PROFESSIONAL TRAINING REPORT

at

SATHYABAMA INSTITUTE OF SCIENCE AND TECHNOLOGY

(Deemed to be University)

Submitted in partial fulfillment of the requirements for the award of

Bachelor of Engineering Degree in Computer Science and Engineering

By

KANTIPUDI PRANATHI

(39110448)

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING
SCHOOL OF COMPUTING

SATHYABAMA INSTITUTE OF SCIENCE AND TECHNOLOGY
JEPPIAAR NAGAR, RAJIV GANDHI SALAI,
CHENNAI – 600119, TAMILNADU

APRIL 2022

SATHYABAMA

INSTITUTE OF SCIENCE AND TECHNOLOGY

(DEEMED TO BE UNIVERSITY)

Accredited with Grade "A" by NAAC

(Established under Section 3 of UGC Act, 1956)

JEPPIAAR NAGAR, RAJIV GANDHI SALAI

CHENNAI– 600119

www.sathyabama.ac.in

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

BONAFIDE CERTIFICATE

This is to certify that this Professional Training Report is the bonafide work of **KANTIPUDI PRANATHI (39110448)** who carried out the training entitled "**PREDICTION OF RAINFALL USING LOGISTIC REGRESSION**" under my supervision from January 2022 to April 2022.

Internal Guide
Dr. A.Jemshia Miriam

Head of the Department

Submitted for Viva voice Examination held on _____

Internal Examiner **External Examiner**

DECLARATION

I, **KANTIPUDI PRANATHI** hereby declare that the professional training report entitled **PREDICTION OF RAINFALL USING LOGISTIC REGRESSION** done by me under the guidance of **Dr. A. Jemshia Miriam** is submitted in partial fulfillment of the requirements for the award of Bachelor of Engineering Degree in Computer Science and Engineering.

DATE:

PLACE: **SIGNATURE OF THE CANDIDATE**

ACKNOWLEDGEMENT

I am pleased to acknowledge my sincere thanks to **Board of Management** of **SATHYABAMA** for their kind encouragement in doing this professional training report and for completing it successfully. I am grateful to them.

I convey my thanks to **Dr. T. Sasikala, M.E., Ph.D.**, **Dean**, School of Computing, **Dr. S. Vigneshwari, M.E., Ph.D., and Dr. L. Lakshmanan, M.E., Ph.D., Heads of the Department** of **Computer Science and Engineering** for providing me necessary support and details at the right time during the progressive reviews.

I would like to express my sincere and deep sense of gratitude to my Project Guide **Dr. A. Jemshia Miriam** for her valuable guidance, suggestions, and constant encouragement paved way for the successful completion of my project work.

I wish to express my thanks to all Teaching and Non-teaching staff members of the **Department of Computer Science and Engineering** who were helpful in many ways for the completion of the project.

TRAINING CERTIFICATE

ABSTRACT

The professional training report introduces current supervised learning models which are based on machine learning algorithm for Rainfall prediction in India. Rainfall is always a major issue across the world as it affects all the major factor on which the human being is depended.

In current, Unpredictable and accurate rainfall prediction is a challenging task. We apply rainfall data of India to different machine learning algorithms and compare the accuracy of classifiers such as SVM, Navie Bayes, Logistic Regression, AdaBoost Classifier, Random Forest and Multilayer Perceptron (MLP).

Our motive if to get the optimized result and a better rainfall prediction. This is done using Python as programming language and NumPy, pandas, matplotlib, sklearn as the libraries used to preprocess, visualize and train the models for prediction. 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.

Keywords-Machine learning, Logistic Regression, Accuracy, Prediction.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF ABBREVIATIONS

ABBREVIATION	EXPANSION
ML	MACHINE LEARNING
LG	LOGISTIC REGRESSION
DB	DECISION BOUNDARY

CHAPTER 1

INTRODUCTION

1.1 OUTLINE OF PROJECT

The main goal of this project is to predict the rainfall in the particular area using machine learning algorithm which gives best accuracy score. In this project logistic regression is used to get best results with more accuracy. Logistic regression is with 84 percent accuracy where as others are with less accuracy score. So, in this project logistic regression is preferred for the data set that is collected from the Kaggle.

1.2 LITERATURE REVIEW

Climate is the important aspect of human life. So, the Prediction should accurate as much as possible. Making a good prediction of climate is always a major task now a day because of the climate change. Now climate change is the biggest issue all over the world. People are working on to detect the patterns in climate change as it affects the economy in production to infrastructure. So as in rainfall also making prediction of rainfall is a challenging task with a good accuracy rate. Making prediction on rainfall cannot be done by the traditional way, so scientist is using machine learning and deep learning to find out the pattern for rainfall prediction.

A bad rainfall prediction can affect the agriculture mostly framers as their whole crop is depend on the rainfall and agriculture is always an important part of every economy. So, making an accurate prediction of the rainfall somewhat good. There are number of techniques are used of machine learning but accuracy is always a matter of concern in prediction made in rainfall. There are number of causes made by rainfall affecting the world ex. Drought, Flood and intense summer heat etc. And it will also affect water resources around the world. Most of the world says that the main cause of this current climate change or global warming is human expansion of the greenhouse gases.

This climate change is impacting the mankind and increasingly influencing their life. This also effecting all the area on which human are depending upon, 3 major area are Water, food and air these are the most important things required by the human to survive. But all these 3 areas are affected due to global warming. This climate changes are not just changing the temperature. The whole water cycle is also get affected.

Steve Oberlin, et.al (2012) proposed various Machine Learning strategies for the Big Data processing. He applied Machine Learning and various techniques from Artificial Intelligence to the complex and powerful data sets. Recommendation engines used by Netflix to see the rating and preferences of audience are one of the applications of Machine Learning. Informatics and Data Mining in which IBM's "Watson" uses different Machine Learning approach to process and depict human language and answer the queries [1]. Linear regression, massaging the data, Perception, k- means are the few strategies used by him for uncovering the relationships and finding patterns in data. The choice of Machine Learning algorithm basically depends on the nature of prediction. The prediction can be estimate type or classification. He also discussed how increasing features can make the algorithm complex and increasing computational requirements.

Part of machine learning includes the uses of tools, methods and techniques which help it form better results. These methods and algorithms provide machine and us a new approach to explore the new knowledge from and given data or the by exploiting the traditional datasets. In some situation, we try to record the behaviour and then model that behaviour. In turn modelling stimulate the people to have a better understanding of the situation. Machine learning method have a slight have a history of statistics. It's helpful for exploring more complicated learning model to take out the true message hidden in large amount of data. Although both machine learning technologies and traditional statistics tools can be applied in data analysis, their fundamental principles and characteristics have a great different.

1.3 OBJECTIVES

The main objective of this project is to predict the rainfall using logistic regression. LG gives the best accuracy for the data which is collected by me through Kaggle. First the dataset is collected from the Kaggle and then import pandas, NumPy, matplotlib, sklearn. Then import the data and then data pre-processing should be done. At-last the predictions are done using the LG algorithm.

The project is done using python programming language and NumPy, pandas, matplotlib, sklearn as the libraries used to pre-process, visualize and train the models for prediction. The results provide a comparison of various evaluation metrics of these machine learning techniques and their reliability to predict the rainfall by analysing the weather data.

In this project using univariate and multivariate analysis we find the missing values and unique values in the dataset. Data modelling and feature scaling is done and finally predicting the results through LG.

CHAPTER 2
AIM AND SCOPE OF THE PRESENT INVESTIGATION

2.1 AIM OF THE PROJECT

The main aim of the project is to give a clear and more accurate and best results for rainfall prediction using the ML algorithm that gives more accuracy rate. The main benefits and risks involved in climate change that effects rainfall on earth. These all concepts of ML algorithms give a clear view in this project.

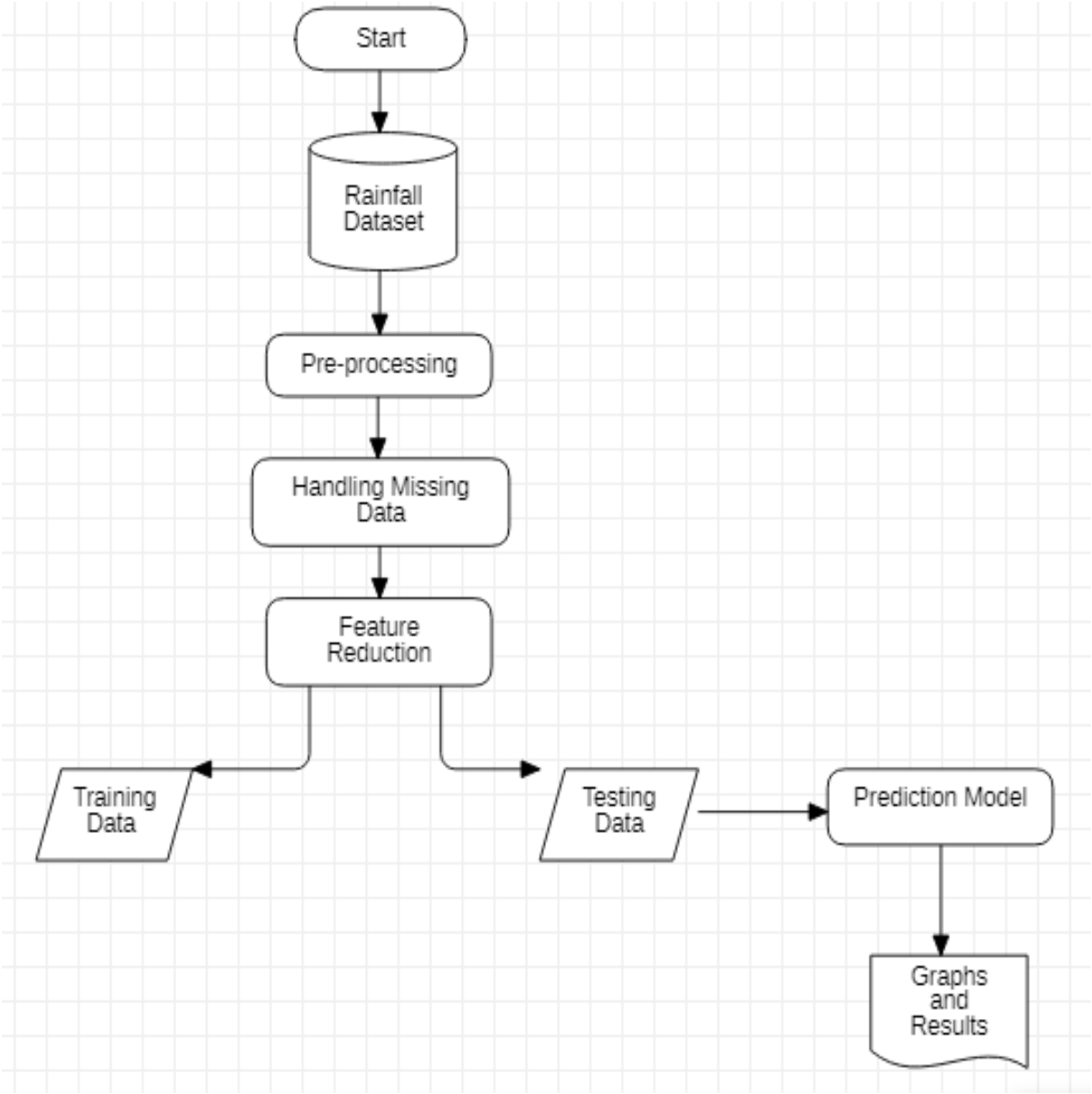
2.2 SCOPE OF THE PROJECT

Identification of extreme weather events to be analyzed by RAIN, by defining appropriate intensity thresholds and by taking into account regional differences in vulnerability and climate. Assessment of the present state-of-the-art forecast systems for extreme weather and their characteristics, to address and estimate their predictive skill. Assessment of the frequency of weather hazards throughout Europe for both the present and future climate.

Rainfall forecasting is very important because heavy and irregular rainfall can have many impacts like destruction of crops and farms, damage of property so a better forecasting model is essential for an early warning that can minimize risks to life and property and also managing the agricultural farms in better way.

In this project, logistic regression has been used for forecasting the probability of rainfall which in turn decides whether it will rain or not. Rainfall Prediction Model has a main objective in prediction of the amount of rain in a specific well or division in advance by using various regression technique and find out which one is best for rainfall prediction. This model also helps the farmer for agriculture to decide the crop, helping the watershed department for water storage and also helps to analyze the ground water level.

2.3 SYSTEM ARCHITECTURE



CHAPTER 3

EXPERIMENTAL OR MATERIALS AND METHODS, ALGORITHMS USED

3.1 ALGORITHM

Logistic regression is one of the most popular Machine Learning algorithms, which comes under the Supervised Learning technique. It is used for predicting the categorical dependent variable using a given set of independent variables.

Logistic regression predicts the output of a categorical dependent variable. Therefore, the outcome must be a categorical or discrete value. It can be either Yes or No, 0 or 1, true or False, etc. but instead of giving the exact value as 0 and 1, it gives the probabilistic values which lie between 0 and 1.

The Logistic regression equation can be obtained from the Linear Regression equation. The mathematical steps to get Logistic Regression equations are given below:

We know the equation of the straight line can be written as:

$$y= b_0 + b_1x_1 + b_2x_2 + b_3x_3 + \cdots + b_nx_n$$

In Logistic Regression y can be between 0 and 1 only, so for this let's divide the above equation by (1-y):

$$\frac{y}{1-y}\text{; 0 for y= 0, and infinity for y=1}$$

But we need range between –[infinity] to +[infinity], then take logarithm of the equation it will become:

$$log\left[\frac{y}{1-y}\right] = b_0 + b_1x_1 + b_2x_2 + b_3x_3 + \cdots + b_nx_n$$

The above equation is the final equation for Logistic Regression.

3.2 LOGISTIC REGRESSION OVERVIEW

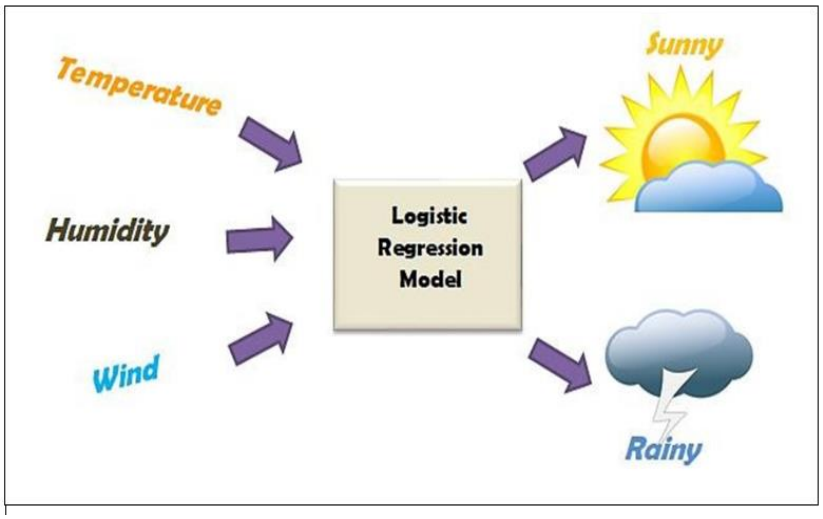


Fig 3.1 Logistic Regression Model

Function used in this algorithm is “Sigmoid” or “Logistic” which is represented as:

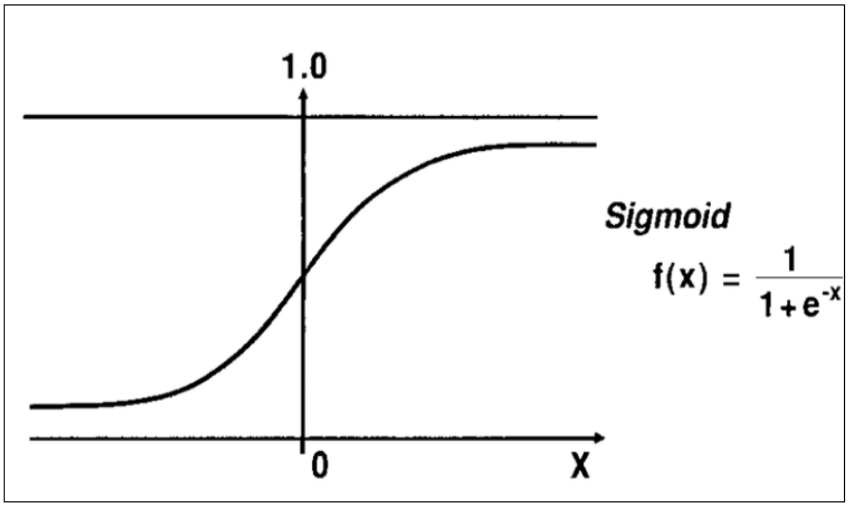


Fig 3.2 Sigmoid Curve

We have a concept called Decision Boundary(db) in Logistic regression. This is a line which splits from one class to other class. By default db will be 0.5 , the process goes like if the resultant data are less than (db)0.5 it can be classified as 0 & if data are greater than (db)0.5 it is 1 and vice versa.

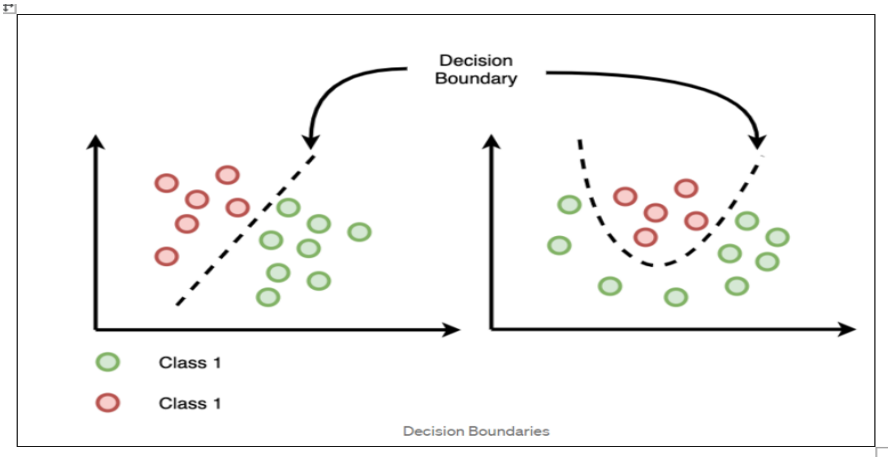


Fig 3.3 Decision Boundaries

3.3 IMPLEMENTATION

Python Implementation of Logistic Regression

To implement the Logistic Regression using Python, we will use the same steps as we have done in previous topics of Regression. Below are the steps:

- Data Pre-processing step
- Fitting Logistic Regression to the Training set
- Predicting the test result
- Test accuracy of the result (Creation of Confusion matrix)
- Visualizing the test set result.

3.4 METHODOLOGY

- **IMPORT LIBRARIES:**

First import the libraries which will help in building the model.

Pandas: Used for data wrangling and analysis.

Numpy: Stands for Numerical Python.

Matplotlib: Matplotlib is a plotting library for the python programming.

Sklearn: The sklearn library contains a lot of efficient tools for machine learning and statistical modeling including classification, regression, clustering and dimensionality reduction.

Logistic Regression: Performs the task to predict values based on the given data.

In [1]:
```
import numpy as np # linear algebra
import pandas as pd # data processing

# import libraries for plotting
import matplotlib.pyplot as plt
import seaborn as sns
%matplotlib inline

import warnings
warnings.filterwarnings("ignore")
```

In [68]:
```
# train a logistic regression model on the training set
from sklearn.linear_model import LogisticRegression
```

- **LOAD THE DATASET:**

Import the data set using pandas library, which contains all the variables.

In [2]:
```
df=pd.read_csv(r"C:\Users\Pranathi\Downloads\weather\weatherAUS.csv")
df
```

- **EXPLORATORY DATA ANALYSIS:**

Preview the dataset:

In [3]:
```
df.head()
```

Out[3]:

	Date	Location	MinTemp	MaxTemp	Rainfall	Evaporation	Sunshine	WindGustDir	WindGustSpeed	WindDir9am	...	Humidity9am	Humidity3pm	Pressure
0	2008-12-01	Albury	13.4	22.9	0.6	NaN	NaN	W	44.0	W	...	71.0	22.0	10
1	2008-12-02	Albury	7.4	25.1	0.0	NaN	NaN	WNW	44.0	NNW	...	44.0	25.0	10
2	2008-12-03	Albury	12.9	25.7	0.0	NaN	NaN	WSW	46.0	W	...	38.0	30.0	10
3	2008-12-04	Albury	9.2	28.0	0.0	NaN	NaN	NE	24.0	SE	...	45.0	16.0	10
4	2008-12-05	Albury	17.5	32.3	1.0	NaN	NaN	W	41.0	ENE	...	82.0	33.0	10

5 rows × 23 columns

View dimensions of dataset:

```
In [4]: df.shape
```

```
Out[4]: (145460, 23)
```

View column names:

```
In [5]: col_names=df.columns
        col_names
```

```
Out[5]: Index(['Date', 'Location', 'MinTemp', 'MaxTemp', 'Rainfall', 'Evaporation',
               'Sunshine', 'WindGustDir', 'WindGustSpeed', 'WindDir9am', 'WindDir3pm',
               'WindSpeed9am', 'WindSpeed3pm', 'Humidity9am', 'Humidity3pm',
               'Pressure9am', 'Pressure3pm', 'Cloud9am', 'Cloud3pm', 'Temp9am',
               'Temp3pm', 'RainToday', 'RainTomorrow'],
              dtype='object')
```

Checking For datatypes of the attributes:

```
In [6]: df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 145460 entries, 0 to 145459
Data columns (total 23 columns):
 #   Column         Non-Null Count   Dtype  
---  ------         --------------   -----  
 0   Date           145460 non-null  object 
 1   Location       145460 non-null  object 
 2   MinTemp        143975 non-null  float64
 3   MaxTemp        144199 non-null  float64
 4   Rainfall       142199 non-null  float64
 5   Evaporation    82670 non-null   float64
 6   Sunshine       75625 non-null   float64
 7   WindGustDir    135134 non-null  object 
 8   WindGustSpeed  135197 non-null  float64
 9   WindDir9am     134894 non-null  object 
 10  WindDir3pm     141232 non-null  object 
 11  WindSpeed9am   143693 non-null  float64
 12  WindSpeed3pm   142398 non-null  float64
 13  Humidity9am    142806 non-null  float64
 14  Humidity3pm    140953 non-null  float64
 15  Pressure9am    130395 non-null  float64
 16  Pressure3pm    130432 non-null  float64
 17  Cloud9am       89572 non-null   float64
 18  Cloud3pm       86102 non-null   float64
```

we can see that that the dataset contains mixture of categorical and numerical variables.

categorical variable have data type: float64

Numerical variable have data type: object

Also, there are missing values in data set, we are gonna explore it later.

View statistical properties of dataset:

```
In [7]: df.describe()
```

- **Univariate Analysis:**

Check for missing values:

```
In [8]: df['RainTomorrow'].isnull().sum()
```

```
Out[8]: 3267
```

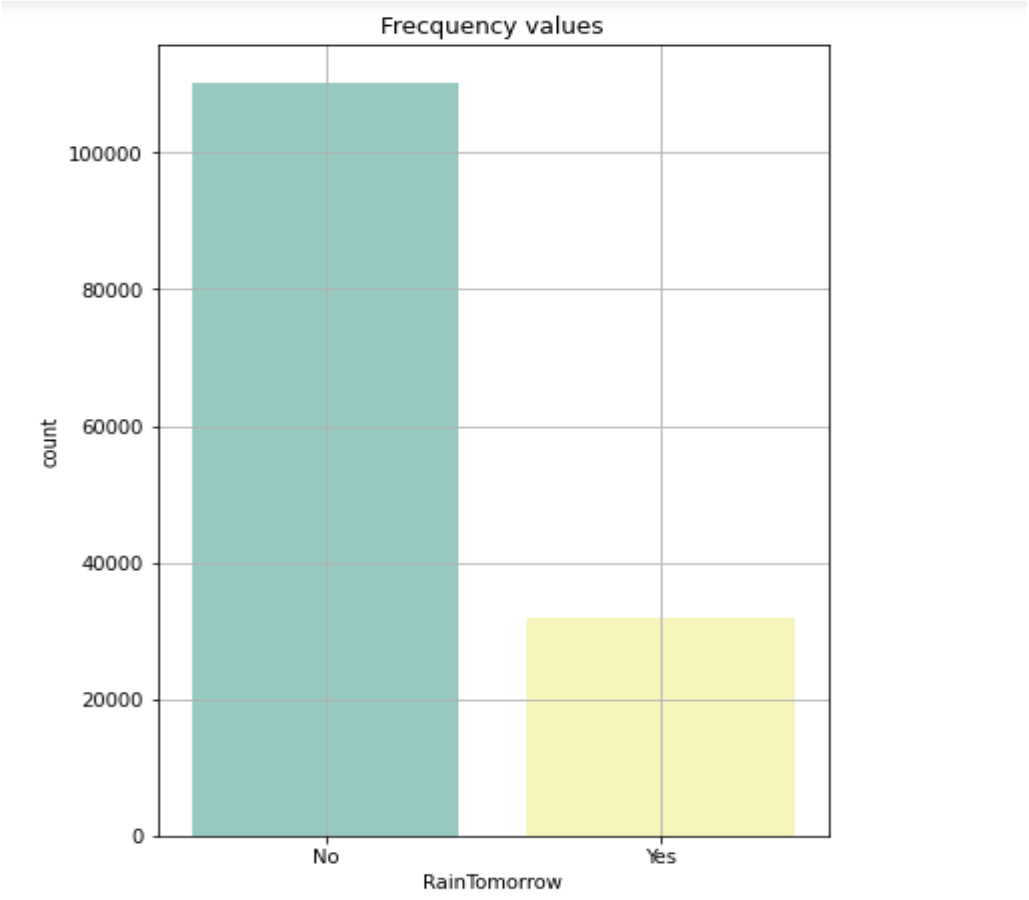
Check for the unique values:

```
In [9]: df['RainTomorrow'].unique()
```

```
Out[9]: array(['No', 'Yes', nan], dtype=object)
```

Visualize frequency distribution of Rain Tomorrow variable:

```
In [13]: plt.ax =plt.subplots(figsize=(6,8))
         ax=sns.countplot(x='RainTomorrow',data=df,palette="Set3")
         #plt.xticks(rotation=90)
         plt.grid()
         plt.title('Frecquency values');
```

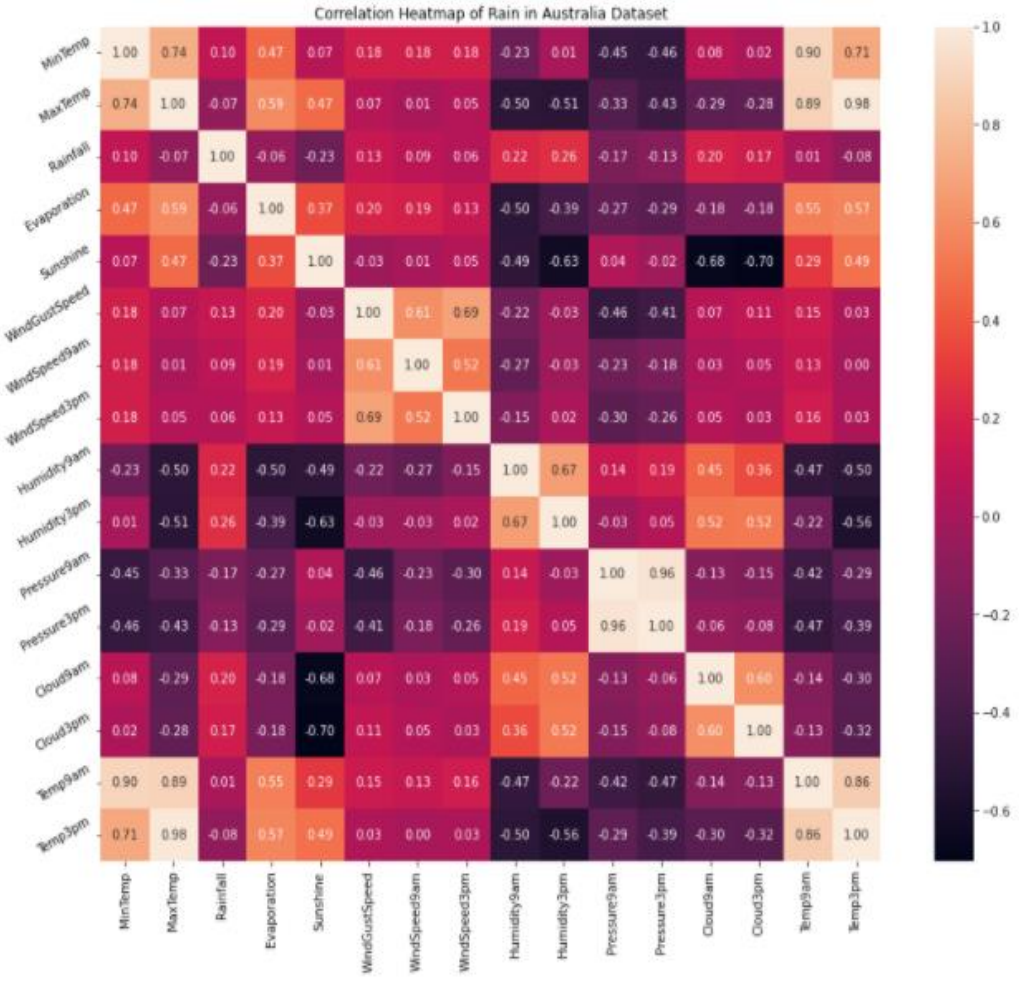


3.4 Frequency values

- **MULTIVARIATE ANALYSIS:**

Correlation Matrix:

```
In [41]: correlation = df.corr()

In [42]: plt.figure(figsize=(16,12))
         plt.title('Correlation Heatmap of Rain in Australia Dataset')
         ax = sns.heatmap(correlation, square=True, annot=True, fmt='.2f', linecolor='white')
         ax.set_xticklabels(ax.get_xticklabels(), rotation=90)
         ax.set_yticklabels(ax.get_yticklabels(), rotation=30)
         plt.show()
```

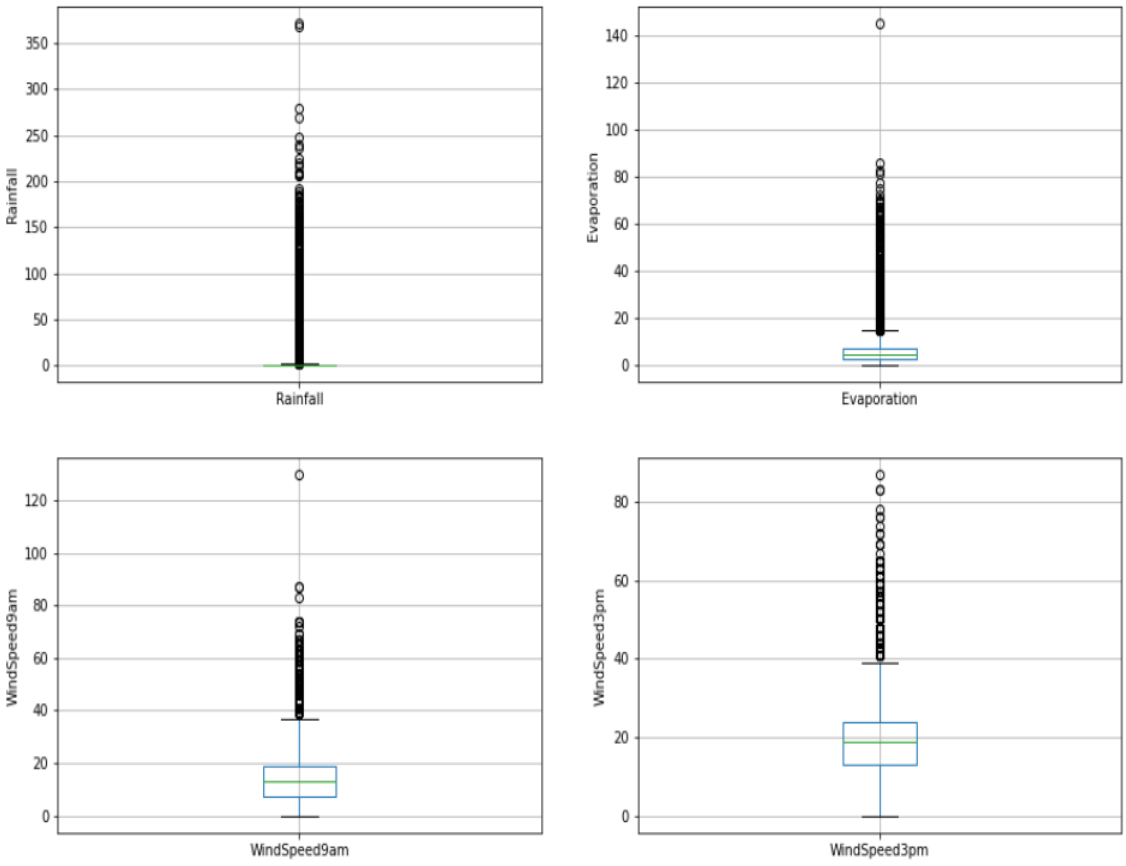


3.5 Heat map

- From the above correlation heat map, we can conclude that:-
- Min Temp and Max Temp variables are highly positively correlated (correlation coefficient = 0.74).
- Min Temp and Temp3pm variables are also highly positively correlated (correlation coefficient = 0.71).

- Min Temp and Temp9am variables are strongly positively correlated (correlation coefficient = 0.90).
- Max Temp and Temp9am variables are strongly positively correlated (correlation coefficient = 0.89).
- Max Temp and Temp3pm variables are also strongly positively correlated (correlation coefficient = 0.98).
- Wind Gust Speed and WindSpeed3pm variables are highly positively correlated (correlation coefficient = 0.69).
- Pressure9am and Pressure3pm variables are strongly positively correlated (correlation coefficient = 0.96).
- Temp9am and Temp3pm variables are strongly positively correlated (correlation coefficient = 0.86).

- **DATA MODELING:**



3.6 Data visualization graphs

Declare feature vector and target variable:

```
In [45]: X = df.drop(['RainTomorrow'], axis=1)

         y = df['RainTomorrow']
```

Split data into separate training and test set:

Split the dataset into two sets i.e. the training set and the testing set. Training set consists of 80% of the dataset and the testing set has 20% of the dataset. The columns with only two distinct values and wanted to make sure that the splitting should split these values in equal proportions. Therefore, use a stratified shuffle shift for train test splitting for better results.

```
In [46]: # split X and y into training and testing sets

         from sklearn.model_selection import train_test_split

         X_train, X_test, y_train, y_test = train_test_split(X, y, test_size = 0.2, random_state = 0)
```

```
In [47]: X_train.shape, X_test.shape

Out[47]: ((116368, 22), (29092, 22))
```

Using the train test split method, the dataset is divided into train sets and test sets. Train the model using a train set and for testing we use test data.

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

Divide the total dataset into three subsets:

- Training data is used for learning the parameters of the model.
- Test data is used to evaluate the fit machine learning mode.

This splitting can prevent the model from over-fitting and to accurately evaluate our model.

The random state parameter is used for initializing the internal random number generator, which will decide the splitting of data into train and test. Setting random_ state a fixed value will guarantee that the same sequence of random numbers is generated each time.

- **FIT THE LOGISTIC REGRESSION MODEL:**

Using Logistic regression module, we fit our train dataset to our ML model.

```
# instantiate the model
logreg = LogisticRegression(solver='liblinear', random_state=0)


# fit the model
logreg.fit(X_train, y_train)
```

```
Out[68]: LogisticRegression(random_state=0, solver='liblinear')
```

- **PREDICT THE OUTPUT:**

Finally, we predict using Logreg.predict() method.

```
In [69]: y_pred_test = logreg.predict(X_test)

         y_pred_test
```

```
Out[69]: array(['No', 'No', 'No', ..., 'Yes', 'No', 'No'], dtype=object)
```

```
In [70]: # probability of getting output as 0 - no rain

         logreg.predict_proba(X_test)[:,0]
```

```
Out[70]: array([0.81049725, 0.74507807, 0.79642875, ..., 0.44923287, 0.6459017 ,
                0.96767155])
```

```
In [71]:  #probability of getting output as 1 - rain

          logreg.predict_proba(X_test)[:,1]
```

```
Out[71]: array([0.18950275, 0.25492193, 0.20357125, ..., 0.55076713, 0.3540983 ,
                0.03232845])
```

- **APPLY METRICS AND GET ACCURACY SCORE:**

```
In [72]: from sklearn.metrics import accuracy_score

         print('Model accuracy score: {0:0.4f}'. format(accuracy_score(y_test, y_pred_test)))
```

```
Model accuracy score: 0.8476
```

CHAPTER 4

DISCUSSION AND RESULTS, PERFORMANCE ANALYSIS

4.1 DISCUSSION AND RESULT:

Climate change is always a major issue for whole world and making any prediction on that is now days pretty difficult and unpredictable. Climate change is due to the current global warming trend is human expansion. Due to this air and oceans are warming, sea level is rising and flooding and drought etc. One of the serious consequences due to this climate change is on Rainfall. Rainfall prediction now days is an arduous task which is taking into the consideration of most of the major world-wide authorities.

In this project core motive is to finding out the algorithm which gives us the good prediction of rainfall. Here we took the rainfall data of Australia from the Kaggle. Below is the information of the accuracy of the algorithms.

```
In [74]: from sklearn.ensemble import AdaBoostClassifier

         rf = AdaBoostClassifier()
         rf.fit(X_train, y_train)
         y_pred_test=rf.predict(X_test)
         print('Model accuracy score: {0:0.4f}'. format(accuracy_score(y_test, y_pred_test)))

         Model accuracy score: 0.8431
```

```
In [72]: from sklearn.metrics import accuracy_score

         print('Model accuracy score: {0:0.4f}'. format(accuracy_score(y_test, y_pred_test)))

         Model accuracy score: 0.8476
```

Here we can see that the Logistic Regression comes up as the best algorithm for the rainfall prediction.

4.2 PERFORMANCE ANALYSIS

By comparing the predicted value and provided value one can identify how exactly Is machine learning working. And can also plot that difference between the test value and predicted value using seaborn.

In statistics, the actual value is the value that is obtained by observation or by measuring the available data. It is also called the observed value. The predicted value is the value of the variable predicted based on the regression analysis.

In [75]:
```
from sklearn.metrics import confusion_matrix

cm=confusion_matrix(y_test, y_pred_test)

print('Confusion matrix\n\n', cm)

print('\nTrue Positives(TP) = ', cm[0,0])

print('\nTrue Negatives(TN) = ', cm[1,1])

print('\nFalse Positives(FP) = ', cm[0,1])

print('\nFalse Negatives(FN) = ', cm[1,0])
```

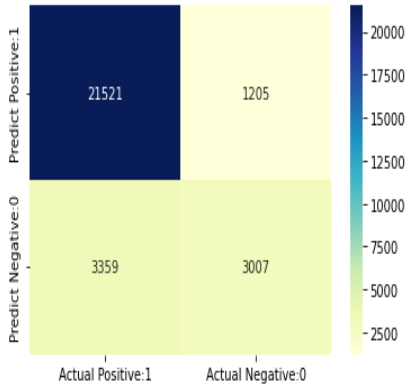
```
Confusion matrix

 [[21521  1205]
 [ 3359  3007]]

True Positives(TP) =  21521

True Negatives(TN) =  3007

False Positives(FP) =  1205

False Negatives(FN) =  3359
```

In [76]:
```
cm_matrix = pd.DataFrame(data=cm, columns=['Actual Positive:1', 'Actual Negative:0'],
                                  index=['Predict Positive:1', 'Predict Negative:0'])
cm_matrix.head()
```

Out[76]:

	Actual Positive:1	Actual Negative:0
Predict Positive:1	21521	1205
Predict Negative:0	3359	3007

In [77]:
```
sns.heatmap(cm_matrix, annot=True, fmt='d', cmap='YlGnBu');
```



CHAPTER 5
SUMMARY AND CONCLUSION

This Project has presented a supervised rainfall learning model which used machine learning algorithms to classify rainfall data. We used different machine learning algorithm to check the accuracy of rainfall prediction. We have compared AdaBoostClassifier, Random Forest, Navie Bayes and Logistic Regression classifiers. From the above we can conclude that Logistic Regression is the Machine learning algorithm which is suitable for rainfall prediction in Australia.

Currently machine learning used in number of 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 it’s pretty helpful with good accuracy score and in rainfall also its gives pretty good predictions. Future plan is to increase the work in Storm predictions and Crop prediction with the rainfall prediction.

REFERENCES

1) https://ukdiss.com/examples/rainfall-prediction-machine-learning.php
2) https://www.kaggle.com/code/ziadshamndy/rain-prediction-analysis-with-85-45-accuracy/notebook
3) https://github.com/Vasanthengineer4949/Rain-Prediction
4) https://www.google.com/search?q=python+modules&rlz=1C1CHBF_enIN914IN914&oq=python+modules&aqs=chrome..69i57j0i433i512j0i512l4j0i10i512j0i512l3.7884j0j7&sourceid=chrome&ie=UTF-8

APPENDIX

SOURCE CODE:

```
import numpy as np # linear algebra
import pandas as pd # data processing
# import libraries for plotting
import matplotlib.pyplot as plt
import seaborn as sns
%matplotlib inline
import warnings
warnings.filterwarnings("ignore")
df=pd.read_csv(r"C:\Users\Pranathi\Downloads\weather\weatherAUS.csv")
df
df.head()
df.shape
col_names=df.columns
col_names
df.info()
df.describe()
df['RainTomorrow'].isnull().sum()
df['RainTomorrow'].unique()
df['RainTomorrow'].value_counts()
RainTomorrow={"Yes":31877,
            'No':110316,
            'Missing values':3267}
print ('The precentage is :')
for key,value in RainTomorrow.items():
    print(key ,':', value/len(df))
plt.ax =plt.subplots(figsize=(6,8))
ax=sns.countplot(x='RainTomorrow',data=df,palette="Set3")
#plt.xticks(rotation=90)
plt.grid()
plt.title('Frecquency values');
categorical= df.select_dtypes(include=['object'])
```

```
categorical.head()
 dict={}
for i in list(df[categorical.columns]):
    dict[i]=df[i].isnull().sum()
pd.DataFrame(dict,index=['number of null values']).transpose()
for var in categorical:
    print(var, ' contains ', len(df[var].unique()), ' labels')
df['Date'] = pd.to_datetime(df['Date'])
df['Year'] = df['Date'].dt.year
df['Month'] = df['Date'].dt.month
df['Day'] = df['Date'].dt.day
df.drop('Date',inplace= True,axis=1)
new_categorical= df.select_dtypes(include=['object'])
new_categorical.head()
new_categorical.isnull().sum()
new_categorical['Location'].unique()
pd.get_dummies(df.Location, drop_first=True).head()
new_categorical['WindGustDir'].unique()
pd.get_dummies(df.WindGustDir,drop_first=True,dummy_na=True).head()
new_categorical['WindDir9am'].unique()
pd.get_dummies(df.WindDir9am,drop_first=True,dummy_na=True).head()
new_categorical['WindDir3pm'].unique()
pd.get_dummies(df.WindDir3pm,drop_first=True,dummy_na=True).head()
new_categorical['RainToday'].unique()
df.RainToday.value_counts()
pd.get_dummies(df.RainToday,drop_first=True,dummy_na=True).head()
Numerical= df.select_dtypes(include=['float64','int'])
Numerical.head()
Numerical.columns
Numerical.isnull().sum()
Numerical.describe()
plt.figure(figsize=(15,10))
plt.subplot(2, 2, 1)
```

```
fig = df.boxplot(column='Rainfall')
 fig.set_title('')
fig.set_ylabel('Rainfall')
plt.subplot(2, 2, 2)
fig = df.boxplot(column='Evaporation')
fig.set_title('')
fig.set_ylabel('Evaporation')
plt.subplot(2, 2, 3)
fig = df.boxplot(column='WindSpeed9am')
fig.set_title('')
fig.set_ylabel('WindSpeed9am')
plt.subplot(2, 2, 4)
fig = df.boxplot(column='WindSpeed3pm')
fig.set_title('')
fig.set_ylabel('WindSpeed3pm')
# find outliers for Rainfall variable
IQR = df.Rainfall.quantile(0.75) - df.Rainfall.quantile(0.25)
Lower_fence = df.Rainfall.quantile(0.25) - (IQR * 1.5)
Upper_fence = df.Rainfall.quantile(0.75) + (IQR * 1.5)
print('Rainfall outliers are values < {lowerboundary} or > 
{upperboundary}'.format(lowerboundary=Lower_fence, 
upperboundary=Upper_fence))
# find outliers for Evaporation variable
IQR = df.Evaporation.quantile(0.75) - df.Evaporation.quantile(0.25)
Lower_fence = df.Evaporation.quantile(0.25) - (IQR * 1.5)
Upper_fence = df.Evaporation.quantile(0.75) + (IQR * 1.5)
print('Evaporation outliers are values < {lowerboundary} or > 
{upperboundary}'.format(lowerboundary=Lower_fence, 
upperboundary=Upper_fence))
# find outliers for WindSpeed9am variable
IQR = df.WindSpeed9am.quantile(0.75) - df.WindSpeed9am.quantile(0.25)
Lower_fence = df.WindSpeed9am.quantile(0.25) - (IQR * 1.5)
Upper_fence = df.WindSpeed9am.quantile(0.75) + (IQR * 1.5)
```

```
print('WindSpeed9am outliers are values < {lowerboundary} or > {upperboundary}'.format(lowerboundary=Lower_fence, upperboundary=Upper_fence))
# find outliers for WindSpeed3pm variable
IQR = df.WindSpeed3pm.quantile(0.75) - df.WindSpeed3pm.quantile(0.25)
Lower_fence = df.WindSpeed3pm.quantile(0.25) - (IQR * 1.5)
Upper_fence = df.WindSpeed3pm.quantile(0.75) + (IQR * 1.5)
print('WindSpeed3pm outliers are values < {lowerboundary} or > {upperboundary}'.format(lowerboundary=Lower_fence, upperboundary=Upper_fence))
correlation = df.corr()
plt.figure(figsize=(16,12))
plt.title('Correlation Heatmap of Rain in Australia Dataset')
ax = sns.heatmap(correlation, square=True, annot=True, fmt='.2f', linecolor='white')
ax.set_xticklabels(ax.get_xticklabels(), rotation=90)
ax.set_yticklabels(ax.get_yticklabels(), rotation=30)
plt.show()
X = df.drop(['RainTomorrow'], axis=1)
y = df['RainTomorrow']
# split X and y into training and testing sets
from sklearn.model_selection import train_test_split
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size = 0.2, random_state = 0)
X_train.shape, X_test.shape
# display categorical variables
categorical = [col for col in X_train.columns if X_train[col].dtypes == 'O']
categorical
# display numerical variables
numerical = [col for col in X_train.columns if X_train[col].dtypes != 'O']
numerical
# check missing values in numerical variables in X_train
X_train[numerical].isnull().sum()
```

```
# check missing values in numerical variables in X_test
X_test[numerical].isnull().sum()
# impute missing values in X_train and X_test with respective column median in X_train
for df1 in [X_train, X_test]:
    for col in numerical:
        col_median=X_train[col].median()
        df1[col].fillna(col_median, inplace=True)
# check again missing values in numerical variables in X_train
X_train[numerical].isnull().sum()
# impute missing categorical variables with most frequent value
for df2 in [X_train, X_test]:
    df2['WindGustDir'].fillna(X_train['WindGustDir'].mode()[0], inplace=True)
    df2['WindDir9am'].fillna(X_train['WindDir9am'].mode()[0], inplace=True)
    df2['WindDir3pm'].fillna(X_train['WindDir3pm'].mode()[0], inplace=True)
    df2['RainToday'].fillna(X_train['RainToday'].mode()[0], inplace=True)
y_train.fillna(y_train.mode()[0], inplace=True)
y_test.fillna(y_test.mode()[0], inplace=True)
# check missing values in categorical variables in X_train
X_train[categorical].isnull().sum()
X_train.isnull().sum()
import numpy as np
def max_value(df3, variable, top):
    return np.where(df3[variable]>top, top, df3[variable])
    X_train['Rainfall'] = max_value(X_train, 'Rainfall', 2)
    X_train['Evaporation'] = max_value(X_train, 'Evaporation', 14.6)
    X_train['WindSpeed9am'] = max_value(X_train, 'WindSpeed9am', 37)
    X_train['WindSpeed3pm'] = max_value(X_train, 'WindSpeed3pm', 40.5)
    X_test['Rainfall'] = max_value(X_test, 'Rainfall', 2)
    X_test['Evaporation'] = max_value(X_test, 'Evaporation', 14.6)
    X_test['WindSpeed9am'] = max_value(X_test, 'WindSpeed9am', 37)
    X_test['WindSpeed3pm'] = max_value(X_test, 'WindSpeed3pm', 40.5)
```

```
X_train[categorical].head()
import category_encoders as ce
encoder = ce.BinaryEncoder(cols=['RainToday'])
X_train = encoder.fit_transform(X_train)
X_test = encoder.transform(X_test)
X_train.head()
X_train = pd.concat([X_train[numerical], X_train[['RainToday_0', 'RainToday_1']],
                     pd.get_dummies(X_train.Location),
                     pd.get_dummies(X_train.WindGustDir),
                     pd.get_dummies(X_train.WindDir9am),
                     pd.get_dummies(X_train.WindDir3pm)], axis=1)
X_train.head()
X_test = pd.concat([X_test[numerical], X_test[['RainToday_0', 'RainToday_1']],
                     pd.get_dummies(X_test.Location),
                     pd.get_dummies(X_test.WindGustDir),
                     pd.get_dummies(X_test.WindDir9am),
                     pd.get_dummies(X_test.WindDir3pm)], axis=1)
X_train.describe()
cols = X_train.columns
from sklearn.preprocessing import MinMaxScaler
scaler = MinMaxScaler()
X_train = scaler.fit_transform(X_train)
X_test = scaler.transform(X_test)
X_train = pd.DataFrame(X_train, columns=[cols])
X_test = pd.DataFrame(X_test, columns=[cols])
X_train.describe()
# train a logistic regression model on the training set
from sklearn.linear_model import LogisticRegression
# instantiate the model
logreg = LogisticRegression(solver='liblinear', random_state=0)
# fit the model
logreg.fit(X_train, y_train)
```

```
y_pred_test = logreg.predict(X_test)
y_pred_test
# probability of getting output as 0 - no rain
logreg.predict_proba(X_test)[:,0]
#probability of getting output as 1 - rain
logreg.predict_proba(X_test)[:,1]
from sklearn.metrics import accuracy_score
print('Model accuracy score: {0:0.4f}'. format(accuracy_score(y_test, y_pred_test)))
from sklearn.model_selection import GridSearchCV
from sklearn.ensemble import RandomForestClassifier
#rf_params = {'n_estimators':[100,150,200],'criterion':['gini','entropy'],}
rf = RandomForestClassifier()
rf.fit(X_train, y_train)
y_pred_test=rf.predict(X_test)
print('Model accuracy score: {0:0.4f}'. format(accuracy_score(y_test, y_pred_test)))
from sklearn.ensemble import AdaBoostClassifier
rf = AdaBoostClassifier()
rf.fit(X_train, y_train)
y_pred_test=rf.predict(X_test)
print('Model accuracy score: {0:0.4f}'. format(accuracy_score(y_test, y_pred_test)))
from sklearn.metrics import confusion_matrix
cm=confusion_matrix(y_test, y_pred_test)
print('Confusion matrix\n\n', cm)
print('\nTrue Positives(TP) = ', cm[0,0])
print('\nTrue Negatives(TN) = ', cm[1,1])
print('\nFalse Positives(FP) = ', cm[0,1])
print('\nFalse Negatives(FN) = ', cm[1,0])
cm_matrix = pd.DataFrame(data=cm, columns=['Actual Positive:1', 'Actual
Negative:0'],
                         index=['Predict Positive:1', 'Predict Negative:0'])
cm_matrix.head()
sns.heatmap(cm_matrix, annot=True, fmt='d', cmap='YlGnBu');
```