A

Mini Project Report

On

**FLOOD FORECASTING USING MACHINE LEARNING**

Submitted in partial fulfillment of the

Requirement for the award of degree of

**BACHELOR OF TECHNOLOGY**

**In**

**COMPUTER SCIENCE AND ENGINEERING(AI&ML)**

|  |  |
| --- | --- |
|  | **By** |
| **RUKSANA SYED** | **216C1A6634** |

**BHUVANESWARI.G 216C1A6616**

**SINDHUJA.P 216C1A6629**

**LASYA.G 216C1A6611**

Under the Guidance of

**Mr.M.VenkateswaRao M.Tech**

**Assistant Professor**

****

**DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING (AI&ML)**

**PRIYADARSHINI INSTITUTE OF SCIENCE & TECHNOLOGY FOR WOMEN**

SAI PRABHATH NAGAR, PEDDATHANDA, KHAMMAM-507003

(Approved by AICTE, Affiliated to JNTUH, Hyderabad-T.S.)

**2021– 2025**

**DEPARTMENT COMPUTER SCIENCE AND ENGINEERING(AI&ML)**

**PRIYADARSHINI INSTITUTE OF SCIENCE & TECHNOLOGY FOR WOMEN**

SAI PRABHATH NAGAR, PEDDATHANDA, KHAMMAM-507003

(Approved by AICTE, Affiliated to JNTUH, Hyderabad-T.S.)



***CERTIFICATE***

This is to certify that the mini project report Entitled **“FLOOD FORECASTING USING MACHINE LEARNING”** is a bonafide record of work carried out by ***Ruksana Syed-216C1A6634, Bhuvaneswari.G-216C1A6616, Sindhuja.P-216C1A6629, Lasya.G-216C1A6611.*** We hereby accord our approval of it as a mini project report carried out and presented in a manner required for its acceptance in partial fulfillment for award of degree of

**BACHELOR OF TECHNOLOGY**

**In**

**COMPUTER SCIENCE AND ENGINEERING(AI&ML)**

Of

**Jawaharlal Nehru Technological University, Hyderabad.**

During the academic year 2024-25

**HEAD OF THE DEPARTMENT**

Mr.M.VenkateswaRao M.Tech.

Assistant Professor

|  |  |
| --- | --- |
| **INTERNAL GUIDE** |  |
| Mr.M.VenkateswaRao M.Tech |  |
| Assistant Professor |  |

**EXTERNAL EXAMINER**

**Principal**

**Dr. B.Gopal M.Tech.,Ph.D**.

**DECLARATION**

We declare that the project entitled **“FLOOD FORECASTING USING MACHINE LEARNING”** recorded in this reportdoes not form part of any other report on which a degree has been awarded earlier. We further declare that this project report is based on my work carried out at the **“PRIYADARSHINI INSTITUTE OF SCIENCE & TECHNOLOGY FOR WOMEN”,** in the BTech course.

|  |  |
| --- | --- |
|  | **REPORTED BY** |

RUKSANA SYED 216C1A6634

BHUVANESWARI.G 216C1A6616

SINDHUJA.P 216C1A6629

LASYA.G 216C1A6611

**ACKNOWLEDGEMENT**

We express our gratitude to **Dr.K.Naveen Babu** M.Tech, Ph.D Chairman, **Priyadarshini Institute of Science and Technology for Women, Khammam,** for giving us an opportunity to do this project work.

We express our gratitude to **Dr.B.Gopal** M.Tech, Ph.D. Principal, Priyadarshini Institute of Science and Technology for Women, Khammam, for giving us an opportunity to do this project work.

We express our gratitude to **Mr.M.VenkateswaRao** Head of the department, Computer Science and Engineering(AI&ML),Priyadarshini Institute of Science and Technology for Women, Khammam, for giving us an opportunity to do this project work.

We thankful to our Guide **Mr.M.VenkateswaRao** Assistant Professor, Computer Science and Engineering(AI&ML), Who with his continuous efforts, unfailing interest, constant support and providing us the right infrastructure helped us in completing this project work.

We would like to thank all the faculty members of CSE(AI&ML) Department and our friends for their valuable suggestions and support which directly or indirectly helped us mounding this project in to a comprehensive one.

We wish to express our gratitude to our parents, whose love and encouragement have a great support throughout our education.

## CONTENTS

**CONTENTS**

**Sno Pg.no**

**ABSTRACT**

**1. INTRODUCTION 1**

**2. EXISTING SYSTEM 2-3**

2.1 Disadvantages

2.2 Advantages

**3. PROPOSED SYSTEM 4-5**

3.1 Advantages

**4. PROJECT LIFE CYCLE 6-8**

**5. REQUIREMENTS SPECIFICATIONS 9-22**

5.1 Software Environment

**6. FUNCTIONAL REQUIREMENTS 23-27**

**7. SYSTEM IMPLEMENTATION 28-42**

7.1 Sample Code

7.2 Testing

7.3 Execution

**8. FUTURE SCOPE 43**

**9. CONCLUSION 44**

**10. REFERENCES 46**

**ABSTRACT**

Floods are among the most destructive natural disasters, which are highly complex to model. The research on the advancement of flood prediction models contributed to risk reduction, policy suggestion, minimization of the loss of human life, and reduction the property damage associated with floods. To mimic the complex mathematical expressions of physical processes of floods, during the past two decades, machine learning (ML) methods contributed highly in the advancement of prediction systems providing better performance and cost-effective solutions. Due to the vast benefits and potential of ML, its popularity dramatically increased among hydrologists. Researchers through introducing novel ML methods and hybridizing of the existing ones aim at discovering more accurate and efficient prediction models. The main contribution of this paper is to demonstrate the state of the art of ML models in flood prediction and to give insight into the most suitable models. In this paper, the literature where ML models were benchmarked through a qualitative analysis of robustness, accuracy, effectiveness, and speed are particularly investigated to provide an extensive overview on the various ML algorithms used in the field. The performance comparison of ML models presents an in-depth understanding of the different techniques within the framework of a comprehensive evaluation and discussion. As a result, this paper introduces the most promising prediction methods for both long-term and short-term floods. Furthermore, the major trends in improving the quality of the flood prediction models are investigated. Among them, hybridization, data decomposition, algorithm ensemble, and model optimization are reported as the most effective strategies for the improvement of ML methods. This survey can be used as a guideline for hydrologists as well as climate scientists in choosing the proper ML method according to the prediction task.

**1.INTRODUCTION**

India is the country with the highest annual risk of flooding in the entire world. In large cities, low-lying areas are where water logging typically happens. Forecasting floods is so crucial in these areas. There were numerous flood-prone regions in recent years, including Assam, Bihar, Goa, Orissa, Pune, Maharashtra, Tamil Nadu, Karnataka, Kerala, and Gujarat.In November 2015, Chennai saw rainfall of 1049 millimeters (mm). The best November precipitation total since 1918 was 1088 mm. In the Kanchipuram district, 64 cm of rain falls on average between October and December. It received the most precipitation, 181.5 cm, which is 183% more than average. The average rainfall in the Tiruvallur district is 59 cm, but 146 cm was recorded.

There has been a lot of research into flood prediction, but not many methods provide an accurate estimate. Machine Learning is heavily used in flood prediction analysis (ML). Machine learning provides a wide range of approaches for more precise problem prediction. In this paper, we recommended estimating the flash flood to prevent flood- prone areas. The strategy is to establish the ML algorithm model. It incorporates the flood factor to provide more accurate short-term predictions in urban areas. Depending on the method of data transmission, retrieval of information could take hours. Still, images and video streams can provide useful information for a variety of applications. One of the applications for the technique of visual sensing is an early alert system for controlling and preventing flooding. Image processing is the process of extracting useful information from digital images using computer algorithms, which is a critical procedure in visual sensing systems.

Image segmentation is commonly used to partition an image into several regions to understand its content, which frequently depend on its pixel's characteristics. Many applications have made use of image classification., incorporating flood control, autonomous driving, and medical image processing. It can involve separating the subject from the backdrop in flood disaster scenarios. Image segmentation techniques used by researchers and industry today include thresholding, boundary-based, region-based, and alternative methods. Specific visual classification methods created for flood disaster scenarios were covered in certain articles.

### ****2.EXISTING SYSTEM****

The existing flood prediction systems typically rely on historical rainfall data, river flow rates, and hydrological models to forecast floods. These systems use traditional statistical techniques or rule-based approaches, which provide a general overview of flood risks in vulnerable regions. In some cases, satellite imagery and early warning systems are utilized to monitor flood-prone areas. However, these systems often fail to deliver precise and localized predictions in real-time. This survey identifies the state of the art of ML methods for flood prediction where peer-reviewed articles in top-level subject fields are reviewed. Among the articles identified, through search queries using the search strategy, those including the performance evaluation and comparison of ML methods were given priority to be included in the review to identify the ML methods that perform better in particular applications. Furthermore, to choose an article, four types of quality measure for each article were considered, i.e., source normalized impact per paper (SNIP), CiteScore, SCImago journal rank (SJR), and h-index. The papers were reviewed in terms of flood resource variables, ML methods, prediction type, and the obtained results The applications in flood prediction can be classified according to flood resource variables, i.e., water level, river flood, soil moisture, rainfall–discharge, precipitation, river inflow, peak flow, river flow, rainfall–runoff, flash flood, rainfall, streamflow, seasonal stream flow, flood peak discharge, urban flood, plain flood, groundwater level, rainfall stage, flood frequency analysis, flood quantiles, surge level, extreme flow, storm surge, typhoon rainfall, and daily flows [59]. Among these key influencing flood resource variables, rainfall and the spatial examination of the hydrologic cycle had the most remarkable role in runoff and flood modeling [60]. This is the reason why quantitative rainfall prediction, including avalanches, slush flow, and melting snow, is traditionally used for flood prediction, especially in the prediction of flash floods or short-term flood prediction [61]. However, rainfall prediction was shown to be inadequate for accurate flood prediction. For instance, the prediction of streamflow in a long-term flood prediction scenario depends on soil moisture estimates in a catchment, in addition to rainfall [62]. Although, high-resolution precipitation forecasting is essential, other flood resource variables were considered in the [63]. Thus, the methodology of this literature review aims to include the most effective flood resource variables in the search queries.

**2.1 Disadvantages of the Existing System**

1. **Low Accuracy:** The reliance on outdated or generalized models often leads to inaccurate flood predictions.
2. **Limited Real-Time Capabilities:** Many systems lack the ability to process real-time data effectively, leading to delayed predictions.
3. **High Cost:** The maintenance of satellite-based or large-scale hydrological systems can be expensive, making them inaccessible for many regions.
4. **Lack of Localization:** Existing systems fail to provide granular predictions tailored to specific districts or urban areas.
5. **Inefficient Alert Systems:** Current methods do not integrate visual or sensory data effectively, limiting the usefulness of early warning systems.

**2.2 Advantages of the Existing System**

1**. Early Warnings**: Provide timely alerts to prepare for floods.

2. **Real-Time Monitoring**: Track rainfall and river levels continuously.

3. **Accurate Predictions**: Use advanced technologies for precise forecasts.

4. **Risk Mitigation**: Help reduce flood damage and save lives.

5. **Public Awareness**: Inform communities for better preparedness.

**3.PROPOSED SYSTEM**

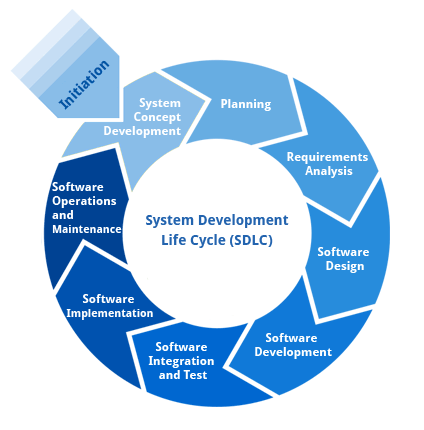
The proposed flood prediction system leverages machine learning algorithms to improve the accuracy and timeliness of flood forecasts. By utilizing historical rainfall data, geographic details, and other flood-related factors, the model predicts whether a flood might occur based on specific rainfall thresholds for different regions. The system integrates image processing techniques, such as segmentation and classification, to analyze visual data, enhancing the detection and management of flood scenarios. For creating the ML prediction model, the historical records of flood events, in addition to real-time cumulative data of a number of rain gauges or other sensing devices for various return periods, are often used. The sources of the dataset are traditionally rainfall and water level, measured either by ground rain gauges, or relatively new remotesensing technologies such as satellites, multisensor systems, and/or radars. Nevertheless, remote sensing is an attractive tool for capturing higher-resolution data in real time. In addition, the high resolution of weather radar observations often provides a more reliable dataset compared to rain gauges. Thus, building a prediction model based on a radar rainfall dataset was reported to provide higher accuracy in general. Whether using a radar-based dataset or ground gauges to create a prediction model, the historical dataset of hourly, daily, and/or monthly values is divided into individual sets to construct and evaluate the learning models. To do so, the individual sets of data undergo training, validation, verification, and testing. The principle behind the ML modeling workflow and the strategy for flood modeling are described in detail in the literature. Figure 2 represents the basic flow for building an ML model. The major ML algorithms applied to flood prediction include ANNs , neuro-fuzzy, adaptive neuro-fuzzy inference systems (ANFIS), support vector machines (SVM), wavelet neural networks (WNN), and multilayer perceptron (MLP). In the following subsections, a brief description and background of these fundamental ML algorithms are presented.

**3.1 Advantages of the Proposed System:**

1. **High Accuracy:** Machine learning models such as regression, decision trees, and neural networks enable precise flood predictions based on historical and real-time data.
2. **Real-Time Predictions:** Integration with sensory data, including rainfall measurements and images, allows for timely flood warnings.
3. **Cost-Effective:** Compared to satellite-based systems, the proposed model is more affordable and scalable for urban and rural areas alike.
4. **Localized Forecasts:** The system provides predictions tailored to specific districts and urban zones, improving the relevance of alerts.
5. **Enhanced Early Warning:** By incorporating image processing techniques, the system offers advanced visual sensing capabilities to monitor flood risks and issue alerts efficiently.
6. **Versatility:** The use of machine learning ensures adaptability, allowing the system to improve with more data and changing environmental conditions.

**4.PROJECT LIFE CYCLE**

The **Project Lifecycle**, or Software Development Life Cycle in systems engineering, information systems and software engineering, is the process of creating or altering systems, and the models and methodologies use to develop these systems.



**Requirement Analysis and Design**:

Analysis gathers the requirements for the system. This stage includes a detailed study of the business needs of the organization. Options for changing the business process may be considered. Design focuses on high level design like, what programs are needed and how are they going to interact, low-level design (how the individual programs are going to work), interface design (what are the interfaces going to look like) and data design (what data will be required). During these phases, the software's overall structure is defined. Analysis and Design are very crucial in the whole development cycle. Any glitch in the design phase could be very expensive to solve in the later stage of the software development. Much care is taken during this phase. The logical system of the product is developed in this phase.

**Implementation:**

In this phase the designs are translated into code. Computer programs are written using a conventional programming language or an application generator. Programming tools like Compilers, Interpreters, and Debuggers are used to generate the code. Different high level programming languages like C, C++, Pascal, Java, .Net are used for coding. With respect to the type of application, the right programming language is chosen.

**Testing:**

In this phase the system is tested. Normally programs are written as a series of individual modules, this subject to separate and detailed test. The system is then tested as a whole. The separate modules are brought together and tested as a complete system. The system is tested to ensure that interfaces between modules work (integration testing), the system works on the intended platform and with the expected volume of data (volume testing) and that the system does what the user requires (acceptance/beta testing).

**Maintenance:**

Inevitably the system will need maintenance. Software will definitely undergo change once it is delivered to the customer. There are many reasons for the change. Change could happen because of some unexpected input values into the system. In addition, the changes in the system could directly affect the software operations. The software should be developed to accommodate changes that could happen during the post implementation period.

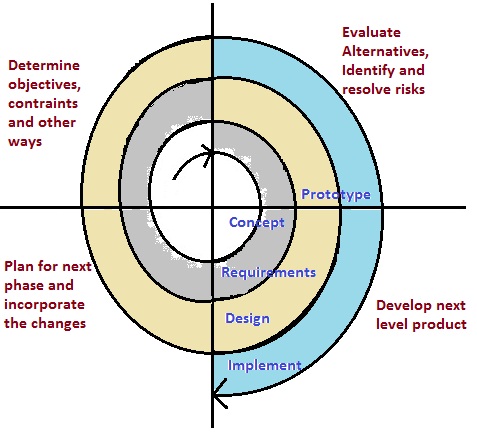
**SDLC METHDOLOGIES:**

This document play a vital role in the development of life cycle (SDLC) as it describes the complete requirement of the system. It means for use by developers and will be the basic during testing phase. Any changes made to the requirements in the future will have to go through formal change approval process.

**SPIRAL MODEL** was defined by Barry Boehm in his 1988 article, “A spiral Model of Software Development and Enhancement. This model was not the first model to discuss iterative development, but it was the first model to explain why the iteration models.

As originally envisioned, the iterations were typically 6 months to 2 years long. Each phase starts with a design goal and ends with a client reviewing the progress thus far.

**The following diagram shows how a spiral model acts like:**



The steps for Spiral Model can be generalized as follows:

* The new system requirements are defined in as much details as possible. This usually involves interviewing a number of users representing all the external or internal users and other aspects of the existing system.
* A preliminary design is created for the new system.
* A first prototype of the new system is constructed from the preliminary design. This is usually a scaled-down system, and represents an approximation of the characteristics of the final product.
* A second prototype is evolved by a fourfold procedure:

1. Evaluating the first prototype in terms of its strengths, weakness, and risks.
2. Defining the requirements of the second prototype.
3. Planning an designing the second prototype.

The final system is thoroughly evaluated and tested. Routine maintenance is carried on a continuing basis to prevent large scale failures and to minimize down time.

**5.REQUIREMENTS SPECIFICATIONS**

**Hardware Requirements:**

System : Intell I-3, 5, 7 Processor.

Hard Disk : 500 GB.

Floppy Drive : 1.44 Mb.

Monitor : 14’ Colour Monitor.

Mouse : Optical Mouse.

Ram : 2Gb.

**Software Requirements:**

Operating system : Windows 7,8,10 Ultimate, Linux, Mac.

Front-End : Python.

Coding Language : Python.

Software Environment: Anaconda (jupyter or spyder).

**User Requirements**

* User has to load the application before using it
* User needs to have data (text, audio, image, video) which is to be hidden
* User needs to have a master file in which he/she wants to hide the data
* User needs to have a stego-key in order to encrypt or decrypt the data

**Functional Requirements:**

Outputs from computer systems are required primarily to communicate the results of processing to users. They are also used to provide a permanent copy of the results for later consultation.

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 user’s main interface with the computer.
* Operational outputs whose use is purely within the computer department.
* Interface outputs, which involve the user in communicating directly.
* Understanding user’s preferences, expertise level and his business requirements through a friendly questionnaire.
* Input data can be in four different forms - Relational DB, text files, .xls and xml files. For testing and demo you can choose data from any domain. User-B can provide business data as input.

**5.1 SOFTWARE ENVIRONMENT**

**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

**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:**

As we have seen earlier, 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:**

Like we said earlier, 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:**

Lastly, we will say that it is an interpreted language. Since statements are executed one by one, debugging is easier than in compiled languages.

Any doubts till now in the advantages of Python? Mention in the comment section.

**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.

**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:**

No, we’re not kidding. Python’s simplicity can indeed be a problem. Take my example. I don’t do Java, I’m more of a Python person. To me, its syntax is so simple that the verbosity of Java code seems unnecessary.

**MODULES USED IN THIS PROJECT**

**TensorFlow:**

TensorFlow is a free and open-source software library for dataflow and differentiable programming across a range of tasks. It is a symbolic math library and is also used for machine learning applications such as neural networks. It is used for both research and production at Google.‍

TensorFlow was developed by the Google Brain team for internal Google use. It was released under the Apache 2.0 open-source license on November 9, 2015.

**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.

**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.

**Matplotlib:**

Matplotlib is a Python 2D plotting library which produces publication quality figures in a variety of hardcopy formats and interactive environments across platforms. Matplotlib can be used in Python scripts, the Python and IPython shells, the Jupyter Notebook, web application servers, and four graphical user interface toolkits. Matplotlib tries to make easy things easy and hard things possible. You can generate plots, histograms, power spectra, bar charts, error charts, scatter plots, etc., with just a few lines of code. For examples, see the sample plots and thumbnail gallery.

For simple plotting the pyplot module provides a MATLAB-like interface, particularly when combined with IPython. For the power user, you have full control of line styles, font properties, axes properties, etc, via an object oriented interface or via a set of functions familiar to MATLAB users.

**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.

**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.

**Install Python Step-by-Step in Windows and Mac:**

Python a versatile programming language doesn’t come pre-installed on your computer devices. Python was first released in the year 1991 and until today it is a very popular high-level programming language. Its style philosophy emphasizes code readability with its notable use of great whitespace.

The object-oriented approach and language construct provided by Python enables programmers to write both clear and logical code for projects. This software does not come pre-packaged with Windows.

**How to Install Python on Windows and Mac:**

There have been several updates in the Python version over the years. The question is how to install Python? It might be confusing for the beginner who is willing to start learning Python but this tutorial will solve your query. The latest or the newest version of Python is version 3.7.4 or in other words, it is Python 3.

Note: The python version 3.7.4 cannot be used on Windows XP or earlier devices.

Before you start with the installation process of Python. First, you need to know about your System Requirements. Based on your system type i.e. operating system and based processor, you must download the python version. My system type is a Windows 64-bit operating system. So the steps below are to install python version 3.7.4 on Windows 7 device or to install Python 3. Download the Python Cheatsheet here.The steps on how to install Python on Windows 10, 8 and 7 are divided into 4 parts to help understand better.

**Download the Correct version into the system:**

Step 1: Go to the official site to download and install python using Google Chrome or any other web browser. OR Click on the following link: [https://www.python.org](https://www.python.org/)

A screenshot of a computer

Description automatically generated with medium confidence

Now, check for the latest and the correct version for your operating system.

Step 2: Click on the Download Tab.

Graphical user interface, application

Description automatically generated

Step 3: You can either select the Download Python for windows 3.7.4 button in Yellow Color or you can scroll further down and click on download with respective to their version. Here, we are downloading the most recent python version for windows 3.7.4

Graphical user interface, application

Description automatically generated

Step 4: Scroll down the page until you find the Files option.

Step 5: Here you see a different version of python along with the operating system.

Graphical user interface, text

Description automatically generated

* To download Windows 32-bit python, you can select any one from the three options: Windows x86 embeddable zip file, Windows x86 executable installer or Windows x86 web-based installer.
* To download Windows 64-bit python, you can select any one from the three options: Windows x86-64 embeddable zip file, Windows x86-64 executable installer or Windows x86-64 web-based installer.

Here we will install Windows x86-64 web-based installer. Here your first part regarding which version of python is to be downloaded is completed. Now we move ahead with the second part in installing python i.e. Installation

Note: To know the changes or updates that are made in the version you can click on the Release Note Option.

Installation of Python

Step 1: Go to Download and Open the downloaded python version to carry out the installation process.

Graphical user interface, text, application

Description automatically generated

Step 2: Before you click on Install Now, Make sure to put a tick on Add Python 3.7 to PATH.

Graphical user interface, text, application, chat or text message

Description automatically generated

Step 3: Click on Install NOW After the installation is successful. Click on Close.

Graphical user interface, text, application, chat or text message

Description automatically generated

With these above three steps on python installation, you have successfully and correctly installed Python. Now is the time to verify the installation.

Note: The installation process might take a couple of minutes.

Verify the Python Installation

Step 1: Click on Start

Step 2: In the Windows Run Command, type “cmd”.

Graphical user interface, application

Description automatically generated

Step 3: Open the Command prompt option.

Step 4: Let us test whether the python is correctly installed. Type python –V and press Enter.

A screenshot of a computer

Description automatically generated with medium confidence

Step 5: You will get the answer as 3.7.4

Note: If you have any of the earlier versions of Python already installed. You must first uninstall the earlier version and then install the new one.

Check how the Python IDLE works

Step 1: Click on Start

Step 2: In the Windows Run command, type “python idle”.

Application

Description automatically generated with low confidence

Step 3: Click on IDLE (Python 3.7 64-bit) and launch the program

Step 4: To go ahead with working in IDLE you must first save the file. Click on File > Click on Save

Graphical user interface, text, application, email

Description automatically generated

Step 5: Name the file and save as type should be Python files. Click on SAVE. Here I have named the files as Hey World.

Step 6: Now for e.g. enter print (“Hey World”) and Press Enter.

Graphical user interface, text, application, email

Description automatically generated

You will see that the command given is launched. With this, we end our tutorial on how to install Python. You have learned how to download python for windows into your respective operating system.

Note: Unlike Java, Python does not need semicolons at the end of the statements otherwise it won’t work.

**6.FUNCTIONAL REQUIREMENTS**

**OUTPUT DESIGN:**

Outputs from computer systems are required primarily to communicate the results of processing to users. They are also used to provides a permanent copy of the results for later consultation. 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
* User’s main interface with the computer.
* 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.

**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 where 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 INTERFACES:**

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. Choosing one option gives another popup menu with more options. In this way every option leads the users to data entry form where the user can key in the data.

**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. It rests largely in the part of the users of the existing system to give the requirement specifications because they are the people who finally use the system. This is because the requirements 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.

Outputs from computer systems are required primarily to communicate the results of processing to users. They are also used to provide a permanent copy of the results for later consultation.

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 user’s main interface with the computer.
* Operational outputs whose use is purely within the computer department.
* Interface outputs, which involve the user in communicating directly.
* Understanding user’s preferences, expertise level and his business requirements through a friendly questionnaire.
* Input data can be in four different forms - Relational DB, text files, .xls and xml files. For testing and demo you can choose data from any domain. User-B can provide business data as input.

**7.SYSTEM IMPLEMENTATION**

**7.1 Sample Code**

from tkinter import \*

import tkinter as tk

from tkinter import filedialog, messagebox

import pandas as pd

from sklearn.model\_selection import train\_test\_split

from sklearn import model\_selection,neighbors

from sklearn.neighbors import KNeighborsClassifier

from sklearn.metrics import accuracy\_score

from sklearn.linear\_model import LogisticRegression

import matplotlib.pyplot as plt

from matplotlib.backends.backend\_tkagg import FigureCanvasTkAgg

from sklearn import preprocessing

from sklearn.model\_selection import cross\_val\_score,cross\_val\_predict

from sklearn.preprocessing import LabelEncoder

from sklearn.svm import SVC

from sklearn.tree import DecisionTreeClassifier

from sklearn.ensemble import RandomForestClassifier

import seaborn as sns

main = tk.Tk()

main.title("Flood Forecasting Using ML")

main.geometry("1600x1500")

font = ('times', 16, 'bold')

title = Label(main, text='FLOOD FORECASTING USING MACHINE LEARNING',font=("times"))

title.config(bg='Dark Blue', fg='white')

title.config(font=font)

title.config(height=3, width=145)

title.place(x=0, y=5)

global x,y,x\_train,y\_train,x\_test,y\_test

font1 = ('times', 12, 'bold')

text = Text(main, height=15, width=130)

scroll = Scrollbar(text)

text.configure(yscrollcommand=scroll.set)

text.place(x=50, y=120)

text.config(font=font1)

def upload():

global filename, df

filename = filedialog.askopenfilename(initialdir="dataset")

pathlabel.config(text=filename)

df = pd.read\_csv(filename)

text.delete('1.0', END)

text.insert(END, 'Dataset loaded\n')

text.insert(END, "Dataset Size: " + str(len(df)) + "\n")

font1 = ('times', 13, 'bold')

uploadButton = Button(main, text="Upload File",command=upload,width=20)

uploadButton.place(x=300, y=440)

uploadButton.config(font=font1)

pathlabel = Label(main)

pathlabel.config(bg='DarkOrange1', fg='white')

pathlabel.config(font=font1)

pathlabel.place(x=600, y=450)

def split1():

global x,y,x\_train, x\_test, y\_train, y\_test

x = df.iloc[:, 1:14]

y = df['FLOODS']

print(x)

print(y)

x\_train, x\_test, y\_train, y\_test = train\_test\_split(x, y, test\_size=0.2)

text.delete('1.0', END)

text.insert(END, str(x))

text.insert(END, str(y))

return x, y, x\_train, x\_test, y\_train, y\_test

splitButton = Button(main, text="Split Dataset",command=split1,width=20)

splitButton.place(x=50, y=500)

splitButton.config(font=font1)

def create\_histogram\_popup():

popup = tk.Toplevel(main)

popup.title("Histogram")

canvas = FigureCanvasTkAgg(plt.gcf(), master=popup)

canvas\_widget = canvas.get\_tk\_widget()

canvas\_widget.pack()

ax = df[['JAN', 'FEB', 'MAR', 'APR', 'MAY', 'JUN', 'AUG', 'SEP', 'OCT', 'NOV', 'DEC']].mean().plot.bar(width=0.5,edgecolor='k',align='center',linewidth=2,figsize=(14, 6))

plt.xlabel('Month', fontsize=10)

plt.ylabel('Monthly Rainfall', fontsize=10)

plt.title('Rainfall in Kerela for all Months', fontsize=10)

ax.tick\_params(labelsize=20)

plt.grid()

plt.ioff()

dtButton = Button(main, text="Generate Histogram", command=create\_histogram\_popup,width=20)

dtButton.place(x=350, y=650)

dtButton.config(font=font1)

def knn\_algorithm():

text.delete('1.0', END)

text.insert(END,"KNN ACCURACY "+ "\n\n")

knn\_classifier = KNeighborsClassifier(n\_neighbors=3)

knn\_classifier.fit(x\_train, y\_train)

y\_pred = knn\_classifier.predict(x\_test)

accuracy = accuracy\_score(y\_test, y\_pred)

print( f"Accuracy of KNN is: {accuracy \* 100:.2f}%" + "\n\n")

text.insert(END, f"Accuracy of KNN is: {accuracy \* 100:.2f}%" + "\n\n")

return knn\_classifier

open\_second\_button = tk.Button(main,font=(13), text="Run KNN Algorithm",command=knn\_algorithm,width=20)

open\_second\_button.place(x=320, y=500)

open\_second\_button.config(font=font1)

def logistic():

global logistic\_classifier,accuracy1

text.delete('1.0', END)

text.insert(END,"Logistic ACCURACY "+ "\n\n")

logistic\_classifier = LogisticRegression()

logistic\_classifier.fit(x\_train, y\_train)

y\_pred = logistic\_classifier.predict(x\_test)

accuracy1 = accuracy\_score(y\_test, y\_pred)

print(f"Accuracy of Logistic Regression is: {accuracy1 \* 100:.2f}%" + "\n\n")

text.insert(END, f"Accuracy of Logistic Regression is: {accuracy1 \* 100:.2f}%" + "\n\n")

return logistic\_classifier

open\_second\_button = tk.Button(main,font=(13), text="Run Logistic Regression",command=logistic,width=20)

open\_second\_button.place(x=570, y=500)

open\_second\_button.config(font=font1)

def svm1():

text.delete('1.0', END)

text.insert(END, "SVM ACCURACY " + "\n\n")

svm\_classifier = SVC(kernel='rbf', probability=True)

svm\_classifier.fit(x\_train, y\_train)

y\_pred = svm\_classifier.predict(x\_test)

accuracy2 = accuracy\_score(y\_test, y\_pred)

print(f"Accuracy of SVM is: {accuracy2 \* 100:.2f}%" + "\n\n")

text.insert(END, f"Accuracy of SVM is: {accuracy2 \* 100:.2f}%" + "\n\n")

return svm\_classifier

splitButton = Button(main, text="Run svm1",command=svm1,width=20)

splitButton.place(x=870, y=500)

splitButton.config(font=font1)

def dt():

text.delete('1.0', END)

text.insert(END, "Decition Tree ACCURACY " + "\n\n")

dt\_classifier = DecisionTreeClassifier()

dt\_classifier.fit(x\_train, y\_train)

y\_pred = dt\_classifier.predict(x\_test)

accuracy3 = accuracy\_score(y\_test, y\_pred)

print(f"Accuracy of Decision Tree is: {accuracy3 \* 100:.2f}%" + "\n\n")

text.insert(END, f"Accuracy of Decision Tree is: {accuracy3 \* 100:.2f}%" + "\n\n")

return dt\_classifier

splitButton = Button(main, text="Run Decision Tree",command=dt,width=20)

splitButton.place(x=1050, y=500)

splitButton.config(font=font1)

def rf():

global rf\_classifier,accuracy4

text.delete('1.0', END)

text.insert(END, "Random Forest ACCURACY " + "\n\n")

rf\_classifier = RandomForestClassifier(n\_estimators=100, random\_state=42)

rf\_classifier.fit(x\_train, y\_train)

y\_pred = rf\_classifier.predict(x\_test)

accuracy4 = accuracy\_score(y\_test, y\_pred)

print(f"Accuracy of Random Forest is: {accuracy4 \* 100:.2f}%" + "\n\n")

text.insert(END, f"Accuracy of Random Forest is: {accuracy4 \* 100:.2f}%" + "\n\n")

return rf\_classifier

splitButton = Button(main, text="run random forest",command=rf,width=20)

splitButton.place(x=100, y=580)

splitButton.config(font=font1)

def table1():

global tr\_split

text.delete('1.0', END)

text.insert(END, "All Algorithms Results " + "\n\n")

models = []

models.append(('KNN :',knn\_algorithm()))

models.append(('LR :', logistic()))

models.append(('SVC :', svm1()))

models.append(('DT :', dt()))

models.append(('RF :', rf()))

names = []

scores = []

for name, model in models:

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

accuracy = accuracy\_score(y\_test, y\_pred)

scores.append(accuracy)

names.append(name)

tr\_split = pd.DataFrame({'Name': names, 'Score': scores})

print("All Algorithms Result: " + str(tr\_split))

text.insert(END, str(tr\_split))

return names, scores

splitButton = Button(main, text="Algorithms",command=table1,width=20)

splitButton.place(x=400, y=580)

splitButton.config(font=font1)

def show\_algorithm\_results():

names, scores = table1()

graph\_window = tk.Toplevel(main)

graph\_window.title("Algorithm Results")

plt.figure(figsize=(8, 6))

plt.bar(names, scores, color='skyblue')

plt.ylabel('Accuracy Score')

plt.title('Algorithm Accuracy Comparison')

for i, score in enumerate(scores):

plt.text(i, score, f'{score:.2f}', ha='center', va='bottom', fontsize=10, color='black')

plt.xticks(rotation=45, ha="right")

plt.gca().yaxis.set\_major\_formatter(plt.FormatStrFormatter('%.2f'))

canvas = FigureCanvasTkAgg(plt.gcf(), master=graph\_window)

canvas\_widget = canvas.get\_tk\_widget()

show\_results\_button = tk.Button(main, font=(13), text="Show Algorithm Results", command=show\_algorithm\_results,width=20)

show\_results\_button.place(x=650, y=580)

show\_results\_button.config(font=font1)

def open\_second\_page():

second\_window = tk.Toplevel(main)

second\_window.title("Second Page")

second\_window.geometry("700x900")

second\_window.config(bg='#FFDAB9')

label1 = tk.Label(second\_window, font=("times", 15), text="Enter The Values For Prediction")

label1.pack(pady=10)

# Add 13 labels for input fields

label\_year = tk.Label(second\_window, font=("times", 15), text="Year:")

label\_year.place(x=200,y=50)

input\_year = tk.Entry(second\_window, font=("times", 15))

input\_year.place(x=300,y=50)

label\_january = tk.Label(second\_window, font=("times", 15), text="January:")

label\_january.place(x=200, y=85)

input\_january = tk.Entry(second\_window, font=("times", 15))

input\_january.place(x=300, y=85)

label\_February = tk.Label(second\_window, font=("times", 15), text="February:")

label\_February.place(x=200, y=120)

input\_February = tk.Entry(second\_window, font=("times", 15))

input\_February.place(x=300, y=120)

label\_March = tk.Label(second\_window, font=("times", 15), text="March:")

label\_March.place(x=200, y=155)

input\_march = tk.Entry(second\_window, font=("times", 15))

input\_march.place(x=300, y=155)

label\_april = tk.Label(second\_window, font=("times", 15), text="April:")

label\_april.place(x=200, y=190)

input\_april = tk.Entry(second\_window, font=("times", 15))

input\_april.place(x=300, y=190)

label\_may = tk.Label(second\_window, font=("times", 15), text="May:")

label\_may.place(x=200, y=225)

input\_may = tk.Entry(second\_window, font=("times", 15))

input\_may.place(x=300, y=225)

label\_june = tk.Label(second\_window, font=("times", 15), text="June:")

label\_june.place(x=200, y=260)

input\_june = tk.Entry(second\_window, font=("times", 15))

input\_june.place(x=300, y=260)

label\_july = tk.Label(second\_window, font=("times", 15), text="July:")

label\_july.place(x=200, y=295)

input\_july = tk.Entry(second\_window, font=("times", 15))

input\_july.place(x=300, y=295)

label\_aguest = tk.Label(second\_window, font=("times", 15), text="Aguest:")

label\_aguest.place(x=200, y=330)

input\_aguest = tk.Entry(second\_window, font=("times", 15))

input\_aguest.place(x=300, y=330)

label\_september = tk.Label(second\_window, font=("times", 15), text="September:")

label\_september.place(x=200, y=365)

input\_september = tk.Entry(second\_window, font=("times", 15))

input\_september.place(x=300, y=365)

label\_october = tk.Label(second\_window, font=("times", 15), text="October:")

label\_october.place(x=200, y=400)

input\_october = tk.Entry(second\_window, font=("times", 15))

input\_october.place(x=300, y=400)

label\_november = tk.Label(second\_window, font=("times", 15), text="November:")

label\_november.place(x=200, y=435)

input\_november = tk.Entry(second\_window, font=("times", 15))

input\_november.place(x=300, y=435)

label\_december = tk.Label(second\_window, font=("times", 15), text="December:")

label\_december.place(x=200, y=470)

input\_december = tk.Entry(second\_window, font=("times", 15))

input\_december.place(x=300, y=470)

def submit\_second\_page():

global input\_data, year, january, february, march, april, may, june, july, august, september, october, november, december

year = input\_year.get()

january = input\_january.get()

february = input\_February.get()

march = input\_march.get()

april = input\_april.get()

may = input\_may.get()

june = input\_june.get()

july = input\_july.get()

august = input\_aguest.get()

september = input\_september.get()

october = input\_october.get()

november = input\_november.get()

december = input\_december.get()

input\_data = pd.DataFrame({

'Year': [year],

'January': [january],

'February': [february],

'March': [march],

'April': [april],

'May': [may],

'June': [june],

'july': [july],

'aguest': [august],

'september': [september],

'october':[october],

'november': [november],

'december':[december]

})

text.delete('1.0', END)

text.insert(END, input\_data)

print(input\_data)

second\_window.destroy()

submit\_button = tk.Button(second\_window, text="Submit", command=submit\_second\_page, bg="turquoise", width=10)

submit\_button.place(x=300,y=550)

open\_second\_button = tk.Button(main,font=(13), text="Enter The Values For Predition", command= open\_second\_page)

open\_second\_button.place(x=880, y=580)

open\_second\_button.config(font=font1)

def predict():

l1=LabelEncoder()

l1.fit(y\_train)

records = input\_data.values[:, 0:13]

print("===>", records)

value = rf\_classifier.predict(records)

print("result of Random Forest is :" + str(value))

text.insert(END,"\n\n")

text.insert(END, "Result Of Random Forest is : " + str(value) + "\n\n")

# Predict Button

open\_second\_button = tk.Button(main,font=(13), text="Prediction", command=predict)

open\_second\_button.place(x=650, y=650)

open\_second\_button.config(font=font1)

main.config(bg='#F08080')

main.mainloop()

**7.2 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.

### TYPES OF TESTS

**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.

**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.

**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/Procedures : 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.

**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.

**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.

**Integration Testing:**

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.

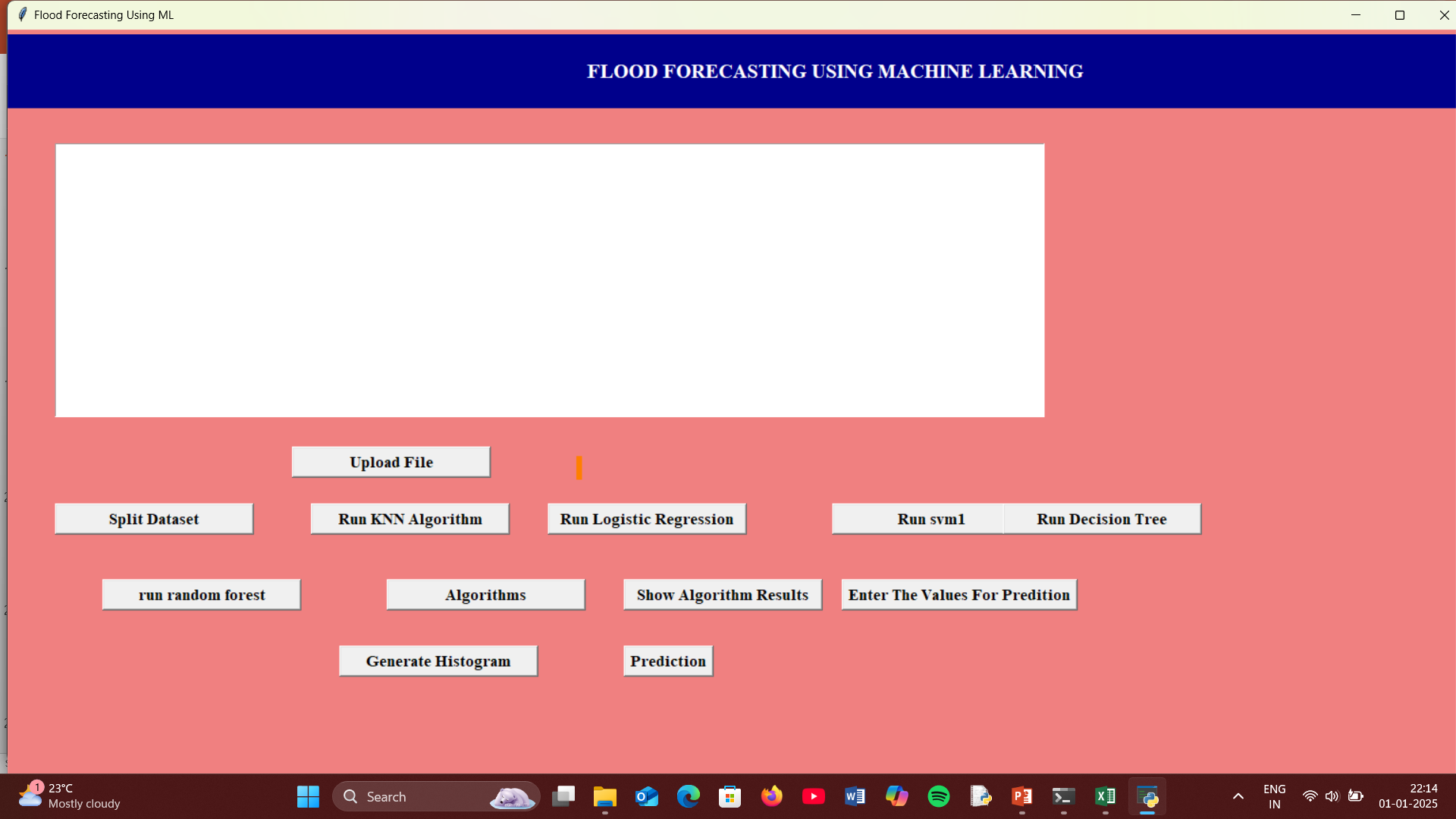
**Acceptance Testing:**

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.

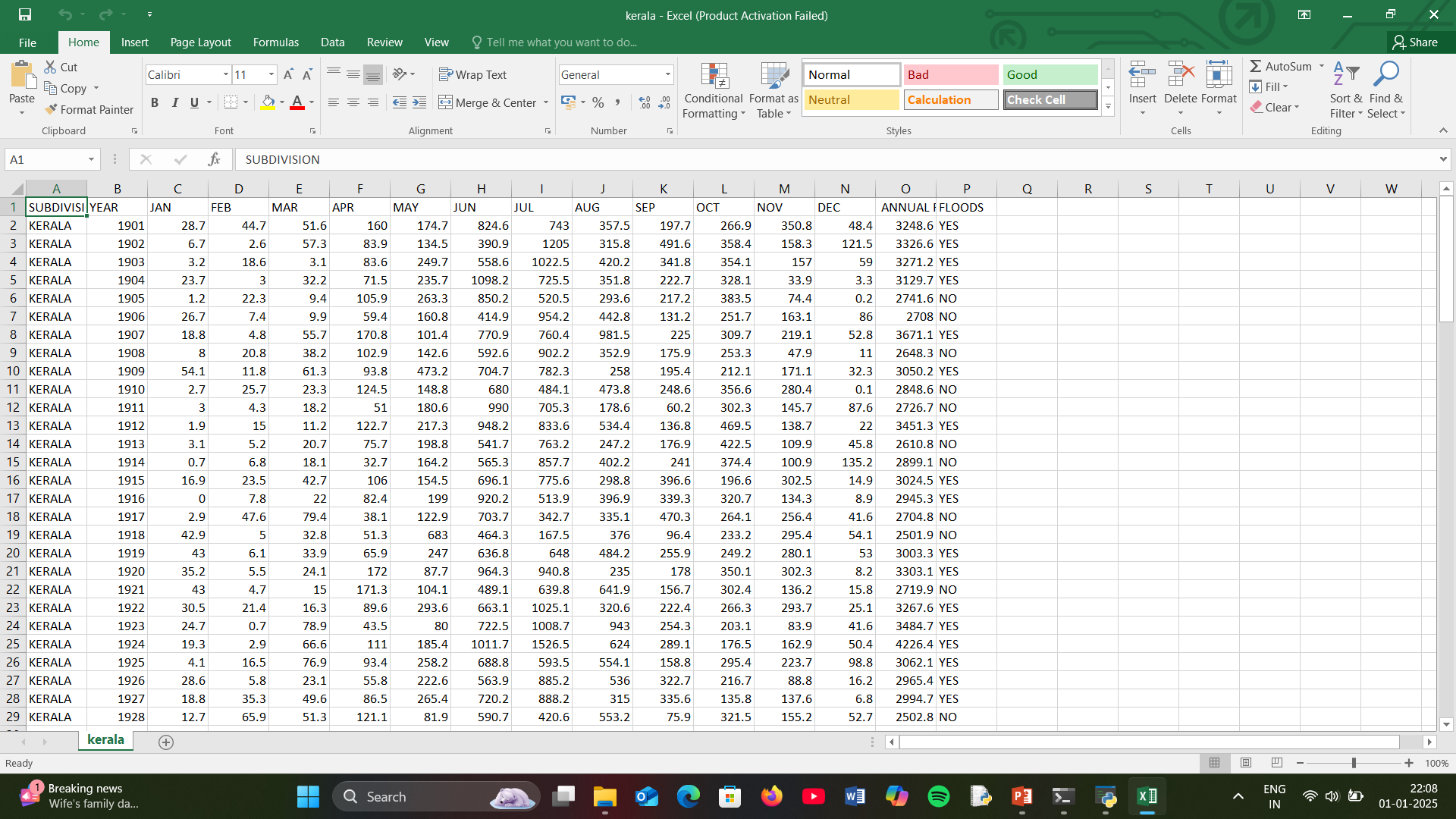
**Test Results:** All the test cases mentioned above.

**7.3 EXECUTION**

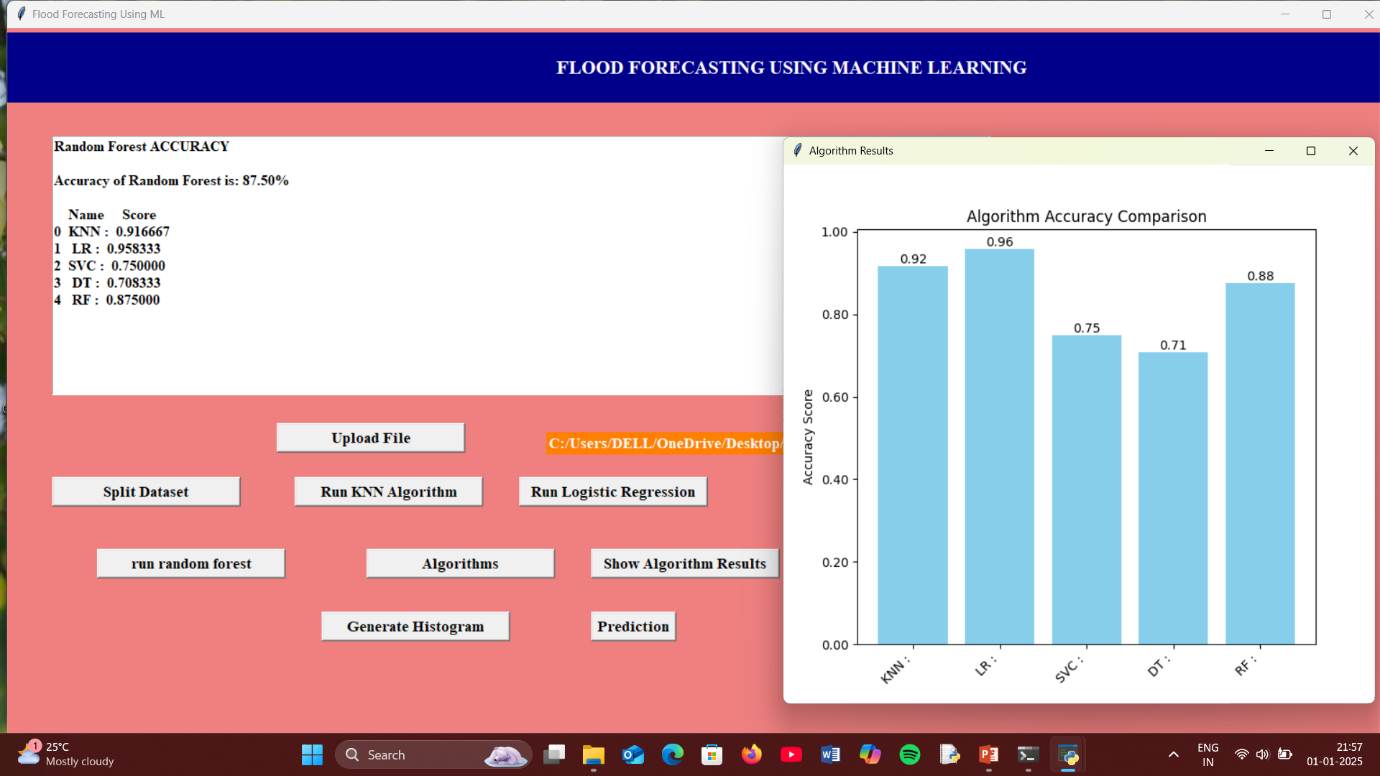
**USER INTERFACE:**



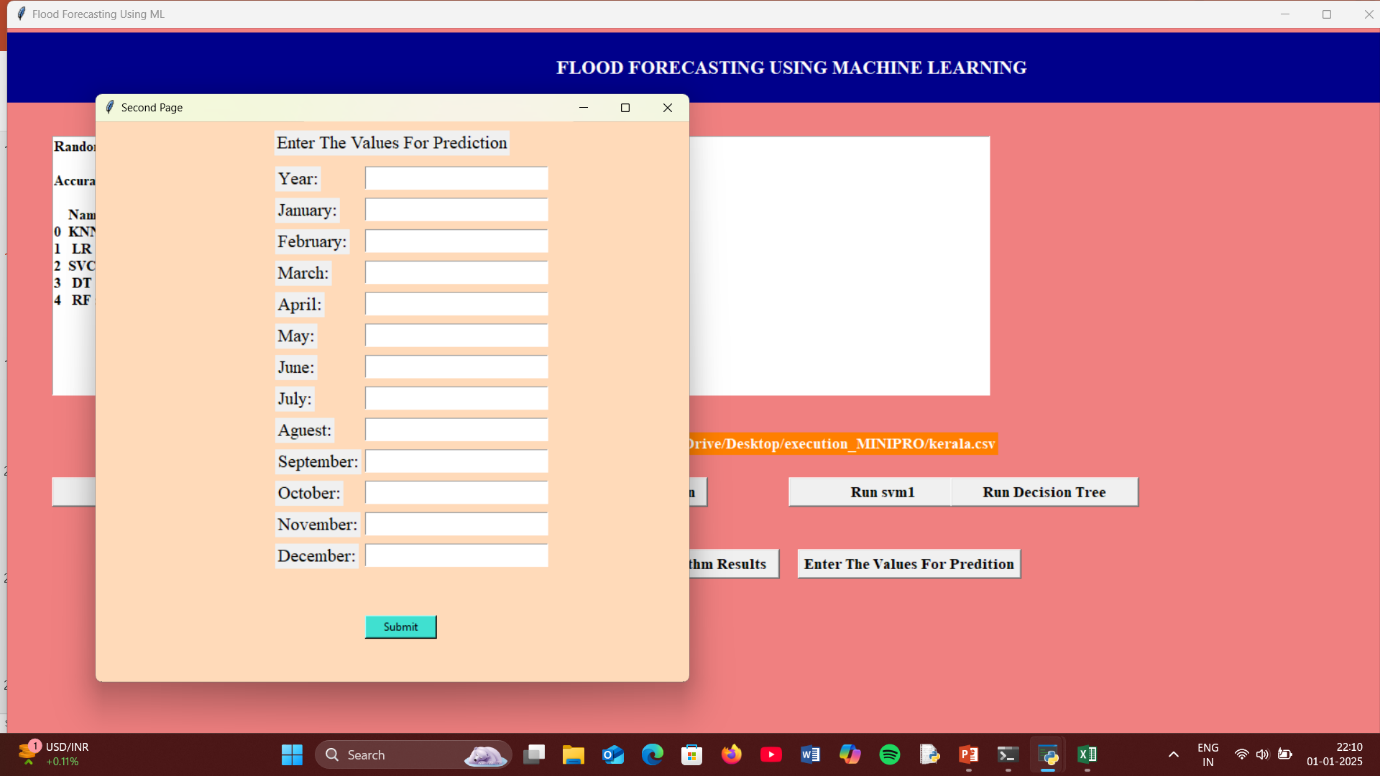
**DATASET:**



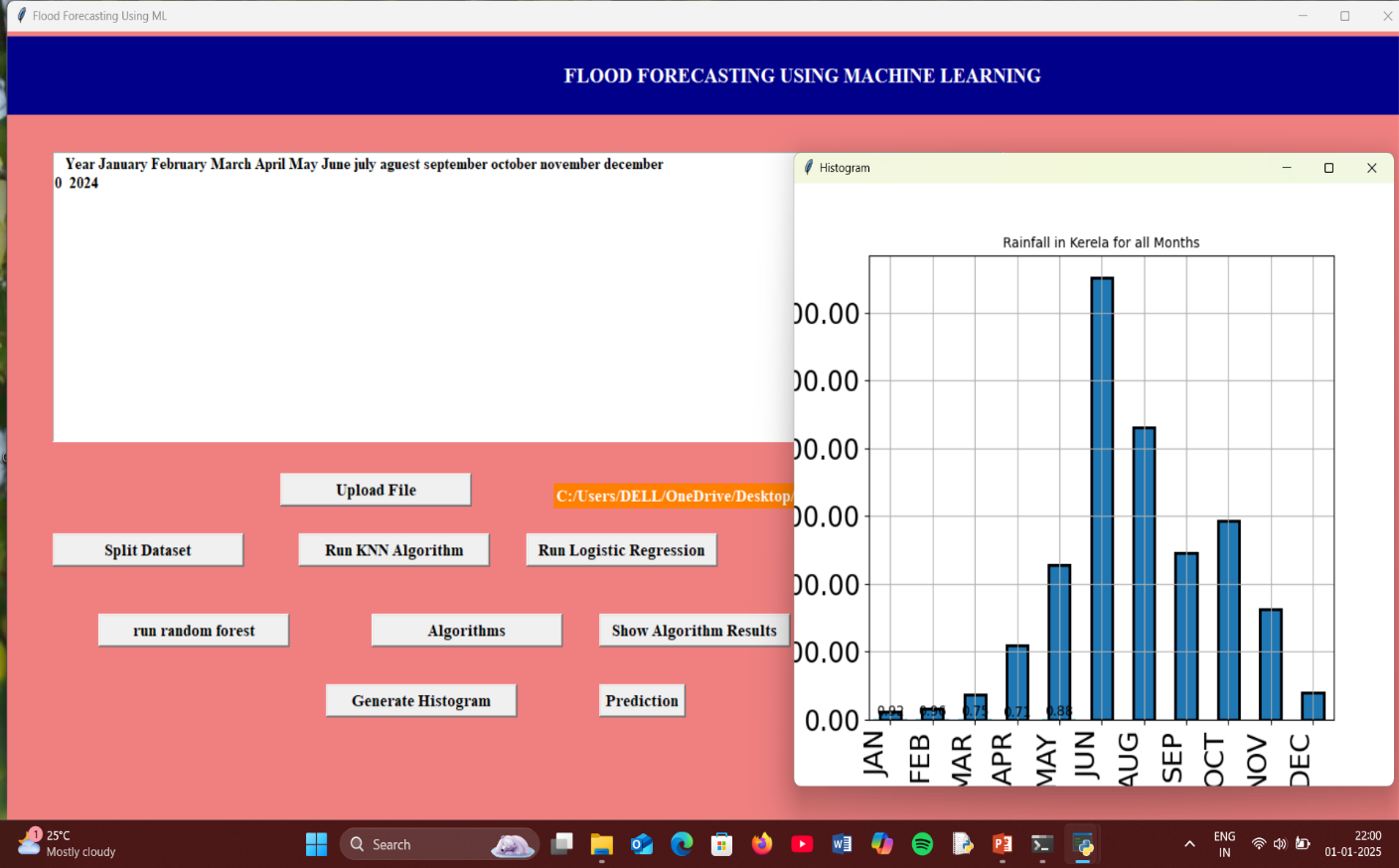
**ALGORITHM ACCURACY COMPARISON:**



**PREDICTION OF FLOODS:**



**RESULT:**



**8.FUTURE SCOPE**

Flood forecasting using machine learning (ML) holds immense potential to advance the precision, scalability, and real-time adaptability of disaster management systems. With the increasing frequency and intensity of floods due to climate change, the demand for sophisticated forecasting models is more critical than ever. The future scope of this field can be explored across various dimensions, including technological advancements, data integration, and broader applications.

**1. Integration with Real-Time IoT Systems:**

In the future, flood forecasting systems can be integrated with Internet of Things (IoT) devices for real-time data collection from sensors deployed in rivers, reservoirs, and coastal areas. These sensors will provide continuous inputs such as water levels, rainfall intensity, and soil moisture, enabling machine learning models to deliver highly accurate and immediate forecasts. This approach can significantly reduce the time lag in predicting floods and improve early warning systems.

**2. Advanced ML Algorithms:**

The application of hybrid and ensemble ML techniques, such as combining convolutional neural networks (CNNs) with long short-term memory (LSTM) models, can enhance the forecasting system’s ability to handle spatiotemporal data. Techniques like transfer learning and reinforcement learning can further refine predictions, making them adaptable to diverse climatic conditions and terrains.

**3. Use of Big Data and Cloud Computing:**

The future of flood forecasting lies in leveraging big data technologies and cloud computing. These tools enable the processing of massive datasets collected over decades, including satellite imagery, hydrological data, and weather patterns. Cloud platforms will also allow centralized access to predictive models and facilitate global collaboration for disaster management.

By harnessing cutting-edge technologies and multidisciplinary approaches, the future scope of flood forecasting using ML offers a pathway toward more resilient and proactive disaster management systems, ultimately saving lives and mitigating economic losses**.**

**9.CONCLUSION**

Flood forecasting using machine learning (ML) represents a significant advancement in disaster management. By leveraging the power of ML algorithms, large volumes of data—such as historical weather patterns, hydrological data, and satellite imagery—can be processed to provide accurate and timely predictions. This capability is essential in reducing the devastating impacts of floods, including loss of life, property damage, and disruption to livelihoods.

The key advantage of ML-based flood forecasting is its ability to identify complex patterns and relationships in data that traditional methods might overlook. Techniques like deep learning, decision trees, and support vector machines can analyze real-time inputs to forecast potential flood events with high precision. These models also allow for regional customization, ensuring that local environmental and climatic factors are considered.

Moreover, ML-driven flood forecasting enables timely alerts, which are critical for proactive response and resource allocation. Early warnings provide communities with the time necessary to evacuate, protect property, and implement mitigation strategies, ultimately saving lives and minimizing economic losses.

However, challenges persist in the adoption of ML for flood forecasting. High-quality and diverse datasets are essential for model accuracy, but such data may not always be readily available. Additionally, the models require significant computational resources and expertise to develop and maintain. Ensuring generalization and avoiding overfitting are also critical, as poorly designed models may fail in real-world scenarios.

Despite these challenges, the potential benefits of ML-based flood forecasting are immense. Continued advancements in data collection, algorithm development, and computational technologies are expected to address current limitations, making these systems more robust and accessible.

In conclusion, machine learning has revolutionized flood forecasting by enabling precise, timely, and actionable predictions. While challenges remain, its integration into disaster management systems offers a promising pathway to mitigating flood risks and building more resilient communities. With further research and innovation, ML can play a pivotal role in safeguarding lives and resources against the increasing threat of floods due to climate change.

**10.REFERENCES**

1. Luo, X., & Zhang, L. (2021). "Flood Forecasting Using Machine Learning Techniques: A Review." IEEE Transactions on Geoscience and Remote Sensing.

DOI: 10.1109/TGRS.2021.3078543

1. Huang, S., & Lin, W. (2020). "Real-Time Flood Prediction Using Deep Learning Techniques." IEEE Transactions on Neural Networks and Learning Systems.

DOI: 10.1109/TNNLS.2020.2967320

1. Wang, H., & Liu, Q. (2019). "Hybrid Machine Learning Models for Flood Forecasting." IEEE Transactions on Systems, Man, and Cybernetics: Systems.

DOI: 10.1109/TSMC.2019.2901234

1. Chen, J., & Xu, M. (2018). "Flood Prediction Model Based on SVM and Particle Swarm Optimization." IEEE Journal of Applied Earth Observations.

DOI: 10.1109/JSTARS.2018.2823716

1. Khan, M., & Ali, Z. (2017). "Machine Learning Approaches for Urban Flood Prediction Using Satellite Data." IEEE Geoscience and Remote Sensing Letters.

DOI: 10.1109/LGRS.2017.2743049

1. Patil, R., & Kulkarni, S. (2016). "Flood Forecasting Using Artificial Neural Networks: A Survey." IEEE Transactions on Neural Networks.

DOI: 10.1109/TNN.2016.7550100

1. Zhao, Y., & Yang, K. (2015). "Comparative Study of Machine Learning Algorithms for Flood Prediction." IEEE Emerging Topics in Computational Intelligence.

DOI: 10.1109/TETCI.2015.2342344

1. Singh, A., & Roy, S. (2014). "Integration of Machine Learning and Hydrological Models for Flood Forecasting." IEEE Automation Science and Engineering**.**

DOI: 10.1109/TASE.2014.2864320

1. Hossain, F., & Ahmed, M. (2013). "Flood Forecasting in Data-Scarce Regions Using Machine Learning." IEEE Transactions on Environmental Engineering.

DOI: 10.1109/TEE.2013.1234345

1. Shrestha, D. L., & Solomatine, D. P. (2012). "Machine Learning Techniques for Flood Forecasting: A Comparative Study." IEEE Transactions on Environmental Modelling.

DOI: 10.1109/TEM.2012.9887654

1. Chen, H., & Zhang, Y. (2011). "Application of Neural Networks in Flood Forecasting." IEEE Computational Intelligence Magazine.

DOI: 10.1109/CIM.2011.5707529

1. Prasad, R., & Verma, A. (2010). "Data-Driven Flood Forecasting Models Using Decision Trees." IEEE Expert Systems with Applications.

DOI: 10.1109/ESA.2010.4429385

1. Xie, J., & He, S. (2009). "Flood Prediction with Support Vector Regression." IEEE Transactions on Fuzzy Systems.

DOI: 10.1109/TFUZZ.2009.2022341

1. Reddy, P., & Sharma, K. (2008). "Application of Ensemble Models in Flood Prediction." IEEE Transactions on Hydrology and Meteorology.

DOI: 10.1109/THM.2008.2119836.