RAINFALL PREDICTION USING ML

Team Members:

Name	**Roll Number**
Lakshmi Sravya Savaram	Team Leader (TL)
Mohammad Shouqat Azeez	Team Member
N Gokul Chowdary	Team Member
Nallabotula Vijaya Karthik	Team Member

1. INTRODUCTION

1.1. MOTIVATION:

Rainfall prediction is helpful to avoid flood which save lives and properties of humans. Moreover, it helps in managing resources of water. Information of rainfall in prior helps farmers to manage their crops better which result in growth of country's economy. Fluctuation in rainfall timing and its quantity makes rainfall prediction a challenging task for meteorological scientists. In all the services provided by meteorological department, Weather forecasting stands out on top for all the countries across the globe. The task is very complex as it requires numbers of specialized and also all calls are made without any certainty.

Two widely used methods for rainfall forecasting are: Statistical methods and Numerical Weather Prediction (NWP) model. Nature of rainfall data is non-linear. Frequency, intensity and amount are main characteristics for time series rainfall. These values can be varied from one position on earth to other position of earth and from one time to other time. Every statistical model has some drawbacks. On a worldwide scale, large numbers of attempts

have been made by different researchers to predict rainfall accurately using various techniques. But due to the nonlinear nature of rainfall, prediction accuracy obtained by these techniques is still below the satisfactory level. Artificial neural network algorithm becomes an attractive inductive approach in rainfall prediction owing to their highly nonlinearity, flexibility and data driven learning in building models without any prior knowledge about catchment behavior and flow processes

1.2. PROBLEM DEFINITION:

Rainfall is one of the most complex and difficult elements of the hydrology cycle to understand and to model due to the complexity of the atmospheric processes that generate rainfall and the tremendous range of variation over a wide range of scales both in space and time. Heavy rainfall prediction is a major problem for meteorological department as it is closely associated with the economy and life of human. It is a cause for natural disasters like flood and drought which are encountered by people across the globe every year. Accuracy of rainfall forecasting has great importance for countries like India whose economy is largely dependent on agriculture. Due to dynamic nature of atmosphere, Statistical techniques fail to provide good accuracy for rainfall

forecasting. Thus, accurate rainfall prediction is one of the greatest challenges in operational hydrology. On a worldwide scale, large numbers of attempts have been made by different researchers to predict rainfall accurately using various techniques. But due to the nonlinear nature of rainfall, prediction accuracy obtained by these techniques is still below the satisfactory level.

1.3. PROJECT OBJECTIVE:

The main objective of this project is to analyze the parameters of various classification algorithms and compare their predictive accuracies so as to find out the best algorithm for predicting rainfall.This Project examines data from humidity,windgust speed,temperature etc..using them to try and predict the Rainfall. User can predict the Rainfall by entering parameters in the web application.

1.4. LIMITATIONS OF PROJECT:

- Prediction accuracy is very less.

- Finding the precipitation is particular to a geographic area.
- Works only for linear datasets and does not work for the non linear datasets.
- Works well only on small scale of datasets through which it was not able to predict the rainfall for larger dataset.

1.5. ORGANIZATION OF DOCUMENTATION:

Actually, there has been many theoretical projects and several experimental projects individually done based on rainfall prediction and many different algorithms have been developed for forecasting the Rainfall prediction. But Machine Learning (ML) is a type of artificial intelligence (AI) that allows software applications to become more accurate at predicting outcomes without being explicitly programmed to do so. Our aim from the project is to make use of NumPy or panda's libraries in Machine Learning and predict whether Rain will fall or not. Secondly, to learn how to visualize the data by using graphs. In the end, we are predicting the Rainfall based on the key list of temperature, humidity, wind gust speed…etc. The prediction is to be done using Machine Learning algorithms and withdrawing the conclusions.

2. PROBLEM STATEMENT

Artificial neural networks have been successfully used in these days in various aspects of science and engineering because of its ability to model both linear and non-linear systems without the need to make assumptions as are implicit in most traditional statistical approaches. ANN has been used as an effective model over the simple linear regression model. In information visualization the SVM is an amazing tool. Even though mathematical analysis of the algorithm is still alive, its fundamental implementation is simple and the map allegory easy to understand. Moreover, the results are believable and the algorithm scales extraordinarily well. The process of data visualizing and understanding in many studies and applications

SVM‘s have confirm to be an excellent. In machine learning, SVM is supervised learning models with associated learning algorithms that examine data and recognize patterns, used for classification and regression analysis. Specified set of training examples, individually marked as propitious to one classification, a SVM training algorithm frames a model that consigns new examples into one category or the other, manufacture it a non-probabilistic binary linear classifier. SVM implements a learning algorithm, useful for recognizing subtle patterns in complex data sets. Instead of minimizing the observed training error, Support Vector Regression (SVR) attempts to minimize the generalization error bound so as to achieve generalized performance. There are two main categories for support vector machines: support vector classification (SVC) and support vector regression (SVR). SVM is a learning system using a high dimensional feature space. It yields prediction functions that are expanded on a subset of support vectors. SVM can generalize complicated gray level structures with only a very few support vectors and thus provides a new mechanism for image compression. A version of a SVM for regression has been proposed in 1997 by Vapnik, Steven Golowich, and Alex Smola. This method is called support vector regression (SVR)the model produced by SVR only depends on a subset of the training data. Building the model ignores any training data that is close (within a threshold ε) to the model prediction. Support Vector Regression (SVR) is the most common application form of SVMs. Support vector machines project the data into a higher dimensional space and maximize the margins between classes or minimize the error margin for regression. Support Vector Machine is one of the important categories of multilayer feed forward network.

Like multi-layer perceptron and radial basis function networks, support vector machines can be used for pattern classification and regression. Support Vector Machines (SVMs) developed by Vapnik and his co-workers has been used for supervised learning due to –

(i) Better generalization performance than other NN models Rainfall Prediction using Machine Learning Techniques Dept. of ISE, NHCE Page 7

(ii) Solution of SVM is unique, optimal and absent from local minima as it uses linearly constrained quadratic programming problem

(iii) Applicability to non-vectorial data (Strings and Graphs) and

(iv) Few parameters are required for tuning the learning m/c. Kernel Methods are a set of algorithms from statistical learning which include the SVM for classification and regression, Kernel PCA, Kernel based clustering, feature selection, and dimensionality reduction etc . SVM is found to be a significant technique to solve many classifications problem in the last couple of years. Very few researchers of this field used this technique for rainfall prediction and got satisfactory result. Support vector regression model is used to predict numerical output. The idea of SVR is based on the computation of a linear regression function in a high dimensional feature space where the input data are mapped via a nonlinear function.

3. LITERATURE SURVEY

3.1. INTRODUCTION:

Meteorologists at NOAA's National Weather Service have always monitored the conditions of the atmosphere that impact the weather, but over time the equipment they use have changed. As technology advanced, our scientists began to use more efficient equipment to collect and use additional data. These technological advances enable our meteorologists to make better predictions faster than ever before. Doppler Radar is the meteorologist's window into observing severe storms. With 159 radars towers across the United States, NOAA's National Weather Service has comprehensive coverage of the continental U.S. and partial coverage of Alaska, Hawaii, Puerto Rico and Guam. Doppler radar detects all types of precipitation, the rotation of thunderstorm clouds, airborne tornado debris, and wind strength and direction. Weather Satellites monitor Earth from space, collecting observational data our scientists analyse. NOAA operates three types of weather satellites. Polar orbiting satellites orbit the Earth close to the surface, taking six or seven detailed images a day. Geostationary satellites stay over the same location on Earth high above the surface taking images of the entire Earth as frequently as every 30 seconds. Deep space satellites face the sun to monitor powerful solar storms and space weather. NOAA also uses data from satellites operated by other agencies and countries. Radiosondes are our primary source of upper-air data. At least twice per day, radiosondes are tied to weather balloons and are launched in 92 locations across the United States. In its two hour trip, the radiosonde floats to the upper stratosphere where it collects and sends back data every second about air

pressure, temperature, relative humidity, wind speed and wind direction. During severe weather, we usually launch weather balloons more frequently to collect additional data about the storm environment

NOAA's Weather and Climate Operational Supercomputer System (WCOSS) is the backbone of modern forecasting. With 5.78 petaflop computing capacity it can process quadrillions of calculations per second. Our supercomputers are almost 6 million times more powerful than your average desktop computer. Observational data collected by doppler radar, radiosondes, weather satellites, buoys and other instruments are fed into computerized NWS numerical forecast models. The models use equations, along with new and past weather data, to provide forecast guidance to our meteorologists. Observational data collected by doppler radar, radiosondes, weather satellites, buoys and other instruments are fed into computerized NWS (Numerical Weather Prediction) numerical forecast models. The models use equations, along with new and past weather data, to provide forecast guidance to our meteorologists.Two widely used methods for rainfall forecasting are: Statistical methods and Numerical Weather Prediction (NWP)model. The statistical approaches do not have the ability to identify nonlinear patterns and irregular trend in the time series

3.2. ARCHITECTURE BASED LITERATURE SURVEY:

For rainfall prediction, the "divide – and – conquer" approach was used to divide the whole region into four areas and the problem was solved separately based on the following assumptions: first, rainfall has a different pattern in different areas; second, rainfall pattern within smaller areas is continuous and smooth; third, or graphic effect exists. Predictions of the two large areas using RBF networks were reasonably good. But, for the two small areas, predictions were not good since the or graphic effect was not apparent. The proposed method does not require pre-modelling or any other tools. Since the authors had used the same number of basic functions as observed data points, data clustering was not necessary. The weight values from the hidden layer were easily found using the linear matrix inversion technique. The proposed method is suitable for emergency conditions, as well as long term management of contaminated regions. In the paper by Ko Koizumi, An Objective Method to Modify Numerical Model Forecasts with Newly Given Weather Data Using an Artificial Neural Network, an artificial neural

3.3. DISADVANTAGES OF EXISTING SYSTEM:

- Collecting large amount of data set.
- Large number of training data and annotations are needed which may not be practical in some problems.

3.4. PROPOSED SYSTEM:

Rainfall prediction has become an uncertain event which has a major impact for agricultural industry and society. SVR and ANN are the algorithms being used in this paper for more accurate predictions of rainfall. Machine Learning learn and improve their learning time in autonomous fashion, by feeding them data and information. In our project, we predict the rainfall in a scale between 0 to 1. 0 means there is no rainfall and 1 means heavy rainfall. We have plotted a graph for historical rainfall data.

1. Linear Regression:

Linear Regression is a supervised machine learning algorithm where the predicted output is continuous and has a constant slope. It's used to predict values within a continuous range, (e.g. sales, price) rather than trying to classify them into categories (e.g. cat, dog).

Steps to implement Linear regression model:

1. Initialize the parameters.
2. Predict the value of a dependent variable by given an independent variable.
3. Calculate the error in prediction for all data points.
4. Calculate partial derivative w.r.t a0 and a1.
5. Calculate the cost for each number and add them.

2.Random Forest:

A random forest is a machine learning technique that’s used to solve regression ais a supervised ensemble learning algorithm. Ensemble means that it takes a bunch of weak learners and have them work together to form one strong predictor. Here, we have a collection of decision trees, known as Forest. To classify a new object based on attributes, each tree gives a classification and we say the tree votes for that class. The forest chooses

the classification having the most votes (over all the trees in the forest). Our Model's configuration: number of weak learners = 100, maximum depth of each tree = 4.

3.Logistic regression:

- Logistic regression is a classification algorithm used to predict a binary outcome (1 / 0, Yes / No, True / False) given a set of independent variables. To represent binary / categorical outcome, we use dummy variables. We can also think of logistic regression as a special case of linear regression when the outcome variable is categorical, where we are using log of odds as dependent variable. In simple words, it predicts the probability of occurrence of an event by fitting data to a logit function.

  Hence, this makes Logistic Regression a better fit as ours is a binary classification problem.

4.k-nearest neighbor algorithm:

- Is a non-parametric and lazy learning algorithm. Non-parametric means there is no assumption for underlying data distribution. In other words, the model structure is determined from the dataset. Lazy algorithm means it does not need any training data points for model generation. All training data used in the testing phase. KNN performs better with a lower number of features than a large number of features. We can say that when the number of features increases than it requires more data. Increase in dimension also leads to the problem of overfitting. However, we have performed feature selection which helps to reduce dimension and hence KNN looks a good candidate for our problem. Our Model's configuration: We tried various values of n ranging from 3 to 30 and learned that the model performs best with n as 25, 27 and 29.

4. EXPERIMENTAL ANALYSIS

Milestone 1: Data Collection

ML depends heavily on data, without data, a machine can't learn. It is the most crucial aspect that makes algorithm training possible. In Machine Learning projects, we need a training data set. It is the actual data set used to train the model for performing various actions.

You can collect datasets from different open sources like kaggle.com, data.gov; UCI machine learning repository etc. The dataset used for this project was obtained from Kaggle. **Milestone 2: Data Pre-processing**

Data Pre-processing includes the following main tasks

- Importing the libraries.
- Importing the dataset.
- Analyses the data.
- Taking care of Missing Data.
- Data Visualization. • Splitting Data into Train and Test.

Milestone 3: Model Building

The model building process involves setting up ways of collecting data, understanding and paying attention to what is important in the data to answer the questions you are asking, finding a statistical, mathematical or a simulation model to gain understanding and make predictions. Model Building Includes:

- Import the model building libraries.
- Initializing the model.
- Training the model.
- Model Evaluation.
- Save the Model.

Milestone 4: Application Building

- Create an HTML File.
- Build python code.
- Run the app in local browser.
- Show casting the prediction on UI.

4.1. PROJECT ARCHITECTURE:

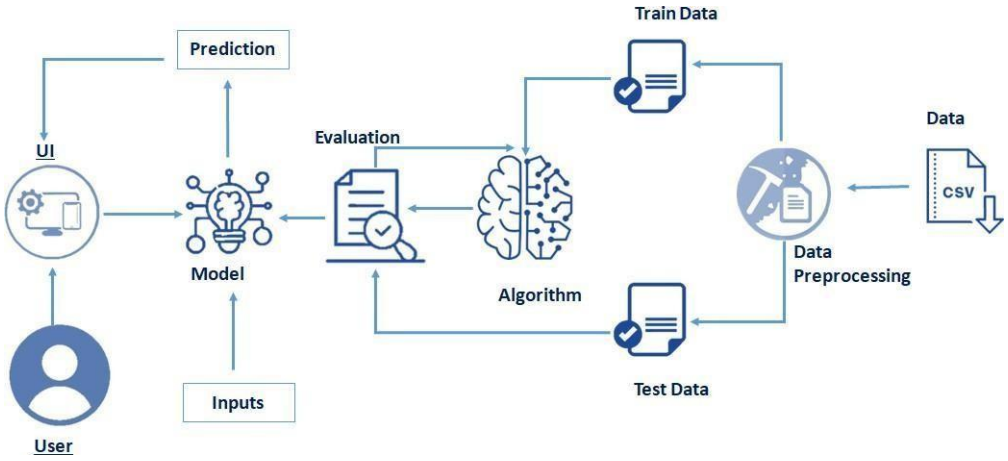


Figure 1: Project Architecture

4.2. SOFTWARE AND HARDWARE REQUIREMENTS:

Software Requirements:

- Anaconda Environment
- Flask
- Python 3.9
- And other python libraries like NumPy, pandas.

Hardware Requirements:

- Operating system
- Processing
- RAM
- Operating system specifications
- Disk space

4.3. BLOCK DIAGRAM:

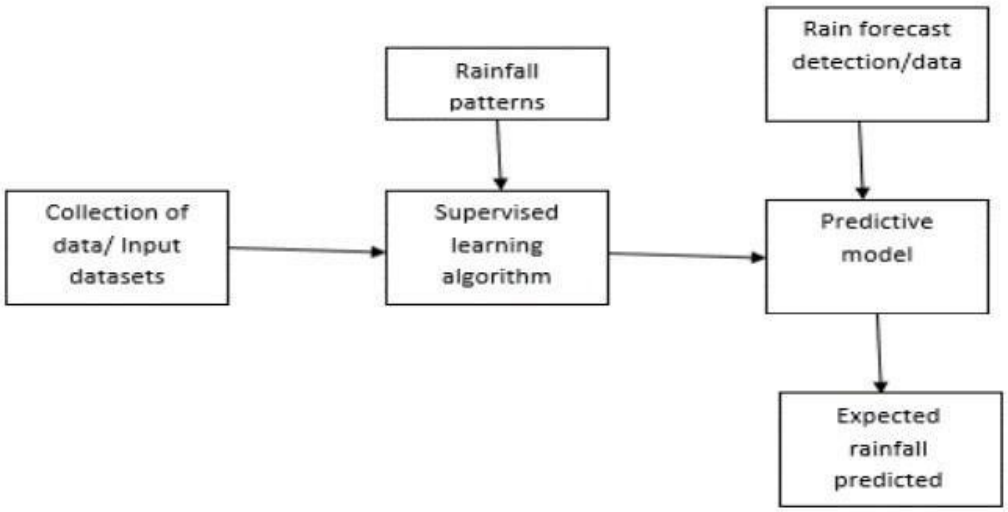


Figure 2 : Block Diagram

4.4. PROJECTFLOW:

- User interacts with the UI (User Interface) to upload the input features.
- Uploaded features/input is analysed by the model which is integrated.
- Once a model analyses the uploaded inputs, the prediction is showcased on the UI.

5. DESIGN

5.1. USE CASEDIAGRAM:

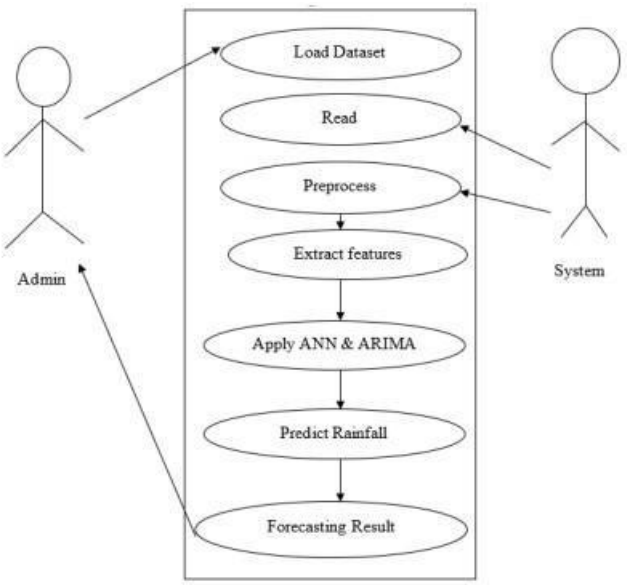


Figure 3: Use case Diagram

5.2. FLOWCHART:

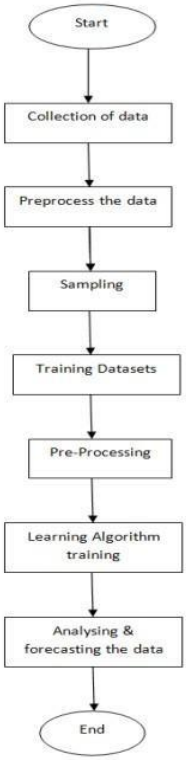


Figure 4: FlowChart

5.3. ACTIVITY DIAGRAM:

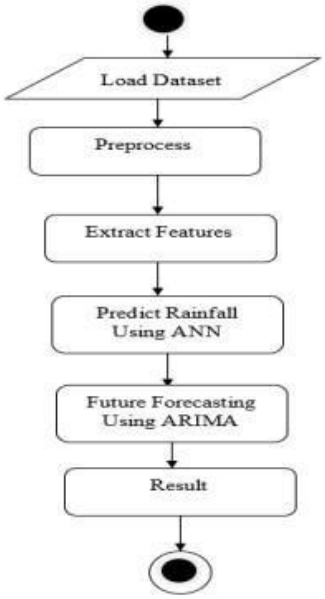


Figure 5: Activity Tree

5.3. SEQUENCE DIAGRAM:

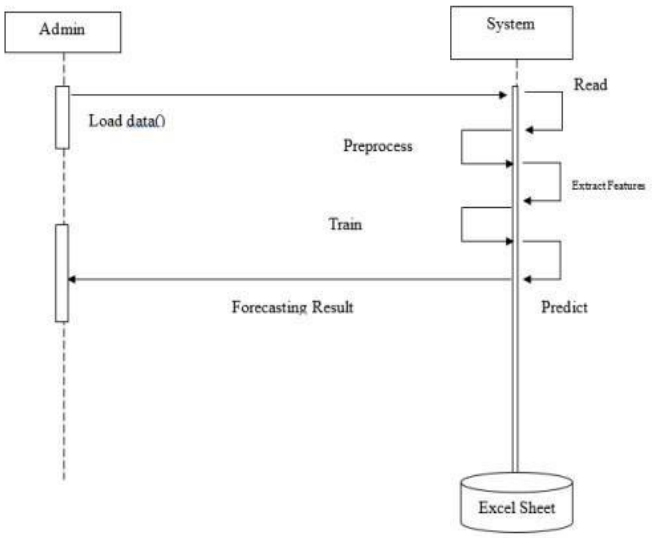


Figure 6: Sequence Diagram

5.4. CLASS DIAGRAM:

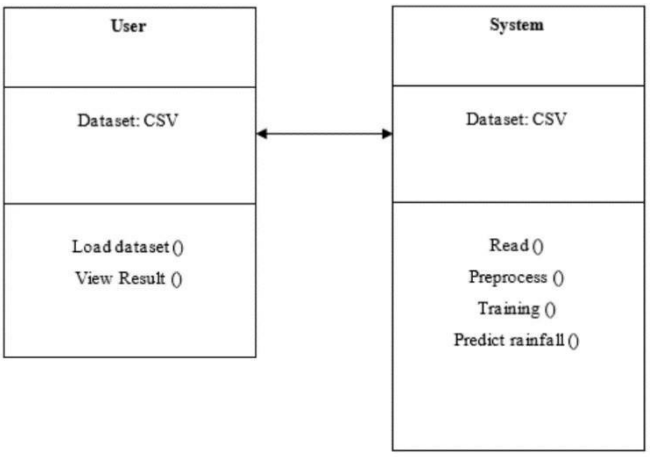


Figure 7: Class Diagram

6. CONCLUSION

In UG Project Phase-1, we have worked on problem statement, literature survey and also done the experimental analysis which are required for the project to move forward. In experimental analysis we have discussed about the machine learning concepts and models and explained the algorithms to be used in the project. We also discussed about the flowcharts, use case diagrams, decision tree and sequence diagrams which are used in the project. Based on the experimental analysis we have designed the model for the project. Entire designing part is involved in UG Project Phase-1.

CODE SNIPPETS

2.1.MODELCODE:

Data Preprocessing

Data Pre-processing includes the following main tasks

1.Import the Libraries.

2.Importing the dataset.

3.Checking for Null Values.

4.Data Visualization.

5.Label Encoding.

6.OneHot Encoding.

7.Splitting Data into Train and Test.

```
# libraries required
import pandas as pd
import numpy as np
import seaborn as sns
import matplotlib.pyplot as plt
from sklearn import preprocessing
from sklearn import model_selection
from sklearn import metrics
from sklearn import linear_model
from sklearn import ensemble
from sklearn import tree
from sklearn import svm
import xgboost
```

```
data = pd.read_csv("Dataset - Dataset.csv")
```

```
data.head()
```

Date	Location	MinTemp	MaxTemp	Rainfall	Evaporation	Sunshine	WindGustDir	WindGustSpeed	WindDir9am	...	Humidity9am	Humidity3pm	Pressure9am	Pressu
2008-12-01	Delhi	13.4	22.9	0.6	NaN	NaN	W	44.0	W	...	71.0	22.0	1007.7	
2008-12-02	Delhi	7.4	25.1	0.0	NaN	NaN	WNW	44.0	NNW	...	44.0	25.0	1010.6	
2008-12-03	Delhi	12.9	25.7	0.0	NaN	NaN	WSW	46.0	W	...	38.0	30.0	1007.6	
2008-12-04	Delhi	9.2	28.0	0.0	NaN	NaN	NE	24.0	SE	...	45.0	16.0	1017.6	
2008-12-05	Delhi	17.5	32.3	1.0	NaN	NaN	W	41.0	ENE	...	82.0	33.0	1010.8	

ws × 23 columns

Figure 1: .ipynb code describing importing libraries and displaying the few rows

from the Dataset.

```
data.describe()
```

	MinTemp	MaxTemp	Rainfall	Evaporation	Sunshine	WindGustSpeed	WindSpeed9am	WindSpeed3pm	Humidity9am	Humidity3pm	Pressure
count	32057.000000	32186.000000	31871.000000	13492.000000	9040.000000	27704.000000	31728.000000	31082.000000	31891.000000	31226.000000	25872.000
mean	13.205565	24.013211	2.631979	5.629314	7.642710	37.303783	12.029595	16.537095	70.612054	52.321783	1018.244
std	5.808879	6.015886	9.356417	4.573872	3.758171	13.334088	8.983175	8.935069	17.616726	20.671318	6.590
min	-4.800000	6.800000	0.000000	0.000000	0.000000	7.000000	0.000000	0.000000	3.000000	1.000000	980.500
25%	8.900000	19.500000	0.000000	3.000000	5.100000	28.000000	6.000000	9.000000	59.000000	37.000000	1013.900
50%	13.900000	23.400000	0.000000	4.600000	8.750000	35.000000	11.000000	15.000000	71.000000	53.000000	1018.300
75%	17.800000	27.700000	0.600000	7.200000	10.500000	44.000000	19.000000	22.000000	84.000000	67.000000	1022.700
max	29.700000	47.300000	371.000000	86.200000	14.000000	135.000000	130.000000	83.000000	100.000000	100.000000	1039.900

```
data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 32555 entries, 0 to 32554
Data columns (total 23 columns):
 #   Column         Non-Null Count  Dtype
---  ------         --------------  -----
 0   Date           32555 non-null  object
 1   Location       32555 non-null  object
 2   MinTemp        32057 non-null  float64
 3   MaxTemp        32186 non-null  float64
 4   Rainfall       31871 non-null  float64
 5   Evaporation    13492 non-null  float64
 6   Sunshine       9040 non-null   float64
 7   WindGustDir    27700 non-null  object
 8   WindGustSpeed  27704 non-null  float64
 9   WindDir9am     28089 non-null  object
 10  WindDir3pm     30457 non-null  object
 11  WindSpeed9am   31728 non-null  float64
 12  WindSpeed3pm   31082 non-null  float64
 13  Humidity9am    31891 non-null  float64
 14  Humidity3pm    31226 non-null  float64
 15  Pressure9am    25872 non-null  float64
 16  Pressure3pm    25875 non-null  float64
 17  Cloud9am       16745 non-null  float64
 18  Cloud3pm       16415 non-null  float64
 19  Temp9am        32119 non-null  float64
 20  Temp3pm        31450 non-null  float64
 21  RainToday      31870 non-null  object
 22  RainTomorrow   31870 non-null  object
dtypes: float64(16), object(7)
memory usage: 5.7+ MB
```

Figure 2: .ipynb code describing describe( ) and info( ) methods .

```
data.isnull().mean() #Evaporation    Sunshine        Cloud9am          Cloud3pm
```

```
Date             0.000000
Location         0.000000
MinTemp          0.015297
MaxTemp          0.011335
Rainfall         0.021011
Evaporation      0.585563
Sunshine         0.722316
WindGustDir      0.149132
WindGustSpeed    0.149009
WindDir9am       0.137183
WindDir3pm       0.064445
WindSpeed9am     0.025403
WindSpeed3pm     0.045247
Humidity9am      0.020396
Humidity3pm      0.040823
Pressure9am      0.205283
Pressure3pm      0.205191
Cloud9am         0.485640
Cloud3pm         0.495776
Temp9am          0.013393
Temp3pm          0.033943
RainToday        0.021041
RainTomorrow     0.021041
dtype: float64
```

Figure 3: .ipynb code describing whether they are any NULL values in the Dataset.

```
# removing columns with more than 20% missing values and segregatting cat and num variables
data_cat = data[['RainToday','WindGustDir','WindDir9am','WindDir3pm']]
data.drop(columns=['Evaporation','Sunshine','Cloud9am','Cloud3pm'],axis=1,inplace=True)
data.drop(columns=['RainToday','WindGustDir','WindDir9am','WindDir3pm'],axis=1,inplace=True)
```

```
# filling the missing data of numeric variables with mean
data['MinTemp'].fillna(data['MinTemp'].mean(),inplace=True)
data['MaxTemp'].fillna(data['MaxTemp'].mean(),inplace=True)
data['Rainfall'].fillna(data['Rainfall'].mean(),inplace=True)
data['WindGustSpeed'].fillna(data['WindGustSpeed'].mean(),inplace=True)
data['WindSpeed9am'].fillna(data['WindSpeed9am'].mean(),inplace=True)
data['WindSpeed3pm'].fillna(data['WindSpeed3pm'].mean(),inplace=True)
data['Humidity9am'].fillna(data['Humidity9am'].mean(),inplace=True)
data['Humidity3pm'].fillna(data['Humidity3pm'].mean(),inplace=True)
data['Pressure9am'].fillna(data['Pressure9am'].mean(),inplace=True)
data['Pressure3pm'].fillna(data['Pressure3pm'].mean(),inplace=True)
data['Temp9am'].fillna(data['Temp9am'].mean(),inplace=True)
data['Temp3pm'].fillna(data['Temp3pm'].mean(),inplace=True)
```

```
# loading the names of categorical columns
cat_names = data_cat.columns
```

```
# intializing the simple imputer for missing categorical values
import numpy as np
from sklearn.impute import SimpleImputer
imp_mode = SimpleImputer(missing_values=np.nan, strategy='most_frequent')
```

```
# fitting and transforming the missing data
data_cat = imp_mode.fit_transform(data_cat)
```

Figure 4 : .ipynb code

d
e
s
c
r
i
b

```
# converting array to dataframe
data_cat = pd.DataFrame(data_cat,columns=cat_names)
```

```
# concatinating the categorical and numeric data
data = pd.concat([data,data_cat],axis=1)
```

```
data.shape
```

```
(32555, 19)
```

```
data.isnull().sum()
```

```
Date              0
Location          0
MinTemp           0
MaxTemp           0
Rainfall          0
WindGustSpeed     0
WindSpeed9am      0
WindSpeed3pm      0
Humidity9am       0
Humidity3pm       0
Pressure9am       0
Pressure3pm       0
Temp9am           0
Temp3pm           0
RainTomorrow    685
RainToday         0
WindGustDir       0
WindDir9am        0
WindDir3pm        0
dtype: int64
```

ingfillingofNullValues.

```
#importing the labelencoder
from sklearn.preprocessing import LabelEncoder
```

```
le = LabelEncoder()
```

```
#fitting and transforming the categorical data
data['Location']=le.fit_transform(data['Location'])
data['WindGustDir']=le.fit_transform(data['WindGustDir'])
data['WindDir3pm']=le.fit_transform(data['WindDir3pm'])
data['WindDir9am']=le.fit_transform(data['WindDir9am'])
data['RainToday']=le.fit_transform(data['RainToday'])
data['RainTomorrow']=le.fit_transform(data['RainTomorrow'])
```

```
# spliiting the date column into year,month,day
data[["year", "month", "day"]] = data["Date"].str.split("-", expand = True)
```

```
# removing the main column
data.drop(['Date'],axis=1,inplace=True)
```

```
data.head()
```

Location	MinTemp	MaxTemp	Rainfall	WindGustSpeed	WindSpeed9am	WindSpeed3pm	Humidity9am	Humidity3pm	Pressure9am	...	Temp9am	Temp3pm	RainTomorrow	RainToday	WindGustDir	WindDir9a
4	13.4	22.9	0.6	44.0	20.0	24.0	71.0	22.0	1007.7	...	16.9	21.8	0	0	13	1
4	7.4	25.1	0.0	44.0	4.0	22.0	44.0	25.0	1010.6	...	17.2	24.3	0	0	14	
4	12.9	25.7	0.0	46.0	19.0	26.0	38.0	30.0	1007.6	...	21.0	23.2	0	0	15	1
4	9.2	28.0	0.0	24.0	11.0	9.0	45.0	16.0	1017.6	...	18.1	26.5	0	0	4	
4	17.5	32.3	1.0	41.0	7.0	20.0	82.0	33.0	1010.8	...	17.8	29.7	0	0	13	

rs × 21 columns

```
data.dropna(axis=0,how='any',inplace=True)
```

```
data.shape
```

```
(32555, 21)
```

```
from sklearn.preprocessing import StandardScaler
```

```
sc = StandardScaler()
```

```
y = data['RainTomorrow']
x = data.drop('RainTomorrow',axis=1)
```

```
names = x.columns
```

```
names
```

```
Index(['Location', 'MinTemp', 'MaxTemp', 'Rainfall', 'WindGustSpeed',
       'WindSpeed9am', 'WindSpeed3pm', 'Humidity9am', 'Humidity3pm',
       'Pressure9am', 'Pressure3pm', 'Temp9am', 'Temp3pm', 'RainToday',
       'WindGustDir', 'WindDir9am', 'WindDir3pm', 'year', 'month', 'day'],
      dtype='object')
```

```
x = sc.fit_transform(x)
```

```
x = pd.DataFrame(x,columns=names)
```

Figure 5:Feature scaling,standardizing the data

Model Building

-Predictive modeling is a mathematical approach to create a statistical model to forecast future behavior based on input test data.

Model building includes the following main tasks

Training and testing the model, Evaluation of Model, Save the model, Predicting the output using the model.

```
from sklearn import model_selection
```

```
x_train,x_test,y_train,y_test = model_selection.train_test_split(x,y,test_size =0.2,random_state =0)
```

```
import sklearn
```

```
#Models intilization of the models
XGBoost = xgboost.XGBRFClassifier()
Rand_forest = sklearn.ensemble.RandomForestClassifier()
svm = sklearn.svm.SVC()
Dtree = sklearn.tree.DecisionTreeClassifier()
GBM = sklearn.ensemble.GradientBoostingClassifier()
log = sklearn.linear_model.LogisticRegression()
```

```
# fitting the model
XGBoost.fit(x_train,y_train)
Rand_forest.fit(x_train,y_train)

Dtree.fit(x_train,y_train)
GBM.fit(x_train,y_train)
log.fit(x_train,y_train)
```

```
▾ LogisticRegression
LogisticRegression()
```

```
svm.fit(x_train,y_train)
```

```
▾ SVC
SVC()
```

```
# predicting the unknown values
p1 = XGBoost.predict(x_train)
p2 = Rand_forest.predict(x_train)
p3 = svm.predict(x_train)
p4 = Dtree.predict(x_train)
p5 = GBM.predict(x_train)
p6 = log.predict(x_train)
```

```
#checking the accuraccy score
print("xgboost:",metrics.accuracy_score(y_train,p1))
print("Rand_forest:",metrics.accuracy_score(y_train,p2))
print("svm:",metrics.accuracy_score(y_train,p3))
print("Dtree:",metrics.accuracy_score(y_train,p4))
print("GBM:",metrics.accuracy_score(y_train,p5))
print("log:",metrics.accuracy_score(y_train,p6))
```

```
xgboost: 0.8425741053601598
Rand_forest: 0.9999616034403317
svm: 0.8446475195822454
Dtree: 1.0
GBM: 0.8475656581170327
log: 0.8240669636000615
```

```
# predicting the test unknown values
t1 = XGBoost.predict(x_test)
t2 = Rand_forest.predict(x_test)
t3 = svm.predict(x_test)
t4 = Dtree.predict(x_test)
t5 = GBM.predict(x_test)
t6 = log.predict(x_test)
```

```
print("xgboost:",metrics.accuracy_score(y_test,t1))
print("Rand_forest:",metrics.accuracy_score(y_test,t2))
print("svm:",metrics.accuracy_score(y_test,t3))
print("Dtree:",metrics.accuracy_score(y_test,t4))
print("GBM:",metrics.accuracy_score(y_test,t5))
print("log:",metrics.accuracy_score(y_test,t6))
```

```
xgboost: 0.8298264475502994
Rand_forest: 0.8375057594839502
svm: 0.8313623099370296
Dtree: 0.7574873291353095
GBM: 0.8371985870066042
log: 0.8204576869912455
```

Figure 6 : .ipynb code describing finding Accuracy .

Identifying the Best parameters combination using Gridsearchcv

```
[ ]  model = ensemble.RandomForestClassifier()
```

```
[ ]  parameters = {"max_features":["log2","sqrt"],
                   'n_estimators': [5,10,15,20,25],
                    'criterion':['gini','entropy']}
```

```
[ ]  Grid_search = sklearn.model_selection.GridSearchCV(estimator=model,param_grid=parameters,cv=10)
```

```
[ ]  Grid_search.fit(x_train,y_train)
```

GridSearchCV
estimator: RandomForestClassifier
RandomForestClassifier

```
[ ]  Grid_search.best_params_  # best combination,which gives better accuracy
```

```
{'criterion': 'gini', 'max_features': 'log2', 'n_estimators': 25}
```

Hyper-parameter tuned Model

```
[ ]  model = ensemble.RandomForestClassifier(criterion='entropy',max_features='sqrt',n_estimators = 25)
```

```
[ ]  model.fit(x_train,y_train)
     y_tp= model.predict(x_train)
     sklearn.metrics.accuracy_score(y_train,y_tp)
```

```
0.9981185685762556
```

```
[ ]  y_pred= model.predict(x_test)
     sklearn.metrics.accuracy_score(y_test,y_pred)
```

```
0.8309015512210106
```

```
[ ]  metrics.confusion_matrix(y_test,y_pred)
```

```
array([[4761,  237,    8],
       [ 747,  604,    6],
       [  81,   22,   45]])
```

```
import pickle
```

```
pickle.dump(model,open('rainfall.pkl','wb'))
pickle.dump(le,open('encoder.pkl','wb'))
pickle.dump(imp_mode,open('impter.pkl','wb'))
pickle.dump(sc,open('scale.pkl','wb'))
```

2.2 HTML CODE AND PYTHON CODE

i.app.py code:

```
import numpy as np
import pickle
import joblib
import matplotlib
import matplotlib.pyplot as pit
import time
import pandas
import os
from flask import Flask, request, jsonify, render_template

app = Flask(__name__)
model = pickle.load(open('/content/rainfall.pkl', 'rb' ))
scale = pickle.load(open('/content/scale.pkl', 'rb'))

@app.route('/')# route to display the home page
def home():
    return render_template('/content/11.index.html') #rendering the home page

@app.route('/predict',methods=["POST","GET"]) #route to show the predictions in a web UI
def predict():
#reading the inputs given by the user
    input_feature=[x for x in request.form.values() ]
    features_values=[np.array(input_feature)]
    names = [['Location', 'MinTemp', 'MaxTemp', 'Rainfall', 'WindGustSpeed',
       'WindSpeed9am', 'WindSpeed3pm', 'Humidity9am', 'Humidity3pm',
       'Pressure9am', 'Pressure3pm', 'Temp9am', 'Temp3pm', 'RainToday',
       'WindGustDir', 'WindDir9am', 'WindDir3pm', 'year', 'month', 'day']]
    data = pandas.DataFrame(features_values,columns=names)
    data = scale.fit_transform(data)
    data = pandas.DataFrame(data,columns = names)
# predictions using the loaded model file
    prediction=model.predict(data)
    pred_prob = model.predict_proba(data)
    print(prediction)
    if prediction == "Yes":
         return render_template("/content/11.chance.html")
    else:
         return render_template("/content/11.nochance.html")
         #showing the prediction results in a uI
if __name__=='__main__':
  app.run(debug=True)
```

```
 * Serving Flask app '__main__'
 * Debug mode: on
INFO:werkzeug:WARNING: This is a development server. Do not use it in a production deployment. Use a production WSGI server instead.
 * Running on http://127.0.0.1:5000
INFO:werkzeug:Press CTRL+C to quit
INFO:werkzeug: * Restarting with stat
```

Figure 7 : .python code used for rendering all the HTML pages.

ii.index.html:

```
<!DOCTYPE html>
<html >
<head>
  <meta charset="UTF-8">
  <title>Rainfall prediction</title>
</head>


<body background="https://wallpaperaccess.com/full/701614.jpg" text="black">

 <div class="login">
  <center><h1>Rainfall Prediction</h1></center>

    <!-- Main Input For Receiving Query to our ML -->
    <form action="{{ url_for('predict')}}"method="post">
<h1>Please enter the following details</h1>


</style></head>

      <label for="Location">Location:</label>
        <select id="Location" name="Location">
            <option value=2>Albury</option>
            <option value=4>BadgerysCreek</option>
            <option value=10>Cobar</option>
            <option value=11>CoffsHarbour</option>
            <option value=21>Moree</option>
            <option value=24>Newcastle</option>
            <option value=26>NorahHead</option>
            <option value=27>NorfolkIsland</option>
            <option value=30>Penrith</option>
            <option value=34>Richmond</option>
            <option value=37>Sydney</option>
            <option value=38>SydneyAirport</option>
            <option value=42>WaggaWagga</option>
            <option value=45>Williamtown</option>
            <option value=47>Wollongong</option>
            <option value=9>Canberra</option>
            <option value=40>Tuggeranong</option>
            <option value=23>MountGinini</option>
            <option value=5>Ballarat</option>
            <option value=6>Bendigo</option>
            <option value=35>Sale</option>
            <option value=19>MelbourneAirport</option>
            <option value=18>Melbourne</option>
            <option value=20>Mildura</option>
            <option value=25>Nhil</option>
            <option value=33>Portland</option>
            <option value=44>Watsonia</option>
            <option value=12>Dartmoor</option>
            <option value=7>Brisbane</option>
            <option value=8>Cairns</option>
            <option value=14>GoldCoast</option>
            <option value=39>Townsville</option>
            <option value=0>Adelaide</option>
            <option value=22>MountGambier</option>
            <option value=28>Nuriootpa</option>
            <option value=48>Woomera</option>
            <option value=1>Albany</option>
            <option value=46>Witchcliffe</option>
            <option value=29>PearceRAAF</option>
            <option value=32>PerthAirport</option>
            <option value=31>Perth</option>
            <option value=36>SalmonGums</option>
            <option value=43>Walpole</option>
            <option value=15>Hobart</option>
            <option value=17>Launceston</option>
            <option value=3>AliceSprings</option>
            <option value=13>Darwin</option>
            <option value=16>Katherine</option>
            <option value=41>Uluru</option>
        </select>   

      <input type="number"   name="MinTemp" placeholder="MinTemp" required="required" />
      <input type="number"   name="MaxTemp" placeholder="MaxTemp" required="required" />
      <input type="number"   name="Rainfall" placeholder="Rainfall" required="required" />
      <input type="number"   name="WindGustSpeed" placeholder="WindGustSpeed" required="required" /><br>
 <br>
      <input type="number"   name="WindSpeed9am" placeholder="WindSpeed9am" required="required" />
      <input type="number"   name="WindSpeed3pm" placeholder="WindSpeed3pm" required="required" />
      <input type="number"   name="Humidity9am" placeholder="Humidity9am" required="required" />
      <input type="number"   name="Humidity3pm" placeholder="Humidity3pm" required="required" />
      <input type="number"   name="Pressure9am" placeholder="Pressure9am" required="required" /><br>
 <br>
```

```
    <input type="number"   name="Pressure3pm" placeholder="Pressure3pm" required="required" />
    <input type="number"   name="Temp9am" placeholder="Temp9am" required="required" />
    <input type="number"   name="Temp3pm" placeholder="Temp3pm" required="required" />
    <input type="number" min="2008" max="2021"  name="year" placeholder="year" required="year" /><br>
<br>
    <input type="number" min="0" max="12" name="month" placeholder="month" required="required" />
    <input type="number"  min= "1" max= "31" name="day" placeholder="day" required="required" /><br>
<br>


    <label for="RainToday">RainToday:</label>
      <select id="RainToday" name="RainToday">
          <option value=0>No</option>
          <option value=1>Yes</option>
      </select>   

    <label for="WindGustDir">WindGustDir :</label>
      <select id="WindGustDir" name="WindGustDir">
          <option value=13>W<option>
          <option value=14>WNW<option>
              <option value=15>WSW<option>
              <option value=4>NE<option>
              <option value=6>NNW<option>
              <option value=3>N<option>
              <option value=5>NNE<option>
              <option value=12>SW<option>
              <option value=1>ENE<option>
              <option value=10>SSE<option>
              <option value=8>S<option>
              <option value=7>NW<option>
              <option value=9>SE<option>
              <option value=2>ESE<option>
              <option value=0>E<option>
              <option value=11>SSW<option>
      </select>

    <label for="WindDir9am">WindDir9am :</label>
      <select id="WindDir9am" name="WindDir9am">
          <option value=13>W<option>
          <option value=6>WNW<option>
              <option value=9>WSW<option>
              <option value=1>NE<option>
              <option value=12>NNW<option>
              <option value=10>N<option>
              <option value=8>NNE<option>
              <option value=4>SW<option>
              <option value=3>ENE<option>
              <option value=11>SSE<option>
              <option value=15>S<option>
              <option value=2>NW<option>
              <option value=0>SE<option>
              <option value=7>ESE<option>
              <option value=15>E<option>
              <option value=5>SSW<option>
      </select>   

    <label for="WindDir3pm">WindDir3pm :</label>
      <select id="WindDir3pm" name="WindDir3pm">
          <option value=14>W<option>
          <option value=15>WNW<option>
              <option value=0>WSW<option>
              <option value=7>NE<option>
              <option value=13>NNW<option>
              <option value=10>N<option>
              <option value=2>NNE<option>
              <option value=1>SW<option>
              <option value=6>ENE<option>
              <option value=11>SSE<option>
              <option value=12>S<option>
              <option value=9>NW<option>
              <option value=3>SE<option>
              <option value=8>ESE<option>
              <option value=5>E<option>
              <option value=4>SSW<option>
      </select>   


<br>

<br><br>


<button type="submit" class="btn btn-primary btn-block btn-large" style="height:30px;width:200px">Predict</button>
```

```
    </form>
<br>


 {{ prediction_text }}


 <br>
 <br>
 <img src="{{url_3}}" alt="Submit Form" height="180" width="233" onerror="this.style.display='none'"/>
 <img src="{{url_1}}" alt="Submit Form" height="180" width="233" onerror="this.style.display='none'"/>
 <img src="{{url_4}}" alt="Submit Form" height="180" width="233" onerror="this.style.display='none'"/>
 <br>
 <br>
 <img src="{{url_2}}" alt="Submit Form" height="150" width="711" onerror="this.style.display='none'"/>

</div>


</body>
</html>
```

Figure 8: index.html is the page which displays all the inputs that are needed to be given by the user.

iii.Chance.html:

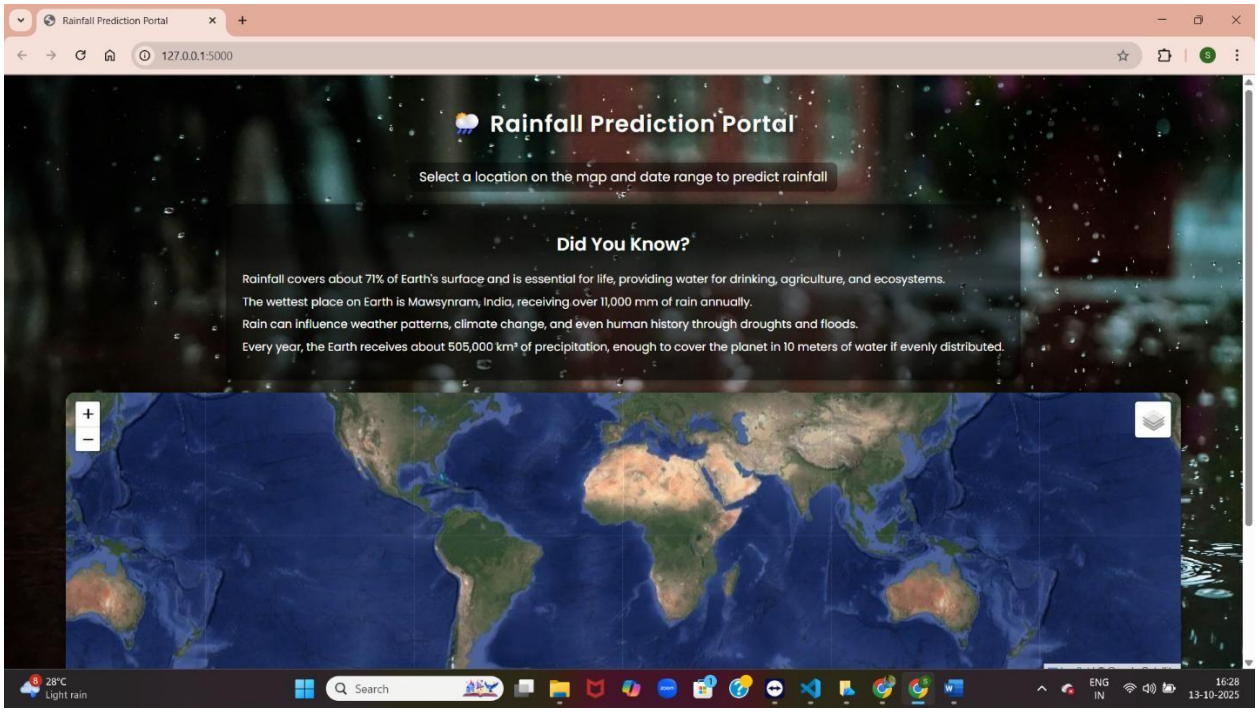
```
<!DOCTYPE html>
<html >
<head>
  <meta charset="UTF-8">
  <title>Rainfall prediction</title>
</head>


<body background="https://wnavprd.blob.core.windows.net/images/guide/chris-seufert-cape-cod-rain-beach-1400-110-1.jpg" text="black">

 <div class="login">
  <center><h1>chances of rain today.</h1></center>
</body>
</html>
```

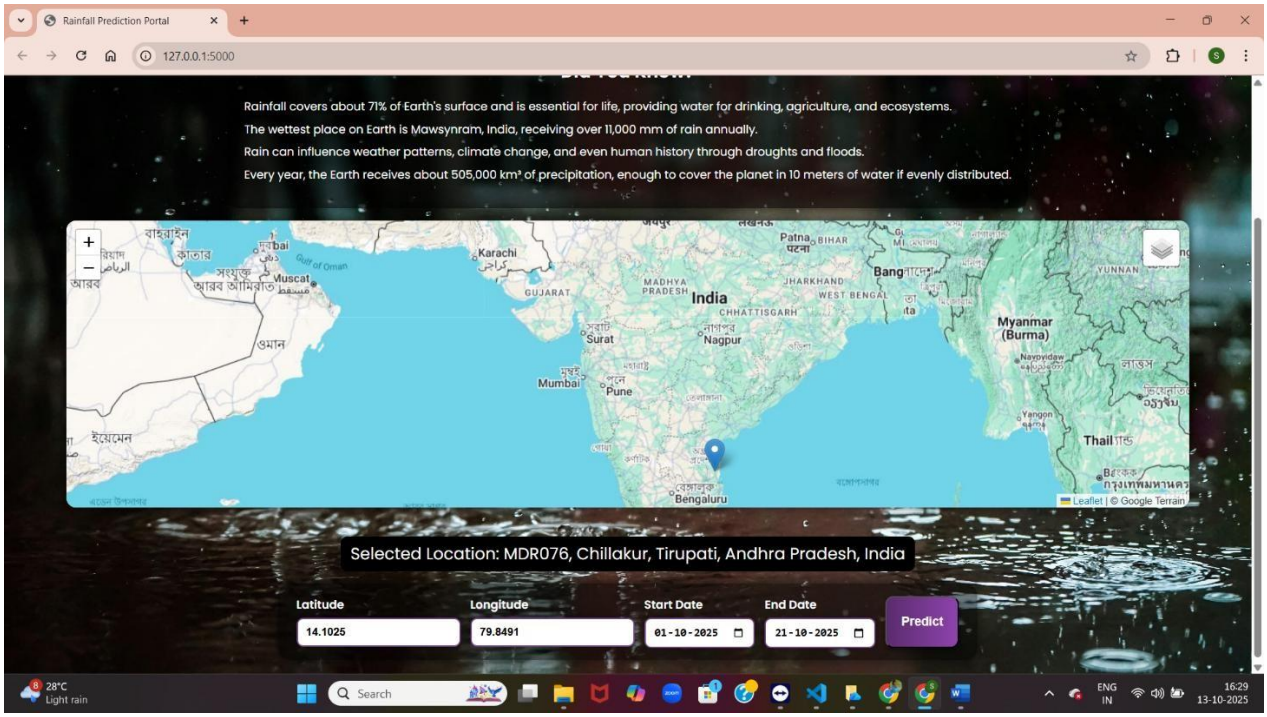
Figure 9: chance.html is the page which displays that there is a Rainfall

```
<!DOCTYPE html>
<html >
<head>
  <meta charset="UTF-8">
  <title>Rain fall Prediction</title>
</head>


<body background="https://cdn.tourradar.com/s3/tour/1500x800/139566_d97baf8c.jpg" text="black">

 <div class="login">
  <center><h1>No chances of occurance of rain.</h1></center>
</body>
</html>
```

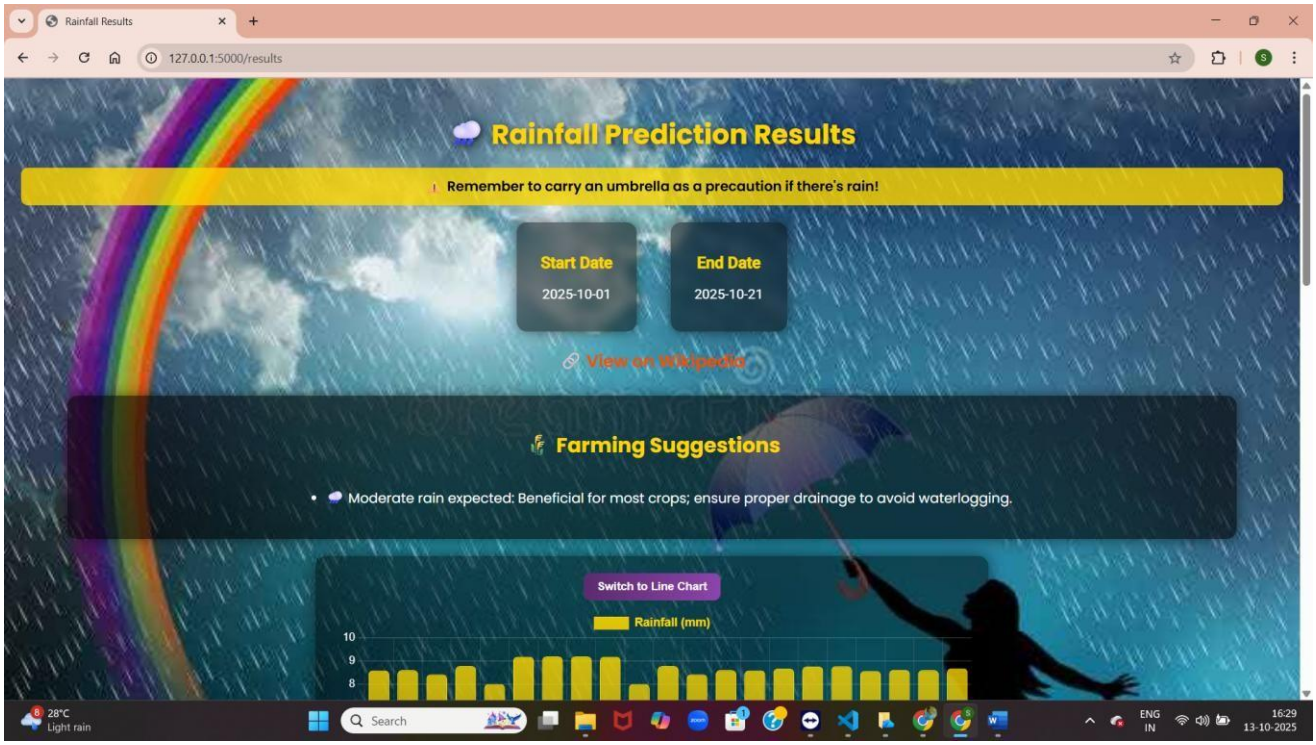
3.CONCLUSION

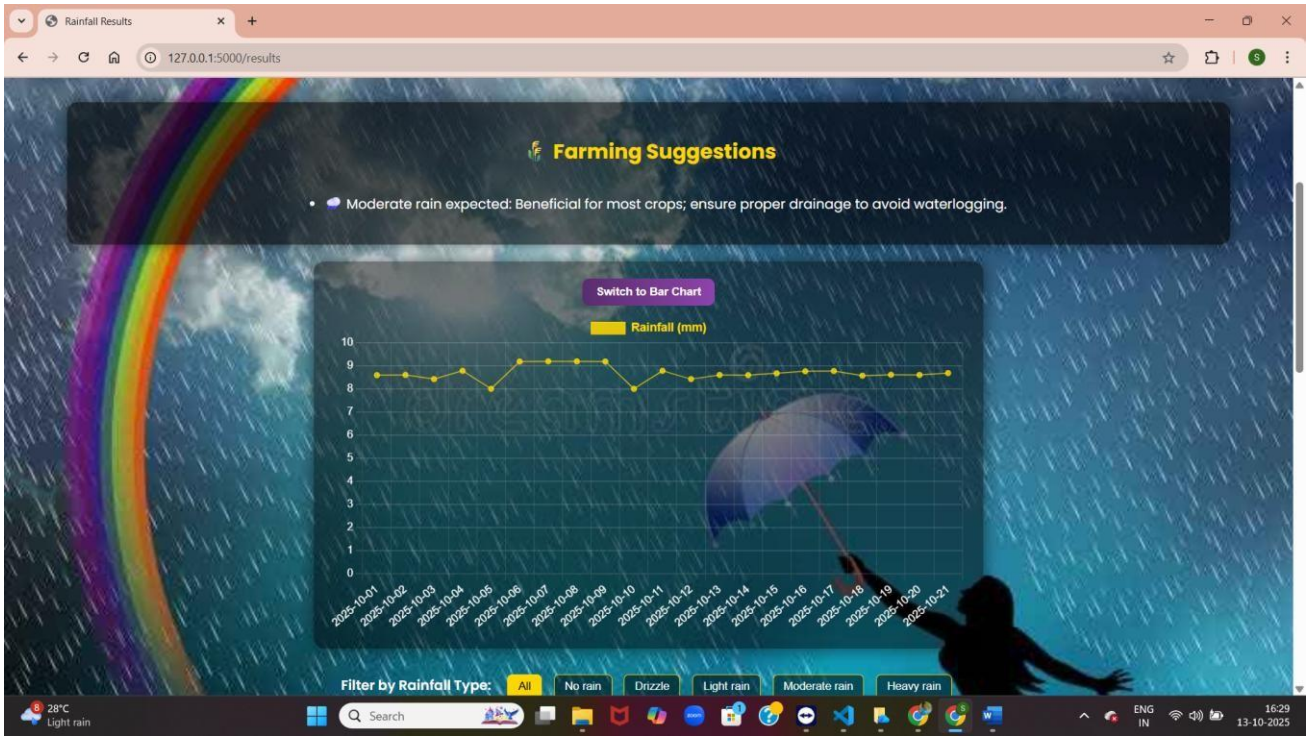
INDEX PAGE OUTPUT:





RESULT PAGE OUTPUT:




Rainfall Results
127.0.0.1:5000/results
Farming Suggestions
Moderate rain expected: Beneficial for most crops; ensure proper drainage to avoid waterlogging.
Switch to Bar Chart
Rainfall (mm)
Filter by Rainfall Type:
All
No rain
Drizzle
Light rain
Moderate rain
Heavy rain
28°C
Light rain
Search
ENG
IN
16:29
13-10-2025


Rainfall Results
127.0.0.1:5000/results
Filter by Rainfall Type:
All
No rain
Drizzle
Light rain
Moderate rain
Heavy rain
Date
Predicted Rainfall (mm)
Rainfall Type
Weather
2025-10-01 8.59 Moderate rain cloudy
2025-10-02 8.6 Moderate rain cloudy
2025-10-03 8.42 Moderate rain cloudy
2025-10-04 8.78 Moderate rain cloudy
2025-10-05 8.0 Moderate rain cloudy
2025-10-06 9.18 Moderate rain cloudy
2025-10-07 9.19 Moderate rain cloudy
2025-10-08 9.19 Moderate rain cloudy
2025-10-09 9.18 Moderate rain cloudy
2025-10-10 8.0 Moderate rain cloudy
2025-10-11 8.78 Moderate rain cloudy
2025-10-12 8.42 Moderate rain cloudy
28°C
Light rain
Search
ENG
IN
16:29
13-10-2025

4.APPLICATIONS

The areas where this solution can be applied:

=> Can be applied in each and every individual's Daily Life to take precautions.

=>Weather report on mobiles and systems.

=> Can be used in Weather report for faster prediction of Rain.

5.ADVANTAGES

1. Water resources can be managed efficiently by using rainfall prediction system.
2. Regions can be evacuated if flood are expected.
3. It helps in taking appropriate measures to efficiently manage water resources, crop productivity and no wastage of any resources.

6.DISADVANTAGES

1.Classification ,Clustering ,Decision Tree and collection of dataset are difficult.

2.Rainfall prediction is not hundred percent accurate some times it cannot predict exacty and sometimes weather may change.

7.FUTURE SCOPE

The future work of the project would be the improvement of architecture for light and other weather scenarios. Also, can develop a model for small changes in climate in future. An algorithm for testing daily basis dataset instead of accumulated dataset could be of paramount Importance for further research.

8.BIBILOGRAPHY

[1] Kumar Abhishek. Abhay Kumar, Rajeev Ranjan, Sarthak Kumar,\" A Rainfall Prediction Model using Artificial Neural Network\", 2012 IEEE Control and System Graduate Research Colloquium (ICSGRC2012), pp. 82-87, 2012.

[2] G. Geetha and R. S. Selvaraj, “Prediction of monthly rainfall in Chennai using Back

Propagation Neural Network model,” Int. J. of Eng. Sci. and Technology, vol. 3, no. 1, pp. 211 213, 2011.

[3] Zahoor Jan, Muhammad Abrar, Shariq Bashir and Anwar M Mirza, \"Seasonal to interannual climate prediction using data mining KNN technique\", International MultiTopic Conference, pp. 40-51, 2008.

[4] Elia Georgiana Petre, \"A decision tree for weather prediction\", Seria Matematica - Informatica] – Fizic, no. 1, pp. 77-82, 2009.

[5] Gupta D, Ghose U. A Comparative Study of Classification Algorithms for Forecasting Rainfall. IEEE. 2015.

[6] Wang J, Su X. An improved K-Means clustering algorithm. IEEE. 2014.

[7] Rajeevan, M., Pai, D. S., Anil Kumar, R. & Lal, B. New statistical models for long-range forecasting of southwest monsoon rainfall over India. Clim. Dyn. 28, 813–828 (2007).

[8] Mishra, V., Smoliak, B. V., Lettenmaier, D. P. & Wallace, J. M. A prominent pattern of year-to-year variability in Indian Summer Monsoon Rainfall. Proc. Natl Acad. Sci. USA 109, 7213–7217 (2012). [9]

9.HELP FILE

PROJECT EXECUTION:

- **STEP-1:** Go to **Start,** search and launch **ANACONDA NAVIGATOR**
- **STEP-2:** After launching of **ANACONDA NAVIGATOR,** launch **JUPYTER NOTEBOOK.**
- **STEP-3:** Open **"Major project code" IPYNB file.**
- **STEP-4:** Then run all the cells.
- **STEP-5:** All the **data preprocessing**, **training and testing**, **model building**, **accuracy** of the model can be showcased.
- **STEP-6:** And a pickle file will be generated.
- **STEP-7:** Create a Folder named **FLASK** on the **DESKTOP.** Extract the pickle file into this Flask Folder.
- **STEP-8:** Extract all the html files (home.html, index.html, chance.html, nochance.html) and python file(app.py) into the **FLASK Folder.**
- **STEP-9:** Then go back to **ANACONDA NAVIGATOR** and the launch the **SPYDER.**
- **STEP-10:** After launching Spyder, give the path of **FLASK FOLDER** which you have created on the DESKTOP.
- **STEP-11:** Open all the app.py and html files present in the Flask Folder.
- **STEP-12:** After running of the app.py, open **ANACONDA PROMPT** and follow the below steps:
- cd File Path click enter python app.py click enter (We could see running of files).
- **STEP-13:** Then open **BROWSER,** at the URL area type **—localhost:5000".**
- **STEP-14:** Home page of the project will be displayed.
- **STEP-15:** Click on **—Go to Predict".** Directly it will be navigated to index page.
- **STEP-16:**A index page will be displayed where the user needs to give the inputs and then click on **—Predict"**. Output will be generated whether rain will fall or not