RAINFALL PREDICTION USING MACHINE LEARNING - PYTHON

A PROJECT REPORT

Submitted by

Goli Jaya Chandran [Reg No: RA2111028010013]

Pasam Uma Maheshwar Reddy [Reg No: RA2111028010072]

Under the Guidance of

Dr. S. NAGADEVI

Assistant Professor, Department of Computing Technologies

in partial fulfillment of the requirements for the degree of

BACHELOR OF TECHNOLOGY

in

COMPUTER SCIENCE AND ENGINEERING

DEPARTMENT OF COMPUTING TECHNOLOGIES

COLLEGE OF ENGINEERING AND TECHNOLOGY

SRM INSTITUTE OF SCIENCE AND TECHNOLOGY

KATTANKULATHUR– 603 203

APRIL 2024

SRM INSTITUTE OF SCIENCE AND TECHNOLOGY

KATTANKULATHUR–603 203

BONAFIDE CERTIFICATE

Certified that **18CSP107L / 18CSP108L** project report titled "**RAINFALL PREDICTION USING MACHINE LEARNING - PYTHON**" is the bonafide work of **GOLI JAYA CHANDRAN [RegNo:RA2111028010013]** and **PASAM UMA MAHESHWAR REDDY [RegNo:RA2111028010013]** who carried out the project work under my supervision. Certified further, that to the best of my knowledge the work reported here in does not form part of any other thesis or dissertation on the basis of which a degree or award was conferred on an earlier occasion for this or any other candidate.

Dr. S. NAGADEVI

SUPERVISOR
Assistant Professor
Department of Computing Technologies

Dr. M. PUSHPALATHA
HEAD OF THE DEPARTMENT
Department of Computing Technologies

Department of Computing Technologies

SRM Institute of Science and Technology
Own Work Declaration Form

Degree/Course **:** B.Tech in Computer Science and Engineering

S**tudent Names** **:** GOLI JAYA CHANDRAN , PASAM UMA MAHESHWAR REDDY

Registration Number: RA2111028010013, RA2111028010072

Title of Work **:** RAINFALL PREDICTION USING MACHINE LEARNING - PYTHON

I/We here by certify that this assessment compiles with the University's Rules and Regulations relating to Academic misconduct and plagiarism, as listed in the University Website, Regulations, and the Education Committee guidelines.

I / We confirm that all the work contained in this assessment is our own except where indicated, and that we have met the following conditions:

- Clearly references / listed all sources as appropriate
- Referenced and put in inverted commas all quoted text (from books, web, etc.)
- Given the sources of all pictures, data etc that are not my own.
- Not made any use of the report(s) or essay(s) of any other student(s)either past or present
- Acknowledged in appropriate places any help that I have received from others (e.g fellow students, technicians, statisticians, external sources)
- Compiled with any other plagiarism criteria specified in the Course hand book / University website

I understand that any false claim for this work will be penalized in accordance with the University policies and regulations.

DECLARATION:

I am aware of and understand the University's policy on Academic misconduct and plagiarism and I certify that this assessment is my / our own work, except where indicated by referring, and that I have followed the good academic practices noted above.

Student 1 Signature:

Student 2 Signature:

Date:

If you are working in a group, please write your registration numbers and sign with the date for every student in your group.

ACKNOWLEDGEMENT

We express our humble gratitude to **Dr. C. Muthamizhchelvan**, Vice-Chancellor, SRM Institute of Science and Technology, for the facilities extended for the project work and his continued support.

We extend our sincere thanks to Dean-CET, SRM Institute of Science and Technology, **Dr. T. V. Gopal**, for his invaluable support.

We wish to thank **Dr. Revathi Venkataraman**, Professor and Chairperson, School of Computing, SRM Institute of Science and Technology, for her support throughout the project work.

We are incredibly grateful to our Head of the Department, **Dr. M. Pushpalatha,** Professor, Department of Computing Technologies, SRM Institute of Science and Technology, for her suggestions and encouragement at all the stages of the project work.

Our inexpressible respect and thanks to our guide, **Dr. S. Nagadevi** Associate Professor, Department of Computing Technologies, SRM Institute of Science and Technology, for providing us with an opportunity to pursue our project under her mentorship. She provided us with the freedom and support to explore the research topics of our interest. Her passion for solving problems and making a difference in the world has always been inspiring.

We sincerely thank all the staff and students of Computing Technologies Department, School of Computing, S.R.M Institute of Science and Technology, for their help during our project. Finally, we would like to thank our parents, family members, and friends for their unconditional love, constant support and encouragement.

GOLI JAYA CHANDRAN [Reg. No: RA2111028010013]

PASAM UMA MAHESHWAR REDDY [Reg. No: RA2111028010072]

ABSTRACT

India is an agricultural country and its economy is largely based upon crop productivity and rainfall. For analyzing the crop productivity, rainfall prediction is require and necessary to all farmers. Rainfall Prediction is the application of science and technology to predict the state of the atmosphere. It is important to exactly determine the rainfall for effective use of water resources, crop productivity and pre planning of water structures. Using different data mining techniques it can predict rainfall. Data mining techniques are used to estimate the rainfall numerically. This paper focuses some of the popular data mining algorithms for rainfall prediction. Naive Bayes, K-Nearest Neighbour algorithm, Decision Tree, Neural Network and fuzzy logic are some of the algorithms compared in this paper. From that comparison, it can analyze which method gives better accuracy for rainfall prediction.

TABLE

LIST OF FIGURES

CHAPTER 1
INTRODUCTION

1.1 BACKGROUND AND BASICS

Rainfall Prediction is one of the most challenging tasks. Though already many algorithms have being proposed but still accurate prediction of rainfall is very difficult. In an agricultural country like India, the success or failure of the crops and water scarcity in any year is always viewed with greatest concern. A small fluctuation in the seasonal rainfall can have devastating impacts on agriculture sector. Accurate rainfall prediction has a potential benefit of preventing causalities and damages caused by natural disasters. Under certain circumstances such as flood and drought, highly accurate rainfall prediction is useful for agriculture management and disaster prevention. In this paper, various algorithms have been analyzed. Data mining techniques are efficiently used in rainfall prediction.

1.2 EXISTING SYSTEM

Agriculture is the strength of our Indian economy. Farmer only depends upon monsoon to be their cultivation. The good crop productivity needs good soil, fertilizer and also good climate. Weather forecasting is the very important requirement of the each farmer. Due to the sudden changes in climate/weather, The people are suffered economically and physically. Weather prediction is one of the challenging problems in current state. The main motivation of this paper to predict the weather using various data mining techniques. Such as classification, clustering, decision tree and also neural networks. Weather related information is also called the meteorological data. In this paper the most commonly used weather parameters are rainfall, wind speed, temperature and cold.

1.3 DISADVANTAGES OF EXISTING SYSTEM

1.Classification

2.Clustering

3.Decision Tree

1.4 POPOSED SYSTEM

Rainfall is important for food production plan, water resource management and all activity plans in the nature. The occurrence of prolonged dry period or heavy rain at the critical stages of the crop growth and development may lead to significant reduce crop yield. India is an agricultural country and its economy is largely based upon crop productivity. Thus rainfall prediction becomes a significant factor in agricultural countries like India. Rainfall forecasting has been one of the most scientifically and technologically challenging problems around the world in the last century.

1.5 ADVANTAGES OF PROPOSED SYSTEM

1.Numerical Weather Pediction

2.Statistical Weather Prediction

3.Synoptic Weather Prediction

CHAPTER 2
LITERATURE SURVEY

- Pritpal Singh et al.[1] Measurable investigation shows the idea of ISMR, which can't be precisely anticipated by insights or factual information. Hence, this review exhibits the utilization of three techniques: object creation, entropy, and artificial neural network (ANN). In view of this innovation, another technique for anticipating ISMR times has been created to address the idea of ISMR. This model has been endorsed and supported by the studio and exploration data. Factual examination of different information and near investigations showing the presentation of the normal technique
- Sam Carmer , Michael Kampouridis, Alex A. Freitas , Antonios Alexandridis et al.[2] The primary impact of this movement is to exhibit the advantages of AI calculations, just as the more prominent degree of clever framework than the advanced rainfall determining methods. We analyze and think about the momentum execution (Markov chain stretched out by rainfall research) with the forecasts of the six most notable AI machines: Genetic programming, Vector relapse support, radio organizations, M5 organizations, M5 models, models - Happy. To work with a more itemized appraisal, we led a rainfall overview utilizing information from 42 metropolitan urban communities.

- Sahar Hadi Poura , Shamsuddin Shahida, Eun-Sung chungb et al. [3] RF was utilized to anticipate assuming that it would rain in one day, while SVM was utilized to foresee downpour on a blustery day. The limit of the Hybrid model was fortified by the decrease of day-by-day rainfall in three spots at the rainfall level in the eastern piece of Malaysia. Crossover models have likewise been found to emulate the full change, the quantity of days straight, 95% of the month-to-month rainfall, and the dispersion of the noticed rainfall

.

- Tanvi Patil, Dr. Kamal Shah et al. [4] The reason for the framework is to anticipate the climate sooner or later. Climatic still up in the air utilizing various sorts of factors all over the place. Of these, main the main highlights are utilized in climate conjectures. Picking something like this relies a great deal upon the time you pick. Underlying displaying is utilized to incorporate the fate of demonstrating, AI applications, data trade, and character examination.

- N.Divya Prabha, P. Radha et al. [5] Contrasted with different spots where rainfall information isn't accessible, it consumes a large chunk of the day to build up a solid water overview for a long time. Improving complex neural organizations is intended to be a brilliant instrument for anticipating the stormy season. This downpour succession was affirmed utilizing a complex perceptron neural organization. Estimations like MSE (Early Modeling), NMSE (Usually Early Error), and the arrangement of informational collections for transient arranging are clear in the examination of different organizations, like Adanaive. AdaSVM.
- Senthamil Selvi S, Seetha et al. [6] In this paper, Artificial Neural Network (ANN) innovation is utilized to foster a climate anticipating strategy to distinguish rainfall utilizing Indian rainfall information. Along these lines, Feed Forward Neural Network (FFNN) was utilized utilizing the Backpropagation Algorithm. Execution of the two models is assessed dependent on emphasis examination, Mean Square Error (MSE) and Magnitude of Relative Error (MRE). This report likewise gives a future manual for rainfall determining.

- YashasAthreya, VaishaliBV, SagarK and SrinidhiHR, et al.[7] This page features rainfall investigation speculations utilizing Machine Learning. The principle motivation behind utilizing this program is to secure against the impacts of floods. This program can be utilized by conventional residents or the public authority to anticipate what will occur before the flood. The flood card, then, at that point, furnish them with the vital help by moving versatile or other important measures.

CHAPTER 3
METHODOLOGY

3.1 System Requirement Specification

3.1.1 HARDWARE REQUIREMENTS:

System	- Windows7/10
Speed	- 2.4GHZ
Hard disk	- 40GB
Monitor	- 15VGA Color
Ram	- 4GB

3.1.2 SOFTWARE REQUIREMENTS:

Coding Language	- PYTHON
IDE	- PYCHARM

3.2 SOFTWARE ENVIRONMENT

Python:

Python is a high-level, interpreted, interactive and object-oriented scripting language. Python is designed to be highly readable. It uses English keywords frequently where as other languages use punctuation, and it has fewer syntactical constructions than other languages.

- **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 is Object-Oriented** − Python supports Object-Oriented style or technique of programming that encapsulates code within objects.

- **Python is a Beginner's Language** − Python is a great language for the beginner-level programmers and supports the development of a wide range of applications from simple text processing to WWW browsers to games.

Python is derived from many other languages, including ABC, Modula-3, C, C++, Algol-68, SmallTalk, and Unix shell and other scripting languages.

Python is copyrighted. Like Perl, Python source code is now available under the GNU General Public License (GPL).

Python is now maintained by a core development team at the institute, although Guido van Rossum still holds a vital role in directing its progress.

Python Features

Python's features include −

- **Easy-to-learn** − Python has few keywords, simple structure, and a clearly defined syntax. This allows the student to pick up the language quickly.
- **Easy-to-read** − Python code is more clearly defined and visible to the eyes.
- **Easy-to-maintain** − Python's source code is fairly easy-to-maintain.
- **A broad standard library** − Python's bulk of the library is very portable and cross-platform compatible on UNIX, Windows, and Macintosh.
- **Interactive Mode** − Python has support for an interactive mode which allows interactive testing and debugging of snippets of code.
- **Portable** − Python can run on a wide variety of hardware platforms and has the same interface on all platforms.
- **Extendable** − You can add low-level modules to the Python interpreter. These modules enable programmers to add to or customize their tools to be more efficient.
- **Databases** − Python provides interfaces to all major commercial databases.
- **GUI Programming** − Python supports GUI applications that can be created and ported to many system calls, libraries and windows systems, such as Windows MFC, Macintosh, and the X Window system of Unix.
- **Scalable** − Python provides a better structure and support for large programs than shell scripting.

Getting Python

The most up-to-date and current source code, binaries, documentation, news, etc., is available on the official website of Python https://www.python.org.

Windows Installation

Here are the steps to install Python on Windows machine.

- Open a Web browser and go to https://www.python.org/downloads/.
- Follow the link for the Windows installer python-XYZ.msifile where XYZ is the version you need to install.
- To use this installer python-XYZ.msi, the Windows system must support Microsoft Installer 2.0. Save the installer file to your local machine and then run it to find out if your machine supports MSI.
- Run the downloaded file. This brings up the Python install wizard, which is really easy to use. Just accept the default settings, wait until the install is finished, and you are done.

The Python language has many similarities to Perl, C, and Java. However, there are some definite differences between the languages.

FirstPythonProgram

Let us execute programs in different modes of programming.

Interactive Mode Programming

Invoking the interpreter without passing a script file as a parameter brings up the following prompt −

```
$ python

Python2.4.3(#1,Nov112010,13:34:43)

[GCC 4.1.220080704(RedHat4.1.2-48)] on linux2

Type"help","copyright","credits"or"license"for more information.

>>>
```

Type the following text at the Python prompt and press the Enter −

```
>>>print"Hello, Python!"
```

If you are running new version of Python, then you would need to use print statement with parenthesis as in **print ("Hello, Python!");**. However in Python version 2.4.3, this produces the following result −

```
Hello, Python!
```

Script Mode Programming

Invoking the interpreter with a script parameter begins execution of the script and continues until the script is finished. When the script is finished, the interpreter is no longer active.

Let us write a simple Python program in a script. Python files have extension **.py**. Type the following source code in a test.py file −

```
print"Hello, Python!"
```

We assume that you have Python interpreter set in PATH variable. Now, try to run this program as follows −

```
$ python test.py
```

This produces the following result −

```
Hello, Python!
```

Flask Framework:

Flask is a web application framework written in Python. Armin Ronacher, who leads an international group of Python enthusiasts named Pocco, develops it. Flask is based on Werkzeug WSGI toolkit and Jinja2 template engine. Both are Pocco projects.

Http protocol is the foundation of data communication in world wide web. Different methods of data retrieval from specified URL are defined in this protocol.

The following table summarizes different http methods –

Sr.No	Methods & Description
1	**GET** Sends data in unencrypted form to the server. Most common method.
2	**HEAD** Same as GET, but without response body
3	**POST** Used to send HTML form data to server. Data received by POST method is not cached by server.
4	**PUT** Replaces all current representations of the target resource with the uploaded content.
5	**DELETE** Removes all current representations of the target resource given by a URL

By default, the Flask route responds to the **GET** requests. However, this preference can be altered by providing methods argument to **route()** decorator.

In order to demonstrate the use of **POST** method in URL routing, first let us create an HTML form and use the **POST** method to send form data to a URL.

Save the following script as login.html

```
<html>
```

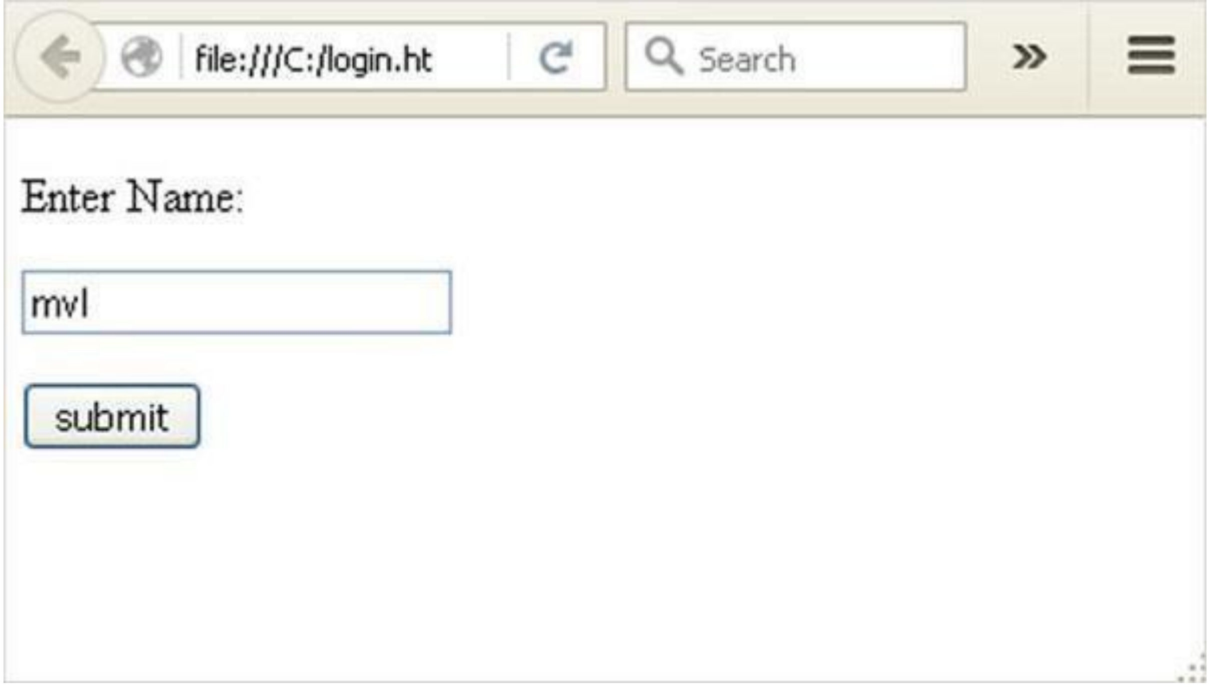
```
<body>

<formaction="http://localhost:5000/login"method="post">

<p>Enter Name:</p>

<p><inputtype="text"name="nm"/></p>

<p><inputtype="submit"value="submit"/></p>

</form>

</body>

</html>
```

Now enter the following script in Python shell.

```
from flask importFlask, redirect,url_for, request

app=Flask(__name__)

@app.route('/success/<name>')

def success(name):

return'welcome %s'% name

@app.route('/login',methods=['POST','GET'])

def login():

ifrequest.method=='POST':

user=request.form['nm']

return redirect(url_for('success',name= user))

else:

user=request.args.get('nm')
```

```
return redirect(url_for('success',name= user))

if__name__=='__main__':

app.run(debug =True)
```

After the development server starts running, open **login.html** in the browser, enter name in the text field and click **Submit**.

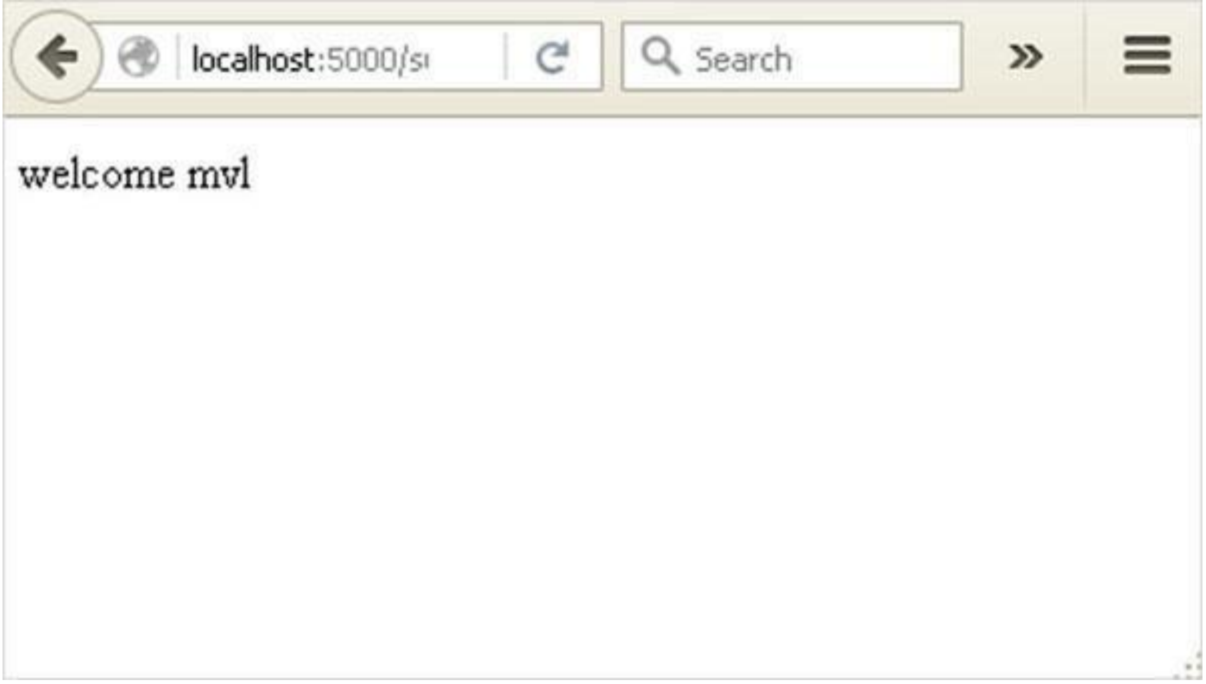


Form data is POSTed to the URL in action clause of form tag.

http://localhost/login is mapped to the **login()** function. Since the server has received data by **POST** method, value of „nm" parameter obtained from the form data is obtained by –

```
user = request.form['nm']
```

It is passed to **„/success"** URL as variable part. The browser displays a **welcome** message in the window.



welcome mvl

Change the method parameter to **„GET"** in **login.html** and open it again in the browser. The data received on server is by the **GET** method. The value of „nm" parameter is now obtained by −

```
User = request.args.get(„nm")
```

Here, **args** is dictionary object containing a list of pairs of form parameter and its corresponding value. The value corresponding to „nm" parameter is passed on to „/success" URL as before.

Python Install

Many PCs and Macs will have python already installed.

To check if you have python installed on a Windows PC, search in the start bar for Python or run the following on the Command Line (cmd.exe):

```
C:\Users\Your Name>python --version
```

To check if you have python installed on a Linux or Mac, then on linux open the command line or on Mac open the Terminal and type:

```
python --version
```

If you find that you do not have python installed on your computer, then you can download it for free from the following website: https://www.python.org/

Python Quickstart

Python is an interpreted programming language, this means that as a developer you write Python (.py) files in a text editor and then put those files into the python interpreter to be executed.

The way to run a python file is like this on the command line:

```
C:\Users\Your Name>python helloworld.py
```

Where "helloworld.py" is the name of your python file.

Let's write our first Python file, called helloworld.py, which can be done in any text editor.

```
helloworld.py
```

```
print("Hello, World!")
```

Simple as that. Save your file. Open your command line, navigate to the directory where you saved your file, and run:

```
C:\Users\Your Name>python helloworld.py
```

The output should read:

```
Hello, World!
```

Congratulations, you have written and executed your first Python program.

The Python Command Line

To test a short amount of code in python sometimes it is quickest and easiest not to write the code in a file. This is made possible because Python can be run as a command line itself.

Type the following on the Windows, Mac or Linux command line:

```
C:\Users\Your Name>python
```

From there you can write any python, including our hello world example from earlier in the tutorial:

```
C:\Users\Your Name>python
Python 3.6.4 (v3.6.4:d48eceb, Dec 19 2017, 06:04:45) [MSC v.1900 32 bit (Intel)] on win32
Type "help", "copyright", "credits" or "license" for more information.
>>> print("Hello, World!")
```

Which will write "Hello, World!" in the command line:

```
C:\Users\Your Name>python
Python 3.6.4 (v3.6.4:d48eceb, Dec 19 2017, 06:04:45) [MSC v.1900 32 bit (Intel)] on win32
Type "help", "copyright", "credits" or "license" for more information.
>>> print("Hello, World!")
Hello, World!
```

Whenever you are done in the python command line, you can simply type the following to quit the python command line interface:

```
exit()
```

Execute Python Syntax

As we learned in the previous page, Python syntax can be executed by writing directly in the Command Line:

```
>>> print("Hello, World!")
Hello, World!
```

Or by creating a python file on the server, using the .py file extension, and running it in the Command Line:

```
C:\Users\Your Name>python myfile.py
```

Python Indentations

Where in other programming languages the indentation in code is for readability only, in Python the indentation is very important.

Python uses indentation to indicate a block of code.

Example

```
if 5 > 2:
  print("Five is greater than two!")
```

Python will give you an error if you skip the indentation:

Example

```
if 5 > 2:
print("Five is greater than two!")
```

Comments

Python has commenting capability for the purpose of in-code documentation.

Comments start with a #, and Python will render the rest of the line as a comment:

Example

Comments in Python:

```
#This is a comment.
print("Hello, World!")
```

Docstrings

Python also has extended documentation capability, called docstrings.

Docstrings can be one line, or multiline.

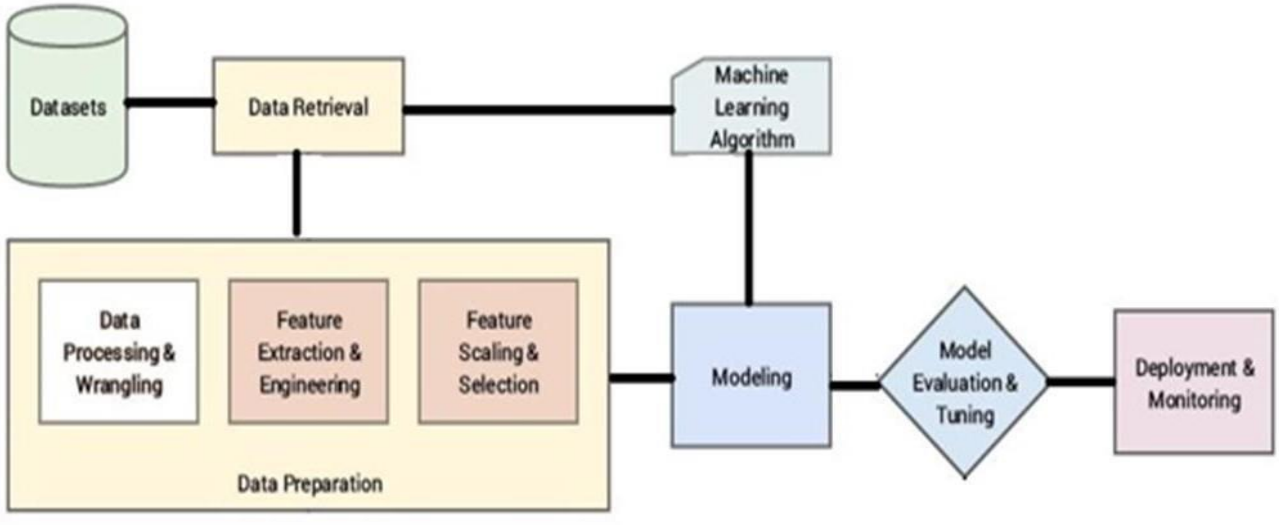
Python uses triple quotes at the beginning and end of the docstring:

Example

Docstrings are also comments:

```
"""This is a
multiline docstring."""
print("Hello, World!")
```

3.3 ARCHITECTURE


Datasets
Data Retrieval
Machine Learning Algorithm
Data Processing & Wrangling
Feature Extraction & Engineering
Feature Scaling & Selection
Data Preparation
Modeling
Model Evaluation & Tuning
Deployment & Monitoring

3.3.1 ARCHITETURE FIGURE

3.4 MODULES

- Data Collection
- Data Cleaning
- Data Selection
- Data Transformation
- Data Mining Stage

Data Collection

The data used for this work was collected from meteorologist‟s centre. The case data covered the period of 2012 to 2015. The following procedures were adopted at this stage of the research: Data Cleaning, Data Selection, Data Transformation and Data Mining.

Data Cleaning

In this stage, a consistent format for the data model was developed which is search missing data, finding duplicated data, and weeding out of bad data. Finally system cleaned data were transformed into a format suitable for data mining.

Data Selection

At this stage, data relevant to the analysis like decision tree was decided on and retrieved from the dataset. The Meteorological dataset had ten attributes in that were using two attributes for future prediction. Due to the nature of the Cloud Form data where all the values are the same and the high percentage of missing values in the sunshine data both were not used in the analysis.

Data Transformation

"This is also known as data consolidation". It is the stage in which the selected data is transformed into forms appropriate for data mining. The data file was saved in Commas

Separated Value (CVS) file format and the datasets were normalized to reduce the effect of scaling on the data.

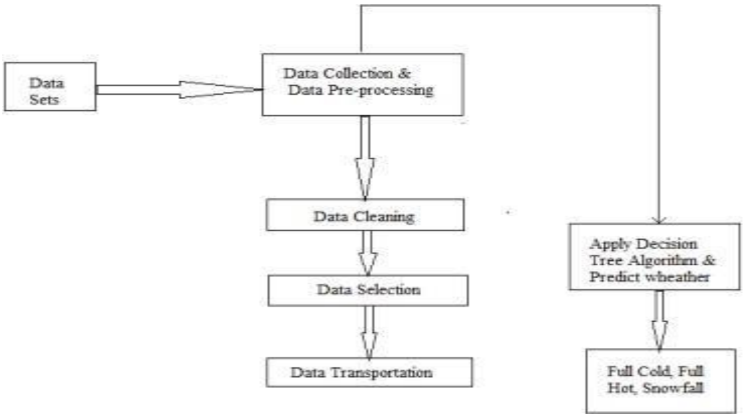
Data Mining Stage

The data mining stage was divided into three phases. At each phase all the algorithms were used to analyse the meteorological datasets. The testing method adopted for this research was percentage split that train on a percentage of the dataset, cross validate on it and test on the remaining percentage. There after interesting patterns representing knowledge were identified.

3.5 DATA FLOW DIAGRAM

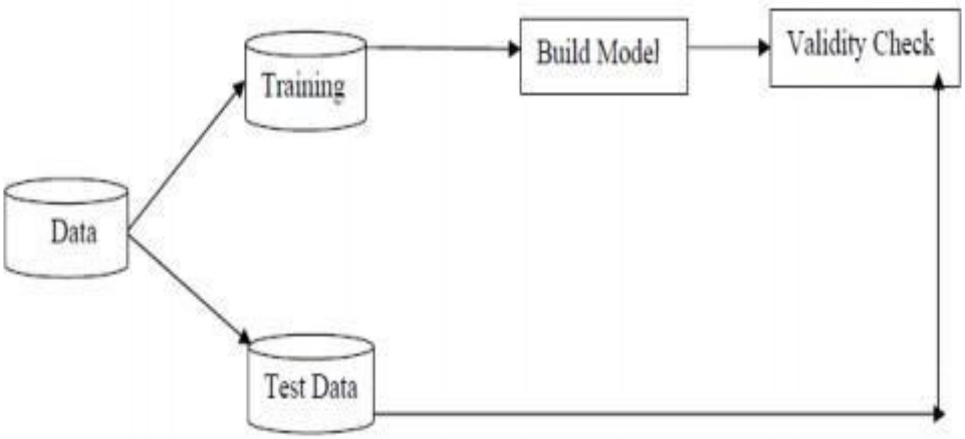
1. The DFD is also called as bubble chart. It is a simple graphical formalism that can be used to represent a system in terms of input data to the system, various processing carried out on this data, and the output data is generated by this system.
2. The data flow diagram (DFD) is one of the most important modeling tools. It is used to model the system components. These components are the system process, the data used by the process, an external entity that interacts with the system and the information flows in the system.
3. DFD shows how the information moves through the system and how it is modified by a series of transformations. It is a graphical technique that depicts information flow and the transformations that are applied as data moves from input to output.
4. DFD is also known as bubble chart. A DFD may be used to represent a system at any level of abstraction. DFD may be partitioned into levels that represent increasing information flow and functional detail.

LEVEL - 0


Data Sets
Data Collection & Data Pre-processing
Data Cleaning
Data Selection
Data Transportation
Apply Decision Tree Algorithm & Predict wheather
Full Cold, Full Hot, Snowfall

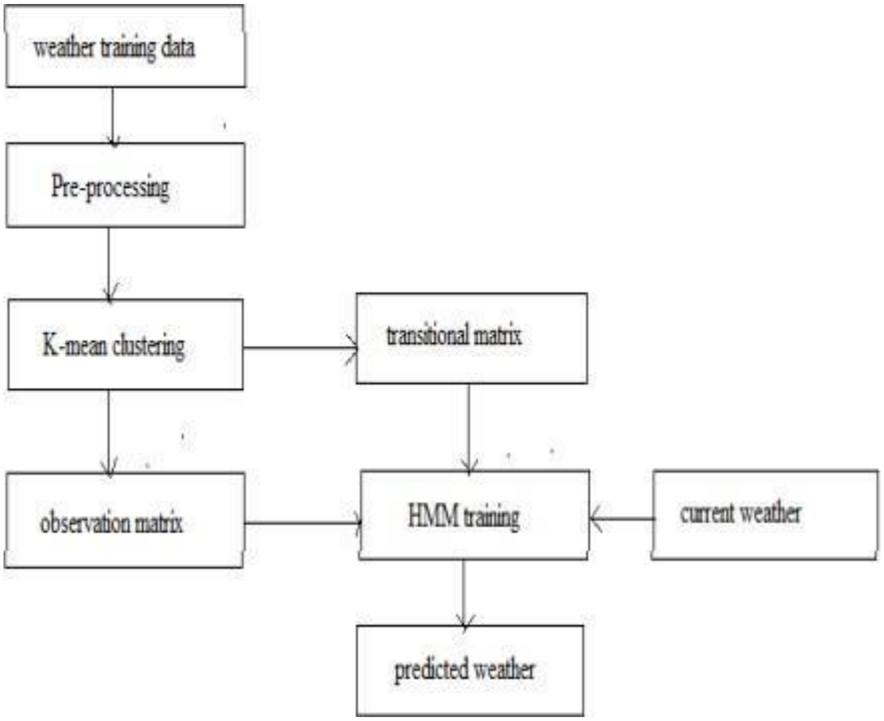
3.5.1 DATAFLOW DIAGRAM 1

LEVEL - 1



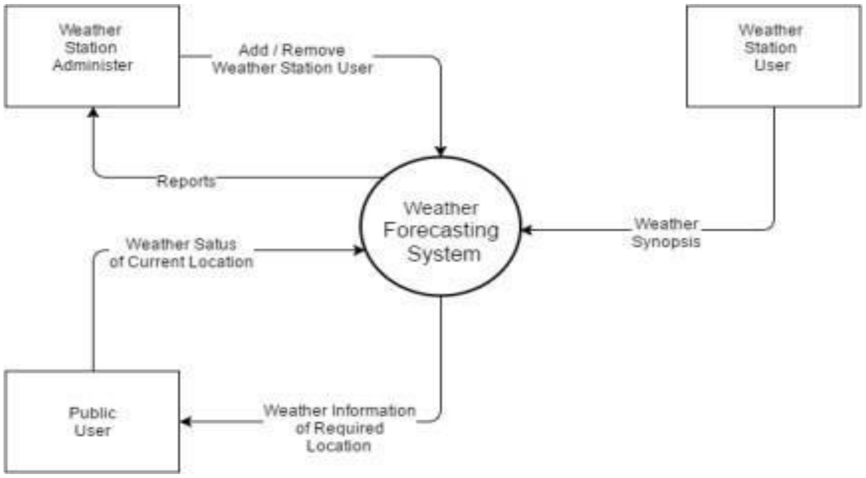
3.5.2 DATAFLOW DIAGRAM 2

LEVEL - 2



3.5.3 DATAFLOW DIAGRAM 3

LEVEL – 3


Weather Station Administer
Add / Remove Weather Station User
Weather Station User
Reports
Weather Forecasting System
Weather Synopsis
Weather Satus of Current Location
Public User
Weather Information of Required Location

3.5.4 DATAFLOW DIAGRAM 4

3.6 UML DIAGRAMS

UML stands for Unified Modeling Language. UML is a standardized general-purpose modeling language in the field of object-oriented software engineering. The standard is managed, and was created by, the Object Management Group.

The goal is for UML to become a common language for creating models of object oriented computer software. In its current form UML is comprised of two major components: a Meta-model and a notation. In the future, some form of method or process may also be added to; or associated with, UML.

The Unified Modeling Language is a standard language for specifying, Visualization, Constructing and documenting the artifacts of software system, as well as for business modeling and other non-software systems.

The UML represents a collection of best engineering practices that have proven successful in the modeling of large and complex systems.

The UML is a very important part of developing objects oriented software and the software development process. The UML uses mostly graphical notations to express the design of software projects.

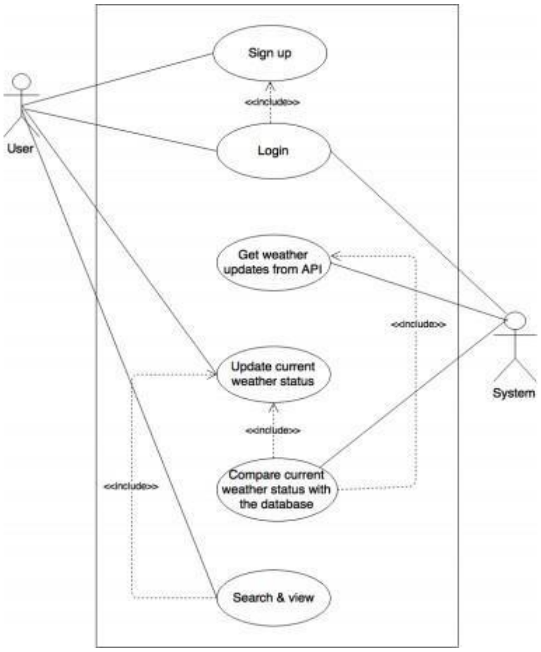
GOALS:

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

1. Provide users a ready-to-use, expressive visual modeling Language so that they can develop and exchange meaningful models.
2. Provide extendibility and specialization mechanisms to extend the core concepts.
3. Be independent of particular programming languages and development process.
4. Provide a formal basis for understanding the modeling language.
5. Encourage the growth of OO tools market.
6. Support higher level development concepts such as collaborations, frameworks, patterns and components.
7. Integrate best practices.

USE CASE DIAGRAM:

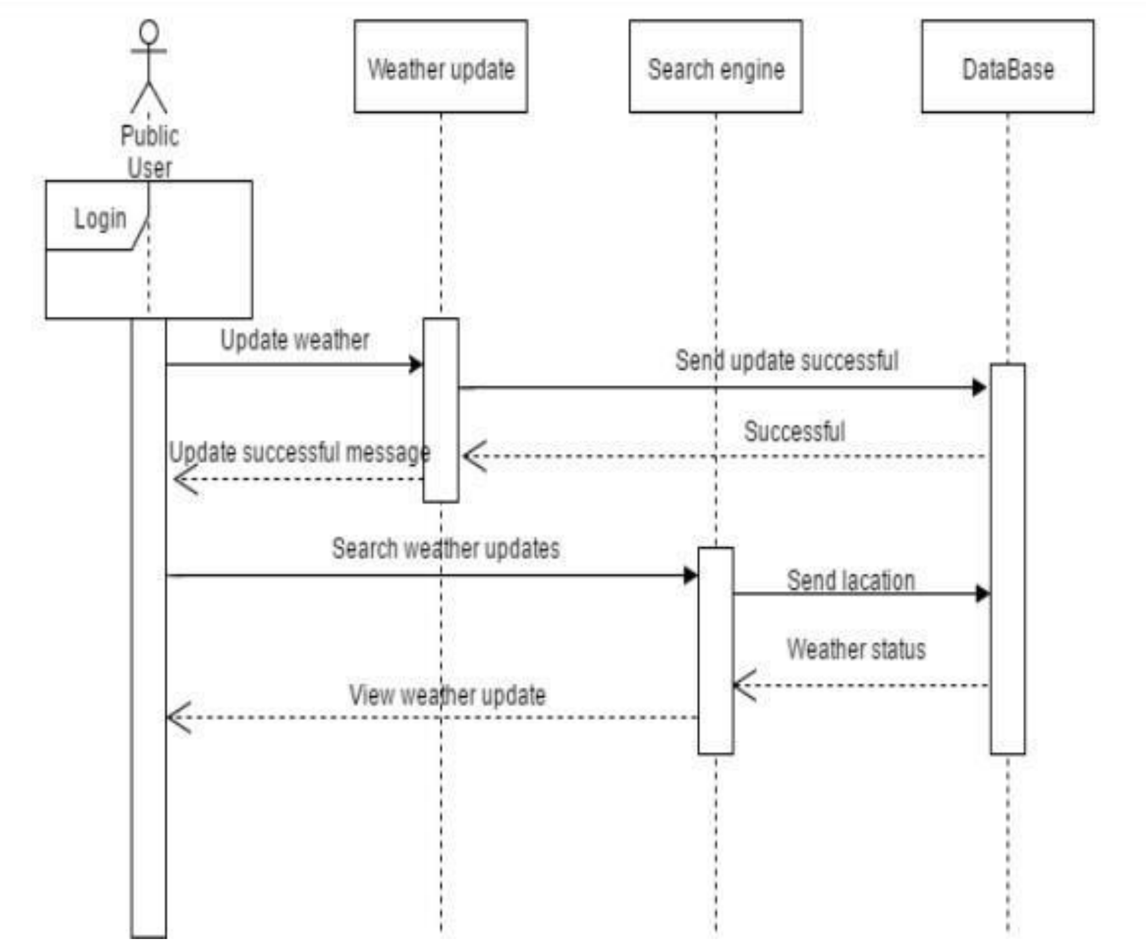
A use case diagram in the Unified Modeling Language (UML) is a type of behavioral diagram defined by and created from a Use-case analysis. Its purpose is to present a graphical overview of the functionality provided by a system in terms of actors, their goals (represented as use cases), and any dependencies 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.


Sign up
<<include>>
User
Login
Get weather updates from API
<<include>>
System
Update current weather status
<<include>>
<<include>>
Compare current weather status with the database
Search & view

3.6.1 USECASE DIAGRAM

SEQUENCE DIAGRAM:

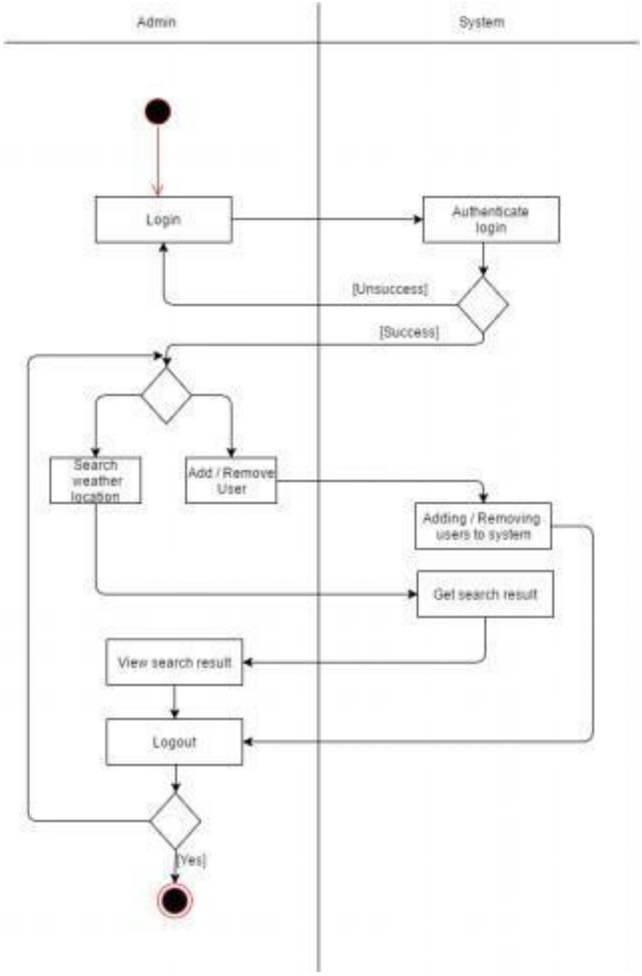
A sequence diagram in Unified Modeling Language (UML) is a kind of interaction diagram that shows how processes operate with one another and in what order. It is a construct of a Message Sequence Chart. Sequence diagrams are sometimes called event diagrams, event scenarios, and timing diagrams.



3.6.2 SEQUENCE DIAGRAM

ACTIVITY DIAGRAM:

Activity diagrams are graphical representations of workflows of stepwise activities and actions with support for choice, iteration and concurrency. In the Unified Modeling Language, activity diagrams can be used to describe the business and operational step-by-step workflows of components in a system. An activity diagram shows the overall flow of control.



3.6.3 ACTIVITY DIAGRAM

3.7 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.

The feasibility study investigates the problem and the information needs of the stakeholders. It seeks to determine the resources required to provide an information systems solution, the cost and benefits of such a solution, and the feasibility of such a solution.

The goal of the feasibility study is to consider alternative information systems solutions, evaluate their feasibility, and propose the alternative most suitable to the organization. The feasibility of a proposed solution is evaluated in terms of its components.

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 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.

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.

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.

3.8 SYSTEM DESIGN AND TESTING PLAN

INPUT DESIGN

The input design is the link between the information system and the user. It comprises the developing specification and procedures for data preparation and those steps are necessary to put transaction data in to a usable form for processing can be achieved by inspecting the computer to read data from a written or printed document or it can occur by having people keying the data directly into the system. The design of input focuses on controlling the amount of input required, controlling the errors, avoiding delay, avoiding extra steps and keeping the process simple. The input is designed in such a way so that it provides security and ease of use with retaining the privacy. Input Design considered the following things:

- What data should be given as input?
- How the data should be arranged or coded?
- The dialog to guide the operating personnel in providing input.
- Methods for preparing input validations and steps to follow when error occur.

OUTPUT DESIGN

A quality output is one, which meets the requirements of the end user and presents the information clearly. In any system results of processing are communicated to the users and to other system through outputs. In output design it is determined how the information is to be displaced for immediate need and also the hard copy output. It is the most important and direct source information to the user. Efficient and intelligent output design improves the system‟s relationship to help user decision-making.

The output form of an information system should accomplish one or more of the following objectives.

- Convey information about past activities, current status or projections of the
- Future.
- Signal important events, opportunities, problems, or warnings.
- Trigger an action.
- Confirm an action.

Test plan

Software testing is the process of evaluation a software item to detect differences between given input and expected output. Also to assess the feature of a software item. Testing assesses the quality of the product. Software testing is a process that should be done during the development process. In other words software testing is a verification and validation process.

Verification

Verification is the process to make sure the product satisfies the conditions imposed at the start of the development phase. In other words, to make sure the product behaves the way we want it to.

Validation

Validation is the process to make sure the product satisfies the specified requirements at the end of the development phase. In other words, to make sure the product is built as per customer requirements.

Basics of software testing

There are two basics of software testing: black box testing and white box testing.

Black box Testing

Black box testing is a testing technique that ignores the internal mechanism of the system and focuses on the output generated against any input and execution of the system. It is also called functional testing.

White box Testing

White box testing is a testing technique that takes into account the internal mechanism of a system. It is also called structural testing and glass box testing. Black box testing is often used for validation and white box testing is often used for verification.

Types of testing:

There are many types of testing like

- Unit Testing
- Integration Testing
- Functional Testing
- System Testing
- Stress Testing
- Performance Testing
- Usability Testing

- Acceptance Testing
- Regression Testing
- Beta Testing

Unit Testing

Unit testing is the testing of an individual unit or group of related units. It falls under the class of white box testing. It is often done by the programmer to test that the unit he/she has implemented is producing expected output against given input.

Integration Testing

Integration testing is testing in which a group of components are combined to produce output. Also, the interaction between software and hardware is tested in integration testing if software and hardware components have any relation. It may fall under both white box testing and black box testing.

Functional Testing

Functional testing is the testing to ensure that the specified functionality required in the system requirements works. It falls under the class of black box testing.

System Testing

System testing is the testing to ensure that by putting the software in different environments (e.g., Operating Systems) it still works. System testing is done with full system implementation and environment. It falls under the class of black box testing.

Stress Testing

Stress testing is the testing to evaluate how system behaves under unfavorable conditions. Testing is conducted at beyond limits of the specifications. It falls under the class of black box testing.

Performance Testing

Performance testing is the testing to assess the speed and effectiveness of the system and to make sure it is generating results within a specified time as in performance requirements. It falls under the class of black box testing.

Usability Testing

Usability testing is performed to the perspective of the client, to evaluate how the GUI is user-friendly? How easily can the client learn? After learning how to use, how proficiently can the client perform? How pleasing is it to use its design? This falls under the class of black box testing.

Acceptance Testing

Acceptance testing is often done by the customer to ensure that the delivered product meets the requirements and works as the customer expected. It falls under the class of black box testing.

Regression Testing

Regression testing is the testing after modification of a system, component, or a group of related units to ensure that the modification is working correctly and is not damaging or imposing other modules to produce unexpected results. It falls under the class of black box testing.

Technical Details:

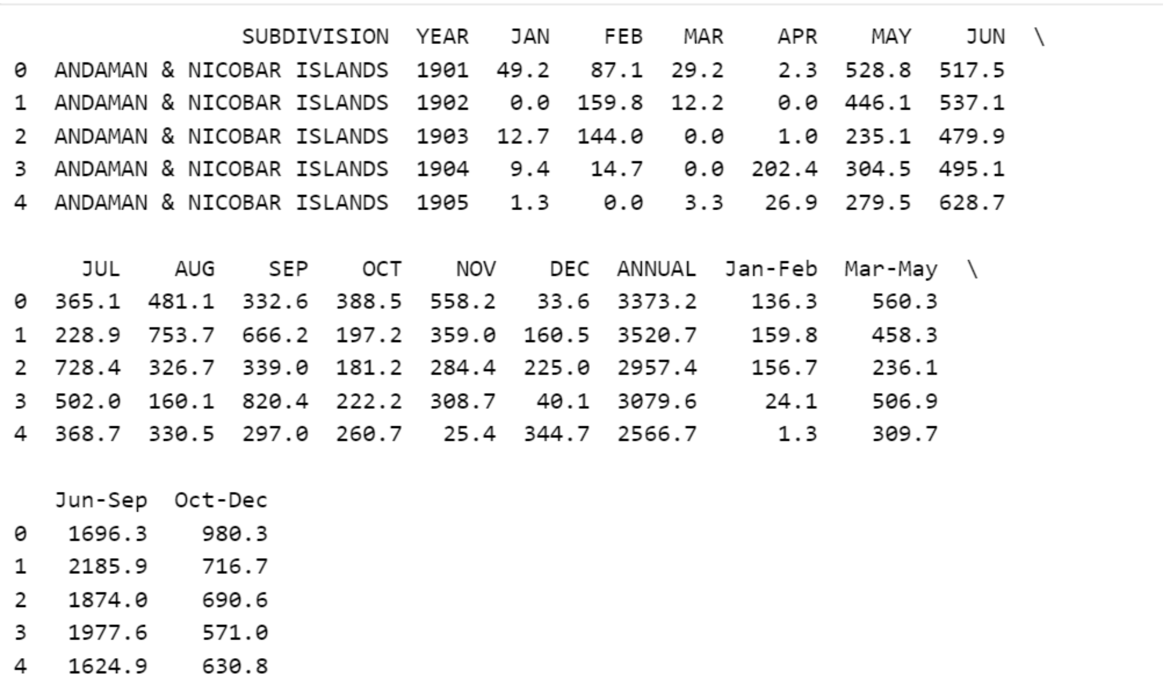
Artificial neural network (ANN) algorithms can accurately estimate precipitation by identifying hidden patterns in easily available historical weather data. This project contributes by providing a comprehensive study and critique of state-of-the-art data mining algorithms for predicting precipitation. To improve the accuracy of forecasting and climate monitoring, meteorological parameters must be determined in a timely manner and with very high accuracy and under controlled conditions. Precipitation estimation is important because of its impact on human livelihoods, water resources, and water consumption. Therefore, ANN can be used to accurately predict when it will rain.

Technical Description:

Artificial neural network (ANN) techniques can effectively predict precipitation by extracting hidden patterns from available features of historical weather data. This project contributes by providing a critical analysis and review of the latest data mining techniques used for precipitation forecasting. Meteorological parameters must be determined in a timely manner and with very high accuracy in a controlled environment to improve the accuracy of forecasting and climate monitoring. Precipitation estimation is therefore of great importance as it affects human livelihoods, water resources and water consumption. Therefore, ANN can be used to effectively predict precipitation.

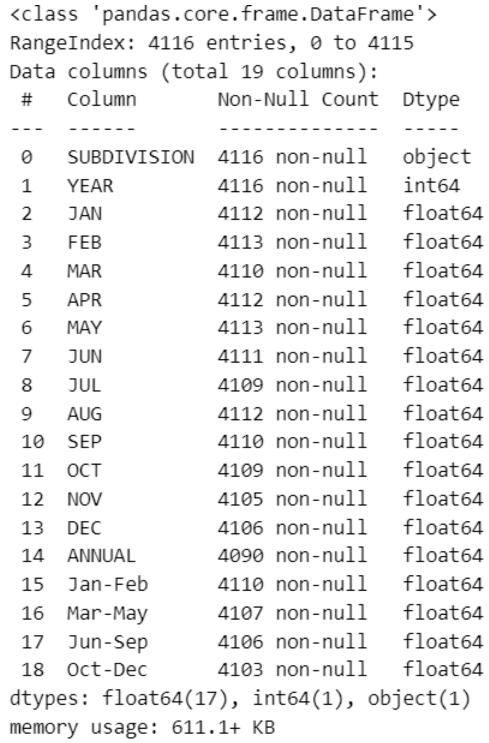
IMPLEMENTATION

```
#Importing Libraries
import numpy as np
import pandas as pd

import matplotlib.pyplot as plt
import seaborn as sns

from sklearn import metrics

#data collection
data = pd.read_csv("rainfall in india 1901-2015.csv")
data.head()
```



```
                 SUBDIVISION  YEAR   JAN    FEB   MAR    APR    MAY    JUN  \
0  ANDAMAN & NICOBAR ISLANDS  1901  49.2   87.1  29.2    2.3  528.8  517.5
1  ANDAMAN & NICOBAR ISLANDS  1902   0.0  159.8  12.2    0.0  446.1  537.1
2  ANDAMAN & NICOBAR ISLANDS  1903  12.7  144.0   0.0    1.0  235.1  479.9
3  ANDAMAN & NICOBAR ISLANDS  1904   9.4   14.7   0.0  202.4  304.5  495.1
4  ANDAMAN & NICOBAR ISLANDS  1905   1.3    0.0   3.3   26.9  279.5  628.7

     JUL    AUG    SEP    OCT    NOV    DEC  ANNUAL  Jan-Feb  Mar-May  \
0  365.1  481.1  332.6  388.5  558.2   33.6  3373.2    136.3    560.3
1  228.9  753.7  666.2  197.2  359.0  160.5  3520.7    159.8    458.3
2  728.4  326.7  339.0  181.2  284.4  225.0  2957.4    156.7    236.1
3  502.0  160.1  820.4  222.2  308.7   40.1  3079.6     24.1    506.9
4  368.7  330.5  297.0  260.7   25.4  344.7  2566.7      1.3    309.7

   Jun-Sep  Oct-Dec
0   1696.3    980.3
1   2185.9    716.7
2   1874.0    690.6
3   1977.6    571.0
4   1624.9    630.8
```

data.info()



```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 4116 entries, 0 to 4115
Data columns (total 19 columns):
 #   Column       Non-Null Count  Dtype  
---  ------       --------------  -----  
 0   SUBDIVISION  4116 non-null   object 
 1   YEAR         4116 non-null   int64  
 2   JAN          4112 non-null   float64
 3   FEB          4113 non-null   float64
 4   MAR          4110 non-null   float64
 5   APR          4112 non-null   float64
 6   MAY          4113 non-null   float64
 7   JUN          4111 non-null   float64
 8   JUL          4109 non-null   float64
 9   AUG          4112 non-null   float64
 10  SEP          4110 non-null   float64
 11  OCT          4109 non-null   float64
 12  NOV          4105 non-null   float64
 13  DEC          4106 non-null   float64
 14  ANNUAL       4090 non-null   float64
 15  Jan-Feb      4110 non-null   float64
 16  Mar-May      4107 non-null   float64
 17  Jun-Sep      4106 non-null   float64
 18  Oct-Dec      4103 non-null   float64
dtypes: float64(17), int64(1), object(1)
memory usage: 611.1+ KB
```

Data pre-processing:

```
# data pre-processing steps
- data.isnull().sum()
- data.duplicated().sum()
- data['SUBDIVISION'].value_counts()
- data.mean()
- data = data.fillna(data.mean())
- data.head(3)
- data.YEAR.unique()
- data.describe()
- data.shape
```

Data visualization:

```
# data visualization
data[["SUBDIVISION","ANNUAL"]].groupby("SUBDIVISION").sum().sort_values(by=
'ANNUAL',ascending=False).plot(kind='barh',stacked=True,
figsize=(15,10))

# displaying each state annual rainfall in mm
plt.xlabel("Rainfall in MM",size=12)
plt.ylabel("Sub-Division",size=12)
plt.title("Annual Rainfall v/s SubDivisions")
plt.grid(axis="x",linestyle="-.")
plt.show()
```

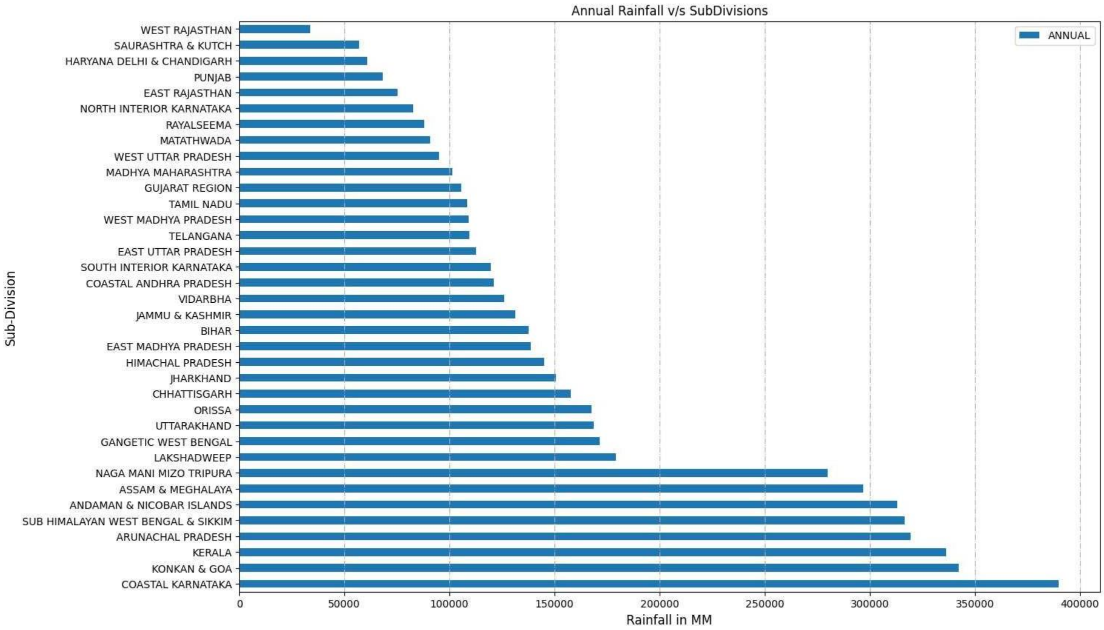


Fig 4.1

```
# Rainfall over years
plt.figure(figsize=(15,8))
data.groupby("YEAR").sum()['ANNUAL'].plot(kind="line",color="r",marker=".")
plt.xlabel("YEARS",size=12)
plt.ylabel("RAINFALL IN MM",size=12)
plt.grid(axis="both",linestyle="-.")
plt.title("Rainfall over Years")
plt.show()
```

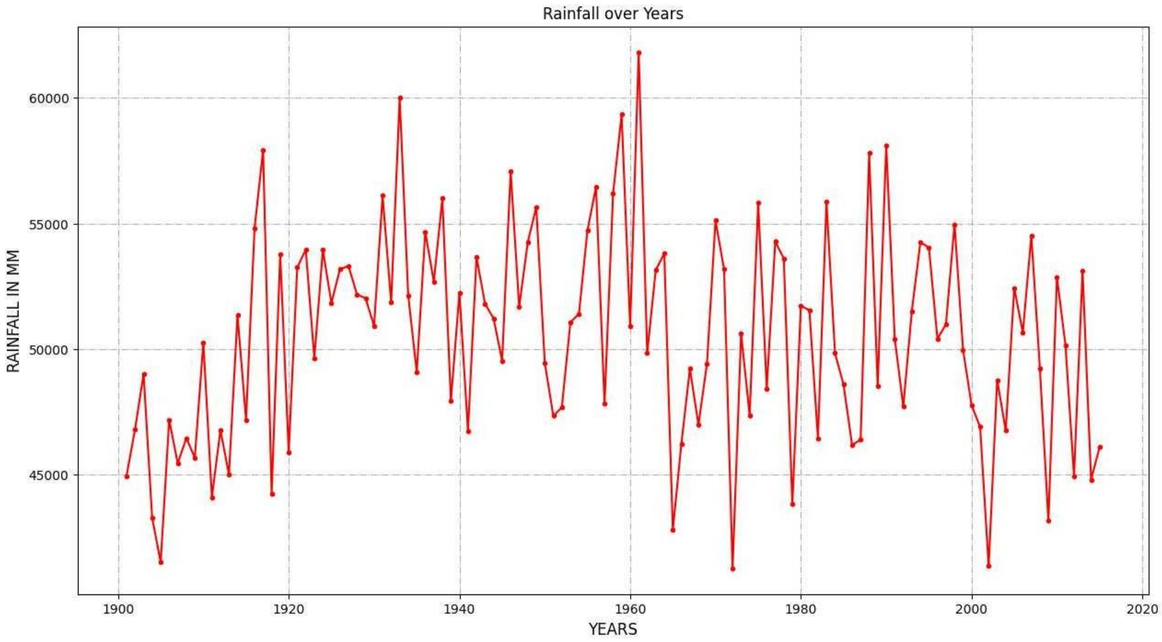


Fig 4.2

```
# Rainfall for monthly over years
data[['YEAR', 'JAN', 'FEB', 'MAR', 'APR', 'MAY', 'JUN', 'JUL','AUG', 'SEP',
      'OCT', 'NOV',
'DEC']].groupby("YEAR").sum().plot(kind="line",figsize=(18,8))
plt.xlabel("Year",size=13)
plt.ylabel("Rainfall  in  MM",size=13)
plt.title("Year v/s Rainfall in each month",size=20)
plt.show()
```

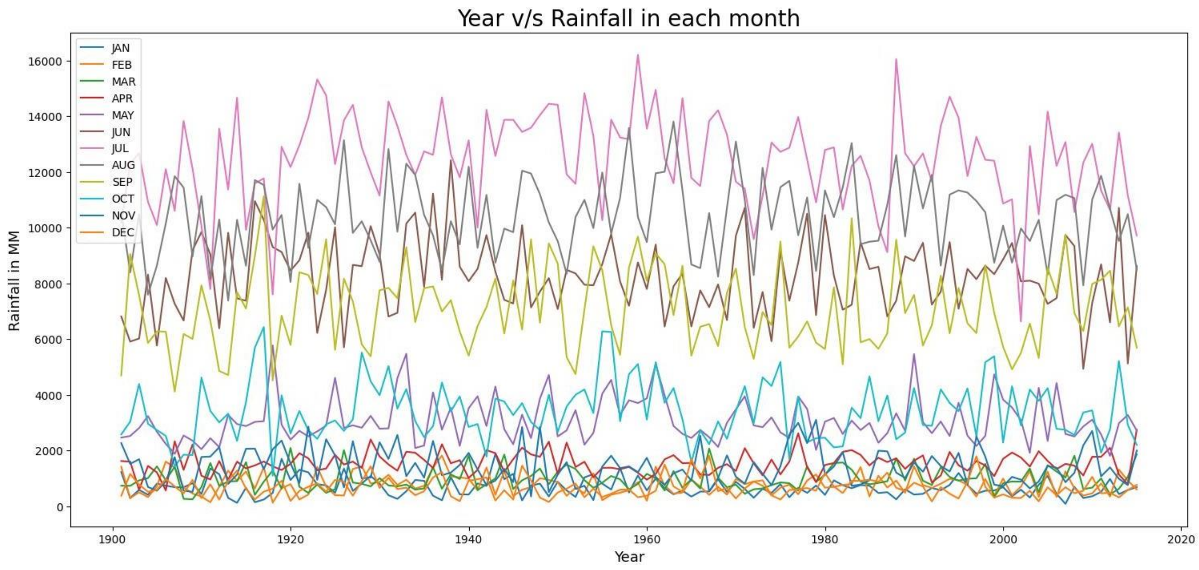


Fig 4.3

```
# grouping months for better analysis
data[['YEAR','Jan-Feb', 'Mar-May',
       'Jun-Sep', 'Oct-Dec']].groupby("YEAR").sum().plot(figsize=(10,7))
plt.xlabel("Year",size=13)
plt.ylabel("Rainfall in MM",size=13)
plt.show()
```

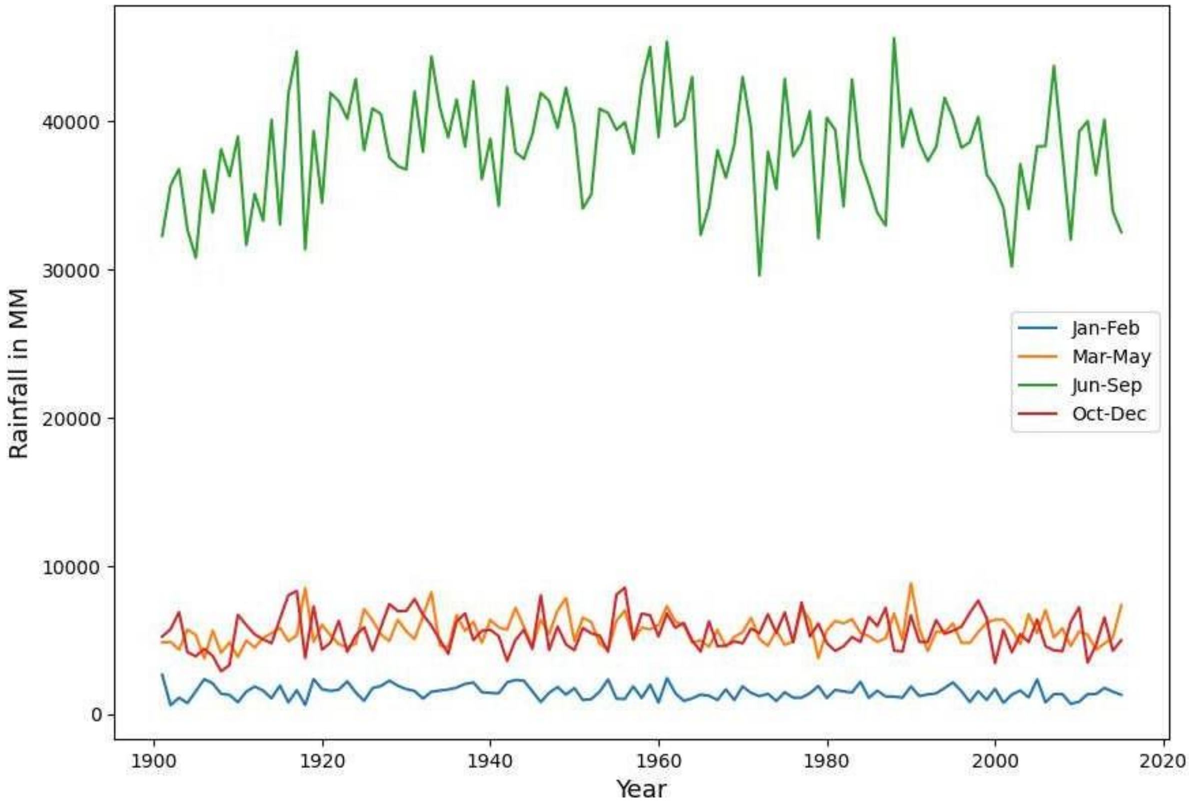


Fig 4.4

```
# grouping months for better analysis
data[['SUBDIVISION', 'Jan-Feb', 'Mar-May','Jun-Sep', 'Oct-Dec']].
groupby("SUBDIVISION").sum().plot(kind="barh",stacked=True,figsize=(16,8))
plt.xlabel("Rainfall in MM",size=12)
plt.ylabel("Sub-Division",size=12)
plt.grid(axis="x",linestyle="-.")
plt.show()
```

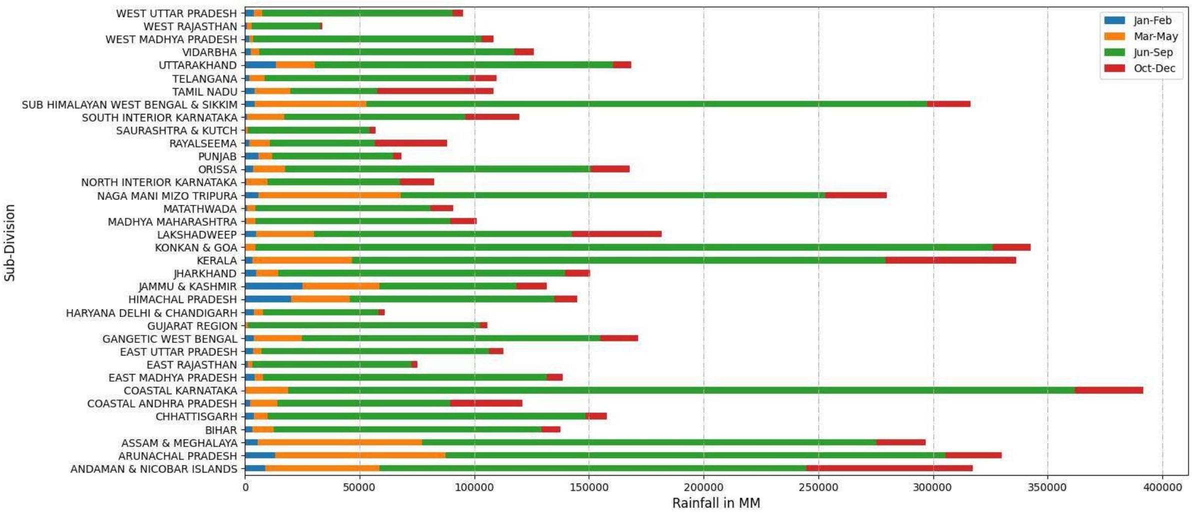


Fig 4.5

```
# ANALYSING States(TamilNadu and Rajasthan) by creating subgroups
TN = data.loc[((data['SUBDIVISION'] == 'TAMIL NADU'))]
TN.head()
```

```
      SUBDIVISION  YEAR   JAN   FEB   MAR   APR    MAY   JUN   JUL    AUG  \
3427   TAMIL NADU  1901  24.5  39.1  21.7  36.0   74.0  41.8  49.3   67.9
3428   TAMIL NADU  1902  67.2   9.8  25.1  21.9   84.7  39.3  55.1  113.8
3429   TAMIL NADU  1903  19.3   7.8   1.7  18.2  128.5  58.5  72.6  115.0
3430   TAMIL NADU  1904  35.2   0.1   0.7  19.5  121.9  34.9  89.0   40.4
3431   TAMIL NADU  1905   6.5   7.5  17.2  64.8   83.7  49.8  39.0  101.8

        SEP    OCT    NOV    DEC  ANNUAL  Jan-Feb  Mar-May  Jun-Sep  Oct-Dec
3427  191.1  122.3  212.3   80.4   960.3     63.6    131.6    350.1    415.0
3428   98.6  282.2  174.9  165.8  1138.2     77.0    131.7    306.7    622.9
3429  210.4  128.1  200.5  203.2  1163.9     27.1    148.4    456.5    531.9
3430   85.7  163.2   23.6   49.1   663.1     35.3    142.1    249.9    235.8
3431   73.5  250.4  123.7    3.2   821.1     14.0    165.7    264.1    377.2
```

```
TN[['JAN', 'FEB', 'MAR', 'APR','MAY', 'JUN','JUL','AUG', 'SEP',
'OCT','NOV','DEC']].mean().plot(kind="bar",width=0.5,linewidth=2)
plt.title("Tamil Nadu Rainfall v/s Months",size=20)
plt.xlabel("Months",size=14)
plt.ylabel("Rainfall in MM",size=14)
plt.grid(axis="both",linestyle="-.")
plt.show()
```

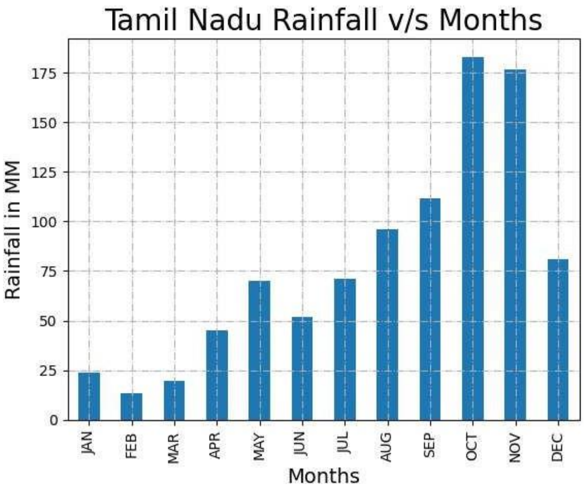


Fig 4.6

```
TN.groupby("YEAR").sum()['ANNUAL'].plot(ylim=(50,1500),color='r',marker='o',li
nestyle='-',linewidth=2,figsize=(12,8));
plt.xlabel('Year',size=14)
plt.ylabel('Rainfall in MM',size=14)
plt.title('Tamil Nadu Annual Rainfall from Year 1901 to 2015',size=20)
plt.grid()
plt.show()
```

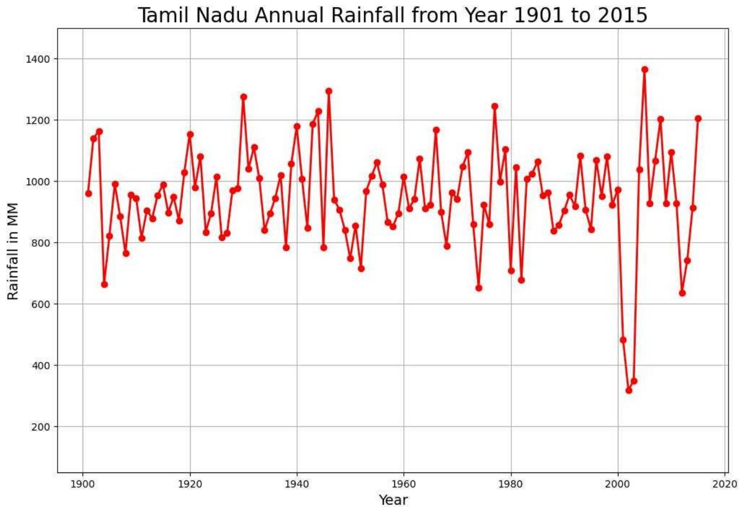


Fig 4.7

```
Rajasthan = data.loc[((data['SUBDIVISION'] == 'WEST RAJASTHAN') |
(data['SUBDIVISION'] == 'EAST RAJASTHAN'))]
Rajasthan.head()
```

```
         SUBDIVISION  YEAR  JAN  FEB   MAR  APR   MAY   JUN    JUL   AUG  \
1817  WEST RAJASTHAN  1901  6.7  0.0   1.1  0.0   6.1   3.0   79.0  59.2
1818  WEST RAJASTHAN  1902  0.0  0.0   0.0  0.5   4.0  49.1   27.0  71.3
1819  WEST RAJASTHAN  1903  1.7  1.3   5.5  0.0   4.2   2.7  154.8  87.1
1820  WEST RAJASTHAN  1904  3.8  2.9  16.3  0.7  11.4  14.6   39.8  45.6
1821  WEST RAJASTHAN  1905  6.3  4.8   0.7  1.3   0.3   4.9   30.1   0.6

       SEP  OCT  NOV  DEC  ANNUAL  Jan-Feb  Mar-May  Jun-Sep  Oct-Dec
1817   1.0  2.1  0.0  0.6   158.9      6.7      7.2    142.2      2.7
1818  41.8  1.8  0.0  0.0   195.6      0.0      4.5    189.2      1.8
1819  49.3  0.1  0.0  0.5   307.0      3.0      9.7    293.8      0.5
1820  21.4  1.4  2.9  7.1   167.9      6.6     28.5    121.4     11.4
1821  64.5  0.0  0.0  0.9   114.4     11.0      2.4    100.1      0.9
```

```
plt.figure(figsize=(10,6))
Rajasthan[['JAN', 'FEB', 'MAR', 'APR','MAY', 'JUN','JUL','AUG', 'SEP',
'OCT','NOV','DEC']].mean().plot(kind="bar",width=0.5,linewidth=2)
plt.title("Rajasthan Rainfall v/s Months",size=20)
plt.xlabel("Months",size=14)
plt.ylabel("Rainfall in MM",size=14)
plt.grid(axis="both",linestyle="-.")
plt.show()
plt.figure(figsize=(10,6))
```

```
Rajasthan[['JAN', 'FEB', 'MAR', 'APR','MAY', 'JUN','JUL','AUG', 'SEP',
'OCT','NOV','DEC']].mean().plot(kind="bar",width=0.5,linewidth=2)
plt.title("Rajasthan Rainfall v/s Months",size=20)
plt.xlabel("Months",size=14)
plt.ylabel("Rainfall in MM",size=14)
plt.grid(axis="both",linestyle="-.")
plt.show()
```

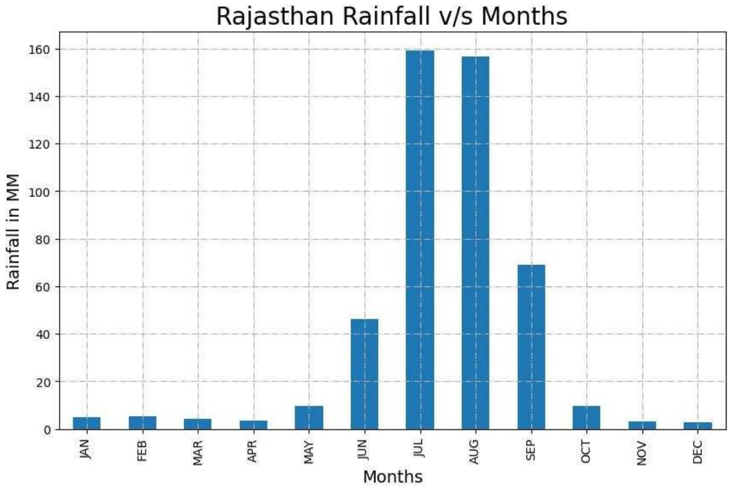


Fig 4.8

```
Rajasthan.groupby("YEAR").mean()['ANNUAL'].plot(ylim=(50,1500),color='r',marke
r='o',linestyle='-',linewidth=2,figsize=(12,8));
plt.xlabel('Year',size=14)
plt.ylabel('Rainfall in MM',size=14)
plt.title('Rajasthan Annual Rainfall from Year 1901 to 2015',size=20)
plt.grid()
plt.show()
plt.figure(figsize=(15,6))
```

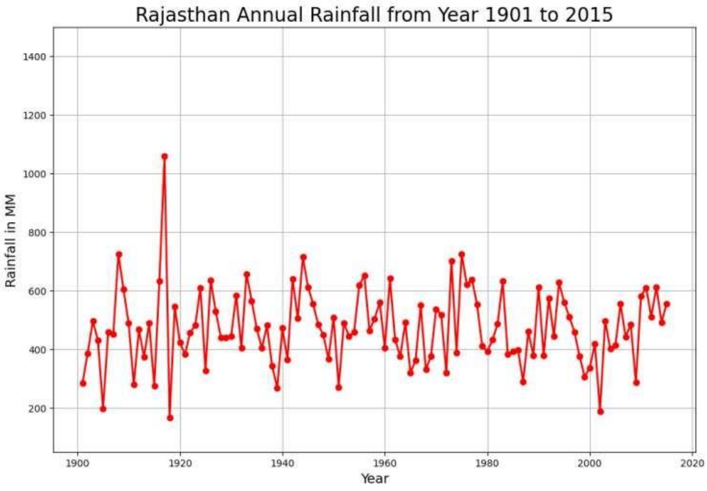


Fig 4.9

```
plt.figure(figsize=(15,6))
sns.heatmap(data[['JAN','FEB','MAR','APR','MAY','JUN','JUL','AUG','SEP','OCT',
'NOV','DEC','ANNUAL']].corr(),annot=True)
plt.show()
```

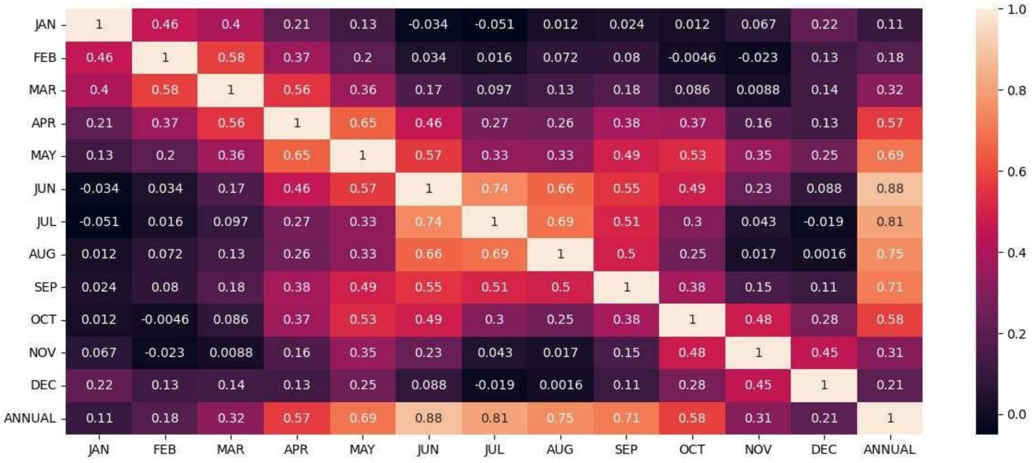


Fig 4.10

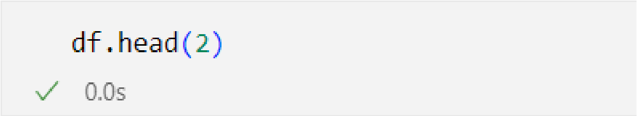
```
#getting unique data and grouping for tamil nadu state
data["SUBDIVISION"].nunique() #output:- 36


group =
data.groupby('SUBDIVISION')['YEAR','JAN','FEB','MAR','APR','MAY','JUN','JUL','AUG','SEP
','OCT','NOV','DEC']
data=group.get_group(('TAMIL NADU'))
data.head()

#extracting useful data
df=data.melt(['YEAR']).reset_index()
df.head()

df= df[['YEAR','variable','value']].reset_index().sort_values(by=['YEAR','index'])
df.head()


df.YEAR.unique()


df.columns=['Index','Year','Month','Avg_Rainfall']
df.head()
```

```
      Index  Year  Month  Avg_Rainfall
 0        0  1901    JAN          24.5
 115    115  1901    FEB          39.1
 230    230  1901    MAR          21.7
 345    345  1901    APR          36.0
 460    460  1901    MAY          74.0
```

```
Month_map={'JAN':1,'FEB':2,'MAR'
:3,'APR':4,'MAY':5,'JUN':6,'JUL':7,'AUG':8,'SEP':9,
   'OCT':10,'NOV':11,'DEC':12}
df['Month']=df['Month'].map(Month_map)
df.head()
```

```
       Index  Year  Month  Avg_Rainfall
0          0  1901      1          24.5
115      115  1901      2          39.1
230      230  1901      3          21.7
345      345  1901      4          36.0
460      460  1901      5          74.0
```

```
df.drop(columns="Index",inplace=True)
```



```
    df.head(2)
 ✓   0.0s
```

```
     Year  Month  Avg_Rainfall
0    1901      1          24.5
115  1901      2          39.1
```

```
df.groupby("Year").sum().plot()
plt.show()
```

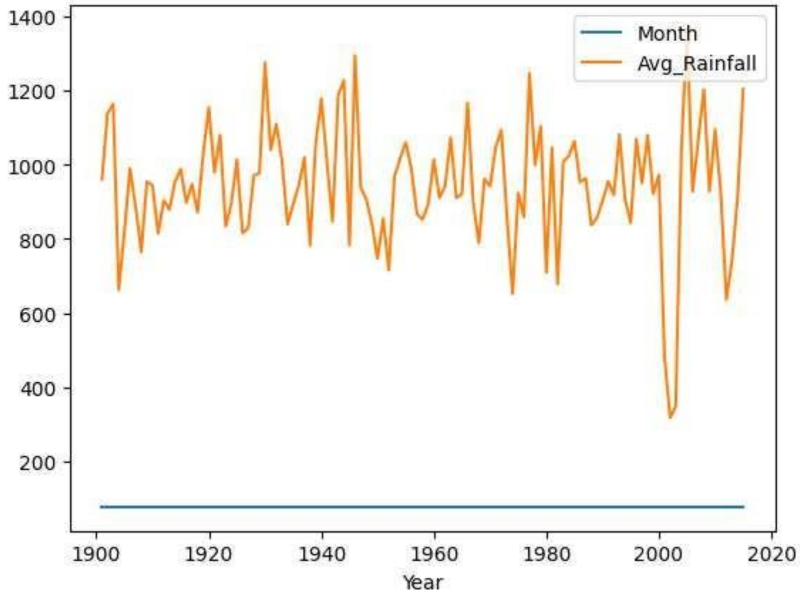
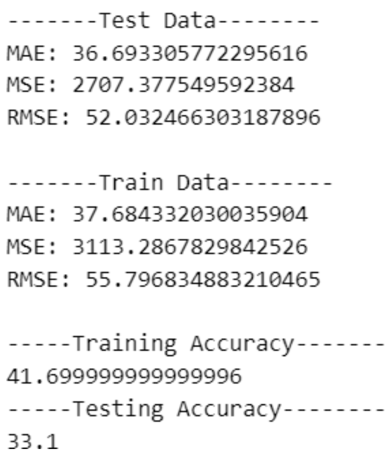


Fig 4.11

Linear Regression Model:

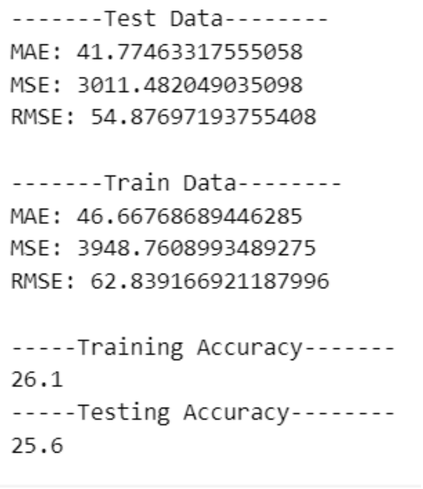
```
X=np.asanyarray(df[['Year','Month']]).astype('int')
y=np.asanyarray(df['Avg_Rainfall']).astype('int')
print(X.shape)
print(y.shape)


from sklearn.model_selection import train_test_split
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.3, random_state=10)


#Linear Regression Model
from sklearn.linear_model import LinearRegression
LR = LinearRegression()
LR.fit(X_train,y_train)

# predicting
y_train_predict=LR.predict(X_train)
y_test_predict=LR.predict(X_test)


print("-------Test Data-------------")
print('MAE:', metrics.mean_absolute_error(y_test, y_test_predict))
print('MSE:', metrics.mean_squared_error(y_test, y_test_predict))
print('RMSE:', np.sqrt(metrics.mean_squared_error(y_test, y_test_predict)))

print("\n-------Train Data------------ ")
print('MAE:', metrics.mean_absolute_error(y_train,y_train_predict))
print('MSE:', metrics.mean_squared_error(y_train, y_train_predict))
print('RMSE:', np.sqrt(metrics.mean_squared_error(y_train, y_train_predict)))

print("\n-----Training Accuracy----------- ")
print(round(LR.score(X_train,y_train),3)*100)
print("-----Testing Accuracy-------------")
print(round(LR.score(X_test,y_test),3)*100)
```



```
-------Test Data--------
MAE: 36.693305772295616
MSE: 2707.377549592384
RMSE: 52.032466303187896

-------Train Data--------
MAE: 37.684332030035904
MSE: 3113.2867829842526
RMSE: 55.796834883210465

-----Training Accuracy-------
41.699999999999996
-----Testing Accuracy--------
33.1
```

Lasso Model:

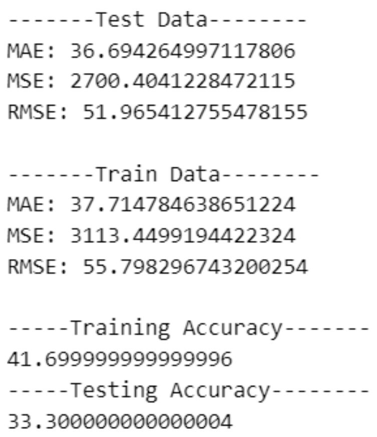
```
# Lasso Model
from sklearn.linear_model import Lasso
from sklearn.model_selection import GridSearchCV

# create a lasso object
lasso = Lasso(max_iter=100000)

# check for best alpha value using GridSearch
parameter={'alpha':[1e-15,1e-10,1e-8,1e-3,1e-2,1,5,1e1,1e2,1e3,1e4,1e5,1e6,1e7]}
lasso_regressor=GridSearchCV(lasso,parameter,scoring='neg_mean_squared_error',cv=5)

lasso_regressor.fit(X_train,y_train)
print("Best Parameter for Lasso:",lasso_regressor.best_estimator_)
lasso=Lasso(alpha=100.0,max_iter=100000)
# fit into the object
lasso.fit(X_train,y_train)

# predicting
y_train_predict=lasso.predict(X_train)
y_test_predict=lasso.predict(X_test)

#lasso regression
from sklearn import metrics
print("-------Test Data-------------")
print('MAE:', metrics.mean_absolute_error(y_test, y_test_predict))
print('MSE:', metrics.mean_squared_error(y_test, y_test_predict))
print('RMSE:', np.sqrt(metrics.mean_squared_error(y_test, y_test_predict)))

print("\n-------Train Data-------------")
print('MAE:', metrics.mean_absolute_error(y_train,y_train_predict))
print('MSE:', metrics.mean_squared_error(y_train, y_train_predict))
print('RMSE:', np.sqrt(metrics.mean_squared_error(y_train, y_train_predict)))

print("\n-----Training Accuracy ---------- ")
print(round(lasso.score(X_train,y_train),3)*100)
print("-----Testing Accuracy ------------")
print(round(lasso.score(X_test,y_test),3)*100)
```



```
-------Test Data--------
MAE: 41.77463317555058
MSE: 3011.482049035098
RMSE: 54.87697193755408

-------Train Data--------
MAE: 46.66768689446285
MSE: 3948.7608993489275
RMSE: 62.839166921187996

-----Training Accuracy-------
26.1
-----Testing Accuracy--------
25.6
```

Ridge Model:

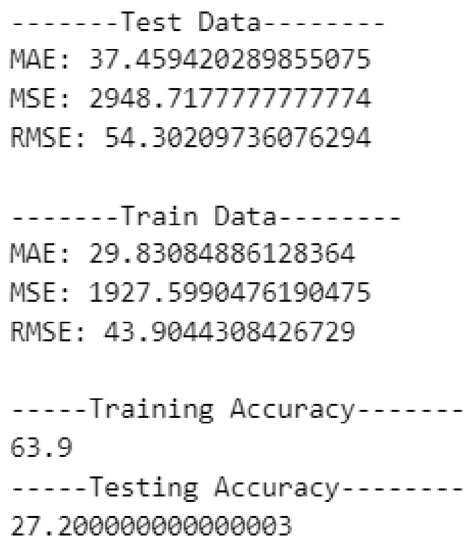
```
# Ridge Model
from sklearn.linear_model import Ridge
from sklearn.model_selection import GridSearchCV

ridge=Ridge()
parameters={'alpha':[1e-15,1e-10,1e-8,1e-3,1e-2,1,5,10,20,30,35,40,45,50,55,100]}
ridge_regressor=GridSearchCV(ridge,parameters,scoring='neg_mean_squared_error',cv=5)
ridge_regressor.fit(X_train,y_train)

print(ridge_regressor.best_params_)
print(ridge_regressor.best_score_)

print("Best Parameter for Ridge:",ridge_regressor.best_estimator_)
ridge=Ridge(alpha=100.0)

# fit into the object
ridge.fit(X_train,y_train)
# predicting
y_train_predict=ridge.predict(X_train)
y_test_predict=ridge.predict(X_test)
# Ridge Model
from sklearn import metrics
print("-------Test Data------------")
print('MAE:', metrics.mean_absolute_error(y_test, y_test_predict))
print('MSE:', metrics.mean_squared_error(y_test, y_test_predict))
print('RMSE:', np.sqrt(metrics.mean_squared_error(y_test, y_test_predict)))

print("\n-------Train Data-----------")
print('MAE:', metrics.mean_absolute_error(y_train,y_train_predict))
print('MSE:', metrics.mean_squared_error(y_train, y_train_predict))
print('RMSE:', np.sqrt(metrics.mean_squared_error(y_train, y_train_predict)))

print("\n-----Training Accuracy ---------")
print(round(ridge.score(X_train,y_train),3)*100)
print("-----Testing Accuracy ----------")
print(round(ridge.score(X_test,y_test),3)*100)
```



```
-------Test Data--------
MAE: 36.694264997117806
MSE: 2700.4041228472115
RMSE: 51.965412755478155

-------Train Data--------
MAE: 37.714784638651224
MSE: 3113.4499194422324
RMSE: 55.798296743200254

-----Training Accuracy-------
41.699999999999996
-----Testing Accuracy--------
33.300000000000004
```

KNN Model:

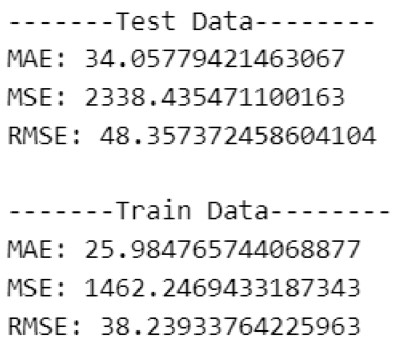
```
from sklearn.neighbors import KNeighborsRegressor
from sklearn import metrics

# create KNN regressor object
knn_regr = KNeighborsRegressor(n_neighbors=5)

# fit the model with the training data
knn_regr.fit(X_train, y_train)

# predict on the test data
y_test_predict = knn_regr.predict(X_test)
y_train_predict = knn_regr.predict(X_train)

# print the evaluation metrics
print("-------Test Data------------")
print('MAE:', metrics.mean_absolute_error(y_test, y_test_predict))
print('MSE:', metrics.mean_squared_error(y_test, y_test_predict))
print('RMSE:', np.sqrt(metrics.mean_squared_error(y_test, y_test_predict)))

print("\n-------Train Data------------")
print('MAE:', metrics.mean_absolute_error(y_train,y_train_predict))
print('MSE:', metrics.mean_squared_error(y_train, y_train_predict))
print('RMSE:', np.sqrt(metrics.mean_squared_error(y_train, y_train_predict)))

print("\n-----Training Accuracy---------- ")
print(round(knn_regr.score(X_train,y_train),3)*100)
print("-----Testing Accuracy ------------")
print(round(knn_regr.score(X_test,y_test),3)*100)
```



```
-------Test Data--------
MAE: 37.459420289855075
MSE: 2948.7177777777774
RMSE: 54.30209736076294

-------Train Data--------
MAE: 29.83084886128364
MSE: 1927.5990476190475
RMSE: 43.9044308426729

-----Training Accuracy-------
63.9
-----Testing Accuracy--------
27.200000000000003
```

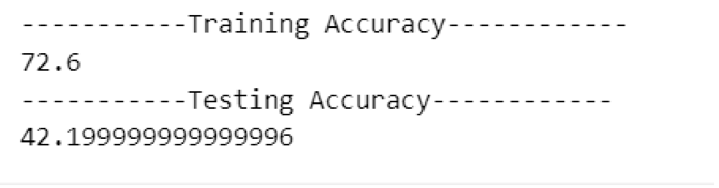
Random Forest Model:

```
# Random Forest Model
from sklearn.ensemble import RandomForestRegressor
random_forest_model = RandomForestRegressor(max_depth=100,
max_features='sqrt', min_samples_leaf=4, min_samples_split=10,
n_estimators=800)
random_forest_model.fit(X_train, y_train)

y_train_predict=random_forest_model.predict(X_train)
y_test_predict=random_forest_model.predict(X_test)

# Random Forest Model
print("-------Test Data------------")
print('MAE:', metrics.mean_absolute_error(y_test, y_test_predict))
print('MSE:', metrics.mean_squared_error(y_test, y_test_predict))
print('RMSE:', np.sqrt(metrics.mean_squared_error(y_test, y_test_predict)))

print("\n-------Train Data------------")
print('MAE:', metrics.mean_absolute_error(y_train,y_train_predict))
print('MSE:', metrics.mean_squared_error(y_train, y_train_predict))
print('RMSE:', np.sqrt(metrics.mean_squared_error(y_train, y_train_predict)))
```



```
-------Test Data--------
MAE: 34.05779421463067
MSE: 2338.435471100163
RMSE: 48.357372458604104

-------Train Data--------
MAE: 25.984765744068877
MSE: 1462.2469433187343
RMSE: 38.23933764225963
```

```
print("-----------Training Accuracy-------------------")
print(round(random_forest_model.score(X_train,y_train),3)*100)
print("-----------Testing Accuracy-------------------")
print(round(random_forest_model.score(X_test,y_test),3)*100)
```



```
-----------Training Accuracy------------
72.6
-----------Testing Accuracy------------
42.199999999999996
```

CHAPTER 4

RESULTS AND DISCUSSION

Decision Tree, Random Forest, Simple Linear Regression and multinomial regression are the classification method used for time series predict in this research work. Two group are separated from the data set for training and for testing the algorithms of classification. To execute the classification algorithms, the tool used is flask webapp data examination. For classification procedure no more than a separation of data is particular from the loaded data. To choose a subset from innovative data, “Select attribute” are utilised by the operative. The preferred subset is then subjected to “X-Validation” operator. It develop the classification representation which is validated by the test data.

CHAPTER 5
CONCLUSION

Weather forecasting is a meteorological work that easy to modify researcher work by applying the numerical weather prediction method. Weather forecasted by using various data mining techniques especially classification clustering and neural network, decision tree. The key aim for improving the classification and prediction performance for the traditional; weather prediction model is designed and developed in this work. But some limitation of the model is also observed, thus in near future need to be review before use of the proposed technique. And also soil there are some issues and challenges in which better implement of data mining technique should be implemented in field of weather forecasting.

REFERENCES:

[1] Xiong, Lihua, and Kieran M. OConnor. “An empirical method to improve the prediction limits of the GLUE methodology in rainfallrunoff modeling.” Journal of Hydrology 349.1-2 (2008): 115-124.

[2] Schmitz, G. H., and J. Cullmann. “PAI-OFF: A new proposal for online flood forecasting in flash flood prone catchments.” Journal of hydrology 360.1-4 (2008): 1-14.

[3] Riordan, Denis, and Bjarne K. Hansen. “A fuzzy casebased system for weather prediction.” Engineering Intelligent Systems for Electrical Engineering and Communications 10.3 (2002): 139-146.

[4] Guhathakurta, P. “Long-range monsoon rainfall prediction of 2005 for the districts and sub-division Kerala with artificial neural network.” Current Science 90.6 (2006): 773-779.

[5] Pilgrim, D. H., T. G. Chapman, and D. G. Doran. “Problems of rainfall-runoff modelling in arid and semiarid regions.” Hydrological Sciences Journal 33.4 (1988): 379-400.

[6] Lee, Sunyoung, Sungzoon Cho, and Patrick M. Wong. “Rainfall prediction using artificial neural networks.” journal of geographic information and Decision Analysis 2.2 (1998): 233-242..

[7] French, Mark N., Witold F. Krajewski, and Robert R. Cuykendall. “Rainfall forecasting in space and time using a neural network.” Journal of hydrology 137.1-4 (1992): 1-31.

[8] Charaniya, Nizar Ali, and Sanjay V. Dudul. “Committee of artificial neural networks for monthly rainfall prediction using wavelet transform.” Business, Engineering and Industrial Applications (ICBEIA), 2011 International Conference on. IEEE, 2011.

[9] Noone, David, and Harvey Stern. “Verification of rainfall forecasts from the Australian Bureau of Meteorology‟s Global Assimilation and Prognosis(GASP) system.” Australian Meteorological Magazine 44.4 (1995): 275-286.

[10] Hornik, Kurt, Maxwell Stinchcombe, and Halbert White. “Multilayer feedforward networks are universal approximators.” Neural networks 2.5 (1989): 359-366.

[11] Haykin, Simon. Neural networks: a comprehensive foundation. Prentice Hall PTR, 1994.

[12] Rajeevan, M., Pulak Guhathakurta, and V. Thapliyal. “New models for long range forecasts of summer monsoon rainfall over North West and Peninsular India.” Meteorology and Atmospheric Physics 73.3-4 (2000): 211-225.