

**PREDICTION OF AIR POLLUTANTS USING SUPERVISED
MACHINE LEARNING**

Submitted in partial fulfillment of the requirements for the award of
Bachelor of Engineering degree in Computer Science and Engineering

by

YARRAGUNTA SRIRAM KRISHNA

(Reg. No. 37110866)

MOHAMMED ABDUL NABI

(Reg. No. 37110881)



**DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING
SCHOOL OF COMPUTING**

SATHYABAMA

**INSTITUTE OF SCIENCE AND TECHNOLOGY
(DEEMED TO BE UNIVERSITY)**

Accredited with Grade "A" by NAAC

**JEPPIAAR NAGAR, RAJIV GANDHI SALAI,
CHENNAI – 600 119**

MARCH - 2021



SATHYABAMA

INSTITUTE OF SCIENCE AND TECHNOLOGY
(DEEMED TO BE UNIVERSITY)

Accredited with "A" grade by NAAC

Jeppiaar Nagar, Rajiv Gandhi Salai, Chennai – 600 119

www.sathyabama.ac.in



DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

BONAFIDE CERTIFICATE

This is to certify that this project report is the bonafide work of **YARRAGUNTA SRIRAM KRISHNA (Reg. No.37110866)** and **MOHAMMED ABDUL NABI (Reg. No. 37110881)** who carried out the project entitled "**PREDICTION OF AIR POLLUTANTS USING SUPERVISED MACHINE LEARNING**" under my supervision from **August 2020 to March 2021**.

Internal Guide

Dr. P. Jeyanthi, M.E., Ph.d.,

Head of the Department

Submitted for Viva voce Examination held on _____

Internal Examiner

External Examiner

DECLARATION

I **YARRAGUNTA SRIRAMKRISHNA** hereby declare that the Project Report entitled “**PREDICTION OF AIR POLLUTANTS USING SUPERVISED MACHINE LEARNING**” is done by me under the guidance of **Dr. P. Jeyanthi, M.E., Ph.d.**, Department of Computer Science and Engineering at Sathyabama Institute of Science and Technology is submitted in partial fulfillment of the requirements for the award of Bachelor of Engineering degree in Computer Science and Engineering.

DATE:

PLACE: CHENNAI

SIGNATURE OF THE CANDIDATE

ACKNOWLEDGEMENT

I am pleased to acknowledge my sincere thanks to **Board of Management** of **SATHYABAMA** for their kind encouragement in doing this project and for completing it successfully. I am grateful to them.

I convey my thanks to **Dr. T. Sasikala, M.E., Ph.D., Dean**, School of Computing, **Dr. S. Vigneswari, M.E., Ph.D.**, and **Dr. L. Lakshmanan, M.E., Ph.D., Heads of the Department** of Computer Science and Engineering for providing me necessary support and details at the right time during the progressive reviews.

I would like to express my sincere and deep sense of gratitude to my Project Guide. **Dr. P. Jeyanthi, M.E., Ph.d.**, for her valuable guidance, suggestions and constant encouragement paved way for the successful completion of my project work.

I wish to express my thanks to all Teaching and Non-teaching staff members of the **Department of Computer Science and Engineering** who were helpful in many ways for the completion of the project.

ABSTRACT

The proposed system depicts various strategies utilized for forecast of Air Quality Index (AQI) utilizing supervised machine learning procedures. The system examines machine learning algorithm for air quality index by computing algorithm accuracy which will bring about the best precision. Moreover, the system exhibits different machine learning accuracy figures from the given dataset with assessment order report which recognizes the perplexity lattice. The outcome shows the adequacy of machine learning suggested calculation method that can be contrasted and best exactness with accuracy, Recall and F1 Score. The air pollution database contains data for each state of India. Six supervised machine learning algorithms, logistic regression, decision tree, support vector machine, random forest tree, Naïve Bayes theorem and K-nearest neighbor are compared and evaluated.

TABLE OF CONTENTS

CHAPTER No	TITLE	PAGE No.
	ABSTRACT	V
	LIST OF TABLES	IX
	LIST OF FIGURES	X
1	INTRODUCTION	1
	1.1 Architecture System	2
2	PROJECT CHARTER	4
	2.1 Domain Overview	4
	2.2 Objective	4
	2.3 Problem Description	5
	2.4 Existing System	5
	2.4.1 Drawbacks	5
3	MODULES DESCRIPTION	6
	3.1 Exploration data analysis of visualization	6
	3.2 Variable Identification Process	7
	3.2.1 Data Validation	7
	3.2.2 Data Pre-processing	8
	3.3 Decision Tree Vs Logistic Regression Algorithm	8
	3.3.1 Logistic Regression	8
	3.3.2 Decision Tree	9
	3.4 Support Vector Machine Vs Random Forests Algorithm	10
	3.4.1 Support Vector Machine	10
	3.4.2 Random Forests	10
	3.5 K-Nearest Neighbor Vs Naïve Bayes Algorithm	11
	3.5.1 K-Nearest Neighbor	11

	3.5.2 Naïve Bayes	11
4	LITERATURE SURVEY	13
	4.1 Photo-Based PM2.5 Concentration Estimation	13
	4.2 Effects of Air Pollution on Hospital Admissions	13
	4.3 Data Cleaning for Data Quality	13
	4.4 Survey on Big Data Analytics	14
	4.5 Prediction Scheme Based on CNN and LSTM for Urban PM2.5 Concentration	14
	4.6 Spatial Characteristics Of Air Pollution	14
	4.7 Big Data Platform for Air Quality analysis and Prediction	15
5	SYSTEM ANALYSIS	16
	5.1 Data Collection and Pre-processing	16
	5.2 Data Analysis and Visualization	17
	5.3 Training and Modeling the dataset	18
6	SYSTEM DESIGN	19
	6.1 Logistic Regression	19
	6.2 Naïve Bayes	20
	6.3 Random Forests	21
	6.4 Support Vector Machines	22
	6.5 K-Nearest Neighbor	23
	6.6 Decision tree	24
7	CODING, TESTING	25
	7.1 Coding	25
	7.1.1 Environment Requirements	25
	7.2 Testing	34

8	CONCLUSION	35
9	FUTURE ENHANCEMENT	36
	A. Machine Learning	37
	A.1 Supervised Machine Learning	37
	A.1.1 Logistic Regression	37
	A.1.2 Naïve Bayes	37
	A.1.3 Random Forests	37
	A.1.4 Support Vector Machine	37
	A.1.5 K-Nearest Neighbors	38
	A.1.6 Decision Tree	38
	B. Parameters	39
	B.1 Precision and Recall	39
	B.2 Air Quality Index	39
	B.3 Confusion Matrix	39
	B.4 F1 Score	39
	B.5 PM2.5	40
	B.6 Learning Curve	40

LIST OF TABLES

TABLE No.	TABLE NAME	PAGE No.
5.1	Data Set Attributes	16
7.1	Comparison Of Accuracy result of each algorithm	34

LIST OF FIGURES

FIGURE No.	FIGURE NAME	PAGE No.
1.1	AQI Category, Pollutants and health breakpoints	2
1.2	Architecture System of the proposed system	3
2.1	Process of Machine Learning	4
5.1	Demonstration of Pollutants Vs Avg. Pollution	17
5.2	Demonstration of Concentration Pollutants	17
6.1	Classification report of Logistic Regression	19
6.2	Logistic Regression Accuracy Graph	19
6.3	Classification report of Naïve Bayes	20
6.4	Naïve Bayes Accuracy Graph	20
6.7	Classification report of Support Vector Machines	22
6.8	SVM Accuracy Graph	22
6.9	Classification report of K-Nearest Neighbor	23
6.10	KNN Accuracy Graph	23
6.11	Classification report of Decision Tree	24
6.12	Decision Tree Accuracy Graph	24
7.1	GUI Screenshot	34

CHAPTER 1

INTRODUCTION

Artificial intelligence and machine learning in recent years has been one of the phenomenal advancement of technology. Instead of just writing commands as standard works, the philosophy of artificial intelligence, in which the system makes its own choices gradually impacts all aspects of society. From the early stage of startups to big platform vendors, the machine learning in its component has become the main focus areas of all companies. Machine learning is an area where an artificial intelligence system gathers data from sensors and learns to behave in an environment. The ability to adapt to machine learning (ml) algorithms was one of the reasons that machine learning is used to predict the air quality index. Six machine learning algorithms are utilized in the system that are as follows, decision tree, support vector machine (svm), naive Bayes classifier, logistic regression, random forests, and k-nearest neighbors are compared.

A lot of researchers introduce some of the algorithms that are used in the proposed system but none of them compares their success as a single study under similar conditions and on the same data for all six.

In India, air pollution is being seen as a widespread problem. Every day air quality quantified attributes are too many percent higher than that of the highest possible values assumed fit for public medical care. The situation is severe in bigger urban areas like Delhi where AQI attained a peak of all time of 999 AQI. The central government as well as the state administration performed a few of the air pollution reduction processes.

The first stage requires the prediction of the air quality index so that the situation can be improved. Created six different classifiers in the project which are based on different algorithms. The analysis of dataset by supervised machine learning technique (SMLT) to capture several information's like, variable identification, uni-variate analysis, bi-variate and multi-variate analysis, missing value treatments and analyze the data.

In India, as in numerous different nations, the record is revolved around six significant contaminations particulate issue under 10 micrometers in measurement (PM10),

particulate issue under 2.5 micrometers in breadth (PM2.5), carbon monoxide (CO), Ammonia (NH₃), nitrogen dioxide (NO₂), and ozone (O₃). By then, a checking station ought to have the option to furnish one with the grouping of a specific poison and its normal over some stretch of time for CO and O₃, the normal is assumed control more than eight hours, while for the other three, it is a normal of 24. The unit of estimation per cubic meter is microgram (or milligram, on account of CO). The following table presents the indicators and the air quality indexes that relate.

AQI (Range)	PM ₁₀ (24hr)	PM _{2.5} (24hr)	NO ₂ (24hr)	CO (8hr)	SO ₂ (24hr)	NH ₃ (24hr)	O ₃ (8hr)
Good (0-50)	0-50	0-30	0-40	0-1.0	0-40	0-200	0-50
Satisfactory (51-100)	51-100	31-60	41-80	1.1-2.0	41-80	201-400	51-100
Moderately (101-200)	101-250	61-90	81-180	2.1-10	81-380	401-800	101-168
Poor (201-300)	251-350	91-120	181-280	10-17	381-800	801-1200	169-208
Very poor (301-400)	351-430	121-250	281-400	17-34	801-1600	1200-1800	209-248
Severe (401-500)	431-500	250+	400+	34+	1600+	1800+	248+

Figure 1.1: AQI Category, Pollutants and health breakpoints

As seen from above each band is represented by the color code that shows how significantly people can grasp it easily. The index categorized air quality simply as good, satisfactory, moderately polluted, poor, very poor and severe.

1.1 Architecture System

Architecture diagram given in figure 1.2 gives the glimpse of the proposed prediction model and its working. The first step is to provide dataset to the prediction model by the user. The dataset supplied to machine learning model is used to train the model. Every new data detail filled at the time of application form acts as a test data set.

The obtained data set and the previous data set are kept in Datawarehouse. Pre-processing and validation are the next stage. Pre-processing corresponds to the data transformations that are performed before the algorithm is processed. Data Pre-processing is a method for transforming raw data into a clean data package. In other terms, once the data is obtained from different outlets, it is raw and can not be evaluated. The validation is the import of the dataset enabled library bundles.

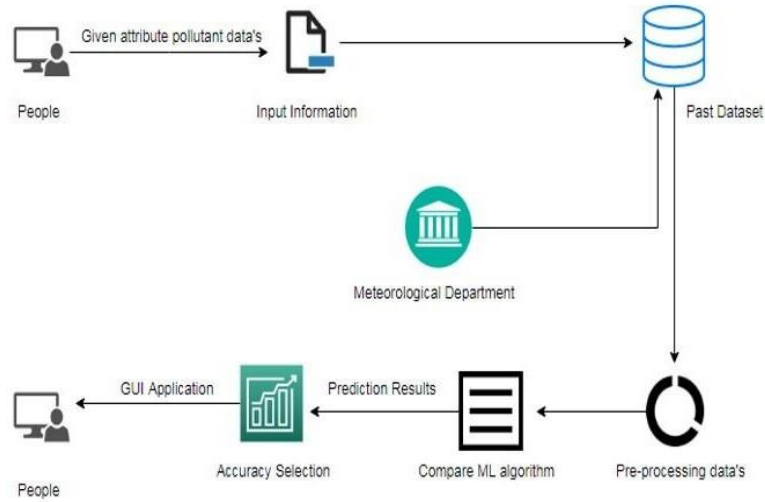


Figure 1.2: Architecture System of the proposed system

It analyses the variable identification by data shape, data type and evaluating the missing values and duplicate values. A validation dataset is a sample of data held back from training the model that is used to give an estimate of model skill while tuning model's and procedures that you can use to make the best use of validation and test datasets when evaluating the models. After pre-processing the data, several machine learning algorithms such as decision tree, Random forest, Naïve Bayes, Logistic Regression, SVM ,etc are used to train the dataset and predict the AQI. The machine learning algorithm which gives the best accuracy is selected to builds the prediction model.

It is necessary to regularly compare the output of several distinctly different learning algorithms and to build a test harness to easily compare multiple simultaneous learning algorithms in Python with scikit-learn. Each model will have different performance characteristics. Using resampling methods like cross validation, you can get an estimate for how accurate each model may be on unseen data. It needs to be able to use these estimates to choose one or two best models from the suite of models that you have created. Finally an interactive GUI (graphical user interface) is developed using Tkinter in python library.

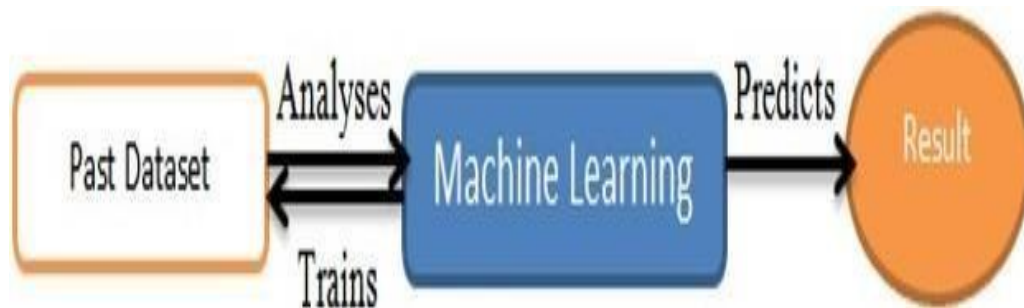
CHAPTER 2

PROJECT CHARTER

2.1 Domain Overview

Machine learning is an intellect category that allows machines the opportunity to know without being programmed directly. Machine learning relies on computer programs which may alter as new data become revealed.

Figure 2.1: Process of Machine Learning



2.2 Objective

The goal is to develop a machine learning model for real-time air quality forecasting, to potentially replace the updatable supervised machine learning classification models by predicting results in the form of best accuracy by comparing supervised algorithm.

2.3 Problem Description

The management and protection of air quality in many industrial and urban areas is actually a critical tasks, owing to specific styles of emissions induced by the usage of transport electric power, etc. the accumulation of toxic gasses presents a significant danger to the quality of life in smart cities. As air pollution rises,needed a effective air quality control models to gather information on air pollutant concentrations and to measure air pollution in all areas.

2.4 Existing System

The concentration of PM2.5 should be calculated via a photographic process model. Observation reveals that the exposure chart displays slightly different manifestations of large and small amounts of PM2.5 and is prone to air quality. For contrast photograph, the structures are destroyed and so the bulk of pixels are usually 0 below a large concentration of PM2.5. The Weibull distribution is used to match the saturation map and to measure the value of the color detail. Finally, a combination of the aforesaid characteristics and a nonlinear mapping technique will estimate the PM2.5 concentration of an image. In contrast with the current state-of - the-art techniques, the empirical and visualized effects of actual data collected confirm the reliability and superiority of the proposed process. Air pollution has become a global problem and automated air quality assessment will provide both human and industrial efficiency with a constructive roadmap.

2.4.1 Drawbacks

- Photo graphic method is not sufficient to calculate PM2.5 and it taken only one pollutants of concentration.
- The work is given based on only one PM2.5 and to collect more monitoring data from other cities to verify the generalization of the work and more factors, e.g., geomorphic conditions.

CHAPTER 3

MODULES DESCRIPTION

3.1 Exploration data analysis of visualization

Data analysis is an essential capability in computational analytics and machine learning. In addition, statistics rely on detailed explanations and data estimates. Data visualization is an effective toolkit to achieve a contextual understanding. It can be helpful as you research and study a dataset which can help to detect trends, missing results, outliers and several other items. Data visualizations are able to be utilized with a little technical awareness to communicate and explain core relations in more graphic and involved plots and maps than cumulative or concrete measures.

Only data becomes useless before it becomes readily visible, such as graphs and charts. The ability to easily imagine examples of data and others is essential for both applied analytics and machine learning. You can consider the various styles of pictures you need to recognize while you are visualizing Python data and how you can use them to fully visualize your personal data.

- How to model time series details using bar charts of lines and categorical numbers.
- How to sum up distributions of data using histograms and box plots.
- How to sum up variables 'connection to scatter plot.

Most machine learning algorithms are prone to attribute values in input data and delivery. Input data outliers will deceive and misrepresent the machine learning algorithms training cycle, which contributes to longer training times, less reliable models and eventually weaker performance.

Well before predictive models of training data are created, outliers will lead to misleading representations and misleading perceptions of gathered information. In descriptive statistics such as mean and standard deviation and in tracks such as histograms and

scatterplots, Outliers will distort the overview distribution of attribute values by consolidating the body of evidence. Finally, outliers may be indicators of data cases, such as fraud detection abnormalities and computer protection, which are important to the issue.

It can not adapt the model to the training data and it can not guarantee whether the model would function correctly on the real data. You need to be sure the model has the correct data characteristics so it will not have too much noise. Cross-validation is a method in which a model is educated using a data-set subset and then tested with an adjacent dataset sub-set.

3.2 Variable Identification Process

Machine learning validation techniques are employed to obtain the machine learning (ML) error rate model, which can be considered as similar to the true data set error rate. You may not need testing strategies if the data volume is sufficiently high to represent the population. To locate the missing value, repeat the value and the data form definition, if the attribute is float or integer. The data sample was used to provide an objective model fit evaluation on the testing dataset when adjusting hyper parameters of the model. The appraisal gets more skewed as expertise is integrated into the model setup on the validation dataset. To test a given model, the validation collection is used, but this is for regular evaluation. As machine learning developers, this data is used to fine-tune the hyperparameter of the model.

3.2.1 Data Validation

Data processing, data analysis, and the process of addressing data material, consistency, and configuration will add up to a time-consuming list of tasks. It helps to understand the data and its properties during the data recognition process; this information can help you select which algorithm to use to construct the model. By regression algorithm, for example, data from time series can be analyzed; classification algorithms can also be utilized to evaluate discrete data.

3.2.2 Data Pre-processing

Pre-processing relates to the preparation of the data until it is passed to the algorithm. Preprocessing data is a method for transforming raw data into a clean collection of data. With other words, if the data were obtained with raw format from various outlets, this is not appropriate for study. The machine learning cycle will prove to be right in order to obtain improved outcomes from the implemented model. For eg, Random Forest algorithm does not accept null values. Certain Machine Learning models require details in a defined format. Therefore, null values from the initial raw data collection must be handled to operate random forest algorithms. And it is also critical that data sets are so structured that more than one machine learning and deep learning algorithm in a given dataset is implemented

3.3 Decision vs Logistic Algorithm

3.3.1 Logistic regression

Analysis of logistic regression seems to be the most common approach of regression which could be used in binary dependent parameter modeling. Logistic regression is a computational methodology which often represents the associations between the independent variables $x_1 x_2 \dots x_n$ and y that is the two alternative categories of discrete dependent variable coded in 0 or 1. Independent variables may be permanent, discrete, binary or combined.

That is, $P(Y=1)$ as a function of X is predicted by the model of logistic regression. Logistic regression Assumptions:

- The dependent variable must be binary for binary logistic regression.
 - The desired result should be the level 1 factor for the binary regression of the dependent variable. Only the meaningful variables should be included.

- The separate variables should be mutually independent. In other words, the model ought to have little.
- The independent variables have a linear relationship with the log odds.
- Logistic regression requires sample sizes quite large.

3.3.2 Decision tree

This algorithm is one of the most powerful. The algorithm of the decision array falls into the supervised learning algorithm category. It works for both continuous and categorical variables of output.

Assumptions of Decision tree:

- At the beginning, considering the whole training set as the root.
- Attributes are considered to be categorical for finding information gain, given the attributes are continuous.
- Recursive distributions are made on the basis of attribute values.
- Using statistical methods for root or internal node ordering attributes.

Decision Trees is a data gathering categorization and analysis technique. Decision trees constitute the fundamental recursive basis for the sequence process of classification, in that one of the disjoint class decision-making structures contains nodes and leaves, is allocated a case identified with a collection of attributes. Each tree node includes testing a specific attribute and every tree leaf denotes a class. The test usually compares the value with a constant of an attribute. Leaf nodes are usually categorized in all cases whereas for classified set is entered or where probabilities are spread over any possible classification.

The process is continued until the termination is completed. It is designed in a recursive dividing-and-conquer top-down fashion. All characteristics should be categorical. They would otherwise be discretized in advance. Top of the tree characteristics have a greater effect on the classification and the knowledge gain principle is used. Too many branches can easily be over-fitted and can reflect disturbances caused by noise or bolts.

3.4 Support Vector Machine vs Random Forest Algorithm

3.4.1 Support Vector Machines

Support Vector Machine has been a collection of similar supervised learning methods for use in categorization and regression. This belongs to a family of generalized linear classifications. One unique aspect of SVM is that the empirical classification error is minimized and geometrical margin maximized at the same time.

- How to detangle the various names used for referring vector support machines.
- The representation that SVM will use when the model is actually stored on the disk.
- How to use a trained representation of the SVM model to make accurate predictions for new data.
- What to learn from the training data on an SVM platform.
- How would the data be better optimized for SVM
- Where can you look for additional SVM information

3.4.2 Random forest

Random forests were a hybrid of trees forecasting flaws in forest general statement since each tree operates on the theory of the random variable within each tree. The error of generalizing forests depends on the strength of each of the forest's trees and the relationship between them on the individual since the percentage of forest trees is vital. By randomly selecting the characteristics for splitting every node, error levels are similar to adaboost, but are richer in noise.

In carrying out the random forest algorithm the following are basic steps:

- From the given dataset, choose N Random Data.
- Create a decision tree on the basis of those N records.
- Pick the number of trees you want and repeat steps 1 and 2 in the algorithm.

- In the event of a problem with regression, for a new record, the value for Y for each tree in the forest is expected. The final value can be calculated by taking the sum of all the forested trees 'values.

3.5 K-Nearest Neighbor vs Naive Bayes Algorithm

3.5.1 K-Nearest Neighbor

K-Nearest Neighbor is a supervised learning algorithms which hold all instances in n-dimensional space corresponding to training data points. When an unknown discrete data is obtained, it analyzes the nearest k number of stored instances (near neighbors) and finally returns the most previewed class and returns the average of k similar neighbors for real-value results. In the nearest distance algorithm, each of the k neighbors assesses the contribution by distance by the subsequent question and gives more weight to the nearest neighbors.

KNN is typically resilient to noisy data since the closest neighbours are combined. The algorithm of K-nearest is a classification algorithm, which is supervised: a group of defined points is needed and is used to learn how to label other points. This searches for a specific point which has such neighbours 'votes in view of the points that are labelled nearest to the new point (the "k" is the amount of neighbourhoods that it checks). It aims at the neighbors' voting points, and the mark that is the most significant one of the neighborhood. Predict the testing collection for the whole training package. By filtering through the entire collection to find the k "closest" instances, KNN prediction for a new case. Close-up is evaluated across all apps of proximity (Euclidean).

3.5.2 Naive Bayes algorithm

Naive Bayes is a statistical classification technique based on Bayes Theorem. It is one of the simplest supervised learning algorithms. Naive Bayes classifier is the fast, accurate and reliable algorithm. Naive Bayes classifiers have high accuracy and speed on large datasets. Naive Bayes classifier assumes that the effect of a particular feature in a class is independent of other features. For example, a loan applicant is desirable

or not depending on his/her income, previous loan and transaction history, age, and location. Even if these features are interdependent, these features are still considered independently. This assumption simplifies computation, and that's why it is considered as naive. This assumption is called class conditional independence. In calculating the probability of a data instance belonging to the class for each attribute for a specified class value. In order to measure a estimate of the likelihood of the instance of increasing category and pick the maximum likelihood for the class rating.

Naive Bayes is a Bayes Theorem-based system of statistical classification. This is one of the basic guided algorithms for learning. A quick, accurate and reliable algorithm is the Naive Bayes classifier. High precision and efficient on broad datasets are the Naive Bayes classifier. The classification of Naive Bayes means that the influence of a certain function in a class is unconditional to others. For example, a bond applicant is suitable or not depending on his / her salary, background experience, age, position and loan experience. Although these characteristics are interdependent, they are still independent. This presumption renders equations easier and is thus called naive. This presumption is regarded as class equality.

CHAPTER 4

LITREATURE SURVEY

4.1 Photo–Based PM2.5 Concentration Estimation

Yue et al. (2019) Estimates the particulate matter content using a photographic process dependent visual methodology. Finally, a confluence of both features described above with a non-linear mapping procedure may project the PM2.5 concentration of the image. Photo graphic method is not sufficient to calculate PM2.5 and it taken only one pollutants of concentration. Research is performed on the context of only one PM2.5 and to obtain further testing data from others, for example geomorphic conditions, to check that research is generalized. Thus the regularity of air pollutant data cannot be more accurately determined and the prediction results obtained.

4.2 Effects of Air Pollution on Hospital Admissions

De Leon et al. (1996) Presented the effects of air pollution on daily hospital admissions for respiratory disease in Poisson regression analysis of day by day tallies of hospital confirmations, modifying for impacts of pattern, occasional and other recurrent variables, day of the week, occasions, flu pestilence, temperature, dampness, and auto connection. Variables of pollution were particulates black smoke (BS), nitrogen dioxide (NO₂), Sulphur dioxide (SO₂), and ozone (O₃).

4.3 Data Cleaning for Data Quality

Challa et al. (2016) Presented Data cleaning techniques to achieve quality data. Now every second data is rapidly generated over the internet for a day and thus it has become a huge task to make the right decision. Surrounded by data but hungry for information,

knowledge plays a vital part for decision-makers to generate more income in industry. Data collected from different sources may have dirty information, data cleaning should be performed before the data is loaded into the warehouse in order to obtain data quality.

4.4 Survey on Big Data Analytics

[Acharjya \(2019\)](#) This paper's primary objective is to research the potential impact of big data problems, issues in open research, and different apparatuses related with it. This paper provides important insights on big data and the principles behind its analysis, which is an important factor for this project because of large size of the sample data being used to provide conclusions.

4.5 Prediction Scheme Based on CNN and LSTM for Urban PM2.5 Concentration

[Qin et al. \(2019\)](#) It explains about the urban air pollution focus expectation is managing a flood of huge ecological checking information and complex changes in air toxins. This requires viable expectation strategies to improve forecast exactness and avert genuine contamination occurrences, along these lines upgrading ecological the executive's basic leadership limit. In this paper, another contamination fixation forecast strategy is proposed dependent on huge measures of natural information and profound deep learning procedures. The proposed strategy incorporates huge information utilizing two sorts of deep network systems.

4.6 Spatial Characteristics of Air Pollution

[Li et al. \(2011\)](#) explains that air pollution is a major and pervasive influence in Chengdu, owing to its effect on human health and well-being. The main pollution induced by soil by the use of petroleum products was sulfur dioxide (SO₂), carbon monoxide (CO), nitrogen dioxide (NO₂) and particulate matter (PM₁₀). In this investigation, the spa-

tial circulation fluctuation of the significant air poisons over the principle city region of Chengdu in early 2010 was broke down and evaluated. The outcomes showed that OK strategy was smarter to produce the forecast maps of toxin fixations than IDW technique. High spatial decent variety of NO₂, SO₂ and PM₁₀ focuses existed, particularly in the east-west course. Between the six principle locales in Chengdu, Chengdu District was dirtied all the more truly in 2010.

4.7 Big data platform for air quality analysis and prediction

Chang et al. (2018) enumerate that air quality can be influenced by plenty of causes, such as traffic, plant emissions, climate, waste disposal, etc. A semantic ETL (Extract-Transform Load) system for AQ prediction on the cloud platform is discussed in this paper. Using ontology to refocus PM_{2.5} from different data sources and also to merge this data into the integrated database with the same concept, however, with different naming. Used the cloud-based ETL platform, consisting of device nodes and storage nodes. The computing nodes are used for predicting data mining algorithms and storage modes for the collection, preprocessing and analysis of retrieved data. They make use of restful Web services as the front end API to get analyzed data; used the browser to demonstrate the visualized results to show estimates and forecasts. It shows that the cloud platform big data access framework can be effective in air quality analysis.

CHAPTER 5

SYSTEM ANALYSIS

The raw recorded data contains meteorological information for various cities in India. The idea is to use various machine learning techniques such as Logistic Regression, Support Vector Machines, Random Forests, K-Nearest Neighbors, Naïve Bayes Classifier and Decision Tree for model building and research purposes.

5.1 Data collection and Pre-processing

Data is collected from the Central Pollution Control Board (CPCB). This data set comprises of nine attributes that are Country, State, City, Place, Last updated, min, max, average and air pollutants like PM10, PM2.5, NO2, CO, SO2, NH3, and O3(Ozone). Data is collected every day to anticipate the behavior of these contaminants in upcoming days. Table 2 shows the attributes of the dataset. After data collection, the data is pre-processed. In pre-processing, the data is cleaned by noise removal and the missing values are filled in.

Table 5.1: Data Set Attributes

Variable	Description
Country	Home country (India)
State	Indian states name lists
City	City names for each state
Place	Place names for each city
Lastupdate	Date and time (DD/MM/YYYY HH:MM)
Avg	Average range of pollutants
Max	Maximum range of pollutants
Min	Minimum range of pollutants
Pollutants	Pollutants name

5.2 Data analysis and visualization

Data visualization is an essential specialist knowledge in applied statistics and the statistics of machine learning rely practically on quantitative explanations, and software analyses are valuable instruments for obtaining qualitative understanding. This helps to discover and know about a dataset and help to identify trends in fraudulent data transfers and more. Information visualizations are feasible with low domain knowledge to communicate and display core connections through graphs and maps which are more emotional and consumers than associative or substantive steps.

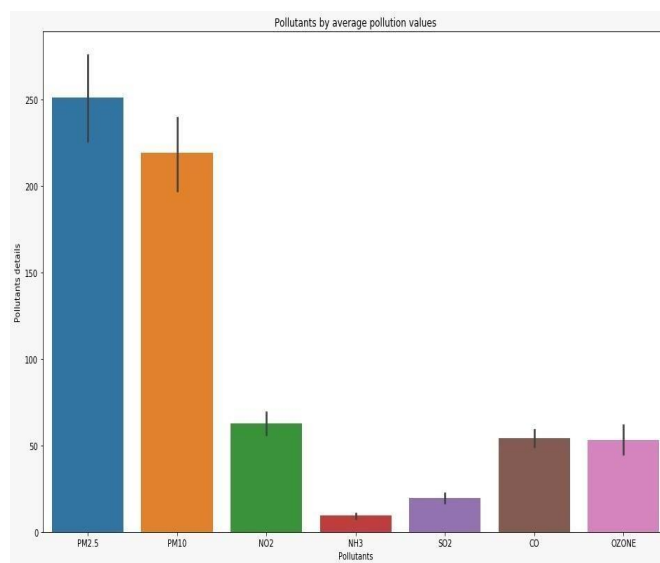


Figure 5.1: Demonstration of Pollutants vs Avg Pollution

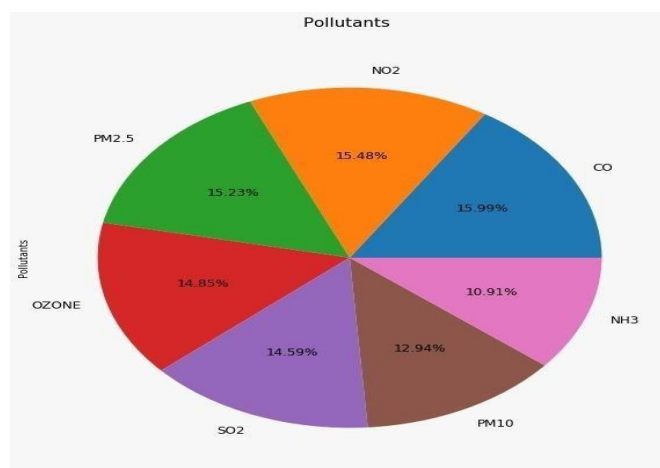


Figure 5.2: Demonstration of Concentration Pollutants

5.3 Training and Moduling dataset

For the estimation of results, the data collection is split into two groups, that is, a training collection, and a test set that typically utilizes 7:3 ratios to distinguish the training sample from the test sample. The data model built using Random Forest, Logistic Rectification, Decision Tree Algorithms, KNN, Support Vector Classifier (SVC) and Naive Bayes is applied to the training set and the Test Set Prediction is conducted on the basis of the precision and the test output.

CHAPTER 6

SYSTEM DESIGN

Six Machine Learning techniques i.e. Logistic Regression, Naive Bayes method, Random Forests algorithm, Support Vector Machine (SVM), K- Nearest Neighbor (KNN) and Decision Tree algorithm, were used for the purpose of building a prediction model.

6.1 Logistic Regression

Analysis of logistic regression seems to be the most common approach of regression which could be used in binary dependent parameter modeling. Logistic regression is a computational methodology which often represents the associations between the independent variables $x_1 x_2 \dots x_n$ and y that is the two alternative categories of discrete dependent variable coded in 0 or 1. The independent variables may be continuous, discrete, binary or a combination of them. After applying Logistic Regression, the prediction accuracy obtained was 97.46.

```
Classification report of Logistic Regression Results:

```

	precision	recall	f1-score	support
0	0.97	0.97	0.97	73
1	0.99	0.99	0.99	175
accuracy			0.98	248
macro avg	0.98	0.98	0.98	248
weighted avg	0.98	0.98	0.98	248

```

Cross validation test results of accuracy:
[0.94117647 1. 1. 1. 1. 1.
 1. 0.94117647 1. 1. 1. 1.
 0.94117647 1. 1. 1. 1. 0.94117647
 0.94117647 1. 1. 1. 1. 0.94117647
 0.94117647 0.88235294 0.94117647 1. 1. 0.82352941
 1. 0.94117647 1. 0.9375 0.9375 0.9375
 0.8125 1. 1. 1. 1. 1.
 1. 1. 1. 1. 1. 0.93333333
 1. 1. ]

Accuracy result of Logistic Regression is: 97.46960784313725

Confusion Matrix result of Logistic Regression is:
[[ 71  2]
 [ 2 173]]

Sensitivity : 0.9726027397260274
Specificity : 0.9885714285714285
```

Figure 6.1: Classification report Logistic Regression

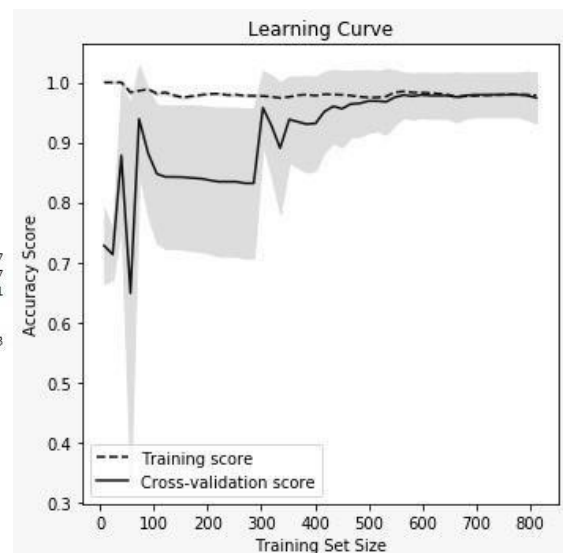


Figure 6.2: Logistic Regression Accuracy graph

6.2 Naïve Bayes

Naïve Bayes is quite a basic learning algorithm that uses Bayes and strongly assumes that the attributes in the groups are conditionally separate. While this presumption of freedom is often broken in reality, the naive Bayes also provides efficient consistency in classification. Combined with its analytical performance and many other attractive attributes, this leads to the use of naive Bayes in practice. After applying Naive Bayes Classifier, the prediction accuracy obtained was 97.38

```
Classification report of Naive Bayes Results:

              precision    recall  f1-score   support

     0       0.94         0.99         0.96         73
     1       0.99         0.97         0.98        175

 accuracy          0.98
 macro avg         0.96         0.98         0.97         248
 weighted avg      0.98         0.98         0.98         248

Cross validation test results of accuracy:
[1. 1. 1. 1. 1. 0.94117647
 1. 1. 1. 1. 0.94117647 1.
 0.94117647 1. 1. 0.94117647 1. 0.94117647
 0.88235294 0.94117647 1. 1. 0.94117647 0.94117647
 1. 0.94117647 0.88235294 0.94117647 0.94117647 0.88235294
 1. 0.94117647 1. 1. 1. 0.9375 1.
 0.875 1. 1. 1. 0.9375 1.
 1. 1. 1. 1. 1. 1.
 1. 1. ]

Accuracy result of Naive Bayes is: 97.38235294117646

Confusion Matrix result of Naive Bayes is:
[[ 72  1]
 [ 5 170]]

Sensitivity : 0.9863013698630136
Specificity : 0.9714285714285714
```

Figure 6.3: Classification report of Naïve Bayes

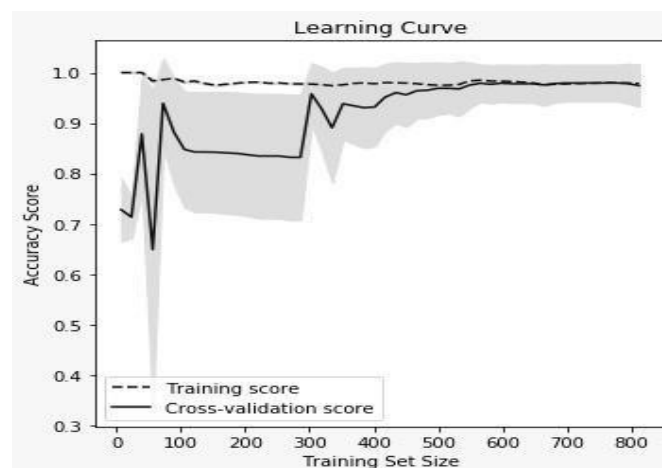


Figure 6.4: Naïve Bayes Accuracy graph

6.3 Random Forest

Random forests were a hybrid of trees forecasting flaws in forest general statement since each tree operates on the theory of the random variable within each tree. The error of generalizing forests depends on the strength of each of the forest's trees and the relationship between them on the individual since the percentage of forest trees is vital. By randomly selecting the characteristics for splitting every node, error levels are similar to adaboost, but are richer in noise. After applying Random Forests algorithm, the prediction accuracy obtained was 99.16

```

Classification report of Random Forest Results:

              precision    recall  f1-score   support

     0       0.99      0.99      0.99         73
     1       0.99      0.99      0.99        175

 accuracy          0.99          248
 macro avg          0.99          248
weighted avg          0.99          248

Cross validation test results of accuracy:
[0.94117647 1.         1.         1.         1.         0.94117647
 1.         1.         1.         1.         1.         1.
 0.94117647 1.         1.         1.         1.         0.94117647
 1.         1.         1.         1.         1.         1.
 1.         1.         1.         0.94117647 1.         0.94117647
 1.         1.         1.         1.         0.9375      1.
 1.         1.         1.         1.         1.         1.
 1.         1.         1.         1.         1.         1.
 1.         1.         ]

Accuracy result of Random Forest is: 99.16911764705883

Confusion Matrix result of Random Forest is:
[[ 72  1]
 [ 1 174]]

Sensitivity : 0.9863013698630136
Specificity : 0.9942857142857143

```

Figure 6.5: Classification report of Random Forest

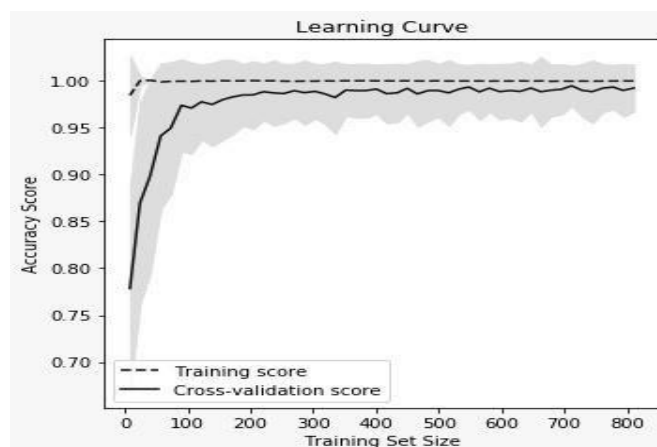


Figure 6.6: Random Forest Accuracy graph

6.4 Support Vector Machine

Support Vector Machine has been a collection of similar supervised learning methods for use in categorization and regression. This belongs to a family of generalized linear classifications. One unique aspect of SVM is that the empirical classification error is minimized and geometrical margin maximized at the same time. After applying SVM, the prediction accuracy obtained was 70.65

```
Classification report of Support Vector Machines Results:

              precision    recall  f1-score   support

     0       0.00         0.00         0.00         73
     1       0.71         1.00         0.83        175

 accuracy          0.35
 macro avg         0.35         0.50         0.41        248
 weighted avg      0.50         0.71         0.58        248

Cross validation test results of accuracy:
[0.70588235 0.70588235 0.70588235 0.70588235 0.70588235 0.70588235
0.70588235 0.70588235 0.70588235 0.70588235 0.70588235 0.70588235
0.70588235 0.70588235 0.70588235 0.70588235 0.70588235 0.70588235
0.70588235 0.70588235 0.6875 0.6875 0.6875 0.6875
0.6875 0.6875 0.6875 0.6875 0.6875 0.6875
0.73333333 0.73333333 0.73333333 0.73333333 0.73333333 0.73333333
0.73333333 0.73333333]

Accuracy result of Support Vector Machines is: 70.65980392156862

Confusion Matrix result of Support Vector Machines is:
[[ 0 73]
 [ 0 175]]

Sensitivity : 0.0
Specificity : 1.0
```

Figure 6.7: Classification report of Support Vector Machines

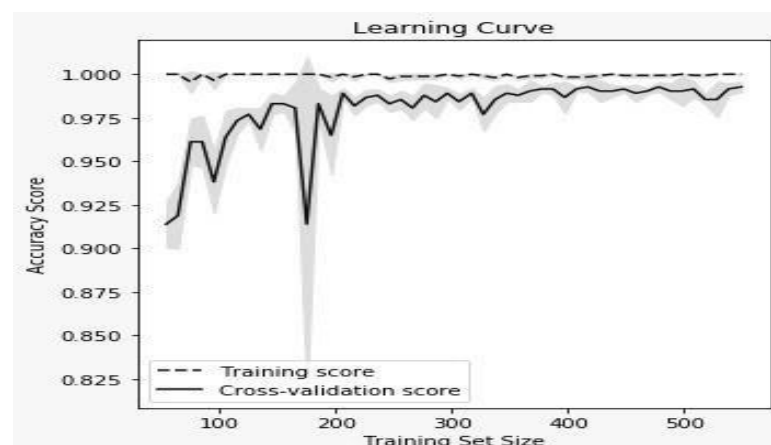


Figure 6.8: SVM Accuracy graph

6.5 K-Nearest Neighbors

K-Nearest Neighbors is the simplest and most direct method for classification where the distribution of data has had no firsthand knowledge. This rule essentially preserves the whole training set in the process of learning and assigns a class representing the majority mark of its closest neighbors in the training set to each submission. The next most simple form of KNN is the nearest neighbor's law (NN) when $N=1$. After applying KNN, the prediction accuracy obtained was 97.61

```

Classification report of K-Nearest Neighbor Results:
              precision    recall  f1-score   support

     0       0.96       0.97       0.97         73
     1       0.99       0.98       0.99        175

 accuracy          0.98
 macro avg          0.98
 weighted avg       0.98

Cross validation test results of accuracy:
[1. 1. 1. 0.94117647 0.94117647 0.94117647
 1. 0.94117647 1. 0.94117647 1. 1.
 0.94117647 1. 1. 1. 1. 1.
 1. 1. 1. 1. 0.82352941 1.
 0.94117647 1. 0.94117647 0.94117647 0.94117647 0.94117647
 0.94117647 0.94117647 1. 0.9375 1. 1.
 0.875 1. 1. 1. 0.9375 1.
 1. 1. 1. 1. 1. 1.
 1. 1. ]

Accuracy result of K-Nearest Neighbor is: 97.61764705882354

Confusion Matrix result of K-Nearest Neighbor is:
[[ 71  2]
 [ 3 172]]

Sensitivity : 0.9726027397260274

Specificity : 0.9828571428571429

```

Figure 6.9: Classification report of K-Nearest Neighbors

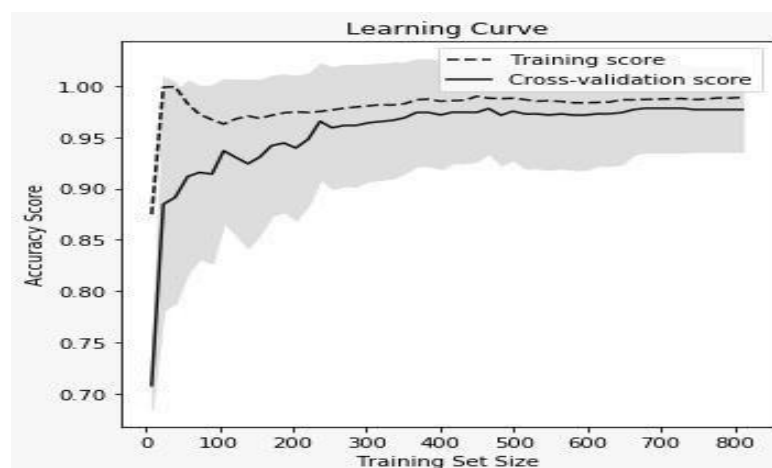


Figure 6.10: KNN Accuracy graph

6.6 Decision Tree

Decision Trees is a data gathering categorization and analysis technique. Decision trees constitute the fundamental recursive basis for the sequence process of classification, in that one of the disjoint class decision-making structures contains nodes and leaves, is allocated a case identified with a collection of attributes. Each tree node includes testing a specific attribute and every tree leaf denotes a class. The test usually compares the value with a constant of an attribute. Leaf nodes are usually categorized in all cases whereas for classified set is entered or where probabilities are spread over any possible classification. After applying Decision Tree algorithm, the prediction accuracy obtained was 99.88

```

Classification report of Decision Tree Classifier Results:

              precision    recall  f1-score   support

     0               1.00        1.00        1.00         73
     1               1.00        1.00        1.00        175

 accuracy               1.00                1.00        248
 macro avg              1.00                1.00        248
 weighted avg           1.00                1.00        248

Cross validation test results of accuracy:
[[1. 1. 1. 1. 1. 1.
 1. 1. 1. 1. 1. 1.
 1. 1. 1. 1. 1. 0.94117647
 1. 1. 1. 1. 1. 1.
 1. 1. 1. 1. 1. 1.
 1. 1. 1. 1. 1. 1.
 1. 1. 1. 1. 1. 1.
 1. 1. 1. 1. 1. 1.
 1. 1. 1. 1. 1. 1.
 1. 1. 1. 1. 1. 1.]]

Accuracy result of Decision Tree Classifier is: 99.88235294117646

Confusion Matrix result of Decision Tree Classifier is:
[[ 73  0]
 [ 0 175]]

Sensitivity : 1.0
Specificity : 1.0

```

Figure 6.11: Classification report of Decision Tree

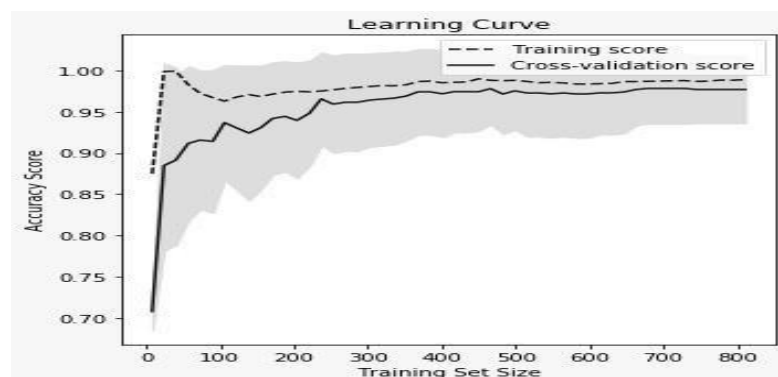


Figure 6.12: Decision Tree Accuracy graph

CHAPTER 7

CODING, TESTING

7.1 Coding

7.1.1 Environmental Requirements

Software Requirements

- Operating System : Windows

Software Used

- Anaconda with Jupyter Notebook

Hardware requirements

- Processor : Intel i5
- Hard disk : 2 GB
- RAM : minimum 2 GB

```

26., 336., 201., 55., 392., 307., 53., 439., 378., 102., 38.,
349., 32., 380., 393., 308., 58., 395., 320., 119., 410., 381.,
354., 269., 60., 408., 364., 126., 185., 421., 372., 82., 81.,
69., 42., 37., 305., 103., 48., 359., 247., 13., 49., 390.,
339., 431., 413., 223., 377., 229., 265., 207., 149., 121., 321.,
130., 67., 98., 266., 105., 9., 195., 107., 66., 1., 78.,
139., 99., 120., 190., 165., 43., 104., 92., 148., 100., 337.,
318., 214., 161., 96., 113., 83., 141., 180., 177., 142., 87.,
122., 111., 203., 146., 90., 118., 124., 240., 232., 157., 154.,
160., 112., 62., 276., 253., 176., 238., 204., 156., 193., 91.,
125., 231., 221., 420., 371., 129., 406., 325., 144., 373., 370.,
127., 115., 241., 341., 164., 287., 167., 138., 297., 277., 217.,
163., 302., 94., 280., 183., 57., 136.])

```

In [95]:

```

df['APLevels'] = df.Avg.map({70.: 'Above_70', 76.: 'Above_70', 73.: 'Above_70', 5.: 'Below_10', 41.: 'Above_40',
                                44.: 'Above_40', 29.: 'Above_20', 30.: 'Above_30', 108.: 'Above_100',
                                46.: 'Above_40',
                                64.: 'Above_60',
                                61.: 'Above_60', 2.: 'Below_10', 10.: 'Above_10', 16.: 'Above_10', 36.: 'Above_30',
                                65.: 'Above_60',
                                15.: 'Above_10', 14.: 'Above_10', 71.: 'Above_70', 39.: 'Above_30',
                                3.: 'Below_10',
                                6.: 'Below_10', 68.: 'Above_60', 342.: 'Above_300', 12.: 'Above_10', 11.: 'Above_10', 54.: 'Above_50',
                                56.: 'Above_50', 436.: 'Above_400', 457.: 'Above_450',
                                34.: 'Above_30', 95.: 'Above_90',
                                389.: 'Above_350', 314.: 'Above_300', 131.: 'Above_125', 89.: 'Above_80', 20.: 'Above_20', 418.: 'Above_400',
                                333.: 'Above_300', 40.: 'Above_40', 86.: 'Above_80',
                                63.: 'Above_10', 329.: 'Above_300',
                                257.: 'Above_250', 25.: 'Above_20', 114.: 'Above_100', 45.: 'Above_40', 311.: 'Above_300', 345.: 'Above_300',
                                17.: 'Above_10', 21.: 'Above_20', 388.: 'Above_350',
                                317.: 'Above_300', 28.: 'Above_20',
                                106.: 'Above_100', 27.: 'Above_20', 385.: 'Above_350', 360.: 'Above_350', 85.: 'Above_80', 84.: 'Above_80',
                                401.: 'Above_400', 350.: 'Above_350', 80.: 'Above_80',
                                72.: 'Above_70', 361.: 'Above_350',
                                278.: 'Above_250', 93.: 'Above_90', 51.: 'Above_50', 150.: 'Above_150', 228.: 'Above_200', 383.: 'Above_350',
                                59.: 'Above_50', 75.: 'Above_70', 356.: 'Above_350',
                                304.: 'Above_300', 50.: 'Above_50',
                                7.: 'Below_10', 8.: 'Below_10', 323.: 'Above_300', 22.: 'Above_20', 52.: 'Above_50', 396.: 'Above_350',
                                272.: 'Above_250', 47.: 'Above_40', 88.: 'Above_80',
                                426.: 'Above_400', 397.: 'Above_350',
                                19.: 'Above_10', 33.: 'Above_30', 97.: 'Above_90', 411.: 'Above_400', 374.: 'Above_350', 101.: 'Above_100',
                                35.: 'Above_30', 274.: 'Above_250', 24.: 'Above_20', 31.: 'Above_30',
                                110.: 'Above_100',
                                18.: 'Above_10', 77.: 'Above_70', 379.: 'Above_350', 235.: 'Above_200', 187.: 'Above_175',
                                4.: 'Below_10', 433.: 'Above_400', 414.: 'Above_400', 74.: 'Above_70',
                                79.: 'Above_70',
                                358.: 'Above_350',
                                26.: 'Above_20', 336.: 'Above_300', 201.: 'Above_200', 55.: 'Above_50', 392.: 'Above_350', 307.: 'Above_300',
                                53.: 'Above_50', 439.: 'Above_400', 378.: 'Above_350', 102.: 'Above_100', 38.: 'Above_30',
                                349.: 'Above_300', 32.: 'Above_30', 380.: 'Above_350', 393.: 'Above_350', 308.: 'Above_300', 58.: 'Above_50',
                                395.: 'Above_350', 320.: 'Above_300', 119.: 'Above_100', 410.: 'Above_400', 381.: 'Above_350',
                                354.: 'Above_350', 269.: 'Above_250', 60.: 'Above_60', 408.: 'Above_400', 364.: 'Above_350', 126.: 'Above_125',
                                185.: 'Above_175', 421.: 'Above_70', 372.: 'Above_350', 82.: 'Above_80', 81.: 'Above_80',
                                69.: 'Above_60', 42.: 'Above_40', 37.: 'Above_30', 305.: 'Above_300', 103.: 'Above_100', 48.: 'Above_40',
                                359.: 'Above_350', 247.: 'Above_200', 13.: 'Above_10',
                                49.: 'Above_40', 390.: 'Above_350',
                                339.: 'Above_300', 431.: 'Above_400', 413.: 'Above_400', 223.: 'Above_200', 377.: 'Above_350', 229.: 'Above_200',
                                265.: 'Above_250', 207.: 'Above_200', 149.: 'Above_125', 121.: 'Above_121'

```

```

100', 321.: 'Above_300',
    130.: 'Above_125', 67.: 'Above_60', 98.: 'Above_90', 266.: 'Above_250', 105.: 'Above_100', 9.
.: 'Below_10',
    195.: 'Above_175', 107.: 'Above_100', 66.: 'Above_60',
1.: 'Below_10', 78.: 'Above_70',
    139.: 'Above_125', 99.: 'Above_90', 120.: 'Above_100', 190.: 'Above_175', 165.: 'Above_150', 43
.: 'Above_40',
    104.: 'Above_100', 92.: 'Above_90', 148.: 'Above_125', 100.: 'Above_
00', 337.: 'Above_300',
    318.: 'Above_300', 214.: 'Above_200', 161.: 'Above_150', 96.: 'Above_90', 113.: 'Above_100', 83
.: 'Above_80',
    141.: 'Above_125', 180.: 'Above_175', 177.: 'Above_175', 142.: 'Above
125', 87.: 'Above_70',
    122.: 'Above_100', 111.: 'Above_100', 203.: 'Above_200', 146.: 'Above_125', 90.: 'Above_90', 118
.: 'Above_100',
    124.: 'Above_100', 240.: 'Above_200', 232.: 'Above_200', 157.: 'Above
150', 154.: 'Above_150',
    160.: 'Above_150', 112.: 'Above_100', 62.: 'Above_60', 276.: 'Above_250', 253.: 'Above_250', 176
.: 'Above_175',
    238.: 'Above_200', 204.: 'Above_200', 156.: 'Above_150', 193.: 'Above
175', 91.: 'Above_90',
    125.: 'Above_125', 231.: 'Above_200', 221.: 'Above_200', 420.: 'Above_400', 371.: 'Above_350', 12
9.: 'Above_125',
    406.: 'Above_400', 325.: 'Above_300', 144.: 'Above_125', 373.: 'Above
350', 370.: 'Above_350',
    127.: 'Above_125', 115.: 'Above_100', 241.: 'Above_200', 341.: 'Above_300', 164.: 'Above_150', 28
7.: 'Above_250',
    167.: 'Above_150', 138.: 'Above_125', 297.: 'Above_250', 277.: 'Above
250', 217.: 'Above_200',
    163.: 'Above_150', 302.: 'Above_300', 94.: 'Above_90', 280.: 'Above_250', 183.: 'Above_150', 57
.: 'Above_50', 136.: 'Above_125'})

```

In [96]:

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

Out[96]:

```

array(['Above_70', 'Below_10', 'Above_40', 'Above_20', 'Above_30',
      'Above_100', 'Above_60', 'Above_10', 'Above_300', 'Above_50',
      'Above_400', 'Above_450', 'Above_90', 'Above_350', 'Above_125',
      'Above_80', 'Above_250', 'Above_150', 'Above_200', 'Above_175'],
      dtype=object)

```

In [97]:

```

df['Above_70'] = df.APLevels.map({'Above_70':1, 'Below_10':0, 'Above_40':0,
                                  'Above_20':0, 'Above_30':0,
                                  'Above_100':0, 'Above_60':0, 'Above_10':0, 'Above_300':0,
                                  'Above_50':0,
                                  'Above_400':0, 'Above_450':0, 'Above_90':0, 'Above_350':0, 'Above_125':0,
                                  'Above_80':0, 'Above_250':0, 'Above_150':0, 'Above_200':0, 'Above_175':0})

```

In [98]:

```

df['Below_10'] = df.APLevels.map({'Above_70':0, 'Below_10':1, 'Above_40':0,
                                  'Above_20':0, 'Above_30':0,
                                  'Above_100':0, 'Above_60':0, 'Above_10':0, 'Above_300':0,
                                  'Above_50':0,
                                  'Above_400':0, 'Above_450':0, 'Above_90':0, 'Above_350':0, 'Above_125':0,
                                  'Above_80':0, 'Above_250':0, 'Above_150':0, 'Above_200':0, 'Above_175':0})

```

In [99]:

```

df['Above_40'] = df.APLevels.map({'Above_70':0, 'Below_10':0, 'Above_40':1,
                                  'Above_20':0, 'Above_30':0,
                                  'Above_100':0, 'Above_60':0, 'Above_10':0, 'Above_300':0,
                                  'Above_50':0,
                                  'Above_400':0, 'Above_450':0, 'Above_90':0, 'Above_350':0, 'Above_125':0,
                                  'Above_80':0, 'Above_250':0, 'Above_150':0, 'Above_200':0, 'Above_175':0})

```

In [100]:

In [149]:

Out[149]:

In [150]:

28

Out[150]:

```
array([1, 1, 1, 0, 0, 0, 0, 0, 2, 0, 1, 1, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0,
      1, 1, 0, 0, 0, 1, 4, 0, 0, 1, 1, 5, 0, 0, 1, 1, 5, 0, 0, 1, 4, 4,
      2, 0, 0, 1, 0, 1, 0, 5, 4, 0, 0, 0, 1, 0, 4, 3, 0, 2, 0, 4, 4, 1,
      0, 0, 1, 1, 4, 4, 0, 2, 0, 4, 4, 1, 1, 0, 5, 4, 1, 0, 0, 1, 1, 4,
      3, 1, 0, 0, 1, 2, 3, 4, 1, 0, 0, 1, 0, 4, 4, 1, 0, 0, 4, 1, 0, 0,
      1, 4, 3, 0, 0, 0, 1, 0, 5, 4, 1, 0, 0, 1, 0, 5, 4, 2, 0, 0, 0, 0,
      4, 3, 1, 0, 0, 4, 4, 2, 0, 0, 1, 0, 4, 3, 2, 0, 0, 1, 0, 5, 5, 1,
      0, 0, 1, 0, 4, 0, 0, 0, 0, 4, 3, 1, 0, 0, 1, 1, 4, 4, 1, 0, 0, 1,
      1, 5, 4, 2, 0, 0, 1, 0, 4, 5, 1, 1, 0, 4, 4, 1, 0, 0, 1, 0, 4, 4,
      0, 0, 0, 1, 0, 4, 4, 2, 0, 1, 0, 5, 4, 2, 0, 0, 2, 0, 4, 3, 1, 0,
      0, 5, 4, 2, 0, 0, 4, 0, 2, 1, 4, 1, 0, 0, 1, 1, 4, 1, 0, 1, 0, 4, 4,
      1, 0, 0, 0, 0, 5, 4, 1, 0, 0, 2, 0, 4, 3, 1, 0, 0, 1, 0, 4, 4, 1,
      0, 0, 1, 1, 5, 5, 2, 0, 0, 1, 0, 3, 1, 1, 0, 4, 2, 0, 1, 1, 3, 2,
      0, 0, 0, 3, 3, 0, 0, 0, 0, 0, 2, 0, 0, 0, 0, 2, 0, 1, 0, 4, 0, 0,
      1, 0, 2, 0, 0, 0, 1, 1, 1, 1, 0, 0, 3, 2, 0, 0, 1, 0, 0, 1, 1, 0,
      0, 0, 0, 2, 2, 2, 1, 0, 0, 1, 1, 1, 0, 0, 0, 0, 0, 0, 0, 0, 1, 1,
      0, 0, 0, 0, 1, 1, 1, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 2, 1, 0,
      0, 0, 2, 1, 1, 1, 0, 0, 0, 2, 2, 2, 0, 0, 0, 0, 1, 2, 2, 0, 0,
      0, 0, 1, 0, 2, 1, 2, 4, 4, 0, 0, 0, 1, 1, 3, 2, 1, 0, 0, 1, 1, 1,
      2, 0, 0, 0, 0, 1, 2, 2, 0, 0, 0, 0, 0, 1, 2, 1, 0, 1, 0, 2, 2, 1,
      0, 0, 0, 0, 0, 1, 1, 0, 0, 0, 2, 2, 2, 0, 1, 2, 0, 2, 2, 1, 0, 1,
      0, 1, 2, 2, 0, 0, 0, 0, 3, 2, 0, 0, 0, 1, 0, 0, 0, 2, 0, 1, 2, 0,
      0, 0, 1, 2, 0, 0, 0, 1, 1, 0, 0, 0, 0, 1, 2, 0, 0, 0, 0, 0, 2,
      2, 0, 0, 0, 3, 3, 0, 0, 2, 0, 2, 2, 0, 0, 0, 0, 0, 2, 2, 0,
      0, 0, 2, 2, 1, 0, 0, 0, 1, 2, 2, 2, 0, 0, 0, 0, 4, 4, 1, 0, 0, 0,
      0, 2, 1, 1, 0, 0, 1, 1, 3, 3, 1, 0, 0, 0, 3, 2, 1, 0, 0, 1, 0, 3,
      3, 1, 0, 0, 1, 2, 2, 0, 0, 0, 0, 0, 1, 2, 0, 0, 0, 1, 0, 2, 2, 0,
      0, 0, 1, 0, 1, 0, 0, 1, 0, 2, 0, 0, 0, 1, 0, 0, 0, 0, 0, 1, 1, 0,
      0, 0, 0, 0, 1, 2, 1, 0, 0, 0, 1, 2, 2, 0, 0, 0, 0, 0, 2, 2, 1, 0,
      0, 0, 0, 2, 1, 0, 0, 0, 1, 4, 1, 0, 0, 0, 4, 3, 0, 0, 0, 0, 3,
      5, 4, 2, 0, 0, 1, 2, 5, 4, 2, 0, 0, 1, 0, 5, 4, 2, 0, 0, 1, 0, 4,
      2, 0, 1, 1, 4, 0, 0, 0, 4, 1, 0, 1, 1, 4, 0, 0, 0, 0, 4, 1, 0,
      0, 1, 4, 4, 2, 0, 0, 1, 1, 4, 3, 1, 0, 0, 1, 1, 5, 5, 2, 0, 0, 1,
      0, 4, 4, 0, 1, 0, 4, 4, 2, 0, 0, 0, 3, 2, 1, 0, 0, 1, 0, 2, 0, 1,
      2, 0, 0, 0, 3, 3, 2, 0, 1, 1, 0, 4, 3, 0, 0, 0, 0, 2, 4, 4, 1,
      0, 0, 0, 2, 3, 2, 1, 0, 0, 0, 1, 2, 2, 0, 0, 0, 0, 0, 0], dtype=int64)
```

In [151]:

```
Xsp_test= df[l4]
ysp_test =df[['Source_of_Pollutants']]
np.ravel(ysp_test)
```

Out[151]:

```
array([0, 1, 2, 3, 4, 5, 6, 5, 6, 0, 1, 2, 3, 4, 5, 6, 0, 1, 2, 3, 4, 5,
      0, 1, 2, 3, 4, 6, 0, 2, 4, 5, 6, 0, 2, 4, 5, 0, 1,
      2, 3, 4, 5, 6, 5, 6, 0, 1, 2, 3, 4, 5, 6, 0, 1, 2, 5, 6, 0, 1, 2,
      3, 4, 5, 6, 0, 1, 2, 5, 6, 0, 1, 2, 3, 4, 5, 6, 0,
      1, 2, 3, 4, 5, 6, 0, 1, 2, 3, 4, 5, 6, 0, 1, 2, 5, 6, 0, 2, 3, 4,
      5, 0, 1, 2, 3, 4, 5, 6, 0, 1, 2, 3, 4, 5, 6, 0, 1, 2, 3, 4, 5, 6,
      0, 1, 2, 5, 6, 0, 1, 2, 3, 4, 5, 6, 0, 1, 2, 3, 4, 5, 6, 0, 1, 2,
      3, 4, 5, 6, 0, 2, 4, 5, 6, 0, 1, 2, 3, 4, 5, 6, 0, 1, 2, 3, 4, 5,
      6, 0, 1, 2, 3, 4, 5, 6, 0, 1, 2, 5, 6, 0, 1, 2, 3, 4, 5, 6, 0, 1,
      2, 3, 4, 5, 6, 0, 1, 2, 3, 5, 6, 0, 1, 2, 3, 4, 5, 6, 0, 1, 2, 5,
      6, 0, 1, 2, 3, 4, 5, 6, 0, 1, 2, 3, 4, 5, 6, 0, 2, 4, 5, 6, 0, 1,
      2, 3, 4, 5, 6, 0, 1, 2, 3, 4, 5, 6, 0, 1, 2, 3, 4, 5, 6, 0, 1, 2,
      2, 3, 4, 5, 6, 0, 1, 2, 3, 4, 5, 6, 0, 1, 2, 3, 4, 5, 6, 0, 1, 2,
      3, 4, 5, 6, 0, 1, 2, 3, 4, 5, 6, 0, 2, 4, 5, 0, 2, 4, 5, 6, 0, 2,
      4, 5, 6, 0, 1, 2, 3, 4, 5, 6, 0, 2, 4, 5, 6, 0, 4, 5, 6, 1, 2, 3,
      4, 6, 0, 2, 4, 5, 6, 0, 1, 2, 3, 4, 6, 1, 2, 4, 5, 5, 6, 0, 1, 2,
      3, 4, 5, 6, 0, 1, 2, 3, 4, 5, 6, 0, 2, 3, 4, 5, 1, 2, 4, 5, 0, 1,
      2, 3, 4, 5, 6, 0, 1, 2, 3, 4, 5, 6, 0, 1, 2, 3, 4, 5, 6, 0, 1, 2,
      3, 4, 5, 6, 0, 1, 2, 3, 4, 5, 6, 0, 1, 2, 3, 4, 5, 6, 0, 1, 2, 3,
      4, 5, 6, 0, 1, 2, 3, 4, 5, 6, 0, 1, 2, 3, 4, 5, 6, 0, 1, 2, 3,
      4, 5, 6, 0, 1, 4, 5, 0, 1, 2, 3, 4, 5, 6, 0, 1, 2, 3, 4, 5, 6, 0,
      1, 2, 3, 4, 5, 6, 0, 1, 2, 3, 4, 5, 6, 0, 1, 2, 4, 5, 6, 0, 1, 2,
      3, 4, 5, 6, 0, 1, 2, 3, 4, 5, 6, 0, 1, 2, 4, 5, 6, 0, 1, 2, 4, 5,
      6, 0, 1, 2, 3, 4, 5, 6, 0, 1, 2, 3, 4, 5, 6, 3, 4, 5, 6, 0, 1, 2,
      3, 6, 0, 1, 2, 3, 4, 5, 0, 1, 2, 3, 4, 5, 6, 0, 1, 2, 3, 4, 5, 6, 0,
      1, 2, 3, 5, 6, 0, 1, 2, 3, 4, 5, 6, 0, 1, 2, 3, 4, 5, 6, 0, 1, 2,
      4, 5, 0, 1, 2, 3, 4, 5, 6, 0, 1, 2, 3, 4, 5, 6, 0, 1, 2, 3, 4, 5,
      6, 0, 1, 2, 3, 4, 5, 6, 0, 1, 2, 4, 5, 6, 0, 1, 2, 3, 4, 5, 6, 0,
      1, 2, 4, 5, 6, 0, 1, 2, 3, 4, 5, 6, 0, 1, 2, 3, 4, 5, 6, 0, 1, 2,
      3, 4, 5, 6, 0, 2, 4, 5, 6, 0, 2, 3, 4, 5, 0, 2, 4, 5, 6, 0, 1, 2,
```

```

3, 4, 5, 6, 0, 1, 2, 3, 4, 5, 6, 0, 1, 2, 3, 4, 5, 6, 0, 1, 2, 3,
4, 5, 6, 0, 2, 4, 5, 6, 5, 6, 0, 2, 4, 5, 6, 0, 1, 2, 3, 4, 5, 6,
0, 1, 2, 3, 4, 5, 6, 0, 1, 2, 3, 4, 5, 6, 0, 1, 2, 3, 4, 5, 6, 0,
2, 4, 5, 6, 0, 2, 4, 5, 6, 0, 2, 4, 5, 6, 0, 2, 4, 5, 6, 0, 2, 3,
4, 5, 0, 1, 2, 3, 4, 5, 6, 0, 1, 2, 3, 4, 5, 6, 0, 1, 2, 3, 4, 5,
6, 0, 1, 2, 5, 6, 0, 1, 2, 4, 5, 6, 0, 1, 2, 3, 4, 5, 6, 1, 4, 5,
1, 2, 4, 5, 6, 0, 1, 2, 3, 4, 5, 6, 0, 1, 2, 3, 4, 5, 6, 0, 1, 2,
3, 4, 5, 6, 0, 1, 2, 3, 4, 5, 6, 0, 1, 2, 3, 4, 5, 6], dtype=int64)

```

In [152]:

```

def NBpol():

    from sklearn.tree import DecisionTreeClassifier

    gnb = DecisionTreeClassifier()

    gnb=gnb.fit(X,np.ravel(y))

    # calculating accuracy.....
    from sklearn.metrics import accuracy_score
    y_pred=gnb.predict(X_test)
    print(accuracy_score(y_test, y_pred))
    print(accuracy_score(y_test, y_pred,normalize=False))
    # .....

    terms = [st.get(),ct.get(),aplev.get()]
    for k in range(0,len(l4)):
        for z in terms:
            if(z==l4[k]):
                l5[k]=1

    inputtest = [l5]
    predict =gnb.predict(inputtest)
    predicted=predict[0]

    h='no'
    for a in range(0,len(pol)):
        if(predicted == a):
            h='yes'
            break

    if (h=='yes'):
        t3.delete("1.0", END)
        t3.insert(END, pol[a])
    else:
        t3.delete("1.0", END)
        t3.insert(END, "Not Found")

```

In [153]:

```

def NBagi():

    from sklearn.tree import DecisionTreeClassifier

    gnb = DecisionTreeClassifier()

    gnb=gnb.fit(GO,np.ravel(A))

    # calculating accuracy.....
    from sklearn.metrics import accuracy_score
    t_pred=gnb.predict(GO_test)
    print(accuracy_score(A_test, t_pred))
    print(accuracy_score(A_test, t_pred,normalize=False))
    # .....

    terms = [st.get(),ct.get(),aplev.get()]
    for m in range(0,len(l4)):
        for n in terms:
            if(n==l4[m]):
                l6[m]=1

```

```
canvas.pack()
photo = PhotoImage(file = 'im7.png')
canvas.create_image(0,0,image=photo,anchor=NW)
```

Out[156]:

1

In [157]:

```
# gui_stuff.....

# entry variables
st= StringVar()
st.set(None)
ct= StringVar()
ct.set(None)

aplev= StringVar()
aplev.set(None)

Name= StringVar()
```

In [158]:

```
# Heading
w2 = Label(canvas, justify=CENTER, text="AIR QUALITY PREDICTION - INDIA", fg="yellow",bg="gray")
w2.config(font=("Elephant", 15))
w2.grid(row=1, column=0, columnspan=2, padx=100)
```

In [159]:

```
# labels
NameLb = Label(canvas, text="NAME:", fg="yellow", bg="black")
NameLb.grid(row=3, column=0, pady=15, sticky=W)
```

In [160]:

```
stLb = Label(canvas, text="STATE NAME:", fg="yellow", bg="black")
stLb.grid(row=4, column=0, pady=10, sticky=W)

ctLb = Label(canvas, text="CITY NAME:", fg="yellow", bg="black")
ctLb.grid(row=5, column=0, pady=10, sticky=W)

aplevLb = Label(canvas, text="AQI VALUE:", fg="yellow", bg="black")
aplevLb.grid(row=6, column=0, pady=10, sticky=W)
```

In [161]:

```
nbpolLb = Label(canvas, text="Pollution Prediction:", fg="yellow", bg="black")
nbpolLb.grid(row=3, column=2, pady=10, sticky=W)
```

In [162]:

```
nbALb = Label(canvas, text="AQI-Cause:", fg="yellow", bg="black")
nbALb.grid(row=4, column=2, pady=10, sticky=W)
```

In [163]:

```
nbSPLb = Label(canvas, text="Sources of Pollutants:", fg="yellow", bg="black")
nbSPLb.grid(row=7, column=2, pady=10, sticky=W)
```

In [164]:

```
# entries
OPTIONS1 = sorted(l1)
OPTIONS2 = sorted(l2)
```

```
OPTIONS3 = sorted(l3)
```

In [165]:

```
NameEn = Entry(canvas, textvariable=Name)
NameEn.grid(row=3, column=0)
```

In [166]:

```
stEn = OptionMenu(canvas, st,*OPTIONS1)
stEn.grid(row=4, column=0)

ctEn = OptionMenu(canvas, ct,*OPTIONS2)
ctEn.grid(row=5, column=0)

aplevEn = OptionMenu(canvas, aplev,*OPTIONS3)
aplevEn.grid(row=6, column=0)
```

In [167]:

```
nbpol = Button(canvas, text="Check:Prediction of Pollutants", command=NBpol,bg="green",fg="white")
nbpol.grid(row=3, column=1,padx=10)
```

In [168]:

```
nbA = Button(canvas, text="Check:AQI-Stages", command=NBaqi,bg="green",fg="white")
nbA.grid(row=5, column=1,padx=10)
```

In [169]:

```
nbSP = Button(canvas, text="Check:Sources", command=NBsp,bg="green",fg="white")
nbSP.grid(row=8, column=1,padx=10)
```

In [170]:

```
t3 = Text(canvas, height=1, width=10,bg="black",fg="white")
t3.grid(row=3, column=2 , padx=10)
```

In [171]:

```
t4 = Text(canvas, height=5, width=50,bg="black",fg="red")
t4.grid(row=5, column=2 , padx=20)
```

In [172]:

```
t5 = Text(canvas, height=5, width=50,bg="black",fg="green")
t5.grid(row=8, column=2 , padx=20)
```

In [173]:

```
# Clear entry widget
def clear_text():
    root.delete('1.0',END)
def clear_display_result():
    t3.delete('1.0',END)
    t4.delete('1.0',END)
    t5.delete('1.0',END)
```

In [174]:

```
# Clear Text with position 1.0
def clear_text_file():
    root.destroy()
```

In [175]:

```
button3=Button(canvas,text="Reset", command=clear_display_result,width=12,bg="red",fg="white")
button3.grid(row=8,column=0,padx=10,pady=10)
```

In []:

In [176]:

```
root.mainloop()
```

```
0.7106598984771574
560
1.0
788
0.7106598984771574
560
```

In []:

In []:

7.2 Testing

The classification report of all the six algorithms was analyzed and evaluated on the data set and compared each algorithm accuracy with each which other that can be seen in Table 7.1.

Algorithm	Accuracy
Logistic Regression	97.46
Naive Bayes	97.38
Random Forest	99.16
Support Vector Machine	70.65
K-Nearest Neighbors	97.61
Decision Tree	99.88

Table 7.1: Comparison of Accuracy result of each algorithm

From Table 5.1 the decision tree with 99.88 percent precision is perhaps the most effective methodology, whereas the least accurate algorithm is Support Vector Machine Classifier with an accuracy of 70.65. On the other hand the Random Forest has almost near accuracy that of Decision Tree with an accuracy of 99.16, this is due to the fact that random forest is in itself a kind of decision tree and hence resembles almost same accuracy as decision tree. Also the logistic regression, Naïve Bayes and KNN have same accuracy of 97 approximately.

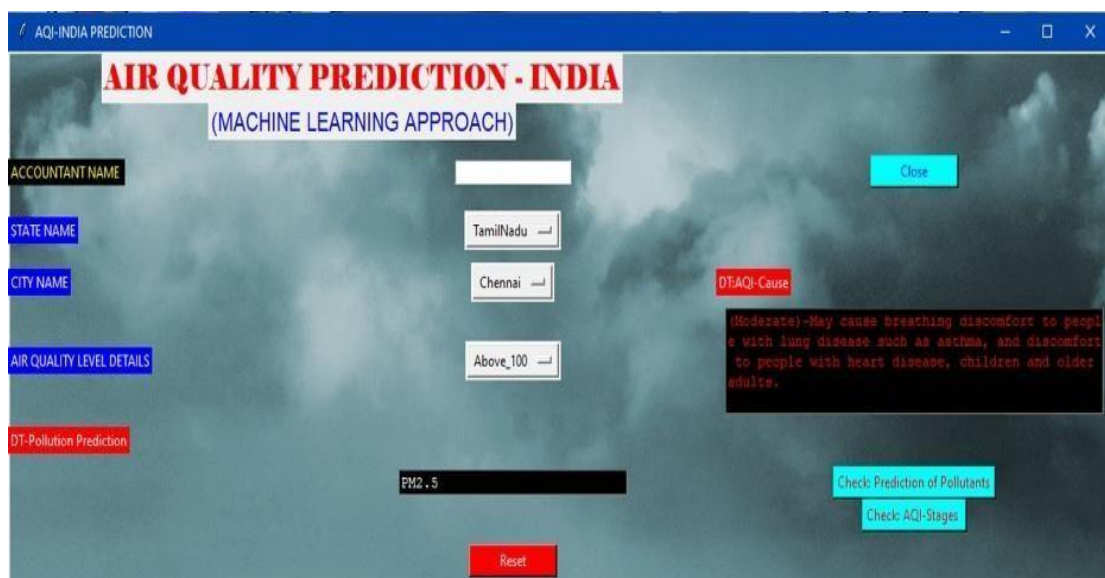


Figure 7.1: GUI Screenshot

CHAPTER 8

CONCLUSION

Prevention of air pollution is the need of the hour, so a powerful machine learning system was established with the help of prediction model. Prediction of pollution events has become most important issue in major cities in India due to the increased urbanization of the population and the associated impact of traffic volumes. Data from a variety of heterogeneous resources were used and involved collection and cleansing for use in machine learning algorithms. The number of model parameters and optimized outputs were reduced with help of structure regularization which in turn, alleviated model complexity. The Decision Tree Algorithm gave the best results among all the algorithms, with an overall accuracy of 99.8. The prediction model precision findings, helped in evaluating and contrasting current work on air quality assessment which is based upon Big Data Analytics and Machine Learning.

CHAPTER 9

FUTURE ENHANCEMENT

India meteorological department wants to automate the detecting of air quality whether good or not from eligibility process (real time). Many efforts from both local and state government are done in order to understand and predict air quality index aiming improved public health. In order to optimize the Artificial Intelligence environment, the prediction system was implemented. With the advancement of IoT infrastructures, big data technologies, and machine learning techniques, real-time air quality monitor and evaluation is desirable for future smart cities. For future work, the commonalities between nearby meteorology stations and combine them in a MTL framework, which may provide a further boosting for the prediction can be considered. This work can be extended by considering data changes over time for real-time forecasting. This can be achieved by building online models that adapt automatically to changes in environment. Also, more data can be included to increase data seasonality.

APPENDIX A

MACHINE LEARNING

A.1 Supervised Machine Learning

A supervised learning algorithm learns from labeled training data, helps to predict outcomes for unforeseen data.

A.1.1 Logistic Regression

Logistic regression is a computational methodology which often represents the associations between the independent variables x_1 x_2 ... x_n and y that is the two alternative categories of discrete dependent variable coded in 0 or 1

A.1.2 Naïve Bayes

A basic learning algorithm that uses Bayes and strongly assumes that the attributes in the groups are conditionally separate.

A.1.3 Random Forest

Random forests were a hybrid of trees forecasting flaws in forest general statement since each tree operates on the theory of the random variable within each tree.

A.1.4 Support Vector Machine

Support Vector Machine has been a collection of similar supervised learning methods for use in categorization and regression

A.1.5 K-Nearest Neighbors

K-Nearest Neighbors is the simplest and most direct method for classification where the distribution of data has had no firsthand knowledge. This rule essentially preserves the whole training set in the process of learning and assigns a class representing the majority mark of its closest neighbors in the training set to each submission

A.1.6 Decision Trees

Decision Trees is a data gathering categorization and analysis technique. Decision trees constitute the fundamental recursive basis for the sequence process of classification, in that one of the disjoint class decision-making structures contains nodes and leaves, is allocated a case identified with a collection of attributes

APPENDIX B

PARAMETERS

B.1 Precision and Recall

Pattern detection is a fraction of the appropriate instances within the retrieved instances, with knowledge retrieval and classification (machine study), accuracy (also called optimistic predictive value), while recall (also known as sensitivity) is the fraction of the total number of the specific instances that were currently retrieved . Accuracy and alert are both dependent on awareness and significance calculation.

B.2 Air Quality Index

The state and local agencies use an Air Quality Index (AQI), in order to disclose to the general population how safe or how sterile the environment is actually predictable. Various nations have their own measures of air quality, which are compatible with various national criteria of air quality.

B.3 Confusion Matrix

A confusion matrix is a table that is often used to describe the performance of a classification model (or “classifier”) on a set of test data for which the true values are known. It allows the visualization of the performance of an algorithm.

B.4 F1 Score

The indicator of precision of a check is the F1 (also F-point or F-mount). It takes into consideration both the precision p and the warning r of the check in order to measure

the score: p is the proportion of correct positive results determined by the number of possible results recorded by the classifier and r is the number of correct positive results segregated by all samples involved.

B.5PM2.5

PM2.5 applies to particle atmosphere that is smaller than 2.5 micrometers in diameter and is about 3 percent of the thickness of the hair of humans. The particles usually known as PM2.5 are so tiny that even an electron microscope can be observed. These are also smaller than PM10, which are 10 micrometers or fewer and are known as tiny particles.

B.6Learning Curve

In machine learning, the validity and testing performance for an estimator for a variety of simulation samples is seen in the learning curve (or testing curve). This is a guide to understand how much more training data will improve a machine understand algorithm, and if the estimator has a variance flaw or bias mistake. It is also a aid. When both the validity score and the training scores are converged to an extremely low value with the growing scale of the training collection, further training results does not gainmuch.

REFERENCES

1. Acharjya, D. P. (2019). "A survey on big data analytics: challenges, open research issues and tools.
2. Challa, J. S., Goyal, P., Nikhil, S., Mangla, A., Balasubramaniam, S. S., and Goyal, N. (2016). "Dd-rtree: A dynamic distributed data structure for efficient data distribution among cluster nodes for spatial data mining algorithms." *2016 IEEE International Conference on Big Data (Big Data)*. 27–36.
3. Chang, Y. S., Lin, K., Tsai, Y., Zeng, Y., and Hung, C. (2018). "Big data platform for air quality analysis and prediction." *2018 27th Wireless and Optical Communication Conference (WOCC)*. 1–3.
4. De Leon, A., Anderson, H., Bland, J., Strachan, D., and Bower, J. (1996). "Effects of air pollution on daily hospital admissions for respiratory disease in london between 1987-88 and 1991-92." *Journal of epidemiology and community health*, 50 Suppl 1, s63–70.
5. J. Gao, C.-L. Xie, and C.-Q. Tao, "Big data validation and quality assurance - issues, challenges, and needs," *IEEE Symposium on Service-Oriented System and Engineering*, IEEE Computer Society, Oxford, UK, April, 2016.
6. Li, S., Song, S., and Fei, X. (2011). "Spatial characteristics of air pollution in the main city area of chengdu, china." *2011 19th International Conference on Geoinformatics*, 1–4.
7. Qin, D., Yu, J., Zou, G., Yong, R., Zhao, Q., and Zhang, B. (2019). "A novel combined prediction scheme based on cnn and lstm for urban pm2.5 concentration." *IEEE Access*, 7, 20050–20059.
8. S. Y. Muhammad, M. Makhtar, A. Rozaimie, A. Abdul, and A. A. Jamal, "Classification model for air quality using machine learning techniques," *International Journal of Software Engineering and Its Applications*, pp. 45-52, 2015.
9. T. Chiwewe and J. Ditsela, "Machine learning based estimation of Ozone using spatio-temporal data from air quality monitoring stations," presented at 2016 IEEE 14th International Conference on Industrial Informatics (INDIN), IEEE, 2016.
10. Yue, G., Gu, K., and Qiao, J. (2019). "Effective and efficient photo -based pm2.5 concentration estimation." *IEEE Transactions on Instrumentation and Measurement*, PP, 1–10