Machine Learning Techniques For Predicting Over Rainfall

A PROJECT REPORT

Submitted as a Jth Component for the course

MCA

by

Shashank Kumar(20MCA0142)

Krati Jain(20MCA0115)

Prachi Jha(20MCA0143)

Under the Guidance of

Prof. Chellatamilan T.

VIT®

Vellore Institute of Technology

(Deemed to be University under section 3 of UGC Act, 1956)

School of Information Technology & Engineering

June,2021

DECLARATION BY THE CANDIDATE

I hereby declare that the project report entitled **"Machine learning techniques for predicting over rainfall"** submitted by me to VIT University, Vellore in partial fulfilment of the requirement for the award of the degree of **MCA** is a record of Jth component of project work carried out by me under the guidance of **Prof.Chellatamilan T**. I further declare that the work reported in this project has not been submitted and will not be submitted, either in part or in full, for the award of any other degree or diploma in this institute or any other institute or university.

Place: Vellore

Signature of the Candidate

Date: 5 June 2021

Name of the Student

Krati Jain (20MCA0115)

Shashank Kumar (20MCA0142)

Prachi Jha(20MCA0143)

TABLE OF CONTENTS

CHAPTER 1

Introduction

Rainfall prediction is one of the challenging and uncertain tasks which has a significant impact on human society. Timely and accurate predictions can help to proactively reduce human and financial loss. This study presents a set of experiments which involve the use of prevalent machine learning techniques to build models to predict whether it is going to rain tomorrow or not based on weather data for that particular day in major cities of Australia. This comparative study is conducted concentrating on three aspects: modeling inputs, modeling methods, and pre-processing techniques. The results provide a comparison of various evaluation metrics of these machine learning techniques and their reliability to predict the rainfall by analyzing the weather data.The objectives to offer end to cease system studying existence cycle proper from Data pre-processing to implementing models to comparing them. Data Pre-processing steps consist of imputing missing values, characteristic transformation, encoding specific features, characteristic scaling and feature selection. We carried out fashions which include CatBoostClassifier, RandomForestClassifier, Logistic Regression, K Nearest Neighbour, Rule-primarily based totally and Ensembles. For assessment purpose, used Accuracy, Precision, Recall, F-Score and Area Under Curve as evaluation metrics. For our experiments, we educate our classifiers the usage of Australian weather data accrued from numerous climate stations in Australia.

1.1. Significance of Study

Rainfall prediction is significant not only on the micro but also on the macro level. The study is of significance with respect to its vital contribution in the field of agriculture, water reserve management, flood prediction and management with an intention to ease the people by keeping them updated with the weather and rainfall prediction. It is also important to be utilized by the agricultural industries for keeping their crops safe and ensure the production of seasonal fruits and vegetables by updated rainfall prediction. The study will also be significant for the flood management authorities as more precise and accurate prediction for heavy monsoon rains will keep the authorities alert and focused for an upcoming event that of which the destruction could be minimized by taking precautionary measures. The rainfall prediction will impressively help in dealing with the increasing issue of water resource management; as water is a scarce resource and it needs to get saved for the benefit of human beings themselves.

CHAPTER 2

Literature Survey and Review

Rainfall prediction is not an easy job especially when expecting the accurate and precise digits for predicting the rain. The rainfall prediction is commonly used to protect the agriculture and production of seasonal fruits and vegetables and to sustain their production and quality in relation to the amount of rain required by them (Lima & Guedes, 2015).The rainfall prediction uses several networks and algorithms and obtains the data to be given to the agriculture and production departments. The rainfall prediction is necessary and mandatory especially in the areas where there is heavy rainfall and it's more often expected (Amoo & Dzwairo, 2016). There are huge economies like those of Asia like India and China that that earn a large proportion of their revenue from agriculture and for these economies; rainfall prediction is actually very important (Darji, Dabhi, & Prajapati, 2015). The rainfall forecasting is prevailing as a popular research in the scientific areas in the modern world of technology and innovation; as it has a huge impact on just the human life but the economies and the living beings as a whole. Rainfall prediction with several Logistic Regressionbeen analyzed previously and the researchers are still trying hard to achieve the more perfect and accurate results in the field of rainfall prediction (Biswas, et al., 2016). The prediction of seasonal rainfall on monthly basis by using the surface data to form annual prediction is also essential for the agricultural activities and therefore the production and supervision of the agriculture and crops. It could be done by recognizing the variations in the supply of moisture in the air. The case of African region illustrates that how this succeeded and how West Africa advantaged from the rainfall prediction in managing their agricultural activities (Omotosho, Balogun, & Ogunjobi, 2000).

[1] Nikhil Oswal**,"** Predicting Rainfall using Machine Learning Techniques",2019.In this In this paper, we explored and applied several pre-processing steps and learned there impact on the overall performance of our classifiers. We also carried a comparative study of all the classifiers with different input data and observed how the input data can affect the model predictions.

[2] Aakash Parmar, Kinjal Mitshree, Mithila Sompura**,"** Machine Learning Techniques For Rainfall Prediction: A Review",2017. Due to dynamic nature of atmosphere, Statistical techniques fail to provide good accuracy for rainfall forecasting. Nonlinearity of rainfall data makes Artificial Neural Network a better technique. Review work and comparison of different approaches and algorithms used by researchers for rainfall prediction is shown in a tabular form. Intention of this paper is to give non-experts easy access to the techniques and approaches used in the field of rainfall prediction.

[3] A. Naik and S. Pathan ,used the ANN model for rainfall prediction. They modified back propagation algorithm which was more robust than the simple back propagation algorithm

[4] Jyothis Joseph, Ratheesh T ," Rainfall Prediction using Data Mining Techniques",2013. n. In this paper two methods such as classification and clustering are implemented. The neural network Bayesian regularization has been applied in the implementation.

CHAPTER 3

System Design

In this, the overall system design includes four major components:

- Data Exploration and Analysis, Data Pre-processing, Model Implementation, and Model Evaluation

3.1 Data Exploration and Analysis

Exploratory Data Analysis is valuable to machine learning problems since it allows to get closer to the certainty that the future results will be valid, correctly interpreted, and applicable to the desired business contexts [4]. Such level of certainty can be achieved only after raw data is validated and checked for anomalies, ensuring that the data set was collected without errors.

3.2 Data Pre-Processing

Data pre-processing is a data mining technique that involves transforming raw data into an understandable format. Real-world data is often incomplete, inconsistent, and/or lacking in certain behaviours or trends, and is likely to contain many errors. We have carried below pre-processing steps:

3.2.1 Missing Values: It becomes one of the important step. To impute the missing values, we will group our instances based on the location and date and thereby replace the null values by there respective mean values.

3.2.2 Feature Expansion: Date feature can be expanded to Day, Month and Year and then these newly created features can be further used for other pre-processing Step.

3.2.3 Categorical Values: Categorical feature is one that has two or more categories but there is no intrinsic ordering to the categories. We have a few categorical features - WindGustDir, WindDir9am, WindDir3pm with 16 unique values. Now it gets complicated for machines to understand texts and process them, rather than numbers, since the models are based on mathematical equations and calculations.

3.2.4 Feature Scaling: Our data set contains features with highly varying magnitudes and range. But since, most of the machine learning algorithms use Euclidean distance

between two data point in their computations, this is a problem. The features with high magnitudes will weigh in a lot more in the distance calculations than features with low magnitudes. To suppress this effect, we need to bring all features to the same level of magnitudes. This can be achieved by
scaling.

3.2.5 Feature Selection- Feature Selection is the process where you automatically or manually select those features which contribute most to our prediction variable or output. Having irrelevant features in data can decrease the accuracy of the models and make the model learn based on irrelevant features. Feature selection helps to reduce over fitting, improves accuracy and reduces training time

3.3 Models

The following classification algorithms have been used to build prediction models to perform the experiments:

3.3.1 Logistic Regression – It 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.

3.3.2 K- Nearest Neighbour- 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.

3.3.3 Random Forest – It is 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.

3.3.4 CatBoost – It is an algorithm for gradient boosting on decision trees. It is developed by Yandex researchers and engineers, and is used for search, recommendation systems, personal assistant, self-driving cars, weather prediction and many other tasks at Yandex and in other companies, including CERN, Cloudflare, Careem taxi.

3.4 Evaluation

3.4.1 Accuracy- It is the ratio of number of correct predictions to the total number of input samples. It works well only if there are equal number of samples belonging to each class.

3.4.2 F1 Score -It is the Harmonic Mean between precision and recall. The range of F1 Score is [0, 1]. It tells how precise our classifier is (how many instances it classifies correctly), as well as how robust it is (it does not miss a significant number of instances). High precision but lower recall, gives you an extremely accurate, but it then misses a large number of instances that are difficult to classify. The greater the F1 Score, the better is the performance of our model.

3.4.3 Confusion Matrix- It gives us a matrix as output and describes the complete performance of the model. It focuses on True Positives - the cases in which we predicted YES and the actual output was also YES; True Negatives - the cases in which we predicted NO and the actual output was NO; False Positives - the cases in which we predicted YES and the actual output was NO; False Negatives - the cases in which we predicted NO and the actual output was YES

3.5 Experiments and Results

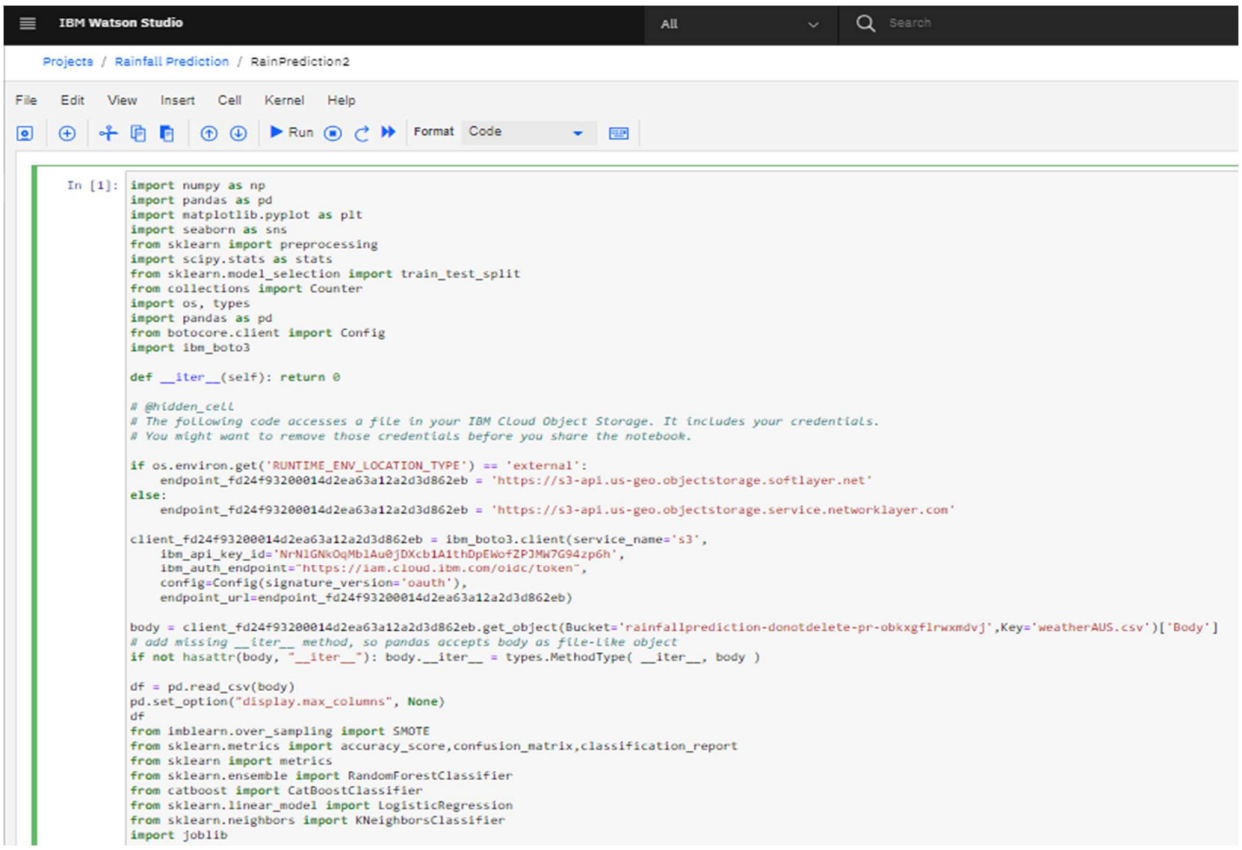
For all the experiments and development of classifiers, we used Python 3 and Jupyter Notebook. We used libraries such as Scikit Learn, Mat-plotlib, Seaborn, Pandas, Numpy and Imblearn. We carried experiments with different input data; one with the original dataset, then with the undersampled dataset and last one with the oversampled dataset. We splitted out dataset in ratio of 75:25 for training and testing purpose.

CHAPTER 4

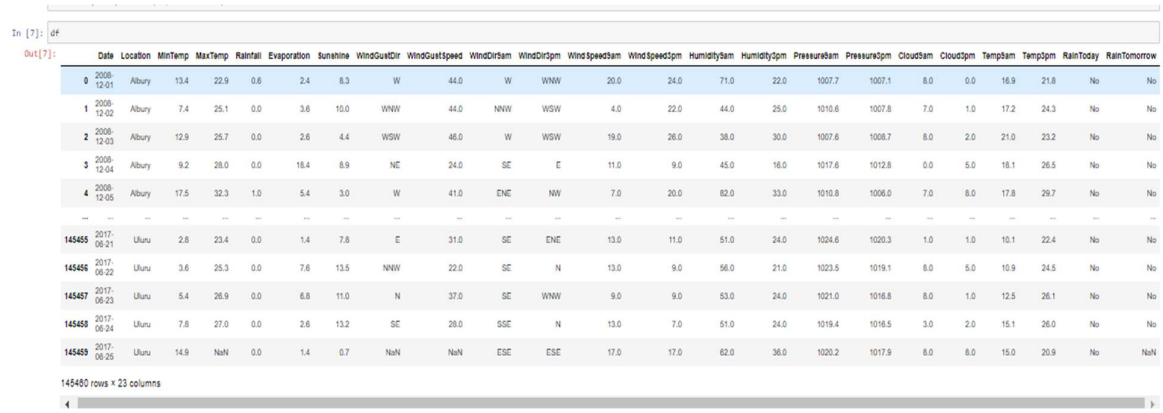
System Implementation

4.1 Codes:

- Libraries:



- CSV Reading:



In [7]: df

Out[7]:

	Date	Location	MinTemp	MaxTemp	Rainfall	Evaporation	Sunshine	WindGustDir	WindGustSpeed	WindDir9am	WindDir3pm	WindSpeed9am	WindSpeed3pm	Humidity9am	Humidity3pm	Pressure9am	Pressure3pm	Cloud9am	Cloud3pm	Temp9am	Temp3pm	RainToday	RainTomorrow
0	2008-12-01	Albury	13.4	22.9	0.6	2.4	8.3	W	44.0	W	WNW	20.0	24.0	71.0	22.0	1007.7	1007.1	8.0	0.0	16.9	21.8	No	No
1	2008-12-02	Albury	7.4	25.1	0.0	3.6	10.0	WNW	44.0	NNW	WSW	4.0	22.0	44.0	25.0	1010.6	1007.8	7.0	1.0	17.2	24.3	No	No
2	2008-12-03	Albury	12.9	25.7	0.0	2.6	4.4	WSW	46.0	W	WSW	19.0	26.0	38.0	30.0	1007.6	1008.7	8.0	2.0	21.0	23.2	No	No
3	2008-12-04	Albury	9.2	28.0	0.0	18.4	8.9	NE	24.0	SE	E	11.0	9.0	45.0	16.0	1017.6	1012.8	0.0	5.0	18.1	26.5	No	No
4	2008-12-05	Albury	17.5	32.3	1.0	5.4	3.0	W	41.0	ENE	NW	7.0	20.0	82.0	33.0	1010.8	1006.0	7.0	8.0	17.8	29.7	No	No
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
145455	2017-06-21	Uluru	2.8	23.4	0.0	1.4	7.8	E	31.0	SE	ENE	13.0	11.0	51.0	24.0	1024.6	1020.3	1.0	1.0	10.1	22.4	No	No
145456	2017-06-22	Uluru	3.6	25.3	0.0	7.6	13.5	NNW	22.0	SE	N	13.0	9.0	56.0	21.0	1023.5	1019.1	8.0	5.0	10.9	24.5	No	No
145457	2017-06-23	Uluru	5.4	26.9	0.0	6.8	11.0	N	37.0	SE	WNW	9.0	9.0	53.0	24.0	1021.0	1016.8	8.0	1.0	12.5	26.1	No	No
145458	2017-06-24	Uluru	7.8	27.0	0.0	2.6	13.2	SE	28.0	SSE	N	13.0	7.0	51.0	24.0	1019.4	1016.5	3.0	2.0	15.1	26.0	No	No
145459	2017-06-25	Uluru	14.9	NaN	0.0	1.4	0.7	NaN	NaN	ESE	ESE	17.0	17.0	62.0	36.0	1020.2	1017.9	8.0	8.0	15.0	20.9	No	NaN

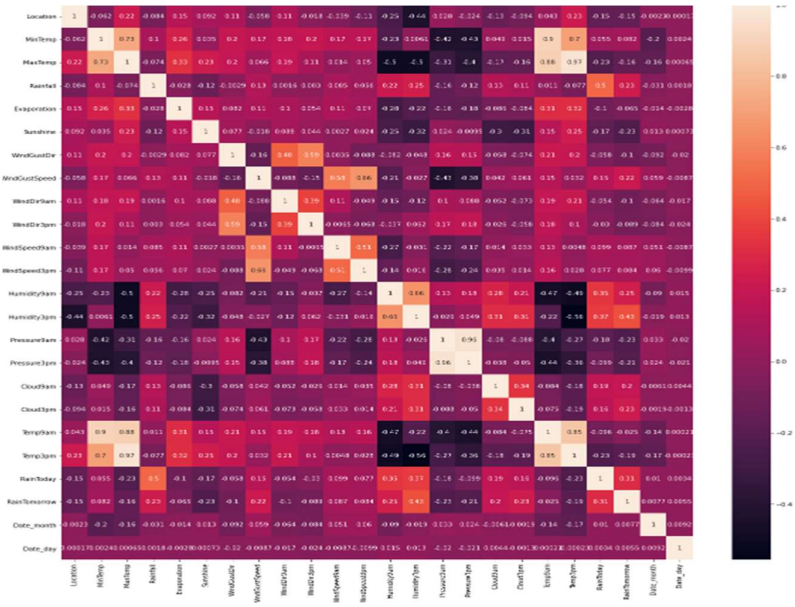
145460 rows × 23 columns

- Finding missing values

```
In [3]: numerical_feature = [feature for feature in df.columns if df[feature].dtypes != 'O']
        discrete_feature=[feature for feature in numerical_feature if len(df[feature].unique())<25]
        continuous_feature = [feature for feature in numerical_feature if feature not in discrete_feature]
        categorical_feature = [feature for feature in df.columns if feature not in numerical_feature]
        print("Numerical Features Count {}".format(len(numerical_feature)))
        print("Discrete feature Count {}".format(len(discrete_feature)))
        print("Continuous feature Count {}".format(len(continuous_feature)))
        print("Categorical feature Count {}".format(len(categorical_feature)))

        Numerical Features Count 16
        Discrete feature Count 2
        Continuous feature Count 14
        Categorical feature Count 7
```

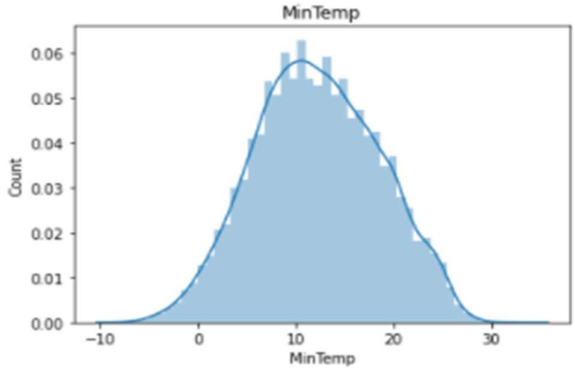
```
In [4]: # Handle Missing Values
        df.isnull().sum()*100/len(df)

Out[4]: Date             0.000000
        Location         0.000000
        MinTemp          1.020899
        MaxTemp          0.866905
        Rainfall         2.241853
        Evaporation     43.166506
        Sunshine        48.009762
        WindGustDir      7.098859
        WindGustSpeed    7.055548
        WindDir9am       7.263853
        WindDir3pm       2.906641
        WindSpeed9am     1.214767
        WindSpeed3pm     2.105046
        Humidity9am      1.824557
        Humidity3pm      3.098446
        Pressure9am     10.356799
        Pressure3pm     10.331363
        Cloud9am        38.421559
        Cloud3pm        40.807095
        Temp9am          1.214767
        Temp3pm          2.481094
        RainToday        2.241853
        RainTomorrow     2.245978
        dtype: float64
```

- Heap Map creation

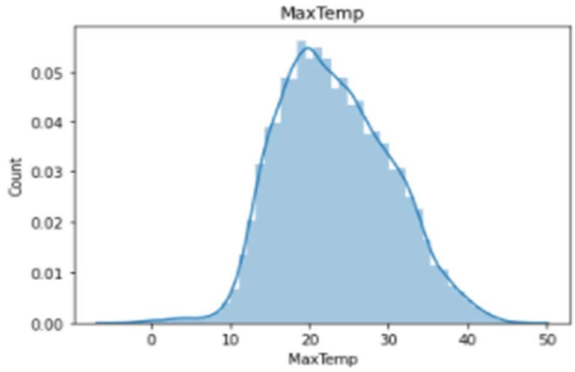


- Printing Distribution Plot

```
In [9]: for feature in continuous_feature:
            data=df.copy()
            sns.distplot(df[feature])
            plt.xlabel(feature)
            plt.ylabel("Count")
            plt.title(feature)
            plt.figure(figsize=(15,15))
            plt.show()
```

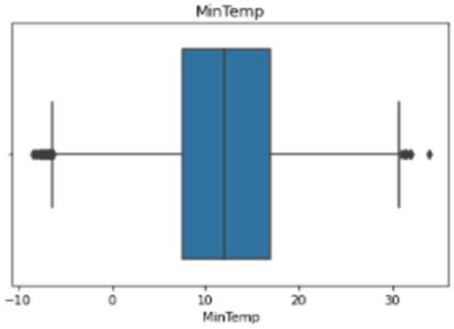


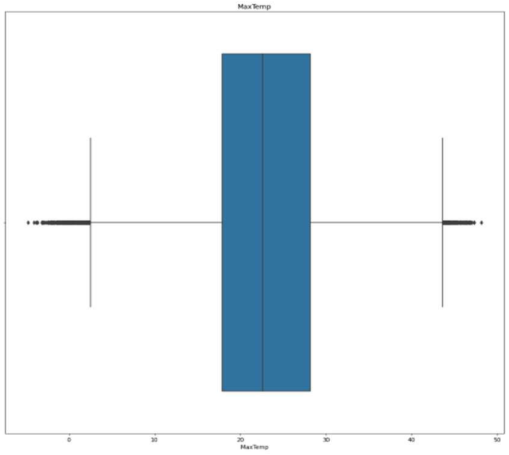

```
<Figure size 1080x1080 with 0 Axes>
```

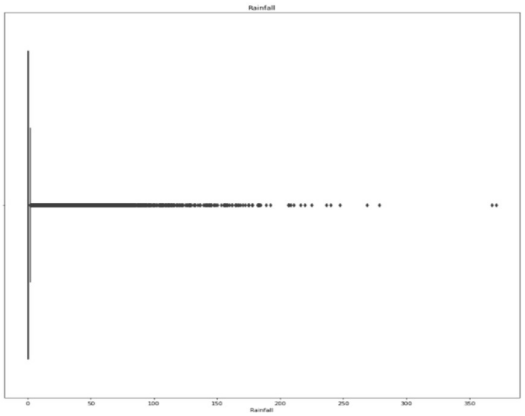


- Boxplot for all the continuous features to see outliers

```
In [10]: #A for loop is used to plot a boxplot for all the continuous features to see the outliers
         for feature in continuous_feature:
             data=df.copy()
             sns.boxplot(data[feature])
             plt.title(feature)
             plt.figure(figsize=(15,15))
```







- Filling null values in continuous features

```
In [13]: for feature in continuous_feature:
             if(df[feature].isnull().sum()*100/len(df))>0:
                 df[feature] = df[feature].fillna(df[feature].median())
```

```
In [14]: df.isnull().sum()*100/len(df)
```

```
Out[14]: Date             0.000000
         Location         0.000000
         MinTemp          0.000000
         MaxTemp          0.000000
         Rainfall         0.000000
         Evaporation      0.000000
         Sunshine         0.000000
         WindGustDir      7.098859
         WindGustSpeed    0.000000
         WindDir9am       7.263853
         WindDir3pm       2.906641
         WindSpeed9am     0.000000
         WindSpeed3pm     0.000000
         Humidity9am      0.000000
         Humidity3pm      0.000000
         Pressure9am      0.000000
         Pressure3pm      0.000000
         Cloud9am         0.000000
         Cloud3pm         0.000000
         Temp9am          0.000000
         Temp3pm          0.000000
         RainToday        2.241853
         RainTomorrow     2.245978
         dtype: float64
```

- Filling all Nan Values & dummies values in following columns:

```
In [14]: def mode_nan(df,variable):
             mode=df[variable].value_counts().index[0]
             df[variable].fillna(mode,inplace=True)
         mode_nan(df,"Cloud9am")
         mode_nan(df,"Cloud3pm")
```

```
In [15]: df["RainToday"] = pd.get_dummies(df["RainToday"], drop_first = True)
         df["RainTomorrow"] = pd.get_dummies(df["RainTomorrow"], drop_first = True)
         df
```

- Assigning numerical values

```
In [19]: windgustdir = {'NNW':0, 'NW':1, 'WNW':2, 'N':3, 'W':4, 'WSW':5, 'NNE':6, 'S':7, 'SSW':8, 'SW':9, 'SSE':10,
                'NE':11, 'SE':12, 'ESE':13, 'ENE':14, 'E':15}
         winddir9am = {'NNW':0, 'N':1, 'NW':2, 'NNE':3, 'WNW':4, 'W':5, 'WSW':6, 'SW':7, 'SSW':8, 'NE':9, 'S':10,
                 'SSE':11, 'ENE':12, 'SE':13, 'ESE':14, 'E':15}
         winddir3pm = {'NW':0, 'NNW':1, 'N':2, 'WNW':3, 'W':4, 'NNE':5, 'WSW':6, 'SSW':7, 'S':8, 'SW':9, 'SE':10,
                'NE':11, 'SSE':12, 'ENE':13, 'E':14, 'ESE':15}
         df["WindGustDir"] = df["WindGustDir"].map(windgustdir)
         df["WindDir9am"] = df["WindDir9am"].map(winddir9am)
         df["WindDir3pm"] = df["WindDir3pm"].map(winddir3pm)
```

```
In [20]: df["WindGustDir"] = df["WindGustDir"].fillna(df["WindGustDir"].value_counts().index[0])
         df["WindDir9am"] = df["WindDir9am"].fillna(df["WindDir9am"].value_counts().index[0])
         df["WindDir3pm"] = df["WindDir3pm"].fillna(df["WindDir3pm"].value_counts().index[0])
```

```
In [25]: location = {'Portland':1, 'Cairns':2, 'Walpole':3, 'Dartmoor':4, 'MountGambier':5,
                'NorfolkIsland':6, 'Albany':7, 'Witchcliffe':8, 'CoffsHarbour':9, 'Sydney':10,
                'Darwin':11, 'MountGinini':12, 'NorahHead':13, 'Ballarat':14, 'GoldCoast':15,
                'SydneyAirport':16, 'Hobart':17, 'Watsonia':18, 'Newcastle':19, 'Wollongong':20,
                'Brisbane':21, 'Williamtown':22, 'Launceston':23, 'Adelaide':24, 'MelbourneAirport':25,
                'Perth':26, 'Sale':27, 'Melbourne':28, 'Canberra':29, 'Albury':30, 'Penrith':31,
                'Nuriootpa':32, 'BadgerysCreek':33, 'Tuggeranong':34, 'PerthAirport':35, 'Bendigo':36,
                'Richmond':37, 'WaggaWagga':38, 'Townsville':39, 'PearceRAAF':40, 'SalmonGums':41,
                'Moree':42, 'Cobar':43, 'Mildura':44, 'Katherine':45, 'AliceSprings':46, 'Nhil':47,
                'Woomera':48, 'Uluru':49}
         df["Location"] = df["Location"].map(location)
```

- IQR range for handling Outliers:

```
In [35]: IQR=df.MinTemp.quantile(0.75)-df.MinTemp.quantile(0.25)
         lower_bridge=df.MinTemp.quantile(0.25)-(IQR*1.5)
         upper_bridge=df.MinTemp.quantile(0.75)+(IQR*1.5)
         print(lower_bridge, upper_bridge)

         -5.950000000000002 30.450000000000003
```

```
In [36]: df.loc[df['MinTemp']>=30.45,'MinTemp']=30.45
         df.loc[df['MinTemp']<=-5.95,'MinTemp']=-5.95
```

```
In [37]: IQR=df.MaxTemp.quantile(0.75)-df.MaxTemp.quantile(0.25)
         lower_bridge=df.MaxTemp.quantile(0.25)-(IQR*1.5)
         upper_bridge=df.MaxTemp.quantile(0.75)+(IQR*1.5)
         print(lower_bridge, upper_bridge)

         2.700000000000001 43.5
```

```
In [38]: df.loc[df['MaxTemp']>=43.5,'MaxTemp']=43.5
         df.loc[df['MaxTemp']<=2.7,'MaxTemp']=2.7
```

```
In [39]: IQR=df.Rainfall.quantile(0.75)-df.Rainfall.quantile(0.25)
         lower_bridge=df.Rainfall.quantile(0.25)-(IQR*1.5)
         upper_bridge=df.Rainfall.quantile(0.75)+(IQR*1.5)
         print(lower_bridge, upper_bridge)

         -0.8999999999999999 1.5
```

- Cleaned CSV Data

```
In [28]: df
```

Out[28]:

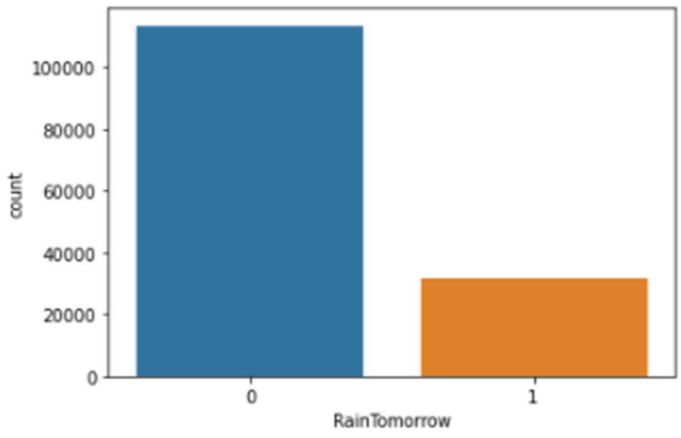
	Date	Location	MinTemp	MaxTemp	Rainfall	Evaporation	Sunshine	WindGustDir	WindGustSpeed	WindDir9am	WindDir3pm	WindSpeed9am	WindSpeed3pm	Humidity9am	Humidity3pm	Pressure9am
0	2008-12-01	30	13.4	22.9	0.6	2.4	8.3	4.0	44.0	5.0	3.0	20.0	24.0	71.0	22.0	1007.7
1	2008-12-02	30	7.4	25.1	0.0	3.6	10.0	2.0	44.0	0.0	6.0	4.0	22.0	44.0	25.0	1010.6
2	2008-12-03	30	12.9	25.7	0.0	2.6	4.4	5.0	46.0	5.0	6.0	19.0	26.0	38.0	30.0	1007.6
3	2008-12-04	30	9.2	28.0	0.0	18.4	8.9	11.0	24.0	13.0	14.0	11.0	9.0	45.0	16.0	1017.6
4	2008-12-05	30	17.5	32.3	1.0	5.4	3.0	4.0	41.0	12.0	0.0	7.0	20.0	82.0	33.0	1010.8
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
145455	2017-06-21	49	2.8	23.4	0.0	1.4	7.8	15.0	31.0	13.0	13.0	13.0	11.0	51.0	24.0	1024.6
145456	2017-06-22	49	3.6	25.3	0.0	7.6	13.5	0.0	22.0	13.0	2.0	13.0	9.0	56.0	21.0	1023.5
145457	2017-06-23	49	5.4	26.9	0.0	6.8	11.0	3.0	37.0	13.0	3.0	9.0	9.0	53.0	24.0	1021.0
145458	2017-06-24	49	7.8	27.0	0.0	2.6	13.2	12.0	28.0	11.0	2.0	13.0	7.0	51.0	24.0	1019.4
145459	2017-06-25	49	14.9	22.6	0.0	1.4	0.7	4.0	39.0	14.0	15.0	17.0	17.0	62.0	36.0	1020.2

145460 rows × 25 columns

- Count Plot of Rain Tomorrow

```
In [30]: sns.countplot(df["RainTomorrow"])
```

```
Out[30]: <matplotlib.axes._subplots.AxesSubplot at 0x7f4a74b98310>
```

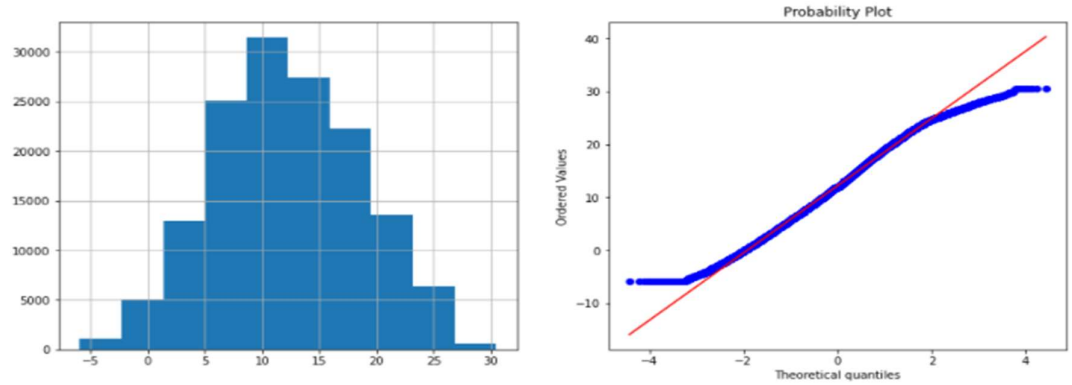


- Probability Plot to check fitting of the model

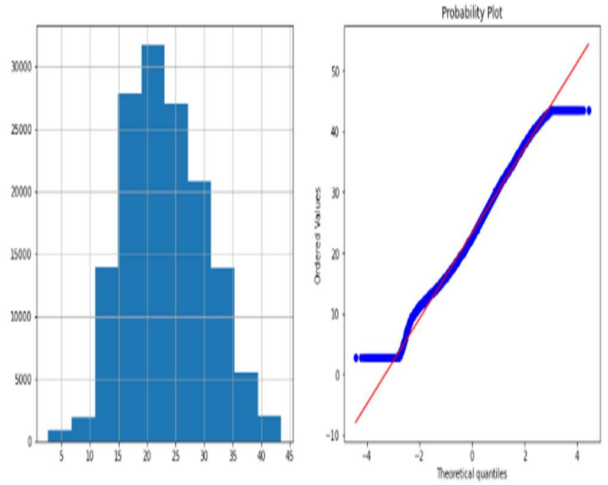
In [58]:
```
def qq_plots(df, variable):
    plt.figure(figsize=(15,6))
    plt.subplot(1, 2, 1)
    df[variable].hist()
    plt.subplot(1, 2, 2)
    stats.probplot(df[variable], dist="norm", plot=plt)
    plt.show()
```

In [59]:
```
for feature in continuous_feature:
    print(feature)
    plt.figure(figsize=(15,6))
    plt.subplot(1, 2, 1)
    df[feature].hist()
    plt.subplot(1, 2, 2)
    stats.probplot(df[feature], dist="norm", plot=plt)
    plt.show()
```

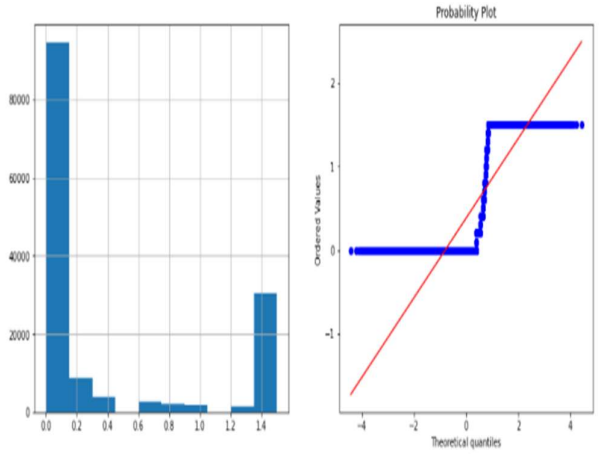
MinTemp



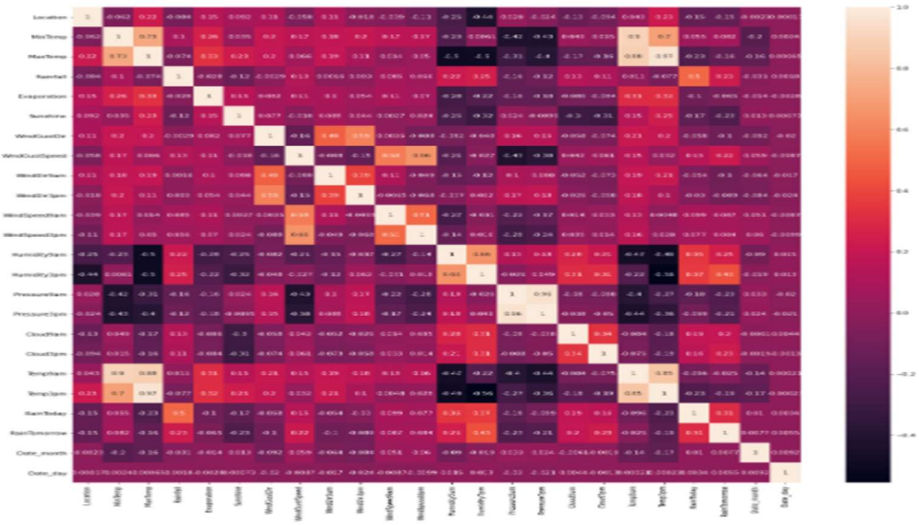
MaxTemp



Rainfall



- Heatmap of Cleaned Data



- Writing DataFrame to CSV File

```
In [60]: df.to_csv("preprocessed_2.csv", index=False)
```

- Splitting data into train and test data

```
In [64]: X_train, X_test, y_train, y_test = train_test_split(X,Y, test_size =0.2, stratify = Y, random_state = 0)
```

```
In [65]: y_train
```

```
Out[65]: 43447     0
         140904    0
         54615     0
         109793    0
         74462     0
                  ..
         106567    0
         52791     1
         56339     0
         82299     1
         115175    0
         Name: RainTomorrow, Length: 116368, dtype: uint8
```

```
In [66]: sm=SMOTE(random_state=0)
         X_train_res, y_train_res = sm.fit_resample(X_train, y_train)
         print("The number of classes before fit {}".format(Counter(y_train)))
         print("The number of classes after fit {}".format(Counter(y_train_res)))
```

```
The number of classes before fit Counter({0: 90866, 1: 25502})
The number of classes after fit Counter({0: 90866, 1: 90866})
```

- CatBoostClassifier

```
In [67]: cat = CatBoostClassifier(iterations=2000, eval_metric = "AUC")
         cat.fit(X_train_res, y_train_res)
```

```
Learning rate set to 0.050311
0:      total: 570ms    remaining: 18m 59s
1:      total: 740ms    remaining: 12m 19s
2:      total: 907ms    remaining: 10m 3s
3:      total: 1.08s    remaining: 8m 59s
4:      total: 1.28s    remaining: 8m 30s
5:      total: 1.57s    remaining: 8m 42s
6:      total: 1.71s    remaining: 8m 6s
7:      total: 1.86s    remaining: 7m 43s
8:      total: 2.04s    remaining: 7m 30s
9:      total: 2.16s    remaining: 7m 9s
10:     total: 2.44s    remaining: 7m 20s
11:     total: 2.79s    remaining: 7m 42s
12:     total: 3.08s    remaining: 7m 50s
13:     total: 3.38s    remaining: 8m
14:     total: 3.67s    remaining: 8m 6s
15:     total: 3.85s    remaining: 7m 57s
16:     total: 3.98s    remaining: 7m 43s
17:     total: 4.12s    remaining: 7m 33s
18:     total: 4.23s    remaining: 7m 21s
```

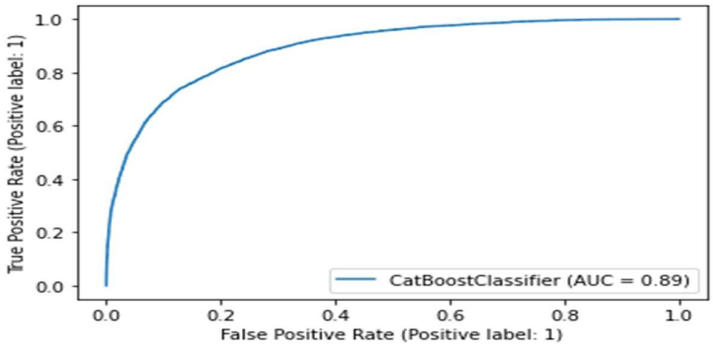
```
In [68]: y_pred = cat.predict(X_test)
         print(confusion_matrix(y_test,y_pred))
         print(accuracy_score(y_test,y_pred))
         print(classification_report(y_test,y_pred))
```

```
[[21518  1199]
 [ 2792  3583]]
0.8628145194555205
              precision    recall  f1-score   support

           0       0.89      0.95      0.92     22717
           1       0.75      0.56      0.64      6375

    accuracy                           0.86     29092
   macro avg       0.82      0.75      0.78     29092
weighted avg       0.86      0.86      0.86     29092
```

- Plotting Roc Curve to show its performance

```
In [68]: metrics.plot_roc_curve(cat, X_test, y_test)
         metrics.roc_auc_score(y_test, y_pred, average=None)
```

```
Out[68]: 0.7517801180251119
```



- RandomForestClassifier

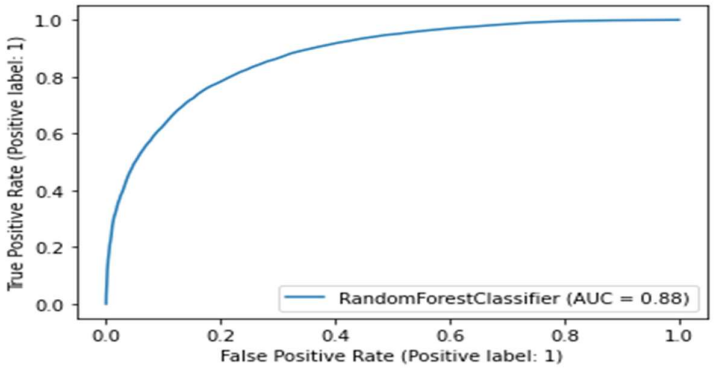
```
In [70]: rf=RandomForestClassifier()
         rf.fit(X_train_res,y_train_res)

Out[70]: RandomForestClassifier()
```

```
In [71]: y_pred1 = rf.predict(X_test)
         print(confusion_matrix(y_test,y_pred1))
         print(accuracy_score(y_test,y_pred1))
         print(classification_report(y_test,y_pred1))
```

```
[[20670  2047]
 [ 2473  3902]]
0.8446308263440121
              precision    recall  f1-score   support

           0       0.89      0.91      0.90     22717
           1       0.66      0.61      0.63      6375

    accuracy                           0.84     29092
   macro avg       0.77      0.76      0.77     29092
weighted avg       0.84      0.84      0.84     29092
```

- Plotting Roc Curve to show its performance

```
In [71]: metrics.plot_roc_curve(rf, X_test, y_test)
         metrics.roc_auc_score(y_test, y_pred1, average=None)

Out[71]: 0.7603041067111355
```



- LogisticRegression

```
In [73]: logreg = LogisticRegression()
         logreg.fit(X_train_res, y_train_res)
```

```
C:\Users\shash\anaconda3\lib\site-packages\sklearn\linear_model\_logistic.py:763: ConvergenceWarning: lbfgs failed to converge
(status=1):
STOP: TOTAL NO. of ITERATIONS REACHED LIMIT.

Increase the number of iterations (max_iter) or scale the data as shown in:
    https://scikit-learn.org/stable/modules/preprocessing.html
Please also refer to the documentation for alternative solver options:
    https://scikit-learn.org/stable/modules/linear_model.html#logistic-regression
  n_iter_i = _check_optimize_result(
```

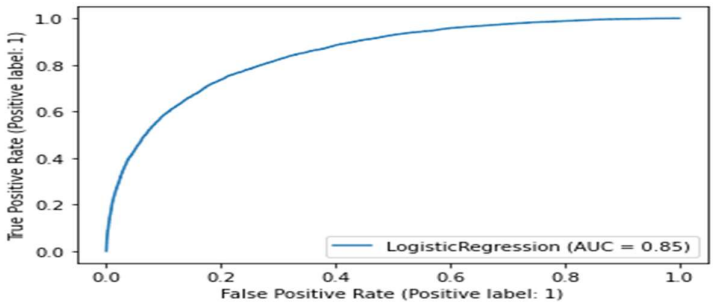
```
Out[73]: LogisticRegression()
```

```
In [74]: y_pred2 = logreg.predict(X_test)
         print(confusion_matrix(y_test,y_pred2))
         print(accuracy_score(y_test,y_pred2))
         print(classification_report(y_test,y_pred2))
```

```
[[17544  5173]
 [ 1524  4851]]
0.7697992575278427
              precision    recall  f1-score   support

           0       0.92      0.77      0.84     22717
           1       0.48      0.76      0.59      6375

    accuracy                           0.77     29092
   macro avg       0.70      0.77      0.72     29092
weighted avg       0.82      0.77      0.79     29092
```

- Plotting Roc Curve to show its performance

```
In [74]: metrics.plot_roc_curve(logreg, X_test, y_test)
         metrics.roc_auc_score(y_test, y_pred2, average=None)

Out[74]: 0.7686327886086863
```

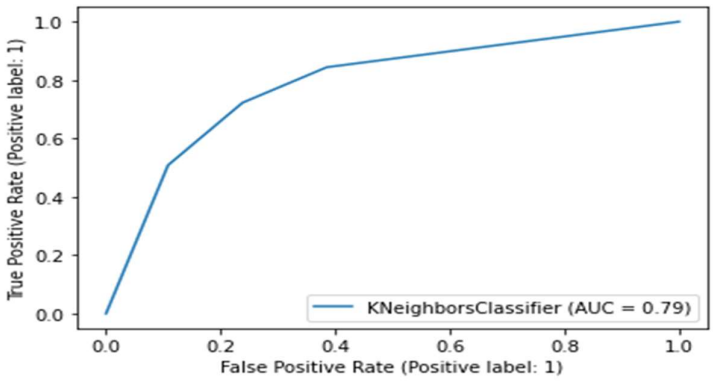


- KNeighborsClassifier

```
In [76]: knn = KNeighborsClassifier(n_neighbors=3)
         knn.fit(X_train_res, y_train_res)

Out[76]: KNeighborsClassifier(n_neighbors=3)

In [77]: y_pred4 = knn.predict(X_test)
         print(confusion_matrix(y_test,y_pred4))
         print(accuracy_score(y_test,y_pred4))
         print(classification_report(y_test,y_pred4))

[[17306  5411]
 [ 1770  4605]]
0.7531623814106971
              precision    recall  f1-score   support

           0       0.91      0.76      0.83     22717
           1       0.46      0.72      0.56      6375

    accuracy                           0.75     29092
   macro avg       0.68      0.74      0.70     29092
weighted avg       0.81      0.75      0.77     29092
```

- Plotting Roc Curve to show its performance

```
In [77]: metrics.plot_roc_curve(knn, X_test, y_test)
         metrics.roc_auc_score(y_test, y_pred4, average=None)

Out[77]: 0.7420806392724806
```



- Dumping files of best Accuracy Model

```
In [79]: joblib.dump(cat, "cat.pkl")
Out[79]: ['cat.pkl']
```

- Connecting Api In IBM Watson

```
In [80]: from ibm_watson_machine_learning import APIClient

wml_credentials = {
                    "url": "https://us-south.ml.cloud.ibm.com",
                    "apikey":"x7aeFn41_DfFa5lYAYCj-lT9BpYcc6fzeurJ6SMbelS0"
                   }

client = APIClient(wml_credentials)
```

- Getting SpaceID

```
In [81]: client.spaces.list()

Note: 'limit' is not provided. Only first 50 records will be displayed if the number of records exceed 50
------------------------------------  ------  ------------------------
ID                                    NAME    CREATED
eea7ea52-c17d-4c8e-a335-47ab7b980efa  models  2021-06-02T12:58:36.088Z
------------------------------------  ------  ------------------------

In [82]: space_id="eea7ea52-c17d-4c8e-a335-47ab7b980efa"

In [83]: client.set.default_space(space_id)
Out[83]: 'SUCCESS'
```

- Now Using Dumped Model in Flask Application(cat.pkl)

```
from flask import Flask,render_template,url_for,request,jsonify

from flask_cors import cross_origin
import pandas as pd
import numpy as np
import datetime
import pickle

app = Flask(__name__, template_folder="template")
model = pickle.load(open("cat.pkl", "rb"))
print("Model Loaded")

@app.route("/",methods=['GET'])
@cross_origin()
def home():
    return render_template("index.html")

@app.route("/predict",methods=['GET', 'POST'])
@cross_origin()
def predict():
    if request.method == "POST":
        # DATE
        date = request.form['date']
        day = float(pd.to_datetime(date, format="%Y-%m-%dT").day)
        month = float(pd.to_datetime(date, format="%Y-%m-%dT").month)
        # MinTemp
        minTemp = float(request.form['mintemp'])
        # MaxTemp
        maxTemp = float(request.form['maxtemp'])
        # Rainfall
        rainfall = float(request.form['rainfall'])
        # Evaporation
        evaporation = float(request.form['evaporation'])
        # Sunshine
        sunshine = float(request.form['sunshine'])
        # Wind Gust Speed
        windGustSpeed = float(request.form['windgustspeed'])
        # Wind Speed 9am
        windSpeed9am = float(request.form['windspeed9am'])
        # Wind Speed 3pm
        windSpeed3pm = float(request.form['windspeed3pm'])
        # Humidity 9am
        humidity9am = float(request.form['humidity9am'])
        # Humidity 3pm
        humidity3pm = float(request.form['humidity3pm'])
        # Pressure 9am
        pressure9am = float(request.form['pressure9am'])
```

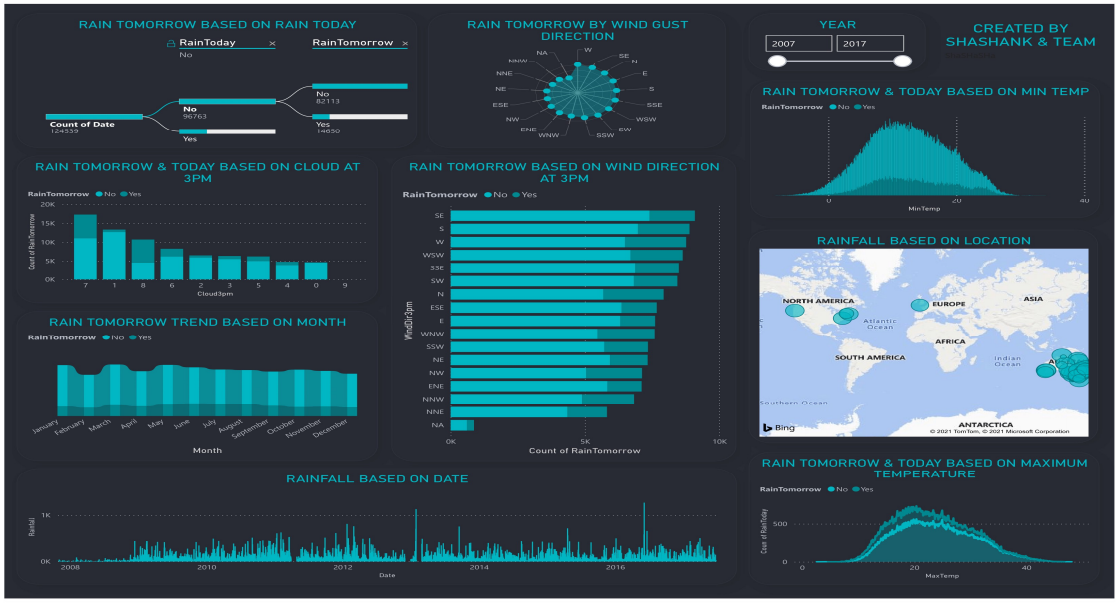
```
        # Pressure 3pm
        pressure3pm = float(request.form['pressure3pm'])
        # Temperature 9am
        temp9am = float(request.form['temp9am'])
        # Temperature 3pm
        temp3pm = float(request.form['temp3pm'])
        # Cloud 9am
        cloud9am = float(request.form['cloud9am'])
        # Cloud 3pm
        cloud3pm = float(request.form['cloud3pm'])
        # Cloud 3pm
        location = float(request.form['location'])
        # Wind Dir 9am
        winddDir9am = float(request.form['winddir9am'])
        # Wind Dir 3pm
        winddDir3pm = float(request.form['winddir3pm'])
        # Wind Gust Dir
        windGustDir = float(request.form['windgustdir'])
        # Rain Today
        rainToday = float(request.form['raintoday'])

        input_lst = [location , minTemp , maxTemp , rainfall , evaporation , s
unshine ,
                     windGustDir , windGustSpeed , winddDir9am , winddDir3pm ,
 windSpeed9am , windSpeed3pm ,
                     humidity9am , humidity3pm , pressure9am , pressure3pm , c
loud9am , cloud3pm , temp9am , temp3pm ,
                     rainToday , month , day]
        pred = model.predict(input_lst)
        output = pred
        if output == 0:
            return render_template("after_sunny.html")
        else:
            return render_template("after_rainy.html")
    return render_template("predictor.html")

if __name__=='__main__':
    app.run(debug=True)
```

CHAPTER 5

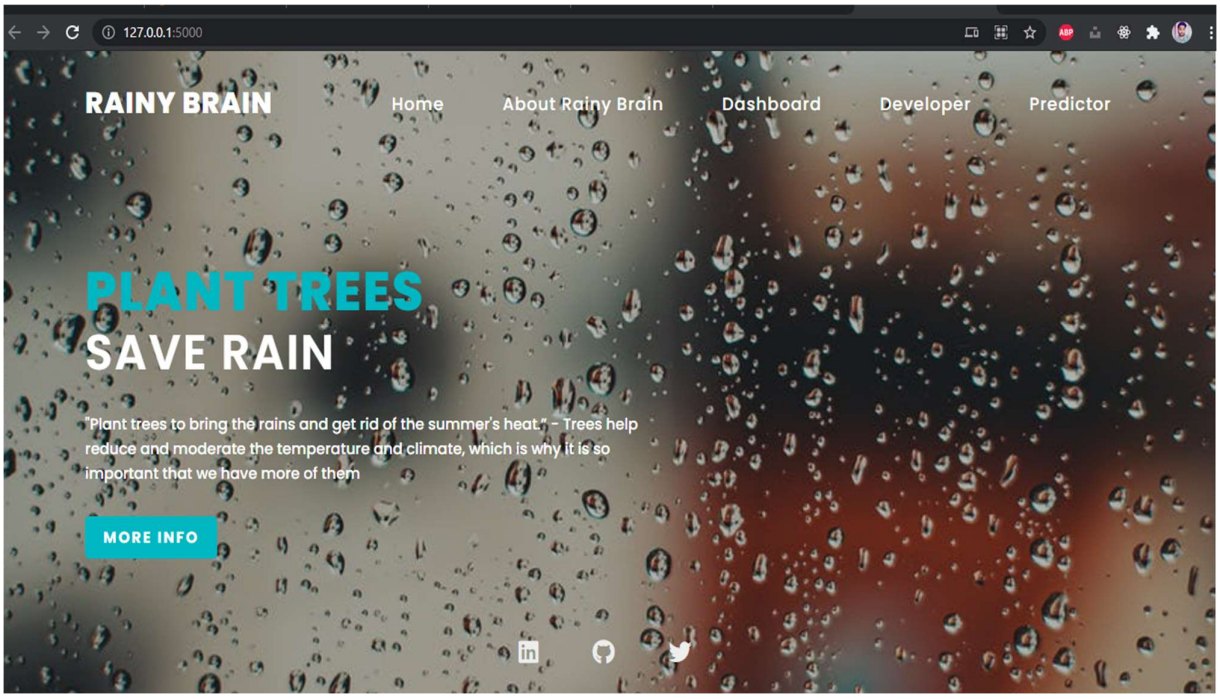
Results and Discussion

5.1 Output/Results

- Performed Visualizations on PowerBI Platform



- Home Page



➢ About

About Rainy Brain

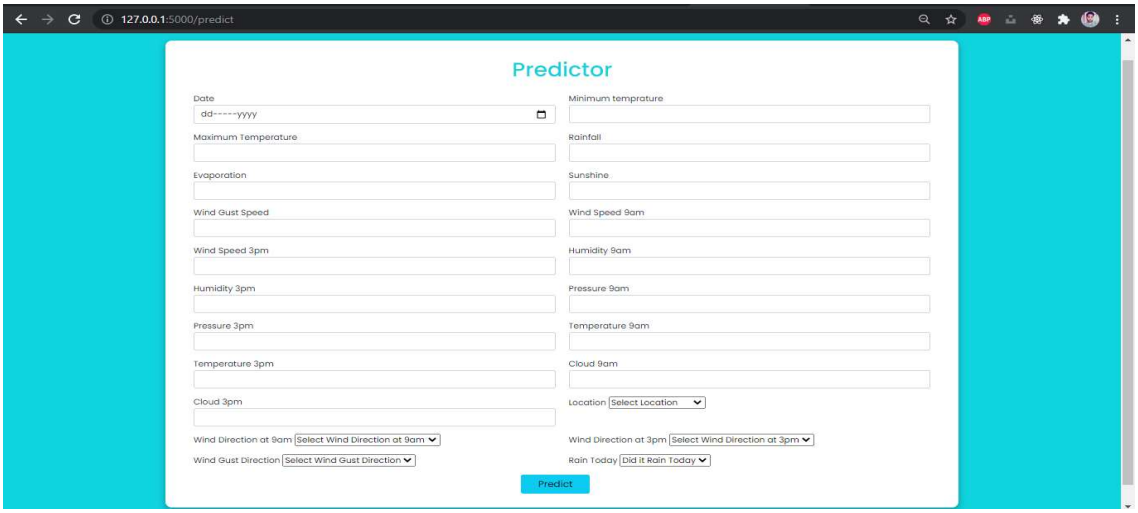
Rainy Brain is a web app which has a Machine Learning model running at the back. The purpose of developing this app is to predict whether it will rain the next day or not. This model is based on the Rain Prediction in Australia dataset.More than 80% of Australia has an annual rainfall of less than 600 mm which is less among the all continents other than Antartica which recieves less rainfall. A place inland near Lake Eyre would only receive 81 mm of rain annually. The average annual rainfall in the Australian desert is low, ranging from 81 to 250 mm. Thunderstorms are relatively common in the region, with an annual average of 15 to 20 thunderstorms. The southern parts of Australia get the usual westerly winds and rain-bearing cold fronts that come when high–pressure systems move towards northern Australia during winter. Cold snaps may bring frosts inland, though temperatures near the coast are mild or near mild all year round. Summers in southern Australia are generally dry and hot with coastal sea breezes. During a lengthy dry spell, hot and dry winds from the interior can cause bushfires in some southern and eastern states, though most commonly Victoria and New South Wales. The tropical areas of northern Australia have a wet summer because of the monsoon. During "the wet", typically October to April, humid north-westerly winds bring showers and thunderstorms. Occasionally, tropical cyclones can bring heavy rainfall to tropical coastal regions, which is also likely to reach further inland.

➢ Developers

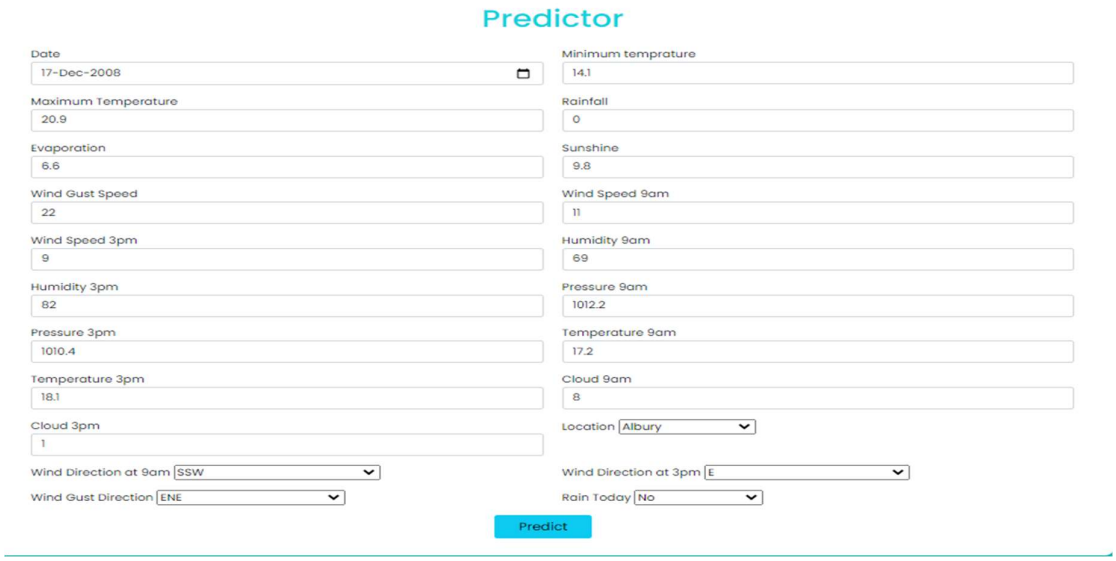
Developer

- Shashank Kumar (20MCA0142)
- Prachi Jha (20MCA0143)
- Krati Jain (20MCA0115)

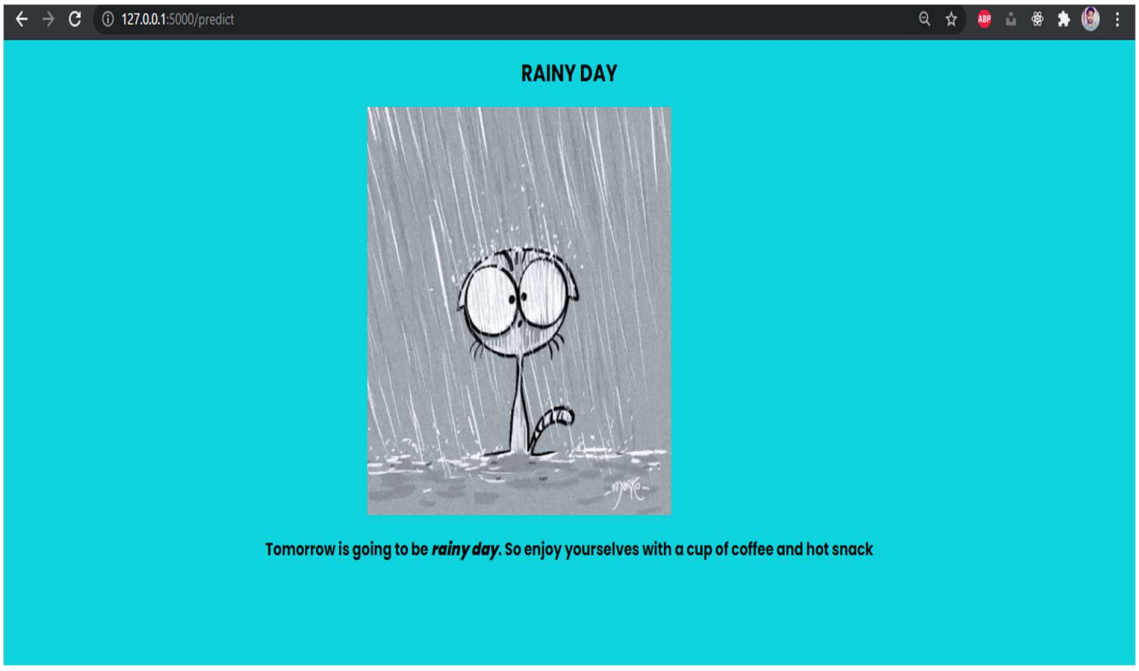
➢ Prediction Page



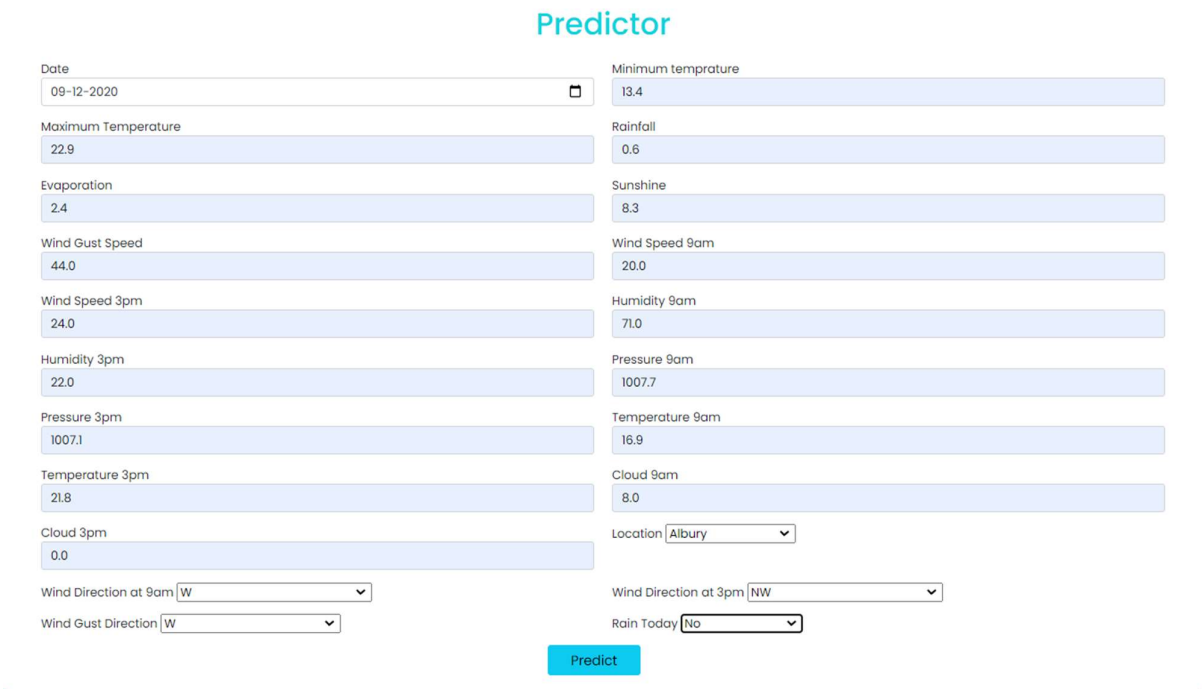
➢ Predicting data for Rainy Day



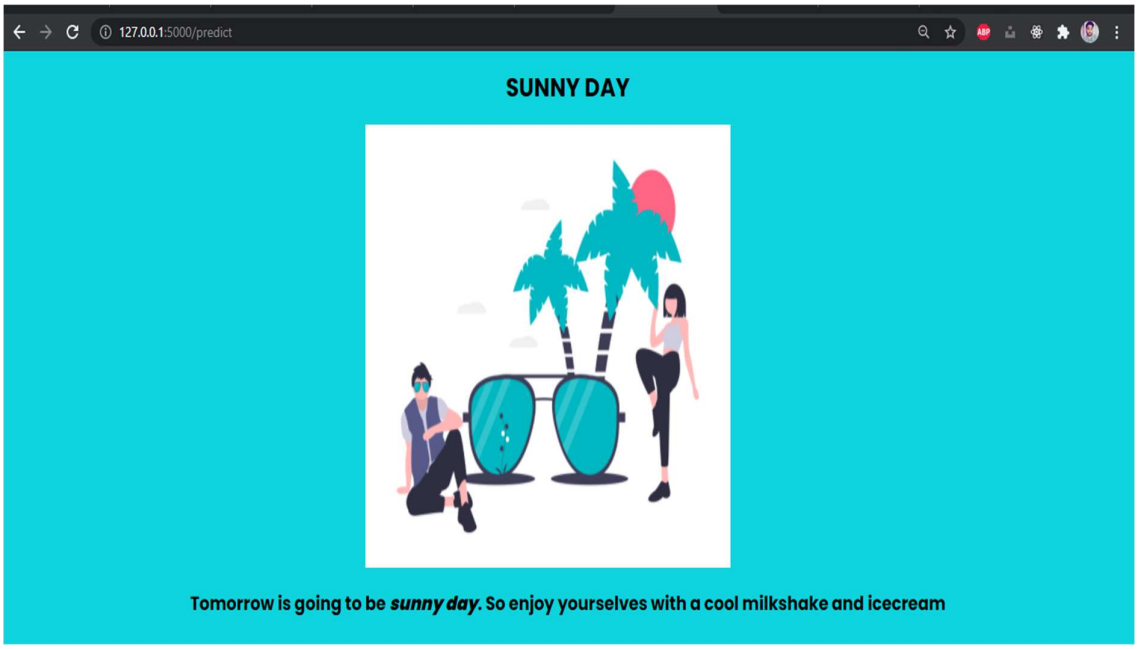
Output:-



➢ Sunny Day



Output:-



CHAPTER 6

Conclusion

Rainfall is one the most significant natural phenomenon that is not only important for the human beings only but the living beings. Due to the changing climatic conditions, rainfall cycles are also changing and the temperature of the earth is rising. The changing temperature is also affecting the agriculture, industry and sometimes may cause flooding and land slide. Therefore, it is essential for the human beings to keep a check upon this natural phenomenon in order to survive. The water is a scarce natural resource without which human life is impossible and also there is no substitute to this natural resource. Thus, predicting the rainfall for agriculture and water reserves, also it also good for keeping human beings alert of natural disasters like flood and landslide. However, to overcome these issues and meet the demands, a system to forecast rainfall is essential using artificial intelligence of neural that is popular within the modern technology.

In this, we explored and applied several pre-processing steps and learned their impact on the overall performance of our classifiers. We also carried a comparative study of all the classifiers with different input data and observed how the input data can affect the model predictions. We can conclude that Australian weather is uncertain and there is no such correlation among rainfall and the respective region and time. We figured certain patterns and relationships among data which helped in determining important features. As we have a huge amount of data, we can apply Deep Learning models such as Cat Boost Classifier, Random Forest Classifier, Logistic Regression and K Neighbors Classifier. It would be great to perform a comparative study between the Machine learning classifiers and Deep learning models.

Currently machine learning used in no. industries. As the data increases the complexity of that data will increase and for that we are using machine for the better understanding of that data. In Weather predictions its pretty helpful with good accuracy score and in rainfall also its gives pretty good predictions. In future we are planning to increase our work in Storm predictions and Crop prediction with the rainfall prediction

CHAPTER 7

References

1) P. Goswami and Srividya, "A novel Neural Network design for long range prediction of rainfall pattern," Current Sci.(Bangalore), vol. 70, no. 6, pp. 447-457, 1996.
2) C. Venkatesanet, S. D. Raskar , S. S. Tambe , B. D. Kulkarni , and R. N. Keshavamurty , "Prediction of all India summer monsoon rainfall using Error-Back-Propagation Neural Networks," Meteorology and Atmospheric Physics, pp. 225-240, 1997.
3) A. K. Sahai, M. K. Soman, and V. Satyan, "All India summer monsoon rainfall prediction using an Artificial Neural Network," Climate dynamics, vol. 16, no. 4, pp. 291-302, 2000.
4) S. Kannan , Subimal Ghosh, "Prediction of daily rainfall state in a river basin using statistical downscaling from GCM output, Stochastic Environmental Research and Risk Assessment. May 2011, Volume 25, Issue 4, pp 457-474.
5) Najmeh Khalili , Saeed Reza Khodashenas et al. "Daily Rainfall Forecasting for Mashhad Synoptic Station using Artificial Neural Networks" 2011 International Conference on Environmental and Computer Science IPCBEE vol.19(2011) © (2011) IACSIT Press, Singapore.
6) Enireddy.Vamsidhar K.V.S.R.P.Varma et al. "Prediction of Rainfall Using Backpropagation Neural Network Model" (IJCSE) International Journal on Computer Science and Engineering Vol. 02, No. 04, 2010, pp. 1119-1121.
7) A. Elshafie, A. Shehata,Hasan G. El Mazoghi et al. "Artificial neural network technique for rainfall forecasting applied to Alexandria, Egypt" International Journal of the Physical Sciences Vol. 6(6), pp. 1306– 1316, 18 March, 2011 ISSN 1992-1950 ©2011 Academic Journals.
8) Herbert K. H. Lee ISDS, Duke University, "A Framework for Nonparametric Regression Using Neural Networks", Technical Report 00-32, Duke University, Institute of Statistics and Decision Sciences.
9) Khaled Hammouda ,Prof. Fakhreddine Karray ''A Comparative Study of Data Clustering Techniques'' SYDE 625: Tools of Intelligent Systems Design. Course Project
10) [Online] Imbalanced-learn Documentation Available: https://imbalanced-learn. readthedocs.io/en/stable/introduction.html