A Project Report on

Comparative Analysis Study for Air Quality Prediction in Smart Cities Using Machine Learning

Submitted in partial fulfillment for award of

Bachelor of Technology

Degree

in

Computer Science and Engineering

By

R. Bindu Sri (Y20ACS540) **P. Pooja (Y20ACS529)**

SK. Aiman Sabaha (Y20ACS560) **R. P. R .Guptha (Y20ACS549)**

Under the guidance of

Mr. R. Veeramohana Rao, Asst. Professor

Department of Computer Science and Engineering

Bapatla Engineering College

(Autonomous)

(Affiliated to Acharya Nagarjuna University)

BAPATLA – 522 102, Andhra Pradesh, INDIA

2023-2024

Department of
Computer Science and Engineering

CERTIFICATE

This is to certify that the project report entitled **Comparative Analysis Study for Air Quality Prediction in Smart Cities Using Machine Learning** that is being submitted by R. Bindu Sri (Y20ACS540), P. Pooja(Y20ACS529), SK. Aiman Sabaha (Y20ACS560), R. P. R. Guptha (Y20ACS549) in partial fulfillment for the award of the Degree of Bachelor of Technology in Computer Science & Engineering to the Acharya Nagarjuna University is a record of bonafide work carried out by them under our guidance and supervision.

Date:

Signature of the Guide	**Signature of the HOD**
Mr. R. Veeramohana Rao	**Dr. M. Rajesh Babu**
Asst.Professor	**Associate Professor**

DECLARATION

We declare that this project work is composed by ourselves, that the work contained herein is our own except where explicitly stated otherwise in the text, and that this work has not been submitted for any other degree or professional qualification except as specified.

R. Bindu Sri (Y20ACS540)

P. Pooja (Y20ACS529)

SK. Aiman Sabaha(Y20ACS560)

R. P. R. Guptha (Y20ACS549)

Acknowledgement

We sincerely thank the following distinguished personalities who have given their advice and support for successful completion of the work.

We are deeply indebted to our most respected guide **Mr. R. Veeramohana Rao**, Assistant Professor, Department of CSE, for his valuable and inspiring guidance, comments, suggestions and encouragement.

We extend our sincere thanks to **Dr. M. Rajesh Babu**, Assoc. Prof. & Head of the Dept. for extending his cooperation and providing the required resources.

We would like to thank our beloved Principal **Dr. Nazeer Shaik** for providing the online resources and other facilities to carry out this work.

We would like to express our sincere thanks to our project coordinator **Dr. N. Sudhakar,** Prof. Dept. of CSE for his helpful suggestions in presenting this document.

We extend our sincere thanks to all other teaching faculty and non-teaching staff of the department, who helped directly or indirectly for their cooperation and encouragement.

R. Bindu Sri (Y20ACS540)

P. Pooja (Y20ACS529)

SK. Aiman Sabaha(Y20ACS560)

R. P. R. Guptha (Y20ACS549)

Table of Contents

List Of Figures

List of Tables

Abstract

Air quality prediction plays a crucial role in environmental management, public health, and urban planning. With the increasing concern over air pollution and its detrimental effects on human health and the environment, accurate prediction of air quality has become imperative. Machine learning techniques have emerged as powerful tools in this domain, enabling the development of predictive models based on various environmental and meteorological factors. This abstract presents a concise overview of the methodologies and applications of air quality prediction using machine learning algorithms. We discuss the importance of data collection from diverse sources such as monitoring stations, weather stations, and satellite imagery. Preprocessing techniques to handle missing data, outliers, and feature engineering methods are outlined. We highlight the significance of model selection, training, evaluation, and deployment in achieving accurate air quality predictions.

The Proposed System uses few Machine Learning Algorithms Such as Cat Boost, Light Gradient Boosting. We use flask module to represent graphical user interface and import the NumPy, Pandas, Matplotlib modules to access information from the dataset. We take the dataset and train the system in order to predict the air quality by taking different pollutants as input to the system.

Keywords: dataset, Machine learning-Regression methods, python.

1 Introduction

As the world is developing, Cities are getting smarter with technology, but they're still struggling with a big problem: air pollution. Bad air quality harms our health and the environment. To tackle this, we're using machines to predict air quality in cities. Our study is like a big comparison test. We're trying out different ways machines learn to see which one is best at predicting air quality. Especially, in smart cities air pollution has become a severe problem. For this we've tried to predict the air quality so that we can know whether the air we breathe is safe or not.

1.1 Objective

The main objective is to monitor real-time air pollutants by providing awareness for protecting human health and evaluate the effectiveness of air quality using different regression models. Enhancing the prediction level of air this can save the time of the users, to take the decision that how was the air is. Increase accuracy and also gives result fast.

1.2 Overview of the project

A comparative analysis study for air quality prediction in small cities using machine learning models is a multifaceted endeavor aimed at assessing the efficacy of various predictive algorithms in forecasting air quality parameters. With a focus on smaller urban areas, which often contend with unique challenges in air quality monitoring and prediction compared to their larger counterparts, such studies play a crucial role in addressing public health concerns and informing environmental management strategies.

To initiate such a study, a comprehensive literature review is conducted to glean insights from previous research endeavors in the field. This review encompasses a wide array of studies exploring air quality prediction methodologies, with particular emphasis on investigations tailored to small cities. By examining the machine learning algorithms commonly employed and any prior comparative analyses conducted, researchers gain valuable context for designing their study and selecting appropriate methodologies.

Central to the study's methodology is the meticulous collection of air quality data from monitoring stations strategically positioned within small cities. This data encompasses a spectrum of pollutants, including particulate matter (PM2.5 and PM10), ozone (O3), nitrogen dioxide (NO2), sulfur dioxide (SO2), carbon monoxide (CO), and meteorological variables such as temperature, humidity, wind speed, and precipitation. Ensuring the data's accuracy, representativeness, and temporal coverage is critical for robust analysis.

Once the data is acquired, preprocessing steps are undertaken to clean the dataset, handle missing values, and engineer relevant features. Subsequently, the dataset is partitioned into training and testing sets, with careful consideration given to temporal dynamics to prevent data leakage. With the data prepared, a suite of machine learning models is selected for evaluation, encompassing linear regression, decision trees, random forests, support vector machines, gradient boosting algorithms (e.g., CatBoost, LightGBM), and neural networks.

Each selected model undergoes rigorous training on the training dataset, followed by evaluation using appropriate metrics such as mean squared error (MSE) and R-squared (R^2) coefficient of determination. Cross-validation techniques are employed to assess model generalization and robustness. The resulting performance metrics serve as the basis for comparative analysis, enabling researchers to discern the strengths and weaknesses of each model in predicting air quality parameters in small cities.

Here we proposed an Air Quality Prediction project using Machine Learning Regression Techniques which predicts the Air Quality through a dataset of different pollutants in different cities. Here we are using Light Gradient Boosting and CatBoost algorithms to predict the air quality. It tells whether the air is safe or not through a user interface. Enhancing the prediction level of air this can save the time of the users, to take the decision that how was the air is. Increase accuracy and also gives result fast.

1.3 Machine Learning

Machine learning is an application of artificial intelligence (AI) that provides systems the ability to automatically learn and improve from experience without being explicitly programmed. Machine learning focuses on the development of computer programs that for themselves. For example, medical diagnosis, image processing, prediction, classification, etc. The intelligent systems built on machine learning algo- rhythms have the capability to learn from past experience or historical data.

1.4 Working of Machine Learning

Machine learning algorithm has two tracks: Training and Testing. Prediction of a air quality by using pollutant's concentration and history machine learning technology is striving from past decades. Machine learning operates on the principle of learning from data to make predictions, identify patterns, or make decisions without explicit programming. At its core, the process begins with data collection from diverse sources such as databases, sensors, or web APIs. This raw data undergoes preprocessing, involving cleaning, transformation, and encoding to prepare it for analysis. Next, a suitable machine learning model is selected based on the nature of the problem, data type, and computational resources available. Machine learning is inherently iterative, allowing for continuous improvement through the collection of additional data, refinement of preprocessing techniques, and experimentation with different algorithms. This iterative approach fosters ongoing learning and adaptation, enabling machine learning models to remain effective in dynamically changing environments. Overall, machine learning plays a pivotal role in various domains, driving innovation and advancing technology by leveraging the power of data to extract valuable insights and inform decision-making processes.

1.5 Supervised Machine Learning

Supervised learning represents a fundamental paradigm in machine learning where algorithms learn from labeled datasets, consisting of input features paired with corresponding target labels or outcomes. The training process involves exposing the algorithm to this labeled data, allowing it to adjust its internal parameters iteratively to minimize the discrepancy between predicted outputs and true labels. This optimization typically occurs through techniques like gradient descent, aiming to refine the model's ability to generalize patterns from the training data to unseen examples.

Types of Supervised Learning:

1. Classification
2. Regression

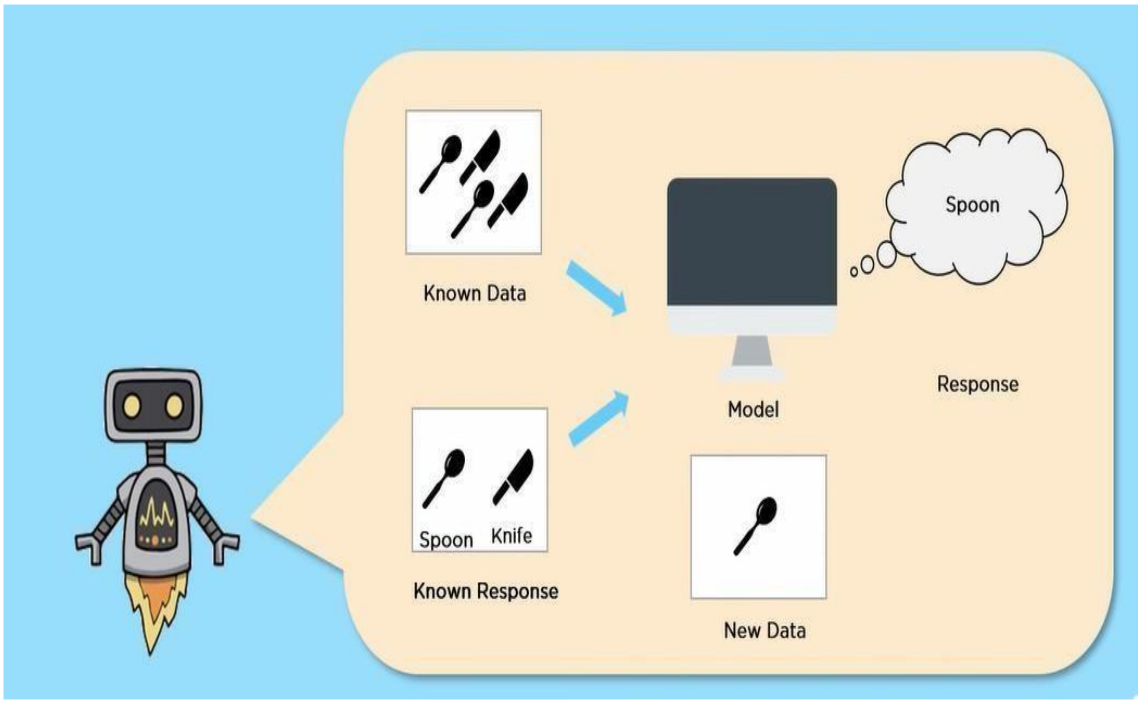


Figure 1.1 Supervised Machine Learning

Classification: In classification tasks, the goal is to predict a categorical label or class for each input instance. The output space is discrete and finite. Common examples include predicting whether an email is spam or not, classifying images into different categories (e.g., cat vs. dog), or predicting whether a patient has a certain disease based on medical data.

Regression: In regression tasks, the goal is to predict a continuous numeric value for each input instance. The output space is continuous. Common examples include predicting house prices based on features such as location, size, and amenities, forecasting stock prices based on historical data, or estimating the temperature based on weather variables. Example algorithms include linear regression, decision trees, random forests, support vector regression, and neural networks, catBoost, Light Gradient boosting.

Supervised learning finds application across diverse domains, including natural language processing, computer vision, healthcare, finance, and marketing. Its versatility and effectiveness in making predictions or classifications based on labeled data underscore its significance as a foundational approach in machine learning. As advancements continue to enhance algorithms and methodologies, supervised learning remains a pivotal area driving innovation and powering real-world applications across industries.

1.6 Unsupervised Machine Learning

In unsupervised learning, the algorithm learns from unlabeled data, where the training examples do not have associated target labels. The goal is to discover hidden patterns, structures, or relationships within the data.

Unsupervised learning tasks can include clustering, dimensionality reduction, and association rule learning:

Clustering: In clustering tasks, the algorithm groups similar instances together into clusters based on some similarity measure. Example algorithms include k-means clustering, hierarchical clustering, and DBSCAN.

Dimensionality Reduction: In dimensionality reduction tasks, the algorithm reduces the number of input features while preserving the most important information. Example algorithms include principal component analysis (PCA) and t-distributed stochastic neighbor embedding (t-SNE).

Association Rule Learning: In association rule learning tasks, the algorithm discovers interesting relationships or associations between variables in large datasets. Example algorithms include Apriori and FP-growth.

Here is how an unsupervised algorithm works:

a. You feed it an example input (without the associated output).

b. You repeat the above step many times. Eventually, the algorithm clusters your inputs into groups.

c. Now, you can feed it a new input, and the algorithm will predict which cluster it belongs to.

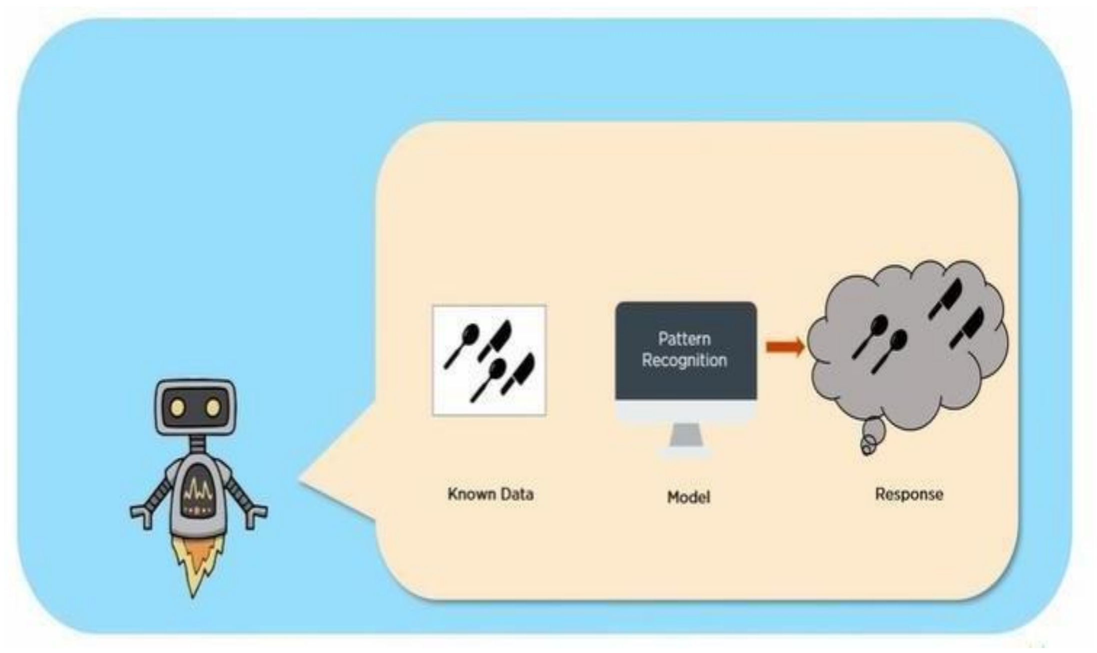


Figure 1.2   UnSupervised Machine Learning

1.7 Reinforcement Learning

It is a reward based/feedback based learning. It uses an agent and an environment to produce actions and rewards. On accomplishing a task, the agent receives a reward. In this learning, there is no predefined target variable. It follows “trial and error method” to get the desired solution.

Training Process:

Reinforcement learning algorithms do not need any external supervision to train the models just like unsupervised. They learn from the feedback to predict the correct output.

Types of problems can be solved by Reinforcement Machine Learning:

Reinforcement learning algorithms are generally used to solve **"Reward based problems"**.

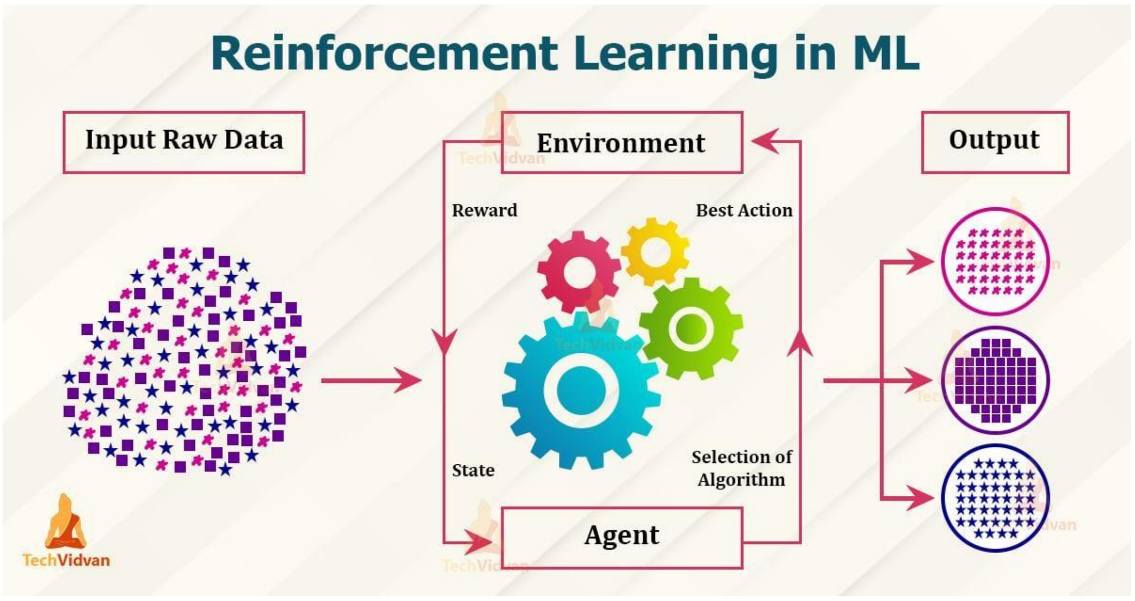


Figure 1.3 Reinforcement Machine Learning

Confusion Matrix:

A Confusion Matrix is a matrix that summarizes the performance of a machine learning model on a set of test data. It is a means of displaying the number of accurate and inaccurate instances based on the model's predictions. It is often used to measure the performance of classification models, which aim to predict a categorical label for each input instance. The matrix displays the number of instances produced by the model on the test data.

• True Positives (TP):occur when the model accurately predicts a positive data point.
• True Negatives (TN):occur when the model accurately predicts a negative data point.
• False Positives (FP): occur when the model predicts a positive data point incorrectly.
• False Negatives (FN):occur when the model mispredicts a negative data point. When assessing a classification model’s performance, a confusion matrix is essential. It offers a thorough analysis of true positive, true negative, false positive, and false negative predictions, facilitating a more profound comprehension of a model’s recall, accuracy, precision. and overall effectiveness in class distinction. When there is an uneven class distribution in a dataset, this matrix is especially helpful in evaluating a model’s performance beyond basic accuracy metrics.

Performance Metrics:

• Accuracy: Accuracy is the ratio of correctly predicted instances to the total instances in the dataset. It is a measure of overall model correctness.

$$Accuracy=TP+TN/TP+TN+FP+FN$$

•Precision: Precision is the ratio of correctly predicted positive instances to the total predicted positive instances.

$$Precision=TP/TP+FP$$

•Recall: Recall is the ratio of correctly predicted positive instances to all actual positive instances.

$$Recall=TP/TP+FN$$

•F1-Score: The F1 score is a single metric that combines both precision and recall into a single number. It provides a way to balance these two metrics, giving you a better overall understanding of how well your model is performing.

F1-Score=2∗Precision∗Recall/(Precision+Recall)

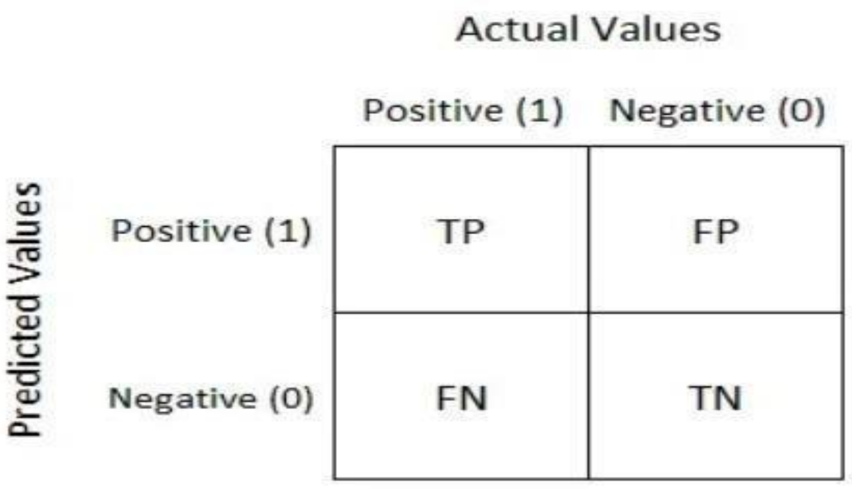


Figure 1.4 Confusion Matrix

Benefits of Performance Metrics:

Certainly, Each performance metric accuracy, precision, recall, and F1 score offers unique insights into the performance of a machine learning model. Accuracy gives a general sense of how often the model is correct, which is useful for balanced datasets. Precision focuses on the proportion of correctly predicted positive instances among all positive predictions, helping to minimize false alarms. Recall, on the other hand, emphasizes the ability to capture all actual positive instances, making it valuable for scenarios where missing positives is costly. The F1 score strikes a balance between precision and recall, providing a single metric to consider both false positives and false negatives equally.

1.8 Advantages of Machine Learning

Machine Learning can review large volumes of data and discover specific trends and patterns that would not be apparent to humans, For instance, for an e-commerce website like Amazon, it serves to understand the browsing behaviors and purchase histories of its users to help cater to the right products, deals, and reminders relevant to them. It uses the results to reveal relevant advertisements to them.

Machine learning offers a multitude of advantages across diverse domains. Through automation, it streamlines tasks, such as data analysis and decision-making, saving time and resources. Its predictive capabilities enable forecasting future trends and events, while personalization tailors experiences based on user behavior. Machine learning optimizes processes, enhances efficiency, and extracts insights from vast datasets, facilitating informed decision-making and continuous improvement. Detecting anomalies and fraud, supporting decision-making, and scalability further underscore its versatility and applicability in numerous fields, making it a pivotal tool for innovation and problem-solving.

2 Literature Survey

Many researchers have used machine learning techniques like Random Forest, Linear Regression and Decision trees to predict the quality of air.

Patil et al. [1] extensively reviewed different methodologies and techniques to analyze the concentration level of air pollution and the prediction of AQI. This study highlighted the performance of these analytical methods and presented the importance of calculating AQI as a significant measure for assessing pollution levels and how it dramatically influences human health and the environment.

Nandini et al. [2] used Decision Trees and Multinomial Logistic Regression to forecast and analyze air quality pollutant levels, achieving better accuracy with Multinomial Logistic Regression compared to Decision Tree. The research findings indicated that the Random Forest algorithm outperformed the other algorithms, demonstrating its high performance in predicting the AQI and air pollution levels.

Moreover, Pasupuleti et al. [3] conducted a study comparing Random Forest, Decision Tree, and Linear regression models for predicting air pollutants and meteorological conditions in the Arduino platform. The study found that the Random Forest model provided better performance by reducing errors caused by overfitting. However, it was noted that the Random Forest model required more memory and incurred higher costs.

Similarly, Maleki et al. [4] utilized the ANN approach to predict the concentration levels different air pollutants such as NO2 and SO2. This study applied in several monitoring areas. In this study the authors considered the effect of set parameters such as time, date, and meteorological data to offer a robust air quality predictive model.

Mahalingam et al. [5] proposed using ANN and SVM algorithms to predict the AQI in the smart city with impressive accuracies, mainly the Medium Gaussian SVM function. To predict the AQI and air pollution levels.

A noteworthy study by Ameer et al. [6] scrutinized the efficiency of four regression methods, namely Decision Tree, Gradient Boosting, Multilayer Perceptron, and Artificial Neural Network (ANN), in predicting air quality levels. These methods were evaluated based on tracking PM2.5 levels in the air and calculating the AQI.

Ganeshkumar et al. [7] presented an efficient and cost-effective classification model for environmental monitoring and air pollution prediction. Their study the authors used several artificial methods with a cloud platform for data processing, leading to significant time savings, reduced labor efforts, and producing high quality outcomes. This research highlights the importance of integrating cloud platform solutions to enhance the efficiency and accuracy of monitoring and air quality prediction models.

The primary objective of this study is to address the challenges of time and cost constraints in air quality prediction. It does so by leveraging the efficiency of machine learning techniques in conjunction with the AQI. To achieve this, the study compares three distinct regression approaches to provide the most accurate air quality prediction. To assess their effectiveness, well-established evaluation measures such as Mean Square Error (MSE), R2 score are employed. The ultimate goal is identifying the most efficient and suitable regression model for predicting air quality. This means that optimization considerations encompass factors such as data size and processing time.

2.1 Existing System

Decision Tree Regression: It's like following a flowchart to predict air quality. It's good at handling different situations but might get too specific and make mistakes sometimes.

The flowchart here means, For example, it might ask "Is it raining?" If yes, it might predict one air quality level, and if no, it might predict another.

Linear Regression: It's like drawing a straight line through historical data to predict future air quality. It's easy to understand and tells us how much each factor affects air quality, but it might miss out on complex patterns.

Random Forest: It's like asking a bunch of friends for their opinions on air quality and then averaging them out. It's really accurate and doesn't get confused by noisy data, but it's a bit harder to understand exactly how it works.

We are applying machine learning to obtain the quality of air and produce a precautions based on the result obtained which allows building models to get quickly analyze data and deliver the fast responses to the user, with the use of Cat Boost and Light Gradient Boosting Algorithms.

2.2 Limitations of Existing Systems

a) Less Accuracy in the existing systems.

b) Longer Execution Time.

c) High Complexity.

3 Proposed System

The Proposed System uses few Machine Learning Algorithms such as Catboost, Light Gradient Boost. We use flask module to represent graphical user interface and import the NumPy and pandas modules to access and perform operations on data sets. We take several data sets and train the system in order to predict the air quality by taking pollutants as input. In this system we try to provide the people with an awareness which can anticipate about quality of air in environment with high accuracy. User gives the required pollutants as input. The model predicts the output and results the quality of air and produce a precaution measure based on the result to the Output Screen.

3.1 Architecture

The architecture section provides an overview of the high-level design and structure of the software system. This section describes the overall organization of the system, the key components and modules, and the interactions between them.

3.1.1 Data set

Dataset is collected from Kaggle, it contains two files one is for training dataset, and one is testing data set.

The dataset contains 20 features, they are City, Date, PM2.5, PM10, NO, NO2, NOX, NH3, CO, SO2, OZONE, Benzene, Toluene, Xylene, Temperature, Humidity, wind speed, pressure.

The dataset is used to predict AQI based on the concentrations of various pollutants. It undergoes preprocessing, sub-index calculation, and modeling to build a predictive model for AQI.

3.1.2 Attribute selection

Attribute Selection is one of the core concepts in machine learning which hugely impacts the performance of your model. The data features that you use to train your machine learning models have a huge influence on the performance you can achieve. Attribute selection and Data cleaning should be the first and most important step of your model designing.

3.1.3 Reduces Overfitting

Less redundant data means less opportunity to make decisions based on noise.

3.1.4 Improves Accuracy

Less misleading data means modelling accuracy improves.

3.1.5 Reduces Training Time

Fewer data points reduce algorithm complexity and algorithm.

3.1.6 Processing on Data

Data mining is the process of analyzing, extracting data and furnishes the data as knowledge which forms the relationship between the available data. Some of the data mining techniques include association, clustering, classification, and prediction. Various data mining tools are compared to analyze the performance of air and air quality prediction.

Firstly, data collection involves gathering data from diverse sources, which could include databases, APIs, sensors, or files. Once collected, the data often needs cleaning to rectify errors, inconsistencies, or missing values that could compromise the integrity of the dataset. Data cleaning ensures that the dataset is reliable and accurate for analysis.

Additionally, data preprocessing techniques such as normalization, imputation of missing values, and encoding categorical variables may be applied to further enhance the quality of the data and improve model performance.

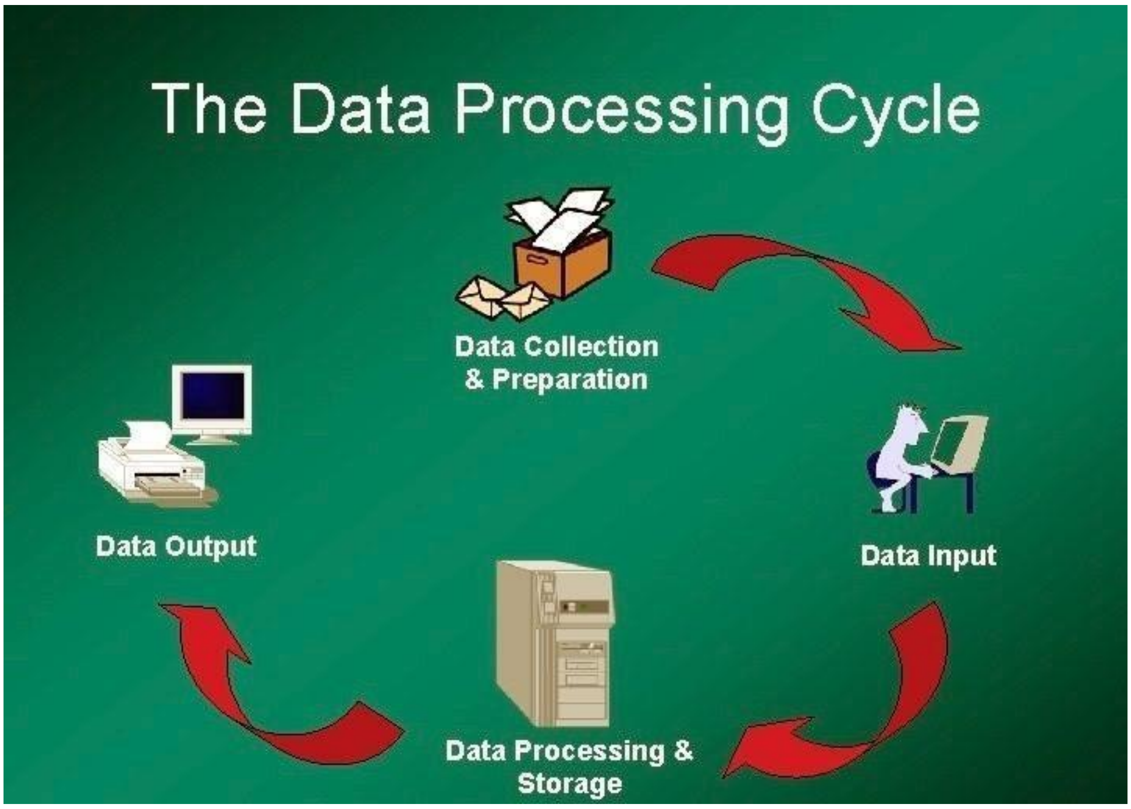

The Data Processing Cycle
Data Collection & Preparation
Data Input
Data Processing & Storage
Data Output

Figure 3.1 Data Processing

3.1.7 Machine Learning Techniques

a) Cat Boost: It is a high-performance open source library for gradient boosting on decision trees. It stands for "Categorical Boosting," emphasizing its ability to efficiently handle categorical variables without the need for extensive preprocessing.

b) Light Gradient Boosting: Light Gradient Boost is a gradient boosting framework developed by Microsoft that is known for its efficiency, speed, and high performance in handling large-scale datasets.

3.2 Features of Proposed System

In the proposed system we are using CatBoost and LightGBM are like super-smart tools for predicting air quality. CatBoost is really good at handling different kinds of information, like if it's raining or sunny, and it's great at not getting too caught up in tiny details, so it doesn't make mistakes by thinking too much about noisy data. Plus, it's super fast thanks to its special tricks that let it use graphics cards to do its work even quicker.

On the other hand, LightGBM is like a speed demon. It's super quick at sorting through lots of data, especially if there's a ton of it to look through. It's also smart enough to focus on the most important stuff first, so it doesn't waste time on things that aren't as crucial. With these tools, predicting air quality becomes much easier and faster, helping us keep an eye on our environment in real-time.

In addition to the robust features of CatBoost and LightGBM, our system incorporates advanced data preprocessing techniques to further enhance air quality prediction accuracy. By cleaning and preparing the input data, we ensure that the models receive high-quality information for training and inference. Moreover, our system integrates real-time data streaming capabilities, allowing for continuous updates and adaptation to changing environmental conditions. This dynamic approach ensures that our predictions remain up-to-date and responsive.

3.2.1 Feasibility Study

Preliminary investigation examines project feasibility, the likelihood the system will be useful to the organization. The main objective of the feasibility study is to test the Technical, Operational and Economical feasibility for adding new modules and debugging old running system. All systems are feasible if they are given unlimited resources and infinite time. There are aspects in the feasibility study portion of the preliminary investigations.

3.2.2 Technical Feasibility

To determine whether the proposed system is technically feasible, we should take into consideration the technical issues involved behind the situation. Technical feasibility center on the existing computer system and to what extent it can support the proposed addition, Python and its libraries are technology software which are used to develop Data Analytics. So, there is no need for additional purchase of any software and these are open source software which are freely available in Internet.

3.2.3 Operational Feasibility

Proposed projects are beneficial only if they can be turned out into information systems that will meet the user's operating requirements. Operational feasibility aspects of the project sire to be taken as an important part of the application implementation. This system is operational feasible since the users are familiar with the technologies and hence there is no need to gear up the personnel to use the system. Also the system is very friendly and easy to use.

3.2.4 Economic Feasibility

To decide whether a project is economically feasible, we have to take into consideration various factors as:

a) Cost benefit analysis

b) Long-term returns

c) Maintenance costs

3.3 Advantages of Proposed System

a) It provides higher accuracy.

b) We leverage not only the structured data but also the text data of pollutants based on the proposed CatBoost, Light Gradient boosting algorithms.

c) We find that by combining these two data the accuracy rate can reach high when we combine these two data.

d) To the best of our knowledge, none of the existing work focused on both data types in the area of air quality analysis.

3.4 Block diagram

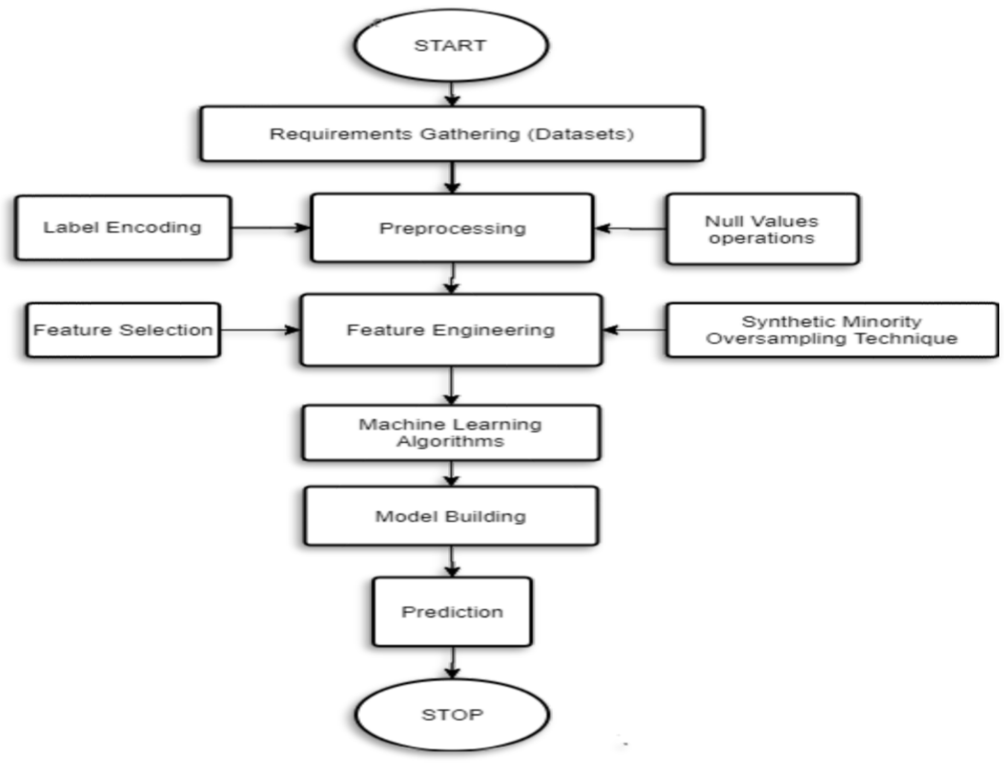


Figure 3.2 Block Diagram

DATA COLLECTION: This initial phase involves gathering relevant data from various sources, such as databases, APIs, sensors, or files.

DATA CLEANING: Raw data often contains errors, missing values, or inconsistencies. Data cleaning involves removing or correcting these issues to ensure the quality and integrity of the dataset.

DATA INTEGRATION: In some cases, data from multiple sources may need to be combined or integrated to create a unified dataset for analysis.

FEATURE SELECTION:

Feature selection becomes crucial in our research following the data preprocessing and exploratory data analysis step. This process involves identifying and selecting the most relevant features related to the AQI, representing the overall air quality. The features in this study based on the preprocessed dataset contain several pollutant information such as CO, SO2, O3, OZONE, NO2, PM10,PM2.5,.. along with their corresponding AQI values.

SPLITTING DATA :

In this stage, the train-test split() method was utilized to split the data into two parts with a ratio of 70:30 for training and testing sets. This means 70% of the total dataset was chosen for training, while the remaining 30% of data was assigned for testing data. With this splitting ratio, the model is trained on a large sufficient portion of the data and evaluated on test data to assess its performance.

4 Software and Hardware Requirements

4.1 Software Requirements

a) Coding language	:	Python
b) Platform	:	Visual Studios
c) GUI Interface	:	flask

4.2 Hardware Requirements

a) Processor	:	Intel i7
b) RAM	:	16GB
c) Hard Disk	:	256GB
d) Operating System	:	Windows 11 required

5 Design

UML stands for Unified Modeling Language. UML is a standardized general purpose modeling language in the field of object-oriented software engineering. The standard is managed, and was created by, the Object Management Group. The goal is for UML to become a common language for creating models of object-oriented computer software. In its current form UML is comprised of two major components: a Meta-model and a notation. In the future, some form of method or process may also be added to or associated with, UML. The Unified Modeling Language is a standard language for specifying. Visualization, Constructing and documenting the artifacts of software system, as well as for business modeling and other non-software systems. The UML represents a collection of best engineering practices that have proven successful in the modeling of large and complex systems. The UML is a very important part of developing object-oriented software and the software development process. The UML uses mostly graphical notations to express the design of software projects.

5.1 Goals

The Primary goals in the design of the UML are as follows:

A. Provide users a ready-to-use, expressive visual modelling Language so that they can develop and exchange meaningful models.

B. Provide extendibility and specialization mechanisms to extend the core concepts.

C. Be independent of particular programming languages and development process.

D. Provide a formal basis for understanding the modelling language.

E. Encourage the growth of OO tools market.

F. Support higher level development concepts such as collaborations, frameworks, patterns and components.

G. Integrate best practices.

5.2 Activity Diagram

Activity diagram is another important diagram in UML to describe the dynamic aspects of the system. Activity diagram is basically a flowchart to represent the flow from one activity to another activity. The activity can be described as an operation of the system.

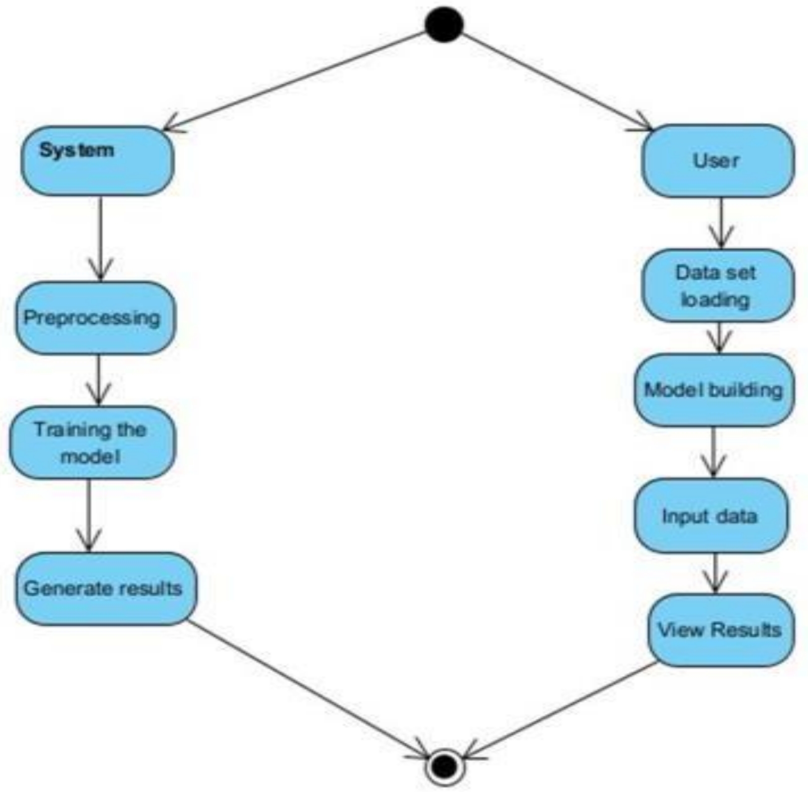


Figure 5.1 Activity Diagram

5.3 Use Case Diagram

A use case diagram in the Unified Modeling Language (UML) is a type of behavioral diagram defined by and created from a Use-case analysis.

Its purpose is to present a graphical overview of the functionality provided by a system in terms of actors, their goals (represented as use cases), and any dependencies between those use cases.

The main purpose of a use case diagram is to show what system functions are performed for which actor. Roles of the actors in the system can be depicted.

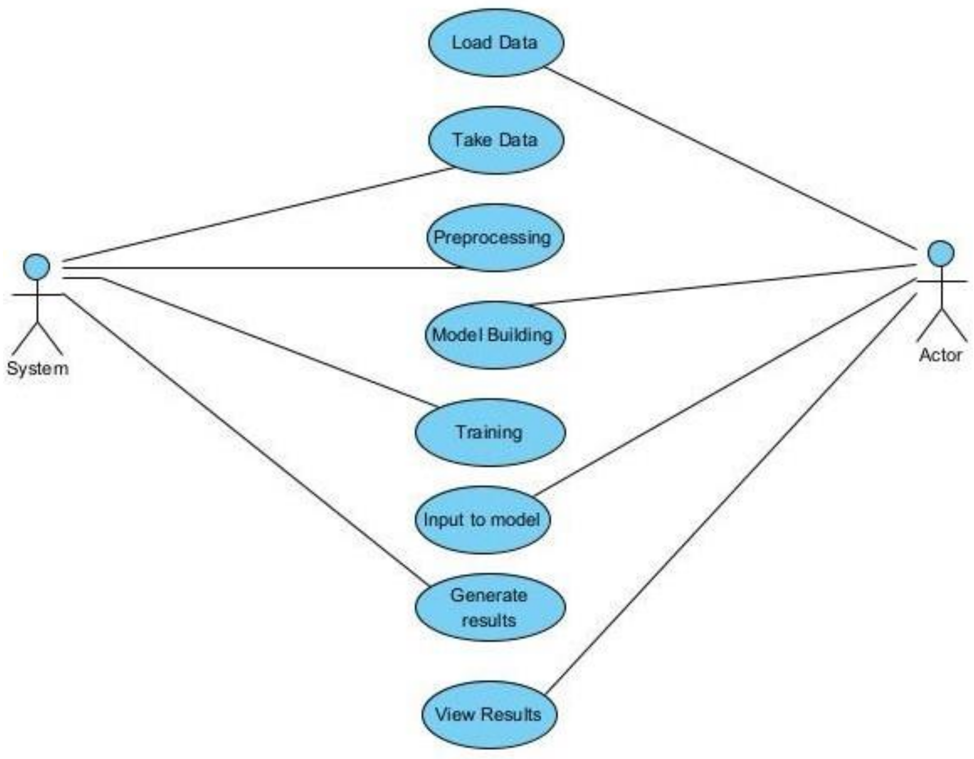


Figure 5.2 Use Case Diagram

5.4 Class Diagram

In software engineering, a class diagram in the Unified Modeling Language (UML) is a type of static structure diagram that describes the structure of a system by showing the system's classes, their attributes, operations (or methods), and the relationships among the classes. It explains which class contains information.

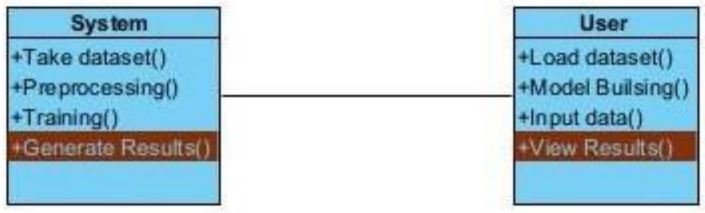


Figure 5.3 Class Diagram

5.5 Sequence Diagram

A sequence diagram in Unified Modeling Language (UML) is a kind of interaction diagram that shows how processes operate with one another and in what order.

It is a construct of a Message Sequence Chart. Sequence diagrams are sometimes called event diagrams, event scenarios, and timing diagrams.

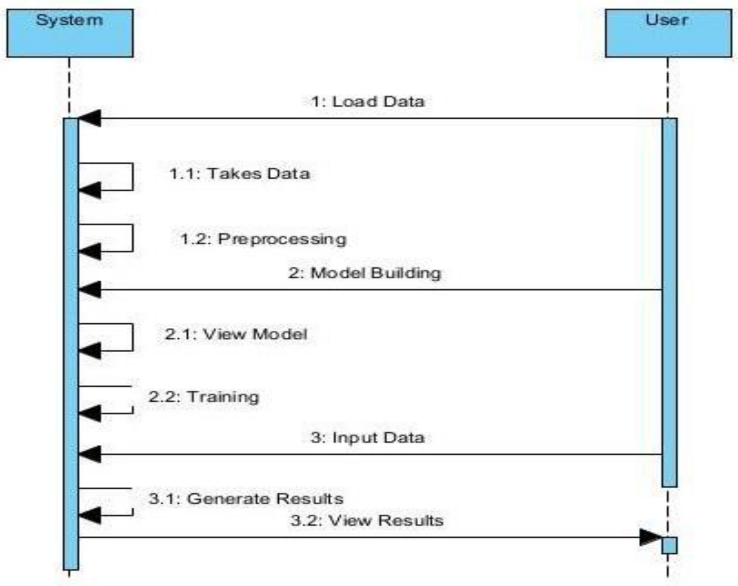


Figure 5.4 Sequence Diagram

5.6 ER Diagram

An Entity–relationship model (ER model) describes the structure of a database with the help of a diagram, which is known as Entity Relationship Diagram (ER Diagram). An ER model is a design or blueprint of a database that can later be implemented as a database. The main components of E-R model are: entity set and relationship set.

An ER diagram shows the relationship among entity sets. An entity set is a group of similar entities and these entities can have attributes. In terms of DBMS, an entity is a table or attribute of a table in database, so by showing relationship among tables and their attributes, ER diagram shows the complete logical structure of a database. Let's have a look at a simple ER diagram to understand this concept.

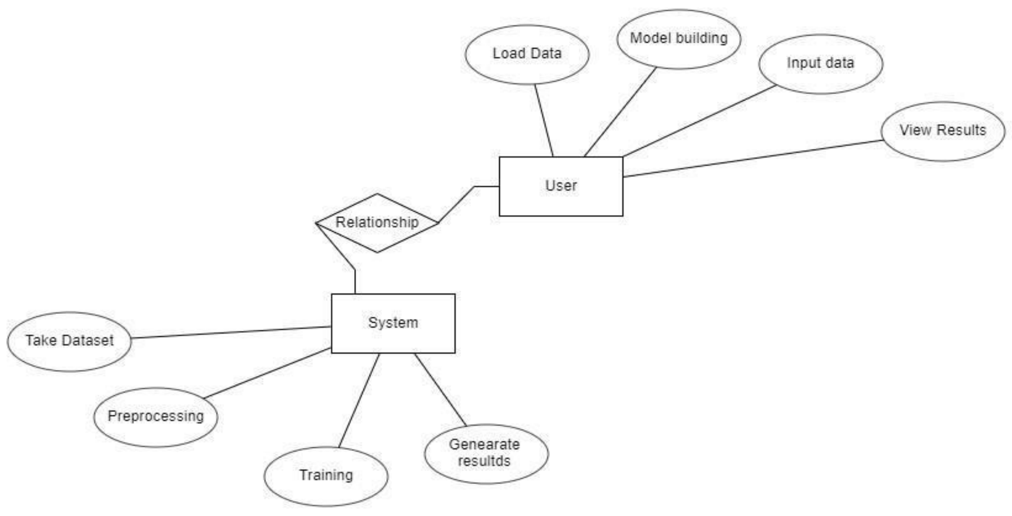


Figure 5.5 ER Diagram

5.7 Collaboration Diagram

In collaboration diagram the method call sequence is indicated by some numbering technique as shown below. The number indicates how the methods are called one after another. We have taken the same order management system to describe the collaboration diagram. The method calls are similar to that of a sequence diagram. But the difference is that the sequence diagram does not describe the object organization whereas the collaboration diagram shows the object organization.

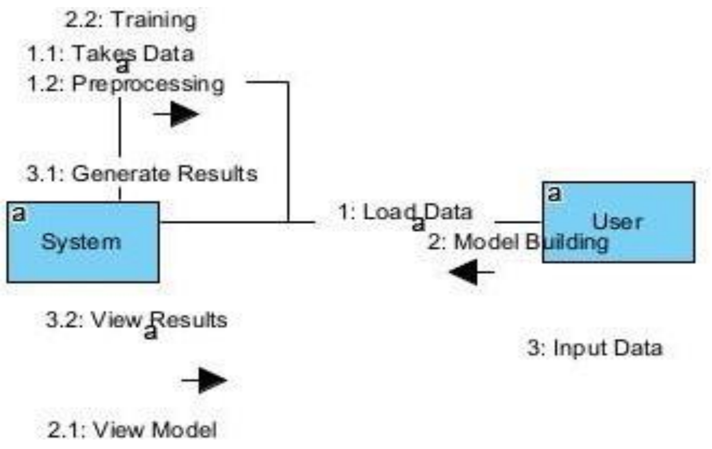


Figure 5.6 Collaboration Diagram

5.8 Data Flow Diagram

A data flow diagram (DFD) is a visual representation of the flow of data within an information system. It illustrates how data moves between processes, data stores, and external entities. Processes, depicted as circles or ovals, represent activities that transform data inputs into outputs. Data flows, shown as arrows, depict the movement of data between different components such as processes and data stores. Data stores, represented by open rectangles, indicate where data is stored within the system, such as databases or file systems. External entities, depicted as rectangles, are sources or destinations of data outside the system's boundaries. DFDs can be created at different levels of abstraction, from a high-level context diagram that provides an overview of the entire system to more detailed level diagrams that break down specific parts of the system. By visualizing data flow, DFDs help analysts and designers understand how data is processed and identify potential areas for improvement. These diagrams serve as a valuable tool for communicating data processing requirements and system architecture to stakeholders, ensuring a clear understanding of how data is managed and manipulated within the system.

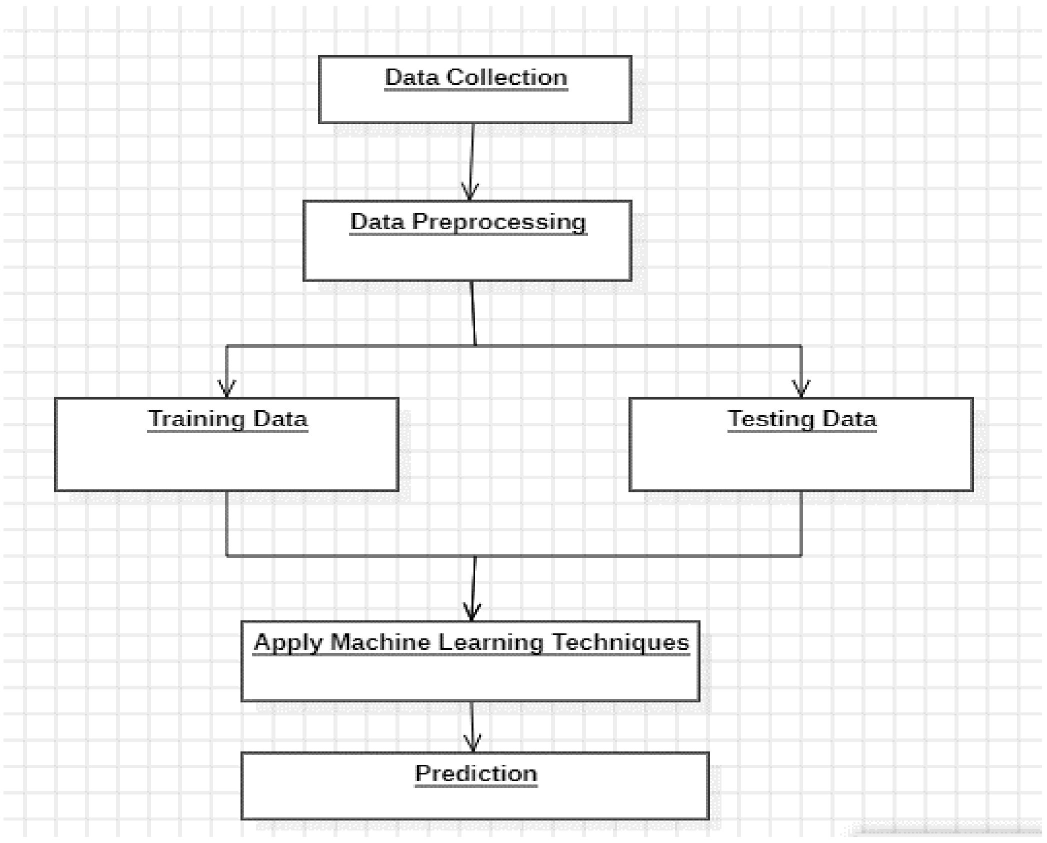


Figure 5.7 Data Flow Diagram

6 System Implementation

6.1 Data Collection

For air quality prediction in smart cities using machine learning, data collection involves gathering diverse datasets including real-time or historical air quality monitoring data on pollutants like PM2.5, PM10, NO2, SO2, CO, O3, and other pollutants from monitoring stations, meteorological data from weather stations or satellites, traffic data from traffic management systems or GPS devices, satellite imagery for detecting pollution sources, geospatial data on land use and urban morphology, crowd sourced data from citizen science initiatives or mobile apps with air quality sensors, and social media data for public perceptions. By integrating and analyzing these datasets using machine learning algorithms, predictive models can be developed to forecast air quality levels, identify pollution trends, and support decision-making for mitigating air pollution in smart cities and produce the precaution measures.

6.1.1 Dataset:

Dataset is collected from Kaggle, it contains two files one is for training dataset and

one is testing data set.

6.2 Random Forest

Random forest is a supervised learning algorithm which is used for both classification as well as regression. But however, it is mainly used for classification problems. As we know that a forest is made up of trees and more trees means more robust forest. Similarly, random forest algorithm creates decision trees on data samples and then gets the prediction from each of them and finally selects the best solution by means of voting. It is an ensemble method which is better than a single decision tree because it reduces the over-fitting by averaging the result. We can understand the working of Random Forest algorithm from figure-6.1 with the help of following steps :

a) First, start with the selection of random samples from a given dataset.
b) Next, this algorithm will construct a decision tree for every sample. Then it will get the prediction result from every decision tree.
c) In this step, voting will be performed for every predicted result.
d) At last, select the most voted prediction result as the final prediction result.

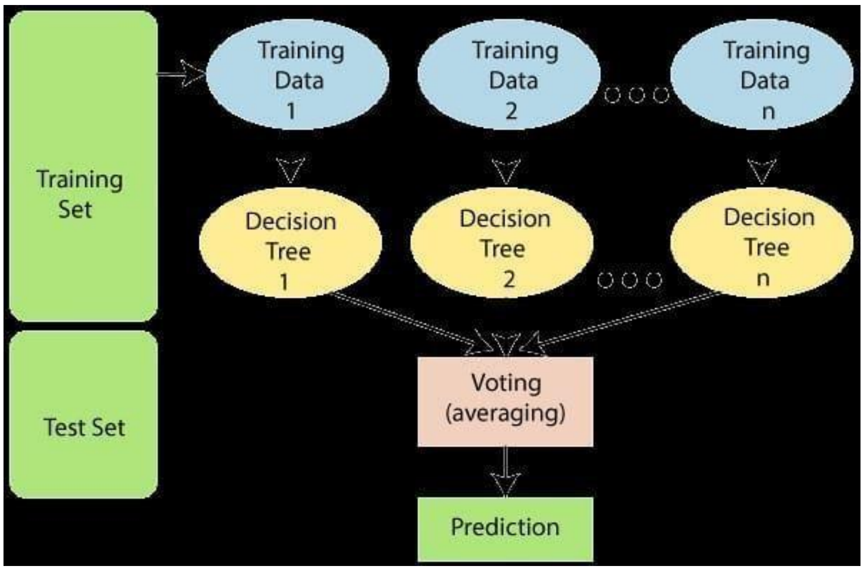


Figure 6.1 Random Forest

6.3 Decision Tree

Decision Tree algorithm belongs to the family of supervised learning algorithms. Unlike other supervised learning algorithms, decision tree algorithm can be used for solving regression and classification problems too.

The general motive of using Decision Tree is to create a training model which can use to predict class or value of target variables by learning decision rules inferred from prior data (training data).

The understanding level of Decision Trees algorithm is so easy compared with other classification algorithms. The decision tree algorithm tries to solve the problem, by using tree representation. Each internal node of the tree corresponds to an attribute, and each leaf node corresponds to a class label.

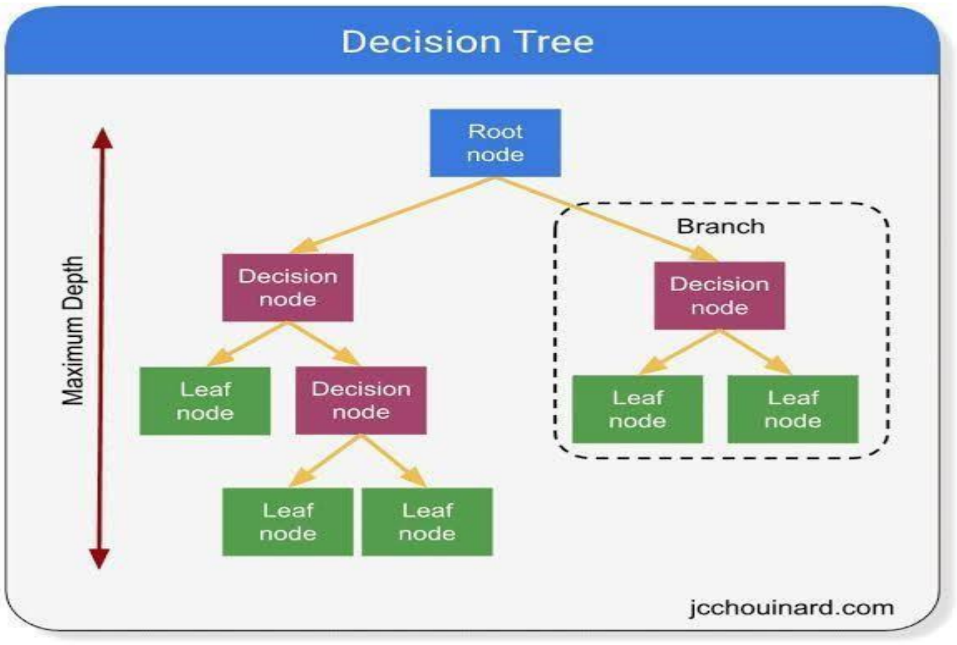


Figure 6.2 Decision Tree

Splitting the Dataset: Once the attribute is selected, the dataset is split into subsets based on the possible values of that attribute. Each subset corresponds to a branch in the decision tree.

Building the Tree: As the recursive splitting proceeds, the decision tree structure is built, with each node representing a decision based on an attribute and each leaf node representing the final prediction or classification.

Prediction: Once the decision tree is constructed, it can be used to make predictions on new data by traversing the tree from the root node down to a leaf node based on the values of the input features. The prediction at the leaf node represents the final decision or output of the model.

Evaluation: The performance of the decision tree model is evaluated using appropriate metrics, such as accuracy, precision, recall, or mean squared error, depending on whether it is used for classification or regression tasks.

6.4 Linear Regression

Linear regression is one of the easiest and most popular Machine Learning algorithms. It is a statistical method that is used for predictive analysis. Linear regression makes predictions for continuous/real or numeric variables such as **sales, salary, age, product price,** etc. Linear regression algorithm shows a linear relationship between a Dependent(y) and one or more independent (y) variables, hence called as linear regression. Since linear regression shows the linear relationship, which means it finds how the value of the dependent variable is changing according to the value of the independent variable. The linear regression model provides a sloped straight line representing the relationship between the variables.

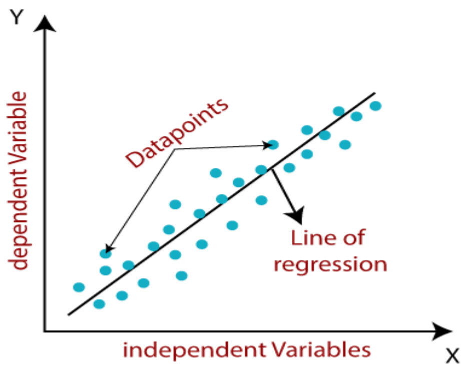


Figure 6.3 Linear Regression

6.5 CatBoost

CatBoost is a high-performance open source library for gradient boosting on decision trees. CatBoost is an algorithm for gradient boosting on decision trees. It is developed by Yandex researchers and engineers, and is used for search, recommendation systems, personal assistant, self-driving cars, weather prediction and many other tasks at Yandex and in other companies, including CERN, Cloudflare and Careem taxi. It is in open-source and can be used by anyone. CatBoost, the new kid on the block, has been around for a little more than a year now, and it is already threatening XGBoost, LightGBM.

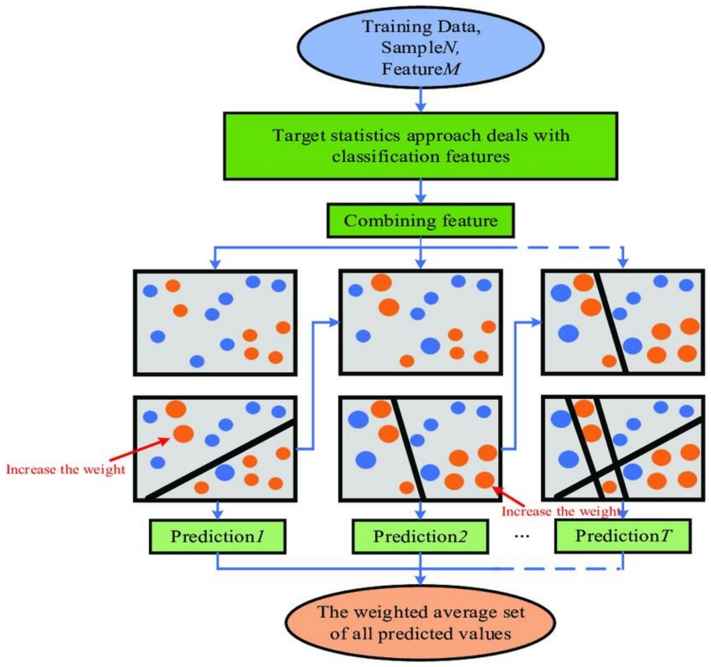


Figure 6.4 CatBoost Regression

1. Gradient Boosting: CatBoost follows the gradient boosting framework, which combines multiple weak learners (typically decision trees) sequentially to create a strong predictive model. It minimizes a loss function by adding new models that correct errors made by existing models.

2. Handling Categorical Features: CatBoost automatically handles categorical features without the need for preprocessing like one-hot encoding or label encoding. It uses an efficient method to deal with categorical variables during training, which reduces the risk of overfitting.

3. Optimized Tree Building: CatBoost employs a novel algorithm for building decision trees, which significantly speeds up the training process. It uses a variant of the gradient boosting algorithm called "Ordered Boosting" to build trees in a more efficient and optimized manner.

4. Regularization: CatBoost provides built-in methods for preventing overfitting, such as depth regularization and feature combinations. These techniques help generalize the model and improve its performance on unseen data.

5. Performance: CatBoost often performs well out-of-the-box with minimal hyperparameter tuning, thanks to its effective handling of categorical variables and built-in regularization techniques.

6.6 Light Gradient Boosting

Light Gradient Boost is a gradient boosting framework developed by Microsoft that is known for its efficiency, speed, and high performance in handling large-scale datasets. It excels in efficiency and speed due to its lightweight design and innovative techniques such as histogram-based splitting for categorical features and leaf-wise growth strategy. Light GBM introduces Gradient-Based One-Side Sampling (GOSS) to reduce training data size without sacrificing model performance and Exclusive Feature Bundling (EFB) to enhance memory usage and accelerate training. With support for parallel and GPU learning, Light GBM is capable of handling large-scale datasets and a wide range of machine learning tasks including classification, regression, ranking, and recommendation systems, making it a popular choice for practitioners seeking both accuracy and efficiency in their models.

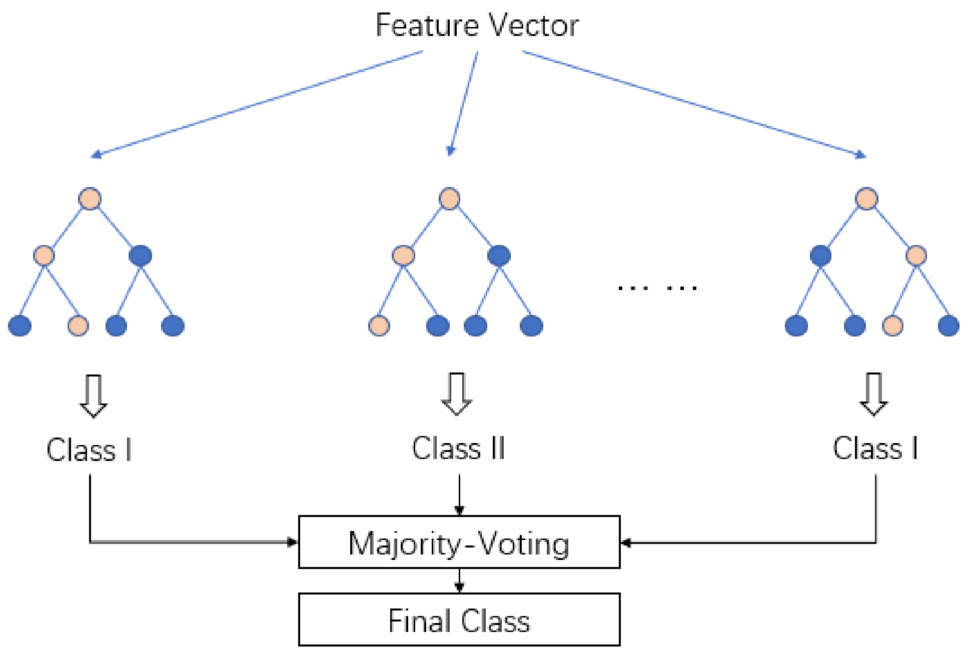


Figure 6.5 Light Gradient Boosting Algorithm

1. Leaf-Wise Growth: LightGBM employs a leaf-wise growth strategy when constructing decision trees. Instead of growing the tree level by level, LightGBM selects the leaf with the maximum delta loss (improvement in the objective function) to split at each step.

2. Model Evaluation: LightGBM evaluates the performance of the trained model using a separate validation dataset. It monitors metrics such as MSE, MAE, or R-squared to assess the model's accuracy.

3. Prediction: Once training is complete, LightGBM can be used to make predictions on new data. Given the features of a new instance, LightGBM traverses the ensemble of decision trees and aggregates their predictions to produce the final output.

7 Testing

Software Testing is a process of executing the application with an intent to find any software bugs. It is used to check whether the application met its expectations and all the functionalities of the application are working. The final goal of testing is to check whether the application is behaving in the way it is supposed to under specified conditions. All aspects of the code are examined to check the quality of application. The primary purpose of testing is to detect software failures so that defects may be uncovered and corrected.

7.1 Testing Levels

There are various testing levels based on the specificity of the test.

7.1.1 Unit testing

Unit testing refers to tests conducted on a section of code in order to verify the functionality of that piece of code. This is done at the function level.

7.1.2 Integration Testing

Integration testing is any type of software testing that seeks to verify the interfaces between components of a software design. Its primary purpose is to expose the defects associated with the interfacing of modules.

7.1.3 System Testing

System testing tests a completely integrated system to verify that the system meets its requirements.

7.1.4 Performance Testing

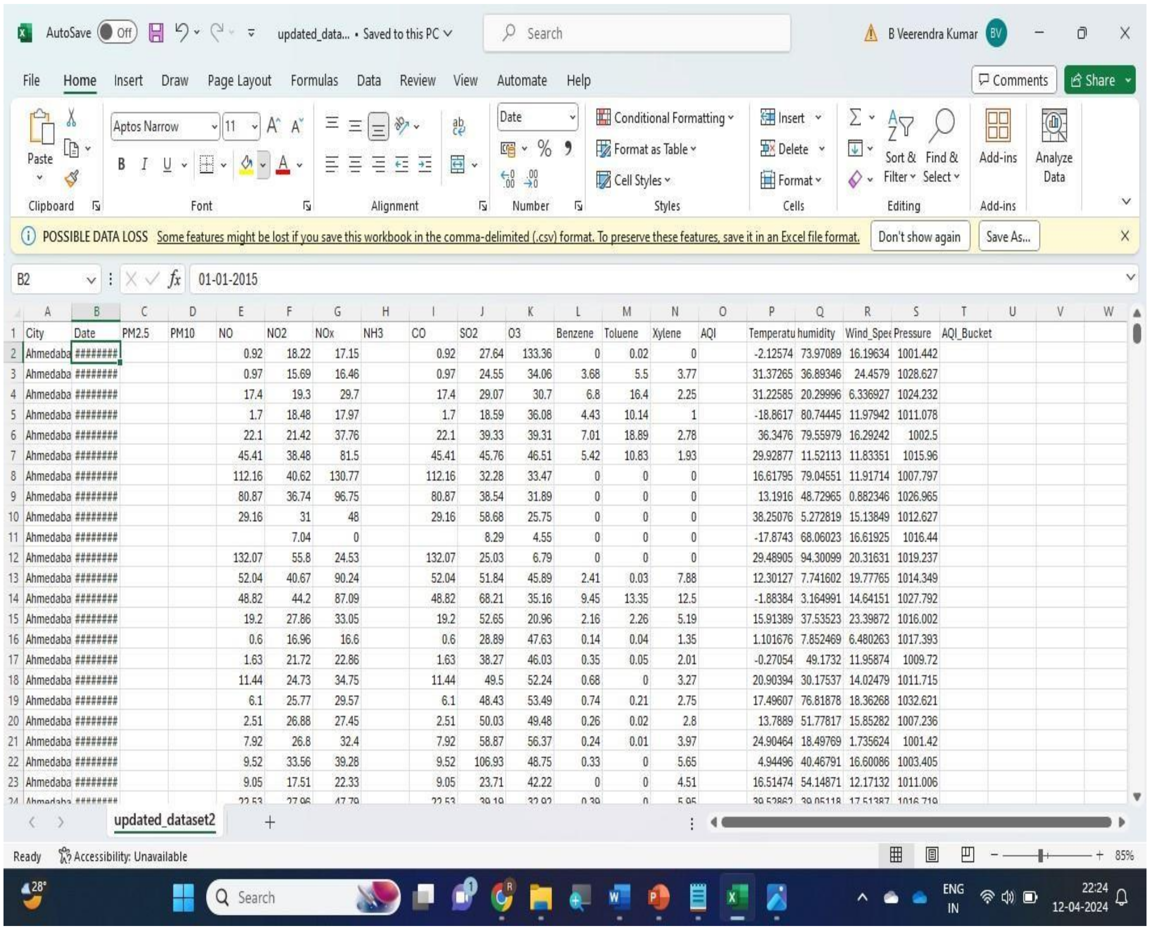
Performance testing is the process of determining the speed or effectiveness of a computer, network, software program or devices such as response time or millions.

8 Code Implementation

Code implementation refers to the process of translating a design or idea into a functioning software program. It involves writing code in a programming language that can be compiled or interpreted by a computer.

8.1 Dataset

The dataset was taken from then Kaggle website where it contains the concentration of each pollutant present in air, each pollutant may have different concentration because of weather present in different cities.



POSSIBLE DATA LOSS Some features might be lost if you save this workbook in the comma-delimited (.csv) format. To preserve these features, save it in an Excel file format.

B2 01-01-2015

	City	Date	PM2.5	PM10	NO	NO2	NOx	NH3	CO	SO2	O3	Benzene	Toluene	Xylene	AQI	Temperatu	humidity	Wind_Spe	Pressure	AQI_Bucket
2	Ahmedaba	########			0.92	18.22	17.15		0.92	27.64	133.36	0	0.02	0		-2.12574	73.97089	16.19634	1001.442	
3	Ahmedaba	########			0.97	15.69	16.46		0.97	24.55	34.06	3.68	5.5	3.77		31.37265	36.89346	24.4579	1028.627	
4	Ahmedaba	########			17.4	19.3	29.7		17.4	29.07	30.7	6.8	16.4	2.25		31.22585	20.29996	6.336927	1024.232	
5	Ahmedaba	########			1.7	18.48	17.97		1.7	18.59	36.08	4.43	10.14	1		-18.8617	80.74445	11.97942	1011.078	
6	Ahmedaba	########			22.1	21.42	37.76		22.1	39.33	39.31	7.01	18.89	2.78		36.3476	79.55979	16.29242	1002.5	
7	Ahmedaba	########			45.41	38.48	81.5		45.41	45.76	46.51	5.42	10.83	1.93		29.92877	11.52113	11.83351	1015.96	
8	Ahmedaba	########			112.16	40.62	130.77		112.16	32.28	33.47	0	0	0		16.61795	79.04551	11.91714	1007.797	
9	Ahmedaba	########			80.87	36.74	96.75		80.87	38.54	31.89	0	0	0		13.1916	48.72965	0.882346	1026.965	
10	Ahmedaba	########			29.16	31	48		29.16	58.68	25.75	0	0	0		38.25076	5.272819	15.13849	1012.627	
11	Ahmedaba	########				7.04	0			8.29	4.55	0	0	0		-17.8743	68.06023	16.61925	1016.44	
12	Ahmedaba	########			132.07	55.8	24.53		132.07	25.03	6.79	0	0	0		29.48905	94.30099	20.31631	1019.237	
13	Ahmedaba	########			52.04	40.67	90.24		52.04	51.84	45.89	2.41	0.03	7.88		12.30127	7.741602	19.77765	1014.349	
14	Ahmedaba	########			48.82	44.2	87.09		48.82	68.21	35.16	9.45	13.35	12.5		-1.88384	3.164991	14.64151	1027.792	
15	Ahmedaba	########			19.2	27.86	33.05		19.2	52.65	20.96	2.16	2.26	5.19		15.91389	37.53523	23.39872	1016.002	
16	Ahmedaba	########			0.6	16.96	16.6		0.6	28.89	47.63	0.14	0.04	1.35		1.101676	7.852469	6.480263	1017.393	
17	Ahmedaba	########			1.63	21.72	22.86		1.63	38.27	46.03	0.35	0.05	2.01		-0.27054	49.1732	11.95874	1009.72	
18	Ahmedaba	########			11.44	24.73	34.75		11.44	49.5	52.24	0.68	0	3.27		20.90394	30.17537	14.02479	1011.715	
19	Ahmedaba	########			6.1	25.77	29.57		6.1	48.43	53.49	0.74	0.21	2.75		17.49607	76.81878	18.36268	1032.621	
20	Ahmedaba	########			2.51	26.88	27.45		2.51	50.03	49.48	0.26	0.02	2.8		13.7889	51.77817	15.85282	1007.236	
21	Ahmedaba	########			7.92	26.8	32.4		7.92	58.87	56.37	0.24	0.01	3.97		24.90464	18.49769	1.735624	1001.42	
22	Ahmedaba	########			9.52	33.56	39.28		9.52	106.93	48.75	0.33	0	5.65		4.94496	40.46791	16.60086	1003.405	
23	Ahmedaba	########			9.05	17.51	22.33		9.05	23.71	42.22	0	0	4.51		16.51474	54.14871	12.17132	1011.006	

updated_dataset2

Figure 8.1 Features in dataset

The dataset contains 20 features, they are City, Date, PM2.5, PM10, NO, NO2, NOX, NH3, CO, SO2, OZONE, Benzene, Toluene, Xylene, Temperature, Humidity, wind speed, pressure.

The dataset is used to predict AQI based on the concentrations of various pollutants. It undergoes preprocessing, sub-index calculation, and modeling to build a predictive model for AQI.

AQI CALCULATION:

As mentioned before AQI is one of the most crucial parameter have been used for monitoring the air quality in particular cities. It provides a standard measure that quantifies air pollution and helps understand its effects on human health and environment. AQI is a numerical value within a defined range, typically from 0 to 500. A higher value of AQI indicates poorer air quality and the existence of harmful air pollutants. Each pollutant has specific constraints and specific averaging periods to ensure accurate assessment such as the period is 8-hour maximum for Q3 and 24-hour average concentrations for SO2, PM10, CO, NO2, PM2.5. To calculate the AQI, the concentrations of these air pollutants are categorized into sub-indices. These sub-indices were defined based on predefined ranges that help to give the level of air quality, ranging from ‘‘good’’ to ‘‘hazardous.’’ Where the highest value of sub-index among the air pollutants represents the overall air quality index for a certain location.

8.2 Steps for implementation process

Step-1: Data Preparation

- Install necessary libraries: numpy, pandas, matplotlib, seaborn, scikit-learn, lightgbm.
- Import libraries.
- Load dataset updated_dataset2.csv into a DataFrame df.
- Check for missing values: print(df.isnull().sum()).
- Convert the date column to datetime format and set it as the index.

Step-2: Data Preprocessing

- Fill missing values with mean for each pollutant.
- Define sub-index calculation functions for pollutants (PM10, PM2.5, SO2, NOx, NH3, CO, O3).
- Calculate sub-indices for each pollutant.
- Fill missing AQI values with the maximum sub-index value.
- Bucketize AQI into categories: Good, Satisfactory, Moderate, Poor, Very Poor, Severe.

Step-3: Exploratory Data Analysis (EDA)

- Visualize correlation among numeric features using a heatmap.
- Plot average AQI for each city over the last 5 years.

Step-4: Model Training and Evaluation

- Encode categorical features using LabelEncoder.

- Transform target variable (AQI) using log transformation for normalization.
- Split data into features (X) and target (y) variables.
- Split data into training and testing sets.
- Train and evaluate models such as:
    - Linear Regression
    - Decision Tree Regressor
    - Random Forest Regressor
    - LightGBM Regressor (Light Gradient Boosting Machine)
    - CatBoost Regressor
- Evaluate models using mean squared error (MSE), cross-validation score, and R2 score.

Step-5: Model Selection and Saving

- Choose CatBoost as the final model.
- Save the trained CatBoost model using pickle: with open('cb_model.pkl', 'wb').

Step-6: Building web Application

- By Using HTML files in the Visual Studio Code Platform we are going to build a web application to show the result of Air Quality Index, Quality of air in different categories like Good, Satisfactory, Moderate, Poor, Very Poor, Severe. And producing the precaution to the user how to survive in that environment.

Code url:

https://github.com/y20acs540/FinalYearProject.git

Heatmap

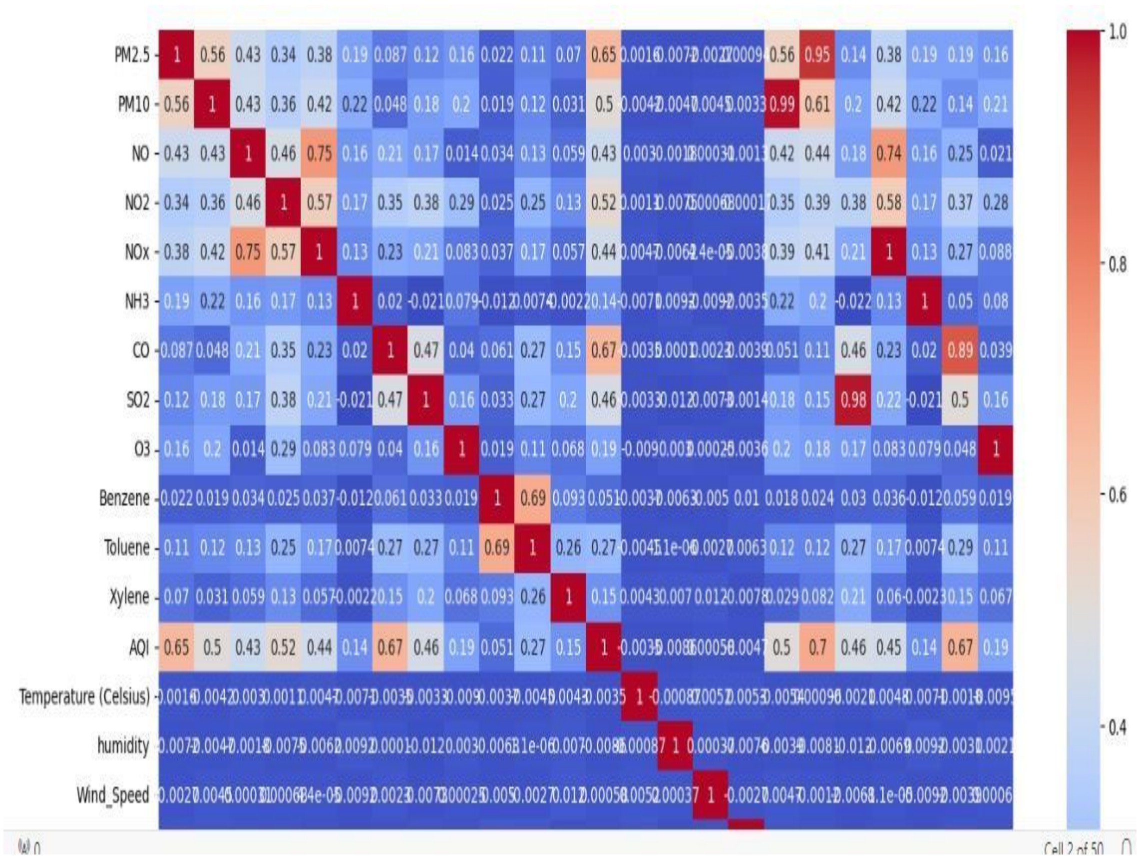


Figure 8.2 Heat Map

This code segment generates a heatmap visualization of the correlation matrix, providing a visual representation of how strongly each numeric feature is correlated with every other numeric feature in the dataset. It helps in identifying relationships and dependencies between different features, which can be useful for feature selection and understanding the dataset's structure.

Average AQI Value in last 5 years Graph

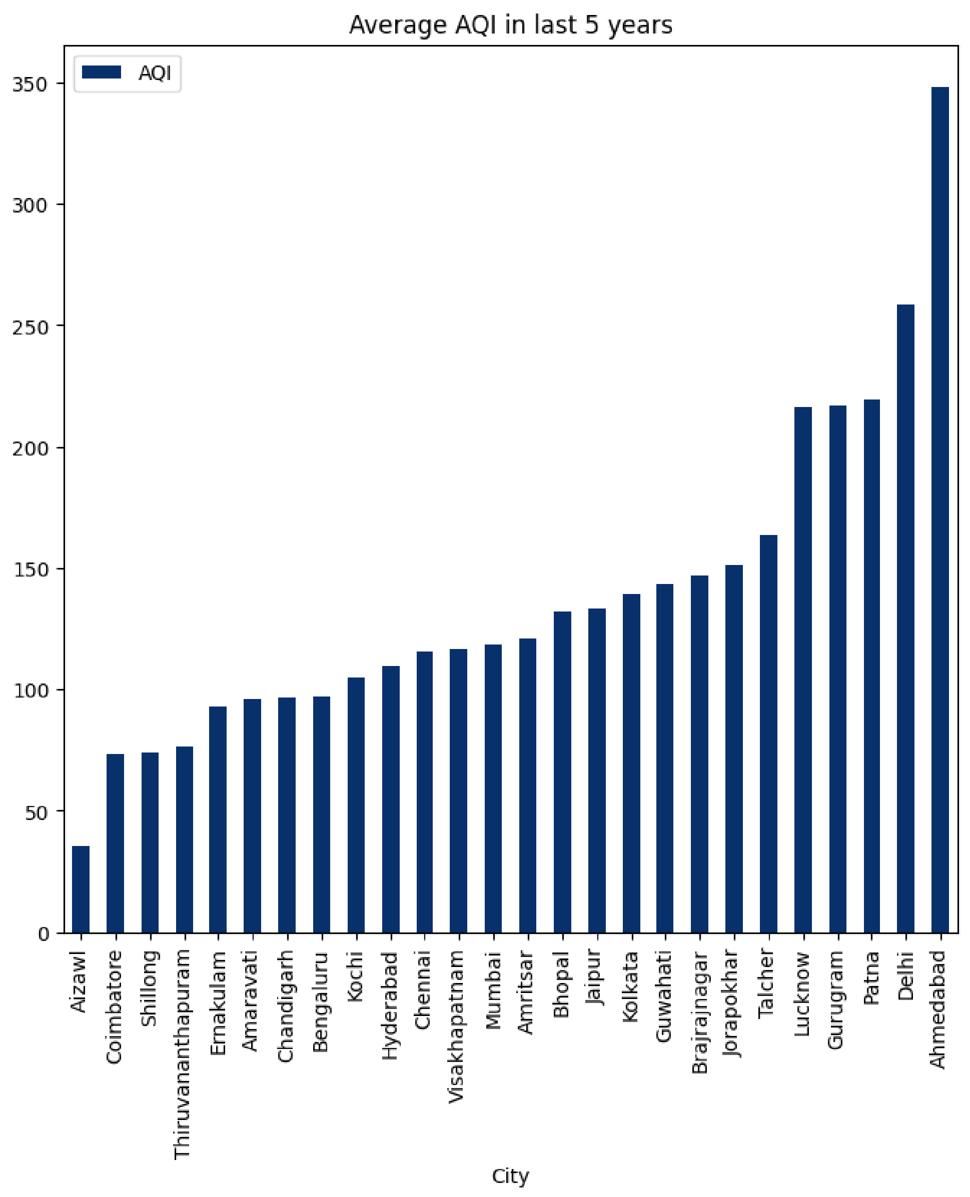


Figure 8.3 AQI Value Graph

The resulting plot visualizes the average AQI for each city over the specified time period, with bars representing the average AQI values. This visualization helps in comparing the air quality among different cities and identifying any significant differences or patterns.

8.3 Libraries Imported

Pandas:

A library for data manipulation and analysis that provides powerful data structures for working with structured data.

• This library is used to work with data sets.

• A data set is a collection of data. This library mainly works with the “Tabular data”. • It uses most of the functionalities of NumPy.

• This library offers various data structures and operations to manipulate “Numerical data & Time series data”.

• It offers 3 data structures:

1. Series: To represent 1D data. (Single column data)

2. Data Frame: To represent 2D data. (Tabular data)

3. Panel: To represent 3D data.

• “Series and Data Frame” are the most widely used pandas data structures. Each column in a data frame is called a “Series”. A data frame is a collection of series.

• Importing and Analyzing data is much easier using pandas.

Sklearn:

A library that provides a wide range of machine learning models and tools for classification, regression, clustering, and dimensionality reduction.

The sklearn module, also known as scikit-learn, is a popular machine learning library in Python that provides a simple and efficient toolset for data mining and data analysis tasks. It is built upon other Python libraries such as NumPy, SciPy, and Matplotlib, making it easy to integrate into existing data science workflows.

The sklearn module, also known as scikit-learn, is a comprehensive machine learning library in Python that offers a user-friendly interface and efficient implementations of various algorithms for classification, regression, clustering, dimensionality reduction, model selection, and preprocessing. With its consistent API, scikit-learn simplifies experimentation with different algorithms without the need to learn new syntax for each one. It provides tools for data preprocessing, model evaluation, and hyperparameter tuning, as well as support for building complex machine learning pipelines.

Numpy:

- "NumPy" stands for "Numerical Python".
- This library is used to perform mathematical and logical operations on arrays.
- This library is used to work with arrays.
- This is the core library for numerical and scientific computing in Python.
- It provides high-performance multidimensional array object and tools to work with arrays.
- In NumPy, dimensions are called "Axes".
- NumPy arrays are fast, convenient and occupies less memory.

• In Python, we have lists that serves the purpose of arrays, but they are slow to process.

• This library provides an array object which is faster than traditional python lists.

• In lists, all elements are comma separated. Whereas in NumPy, all elements are space separated.

• These arrays are stored at one continuous place in memory.

• Use the following command to install NumPy library:

```
pip install numpy
```

Matplotlib:

A 2D plotting library that allows you to create a wide variety of charts and graphs, including scatter plots, histograms, and bar charts.

• It is one of the most popular python library which is used for “Data Visualization”.

• It consists of several plots like line, bar chart, pie chart, Histogram, Pie chart, Scatter plot.

• Matplotlib contains 2 sub modules to create various types of plots.

a. Pyplot

b. Pylab

• The area/place which is used to create a chart/graph is called “Figure”.

• Inside the figure, we are having multiple “Axes”, is a place where we are drawing a chart. • Each axes is having 2 axis (i.e) “x-axis” and “y-axis”.

• “Pyplot” is a Matplotlib module that provides a MATLAB(Matrix Laboratory) like interface.

• “Pyplot” provides various functions that can be used to create & decorate a figure.

Seaborn :

A library built on top of Matplotlib that provides more advanced visualizations, including heatmaps, pair plots, and violin plots. Pickle: it is a library in python used for serializing and deserializing objects.

Seaborn is a Python data visualization library built on top of Matplotlib. It provides a high-level interface for creating attractive and informative statistical graphics. Seaborn is particularly useful for visualizing relationships and distributions in datasets, making it a valuable tool for data exploration and analysis.

Additionally, Seaborn's integration with Pandas data structures streamlines the visualization workflow, allowing users to seamlessly pass DataFrame objects to plotting functions without extensive data manipulation. Its intuitive color palettes and customizable themes and styles further enhance the aesthetics of visualizations, ensuring they are both visually appealing and effective in conveying information. Faceted plotting capabilities enable the creation of multiple plots arranged in a grid based on categorical variables, facilitating comparison across different subsets of the data. In essence, Seaborn empowers users to explore, analyze, and communicate complex datasets with ease and clarity, making it an indispensable tool in the data science toolkit.

9 Results

Comparison of diverse Regression techniques in machine Learning for predicting the best model for prediction of air quality. Here we have compared the models such as Decision Tree Regression, Random Forest Regression, Linear Regression, Cat Boost Regression, Light Gradient Boosting Algorithm and we got the accuracies like shown in Table-9.1. Here we have got highest accuracy for Cat Boost Regression, which is the best model in performance and a modern model to handle complex dataset. It follows categorical data.

Table 9.1 Accuracies of the models

SNO	MODEL	ACCURACY
1	Decision Tree Regression	82%
2	Random Forest Regression	91.2%
3	Linear Regression	79%
4	Cat Boost Regression	91.6%
5	Light Gradient Boosting Algorithm	91.3%

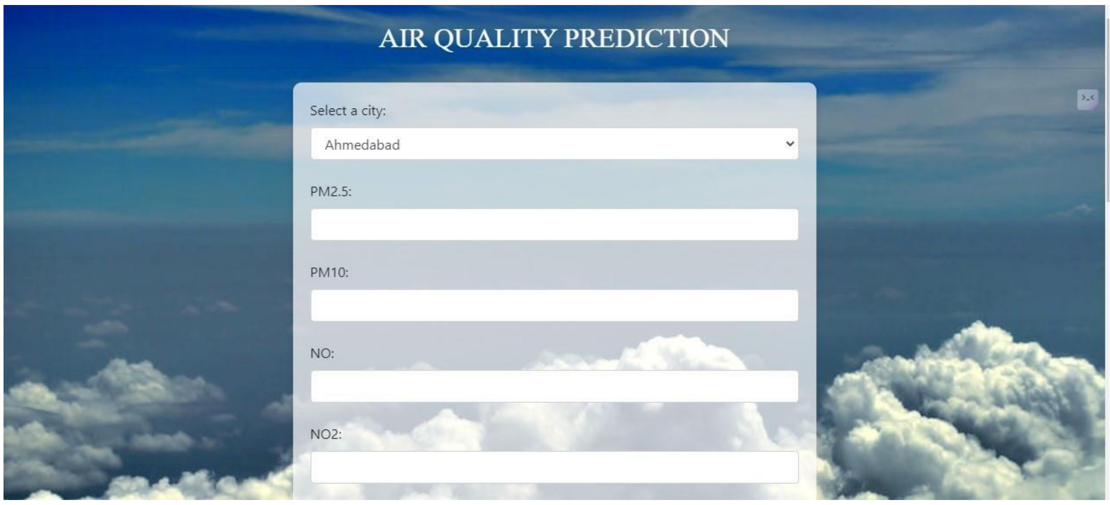
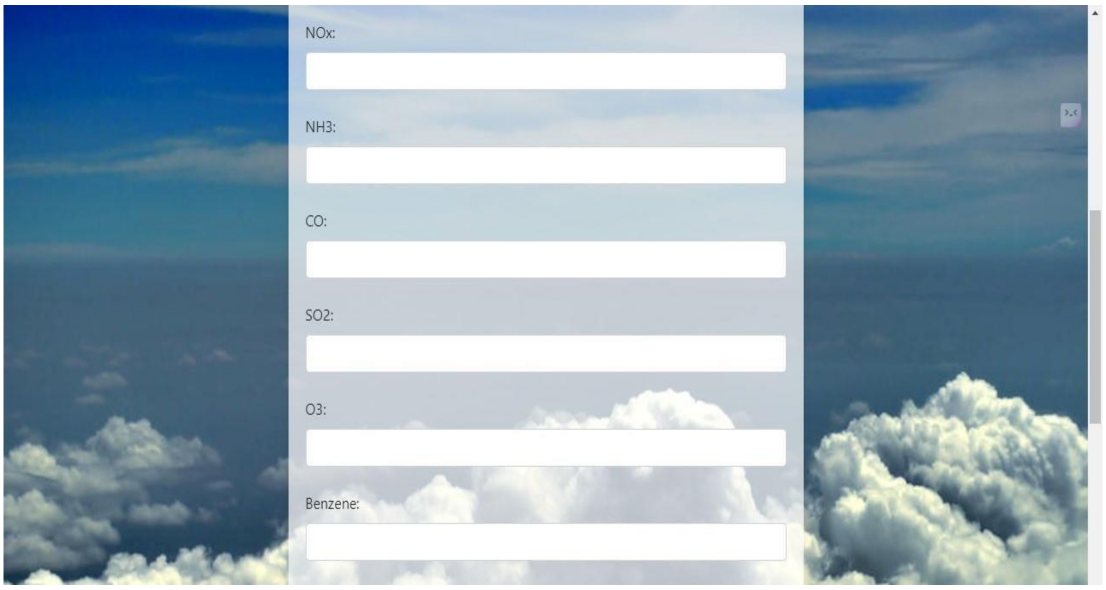
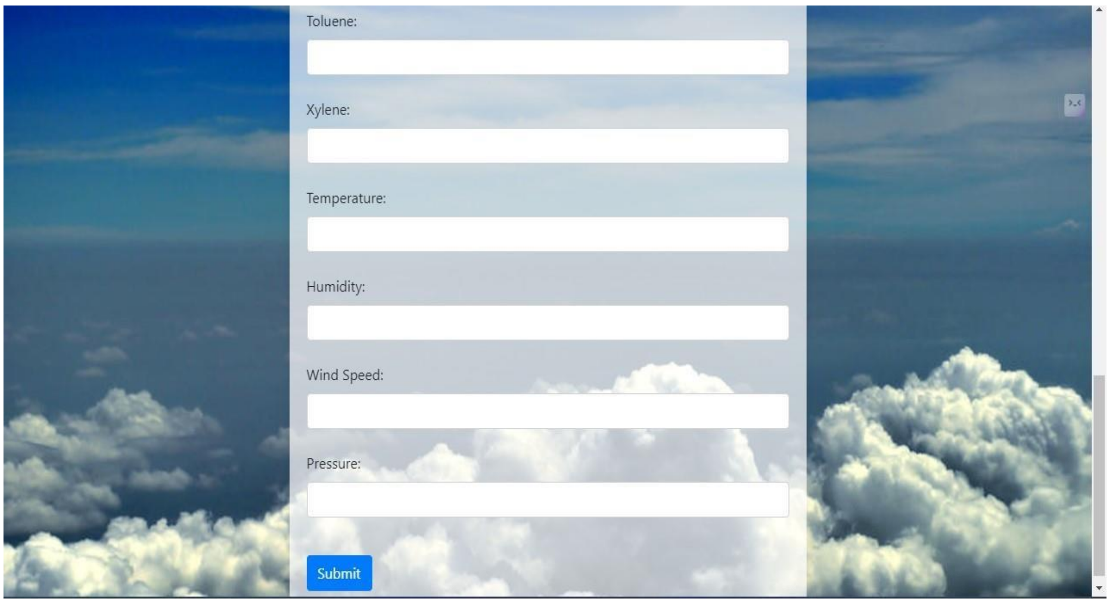
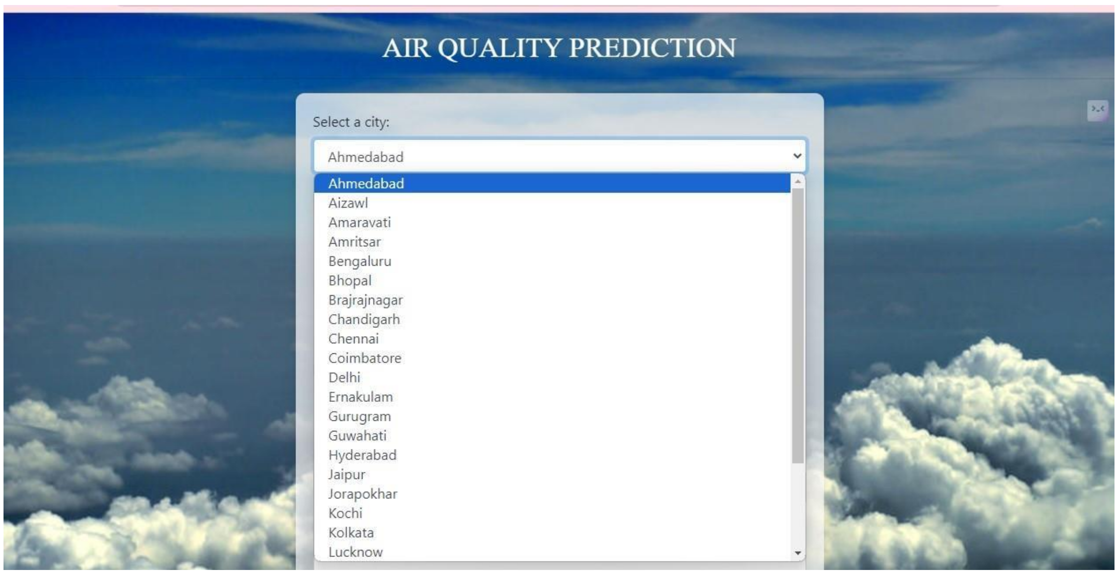


Figure 9.1   Output Home Page for CatBoost Model

Once the code in visual studio code is run it will provide an url that helps us to redirect to this home page. Here we chosen the Cat Boost for prediction because it got better accuracy then all the above compared models.





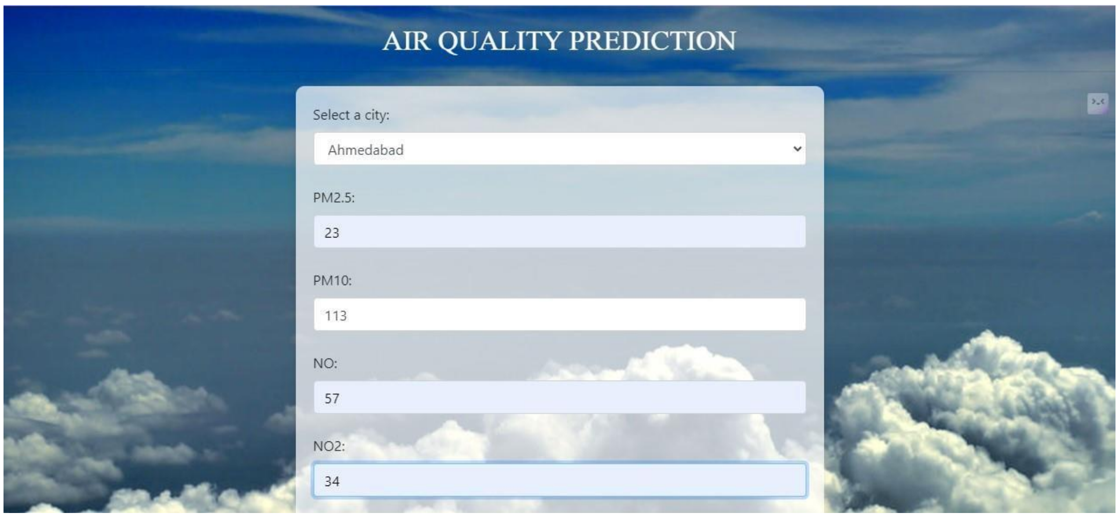

AIR QUALITY PREDICTION

Select a city:

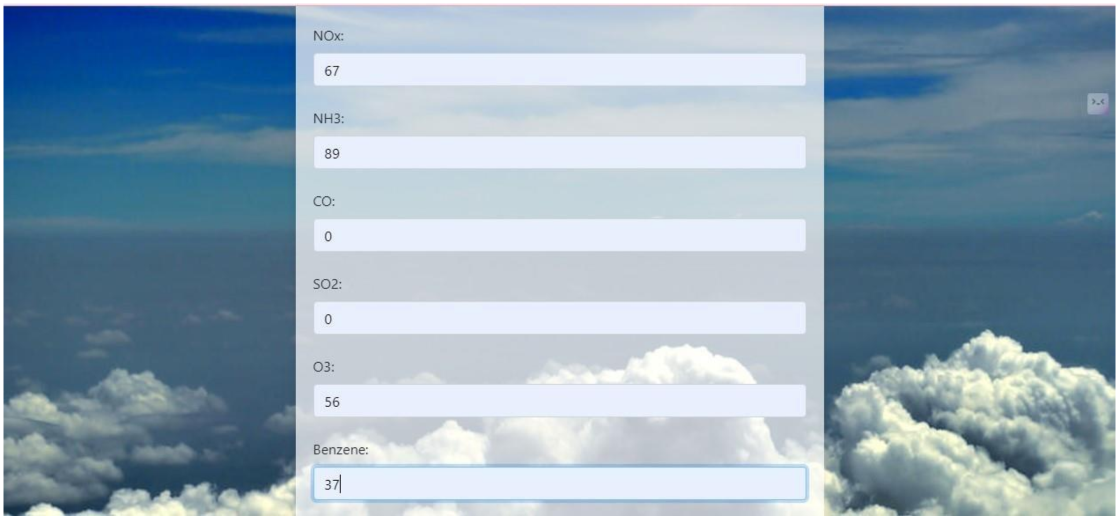
Ahmedabad

Ahmedabad
Aizawl
Amaravati
Amritsar
Bengaluru
Bhopal
Brajrajnagar
Chandigarh
Chennai
Coimbatore
Delhi
Ernakulam
Gurugram
Guwahati
Hyderabad
Jaipur
Jorapokhar
Kochi
Kolkata
Lucknow

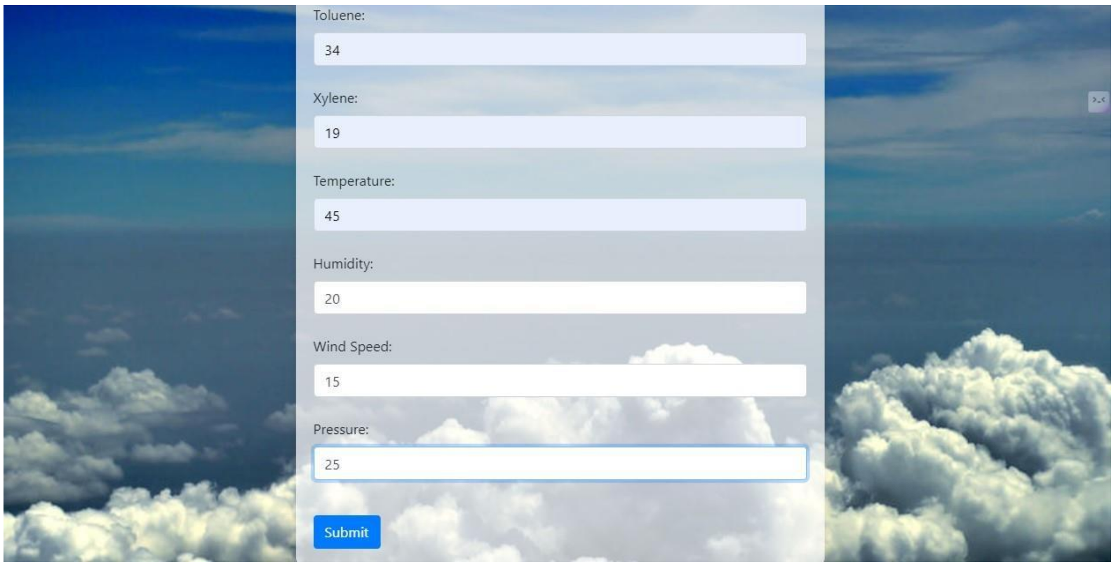
This image will shows the cities available in the dataset , here we have used the cities like Ahmedabad, Aizawl, Amaravati, Bengaluru, Visakhapatnam, Hyderabad,… ....for predicting the air quality present in environment based the pollutants values.


AIR QUALITY PREDICTION

Select a city:	Ahmedabad
PM2.5:	23
PM10:	113
NO:	57
NO2:	34

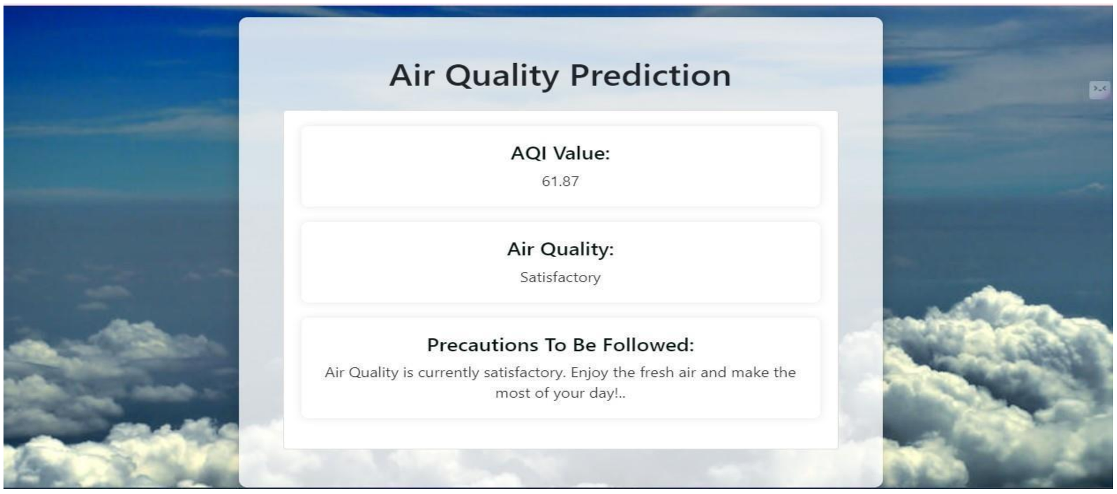


NOx:	67
NH3:	89
CO:	0
SO2:	0
O3:	56
Benzene:	37



Toluene:	34
Xylene:	19
Temperature:	45
Humidity:	20
Wind Speed:	15
Pressure:	25

This Page describes Providing the input to the System for the final prediction. On this page, you can input your city to retrieve personalized air quality predictions tailored to your area. Additionally, you can select specific pollutants or environmental factors of interest, such as PM2.5, ozone, nitrogen dioxide, or carbon monoxide, to further customize your analysis.



Air Quality Prediction

AQI Value:
61.87

Air Quality:
Satisfactory

Precautions To Be Followed:
Air Quality is currently satisfactory. Enjoy the fresh air and make the most of your day!..

Figure 9.2 Final Result Predicted Page for CatBoost Model

In the above Figure 9.2 It showing the AQI value, Air Quality and Precaution that we need to take into account for survival of human life.

In the final result of the project, the Air Quality Index (AQI) serves as a pivotal indicator, offering a comprehensive assessment of the air quality based on various pollutants present in the atmosphere. This index categorizes air quality into different levels, ranging from "good" to "severe," each representing varying degrees of health risks to the population. With this information, individuals, communities, and policymakers can make informed decisions regarding outdoor activities, public health measures, and environmental policies. For instance, during periods of "good" air quality, outdoor activities can be encouraged, while during "severe" conditions, precautions such as reducing outdoor exposure, particularly for vulnerable groups like children and the elderly, can be advised.

10 Conclusion

In this present investigation, we have discussed about different models and proposed best fit model that can be employed to get good prediction of air quality by using air quality parameters.

Our machine learning project aimed at predicting air quality, both CatBoost and Light Gradient Boost algorithms offer strong, accurate and efficient modeling. CatBoost has the ability to handle categorical variables without extensive preprocessing makes it particularly well-suited for datasets with diverse feature types. Light gradient boost is speed and scalable which makes it a convincing choice for large-scale datasets commonly encountered in air quality prediction projects.

11 Future Work

For future work recommendations, we explore diverse machine-learning approaches for predicting air quality and air pollution in smart cities. Additionally, Development of IoT devices for air quality and also this work can be further expanded to develop an app to predict air quality. This endeavor provides valuable insight into identifying air quality levels and contributes to more effective air quality management approaches.

In addition to that we need to expand sensors development, conducting feature importance analysis and quality in predictions for essential steps. Collaboration with policymakers and urban planners to build predictive models into decision-making processes and analysing long-term trends in air quality data can further enhance the effectiveness of air quality management strategies in smart cities.

12 Bibliography

[1] S. Ameer, M. A. Shah, A. Khan, H. Song, C. Maple, S. U. Islam, and M. N. Asghar,‘‘Comparativeanalysisofmachinelearningtechniquesforpredicting air quality in smart cities,’’ IEEE Access, vol. 7, pp. 128325–128338, 2019.

[2] M. Batty, K. W. Axhausen, F. Giannotti, A. Pozdnoukhov, A. Bazzani, M.Wachowicz,G.Ouzounis,andY.Portugali,‘‘Smart cities of the future,” Eur. Phys. J. Special Topics, vol. 214, no. 1, pp. 481–518, Nov. 2012.

[3] Shorouq Al-eidi Yahya Tashtoush 1,fathi Amsaad 2,omardarwish 4,ali Alqahtani 3,(senior Member, IEEE), 5, And Niveshitha Niveshitha2, (Graduate Student Member, IEEE), “Comparative Analysis Study for Air Quality Prediction in Smart Cities Using Regression Techniques,” Received 14 September 2023, accepted 29 September 2023, date of publication 10 October 2023, date of current version 23 October 2023.

[4] Yuan, Y., Wang, Y., & Hao, Z. (2021). Prediction of air quality index in smart cities based on machine learning.Kaggle datasets: Websites like Kaggle often host datasets related to air quality, which can be used for machine learning projects.

[5] K. Nandini and G. Fathima, ‘‘Urban air quality analysis and prediction using machine learning,’’ in Proc. 1st Int. Conf. Adv. Technol. Intell. Control, Environment, Comput. Commun. Eng. (ICATIECE), Mar. 2019, pp. 98–102.

[6] G.OliveriConti,B.Heibati,I.Kloog,M.Fiore,andM.Ferrante,‘‘Areview of AirQ models and their applications for forecasting the air pollution health outcomes,’’ Environmental Science and Pollution Research, vol. 24, no. 7, pp. 6426–6445, Mar. 2017.

[7] G. R. Kingsy, R. Manimegalai, D. M. S. Geetha, S. Rajathi, K. Usha, and B. N. Raabiathul, ‘‘Air pollution analysis using enhanced K-means clustering algorithm for real time sensor data,’’ in Proc. IEEE Region 10 Conf. (TENCON), Nov. 2016, pp. 1945–1949.