**Weather Rainfall Prediction**

**TABLE OF CONTENTS**

|  |  |  |
| --- | --- | --- |
| **CHAPTER** | **TITLE** | **PAGE NO** |
|  | **ABSTRACT** |  |
|  | **INTRODUCTION** |  |
|  | * 1. General Introduction |  |
|  | * 1. Project Objectives |  |
|  | **SYSTEM** **PROPOSAL** |  |
|  | * 1. Existing System |  |
|  | * + 1. Disadvantages |  |
|  | * 1. Proposed System |  |
|  | 2.2.1 Advantages |  |
|  | * 1. Literature Survey |  |
|  | **SYSTEM DIAGRAMS** |  |
|  | * 1. Architecture Diagram |  |
|  | * 1. Flow Diagram |  |
|  | * 1. UML Diagrams |  |
|  | **IMPLEMENTATION** |  |
|  | * 1. Modules |  |
|  | * 1. Modules Description |  |
|  | **SYSTEM** **REQUIREMENTS** |  |
|  | * 1. Hardware Requirements |  |
|  | * 1. Software Requirements |  |
|  | * 1. Software Description |  |
|  | * 1. Testing of Products |  |
|  | **CONCLUSION** |  |
| **7.** | **FUTURE** **ENHANCEMENT** |  |
| **8.** | **SAMPLE** **CODING** |  |
| **9.** | **SAMPLE** **SCREENSHOT** |  |
| **10.** | **REFERENCES** |  |

**ABSTRACT**

Agriculture is a sector that plays a crucial role in the economies of many countries around the globe, like India where it contributes 16% of the total economy. Weather forecasting is one of the challenges faced by this sector, due to its dynamic and turbulent nature, the statistical methods fail to provide forecasting at an accurate precision. This paper aims to develop an accurate way to predict the temperature forecast using machine learning techniques especially using Long Short Term memory networks (LSTM) and random forest classifier. Despite the advances made, there are still significant obstacles to overcome in expanding the use of weather forecasts in the agricultural sector due to the dynamics in climate changes. These include the need for improved model accuracy, quantitative evidence of the utility of climate predictions as instruments for agricultural risk management and addressing major chances of disease incidence which are usually seasonal and depends on parameters like temperature and rainfall. The goal of this study is to forecast parameters that could help farmers to make an informed decision so as to reduce the losses by taking required proactive measures. The system is developed the machine learning algorithm for detecting or forecasting the weather effectively. The system is also developed the machine learning algorithm such as Logistic Regression and Xgboost. The experimental results shows that accuracy, precision, recall, f1-score, ROC curve and confusion matrix.

**CHAPTER 1**

**INTRODUCTION**

* 1. **General Introduction:**

Weather forecasting is one of the crucial and complex tasks that is consummated by meteorologists. Weather forecasting answers the basic questions like what is the weather expected to be tomorrow? Is it going to rain today? One may conclude whether it is going to be sunny, foggy, misty, or cloudy on a particular day and plan their business accordingly. Considering the significance of forecasting in everyday life, meteorologists strive for near accuracy in their predictions.

The agriculture sector is another field that depends on the forecast to a vast extent. Weather accounts for the annual profit or loss of farm production directly or indirectly. In many countries, agriculture is the main source for their economic development. Crop loss can be reduced by making adjustments based on timely and accurate weather forecasts. So, getting a gist of the factors like the amount of humidity precipitation, temperature, upcoming rainfall, or precisions of natural disasters like floods, droughts, storms, hurricanes, etc. helps the farmers to manage their jobs, minimize the damage of property, and selecting crops that are most suited to the predicted climatic conditions. To successfully foster growth and ensure food security in this changing environment, accurate weather prediction is needed.

There are many techniques and algorithms that are used for predictions. Weather forecasting has time series data and in this paper a temperature prediction using Auto Regressive Integrated Moving Average (ARIMA) model and LSTM is deployed. It helps to predict the temperature of a particular season which is beneficial in agriculture to be known well in advance for early identification and hence mitigation of diseases in crops. The machine learning models usually consider three types of input features usually to be fed into the models. One consists of using the meteorological or environmental variables. Second category uses the historical temperature data and the final one which combines the former and later. Similarly the performance of the machine learning models hugely depends on the horizon of forecast time whether it is a short term prediction or long time. Another factor which affects the performance is the spatial scale. Global scale predictions are found to have smaller errors than the local scale predictions which are done for one particular station.

Weather simply refers to the condition of air on the earth at a given place and time. It is a continuous, data-intensive, multidimensional, dynamic and chaotic process. These properties make weather forecasting is a formidable challenge. Forecasting is the process of estimation in unknown situations from the historical data. Weather forecasting is one of the most scientifically and technologically challenging problems around the world in the last century. To make an accurate prediction is indeed, one of the major challenges that meteorologists are facing all over the world. Since ancient times, weather prediction has been one of the most interesting and fascinating domains. Scientists have tried to forecast meteorological characteristics using a number of methods, some of these methods being more accurate than others.

Knowledge of meteorology forms the basis of scientific weather forecasting, which revolves around predicting the state of the atmosphere for a given location. Weather forecasting as practiced by humans is an example of having to make judgments in the presence of uncertainty. Weather forecasts are often made by collecting quantitative data about the current state of the atmosphere and using scientific understanding of atmospheric processes to project how the atmosphere will evolve in future. Over the last few years the necessity of increasing knowledge about the cognitive process in weather forecasting has been recognized. For human practitioners, forecasting the weather becomes a task for which the details can be uniquely personal, although most human forecasters use approaches based on the science of meteorology in common to deal with the challenges of the task.

Weather forecasting entails predicting how the present state of the atmosphere will change. Present weather conditions are obtained by ground observations, observations from ships, observation from aircraft, radio sounds, doppler radar and satellites. This information is sent to meteorological centers where the data are collected, analyzed and made into a variety of charts, maps and graphs. Modern high-speed computers transfer the many thousands of observations onto surface and upper-air maps.

Weather forecasts provide critical information about future weather. There are various techniques involved in weather forecasting, from relatively simple observation of the sky to highly complex computerized mathematical models. Weather prediction could be one day/one week or a few months ahead. The accuracy of weather forecasts however, falls significantly beyond a week. Weather forecasting remains a complex business, due to its chaotic and unpredictable nature.

It remains a process that is neither wholly science nor wholly art. It is known that persons with little or no formal training can develop considerable forecasting skill For example, farmers often are quite capable of making their own short term forecasts of those meteorological factors that directly influence their livelihood, and a similar statement can be made about pilots, fishermen, mountain climbers, etc. Weather phenomena, usually of a complex nature, have a direct impact on the safety and/or economic stability of such persons. Accurate weather forecast models are important to third world countries, where the entire agriculture depends upon weather. It is thus a major concern to identify any trends for weather parameters to deviate from its periodicity, which would disrupt the economy of the country. This fear has been aggravated due to threat by the global warming and greenhouse effect. The impact of extreme weather phenomena on society is growing more and more costly, causing infrastructure damage, injury and the loss of life.

As practiced by the professionally trained meteorologist, weather forecasting today is a highly developed skill that is grounded in scientific principle and method and that makes use of advanced technological tools. The notable improvement in forecast accuracy that has been achieved since 1950 is a direct outgrowth of technological developments, basic and applied research, and the application of new knowledge and methods by weather forecasters. High-speed computers, meteorological satellites, and weather radars are tools that have played major roles in improving weather forecasts.

Several other factors have contributed significantly to this increase in forecasting accuracy. One is the development of statistical methods for enhancing the scope and accuracy of model predictions. Another is the improved observational capability afforded by meteorological satellites. A third primary reason for the increase in accuracy is the continued improvement of the initial conditions prepared for the forecast models. Statistical methods allow a wider variety of meteorological elements to be predicted than do the models alone, and they tailor the geographically less precise model forecasts to specific locations. Satellites now provide the capability for nearly continuous viewing and remote sensing of the atmosphere on a global scale. The improvement in initial conditions is the result of an increased number of observations and better use of the observations in computational techniques.

**Traditional weather forecasting**: Weather forecasting is the application of science and technology to predict the state of the atmosphere for a given location. Weather forecasts are made by collecting quantitative data about the current state of the atmosphere and using scientific understanding of atmospheric processes to project how the atmosphere will evolve. There are a variety of end users to weather forecasts. Weather warnings are important forecasts because they are used to protect life and property.

In ancient times, forecasting was mostly based on weather pattern observation. Over the years, the study of weather patterns has resulted in various techniques for rainfall forecasting. Present rainfall forecasting embodies a combination of computer models, interpretation, and an acquaintance of weather patterns. The following technique was used for existing weather prediction. Use of a barometer

Measurements of barometric pressure and the pressure tendency have been used in forecasting since the late 19th century. The larger the change in pressure, the larger the change in weather can be expected. If the pressure drop is rapid, a low pressure system is approaching, and there is a greater chance of rain.

**Looking at the sky**

Along with pressure tendency, the condition of the sky is one of the most important parameters used to forecast weather in mountainous areas. Thickening of cloud cover or the invasion of a higher cloud deck is an indication of rain in the near future. At night, high thin clouds can lead to halos around the moon, which indicates the approach of a warm front and its associated rain. Morning fog portends fair conditions, as rainy conditions are preceded by wind or clouds which prevent fog formation

**Nowcasting**

The forecasting of the weather within the next six hours is often referred to as now casting. In this time range, it is possible to forecast smaller features such as individual showers and thunderstorms with reasonable accuracy, as well as other features too small to be resolved by a computer model. A human, given the latest radar, satellite and observational data will be able to make a better analysis of the small scale features present and so will be able to make a more accurate forecast for the following few hours [RR03].

**Numerical Weather Prediction model**

Numerical Weather Prediction (NWP) is the science of predicting the weather using models of the atmosphere and computational techniques. Current weather conditions are used at the input of the mathematical models of the atmosphere to predict the weather. This model usually provides surrounding point around the wind farm with a spatial resolution of a few kilometres.

NWP uses the power of computers to make a forecast. A forecaster examines how the features predicted by the computer will interact to produce the day's weather. The NWP method is flawed in that the equations used by the models to simulate the atmosphere are not precise [Ry02].

A number of weather forecasting agencies operate modelling centers where supercomputers are used to run NWP models that span the entire globe. These include the National Center for Environmental Prediction (NCEP) in the United States, the United Kingdom Meteorological Office (UKMO), and the European Centre for Medium-range Weather Forecasts (ECMWF). Although costly, a global approach to NWP is essential, especially for long-range forecasting. For this reason, achieving accurate forecasts requires an accurate analysis from which to get the model started. This involves a computer-based process called data assimilation, in which the most recent weather observations from around the world are combined with model forecasts to create a global analysis of current conditions. This becomes the starting point for the next run of the NWP model, and is the computer equivalent of the manual analysis cycle that forecasters carry out on an on-going basis. Global models play a key role in modern weather forecasting, and meteorologists at Met Service routinely use the NCEP, UKMO and ECMWF models to assist with day-to-day production of forecasts and weather warnings. These models give insight into the behavior of weather systems on a large scale, without much emphasis on local detail.

**Ensemble Forecasting**

To predict the weather forecast meteorologists have developed atmospheric models that approximate the atmosphere by using ensemble forecasting to describe how atmospheric temperature, pressure and moisture will change over time. The equations are programmed into a computer and the data on the present atmospheric conditions are fed into the computer. The computer solves the equations to determine how the different atmospheric variables will change over the next few minutes. The computer repeats this procedure again and again using the output from one cycle as the input for the next cycle. For some desired time in the future, the computer prints its calculated information. It then analyses the data, drawing the lines for the projected position of the various pressure systems. A forecaster uses the prognostic chart as a guide to predicting the weather. There are many atmospheric models that represent the atmosphere, with each one interpreting the atmosphere in a slightly different way. Weather forecasts made for 12 and 24 hours are typically accurate. Forecasts made for two or three days are usually good. Beyond above five days, forecast accuracy falls off rapidly.

Weather information can also come from remote sensing, particularly radar and satellites.

**Weather satellites**

Weather satellites have been increasingly important sources of weather data since the first one was launched in 1952. Weather satellites are the best way to monitor large scale systems, like storms. Satellites can also monitor the spread of ash from a volcanic eruption, smoke from fires, and pollution. They are able to record long-term changes. Figure 1.1 shows one of the geostationary satellites that monitors conditions over the world. Weather satellites may observe all energy from all wavelengths in the electromagnetic spectrum. Most important are the visible light and infrared (heat) frequencies.

* 1. **Objectives:**

The main objective of our project is,

* To detect or to forecast or to analyze the weather effectively.
* To implement the ARIMA model for forecasting the data.
* To implement the different classification like deep and machine learning algorithms for better performances.
* To enhance the overall performance for classification algorithms.

**CHAPTER 2**

**SYSTEM PROPOSAL**

**2.1 EXISTING SYSTEM:**

In existing system, Agriculture is a sector that plays a crucial role in the economies of many countries around the globe, like India where it contributes 16% of the total economy. Weather forecasting is one of the challenges faced by this sector, due to its dynamic and turbulent nature, the statistical methods fail to provide forecasting at an accurate precision. This paper aims to develop an accurate way to predict the temperature forecast using machine learning techniques especially using Long Short Term memory networks (LSTM). Despite the advances made, there are still significant obstacles to overcome in expanding the use of weather forecasts in the agricultural sector due to the dynamics in climate changes. These include the need for improved model accuracy, quantitative evidence of the utility of climate predictions as instruments for agricultural risk management and addressing major chances of disease incidence which are usually seasonal and depends on parameters like temperature and rainfall. The goal of this study is to forecast parameters that could help farmers to make an informed decision so as to reduce the losses by taking required proactive measures. This paper provides a detailed analysis of weather forecasting techniques and explores future research goals in this field.

**2.1.1 DISADVANTAGES:**

* It doesn’t efficient for large volume of data’s
* Theoretical limits.
* It doesn’t implement the machine learning.
* The process is implemented without removing the unwanted data.
  1. **PROPOSED SYSTEM:**

In this system, the weather dataset was taken as input. The input data was taken from the dataset repository. Then, we have to implement the data preprocessing step.in this step, we have to handle the missing values for avoid wrong prediction. If there is present any missing values in our input data, we have to replace the missing values by zero or Nan values. Then we have to implement the ARIMA model for forecasting or predicting the weather. Next, we have to implement the data splitting. In this step, we have to split the data into test is used for prediction and train is used for evaluation. Then we can implement the different machine learning algorithms such as xgboost and logistic regression Classifier for detecting or forecasting the weather. Finally, the experimental results shows that the performance metrics such as accuracy, precision, recall, ROC curve and confusion matrix. Then we can compare the results for both algorithms based on accuracy.

**2.2.1 ADVANTAGES:**

* It is efficient for large number of datasets.
* Time consumption is low.
* The process is implemented with removing the unwanted data.

**2.3 LITERATURE SURVEY:**

**2.3.1Climate change detection in penang island using deterministic interpolation methods, 2020**

**Author*:*** Chang Kok Yang1, Fam Pei Shan2, Tay Lea Tien3

**Methodology:**

This study focuses mainly on investigating the indication of climate change in Penang Island over the period of 2003-2018 by utilising sound application procedures of proven analysis methods. Two deterministic interpolation methods are used to produce new estimation points based on the precipitation data to enrich the monitoring network of rainfall stations in Penang Island. Monthly and monthly-average precipitation maps for Penang Island are produced by using inverse distance weighting interpolation method. Results reveal that seven out of twelve months of a year show increasing precipitation trends over the period of study and March is the only month that shows a decreasing trend in precipitation. Monthly-average precipitation in Penang Island also displays a gradual trend of precipitation increase over the period of study, further conforming the finding of monthly precipitation increase over the period of study. The finding of this study provides insight for local agriculturists and ministry to make better decision in response to climate change in Penang.

**Advantage:**

* The prediction is accurate.

**Disadvantage:**

* Time consumption is high.
* Theoretical limits

# **2.3.2Climate change now detectable from any single day of weather at global scale,2019**

**Author:** Sebastian Sippel  1,2,3\*, Nicolai Meinshausen2, Erich M. Fischer  1, Enikő Székely  4 and Reto Knutti

# **Methodology:**

Our detection approach invokes statistical learning and climate model simulations to encapsulate the relationship between spatial patterns of daily temperature and humidity, and key climate change metrics such as annual global mean temperature or Earth’s energy imbalance. Observations are projected onto this relationship to detect climate change. The fingerprint of climate change is detected from any single day in the observed global record since early 2012, and since 1999 on the basis of a year of data. Detection is robust even when ignoring the long-term global warming trend. This complements traditional climate change detection, but also opens broader perspectives for the communication of regional weather events, modifying the climate change narrative: while changes in weather locally are emerging over decades, global climate change is now detected instantaneously.

**Advantage**:

* The advantages of this approach include, first, that the spatial pattern response to external forcing is encapsulated in the fingerprint, but regions with large internal variability or where different climate models disagree with each other receive less weight in the fingerprint.

**Disadvantage:**

* The prediction is not accurate.
* Less prediction

**2.3.3 Climate Change Assessment using Climate Indices Approach: A Brief Overview, 2020**

# **Author:** Waleed Ahmad

# **Methodology:**

Climate of the world is increasingly becoming unpredictable in terms of the prevailing hydro-meteorological trends and the subsequent impact on the water resources of different regions by changing frequency, duration and magnitude. In order to better assess such climate impacts for the future, Global and Regional Climate Models are utilized to accurately simulate the most likely changes in the climate based on the current prevailing trends. World Climate Research Program (WCRP) approved Expert Team on Climate Change Detection Monitoring and Indices (ETCCDMI) Climate Indices technique is one of the widely used approach in assessing such trends in climate based on meteorological variables of the regions. Climate Indices approach utilizes daily temperature and precipitation variables on daily time steps to compute 27 core indices for assessing changing trends in the climate of the region. This study illustrates the use of climate indices approach for climate change assessment.

**Advantage:**

* Accuracy is high.
* The persistent problem that researchers faced while canvassing for evidence of changing climate was the lack and inadequate cache of observed historical weather data for consistent and homogenous portrayal of climate change and hence, thereof, the inability to detect high quality and consistent changes in climate variables.

**Disadvantage:**

* The process is implemented without removing unwanted data.

**CHAPTER 3**

**SYSTEM DIAGRAMS**

**3.1 SYSTEM ARCHITECTURE:**

Input Data

Data preprocessing

Data Splitting

Classification

Result Generation

FIGURE 3.1: SYSTEM ARCHITECTURE

**3.2 FLOW DIAGRAM**

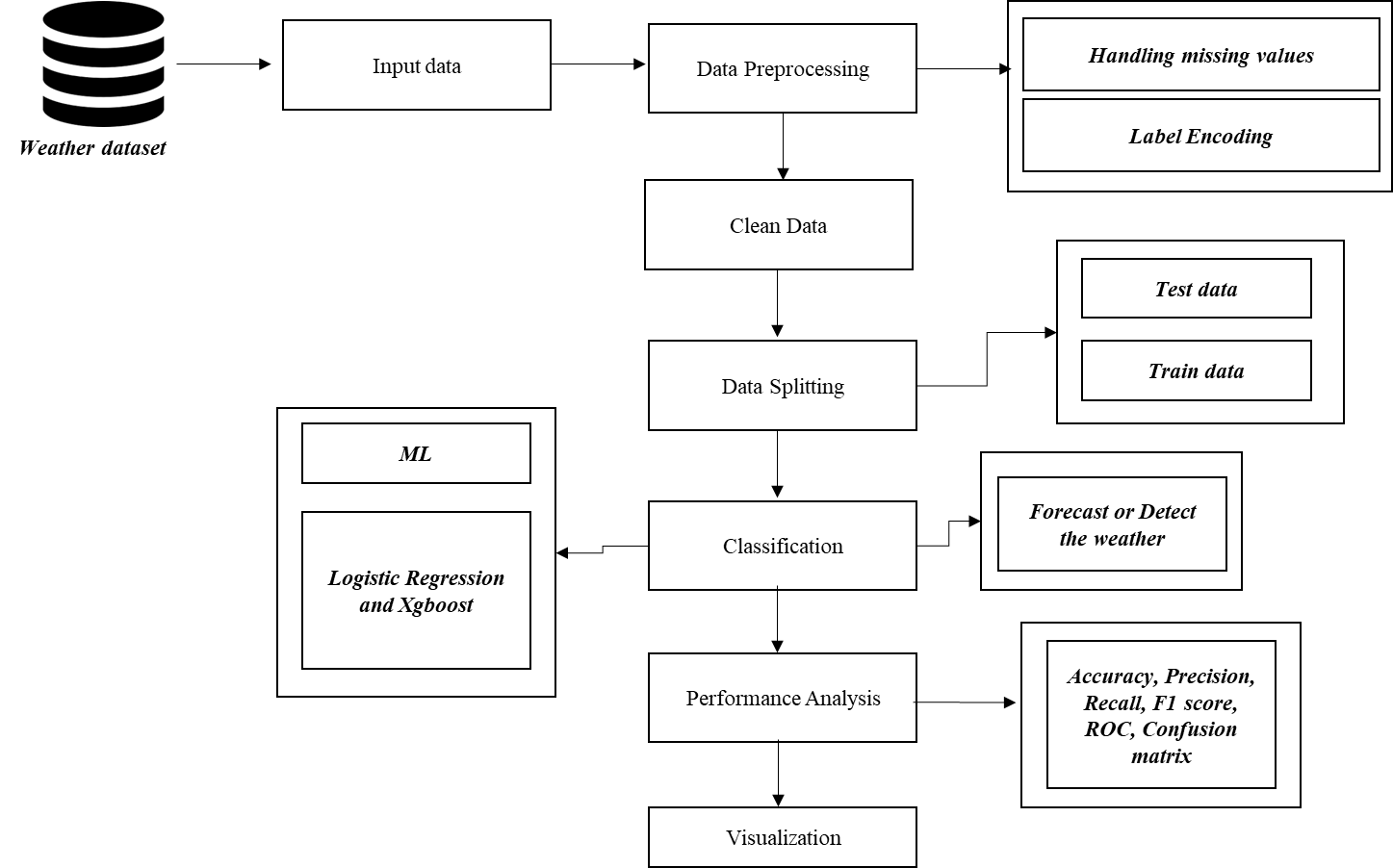
****

FIGURE 3.2: FLOW DIAGRAM

**3.3 UML DIAGRAMS:**

UML stands for Unified Modelling Language. UML is a standardized general-purpose modelling 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 Modelling Language is a standard language for specifying, Visualization, Constructing and documenting the artifacts of software system, as well as for business modelling and other non-software systems.

The UML represents a collection of best engineering practices that have proven successful in the modelling 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.

**3.3.1 USE CASE DIAGRAM:**

Use-case diagrams describe the high-level functions and scope of a system. These diagrams also identify the interactions between the system and its actors. The use cases and actors in use-case diagrams describe what the system does and how the actors use it, but not how the system operates internally.

A use case is a list of actions or event steps typically defining the interactions between a role (known in the Unified Modelling Language (UML) as an actor) and a system to achieve a goal. The actor can be a human or other external system.

UML use case diagrams are ideal for:

* Representing the goals of system-user interactions
* Defining and organizing functional requirements in a system
* Specifying the context and requirements of a system
* Modelling the basic flow of events in a use case

**Notations:**

* **Use cases**: Horizontally shaped ovals that represent the different uses that a user might have.
* **Actors**: Stick figures that represent the people actually employing the use cases.
* **Associations**: A line between actors and use cases. In complex diagrams, it is important to know which actors are associated with which use cases.
* **System boundary boxes**: A box that sets a system scope to use cases. All use cases outside the box would be considered outside the scope of that system. For example, Psycho Killer is outside the scope of occupations in the chainsaw example found below.
* **Packages**: A UML shape that allows you to put different elements into groups. Just as with component diagrams, these groupings are represented as file folders.

System

User

FIGURE 3.3.1: USE CASE DIAGRAM

**3.3.2 ACTIVITY DIAGRAM:**

This shows the flow of events within the system. The activities that occur within a use case or within an objects behaviour typically occur in a sequence. An activity diagram is designed to be simplified look at what happens during an operations or a process. Each activity is represented by a rounded rectangle the processing within an activity goes to compilation and then an automatic transmission to the next activity occurs. An arrow represents the transition from one activity to the next. An activity diagram describes a system in terms of activities. Activities are the state that represents the execution of a set of operations.

These are similar to flow chart diagram and dataflow.

**Initial state**: which state is starting the process?

**Action State**: An action state represents the execution of an atomic action, typically the invocation of an operation. An action state is a simple state with an entry action whose only exit transition is triggered by the implicit event of completing the execution of the entry action.

**Transition**: A transition is a directed relationship between a source state vertex and a target state vertex. It may be part of a compound transition, which takes the static machine from one static configuration to another, representing the complete response of the static machine to a particular event instance.

**Final state:** A final state represents the last or "final" state of the enclosing composite state. There may be more than one final state at any level signifying that the composite state can end in different ways or conditions.

When a final state is reached and there are no other enclosing states it means that the entire state machine has completed its transitions and no more transitions can occur.

**Decision**: A state diagram (and by derivation an activity diagram) expresses decision when guard conditions are used to indicate different possible transitions that depend on Boolean conditions of the owning object.

Input Data

Preprocessing

Data Splitting

Result Generation

Classification

FIGURE 3.3.2: ACTIVITY DIAGRAM

**3.3.3 SEQUENCE DIAGRAM:**

Sequence diagrams document the interactions between classes to achieve a result, such as a use case. Because UML is designed for object-oriented programming, these communications between classes are known as messages. The Sequence diagram lists objects horizontally, and time vertically, and models these messages over time.

**Graphical Notation**: In a Sequence diagram, classes and actors are listed as columns, with vertical lifelines indicating the lifetime of the object over time.

**Object**: Objects are instances of classes, and are arranged horizontally. The pictorial representation for an Object is a class (a rectangle) with the name prefixed by the object.

**Lifeline** The Lifeline identifies the existence of the object over time. The notation 2for a Lifeline is a vertical dotted line extending from an object.

**Activation**: Activations, modelled as rectangular boxes on the lifeline, indicate when the object is performing an action.

**Message**: Messages, modelled as horizontal arrows between Activations.

Input Data

Preprocessing

Data Splitting

Classification

Select data

Clean data

ML

Load data

Data splitting

Weather prediction

FIGURE 3.3.3: SEQUENCE DIAGRAM

**3.3.4 ER DIAGRAM:**

An Entity Relationship (ER) Diagram is a type of flowchart that illustrates how “entities” such as people, objects or concepts relate to each other within a system.

ER Diagrams are most often used to design or debug relational databases in the fields of software engineering, business information systems, education and research.

Also known as ERDs or ER Models, they use a defined set of symbols such as rectangles, diamonds, ovals and connecting lines to depict the interconnectedness of entities, relationships and their attributes.

They mirror grammatical structure, with entities as nouns and relationships as verbs.

**Notation:**

### **Entity**

A definable thing—such as a person, object, concept or event—that can have data stored about it. Think of entities as nouns. Examples: a customer, student, car or product. Typically shown as a rectangle.

**Entity type:**A group of definable things, such as students or athletes, whereas the entity would be the specific student or athlete. Other examples: customers, cars or products.

**Entity set:** Same as an entity type, but defined at a particular point in time, such as students enrolled in a class on the first day.

Other examples: Customers who purchased last month, cars currently registered in Florida. A related term is instance, in which the specific person or car would be an instance of the entity set.

**Entity categories:** Entities are categorized as strong, weak or associative. A **strong entity** can be defined solely by its own attributes, while a **weak entity** cannot. An associative entity associates entities (or elements) within an entity set.

**Entity keys:** Refers to an attribute that uniquely defines an entity in an entity set. Entity keys can be super, candidate or primary. **Super key:**A set of attributes (one or more) that together define an entity in an entity set.

**Candidate key:**A minimal super key, meaning it has the least possible number of attributes to still be a super key. An entity set may have more than one candidate key. **Primary key:**A candidate key chosen by the database designer to uniquely identify the entity set. **Foreign key:**Identifies the relationship between entities.

### **Relationship**

How entities act upon each other or are associated with each other. Think of relationships as verbs.

For example, the named student might register for a course.

The two entities would be the student and the course, and the relationship depicted is the act of enrolling, connecting the two entities in that way.

Relationships are typically shown as diamonds or labels directly on the connecting lines.

Data selection

Preprocessing

Data Spliitng

Classification

FIGURE 3.3.4: ER DIAGRAM

**3.3.6 CLASS DIAGRAM:**

Class diagrams identify the class structure of a system, including the properties and methods of each class. Also depicted are the various relationships that can exist between classes, such as an inheritance relationship.

Part of the popularity of Class diagrams stems from the fact that many CASE tools, such as Rational XDE, will auto-generate code in a variety of languages, these tools can synchronize models and code, reducing the workload, and can also generate Class diagrams from object-oriented code.

**Graphical Notation:** The elements on a Class diagram are classes and the relationships between them.

**Class**: Classes are building blocks in object-oriented programming. A class is depicted using a rectangle divided into three section.

The top section is name of class; the middle section defines the properties of class. The bottom section list the methods of the class.

**Association:** An Association is a generic relationship between two classes, and is modelled by a line connecting the two classes.

This line can be qualified with the type of relationship, and can also feature multiplicity rule (e.g. one-to-one, one-to-many, many-to-many) for the relationship.

Select data ()

Load data ()

View data ()

Input

Test ()

Train ()

Data SPlitting

LR ()

Xgboost ()

Result ()

Classification

Preprocessing

Missing values ()

Data splitting ()

Weather forecast ()

Prediction

FIGURE 3.3.5: CLASS DIAGRAM

**CHAPTER 4**

**IMPLEMENTATION**

**4.1 MODULES:**

* Data selection
* Data preprocessing
* Data splitting
* Classification
* Prediction
* Result Generation

**4.2 MODULES DESCRIPTION:**

**4.2.1: DATA SELECTION:**

* The input data was collected from dataset repository like UCI, Kaggle or Github.
* In our process, the weather dataset is used.
* The dataset which contains the information such as Formatted Date , Summary, Temperature (C), Apparent Temperature (C) , Humidity , Wind Speed (km/h) , Wind Bearing (degrees) , Visibility (km), Loud Cover , Pressure (millibars) and Daily Summary.
* In python, we have to read or load our input dataset by using the panda’s packages.
* Our dataset, is in the form of ‘.csv’ file extension.

**4.2.2: DATA PREPROCESSING:**

* Data pre-processing is the process of removing the unwanted data from the dataset.
* Pre-processing data transformation operations are used to transform the dataset into a structure suitable for machine learning.
* This step also includes cleaning the dataset by removing irrelevant or corrupted data that can affect the accuracy of the dataset, which makes it more efficient.
* Data processing is one of the most common tasks in many ML applications. This technique is used to transform raw data into a useful and efficient format.
* To do the analysis, the dataset needs to be cleaned, standardized, and noise-free. The entire process is known as text preprocessing.
* Missing data removal
* Label encoding
* **Missing data removal**: In this process, the null values such as missing values and Nan values are replaced by 0.
* Missing and duplicate values were removed and data was cleaned of any abnormalities.
* **Label** **encoding**: Here, we can convert the string into numeric integer value.

**4.2.4: DATA SPLITTING:**

* During the machine learning process, data are needed so that learning can take place.
* In addition to the data required for training, test data are needed to evaluate the performance of the algorithm in order to see how well it works.
* In our process, we considered 70% of the dataset to be the training data and the remaining 30% to be the testing data.
* Data splitting is the act of partitioning available data into two portions, usually for cross-validator purposes.
* One Portion of the data is used to develop a predictive model and the other to evaluate the model's performance.
* Separating data into training and testing sets is an important part of evaluating data mining models.
* Typically, when you separate a data set into a training set and testing set, most of the data is used for training, and a smaller portion of the data is used for testing.

**4.2.5: CLASSIFICATION:**

* In machine learning, classification refers to a predictive modelling problem where a class label is predicted for a given example of input data.
* Classification is the task of predicting a discrete class label. Regression is the task of predicting a continuous quantity.
* In machine learning, classification is a supervised learning concept which basically categorizes a set of data into classes.
* Before classification, we should have split the data into test and train.
* Most of data’s are used for training and smaller portion of the data’s are used for testing.
* Training data is used for evaluate the model and testing data is used for predictive the model.
* After data splitting, we have to implement the classification algorithm.
* In our process, we have to implement the different machine learning algorithm such as logistic regression and xgboost for detecting the weather effectively.
* GBoost, which stands for **Extreme Gradient Boosting**, is a scalable, distributed gradient-boosted decision tree (GBDT) machine learning library. It provides parallel tree boosting and is the leading machine learning library for regression, classification, and ranking problems.
* **Logistic regression** is used to obtain odds ratio in the presence of more than one explanatory variable. The procedure is quite similar to multiple linear regression, with the exception that the response variable is binomial. The result is the impact of each variable on the odds ratio of the observed event of interest.

**4.2.6: RESULT GENERATION:**

The Final Result will get generated based on the overall classification and prediction. The performance of this proposed approach is evaluated using some measures like,

* **Accuracy**

Accuracy of classifier refers to the ability of classifier. It predicts the class label correctly and the accuracy of the predictor refers to how well a given predictor can guess the value of predicted attribute for a new data.

AC= (TP+TN)/ (TP+TN+FP+FN)

* **Precision**

Precision is defined as the number of true positives divided by the number of true positives plus the number of false positives.

Precision=TP/ (TP+FP)

* **Recall**

Recall is the number of correct results divided by the number of results that should have been returned. In binary classification, recall is called sensitivity. It can be viewed as the probability that a relevant document is retrieved by the query.

Recall=TP/ (TP+FN)

* **F1 score:**

The F1-score combines the precision and recall of a classifier into a single metric by taking their harmonic mean. It is primarily used to compare the performance of two classifiers. Suppose that classifier A has a higher recall, and classifier B has higher precision.

F1 score=2\*(Pre \* Sen)/(Pre+Sen)

* **ROC curve:**

The ROC curve shows the trade-off between sensitivity (and TPR) and specificity (1 – FPR). Classifiers that give curves closer to the top-left corner indicate a better performance. As a baseline, a random classifier is expected to give points lying along the diagonal (FPR = TPR).

* **Confusion Matrix:**

The confusion matrix is a matrix used to determine the performance of the classification models for a given set of test data. It can only be determined if the true values for test data are known. The matrix itself can be easily understood, but the related terminologies may be confusing.

**CHAPTER 5**

**SYSTEM REQUIREMENTS**

**5.1 HARDWARE REQUIREMENTS:**

* System : Pentium IV 2.4 GHz
* Hard Disk : 200 GB
* Mouse : Logitech.
* Keyboard : 110 keys enhanced
* Ram : 4GB

**5.2 SOFTWARE REQUIREMENTS:**

* O/S : Windows 7.
* Language : Python
* Front End : Anaconda Navigator – Spyder

**5.3 SOFTWARE DESCRIPTION:**

**5.3.1 Python**

Python is one of those rare languages which can claim to be both *simple* and powerful. You will find yourself pleasantly surprised to see how easy it is to concentrate on the solution to the problem rather than the syntax and structure of the language you are programming in. The official introduction to Python is Python is an easy to learn, powerful programming language. It has efficient high-level data structures and a simple but effective approach to object-oriented programming. Python's elegant syntax and dynamic typing, together with its interpreted nature, make it an ideal language for scripting and rapid application development in many areas on most platforms. I will discuss most of these features in more detail in the next section.

## **5.3.2 Features of Python**

### **Simple**

Python is a simple and minimalistic language. Reading a good Python program feels almost like reading English, although very strict English! This pseudo-code nature of Python is one of its greatest strengths. It allows you to concentrate on the solution to the problem rather than the language itself.

### **Easy to Learn**

As you will see, Python is extremely easy to get started with. Python has an extraordinarily simple syntax, as already mentioned.

### **Free and Open Source**

Python is an example of a FLOSS (Free/Libré and Open Source Software). In simple terms, you can freely distribute copies of this software, read its source code, make changes to it, and use pieces of it in new free programs. FLOSS is based on the concept of a community which shares knowledge. This is one of the reasons why Python is so good - it has been created and is constantly improved by a community who just want to see a better Python.

### **High-level Language**

When you write programs in Python, you never need to bother about the low-level details such as managing the memory used by your program, etc.

### **Portable**

Due to its open-source nature, Python has been ported to (i.e. changed to make it work on) many platforms. All your Python programs can work on any of these platforms without requiring any changes at all if you are careful enough to avoid any system-dependent features.

You can use Python on GNU/Linux, Windows, FreeBSD, Macintosh, Solaris, OS/2, Amiga, AROS, AS/400, BeOS, OS/390, z/OS, Palm OS, QNX, VMS, Psion, Acorn RISC OS, VxWorks, PlayStation, Sharp Zaurus, Windows CE and PocketPC!

You can even use a platform like [Kivy](http://kivy.org) to create games for your computer and for iPhone, iPad, and Android.

### **Interpreted**

This requires a bit of explanation.

A program written in a compiled language like C or C++ is converted from the source language i.e. C or C++ into a language that is spoken by your computer (binary code i.e. 0s and 1s) using a compiler with various flags and options. When you run the program, the linker/loader software copies the program from hard disk to memory and starts running it.

Python, on the other hand, does not need compilation to binary. You just run the program directly from the source code. Internally, Python converts the source code into an intermediate form called bytecodes and then translates this into the native language of your computer and then runs it. All this, actually, makes using Python much easier since you don't have to worry about compiling the program, making sure that the proper libraries are linked and loaded, etc. This also makes your Python programs much more portable, since you can just copy your Python program onto another computer and it just works!

### **Object Oriented**

Python supports procedure-oriented programming as well as object-oriented programming. In procedure-oriented languages, the program is built around procedures or functions which are nothing but reusable pieces of programs. In object-oriented languages, the program is built around objects which combine data and functionality. Python has a very powerful but simplistic way of doing OOP, especially when compared to big languages like C++ or Java.

### **Extensible**

If you need a critical piece of code to run very fast or want to have some piece of algorithm not to be open, you can code that part of your program in C or C++ and then use it from your Python program.

### **Embeddable**

You can embed Python within your C/C++ programs to give scripting capabilities for your program's users.

### **Extensive Libraries**

The Python Standard Library is huge indeed. It can help you do various things involving regular expressions, documentation generation, unit testing, threading, databases, web browsers, CGI, FTP, email, XML, XML-RPC, HTML, WAV files, cryptography, GUI (graphical user interfaces), and other system-dependent stuff. Remember, all this is always available wherever Python is installed. This is called the Batteries Included philosophy of Python.

Besides the standard library, there are various other high-quality libraries which you can find at the Python Package Index.

**5.4 TESTING PRODUCTS:**

System testing is the stage of implementation, which aimed at ensuring that system works accurately and efficiently before the live operation commence. Testing is the process of executing a program with the intent of finding an error. A good test case is one that has a high probability of finding an error. A successful test is one that answers a yet undiscovered error.

Testing is vital to the success of the system. System testing makes a logical assumption that if all parts of the system are correct, the goal will be successfully achieved. . A series of tests are performed before the system is ready for the user acceptance testing. Any engineered product can be tested in one of the following ways. Knowing the specified function that a product has been designed to from, test can be conducted to demonstrate each function is fully operational. Knowing the internal working of a product, tests can be conducted to ensure that “al gears mesh”, that is the internal operation of the product performs according to the specification and all internal components have been adequately exercised.

**5.4.1 UNIT TESTING:**

Unit testing is the testing of each module and the integration of the overall system is done. Unit testing becomes verification efforts on the smallest unit of software design in the module. This is also known as ‘module testing’.

The modules of the system are tested separately. This testing is carried out during the programming itself. In this testing step, each model is found to be working satisfactorily as regard to the expected output from the module. There are some validation checks for the fields. For example, the validation check is done for verifying the data given by the user where both format and validity of the data entered is included. It is very easy to find error and debug the system.

**5.4.2 INTEGRATION TESTING:**

Data can be lost across an interface, one module can have an adverse effect on the other sub function, when combined, may not produce the desired major function. Integrated testing is systematic testing that can be done with sample data. The need for the integrated test is to find the overall system performance. There are two types of integration testing. They are:

i) Top-down integration testing. ii) Bottom-up integration testing.

**5.4.3 TESTING TECHNIQUES/STRATEGIES:**

* **WHITE BOX TESTING:**

White Box testing is a test case design method that uses the control structure of the procedural design to drive cases. Using the white box testing methods, we

Derived test cases that guarantee that all independent paths within a module have been exercised at least once.

* **BLACK BOX TESTING:**

1. Black box testing is done to find incorrect or missing function
2. Interface error
3. Errors in external database access
4. Performance errors.
5. Initialization and termination errors

In ‘functional testing’, is performed to validate an application conforms to its specifications of correctly performs all its required functions. So this testing is also called ‘black box testing’. It tests the external behaviour of the system. Here the engineered product can be tested knowing the specified function that a product has been designed to perform, tests can be conducted to demonstrate that each function is fully operational.

**5.4.4 SOFTWARE TESTING STRATEGIES**

**VALIDATION TESTING:**

After the culmination of black box testing, software is completed assembly as a package, interfacing errors have been uncovered and corrected and final series of software validation tests begin validation testing can be defined as many,

But a single definition is that validation succeeds when the software functions in a manner that can be reasonably expected by the customer

**USER ACCEPTANCE TESTING:**

User acceptance of the system is the key factor for the success of the system. The system under consideration is tested for user acceptance by constantly keeping in touch with prospective system at the time of developing changes whenever required.

**OUTPUT TESTING**:

After performing the validation testing, the next step is output asking the user about the format required testing of the proposed system, since no system could be useful if it does not produce the required output in the specific format. The output displayed or generated by the system under consideration. Here the output format is considered in two ways. One is screen and the other is printed format. The output format on the screen is found to be correct as the format was designed in the system phase according to the user needs. For the hard copy also output comes out as the specified requirements by the user. Hence the output testing does not result in any connection in the system.

**CHAPTER 6**

**CONCLUSION**

This paper proposed an ML and DL-based model to forecast or to detect the weather effectively. This system was proposed for efficient weather prediction using different machine learning algorithm such as logistic regression and xgboost for forecast the weather. Experimental results analysis showed that our proposed method is efficient and can achieve better performance results on average when compared with existing system. Hence the proposed approach provides a perception of implementing a more generalized model for weather prediction.

**CHAPTER 7**

**FUTURE ENHANCEMENT**

In the future, we should like to hybrid the two different machine learning or to hybrid the two different deep learning algorithms. In future, it is possible to provide extensions or modifications to the proposed clustering and classification algorithms to achieve further increased performance. Apart from the experimented combination of data mining techniques, further combinations and other clustering algorithms can be used to improve the detection accuracy.

**CHAPTER 8**

**SAMPLE CODE**

#======================= IMPORT PACKAGES =============================

import pandas as pd

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

import warnings

warnings.filterwarnings('ignore')

from sklearn import preprocessing

from sklearn import metrics

import matplotlib.pyplot as plt

import seaborn as sns

#===================== 1. DATA SELECTION ==============================

#=== READ A DATASET ====

data\_frame=pd.read\_csv("weatherHistory.csv")

print("-------------------------------------------------------")

print("================== 1.Data Selection ===================")

print("-------------------------------------------------------")

print()

print(data\_frame.head(20))

#===================== 2.DATA PREPROCESSING ==========================

#=== CHECK MISSING VALUES ===

print("=====================================================")

print(" 2.Preprocessing ")

print("=====================================================")

print()

print("-------------------------------------------------------------")

print("================ Before Checking missing values =========")

print("-------------------------------------------------------------")

print()

print(data\_frame.isnull().sum())

print()

print("-------------------------------------------------------------")

print("================ After Checking missing values =========")

print("-------------------------------------------------------------")

print()

data\_frame=data\_frame.fillna(0)

print(data\_frame.isnull().sum())

data\_label=data\_frame['Precip Type']

#=== LABEL ENCODING ===

label\_encoder = preprocessing.LabelEncoder()

print("-------------------------------------------------------------")

print("==================== Before label encoding ==================")

print("------------------------------------------- ------------------")

print()

print(data\_frame['Summary'].head(15))

data\_frame['Summary']= label\_encoder.fit\_transform(data\_frame['Summary'].astype(str))

data\_frame['Precip Type']= label\_encoder.fit\_transform(data\_frame['Precip Type'].astype(str))

data\_frame['Daily Summary']= label\_encoder.fit\_transform(data\_frame['Daily Summary'].astype(str))

print("-------------------------------------------------------------")

print("==================== After label encoding ==================")

print("-------------------------------------------------------------")

print()

print(data\_frame['Summary'].head(15))

#=== DROP UNNECCESARY COLUMNS ===

data\_frame=data\_frame.drop('Formatted Date',axis=1)

#=============================== 3. DATA SPLITTING ============================

X=data\_frame.drop('Precip Type',axis=1)

y=data\_frame['Precip Type']

X\_train, X\_test,y\_train, y\_test = train\_test\_split(X, y, test\_size=0.4, random\_state=1)

print("==============================================")

print("---------------- Data Splitting --------------")

print("==============================================")

print()

print("Total No.of data's in dataset: ", data\_frame.shape[0])

print()

print("Total No.of training data's : ", X\_train.shape[0])

print()

print("Total No.of testing data's : ", X\_test.shape[0])

#============================ 5. CLASSIFICATION =============================

#==== RANDOM FOREST ====

from sklearn import linear\_model

lr = linear\_model.LogisticRegression()

lr.fit(X\_train, y\_train)

y\_pred = lr.predict(X\_test)

acc11=metrics.accuracy\_score(y\_pred,y\_test)\*100

cm\_lr=metrics.confusion\_matrix(y\_pred,y\_test)

print("----------------------------------------")

print("LOGISTIC REGRESSION --> LR")

print("------------------------------------")

print()

print("1. Accuracy =",acc11,'%' )

print()

print(metrics.classification\_report(y\_pred,y\_test))

# === CONFUSION MATRIX ===

sns.heatmap(cm\_lr, annot=True)

plt.title("Confusion matrix for LR")

plt.show()

# === XGBOOST ===

import xgboost as xgb

xgbb=xgb.XGBClassifier()

xgbb.fit(X\_train, y\_train)

y\_pred\_xg = xgbb.predict(X\_test)

acc\_xg=metrics.accuracy\_score(y\_pred\_xg,y\_test)\*100

cm\_xg=metrics.confusion\_matrix(y\_pred\_xg,y\_test)

print("----------------------------------------")

print("XGBOOST CLASSIFICATION --> XGB")

print("------------------------------------")

print()

print("1. Accuracy =",acc\_xg,'%' )

print()

print(metrics.classification\_report(y\_pred\_xg,y\_test))

# === CONFUSION MATRIX ===

sns.heatmap(cm\_xg, annot=True)

plt.title("Confusion matrix for Xgboost")

plt.show()

#============================ 6. PREDICTION =============================

#=== PREDICT THE WEATHER ===

print("-----------------------------------------------------------")

print("==================== Prediction ===========================")

print("-----------------------------------------------------------")

print()

for i in range(0,10):

if y\_pred[i]==0:

print("----------------------------")

print()

print([i],"None")

elif y\_pred[i]==1:

print("----------------------------")

print()

print([i],"Rain")

elif y\_pred[i]==2:

print("----------------------------")

print()

print([i],"Snow")

print()

print()

#============================ 7. COMAPRISON =============================

print("---------------------------------------------")

print("================= Comaprison =================")

print("----------------------------------------------")

print()

print()

vals=[acc11,acc\_xg]

inds=range(len(vals))

labels=["LR","XGBOOST"]

fig,ax = plt.subplots()

rects = ax.bar(inds, vals)

ax.set\_xticks([ind for ind in inds])

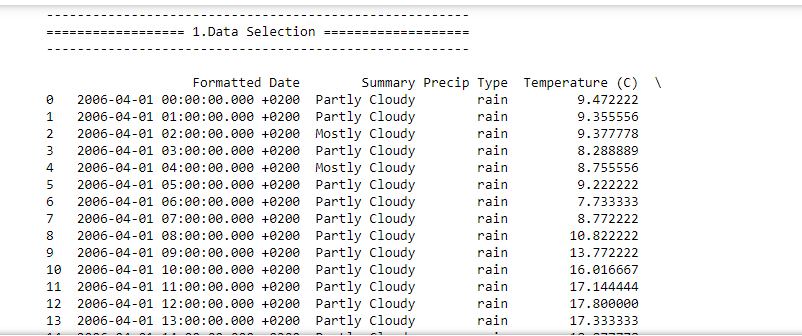
ax.set\_xticklabels(labels)

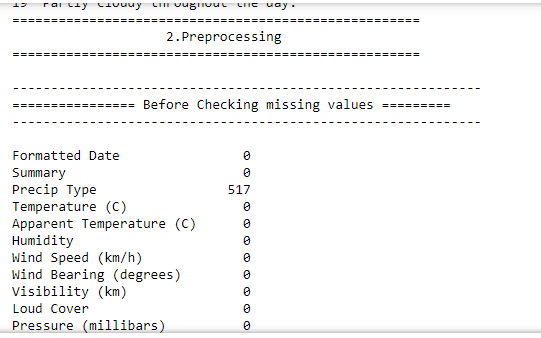
plt.savefig("Performance")

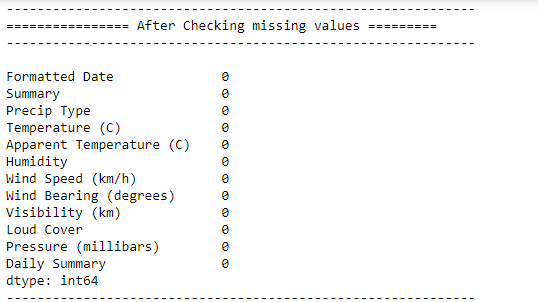
plt.show()

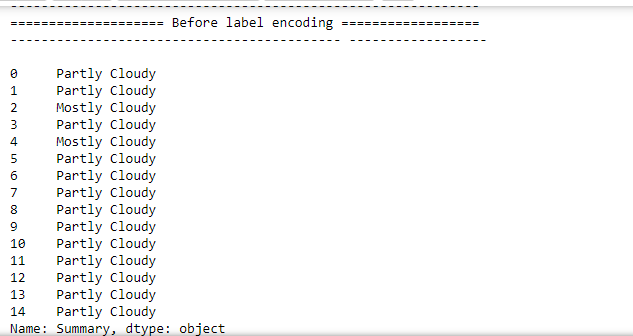
**CHAPTER 9**

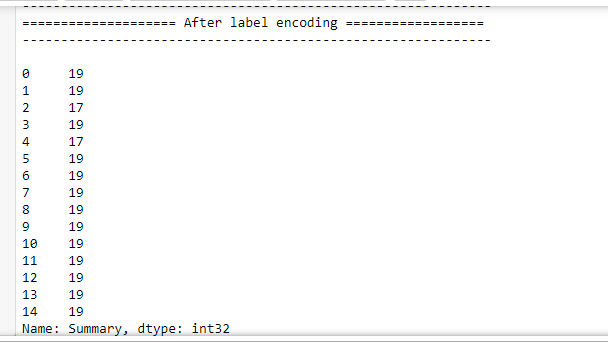
**SCREENSHOTS**

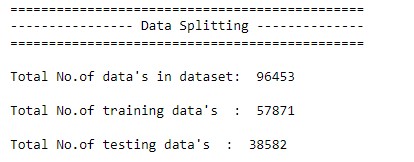


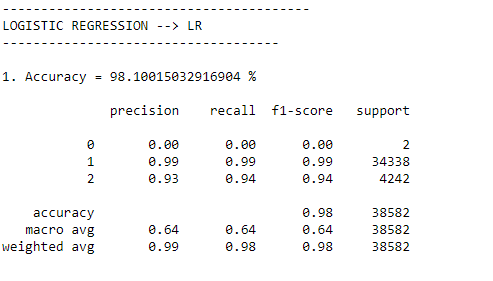


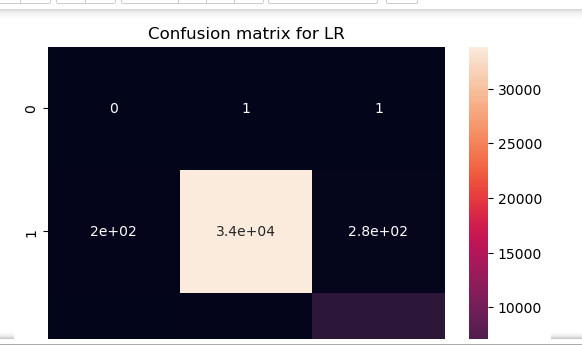


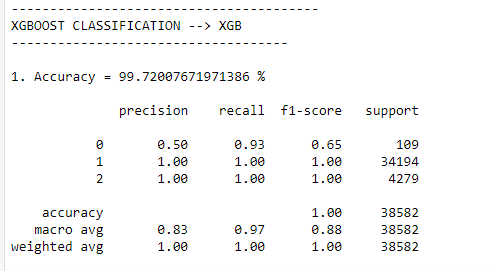


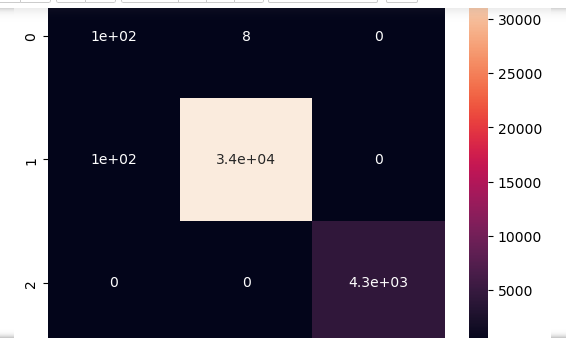


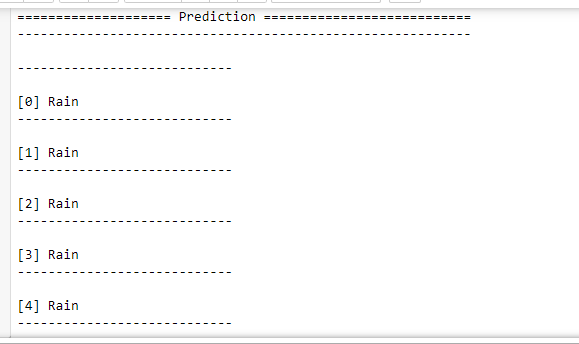












**CHAPTER 10**

**REFERENCES**

1. Imran Maqsood, Muhammad Riaz Khan, and Ajith Abraham. An ensemble of neural networks for weather forecasting, Neural Comput & Applic (2004) 13: 112–122.

2. Ball , R. , Tissot , P. (2006) Demonstration of Artificial Neural Network in Matlab

3. G. R. Gainieva, L. D. Nikitin, M. M. Naimark, N. N. Nazarov, and G. P. Tkachenko, Influence of Batch Composition and Clinkering Properties on the Hot Strength of Coke and Blast-Furnace Operation, ISSN 1068-364X, Coke and Chemistry, 2008, Vol. 51, No. 10, pp. 390–393. © Allerton Press, Inc., 2008

4. http://climate.weatheroffice.gc.ca/climateData/canada\_e.html .The website of the weather data of Canada

5. Haider, Adnan and Hanif, Nadeem, M., Inflation Forecasting In Pakistan Using Artificial Neural Networks

6. Matlab 7.6.0 (R2008a), The Math Works Inc., Product Help- Neural Network Toolbox

7. Dr. S. Santhosh Baboo and I.Kadar Shereef. An e\_cient weather forecasting system using Artificial neural network. International Journal of Environmental Science and Development, 1(4):321-326, 2010.

8. Amanpreet Kaur, J K Sharma, and Sunil Agrawal. Artificial neural networks in forecasting maximum and minimum relative humidity. International Journal of Computer Science and Network Security, 11(5):197-199, May 2011.

9. Sanjay Mathur, Avinash Kumar, and Mahesh Chandra. A feature based neural network model for weather forecasting, World Academy of Science, Engineering and Technology 34 2007.

10. Mohsen Hayati and Zahra Mohebi. Application of artificial neural networks for temperatureforecasting. World Academy of Science, Engineering and Technology, 28:275{279, 2007.

11. Rosmina Bustami, Nabil Bessaih, Charles Bong, and Suhaila Suhaili. Artificial neural network for precipitation and water level predictions of bedup river. IAENG International Journal of Computer Science, 34(2):228-233, 2007.]

12. Sergio Caltagirone. Air temperature prediction using evolutionary arti\_cial neural networks. Master's thesis, University of Portland College of Engineering, 5000 N. Willamette Blvd. Portland, OR 97207, 12 2001.

13. Surajit Chattopadhyay. Multilayered feed forward artificial neural network model to predict the average summer-monsoon rainfall in india. Master's thesis, Department of Mathematics, Techno Model SchoolTechno India Group, EM-4/1, Sector V, Salt Lake, Kolkata 700 091, India, 2006.

14. Tony Hall, Harold E. Brooks, and Charles A. Doswell III. Precipitation forecasting using a neural network, AMS Journal online, Volume 14, Issue 3 (June 1999).

15. N. Q. Hung, M. S. Babel, S. Weesakul, and N. K. Tripathi. An arti\_cial neural network model for rainfall forecasting in bangkok, Thailand. Hydrology and Earth System Sciences, 13(8):1413-1425, 2009.

16. Adrian Visoiu. Neural network based model refinement. Informatica Economica Journal,2008.