Matplotlib is a Python 2D plotting library which produces distribution quality figures in an assortment of printed copy designs and intelligent conditions across stages. Matplotlib can be used in Python scripts, the Python and IPython shells, the Jupyter journal, web application servers, and four graphical UI tool stash.

Seaborn

In python, seaborn is a Python data visualization library based on matplotlib. It provides a high-level interface for drawing attractive and informative statistical graphics.

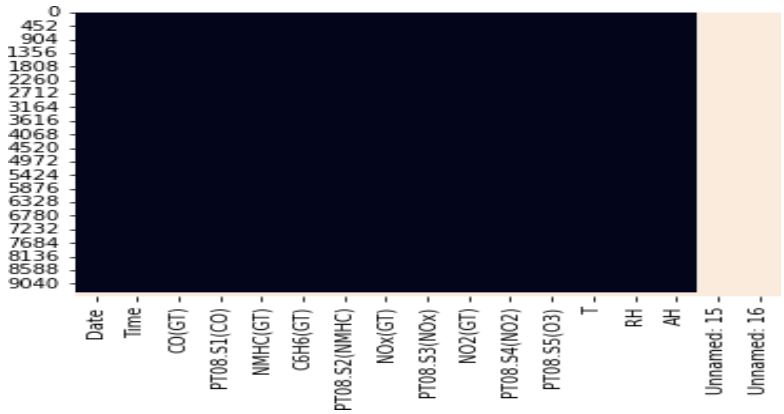
Load the Dataset

```
3. air_data =
   pd.read_csv("https://raw.githubusercontent.com/Kiranreddy
   -2502/Air-Pollution/main/AirQuality.csv", sep=";", decimal=",")
```

As shown in Figure, using Pandas we import our data-set and the file I used here is .csv file. However, to access and to use quickly we use CSV files because of their light weights.

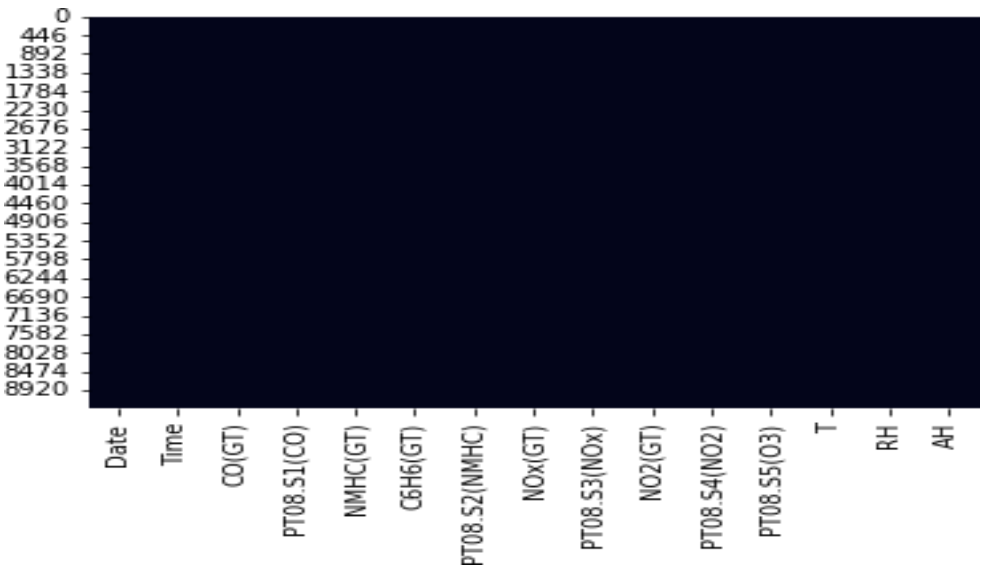
1. Check if there are any null values in the dataset

```
graph = snc.heatmap(air_data.isnull(), cbar=False)
#Plot rectangular data as a color-encoded matrix.
# cbar: bool, optional
# Whether to draw a colorbar.
```



2. Dropping null values

```
air_data.dropna(axis=0, how='all', inplace=True)
# the above line drops all the rows which contain
# complete NULL values and the changes are made in the DataFrame its
elf
air_data.dropna(axis=1, inplace=True)
# the above line drops all the cols which contain
```



3. Converting all datatypes to float, by changing attribute data types which are not float.

```
date=pd.Series(Date)
air_data['Date']=pd.to_numeric(date)
```

```
air_data['Date']=air_data['Date'].astype(float)
air_data.dtypes
# as you can see the dtype of date is converted to float64
```

```
time=pd.Series(Time)
air_data['Time']=pd.to_numeric(time)
```

```
time=pd.Series(Time)
air_data['Time']=pd.to_numeric(time)
```

```
Date             float64
Time             float64
CO(GT)           float64
PT08.S1(CO)      float64
NMHC(GT)         float64
C6H6(GT)         float64
PT08.S2(NMHC)    float64
NOx(GT)          float64
PT08.S3(NOx)     float64
NO2(GT)          float64
PT08.S4(NO2)     float64
PT08.S5(O3)      float64
T                float64
RH               float64
AH               float64
dtype: object
```

4. Extracting features and dropping one the feature and it is the target feature

```
features = air_data
# extracting the features
features=features.drop('C6H6(GT)',axis=1)
# we are dropping this feature because we predict this
target=air_data['C6H6(GT)']
# extracting the target
```

```
[31] print(target)

0        11.9
1         9.4
2         9.0
3         9.2
4         6.5
        ...
9352     13.5
9353     11.4
9354     12.4
9355      9.5
9356     11.9
Name: C6H6(GT), Length: 9357, dtype: float64
```

5. Dropping features which are not necessary

```
# So now we will drop all the features whic are not necessary
features=features.drop('Date',axis=1)
features=features.drop('Time',axis=1)
features=features.drop('T',axis=1)
features=features.drop('RH',axis=1)
features=features.drop('AH',axis=1)
features=features.drop('NMHC(GT)',axis=1)
features.head()
```

6. Splitting dataset into training and testing dataset

```
from sklearn.model_selection import train_test_split

X_train, X_test, y_train, y_test = train_test_split(features, target, test_size=0.3)
# splitting the data into training and testing
```

7. Applying Linear Regression to Predict target feature

```
# Linear Regression

from sklearn.linear_model import LinearRegression

regressor = LinearRegression(normalize=True)
regressor.fit(X_train, y_train)
# applying linear regression

X_test.shape

(2808, 8)

print("Predicted values:", regressor.predict(X_test))
y_pred = regressor.predict(X_test)
y_pred.shape

Predicted values: [12.39143139 16.80788949 24.16845413 ... 10.32024209 15.6292919
  4.19177914]
(2808,)
```

8. Applying Decision tree to predict target feature

```
# Decision Tree Regression

from sklearn.tree import DecisionTreeRegressor

regressor1 = DecisionTreeRegressor()
regressor1.fit(X_train, y_train)

DecisionTreeRegressor()

print("Predicted values:", regressor1.predict(X_test))
y_pred1 = regressor.predict(X_test)
y_pred1.shape

Predicted values: [11.1 16.4 25.  ...  9.6 15.3  5. ]
(2808,)
```

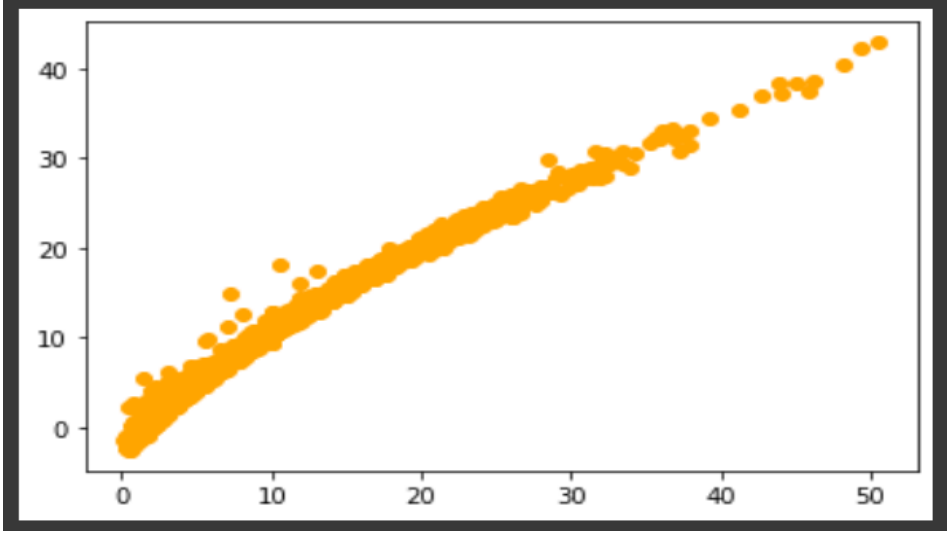
5.4 OUTPUT SCREEN

```
print("R-squared score for liner regression: ", regressor.score(X_test, y_test))

R-squared score for liner regression:  0.9742208927277514
```

```
print("R-squared score for Decision Tree Regression: ", regressor1.score(X_test, y_test))

R-squared score for Decision Tree Regression:  0.9999084761275587
```

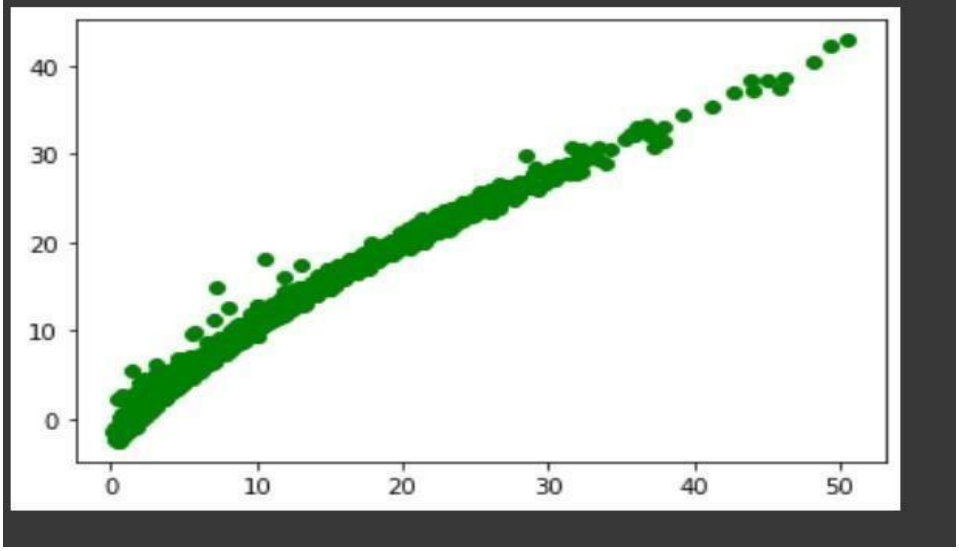
5.4.1 LINEAR REGRESSION

```
plt.scatter(y_test, y_pred, color='orange')
plt.show()
```



5.4.2 DECISION TREE

```
plt.scatter(y_test, y_pred1, color='green')
plt.show()
```



6. TESTING AND RESULTS

6.1. Testing and Test Cases

Programming testing is a basic component of programming quality confirmation and addresses a definitive survey of detail, plan and coding. As a matter of fact, testing is the one stage in the programming systemthat could be seen as damaging as opposed to valuable. A system for programming testing coordinates programming experiment plan strategies into a very much arranged series of steps that outcome in the fruitful development of programming. Testing is the arrangement of exercises that can be arranged aheadof time and led deliberately. The hidden inspiration of program testing is to avow programming quality with techniques that can financially and really apply to both key to both enormous and limited scope frameworks.

Programming framework meets its prerequisites and client assumptions and does not flop in an unsatisfactory way. There are different sorts of test. Each test type tends to a particular testing necessity.

Sr. No	Test case	Input	Expected Output	Actual Output	Remarks
1	Import all the necessary packages and Run them together	In Jupyter command Prompt,using The pip Command toinstall packages.	Successfully Installed	The command has Syntax errors and a re incorrect	The proper Syntax for Importing is Checked and resolved.
2	Loading the Data set into Jupyter notebook	Using the read_csv method, the Dataset file name is passed as an attribute.	Successfully Loads the dataset Without any errors.	Gives a warning and thesystem May slow Down other applications.	Specify the attribute low_memory to be false while Reading the dataset.
3.	Model training	The model is trained with the % of the dataset.	It should give outcomes without Raising any errors.	Not split Properly into a Training and testing. Henc the Accuracy varies.	The dataset is Been split into for better results.
4.	Preprocessing of dataset is not accurate.	Every series is checked individually again for it null/NaN values using isnull().sum() command.	The sum Count lies Within 50% of the data.	The data field's seems to have null value count>50%.	The use of proper replace Commands by declaring correct objects helps in Overcoming the errors.

7. CONCLUSION

Since our model is capable of predicting the current data with 95% accuracy it will successfully predict the upcoming air quality index of any particular data within a given region. With this modelwe can forecast the AQI and alert the respected region of the country also it a progressive learningmodel it is capable of tracing back to the particular location needed attention provided the time series data of every possible region needed attention. The air quality information utilized in this paper originates from the china air quality checking and investigation stage, and incorporates the normal every day fine particulate issue (PM2.5), inhalable particulate issue (PM10), ozone (O3), CO, SO2, NO2 fixation and air quality record(AQI).The essential perspectives that should be viewed as with regards to gauging of the poison focus are its different sources alongside the components that impact its fixation.

8. FUTURE SCOPE

The scope of this project is to investigate a dataset of air pollutants records for India meteorological sector using machine learning technique. To identifying air quality is more difficult. We try to reduce this risk factor behind predicting from Air Quality Index (AQI) of India to safe human so as to save lots of meteorological efforts and assets and to predict whether assigning the air qualityis bad or good.

9. BIBLIOGRAPHY

- [1].Effective and Efficient Photo -Based PM2.5 Concentration Estimation, Guanghui Yue , Ke Gu , and Junfei Qiao, Member, IEEE, 2020.

- [2]. A Machine Learning Approach to Predict Air Quality in California Mauro Castelli Fabiana Martins Clemente,1 Alesˇ Popovicˇ, Received 25 January 2020;Accepted 23 June 2020; Published 4 August 2020.

- [3].Temesegan Walelign Ayele, and Rutvik Mehta, "Air pollution monitoring and prediction IoT," In Second International Conference on Inventive Communication anComputational Technologies (ICICCT), pp. 1741-1745. IEEE, 2018

- [4].J. K. S and D. S. David, "A Novel Based 3D Facial Expression Detection Using Recurrent Neural Network," 2020 International Conference on System, Computation,Automation and Networking (ICSCAN), Pondicherry, India, 2020, pp.1-6, doi: 10.1109/ICSCAN49426.2020.9262287.

- [5].Stalin David D, Saravanan D, "Enhanced Glaucoma Detection Using Ensemble based CNN and Spatially Based Ellipse Fitting Curve Model", Solid State Technology, Volume 63, Issue 6, PP.3581-3598