Air Quality Index Forecasting Via Genetic Algorithm-Based Improved Extreme Learning Machine

A Mini Project Report submitted to
JNTU Hyderabad in partial fulfillment
Of the requirements for the award of the degree

BACHELOR OF TECHNOLOGY

In

COMPUTER SCIENCE AND ENGINEERING

Submitted by

ANKIREDDY SRI UHA VARSHA	**21RG1A0503**
PANNALA BHAVANA	**21RG1A0542**
RIKKALA AKSHITHA	**21RG1A0548**
UBBARA RUSHMITHA	**21RG1A0560**

Under the Guidance of

Mrs. Avyaktha.E
M. Tech
Assistant Professor

DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING
MALLA REDDY COLLEGE OF ENGINEERING FOR WOMEN
An UGC Autonomous Institution
Approved by AICTE New Delhi and Affiliated to JNTUH
Maisammaguda, Medchal (Dist), Hyderabad -500100, Telangana.
OCTOBER 2024

DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING

MALLA REDDY COLLEGE OF ENGINEERING FOR WOMEN

An UGC Autonomous Institution

Approved by AICTE New Delhi and Affiliated to JNTUH

Maisammaguda, Medchal (Dist), Hyderabad -500100, Telangana.

OCTOBER 2024

CERTIFICATE

This is to certify that the Mini project entitled **"AIR QUALITY INDEX FORECASTING VIA GENETIC ALGORITHM BASED IMPROVED EXTREME LEARNING MACHINE"** has been submitted by **ANKIREDDY SRI UHA VARSHA (21RG1A0503), PANNALA BHAVANA (21RG1A0542), RIKKALA AKSHITHA (21RG1A0548) and UBBARA RUSHMITHA (21RG1A0560)** in partial fulfillment of the requirements for the award of **BACHELOR OF TECHNOLOGY** in **COMPUTER SCIENCE & ENGINEERING**. This record of bonafide work carried out by them under my guidance and supervision. ***The result embodied in this mini project report has not been submitted to any other University or Institute for the award of any degree.***

Mrs. Avyaktha.E	**Mrs. K. SHEETAL**
Assistant Professor	Head of the Department
Project Guide	

External Examiner

ACKNOWLEDGEMENT

The Mini Project work carried out by our team in the Department of Computer Science and Engineering, Malla Reddy College of Engineering for Women, Hyderabad. ***This work is original and has not been submitted in part or full for any degree or diploma of any other university.***

We wish to acknowledge our sincere thanks to our project guide **Mrs. Avyaktha.E,** Assistant Professor, Computer Science & Engineering for formulation of the problem, analysis, guidance and her continuous supervision during the course of work.

We acknowledge our sincere thanks to **Dr.Kanaka Durga Returi**, Principal & Professor of Department of Computer Science and Engineering and **Mrs. Sheetal Kulkarni,** Head of the Department, MRCEW and all our faculties of CSE Department for their kind cooperation in making this Mini Project work a success.

We extend our gratitude to **Sri. Ch. Malla Reddy**, Founder Chairman and **Sri. Ch. Mahender Reddy,** Secretary, **Dr. Vaka Murali Mohan,** Director for their kind cooperation in providing the infrastructure for completion of our Mini Project.

We acknowledge our special thanks to the entire teaching faculty and non-teaching staff members of the Computer Science & Engineering Department for their support in making this project work a success.

ANKIREDDY SRI UHA VARSHA	**21RG1A0503**	**-------------------**
PANNALA BHAVANA	**21RG1A0542**	**-------------------**
RIKKALA AKSHITHA	**21RG1A0548**	**-------------------**
UBBARA RUSHMITHA	**21RG1A0560**	**-------------------**

INDEX

LIST OF FIGURES

ABSTRACT

Air quality has always been one of the most important environmental concerns for the general public and society. Using machine learning algorithms for Air Quality Index (AQI) prediction is helpful for the analysis of future air quality trends from a macro perspective. When conventionally using a single machine learning model to predict air quality, it is challenging to achieve a good prediction outcome under various AQI fluctuation trends. In order to effectively address this problem, a genetic algorithm-based improved extreme learning machine (GA-KELM) prediction method is enhanced. First, a kernel method is introduced to produce the kernel matrix which replaces the output matrix of the hidden layer. To address the issue of the conventional limit learning machine where the number of hidden nodes and the random generation of thresholds and weights lead to the degradation of the network learning ability, a genetic algorithm is then used to optimize the number of hidden nodes and layers of the kernel limit learning machine. The thresholds, the weights, and the root mean square error are used to define the fitness function. Finally, the least squares method is applied to compute the output weights of the model. Genetic algorithms are able to find the optimal solution in the search space and gradually improve the performance of the model through an iterative optimization process. In order to verify the predictive ability of GA-KELM, based on the collected basic data of long-term air quality forecast at a monitoring point in a city in China, the optimized kernel extreme learning machine is applied to predict air quality (SO2, NO2, PM10, CO, O3, PM2.5 concentration and AQI), with comparative experiments based CMAQ (Community Multi scale Air Quality), SVM (Support Vector Machines) and DBN-BP (Deep Belief Networks with Back-Propagation). The results show that the proposed model trains faster and makes more accurate predictions.

CHAPTER-1: SYSTEM ANALYSIS

1.1 EXISTING SYSTEM

- The existing system typically uses a single machine learning model, such as traditional neural networks, for predicting air quality.
- Some of the prominent models referenced include the Community Multiscale Air Quality (CMAQ) modeling system, Support Vector Regression (SVR), and Deep Belief Network with Back-Propagation (DBN-BP).

1.1.1 Disadvantages of Existing System

- **Single Model Limitations:** Using a single machine learning model often results in suboptimal performance due to the varied and fluctuating nature of AQI data.
- **Slow Learning:** Traditional neural networks can suffer from slow learning processes.
- **Local Minima:** Neural networks tend to get stuck in local minima, which affects prediction accuracy.
- **Complex Training Process:** Training neural networks can be complex and time-consuming.
- **Random Parameter Selection:** For methods like Extreme Learning Machine (ELM), the number of hidden nodes and the random generation of thresholds and weights lead to a degradation in the network's learning ability and prediction accuracy.

1.2 PROPOSED SYSTEM

- The proposed system is a Genetic Algorithm-based Improved Extreme Learning Machine (GA-KELM).
- This system introduces a kernel method to generate a kernel matrix that replaces the output matrix of the hidden layer in the ELM.
- A genetic algorithm (GA) is used to optimize the number of hidden nodes, the thresholds, and the weights of the ELM.
- The Root Mean Square Error (RMSE) is employed as the fitness function for the GA.
- The Least Squares Method is applied to compute the output weights of the model.

1.2.1 Advantages of Proposed System

- **Improved Prediction Accuracy:** The use of a kernel method in ELM and optimization via GA enhances the prediction accuracy by addressing the random parameter selection issue.
- **Optimized Network Structure:** GA optimizes the number of hidden nodes and layers, as well as the thresholds and weights, leading to a more effective and efficient learning process.
- **Faster Training:** The proposed GA-KELM trains faster compared to traditional methods.
- **Enhanced Stability:** The optimization process via GA improves the stability of the results, reducing variability and enhancing reliability.
- **Iterative Improvement:** GA iteratively finds the optimal solution in the search space, progressively improving model performance.
- **Comparative Superiority:** Experiments demonstrate that GA-KELM outperforms existing models like CMAQ, SVR, and DBN-BP in terms of accuracy and training speed.

1.3 INTRODUCTION

Air pollution is a prevalent environmental problem in the twenty-first century. In light of the rapid industrialization and urbanization, air pollution is getting worse, which greatly affects our living environment and health [1]. Li et al. came to the conclusion that outdoor physical activity poses numerous health risks due to ambient air pollution in China. [2], [3]. According to the Chinese Ambient Air Quality Standards (GB3095-2012), there are six conventional air pollutants used to measure air quality: sulfur dioxide (SO2), nitrogen dioxide (NO2), particulate matter with a particle size less than 10 microns (PM10), particulate matter with a particle size less than 2.5 microns (PM2.5), ozone (O3), and carbon monoxide (CO) [4], [5], [6]. These pollutants have adverse effects on human health. The International Energy Agency estimates that air pollution causes 6.5 million premature deaths per year, while long-term exposure to pollutants, such as fine particles (e.g.,PM2.5) or traffic-related pollutants, is linked to higher rates of lung cancer, coronary heart disease, and other illnesses [7], [8]. Therefore, studies on air quality prediction are particularly important and are considered a key factor for environmental protection. In order to more comprehensively assess the health effects of air pollution, numerous air quality monitoring stations have been set up in major cities. Air quality predictions can be made based on the data collected from these stations. Air quality monitoring, modeling, and accurate predictions are important for having a clear understanding of future pollution levels and their associated health risks. Recently, the inherent property of machine learning algorithms to automatically learn features at multiple levels of abstraction has become increasingly important in providing solutions to this challenging task [9], [10]. However, the model only forecasts PM10 and SO2 levels, and it is also challenging to obtain measurement values needed to construct the dataset [11]. Wu Q. et al. proposed an optimal-hybrid model for daily AQI prediction considering air pollutant factors, with the model's inputs being the six atmospheric pollutants. However, neural networks typically struggle with slow learning, a tendency to fall into local minima, and a complex network training process. Based on the generalized inverse matrix theory, Huang et al. proposed an extreme learning machine (ELM) algorithm with a feedforward neural network that includes a single hidden layer, such that the problems of conventional neural network algorithms are circumvented. The ELM algorithm used to predict the AQI outperformed neural networks in terms of parameter selection, training speed, and prediction accuracy [12]. However, the parameters of the hidden layer nodes and the number of nodes in the test hidden layer are selected at random, which puts the prediction accuracy to a great test. In order to solve the aforementioned problems, we propose to optimize the number of ELM hidden layer nodes, thresholds, and weights, along with an improved genetic algorithm (GA) that uses root mean square error (RMSE) as the fitness function, to obtain the optimal network structure for air quality prediction [14]. The number of hidden layer nodes is updated by continuous coding discretization, the input weights and hidden layer thresholds are updated by continuous coding, and the update thresholds and weights are selected with the number of updated layers to form a hierarchical control structure [15]. The proposed GA-based improved extreme learning machine (GA-KELM) algorithm is applied to air quality prediction, and its performance is compared with that of community multiscale air quality modeling system (CMAQ), support vector regression (SVR), and deep belief network-back propagation (DBN-BP). The results show that the accuracy of the proposed GA-KELM algorithm is reliable for air quality prediction [16]. In this study, an improved extreme learning machine model based on a genetic algorithm is designed and applied to AQI prediction. To verify the effectiveness of the model, we conducted tests on three real-world datasets. The results confirmed that the proposed method has superior performance and outperforms some advanced methods currently in use. The main contributions of this paper are: (1) modifying the ELM activation function or using the kernel function to improve the prediction accuracy, (2) optimizing the ELM using GA to improve the stability of the results and further enhance the prediction accuracy, and (3) obtaining the correlation analysis results of atmospheric environmental quality prediction parameters by comprehensively considering each relevant factor in line with the actual situation. The remainder of this paper is organized as follows. Section II presents related work. Section III describes ELM and the proposed GA-KELM, and illustrates the improvements using the model. Section IV discusses experimental results where GA-KELM is compared with several other methods in terms of prediction results. The last section concludes the entire work and presents directions for future research.

Agarwal and Sahu, conducted air quality prediction studies by employing statistical models. Lary et al.

D. J. Lary, T. Lary, and B. Sattler, combined remote sensing and meteorological data with groundbased PM2.5 observations. Zheng et al.

Y. Zheng, X. Yi, M. Li, R. Li, Z. Shan, E. Chang, and T. Li, proposed a hybrid prediction method that combines a linear regression- based temporal prediction method with an ANN-based spatial prediction method for pollutant concentrations. Zheng et al.

Y. Zheng, L. Capra, O. Wolfson, and H. Yang, used a data-based approach for the next 48 hours of PM2.5 prediction, implementing a prediction model based on linear regression and neural network. They combined meteorological data, weather forecast data, and air quality data from monitoring stations.

T. S. Rajput and N. Sharma, used a multiple regression model to represent the changes in air quality index (AQI), considering ambient temperature, relative humidity, and barometric pressure as the main parameters in the regression model for AQI calculation.

B. Liu, S. Yan, J. Li, G. Qu, Y. Li, J. Lang, and R. Gu, States that these classical methods and models all have the advantages of simple algorithms, easy processing, and acceptable prediction results. However, obtaining precise and specific air quality prediction values remains challenging.

K. Elbaz, I. Hoteit, W. M. Shaban, and S.-L. Shen, proposed a novel deep learning approach that extracts high-level abstractions to capture the spatiotemporal characteristics of NEOM city in Saudi Arabia at hourly and daily intervals. Campbell et al.

P. C. Campbell, Y. Tang, P. Lee, B. Baker, D. Tong, R. Saylor, A. Stein, J. Huang, H.-C. Huang, E. Strobach, J. McQueen, L. Pan, I. Stajner, J. Sims, J. Tirado-Delgado, Y. Jung, F. Yang, T. L. Spero, and R. C. Gilliam, described the development of FV3GFSv16 coupled with the ‘‘stateof-the-art’’ CMAQ model version 5.3.1. Jin et al.

X.-B. Jin, Z.-Y. Wang, W.-T. Gong, J.-L. Kong, Y.-T. Bai, T.-L. Su, H.-J. Ma, and P. Chakrabarti, proposed an interpretable variational Bayesian deep learning model with self-filtering capability for PM2.5 prediction information, which effectively improves prediction accuracy. Zhou et al.

Z. Zhou, W. Deng, Z. Zhu, Y. Hu, J. Ji, Y. Wang, J. Du, and X. Liu, proposed a method based on an improved Grasshopper optimization algorithm to classify the color difference of dyed fabrics using kernel extreme learning machine. In this study, the classification of color differences in dyed fabric images is performed using the kernel limit learning machine, and the kernel function parameters are optimized by the improved Grasshopper optimization algorithm to achieve color difference classification of dyed fabric images. Xue et al.

Y. Xue, Q. Wu, and H. Sun, proposed a GA-based air quality prediction model to optimize the parameters of the weighted extreme learning machine (WELM). Despite the progress made by the aforementioned methods, they also exhibit limitations; their training efficiency is relatively low, and deep learning algorithms are not yet fully mature. These challenges present greater obstacles for the application of deep learning, necessitating improvements to existing models, the development of new models, and the enhancement of their predictive capabilities.

J. Ma, Y. Ding, J. C. P. Cheng, F. Jiang, and Z. Wan, The use of statistical or numerical forecasting techniques is subject to several limitations. Neural networks are widely used because of their unique associative abilities, memory, and distinctive learning.

S. Maji, S. Ghosh, and S. Ahmed, Given the highly nonlinear nature of AQI changes and the strong generalization and nonlinear characterization abilities of neural networks, the nuclear limit learning machine neural network model, also known as kernel extreme learning machine (KELM), is employed to investigate air quality prediction using a real dataset. The weights and threshold values of KELM are optimized using a genetic optimization algorithm.

M. Wang, H. Chen, B. Yang, X. Zhao, L. Hu, Z. Cai, H. Huang, and C. Tong, states that Air quality monitoring stations measure air temperature, wind direction, atmospheric pressure, relative humidity, wind speed and other meteorological parameters, as well as air pollutant concentrations.

M. A. E. Aziz, A. A. Ewees, and A. E. Hassanien, states that Air quality prediction is also challenging due to the rapid changes in pollutant emission and weather conditions. Numerous variables, such as wind speed, temperature, humidity, and pollutants themselves, are highly nonlinear, dynamic, and have inherent interdependencies, making it more challenging to accurately predict air quality at a specific time and place. Therefore, it is essential to figure out how to deal with these factors and exploit them from multivariate time-series data related to air quality. To meet our research goals, this paper proposes GA-KELM, a method for air quality prediction based on an improved extreme learning machine, which in turn is based on an improved GA.

Y. Yang, Y. Wang, and X. Yuan, states that Aiming at the problem of network instability caused by the randomly generated input layer weights and hidden layer thresholds of KELM, a GA is used to optimize the KELM weights and thresholds, thereby improving the model’s performance in terms of prediction accuracy, which is the main objective of this algorithm. In each iteration of the GA, a new offspring population is generated by selection, crossover, and mutation, and the individual with good fitness value is selected. The GA stops iterating when the stopping criteria are satisfied. The GA is used to determine the optimal weights and threshold values, which overcomes the instability of KELM and reduces prediction errors, thus resulting in a more reliable prediction model and improved air quality prediction accuracy. ELM was first proposed by Huang. It is characterized by its fast training and high training accuracy. Feed forward neural networks are mainly based on the gradient descent method.

F. Jiang, J. He, and Z. Zeng, states that Inspired by SVM, the kernel method is introduced into ELM, namely KELM. KELM maps linearly inseparable patterns to a high-dimensional feature space to achieve linear separability and improve the accuracy and robustness of the model.

Q. Li, H. Chen, H. Huang, X. Zhao, Z. Cai, C. Tong, W. Liu, and X. Tian, states that In conventional ELM, input weights and hidden layer neurons are assigned at random, and output weights are then obtained on the basis of input weights and hidden layer neurons. Thus, human experiments are essentially important in customary training, which often results in inaccurate predictions.

G. Huang, H. Zhou, X. Ding, and R. Zhang, states that All the spatiotemporal relationships that dominate the input data are not entirely helpful in predicting the output. Each feature has a level of association with the predicted output. The goal of GA-KELM is to identify spatiotemporal features and the degree of their association with the predicted output learned through its weight matrix.

CHAPTER-3: SYSTEMDESIGN

3.1 System Modules

1. **Upload Dataset**

   Using this module will upload dataset details and then application read and display the dataset values and uses this data to predict the air quality values.

2. **Preprocess Dataset**

   Using this module we will preprocess dataset and then remove missing values and then convert all non-numeric data into numeric data and then shuffle and then split dataset into train and test where application using 80% dataset for training and 20% for testing.

3. **Run Logistic Regression**

   Using this module we will input 80% dataset to Logistic Regression to train a model and 20% test data will be applied on trained model to calculate prediction accuracy.

4. **Run Random Forest Algorithm**

   Using this module we will input 80% dataset to Random Forest to train a model and 20% test data will be applied on trained model to calculate prediction accuracy.

5. **Run Decision Tree Algorithm**

   Using this module we will input 80% dataset to Decision Tree to train a model and 20% test data will be applied on trained model to calculate prediction accuracy.

6. **Run SVM Algorithm**

   Using this module we will input 80% dataset to SVM to train a model and 20% test data will be applied on trained model to calculate prediction accuracy.

7. **Predict AQI values from Test Data**

   Using this module we will upload test data and then random forest will predict whether the quality of air is suitable for breathing or not.

3.2 System Architecture

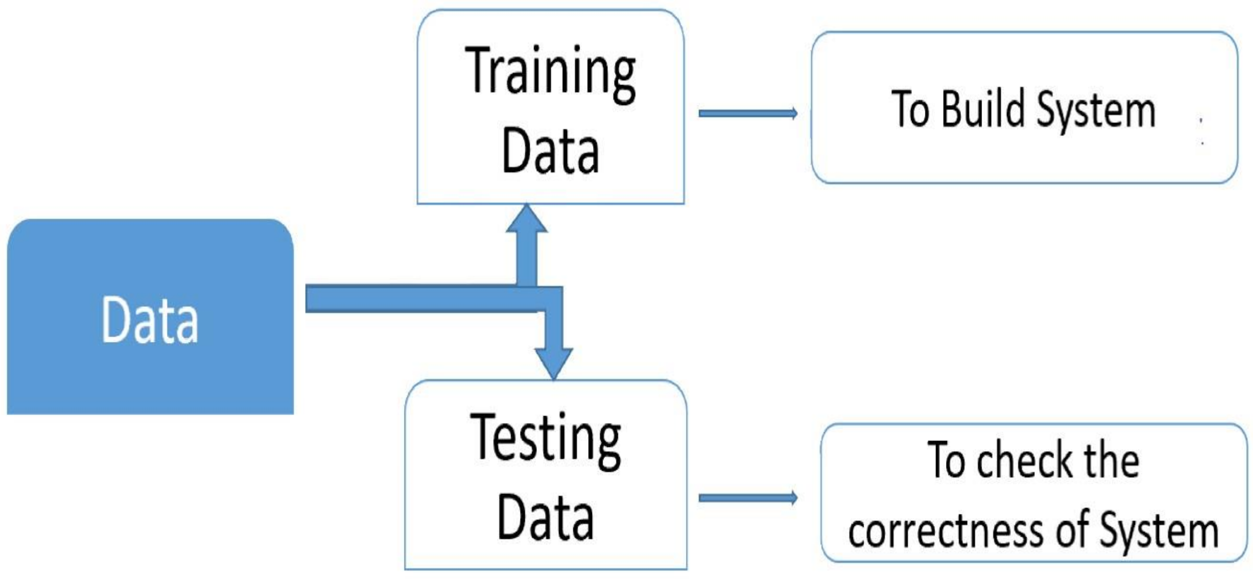


Fig.3.1: System Architecture

3.3 System Requirements

3.3.1 Hardware Requirements

• System	: Intel I-3, 5, 7 Processor.
• Hard Disk	: 512 GB.
• Monitor	: 14’ Color Monitor.
• Mouse	: Optical Mouse.
• RAM	: 8 Gb.

3.3.2 Software Requirements:

• Operating system	: Windows 7,8,10 Ultimate, Linux, Mac.
• Front-End	: Python.
• Coding Language	: Python.
• Software Environment	: Anaconda (jupyter or spyder).

3.4 UML Diagrams

3.4.1 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.

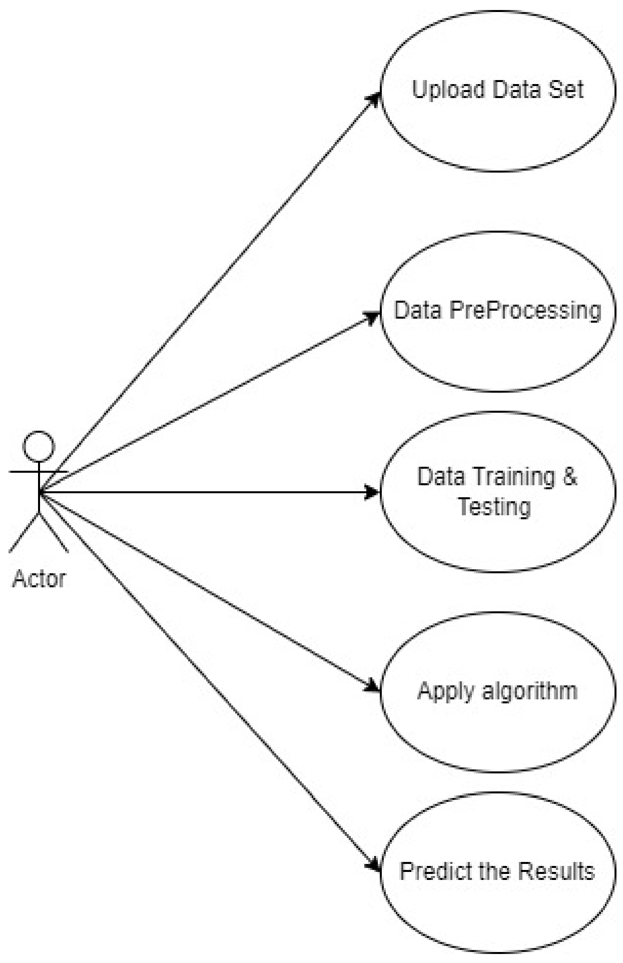


Fig. 3.2 : Use Case Diagram

3.4.2 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.

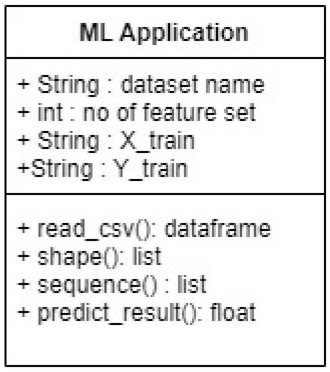


Fig. 3.3: Class Diagram

3.4.3 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.

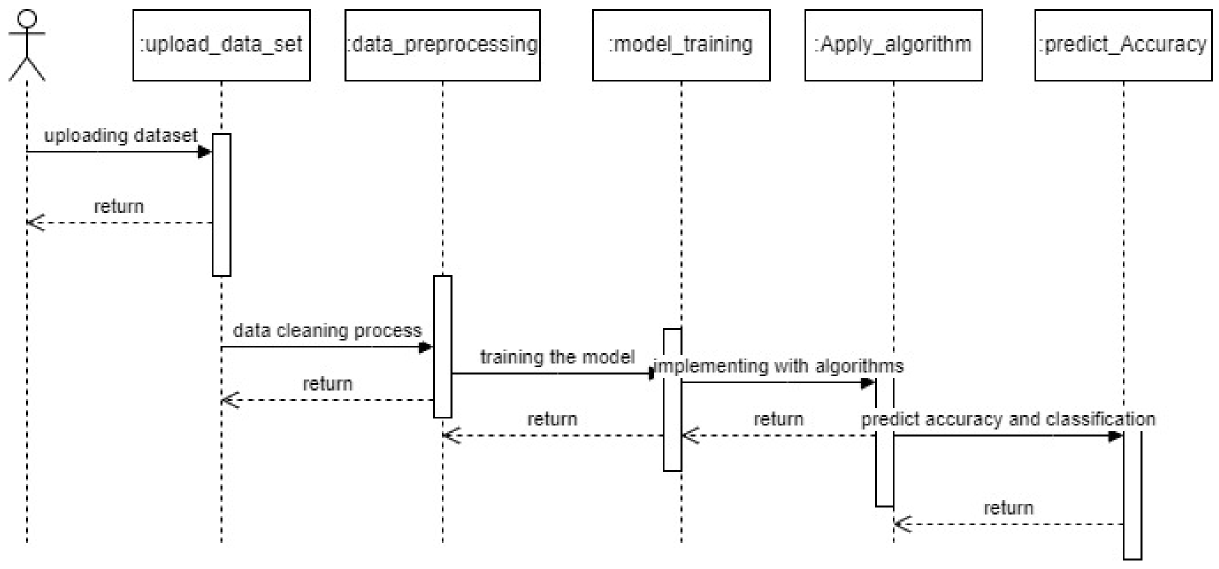


Fig. 3.4 : Sequence Diagram

3.4.4 Flow Chart Diagram

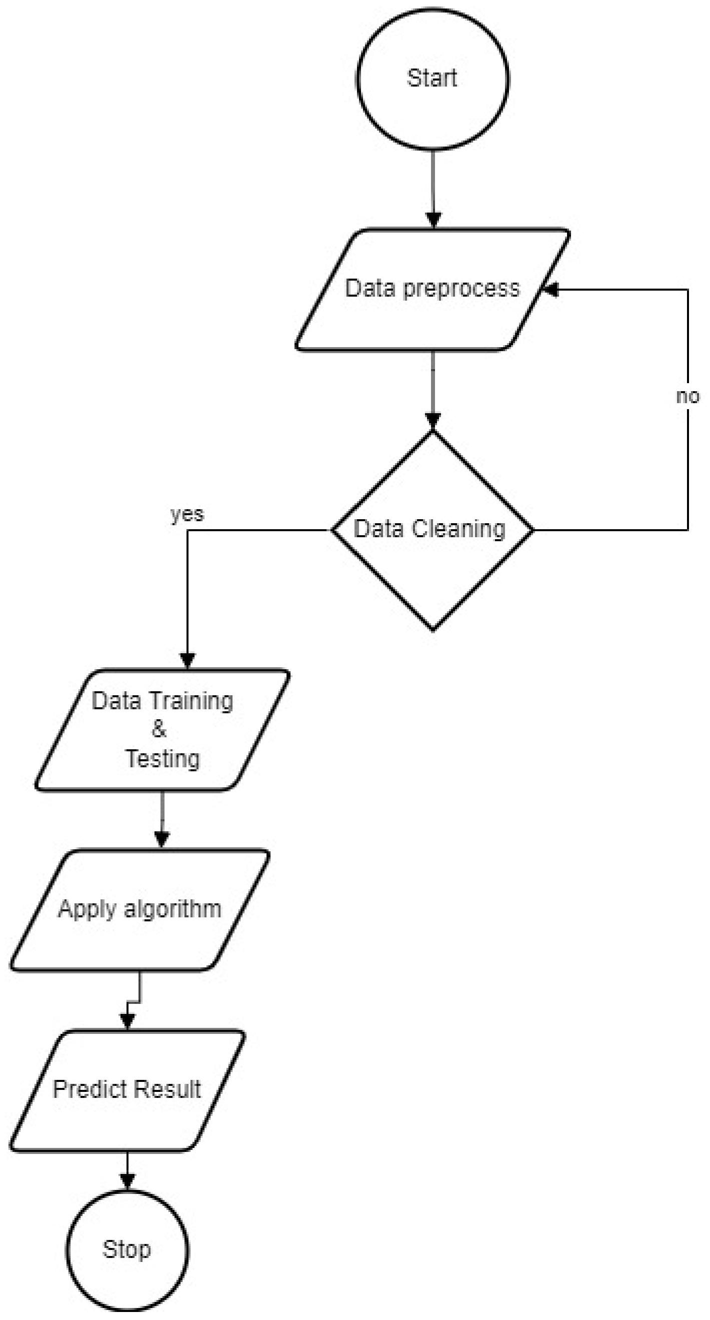


Fig.3.5 : Flow Chart Diagram

CHAPTER-4: INPUT AND OUTPU DESIGN

4.1 INPUT DESIGN

Input design is a part of overall system design. The main objective during the input design is as given below:

- To produce a cost-effective method of input.
- To achieve the highest possible level of accuracy.
- To ensure that the input is acceptable and understood by the user.

INPUT STAGES

The main input stages can be listed as below:

- Data recording
- Data transcription
- Data conversion
- Data verification
- Data control
- Data transmission
- Data validation
- Data correction

INPUT TYPES

It is necessary to determine the various types of inputs. Inputs can be categorized as follows:

- External inputs, which are prime inputs for the system.
- Internal inputs, which are user communications with the system.
- Operational, which are computer department's communications to the system?
- Interactive, which are inputs entered during a dialogue.

INPUT MEDIA

At this stage choice has to be made about the input media. To conclude about the input media consideration has to be given to;

- Type of input
- Flexibility of format
- Speed

- Accuracy
- Verification methods
- Rejection rates
- Ease of correction
- Storage and handling requirements
- Security
- Easy to use
- Portability

Keeping in view the above description of the input types and input media, it can be said that most of the inputs are of the form of internal and interactive. As
Input data is to be the directly keyed in by the user, the keyboard can be considered to be the most suitable input device.

ERROR AVOIDANCE

At this stage care is to be taken to ensure that input data remains accurate form the stage at which it is recorded up to the stage in which the data is accepted by the system. This can be achieved only by means of careful control each time the data is handled.

ERROR DETECTION

Even though every effort is make to avoid the occurrence of errors, still a small proportion of errors is always likely to occur, these types of errors can be discovered by using validations to check the input data.

DATA VALIDATION

Procedures are designed to detect errors in data at a lower level of detail. Data validations have been included in the system in almost every area here there is a possibility for the user to commit errors. The system will not accept invalid data. Whenever an invalid data is keyed in, the system immediately prompts the user and the user has to again key in the data and the system will accept the data only if the data is correct. Validations have been included where necessary.
The system is designed to be a user friendly one. In other words the system has been designed to communicate effectively with the user. The system has been designed with popup menus.

USER INTERFACE DESIGN

It is essential to consult the system users and discuss their needs while designing the user interface:

USER INTERFACE SYSTEMS CAN BE BROADLY CLASIFIED AS:

- User initiated interface the user is in charge, controlling the progress of the user/computer dialogue. In the computer-initiated interface, the computer selects the next stage in the interaction.
- Computer initiated interfaces

In the computer-initiated interfaces the computer guides the progress of the user/computer dialogue. Information is displayed and the user response of the computer takes action or displays further information.

USER INITIATED INTERGFACES

User initiated interfaces fall into two approximate classes:

- Command driven interfaces: In this type of interface the user inputs commands or queries which are interpreted by the computer.
- Forms oriented interface: The user calls up an image of the form to his/her screen and fills in the form. The forms-oriented interface is chosen because it is the best choice.

COMPUTER-INITIATED INTERFACES

The following computer – initiated interfaces were used:

- The menu system for the user is presented with a list of alternatives and the user chooses one; of alternatives.
- Questions – answer type dialog system where the computer asks question and takes action based on the basis of the users reply.

Right from the start the system is going to be menu driven, the opening menu displays the available options

ERROR MESSAGE DESIGN

The design of error messages is an important part of the user interface design. As user is bound to commit some errors or other while designing a system the system should be designed to be helpful by providing the user with information regarding the error he/she has committed. This application must be able to produce output at different modules for different inputs.

PERFORMANCE REQUIREMENTS

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

- The system should be able to interface with the existing system
- The system should be better than the existing system and accurate
- The existing system is completely dependent on the user to perform all the duties.

4.2 OUTPUT DESIGN

Outputs from computer systems are required primarily to communicate the results of processing to users. The various types of outputs in general are:

- External Outputs, whose destination is outside the organization
- Internal Outputs whose destination is within organization and they are the
- Operational outputs whose use is purely within the computer department.
- Interface outputs, which involve the user in communicating directly.

OUTPUT DEFINITION

The outputs should be defined in terms of the following points:

- Type of the output
- Content of the output
- Format of the output
- Location of the output
- Frequency of the output
- Volume of the output
- Sequence of the output

It is not always desirable to print or display data as it is held on a computer. It should be decided as which form of the output is the most suitable.

CHAPTER-5: SOFTWARE ENVIRONMENT

5.1 What is Python?

Below are some facts about Python.

- Python is currently the most widely used multi-purpose, high-level programming language.
- Python allows programming in Object-Oriented and Procedural paradigms. Python programs generally are smaller than other programming languages like Java.
- Programmers have to type relatively less and indentation requirement of the language, makes them readable all the time.
- Python language is being used by almost all tech-giant companies like – Google, Amazon, Facebook, Instagram, Dropbox, Uber… etc.

The biggest strength of Python is huge collection of standard library which can be used for the following –

- Machine Learning
- GUI Applications (like Kivy, Tkinter, PyQt etc.)
- Web frameworks like Django (used by YouTube, Instagram, Dropbox)
- Image processing (like Opencv, Pillow)
- Web scraping (like Scrapy, BeautifulSoup, Selenium)
- Test frameworks
- Multimedia

5.1.1 Advantages of Python

Let's see how Python dominates over other languages.

1. Extensive Libraries

Python downloads with an extensive library and it contain code for various purposes like regular expressions, documentation-generation, unit-testing, web browsers, threading, databases, CGI, email, image manipulation, and more. So, we don't have to write the complete code for that manually.

2. Extensible

Python can be extended to other languages. You can write some of your code in languages like C++ or C. This comes in handy, especially in projects.

3. Embeddable

Complimentary to extensibility, Python is embeddable as well. You can put your Python code in your source code of a different language, like C++. This lets us add scripting capabilities to our code in the other language.

4. Improved Productivity

The language's simplicity and extensive libraries render programmers more productive than languages like Java and C++ do. Also, the fact that you need to write less and get more things done.

5. IOT Opportunities

Since Python forms the basis of new platforms like Raspberry Pi, it finds the future bright for the Internet of Things. This is a way to connect the language with the real world.

6. Simple and Easy

When working with Java, you may have to create a class to print 'Hello World'. But in Python, just a print statement will do. It is also quite easy to learn, understand, and code. This is why when people pick up Python, they have a hard time adjusting to other more verbose languages like Java.

7. Readable

Because it is not such a verbose language, reading Python is much like reading English. This is the reason why it is so easy to learn, understand, and code. It also does not need curly braces to define blocks, and indentation is mandatory. This further aids the readability of the code.

8. Object-Oriented

This language supports both the procedural and object-oriented programming paradigms. While functions help us with code reusability, classes and objects let us model the real world. A class allows the encapsulation of data and functions into one.

9. Free and Open-Source

Python is freely available. But not only can you download Python for free, but you can also download its source code, make changes to it, and even distribute it. It downloads with an extensive collection of libraries to help you with your tasks.

10. Portable

When you code your project in a language like C++, you may need to make some changes to it if you want to run it on another platform. But it isn't the same with Python. Here, you need to code only once, and you can run it anywhere. This is called Write Once Run Anywhere (WORA). However, you need to be careful enough not to include any system-dependent features.

11. Interpreted

It is an interpreted language. Since statements are executed one by one, debugging is easier than in compiled languages.

Advantages of Python over Other Languages

1. Less Coding

Almost all of the tasks done in Python requires less coding when the same task is done in other languages. Python also has an awesome standard library support, so you don't have to search for any third-party libraries to get your job done. This is the reason that many people suggest learning Python to beginners.

2. Affordable

Python is free therefore individuals, small companies or big organizations can leverage the free available resources to build applications. Python is popular and widely used so it gives you better community support.

The 2019 Github annual survey showed us that Python has overtaken Java in the most popular programming language category.

3. Python is for everyone

Python code can run on any machine whether it is Linux, Mac or Windows. Programmers need to learn different languages for different jobs but with Python, you can professionally build web apps, perform data analysis and machine learning, automate things, do web scraping and also build games and powerful visualizations. It is an all-rounder programming language.

5.1.2 Disadvantages of Python

So far, we've seen why Python is a great choice for your project. But if you choose it, you should be aware of its consequences as well. Let's now see the downsides of choosing Python over another language.

1. Speed Limitations

We have seen that Python code is executed line by line. But since Python is interpreted, it often results in slow execution. This, however, isn't a problem unless speed is a focal point for the project. In other words, unless high speed is a requirement, the benefits offered by Python are enough to distract us from its speed limitations.

2. Weak in Mobile Computing and Browsers

While it serves as an excellent server-side language, Python is much rarely seen on the client-side. Besides that, it is rarely ever used to implement smartphone-based applications. One such application is called Carbonnelle. The reason it is not so famous despite the existence of Brython is that it isn't that secure.

3. Design Restrictions

As you know, Python is dynamically-typed. This means that you don't need to declare the type of variable while writing the code. It uses duck-typing. But wait, what's that? Well, it just means that if it looks like a duck, it must be a duck. While this is easy on the programmers during coding, it can raise run-time errors.

4. Underdeveloped Database Access Layers

Compared to more widely used technologies like JDBC (Java DataBase Connectivity) and ODBC (Open DataBase Connectivity), Python's database access layers are a bit underdeveloped. Consequently, it is less often applied in huge enterprises.

5. Simple

Its syntax is so simple that the verbosity of Java code seems unnecessary.

This was all about the Advantages and Disadvantages of Python Programming Language.

5.1.3 History of Python

What do the alphabet and the programming language Python have in common? Right, both start with ABC. If we are talking about ABC in the Python context, it's clear that the programming language ABC is meant. ABC is a general-purpose programming language and programming environment, which had been developed in the Netherlands, Amsterdam, at the CWI (Centrum Wiskunde &Informatica). The greatest achievement of ABC was to influence the design of Python. Python was conceptualized in the late 1980s. Guido van Rossum worked that time in a project at the CWI, called Amoeba, a distributed operating system. In an interview with Bill Venners[1], Guido van Rossum said: "In the early 1980s, I worked as an implementer on a team building a language called ABC at Centrum voor Wiskunde en Informatica (CWI). I don't know how well people know ABC's influence on Python. I try to mention ABC's influence because I'm indebted to everything I learned during that project and to the people who worked on it. "Later on in the same Interview, Guido van Rossum continued: "I remembered all my experience and some of my frustration with ABC. I decided to try to design a simple scripting language that possessed some of ABC's better properties, but without its problems. So I started typing. I created a simple virtual machine, a simple parser, and a simple runtime. I made my own version of the various ABC parts that I liked. I created a basic syntax, used indentation for statement grouping instead of curly braces or begin-end blocks, and developed a small number of powerful data types: a hash table (or dictionary, as we call it), a list, strings, and numbers."

5.1.4 Python Development Steps

Guido Van Rossum published the first version of Python code (version 0.9.0) at alt.sources in February 1991. This release included already exception handling, functions, and the core data types of list, dict, str and others.
Python version 1.0 was released in January 1994. The major new features included in this release were the functional programming tools lambda, map, filter and reduce, which Guido Van Rossum never liked. Six and a half years later in October 2000, Python 2.0 was introduced. This release included list comprehensions, a full garbage collector and it was supporting unicode. Python flourished for another 8 years in the versions 2.x before the next major release as Python 3.0 (also known as "Python 3000" and "Py3K") was released. Python 3 is not backwards compatible with Python 2.x. The emphasis in Python 3 had been on the removal of duplicate programming constructs and modules, thus fulfilling or coming close to fulfilling the 13th law of the Zen of Python: "There should be one -- and preferably only one -- obvious way to do it."Some changes in Python 7.3:

- Print is now a function.
- Views and iterators instead of lists
- The rules for ordering comparisons have been simplified. E.g., a heterogeneous list cannot be sorted, because all the elements of a list must be comparable to each other.
- There is only one integer type left, i.e., int. long is int as well.
- The division of two integers returns a float instead of an integer. "//" can be used to have the "old" behaviour.
- Text Vs. Data Instead of Unicode Vs. 8-bit

Purpose

We demonstrated that our approach enables successful segmentation of intra-retinal layers—

even with low-quality images containing speckle noise, low contrast, and different intensity ranges throughout—with the assistance of the ANIS feature.

Python

Python is an interpreted high-level programming language for general-purpose programming. Created by Guido van Rossum and first released in 1991, Python has a design philosophy that emphasizes code readability, notably using significant whitespace.

Python features a dynamic type system and automatic memory management. It supports multiple programming paradigms, including object-oriented, imperative, functional and procedural, and has a large and comprehensive standard library.

- Python is Interpreted − Python is processed at runtime by the interpreter. You do not need to compile your program before executing it. This is similar to PERL and PHP.

- Python is Interactive − you can actually sit at a Python prompt and interact with the interpreter directly to write your programs.

Python also acknowledges that speed of development is important. Readable and terse code is part of this, and so is access to powerful constructs that avoid tedious repetition of code. Maintainability also ties into this may be an all but useless metric, but it does say something about how much code you have to scan, read and/or understand to troubleshoot problems or tweak behaviors. This speed of development, the ease with which a programmer of other languages can pick up basic Python skills and the huge standard library is key to another area where Python excels. All its tools have been quick to implement, saved a lot of time, and several of them have later been patched and updated by people with no Python background - without breaking.

5.2 MODULES USED IN THIS PROJECT

1. NumPy

NumPy is a general-purpose array-processing package. It provides a high-performance multidimensional array object, and tools for working with these arrays. It is the fundamental package for scientific computing with Python. It contains various features including these important ones:

- A powerful N-dimensional array object

- Sophisticated (broadcasting) functions

- Tools for integrating C/C++ and Fortran code
- Useful linear algebra, Fourier transform, and random number capabilities

Besides its obvious scientific uses, NumPy can also be used as an efficient multi-dimensional container of generic data. Arbitrary datatypes can be defined using NumPy which allows NumPy to seamlessly and speedily integrate with a wide variety of databases.

2. Pandas

Pandas is an open-source Python Library providing high-performance data manipulation and analysis tool using its powerful data structures. Python was majorly used for data munging and preparation. It had very little contribution towards data analysis. Pandas solved this problem. Using Pandas, we can accomplish five typical steps in the processing and analysis of data, regardless of the origin of data load, prepare, manipulate, model, and analyze. Python with Pandas is used in a wide range of fields including academic and commercial domains including finance, economics, Statistics, analytics, etc.

3. Scikit – learn

Scikit-learn provides a range of supervised and unsupervised learning algorithms via a consistent interface in Python. It is licensed under a permissive simplified BSD license and is distributed under many Linux distributions, encouraging academic and commercial use.

5.3 MACHINE LEARNING

What is Machine Learning?

The study of machine learning certainly arose from research in this context, but in the data science application of machine learning methods, it's more helpful to think of machine learning as a means of building models of data.

Fundamentally, machine learning involves building mathematical models to help understand data. "Learning" enters the fray when we give these models tunable parameters that can be adapted to observed data; in this way the program can be considered to be "learning" from the data. Once these models have been fit to previously seen data, they can be used to predict and understand aspects of newly observed data. Understanding the problem setting in machine learning is essential to using these tools effectively, and so we will start with some broad categorizations of the types of approaches we'll discuss here.

Need for Machine Learning

Human beings, at this moment, are the most intelligent and advanced species on earth because they can think, evaluate, and solve complex problems. On the other side, AI is still in its initial stage and have not surpassed human intelligence in many aspects. Then the question is that what is the need to make machine learn? The most suitable reason for doing this is, “to make decisions, based on data, with efficiency and scale”.

Lately, organizations are investing heavily in newer technologies like Artificial Intelligence, Machine Learning and Deep Learning to get the key information from data to perform several real-world tasks and solve problems. We can call it data-driven decisions taken by machines, particularly to automate the process. These data-driven decisions can be used, instead of using programing logic, in the problems that cannot be programmed inherently.

Challenges in Machines Learning

While Machine Learning is rapidly evolving, making significant strides with cyber security and autonomous cars, this segment of AI as whole still has a long way to go. The reason behind is that ML has not been able to overcome number of challenges. The challenges that ML is facing currently are −

1. **Quality of data** − Having good-quality data for ML algorithms is one of the biggest challenges. Use of low-quality data leads to the problems related to data preprocessing and feature extraction.

2. **Time-Consuming task** − Another challenge faced by ML models is the consumption of time especially for data acquisition, feature extraction and retrieval.

3. **Lack of specialist persons** − As ML technology is still in its infancy stage, availability of expert resources is a tough job.

4. **No clear objective for formulating business problems** − Having no clear objective and well-defined goal for business problems is another key challenge for ML because this technology is not that mature yet.

5. **Issue of overfitting & underfitting** − If the model is overfitting or underfitting, it cannot be represented well for the problem.

6. **Curse of dimensionality** − Another challenge ML model faces is too many features of data points. This can be a real hindrance.

7. **Difficulty in deployment** − Complexity of the ML model makes it quite difficult to be deployed in real life.

Applications of Machines Learning

Machine Learning is the most rapidly growing technology and according to researchers we are in the golden year of AI and ML. It is used to solve many real-world complex problems which cannot be solved with traditional approach. Following are some real-world applications of ML –

- Emotion analysis
- Sentiment analysis
- Error detection and prevention
- Weather forecasting and prediction
- Stock market analysis and forecasting
- Speech synthesis
- Speech recognition
- Customer segmentation
- Object recognition
- Fraud detection
- Fraud prevention
- Recommendation of products to customer in online shopping

How to start learning ML?

This is a rough roadmap you can follow on your way to becoming an insanely talented Machine Learning Engineer. Of course, you can always modify the steps according to your needs to reach your desired end-goal!

Step 1 – Understand the Prerequisites

In case you are a genius, you could start ML directly but normally, there are some prerequisites that you need to know which include Linear Algebra, Multivariate Calculus, Statistics, and Python. And if you don't know these, never fear! You don't need a Ph.D. degree in these topics to get started but you do need a basic understanding.

(a) Learn Linear Algebra and Multivariate Calculus

Both Linear Algebra and Multivariate Calculus are important in Machine Learning. However, the extent to which you need them depends on your role as a data scientist. But if you want to focus on R&D in Machine Learning, then mastery of Linear Algebra and Multivariate Calculus is very important as you will have to implement many ML algorithms from scratch.

(b) Learn Statistics

Data plays a huge role in Machine Learning. In fact, around 80% of your time as an ML expert will be spent collecting and cleaning data. And statistics is a field that handles the collection, analysis, and presentation of data.
Some of the key concepts in statistics that are important are Statistical Significance, Probability Distributions, Hypothesis Testing, Regression, etc. Also, Bayesian Thinking is also a very important part of ML which deals with various concepts like Conditional Probability, Priors, and Posteriors, Maximum Likelihood, etc.

(c) Learn Python

Some people prefer to skip Linear Algebra, Multivariate Calculus and Statistics and learn them as they go along with trial and error. But the one thing that you absolutely cannot skip is Python! While there are other languages you can use for Machine Learning like R, Scala, etc. In fact, there are many Python libraries that are specifically useful for Artificial Intelligence and Machine Learning such as Keras, TensorFlow, Scikit-learn, etc.

Step 2 – Learn Various ML Concepts

Some of the basic concepts in ML are:

(a) Terminologies of Machine Learning

- Model – A model is a specific representation learned from data by applying some machine learning algorithm. A model is also called a hypothesis.

- Feature – A feature is an individual measurable property of the data. A set of numeric features can be conveniently described by a feature vector. Feature vectors are fed as input to the model. For example, in order to predict a fruit, there may be features like color, smell, taste, etc.

- Target (Label) – A target variable or label is the value to be predicted by our model. For the fruit example discussed in the feature section, the label with each set of input would be the name of the fruit like apple, orange, banana, etc.

- Training – The idea is to give a set of inputs (features) and it's expected outputs (labels), so after training, we will have a model (hypothesis) that will then map new data to one of the categories trained on.

- Prediction – Once our model is ready, it can be fed a set of inputs to which it will provide a predicted output (label).

(b) Types of Machine Learning

- **Supervised Learning** – This involves learning from a training dataset with labeled data using classification and regression models. This learning process continues until the required level of performance is achieved.

- **Unsupervised Learning** – This involves using unlabelled data and then finding the underlying structure in the data in order to learn more and more about the data itself using factor and cluster analysis models.

- **Semi-supervised Learning** – This involves using unlabelled data like Unsupervised Learning with a small amount of labeled data. Using labeled data vastly increases the learning accuracy and is also more cost-effective than Supervised Learning.

- **Reinforcement Learning** – This involves learning optimal actions through trial and error. So the next action is decided by learning behaviors that are based on the current state and that will maximize the reward in the future.

Advantages of Machine learning

- **Easily identifies trends and patterns -**

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.

- **No human intervention needed (automation)**

With ML, you don't need to babysit your project every step of the way. Since it means giving machines the ability to learn, it lets them make predictions and also improve the algorithms on their own. A common example of this is anti-virus softwares; they learn to filter new threats as they are recognized. ML is also good at recognizing spam.

- **Continuous Improvement**

As ML algorithms gain experience, they keep improving in accuracy and efficiency. This lets them make better decisions. Say you need to make a weather forecast model. As the amount of data you have keeps growing, your algorithms learn to make more accurate predictions faster.

- **Handling multi-dimensional and multi-variety data**

Machine Learning algorithms are good at handling data that are multi-dimensional and multi-variety, and they can do this in dynamic or uncertain environments.

- **Wide Applications**

You could be an e-tailer or a healthcare provider and make ML work for you. Where it does apply, it holds the capability to help deliver a much more personal experience to customers while also targeting the right customers.

Disadvantages of Machine Learning

- **Data Acquisition**

Machine Learning requires massive data sets to train on, and these should be inclusive/unbiased, and of good quality. There can also be times where they must wait for new data to be generated.

- **Time and Resources**

ML needs enough time to let the algorithms learn and develop enough to fulfill their purpose with a considerable amount of accuracy and relevancy. It also needs massive resources to function. This can mean additional requirements of computer power for you.

- **Interpretation of Results**

Another major challenge is the ability to accurately interpret results generated by the algorithms. You must also carefully choose the algorithms for your purpose.

- **High error-susceptibility**

Machine Learning is autonomous but highly susceptible to errors. Suppose you train an algorithm with data sets small enough to not be inclusive. You end up with biased predictions coming from a biased training set. This leads to irrelevant advertisements being displayed to customers. In the case of ML, such blunders can set off a chain of errors that can go undetected for long periods of time.

CHAPTER-6: SYSTEM STUDY

FEASIBILITY STUDY

The feasibility of the project is analyzed in this phase and business proposal is put forth with a very general plan for the project and some cost estimates. During system analysis the feasibility study of the proposed system is to be carried out. This is to ensure that the proposed system is not a burden to the company. For feasibility analysis, some understanding of the major requirements for the system is essential.

Three key considerations involved in the feasibility analysis are

- ECONOMICAL FEASIBILITY
- TECHNICAL FEASIBILITY
- SOCIAL FEASIBILITY

6.1 ECONOMICAL FEASIBILITY

This study is carried out to check the economic impact that the system will have on the organization. The amount of fund that the company can pour into the research and development of the system is limited. The expenditures must be justified. Thus the developed system as well within the budget and this was achieved because most of the technologies used are freely available. Only the customized products had to be purchased.

6.2 TECHNICAL FEASIBILITY

This study is carried out to check the technical feasibility, that is, the technical requirements of the system. Any system developed must not have a high demand on the available technical resources. This will lead to high demands on the available technical resources. This will lead to high demands being placed on the client. The developed system must have a modest requirement, as only minimal or null changes are required for implementing this system.

6.3 SOCIAL FEASIBILITY

The aspect of study is to check the level of acceptance of the system by the user. This includes the process of training the user to use the system efficiently. The user must not feel threatened by the system, instead must accept it as a necessity. The level of acceptance by the users solely depends on the methods that are employed to educate the user about the system and to make him familiar with it. His level of confidence must be raised so that he is also able to make some constructive criticism, which is welcomed, as he is the final user of the system.

CHAPTER-7: SYSTEM TESTING

The purpose of testing is to discover errors. Testing is the process of trying to discover every conceivable fault or weakness in a work product. It provides a way to check the functionality of components, sub- assemblies, assemblies and/or a finished product. It is the process of exercising software with the intent of ensuring that the Software system meets its requirements and user expectations and does not fail in an unacceptable manner. There are various types of test. Each test type addresses a specific testing requirement.

7.1 TYPES OF TESTS

7.1.1 Unit testing

Unit testing involves the design of test cases that validate that the internal program logic is functioning properly, and that program inputs produce valid outputs. All decision branches and internal code flow should be validated. It is the testing of individual software units of the application .it is done after the completion of an individual unit before integration. This is a structural testing, that relies on knowledge of its construction and is invasive. Unit tests perform basic tests at component level and test a specific business process, application, and/or system configuration. Unit tests ensure that each unique path of a business process performs accurately to the documented specifications and contains clearly defined inputs and expected results.

7.1.2. Integration Testing

Integration tests are designed to test integrated software components to determine if they actually run as one program. Testing is event driven and is more concerned with the basic outcome of screens or fields. Integration tests demonstrate that although the components were individually satisfaction, as shown by successfully unit testing, the combination of components is correct and consistent. Integration testing is specifically aimed at exposing the problems that arise from the combination of components.

Software integration testing is the incremental integration testing of two or more integrated software components on a single platform to produce failures caused by interface defects.

The task of the integration test is to check that components or software applications, e.g. components in a software system or – one step up – software applications at the company level – interact without error.

Test Results: All the test cases mentioned above passed successfully. No defects encountered.

7.1.3 Functional testing

Functional tests provide systematic demonstrations that functions tested are available as specified by the business and technical requirements, system documentation, and user manuals.

Functional testing is centered on the following items:

Valid Input : identified classes of valid input must be accepted. Invalid

Input : identified classes of invalid input must be rejected. Functions

: identified functions must be exercised.

Output : identified classes of application outputs must be exercised. Systems

: interfacing systems or procedures must be invoked.

Organization and preparation of functional tests is focused on requirements, key functions, or special test cases. In addition, systematic coverage pertaining to identify Business process flows; data fields, predefined processes, and successive processes must be considered for testing. Before functional testing is complete, additional tests are identified and the effective value of current tests is determined.

7.1.4 System Testing

System testing ensures that the entire integrated software system meets requirements. It tests a configuration to ensure known and predictable results. An example of system testing is the configuration-oriented system integration test. System testing is based on process descriptions and flows, emphasizing pre-driven process links and integration points.

White Box Testing

White Box Testing is a testing in which in which the software tester has knowledge of the inner workings, structure and language of the software, or at least its purpose. It is purpose. It is used to test areas that cannot be reached from a black box level.

Black Box Testing

Black Box Testing is testing the software without any knowledge of the inner workings, structure or language of the module being tested. Black box tests, as most other kinds of tests, must be written from a definitive source document, such as specification or requirements document, such as specification or requirements document. It is a testing in which the software under test is treated, as a black box .you cannot “see” into it. The test provides inputs and responds to outputs without considering how the software works.

Unit Testing

Unit testing is usually conducted as part of a combined code and unit test phase of the software lifecycle, although it is not uncommon for coding and unit testing to be conducted as two distinct phases.

Test strategy and approach

Field testing will be performed manually and functional tests will be written in detail.

Test objectives

- All field entries must work properly.
- Pages must be activated from the identified link.
- The entry screen, messages and responses must not be delayed.

Features to be tested

- Verify that the entries are of the correct format
- No duplicate entries should be allowed
- All links should take the user to the correct page.

7.1.5 Acceptance Testing

User Acceptance Testing is a critical phase of any project and requires significant participation by the end user. It also ensures that the system meets the functional requirements.

Test Results: All the test cases mentioned above passed successfully. No defects encountered.

CHAPTER-8: RESULTS

8.1 Home Page

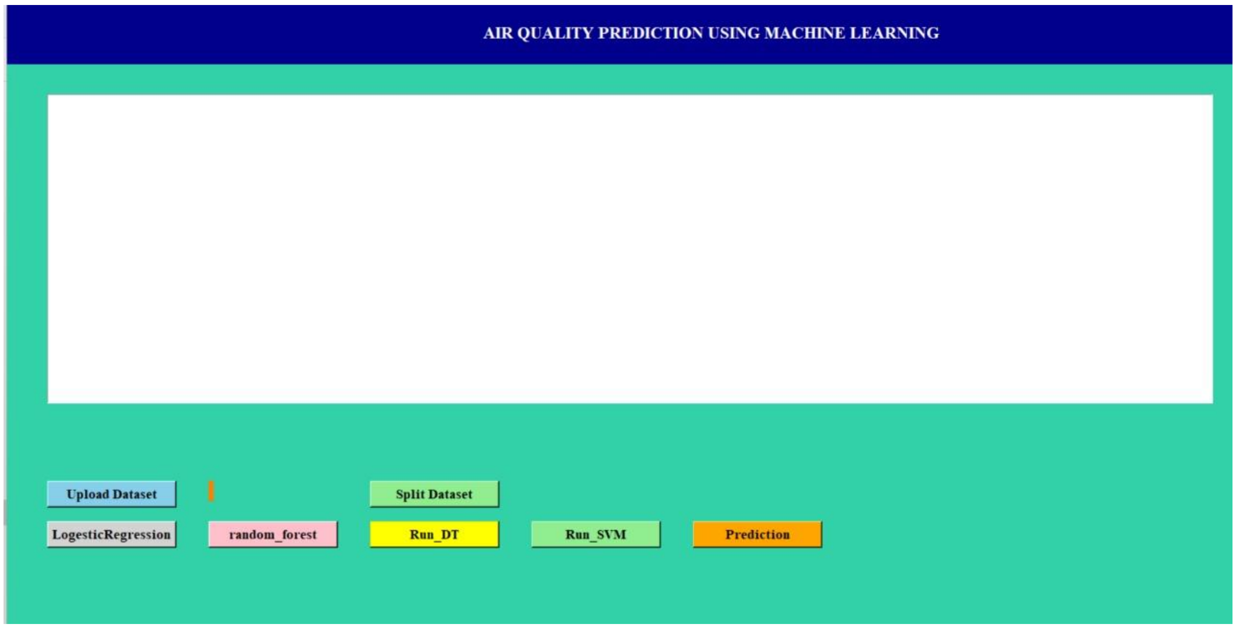


Fig.8.1 : Screenshot of Home Page

8.2 Upload Dataset

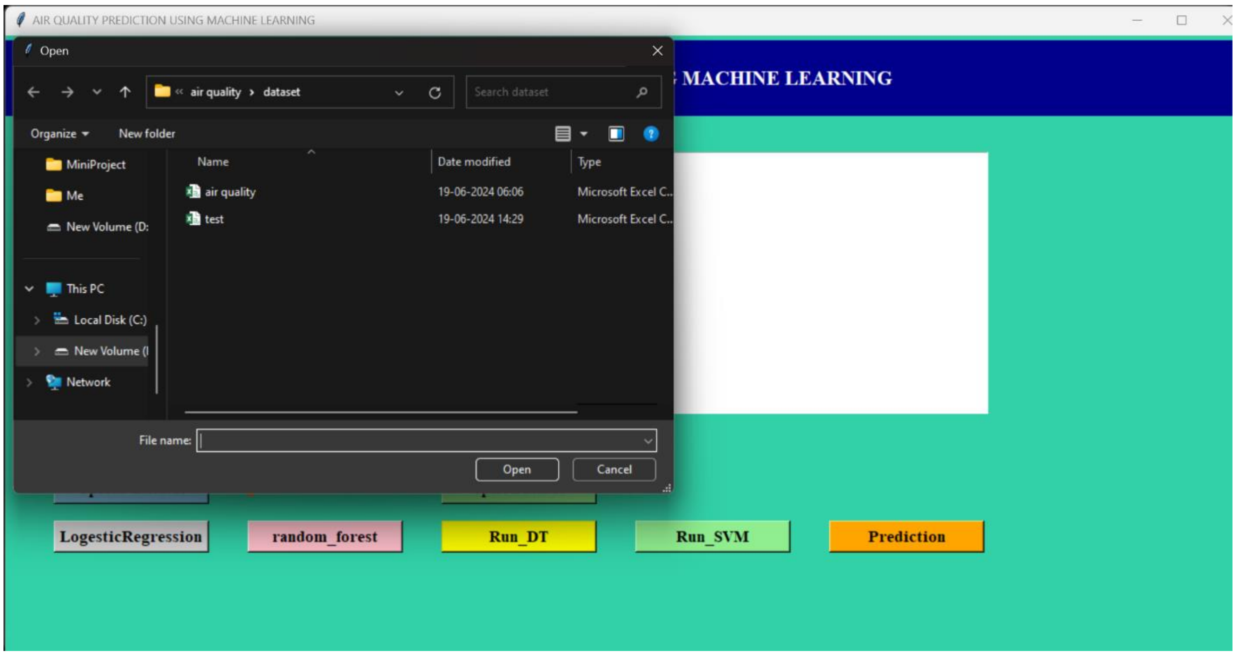
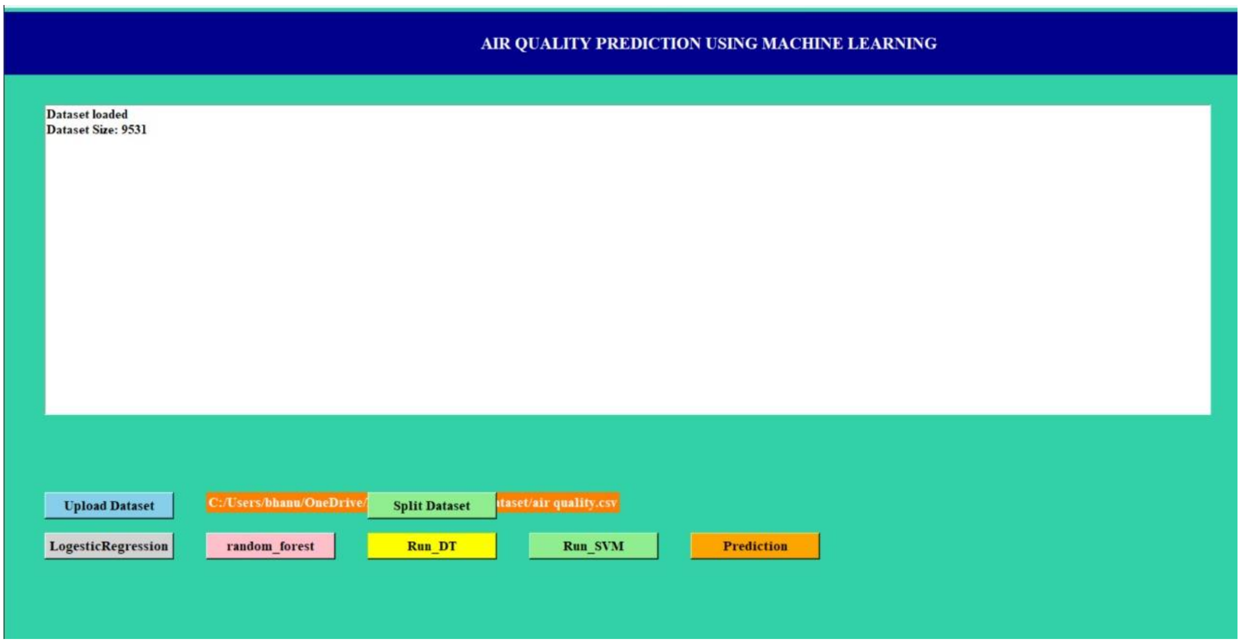


Fig.8.2 : Screenshot of Upload Dataset

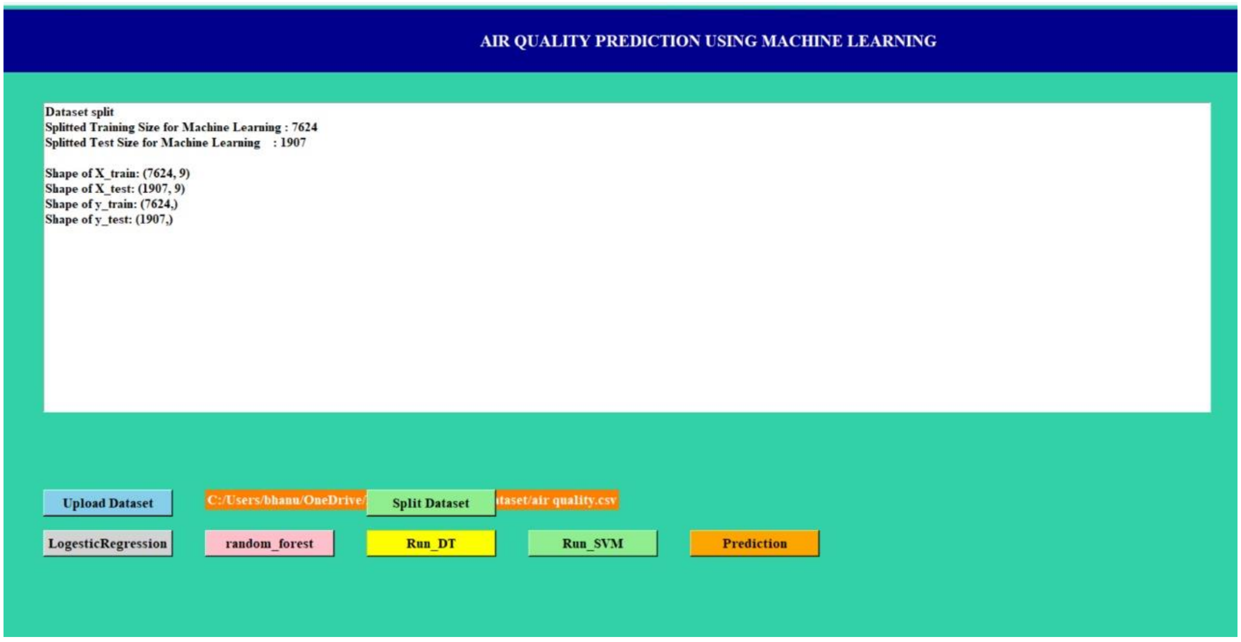
8.3 Dataset Loaded



```
Dataset loaded
Dataset Size: 9531
```

Fig.8.3: Screenshot of Dataset loaded

8.4 Split Dataset

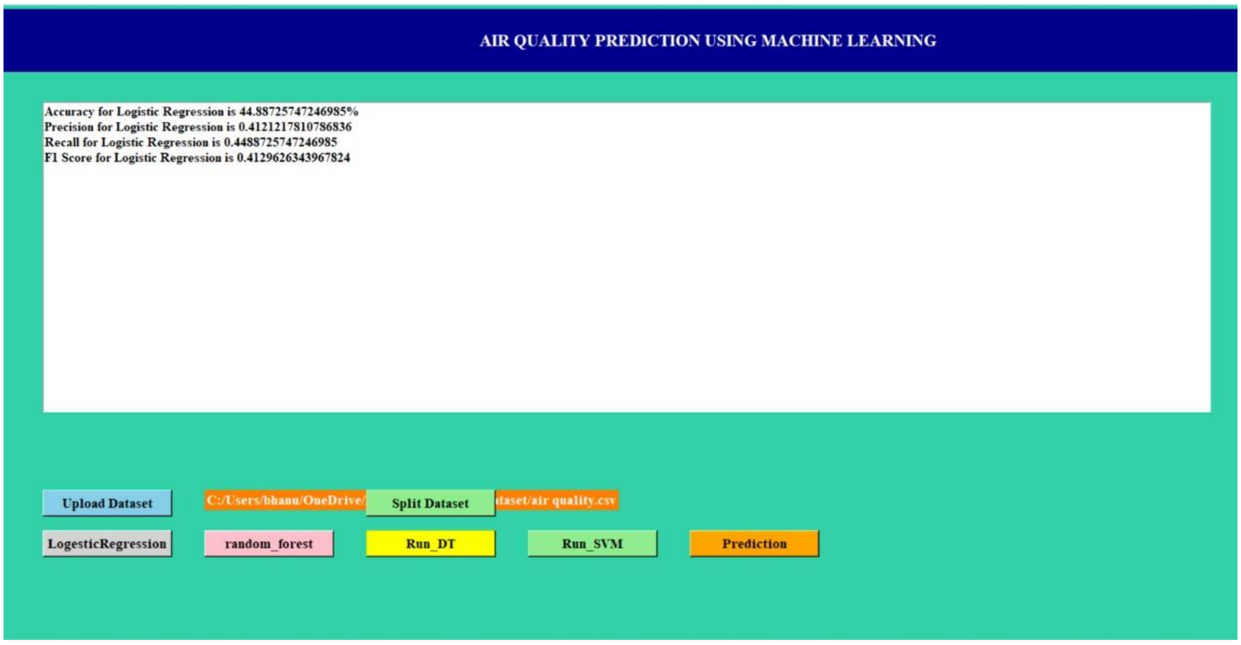


```
Dataset split
Splitted Training Size for Machine Learning : 7624
Splitted Test Size for Machine Learning    : 1907

Shape of X_train: (7624, 9)
Shape of X_test: (1907, 9)
Shape of y_train: (7624,)
Shape of y_test: (1907,)
```

Fig.8.4 : Screenshot of Split Dataset

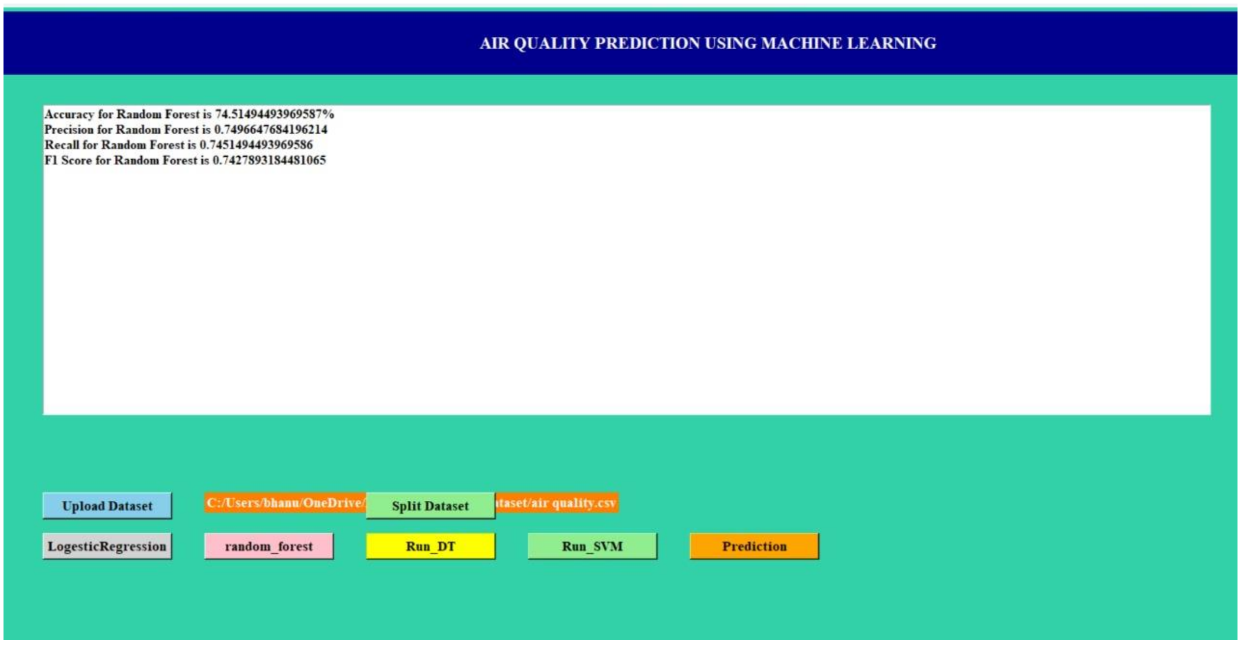
8.5 Logistic Regression



```
Accuracy for Logistic Regression is 44.88725747246985%
Precision for Logistic Regression is 0.4121217810786836
Recall for Logistic Regression is 0.4488725747246985
F1 Score for Logistic Regression is 0.4129626343967824
```

Fig.8.5: Screenshot of Logistic Regression

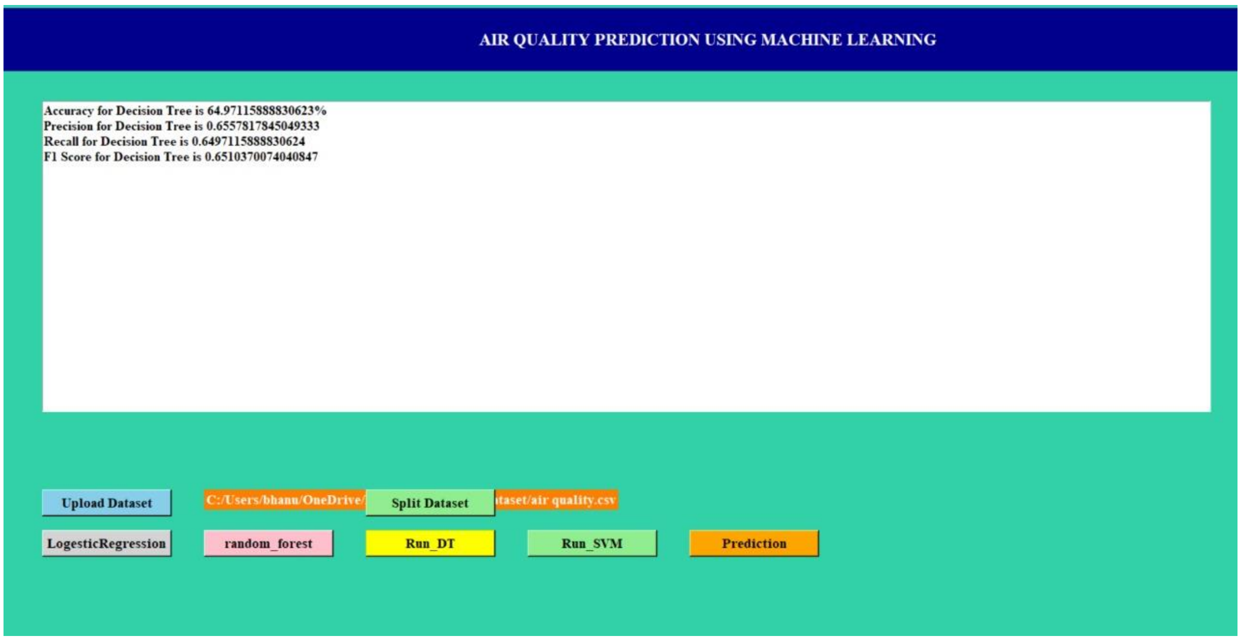
8.6 Random Forest Algorithm



```
Accuracy for Random Forest is 74.51494493969587%
Precision for Random Forest is 0.7496647684196214
Recall for Random Forest is 0.7451494493969586
F1 Score for Random Forest is 0.7427893184481065
```

Fig.8.6 : Screenshot of Random Forest Algorithm

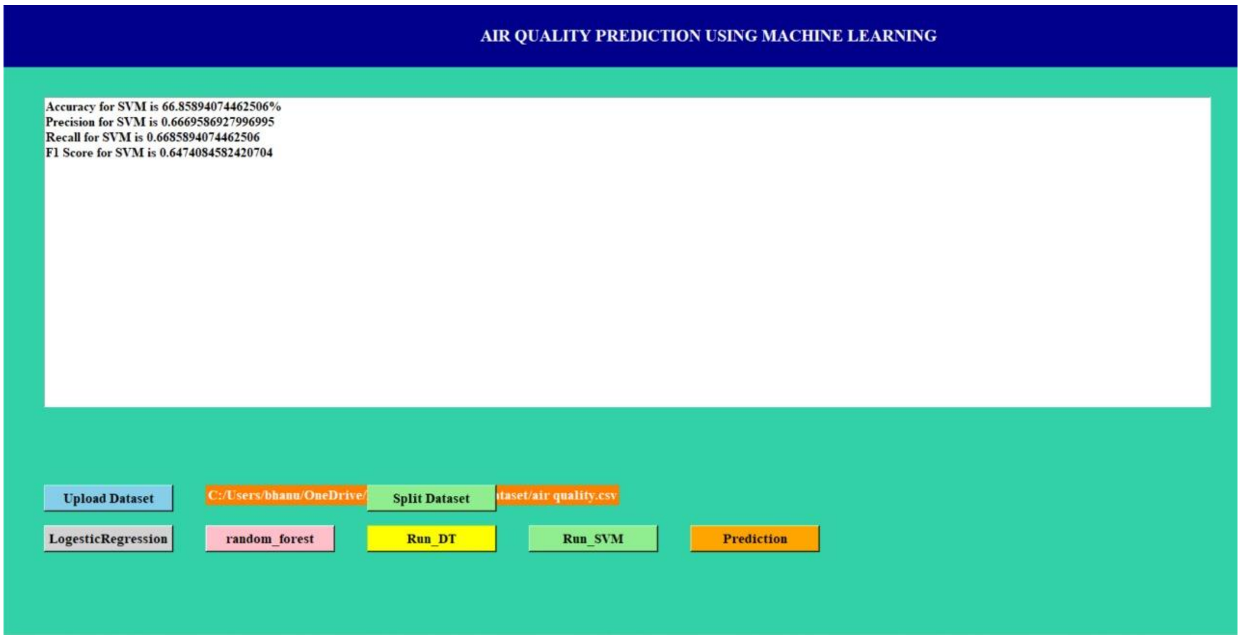
8.7 Decision Tree Algorithm



```
Accuracy for Decision Tree is 64.97115888830623%
Precision for Decision Tree is 0.6557817845049333
Recall for Decision Tree is 0.6497115888830624
F1 Score for Decision Tree is 0.6510370074040847
```

Fig.8.7 : Screenshot of Decision Tree Algorithm

8.8 Support Vector Machine



```
Accuracy for SVM is 66.85894074462506%
Precision for SVM is 0.6669586927996995
Recall for SVM is 0.6685894074462506
F1 Score for SVM is 0.6474084582420704
```

Fig.8.8: Screenshot of Support Vector Machine

8.9 Upload test dataset for prediction

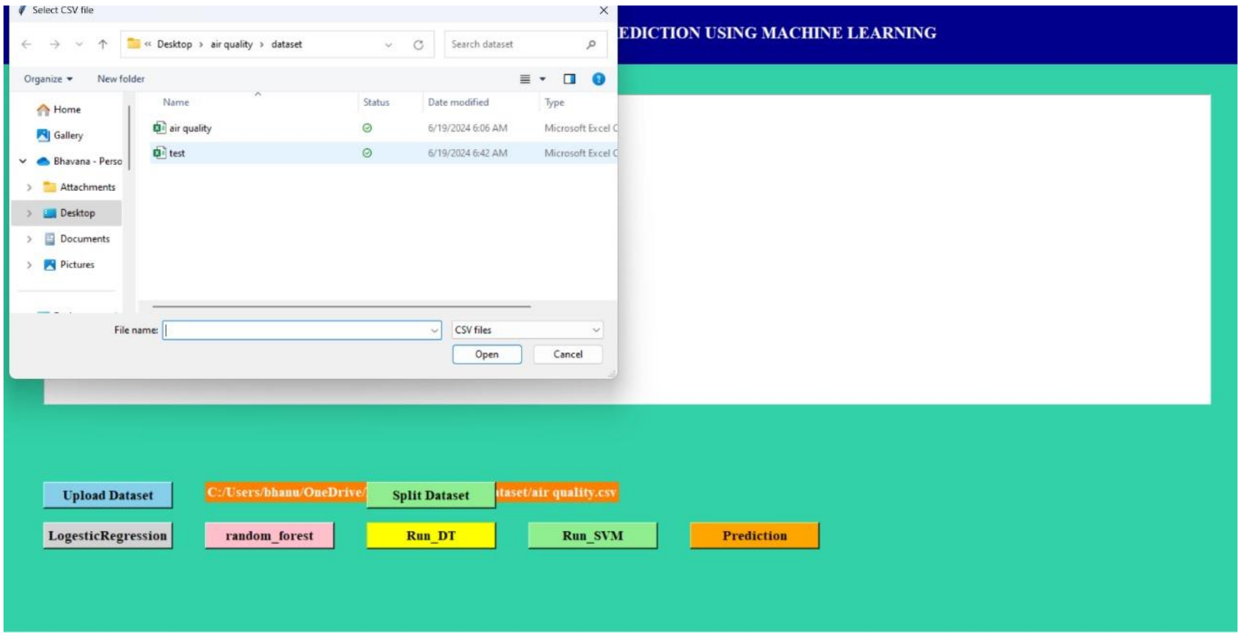
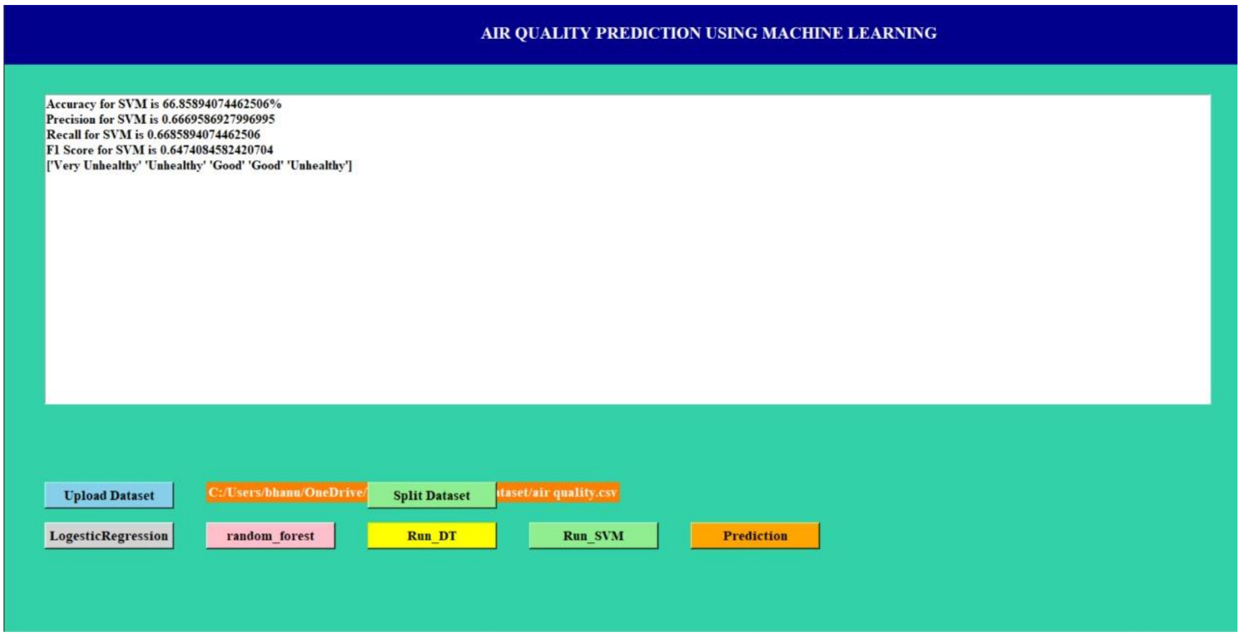


Fig.8.9: Screenshot of Uploading test dataset for prediction

8.10 Output values



```
Accuracy for SVM is 66.85894074462506%
Precision for SVM is 0.6669586927996995
Recall for SVM is 0.6685894074462506
F1 Score for SVM is 0.6474084582420704
['Very Unhealthy' 'Unhealthy' 'Good' 'Good' 'Unhealthy']
```

Fig.8.10 : Screenshot of Output Values

CHAPTER 9: CONCLUSION & FUTURE ENHANCEMENT

CONCLUSION:

The economic development achieved by the country through rapid urbanization is polluting the environment in an alarming way and putting people's lives in danger. Therefore, a correct analysis and accurate prediction of air quality remains a primary condition to achieve the objective of sustainable development. This paper focuses on the problem of prediction model design, and investigates the problems related to the optimization of the model parameters. A GA-KELM model is designed, implemented, and tested. It is experimentally proven to be more efficient than the classical shallow learning and can effectively explore and learn the interdependence of multivariate air quality correlation time series such as temperature, humidity, wind speed, SO2, and PM10. Therefore, the GA-KELM model developed in this study can be used to provide valuable support to vulnerable groups and trigger early warning of adverse air quality events. However, there are still areas for further investigation and improvement. In recent years, numerous advanced algorithms and optimization methods based on genetic algorithms and population intelligence have emerged. Therefore, future research should explore the underlying significance and value of combinatorial intelligence optimization algorithms such as the Limit Learning Machine. Additionally, we acknowledge the need to address the issue of manually setting the number of hidden layer nodes in the optimal Limit Learning Machine. Although the Dynamic Extreme Learning Machine (DELM) algorithm offers adaptive determination of hidden layer nodes without human intervention, further work should be dedicated to this aspect. Moreover, to enhance the accuracy and validity of air quality measurement and assessment, it is crucial to integrate pollutant emission factors and meteorological factors into the evaluation system. This integration will enable a more precise and comprehensive evaluation of air quality. In conclusion, our study highlights the significance of the GA-KELM model in predicting air quality. We have addressed the optimization challenges and demonstrated its superiority over traditional methods. However, there is still room for improvement and further research. Future studies should delve into advanced optimization algorithms based on genetic algorithms and population intelligence, explore the potential of the Limit Learning Machine, and strive for adaptive determination of hidden layer nodes. Furthermore, the integration of pollutant emission factors and meteorological factors into the evaluation system will advance the accuracy and reliability of air quality measurement and assessment.

FUTURE ENHANCEMENT:

Explore the use of more advanced machine learning algorithms like Long Short-Term Memory (LSTM) networks, Convolutional Neural Networks (CNN), or ensemble methods to improve the accuracy of your forecasting models. Integrate big data analytics and Internet of Things (IoT) devices to collect real-time data on environmental factors, traffic patterns, and industrial activities, which can enhance the precision of your predictions. Conduct spatial and temporal analysis to understand the geographical variations in air quality and how it changes over time. This can help in developing localized and time-specific forecasting models. Explore new features or variables that can influence air quality, such as land use data, satellite imagery, or social media data, to improve the predictive power of your models. Implement ensemble modeling techniques like model averaging or stacking to combine the strengths of multiple forecasting models and enhance the overall prediction accuracy. Develop interactive visualization tools or dashboards to present the forecasted air quality index in a user-friendly and accessible manner, enabling stakeholders to make informed decisions based on the predictions. By incorporating these future enhancements into air quality index forecasting, it can potentially improve the reliability and effectiveness of your forecasting models.

CHAPTER-10:BIBLIOGRAPHY

[1] X. Li, L. Jin, and H. Kan, ''Air pollution: A global problem needs local fixes,'' Nature, vol. 570, no. 7762, pp. 437–439, Jun. 2019.

[2] Y. Han, J. C. K. Lam, and V. O. K. Li, ''A Bayesian LSTM model to evaluate the effects of air pollution control regulations in China,'' in Proc. IEEE Big Data Workshop (Big Data), Dec. 2018, pp. 4465–4468.

[3] L. Bai, J. Wang, X. Ma, and H. Lu, ''Air pollution forecasts: An overview,'' Int. J. Environ. Res. Public Health, vol. 15, no. 4, p. 780, 2018.

[4] Y. Ding and Y. Xue, ''A deep learning approach to writer identification using inertial sensor data of air-handwriting,'' IEICE Trans. Inf. Syst., vol. E102-D, no. 10, pp. 2059–2063, 2019.

[5] S.-Q. Dotse, M. I. Petra, L. Dagar, and L. C. De Silva, ''Application of computational intelligence techniques to forecast daily PM10 exceedances in Brunei Darussalam,'' Atmos. Pollut. Res., vol. 9, no. 2, pp. 358–368, Mar. 2018.

[6] M. Jia, A. Komeily, Y. Wang, and R. S. Srinivasan, ''Adopting Internet of Things for the development of smart buildings: A review of enabling technologies and applications,'' Automat. Construct., vol. 101, pp. 111–126, May 2019.

[7] S. Abirami, P. Chitra, R. Madhumitha, and S. R. Kesavan, ''Hybrid spatio-temporal deep learning framework for particulate matter (PM2.5) concentration forecasting,'' in Proc. Int. Conf. Innov. Trends Inf. Technol. (ICITIIT), Feb. 2020, pp. 1–6.

[8] Y. Cheng, S. Zhang, C. Huan, M. O. Oladokun, and Z. Lin, ''Optimization on fresh outdoor air ratio of air conditioning system with stratum ventilation for both targeted indoor air quality and maximal energy saving,'' Building Environ., vol. 147, pp. 11–22, Jan. 2019.

[9] A. C. Cosma and R. Simha, ''Machine learning method for real-time non-invasive prediction of individual thermal preference in transient conditions,'' Building Environ., vol. 148, pp. 372–383, Jan. 2019.

[10] M. Bhowmik, K. Deb, A. Debnath, and B. Saha, ''Mixed phase Fe2O3/Mn3O4 magnetic nanocomposite for enhanced adsorption of methyl orange dye: Neural network modeling and response surface methodology optimization,'' Appl. Organometallic Chem., vol. 32, no. 3, p. e4186, Mar. 2018.

[11] V. Chaudhary, A. Deshbhratar, V. Kumar, and D. Paul, ''Time series based LSTM model to predict air pollutant's concentration for prominent cities in India,'' in Proc. Int. Workshop

Utility-Driven Mining (UDM), Aug. 2018, pp. 1–9.

[12] M. Chen, J. Yang, L. Hu, M. S. Hossain, and G. Muhammad, ‘‘Urban healthcare big data system based on crowdsourced and cloud-based air quality indicators,’’ IEEE Commun. Mag., vol. 56, no. 11, pp. 14–20, Nov. 2018.

[13] R. Chen, X. Wang, W. Zhang, X. Zhu, A. Li, and C. Yang, ‘‘A hybrid CNN-LSTM model for typhoon formation forecasting,’’ GeoInformatica, vol. , no. 3, pp. 375–396, Jul. 2019.

[14] S. Du, T. Li, Y. Yang, and S. Horng, ‘‘Deep air quality forecasting using hybrid deep learning framework,’’ IEEE Trans. Knowl. Data Eng., vol. 33, no. 6, pp. 2412–2424, Jun. 2021.

[15] R. Feng, H.-J. Zheng, H. Gao, A.-R. Zhang, C. Huang, J.-X. Zhang, K. Luo, and J.-R. Fan, ‘‘Recurrent neural network and random forest for analysis and accurate forecast of atmospheric pollutants: A case study in Hangzhou, China,’’ J. Cleaner Prod., vol. 231, pp. 1005–1015, Sep. 2019.

[16] B. S. Freeman, G. Taylor, B. Gharabaghi, and J. Thé, ‘‘Forecasting air quality time series using deep learning,’’ J. Air Waste Manage. Assoc., vol. 68, no. 8, pp. 866–886, Aug. 2018.

[17] S. Mahajan, H.-M. Liu, T.-C. Tsai, and L.-J. Chen, ‘‘Improving the accuracy and efficiency of PM2.5 forecast service using cluster-based hybrid neural network model,’’ IEEE Access, vol. 6, pp. 19193–19204, 2018.

[18] J. Jin, J. Gubbi, S. Marusic, and M. Palaniswami, ‘‘An information framework for creating a smart city through Internet of Things,’’ IEEE Internet Things J., vol. 1, no. 2, pp. 112–121, Apr. 2014.

[19] A. Grover, A. Kapoor, and E. Horvitz, ‘‘A deep hybrid model for weather forecasting,’’ in Proc. 21st ACM SIGKDD Int. Conf. Knowl. DiscoveryData Mining, Aug. 2015, p. 379–386.

[20] A. Agarwal and M. Sahu, ‘‘Forecasting PM2.5 concentrations using statistical modeling for Bengaluru and Delhi regions,’’ Environ. Monit. Assessment, vol. 195, p. 502, Mar. 2023.

[21] D. J. Lary, T. Lary, and B. Sattler, ‘‘Using machine learning to estimate global PM2.5 for environmental health studies,’’ Environ. Health Insights, vol. 9, no. 1, pp. 41–52, 2015.

[22] Y. Zheng, X. Yi, M. Li, R. Li, Z. Shan, E. Chang, and T. Li, ‘‘Forecasting fine-grained air quality based on big data,’’ in Proc. 21th ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining, New York, NY, USA, Aug. 2015, pp. 2267–2276.

[23] Y. Zheng, L. Capra, O. Wolfson, and H. Yang, ‘‘Urban computing: Concepts,

methodologies, and applications,'' ACM Trans. Intell. Syst. Technol., vol. 5, no. 3, p. 38, Sep. 2014.

[24] T. S. Rajput and N. Sharma, ``Multivariate regression analysis of air quality index for Hyderabad city: Forecasting model with hourly frequency,'' Int. J. Appl. Res., vol. 3, no. 8, pp. 443–447, 2017.

[25] Z. Kang and Z. Qu, ``Application of BP neural network optimized by genetic simulated annealing algorithm to prediction of air quality index in Lanzhou,'' in Proc. IEEE Comput. Intell. Appl. (ICCIA), Sep. 2017, pp. 155–160, doi: 10.1109/CIAPP.2017.8167199.

[26] B. Liu, S. Yan, J. Li, G. Qu, Y. Li, J. Lang, and R. Gu, ``A sequence-tosequence air quality predictor based on the n-step recurrent prediction,'' IEEE Access, vol. 7, pp. 43331–43345, 2019.

[27] K. Elbaz, I. Hoteit, W. M. Shaban, and S.-L. Shen, ``Spatiotemporal air quality forecasting and health risk assessment over smart city of NEOM,'' Chemosphere, vol. 313, Feb. 2022, Art. no. 137636.

[28] P. C. Campbell, Y. Tang, P. Lee, B. Baker, D. Tong, R. Saylor, A. Stein, J. Huang, H.-C. Huang, E. Strobach, J. McQueen, L. Pan, I. Stajner, J. Sims, J. Tirado-Delgado, Y. Jung, F. Yang, T. L. Spero, and R. C. Gilliam, ``Development and evaluation of an advanced national air quality forecasting capability using the NOAA Global Forecast System version 16,'' Geosci. Model Develop, vol. 15, no. 8, pp. 3281–3313, Apr. 2022.