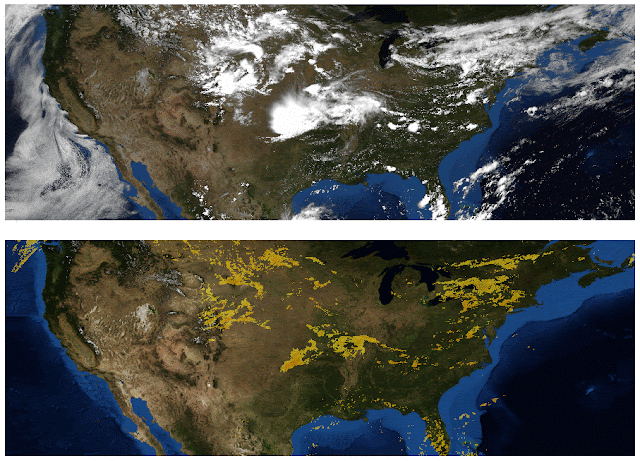
# *A Small Blog On*

# *Rainfall Prediction with Machine Learning*



***By***

***KOUSHIK BISWAS***

***(Data Science Student of***

***“Data Trained”)***

Date:-18/06/22

***Introduction***

Rainfall Prediction is one of the difficult and uncertain tasks that have a significant impact on human society. it’s applications in the various sectors such as in agriculture, utility company and in day to day life. Timely and accurate forecasting can proactively help reduce human and financial loss. This study presents a set of experiments that involve the use of common machine learning techniques to create models that can predict whether it will rain tomorrow or not based on the weather data for that day in major cities in Australia.

***TABLE OF CONTENTS***

* Problem Statement
* Import Data set
* About The Dataset
* Tools Used
* Importing Necessary Libraries
* Data Analysis
* EDA
* Preprocessing
* Bulding Machine Learning Models
* Conclussion

***Problem Statement:-***

Climate is a important aspect of human life. So, the Prediction should accurate as much as possible. In this paper we try to deal with the prediction of the rainfall which is also a major aspect of human life and which provide the major resource of human life which is Fresh Water. Fresh water is always a crucial resource of human survival – not only for the drinking purposes but also for farming, washing and many other purposes.

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. Peoples 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.

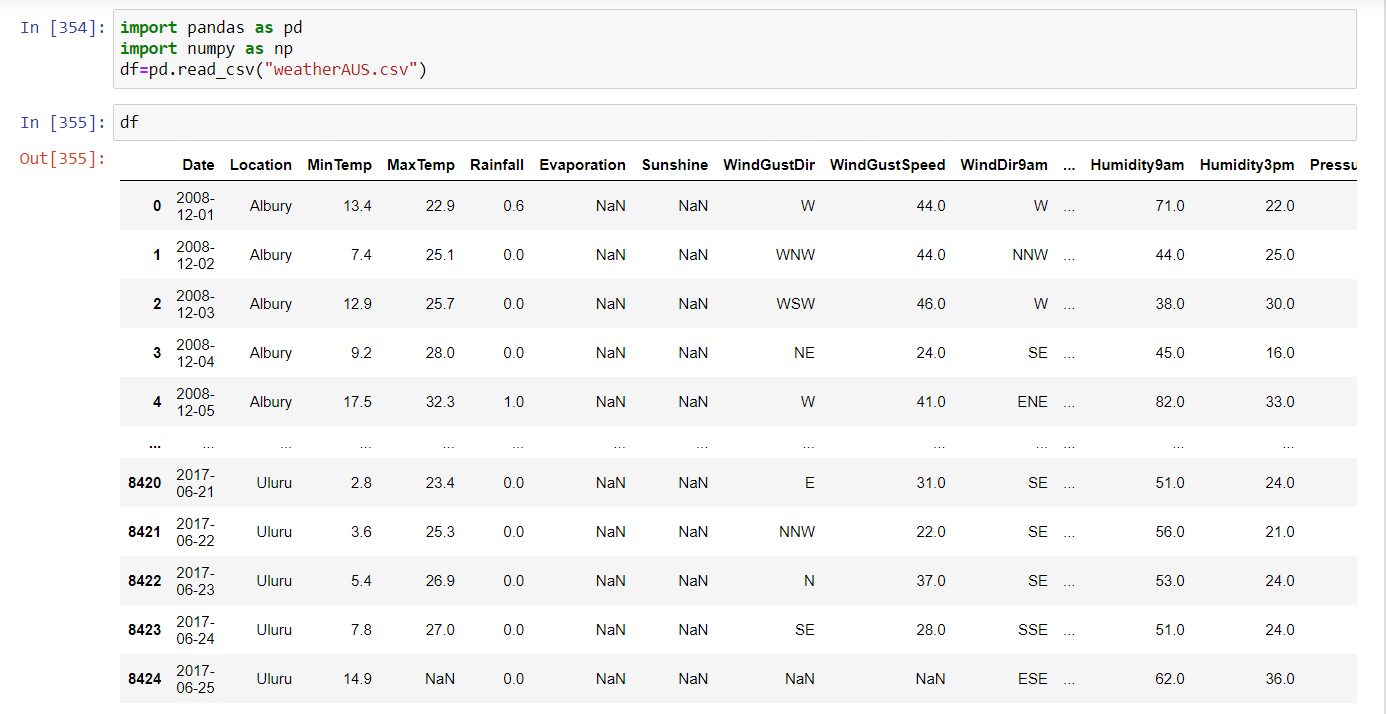
Due to climate change it become difficult for farmer to grow crops, raise animals and catch fish in the same ways and same places as they have done in the past. This also effects the agricultural production. Farmer as the main source of the food for the human but due to this its also get affected. Where as air is also become poisonous there are several no of harmful gases are mixed up in the air which effecting the humans. More people are becoming ill due to this air pollution as for humans it is necessary to breath.

And water which is also an important resource of survival of humans. But due to this climate change availability of the fresh water is decreasing rapidly. No of countries ae facing the shortage of fresh water to drink. This climate changes are not just changing the temperature. The whole water cycle is also get affected. Warmer the world becomes means the atmosphere ha the capacity to hold grater moisture. So, there are changes in the volume of water vapour, rainfall and the flow of water in the atmosphere.

Fresh water is always a crucial resource of human survival – not only for the drinking purposes but also for farming, washing and many other purposes.  It is expected to become increasingly scarce in future, and this partly due to climate change.

The scope of this research is wide. Currently peoples are facing major problem due to this climate change. If we able to make a good prediction of weather this will very helpful for the whole mankind human kind.

***Import The Dataset:-***

Let’s start this task of rainfall prediction by importing the data.

I have imported the dataset which was in the csv file using pandas. The dataset contains 8425 rows and 23 columns having both numerical and categorical data. Here I can make use of PCA to reduce the columns, but this will give huge data loss. To avoid this, I am keeping the dataset as it is.

In this dataset " **RainTomorrow** " is our target variable which has two classes. So this is a "Classification type" problem.

you can download the dataset from the below link.

**Dataset link:** https://raw.githubusercontent.com/dsrscientist/dataset3/main/weatherAUS.csv

***About The Dataset:-***

**Number of columns**: 23

**Date** - The date of observation

**Location** -The common name of the location of the weather station

**MinTemp** -The minimum temperature in degrees celsius

**MaxTemp** -The maximum temperature in degrees celsius

**Rainfall** -The amount of rainfall recorded for the day in mm

**Evaporation** -The so-called Class A pan evaporation (mm) in the 24 hours to 9am

**Sunshine** -The number of hours of bright sunshine in the day.

**WindGustDi** r- The direction of the strongest wind gust in the 24 hours to midnight

**WindGustSpeed** -The speed (km/h) of the strongest wind gust in the 24 hours to midnight

**WindDir9am** -Direction of the wind at 9am

**WindDir3pm** -Direction of the wind at 3pm

**WindSpeed9am** -Wind speed (km/hr) averaged over 10 minutes prior to 9am

**WindSpeed3pm** -Wind speed (km/hr) averaged over 10 minutes prior to 3pm

**Humidity9am** -Humidity (percent) at 9am

**Humidity3pm** -Humidity (percent) at 3pm

**Pressure9am** -Atmospheric pressure (hpa) reduced to mean sea level at 9am

**Pressure3pm** -Atmospheric pressure (hpa) reduced to mean sea level at 3pm

**Cloud9am** - Fraction of sky obscured by cloud at 9am.

**Cloud3pm** -Fraction of sky obscured by cloud

**Temp9am**-Temperature (degrees C) at 9am

**Temp3pm** -Temperature (degrees C) at 3pm

**RainToday** -Boolean: 1 if precipitation (mm) in the 24 hours to 9am exceeds 1mm, otherwise 0

**RainTomorrow** -The amount of next day rain in mm. Used to create response variable . A kind of measure of the "risk".

***Tools Used:-***

* Python 3.8
* NumPy
* Pandas
* Matplotlib
* Seaborn
* Data science
* Machine learning

***Importing Necessary Libraries:-***

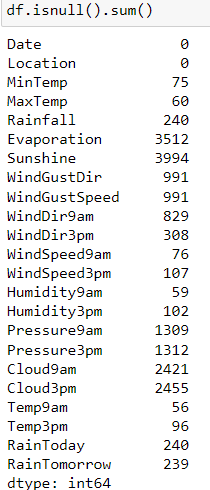
import numpy as np   
import pandas as pd  
import matplotlib.pyplot as plt  
import seaborn as sns  
from sklearn.preprocessing import LabelEncoder  
from sklearn.preprocessing import PowerTransformer  
from sklearn.preprocessing import StandardScaler  
from scipy.stats import zscore  
from matplotlib import rcParams  
from sklearn.metrics import plot\_confusion\_matrix  
from sklearn.naive\_bayes import GaussianNB  
from sklearn import linear\_model  
from sklearn.metrics import accuracy\_score  
from sklearn.metrics import f1\_score  
from sklearn.metrics import roc\_curve, auc  
from sklearn.model\_selection import train\_test\_split  
from sklearn.tree import DecisionTreeClassifier  
from sklearn.ensemble import RandomForestClassifier  
from sklearn.svm import SVC  
from sklearn.ensemble import AdaBoostClassifier  
from sklearn.ensemble import GradientBoostingClassifier  
from sklearn.neighbors import KNeighborsClassifier  
from sklearn.linear\_model import LogisticRegression  
from sklearn.model\_selection import cross\_val\_score  
from sklearn.model\_selection import cross\_val\_predict  
from sklearn.metrics import accuracy\_score, confusion\_matrix, classification\_report  
from sklearn.metrics import precision\_score , accuracy\_score , recall\_score  
from sklearn.model\_selection import GridSearchCV  
from sklearn.metrics import roc\_curve  
import warnings   
warnings.filterwarnings('ignore')

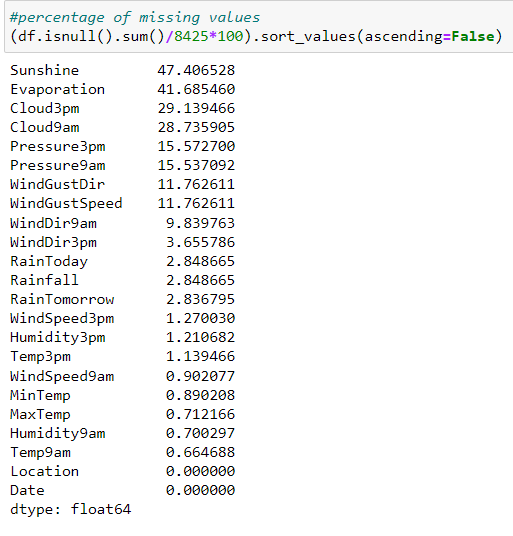
***Data Analysis:-***

In this part I will check

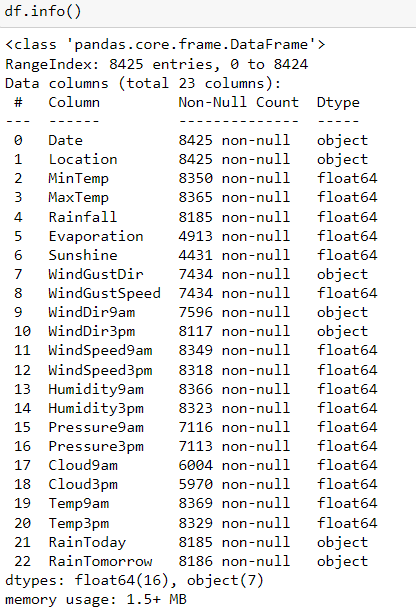
* Shape of the Dataset
* Null Values
* Information
* Description Of The Dataset(mean,std,min,max)
* *Shape of the Dataset*

our dataframe has 8425 rows and 23 columns.It means we have 8425 Day's 23 type of data.

* *Null Values*

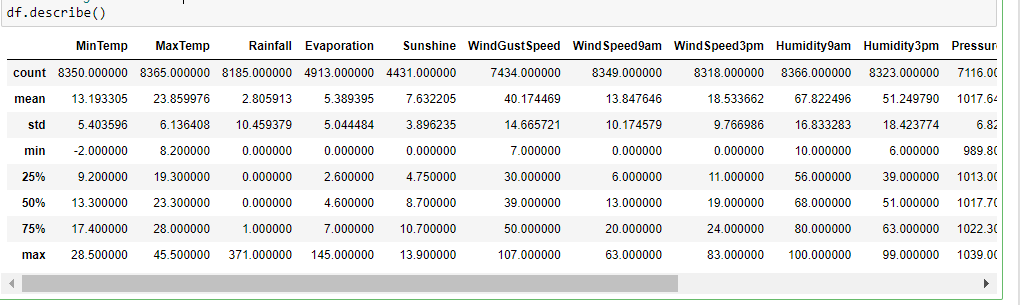


We can clearly see from the dataset that apart from the date and location we have null values in almost all of the features and also the target values.We have to hacdle it

* Information

Two type of value present in our Data set  
-object  
-float

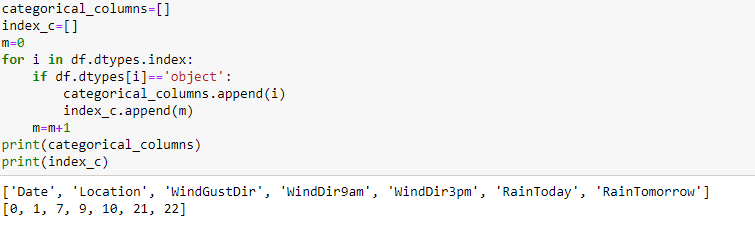
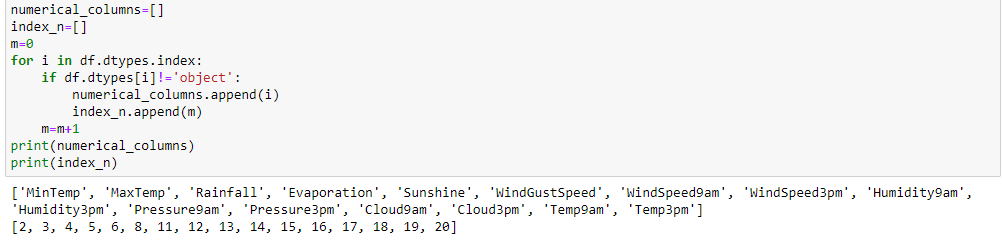
* Description Of The Dataset(mean,std,min,max)



This gives the statistical information of the dataset. There are no negative values and invalid values are present. This gives the summary of numerical data.

***EDA:-***

EDA or Exploratory Data Analysis is **an expression utilized for data science in which the focus is to understand insights of the data through Visualization or by Statistical Analysis**. The steps involved are - Firstly we focus on variable identification - we identify the data type and category of variables.

* **For categoriacal columns:**
* **For numerical columns**:

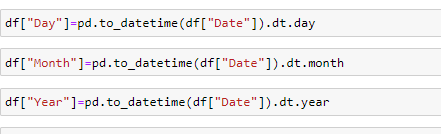
I have seprated our columns as per their type they are having. Means I have created two types of list namely categorical\_column and numerical\_column. I have kept all object type data into categorical\_column list and all numerical type data into numerical\_column.We can see that length of categorical column is less than length of numerical column.

***Univariate Analysis Of Categorical Columns***

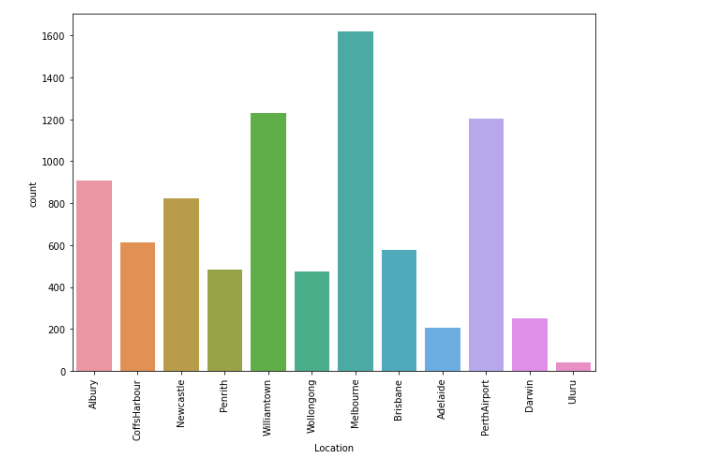
* **Date:**



We can see that weather reports of many places have been stored in one day. We separate the column into three columns, day, month and year for better prediction by the code below and drop the Date column.

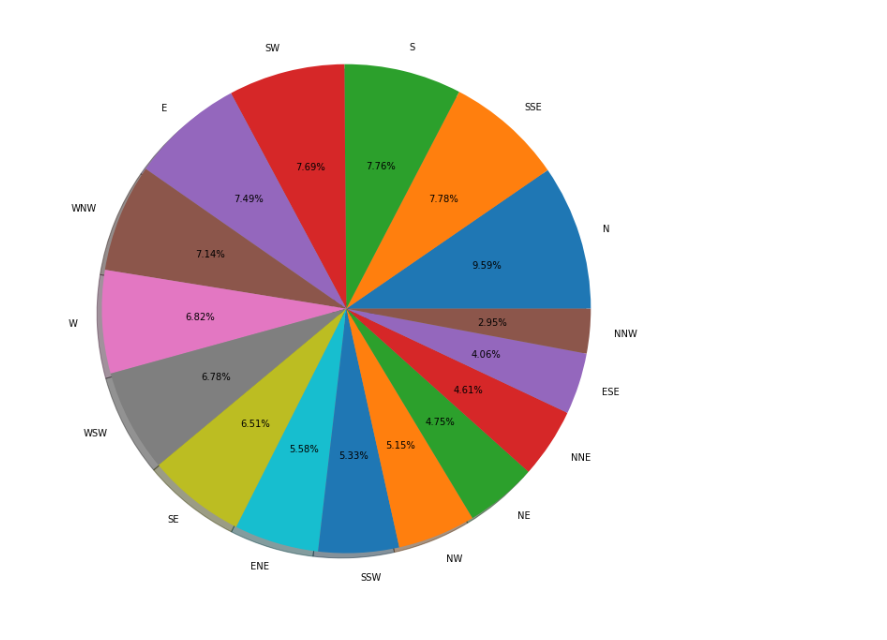


* **Location:**

***Count Plot of Location Column***

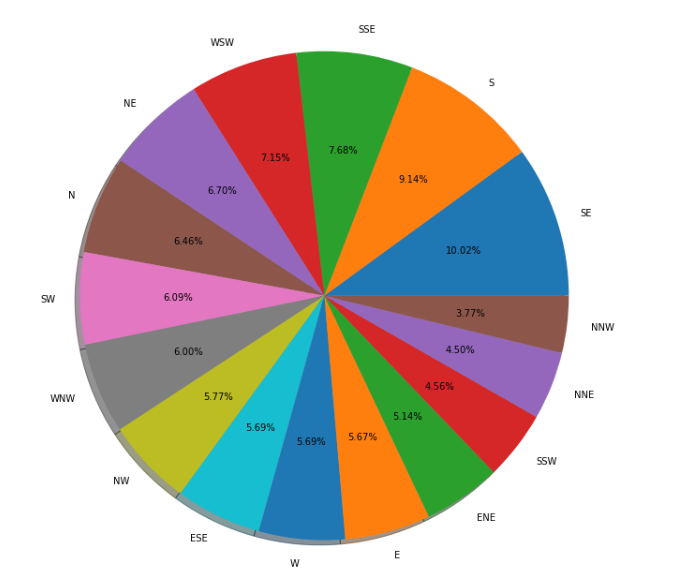
12 location's weather report present in our data.We have the higheste rainfall data from Melbourbne and least from Uluru

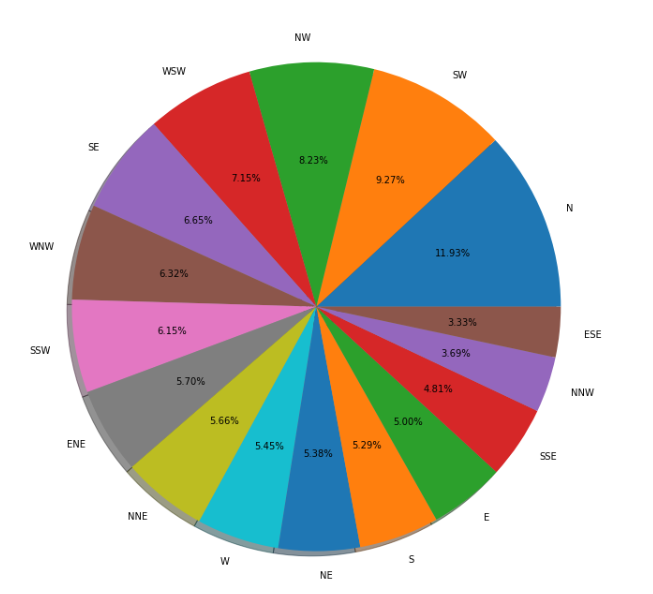
* **WindGustDir:**

**Pie Plot Of WindGusdir**

We can clearly see that the wind gust was strongest towards the North, followed by the SW, SSE, S,WNW.

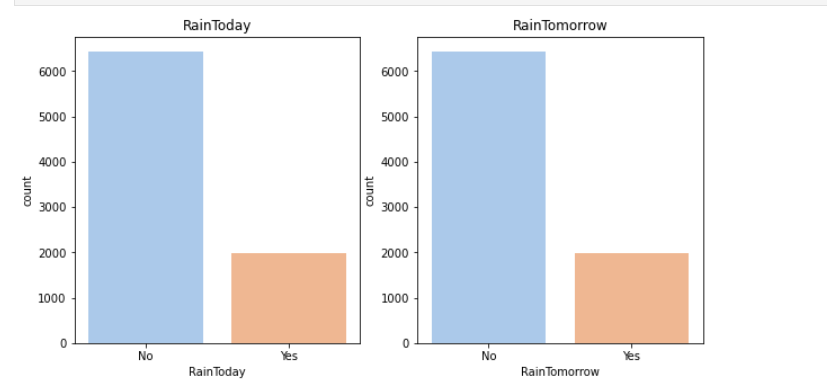
* **Windir9am And Windir3pm:**

**Pie plot of windir9am**

** Pie plot of windir3pm**

In the plots we can clearly see that the wing direction was towards the N at 9am and in the SE at 3pm

* **Rain Today And Rain Tomorrow:**

 **count plot of rain today and rain tomorrow**

From the count plot,we can not see any difference in the rainfall today and tomorrow.Here Rain Tomorrow column is our target column.since in the target column number of yes less than number of no ,the class of target column is imbalanced.We have to handle it.

### **Univariate Analysis of the Numerical columns**

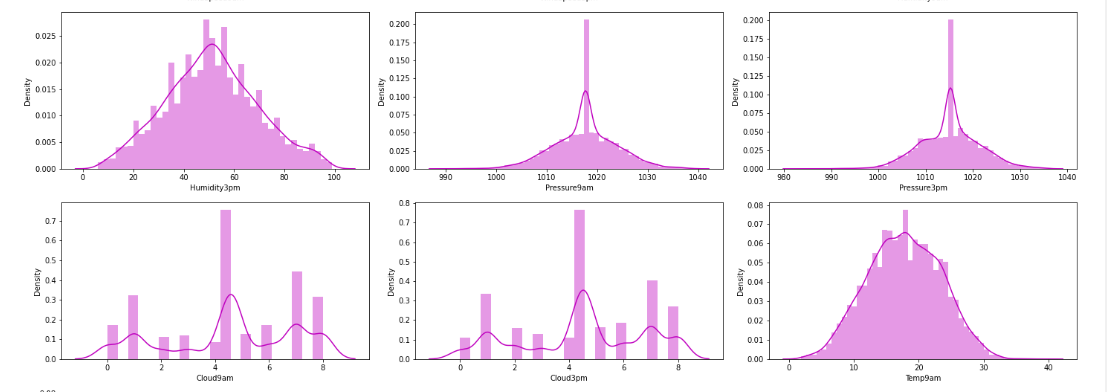
***Dist Plot:-***

**A** distplot plots a univariate distribution of observations. The distplot() function combines the matplotlib hist function with the seaborn kdeplot() and rugplot() functions. Seaborn distplot lets you show a histogram with a line on it.

distplot() function is used to plot the distplot. The distplot represents the univariate distribution of data i.e. data distribution of a variable against the density distribution. The seaborn. distplot() function accepts the data variable as an argument and returns the plot with the density distribution.

To explore a more about distplot [**Distplot**](https://seaborn.pydata.org/generated/seaborn.distplot.html).

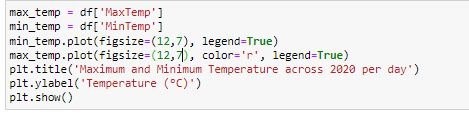
* **Dist Plot of Every Numerical columns:**

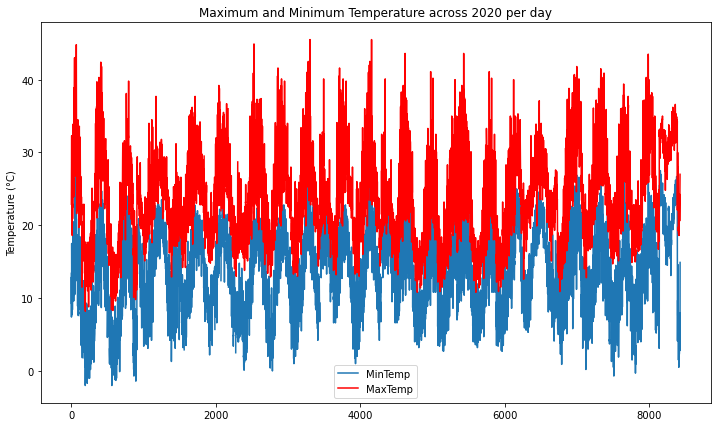


* No skewness Columns(like as a normal distribution curve):  
  -MinTemp  
  -Humidity3pm  
  -Pressure9am  
  -Pressure3pm  
  -Cloud9am  
  -Temp9am  
  -Temp3pm
* positive Skewness columns:  
  -MaxTemp  
  -Rainfall  
  -Evaporation  
  -WindGustSpee  
  -WindSpeed9am  
  -WindSpeed3pm
* Negative Skewness Columns:  
  -Sunshine  
  -Humidity9am  
  -Pressure3pm

***Bivariate Analysis:***

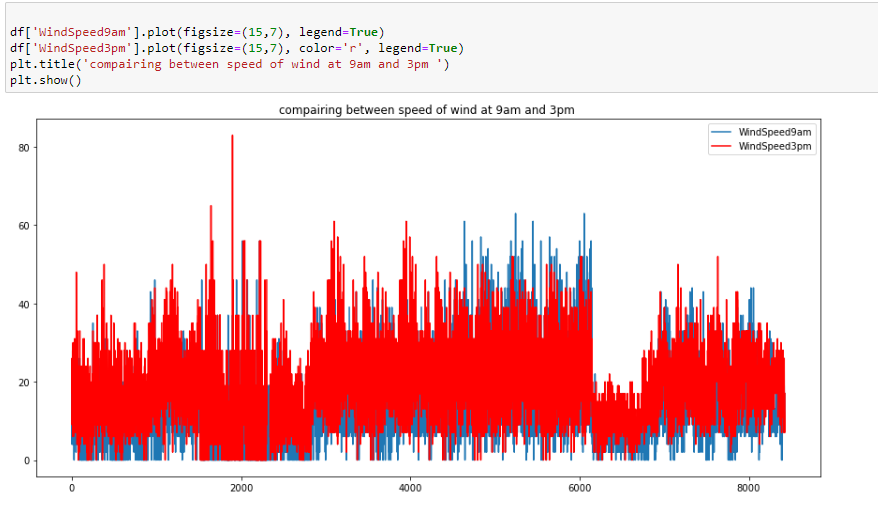
* **MinTemp Vs MaxTemp**

****

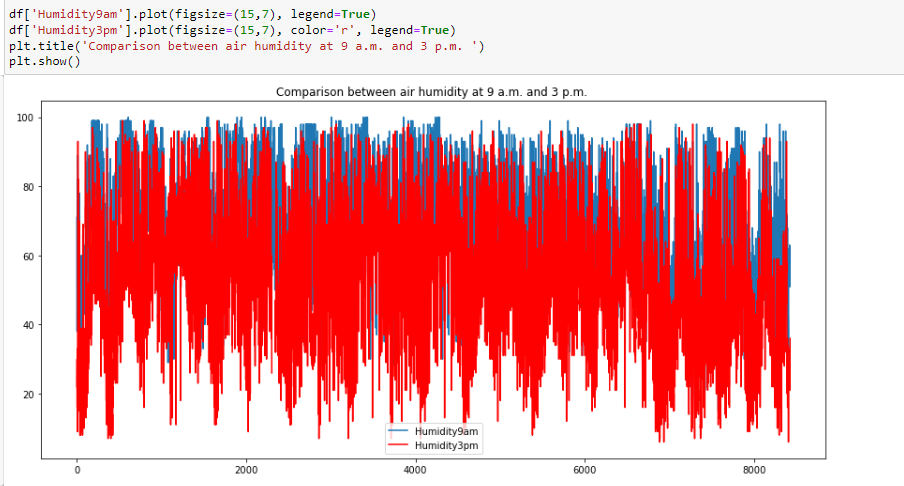
****

From the above plot we can see the difference between the daily minimum and maximum temperature

* **WindSpeed9am Vs WindSpeed3pm**

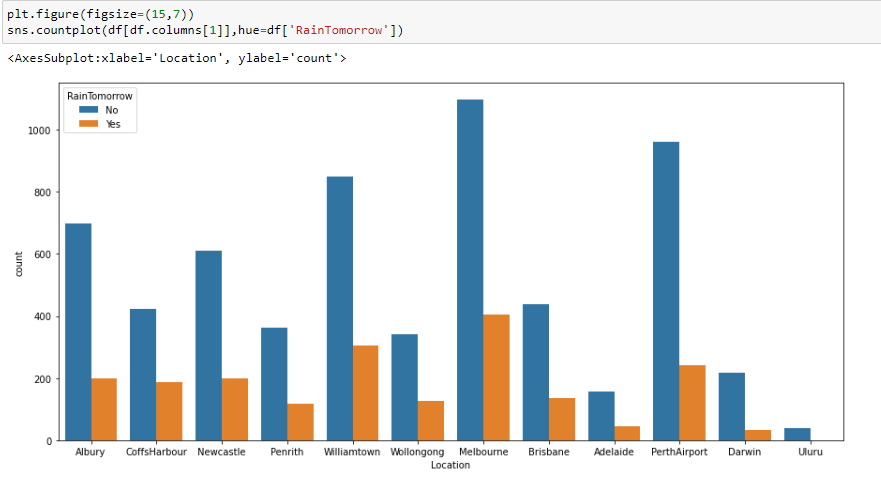
We see the difference between wind speed at 9 in the morning and wind speed at 3 in the afternoon. The wind speed fluctuates at 3 in the afternoon rather than the wind speed at 9 in the morning

* **Humidity9am Vs Humidity3pm**

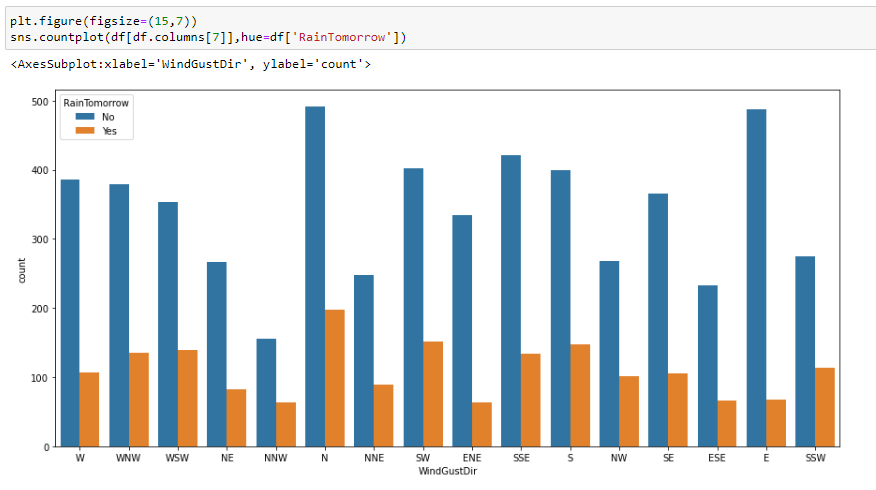


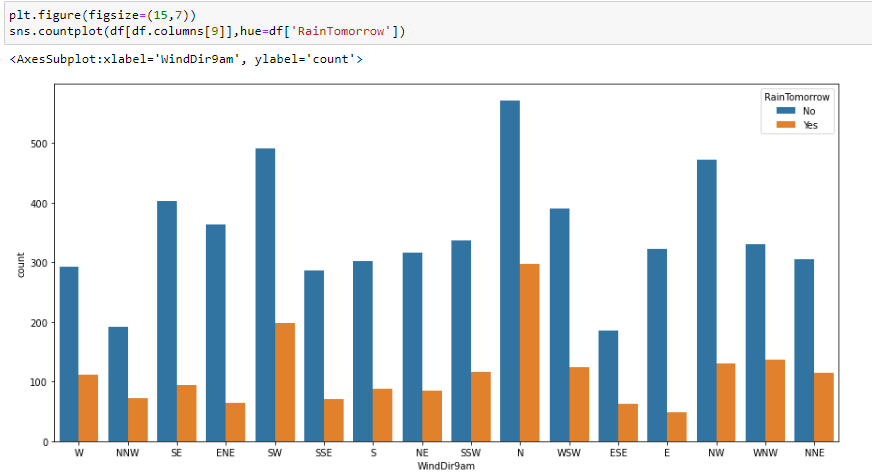
We see the difference between Humidity at 9 in the morning and Humidity at 3 in the afternoon. Morning humidity is always higher than afternoon humidity

* **Location Vs Rain Tomorrow**

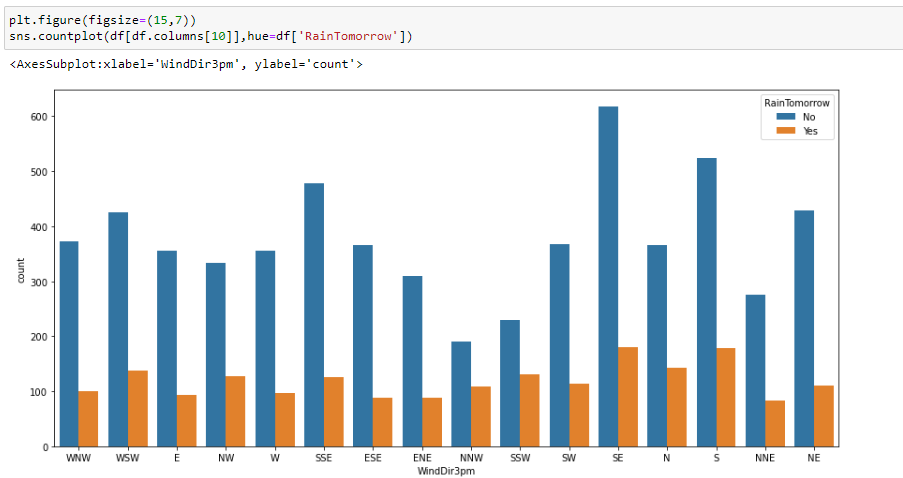


Number of yes less than number of No in rain tomorrow column for each location.we can not see number of yes for Uluru location.

* **WindGustdir Vs Rain Tomorrow**
* If the wind blows east, the chances of rain tomorrow are less than the wind in the other direction.
* If the wind blows NNW, the chances of rain tomorrow are more than the wind in other direction.
* **Windir9am Vs Rain Tomorrow**



* If the wind blows ENE in the morning, the chances of rain tomorrow are less than the wind in the other direction.
* If the wind blows North in the morning, the chances of rain tomorrow are more than the wind in other direction.
* **WEindir3pm Vs Rain Tomorrow**

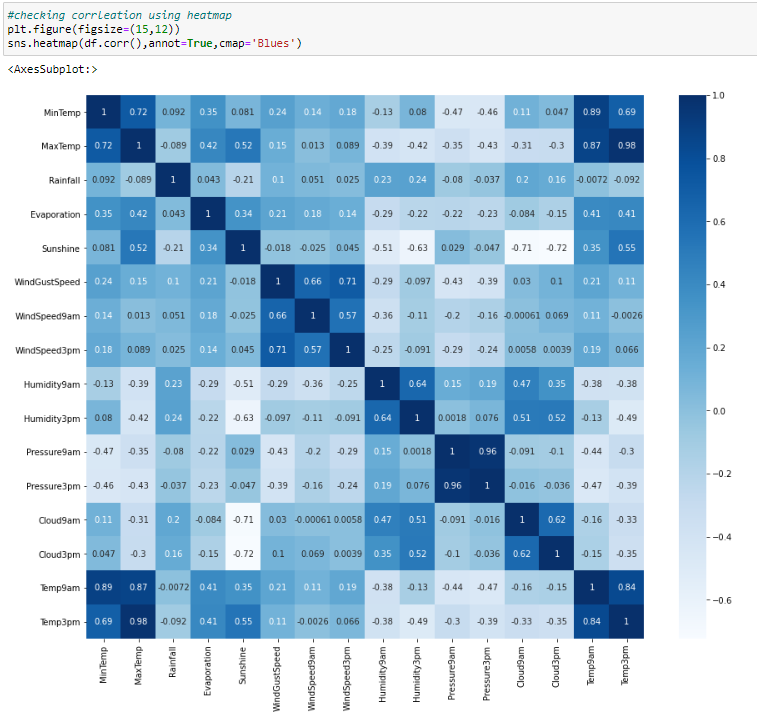


* If the wind blows NE in the after noon, the chances of rain tomorrow are less than the wind in the other direction.
* If the wind blows SSW in the afternoon, the chances of rain tomorrow are more than the wind in other direction.

***Correlation Using Heatmap:***

Correlation is a statistical term describing the degree to which two variables move in coordination with one another. If the two variables move in the same direction, then those variables are said to have a positive correlation. If they move in opposite directions, then they have a negative correlation.

Let’s see the correlation of our dataset using heatmap



The following feature pairs have a strong correlation with each other:

* MaxTemp and MinTemp
* Pressure9h and pressure3h
* Temp9am and Temp3pm
* Evaporation and MaxTemp
* MaxTemp and Temp3pm

In no case is the correlation value equal to a perfect “1”. We are therefore not removing any functionality

***Preprocessing:-***

Data processing is a processes of preparing the raw data and making it suitable for it a machine learning model. It is the first and crucial step while creating a machine learning model.

When creating machine learning project it is note always a case that become across the clean and formatted data and while doing any operation with data it is a mandatory to clean it and put in formatted way. So, for this we use data pre-processing task

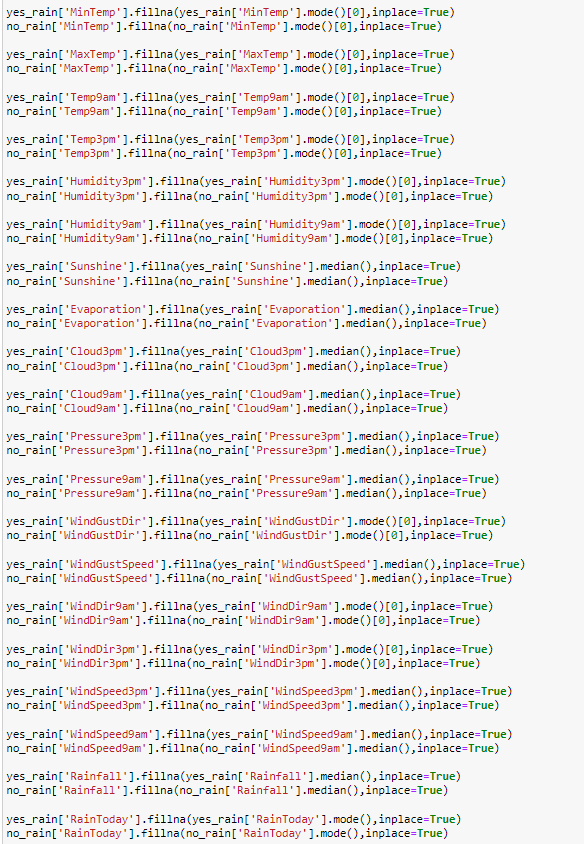
In this part I will do all the parts below

* Missing Value Handling
* Outliers Checking
* Outliers Removing
* Percentage Of Data Loss
* Label Encoding
* Dividing Data In Features And Vectors
* Multicolinearty Checking
* Oversampling
* Skewness Removing
* Data Standardizing
* ***Missing Value Handling:***

**There are 2 primary ways of handling missing values:**

* Deleting the Missing values:-Generally, this approach is not recommended. It is one of the quick and dirty techniques one can use to deal with missing values.
* Imputing the Missing Values:- There are different ways of replacing the missing values
* Replacing With Mean
* Replacing With Mode
* Replacing With Median,etc.

For my dataset,I will use mode for categorical features and median for numerical features by the code below.

After running this code ,we are free from missing values 

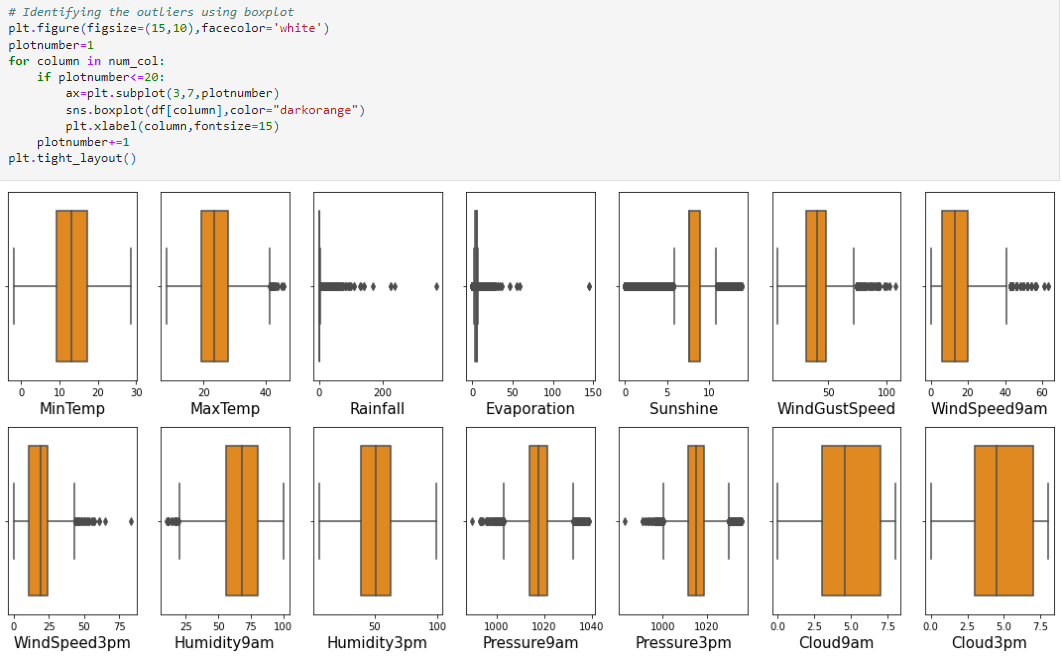
* ***Outliers Checking:***

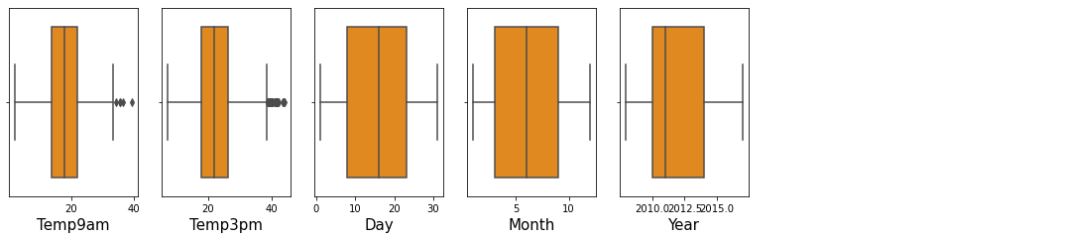
**Identification of potential outliers is important for the following reasons.**

* An outlier may indicate bad data. For example, the data may have been coded incorrectly or an experiment may not have been run correctly. If it can be determined that an outlying point is in fact erroneous, then the outlying value should be deleted from the analysis (or corrected if possible).
* In some cases, it may not be possible to determine if an outlying point is bad data. Outliers may be due to random variation or may indicate something scientifically interesting. In any event, we typically do not want to simply delete the outlying observation. However, if the data contains significant outliers, we may need to consider the use of robust statistical techniques.

Boxplots, histograms, and scatterplots can highlight outliers.

Boxplots display asterisks or other symbols on the graph to indicate explicitly when datasets contain outliers. These graphs use the interquartile method with fences to find outliers. The boxplot below displays our dataset. It’s clear that the outlier is quite different than the typical data value.



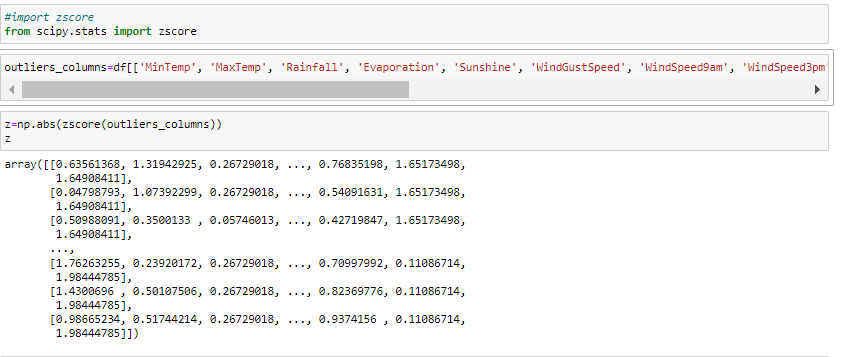


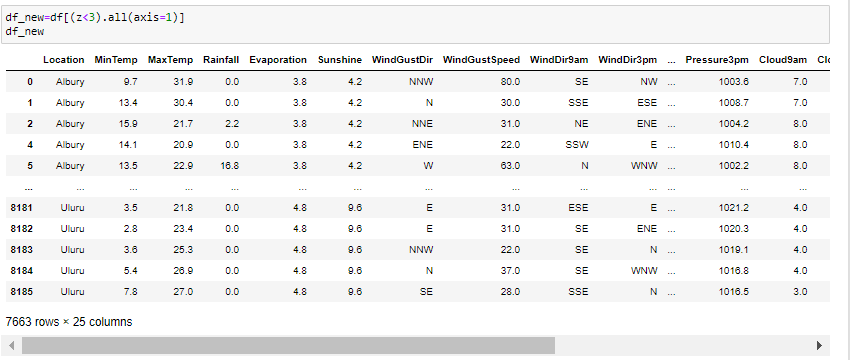
From the above boxplots, we can see that features having outliers are: MaxTemp,Rainfall,Evaporation,Sunshine,WindGustSpeed,WindSpeed9am,Windspeed3pm, Humidity9am, Pressure9am, Pressure3pm, Temp9am, Temp3pm.

* ***Outliers Removing:***

There are many ways to detect and remove Outliers.Here I have used Z-score function defined in scipy library to detect the outliers.

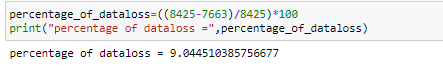
The formula for calculating a z-score is is z = (x-μ)/σ, where x is the raw score, μ is the population mean, and σ is the population standard deviation.

We have created a new dataframe df\_new where no outliers present.Let’s see the code



#### 

* ***Percentage Of Data Loss:***

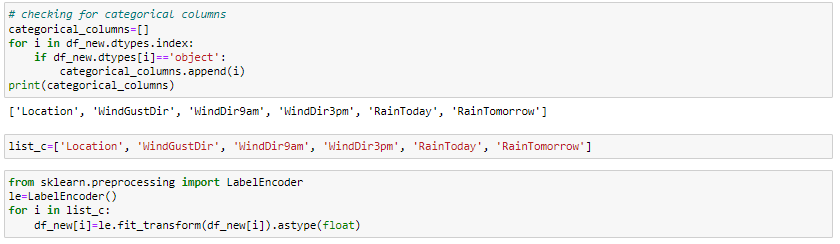


We lost 9 percent of our total data.

* ***Label Encoding:***

Label Encoding refers to converting the labels into a numeric form so as **to convert them into the machine-readable form**. Machine learning algorithms can then decide in a better way how those labels must be operated. It is an important pre-processing step for the structured dataset in supervised learning.

I have applied label encoding method to our cleaned dataframe df\_new and converted the categorical columns into numerical

******.

* ***Dividing Data In Features And Vectors:***

We have to divide our dataset columns into X and Y. X variables so that we could have all the attribute columns and our label / target variable with the Y variable.

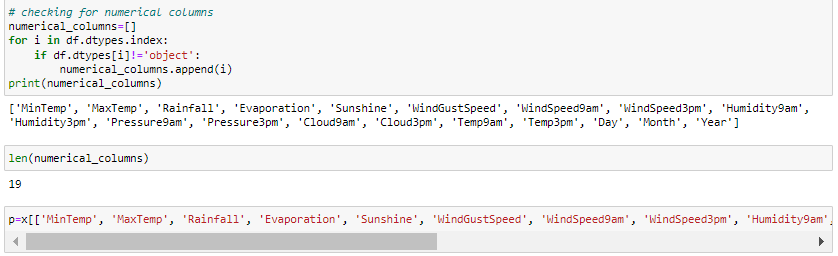


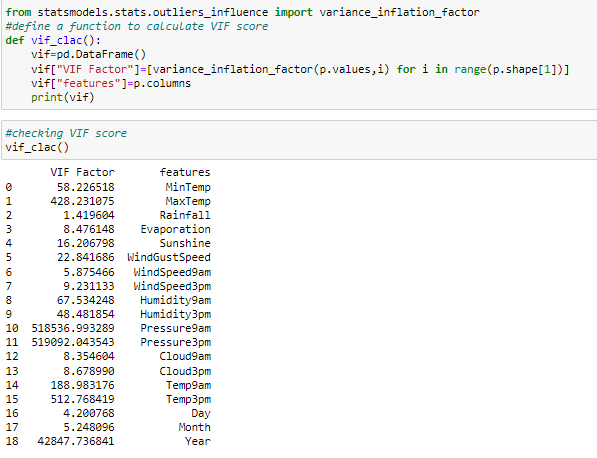
* ***Multicolinearity Checking:***

Multicollinearity occurs when two or more independent variables are highly correlated with one another in a regression model. This means that an independent variable can be predicted from another independent variable in a regression model.

The best way to identify the multicollinearity is to calculate the Variance Inflation Factor (VIF) corresponding to every independent Variable in the Dataset. VIF tells us about how well an independent variable is predictable using the other independent variables.

Let’s see the code how to get VIF score:





We find the multicollinearity in columns below

* Pressure9am
* Temp9am
* Temp3pm
* Year
* Pressure3pm

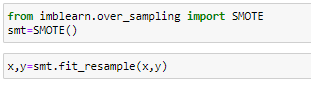
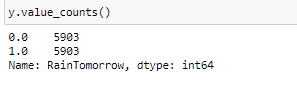
We drop this column from feature of the dataset.

* ***Oversampling:***

When one class of data is the underrepresented minority class in the data sample, over sampling techniques maybe used to duplicate these results for a more balanced amount of positive results in training. Over sampling is used when the amount of data collected is insufficient.

SMOTE is an oversampling technique where the synthetic samples are generated for the minority class. This algorithm helps to overcome the overfitting problem posed by random oversampling.

I have separated feature and label into x and y. And there is a data imbalance issue in target variable. Let’s use SMOTE to overcome from this.

* **Skewness Removing:**

If there are too much skewness in the data, then many statistical model don't work .In skewed data, the tail region may act as an outlier for the statistical model and we know that **outliers adversely affect the model's performance especially regression-based models**.

I used power transformation(method= yeo-johnson) method to remove skewness of the independent dataset.

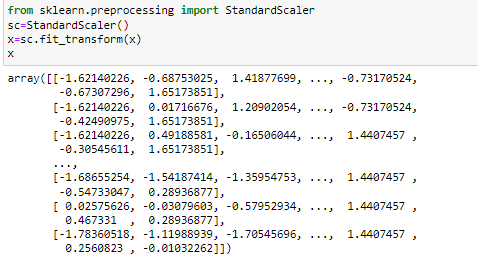
From above we can observe that we have sucessfully removed skewness from our independent dataset.

* ***Data Standardizing:***

Data standardization is a data processing workflow that converts the structure of different datasets into one common format of data.

Data standardization helps improve the quality of your data by transforming and standardizing it. Think of it like a uniform for your databases. By taking this step, you are formatting your records in a way that creates consistency across your systems and makes it easy for businesses to use.

Now let’s see how I did standardize the data:



Since I have done all the pre processing now my data is ready to build the model. Let’s get the predictions by creating some classification algorithms as it is a classification problem.

***Building Machine Learning:-***

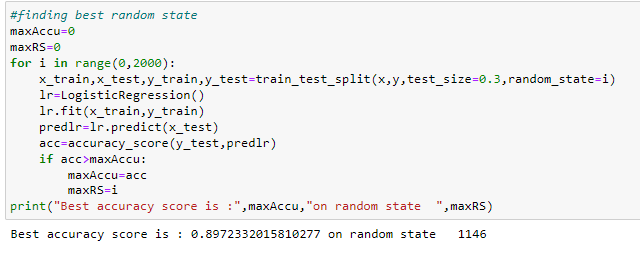
In here we will use various classification algorithm to predict our target. Let's have an overview of the algorithms we will use for our predictions. To read more about these algorithms , just click on the algorithms name.

* [**Logistic Regression**](https://machinelearningmastery.com/logistic-regression-for-machine-learning/) **:**- Logistic regression uses an equation as the representation, very much like linear regression. Input values (x) are combined linearly using weights or coefficient values (referred to as the Greek capital letter Beta) to predict an output value (y). A key difference from linear regression is that the output value being modeled is a *binary value* (0 or 1) rather than a numeric value. Below is an example logistic regression equation:

*y = e^(b0 + b1*x) / (1 + e^(b0 + b1*x))*

Where y is the predicted output, b0 is the bias or intercept term and b1 is the coefficient for the single input value (x).

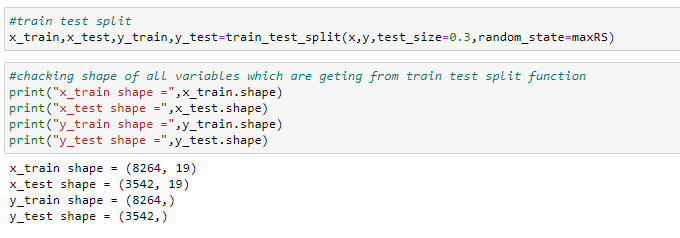
* [**K-Nearest Neighbors**](https://www.datacamp.com/community/tutorials/k-nearest-neighbor-classification-scikit-learn) **:**- In KNN, K is the number of nearest neighbors. The number of neighbors is the core deciding factor. K is generally an odd number if the number of classes is 2. When K=1, then the algorithm is known as the nearest neighbor algorithm. This is the simplest case. Suppose P1 is the point, for which label needs to predict. First, knn find the one closest point to P1 and then the label of the nearest point assigned to P1. Suppose P1 is the point, for which label needs to predict. First, knn find the k closest point to P1 and then classify points by majority vote of its k neighbors. Each object votes for their class and the class with the most votes is taken as the prediction. For finding closest similar points, knkn find the distance between points using distance measures such as Euclidean distance, Hamming distance, Manhattan distance and Minkowski distance.
* [**Support Vector Classifier (SVC)**](https://www.analyticsvidhya.com/blog/2017/09/understaing-support-vector-machine-example-code/) **:**- In the SVM algorithm, we plot each data item as a point in n-dimensional space (where n is number of features you have) with the value of each feature being the value of a particular coordinate. Then, we perform classification by finding the hyper-plane that differentiates the two classes very well. Support Vectors are simply the co-ordinates of individual observation. The SVM classifier is a frontier which best segregates the two classes (hyper-plane/ line).
* [**Random Forest Classifier**](https://www.datacamp.com/community/tutorials/random-forests-classifier-python) **:-** It technically is an ensemble method (based on the divide-and-conquer approach) of decision trees generated on a randomly split dataset. This collection of decision tree classifiers is also known as the forest. The individual decision trees are generated using an attribute selection indicator such as information gain, gain ratio, and Gini index for each attribute. Each tree depends on an independent random sample. In a classification problem, each tree votes and the most popular class is chosen as the final result.
* [**AdaBoost Classifier**](https://www.datacamp.com/community/tutorials/adaboost-classifier-python) **:**- Ada-boost or Adaptive Boosting is one of ensemble boosting classifier proposed by Yoav Freund and *Robert Schapire* in 1996. It combines multiple classifiers to increase the accuracy of classifiers. AdaBoost is an iterative ensemble method. AdaBoost classifier builds a strong classifier by combining multiple poorly performing classifiers so that you will get high accuracy strong classifier. The basic concept behind Adaboost is to set the weights of classifiers and training the data sample in each iteration such that it ensures the accurate predictions of unusual observations.
* [**Decision Tree Classifier**](https://www.datacamp.com/community/tutorials/decision-tree-classification-python) **:**- A decision tree is a flowchart-like tree structure where an internal node represents feature(or attribute), the branch represents a decision rule, and each leaf node represents the outcome. The topmost node in a decision tree is known as the root node. It learns to partition on the basis of the attribute value. It partitions the tree in recursively manner call recursive partitioning. This flowchart-like structure helps you in decision making.
* [**Gaussian**](https://www.geeksforgeeks.org/naive-bayes-classifiers/) **:-** Naive Bayes classifiers are a collection of classification algorithms based on Bayes’ Theorem. It is not a single algorithm but a family of algorithms where all of them share a common principle, i.e. every pair of features being classified is independent of each other.
* ***Finding Best Random State:-***

An algorithm might have multiple points that introduce randomness to the process and thus introduce randomness to the result. One method to make sure our results are constant is to set every possible *random\_state* available in the functions that we use. We can find the best random state by the code below.

### After we have found the value for best random state, we proceeded with the train test split function to create new training and testing datasets and fit them into the models to find our ideal models.

### ***Train\_Test\_Split:-***

The [Train\_Test\_Split](https://scikit-learn.org/stable/modules/generated/sklearn.model_selection.train_test_split.html) is a technique for evaluating the performance of a machine learning algorithm. It can be used for classification or regression problems and can be used for any supervised learning algorithm. The procedure involves taking a dataset and dividing it into two subsets. The first subset is used to fit the model and is referred to as the training dataset. The second subset is not used to train the model; instead, the input element of the dataset is provided to the model, then predictions are made and compared to the expected values. This second dataset is referred to as the test dataset.



Now I will be using 8 differents classification algorithms to get the ideal one for prediction.

I have used evaluation metrics like classification report, confusion matrix, roc score and and cross validation score to get the difference from the model accuracy.



Accuracy score of LogisticRegression() is

0.8972332015810277

[[1589 167]

[ 197 1589]]

precision recall f1-score support

0.0 0.89 0.90 0.90 1756

1.0 0.90 0.89 0.90 1786

accuracy 0.90 3542

macro avg 0.90 0.90 0.90 3542

weighted avg 0.90 0.90 0.90 3542

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Accuracy score of GaussianNB() is

0.8430265386787126

[[1487 269]

[ 287 1499]]

precision recall f1-score support

0.0 0.84 0.85 0.84 1756

1.0 0.85 0.84 0.84 1786

accuracy 0.84 3542

macro avg 0.84 0.84 0.84 3542

weighted avg 0.84 0.84 0.84 3542

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Accuracy score of SVC() is

0.932806324110672

[[1633 123]

[ 115 1671]]

precision recall f1-score support

0.0 0.93 0.93 0.93 1756

1.0 0.93 0.94 0.93 1786

accuracy 0.93 3542

macro avg 0.93 0.93 0.93 3542

weighted avg 0.93 0.93 0.93 3542

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Accuracy score of DecisionTreeClassifier() is

0.9367588932806324

[[1639 117]

[ 107 1679]]

precision recall f1-score support

0.0 0.94 0.93 0.94 1756

1.0 0.93 0.94 0.94 1786

accuracy 0.94 3542

macro avg 0.94 0.94 0.94 3542

weighted avg 0.94 0.94 0.94 3542

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Accuracy score of KNeighborsClassifier() is

0.9285714285714286

[[1553 203]

[ 50 1736]]

precision recall f1-score support

0.0 0.97 0.88 0.92 1756

1.0 0.90 0.97 0.93 1786

accuracy 0.93 3542

macro avg 0.93 0.93 0.93 3542

weighted avg 0.93 0.93 0.93 3542

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Accuracy score of RandomForestClassifier() is

0.9686617730095991

[[1685 71]

[ 40 1746]]

precision recall f1-score support

0.0 0.98 0.96 0.97 1756

1.0 0.96 0.98 0.97 1786

accuracy 0.97 3542

macro avg 0.97 0.97 0.97 3542

weighted avg 0.97 0.97 0.97 3542

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Accuracy score of GradientBoostingClassifier() is

0.9474872953133823

[[1664 92]

[ 94 1692]]

precision recall f1-score support

0.0 0.95 0.95 0.95 1756

1.0 0.95 0.95 0.95 1786

accuracy 0.95 3542

macro avg 0.95 0.95 0.95 3542

weighted avg 0.95 0.95 0.95 3542

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Accuracy score of AdaBoostClassifier() is

0.9282891022021457

[[1627 129]

[ 125 1661]]

precision recall f1-score support

0.0 0.93 0.93 0.93 1756

1.0 0.93 0.93 0.93 1786

accuracy 0.93 3542

macro avg 0.93 0.93 0.93 3542

weighted avg 0.93 0.93 0.93 3542

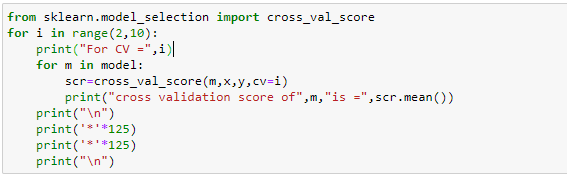
\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

***Cross Validation Score:***

The goal of cross-validation is to test the model's ability to predict new data that was not used in estimating it, in order to flag problems like overfitting or selection bias and to give an insight on how the model will generalize to an independent dataset (i.e., an unknown dataset, for instance from a real problem).

Now we check cross val score of the above models by the code below:



For CV = 2

cross validation score of LogisticRegression() is = 0.8449940708114518

cross validation score of GaussianNB() is = 0.8167880738607487

cross validation score of SVC() is = 0.8514314755209216

cross validation score of DecisionTreeClassifier() is = 0.7944265627646959

cross validation score of KNeighborsClassifier() is = 0.7996781297645266

cross validation score of RandomForestClassifier() is = 0.8320345586989666

cross validation score of GradientBoostingClassifier() is = 0.7519058106047772

cross validation score of AdaBoostClassifier() is = 0.746569540911401

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

For CV = 3

cross validation score of LogisticRegression() is = 0.8543109489227039

cross validation score of GaussianNB() is = 0.8167029739706546

cross validation score of SVC() is = 0.8650652928860927

cross validation score of DecisionTreeClassifier() is = 0.7042087848610379

cross validation score of KNeighborsClassifier() is = 0.82085375753694

cross validation score of RandomForestClassifier() is = 0.8820048777044315

cross validation score of GradientBoostingClassifier() is = 0.8383827388146816

cross validation score of AdaBoostClassifier() is = 0.7309728205287135

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

For CV = 4

cross validation score of LogisticRegression() is = 0.8589676722970212

cross validation score of GaussianNB() is = 0.820597795841564

cross validation score of SVC() is = 0.8779354513512942

cross validation score of DecisionTreeClassifier() is = 0.7890688207754664

cross validation score of KNeighborsClassifier() is = 0.838635294498489

cross validation score of RandomForestClassifier() is = 0.8800487857682712

cross validation score of GradientBoostingClassifier() is = 0.8249905984742667

cross validation score of AdaBoostClassifier() is = 0.809151438261248

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

For CV = 5

cross validation score of LogisticRegression() is = 0.8628653023428626

cross validation score of GaussianNB() is = 0.8208522558754471

cross validation score of SVC() is = 0.8904786035854295

cross validation score of DecisionTreeClassifier() is = 0.8446503494371743

cross validation score of KNeighborsClassifier() is = 0.8590548645233851

cross validation score of RandomForestClassifier() is = 0.9202086832277686

cross validation score of GradientBoostingClassifier() is = 0.8632866281419668

cross validation score of AdaBoostClassifier() is = 0.8404157525926707

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

For CV = 6

cross validation score of LogisticRegression() is = 0.8629527274555918

cross validation score of GaussianNB() is = 0.818655167995503

cross validation score of SVC() is = 0.895390473986082

cross validation score of DecisionTreeClassifier() is = 0.8789478461002201

cross validation score of KNeighborsClassifier() is = 0.8432185773942131

cross validation score of RandomForestClassifier() is = 0.9231702753288337

cross validation score of GradientBoostingClassifier() is = 0.8643827420734805

cross validation score of AdaBoostClassifier() is = 0.8449022695615874

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

For CV = 7

cross validation score of LogisticRegression() is = 0.8687057552159535

cross validation score of GaussianNB() is = 0.8225469304982518

cross validation score of SVC() is = 0.9008042401260795

cross validation score of DecisionTreeClassifier() is = 0.9053758683964127

cross validation score of KNeighborsClassifier() is = 0.8675211730562783

cross validation score of RandomForestClassifier() is = 0.9399381435656321

cross validation score of GradientBoostingClassifier() is = 0.8962296987429517

cross validation score of AdaBoostClassifier() is = 0.879796929920652

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

For CV = 8

cross validation score of LogisticRegression() is = 0.8732028845712186

cross validation score of GaussianNB() is = 0.8226384869780902

cross validation score of SVC() is = 0.9031022920398696

cross validation score of DecisionTreeClassifier() is = 0.9191034518396031

cross validation score of KNeighborsClassifier() is = 0.8851439988976161

cross validation score of RandomForestClassifier() is = 0.9507003008589409

cross validation score of GradientBoostingClassifier() is = 0.9007280901198842

cross validation score of AdaBoostClassifier() is = 0.8853136052546966

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

For CV = 9

cross validation score of LogisticRegression() is = 0.8706622254056255

cross validation score of GaussianNB() is = 0.8228936696784969

cross validation score of SVC() is = 0.8983589969385581

cross validation score of DecisionTreeClassifier() is = 0.8834491322125273

cross validation score of KNeighborsClassifier() is = 0.880230910432674

cross validation score of RandomForestClassifier() is = 0.9345243447667122

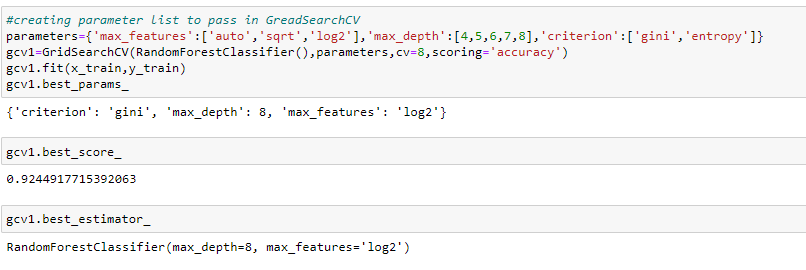
cross validation score of GradientBoostingClassifier() is = 0.8687154269998079

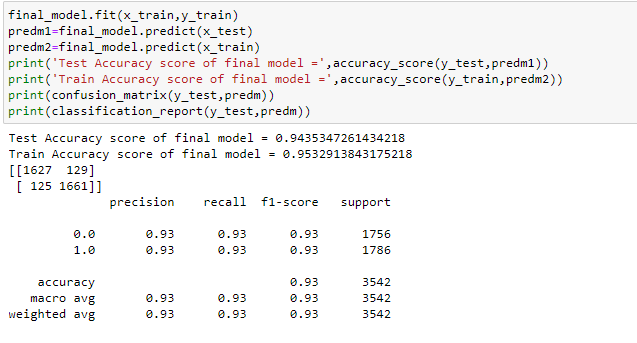
cross validation score of AdaBoostClassifier() is = 0.8679522629036359

|  |  |  |
| --- | --- | --- |
| **Model Name** | **Accuracy Score** | **Cross Val Score**  **(CV=8)** |
| *LogisticRegression* | 89.72% | 87.32% |
| *GaussianNB* | 84.30% | 82.26% |
| *SVC* | 93.28% | 90.31% |
| *DecisionTreeClassifier* | 93.67% | 91.91% |
| *KNeighborsClassifier* | 92.85% | 88.51% |
| *RandomForestClassifier* | 96.86% | 95.07% |
| *GradientBoostingClassifier* | 94.74% | 90.07% |
| *AdaBoostClassifier* | 92.82% | 88.53% |

Since *RandomForestClassifier* Classifier is giving best Accuracy and CV score, I choose *RandomForestClassifier* Classifier as best fitting model. Let’s check whether we can increase the Accuracy score by using Hyper parameter tuning.

#### **Hyperparameter tuning of RandomForestClassifier:-**

Parameters which define the model architecture are referred to as hyperparameters and thus this process of searching for the ideal model architecture is referred to as hyperparameter tuning. There are two types of hyperparameter tuning - Grid Search CV and Randomised Search , here we'll use grid search cv for our further tuning. Read more about [**Hyperparameter Tuning**](https://www.geeksforgeeks.org/hyperparameter-tuning/)**.**

We did not get best accuracy score from RandomForestClassifier after Parameter Tuning . But still we will create the final model with it to avoid underfit or overfit.

we can see that test accuravy score is very close to the test accracy score.so,our model is not underfit or overfit. so this our best model.we save this model for future prediction.

I have created an ROC curve plot and Confusion matrix for the final model.

# *ROCAUC Curve:*

# AUC - ROC curve is a performance measurement for the classification problems at various threshold settings. ROC is a probability curve and AUC represents the degree or measure of separability. It tells how much the model is capable of distinguishing between classes.

# The area under the ROC curve (AUC) results were considered excellent for AUC values between 0.9-1

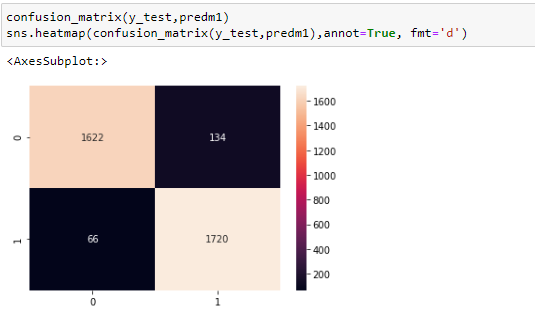
There is a lot more to learn about Roc\_Auc curve , visit the link attached below to eplore about roc\_auc curve . For now the ROC\_AUC curve helps us visualize how well our machine learning classifier is performing. Go through this article to have in depth knowledge of [**Roc\_Auc curve**](https://www.analyticsvidhya.com/blog/2020/06/auc-roc-curve-machine-learning/)**.**

# This is the ROC curve for the final model RandomForest and AUC for RandomForest is 94% which is considered a very good score.

* ***ConfusionMatrix:***

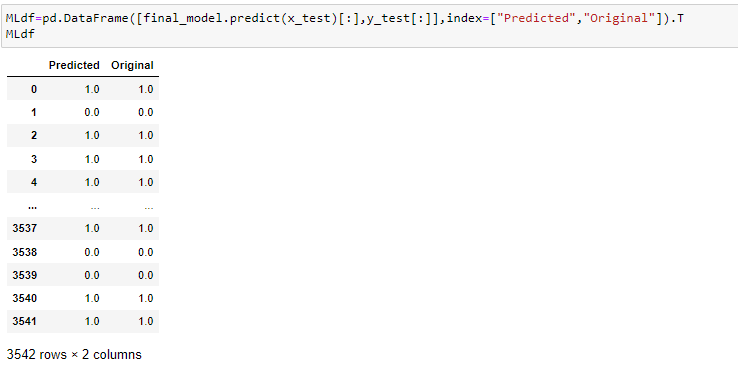
From our confusion matrix, we can calculate five different metrics measuring the validity of our model. **Accuracy (all correct / all) = TP + TN / TP + TN + FP + FN**. Misclassification (all incorrect / all) = FP + FN / TP + TN + FP + FN. Precision (true positives / predicted positives) = TP / TP + FP. A Confusion matrix is an N x N matrix used for **evaluating the performance of a classification model**, where N is the number of target classes. The matrix compares the actual target values with those predicted by the machine learning model.

Read more about [**confusion matrix**](https://scikit-learn.org/stable/modules/generated/sklearn.metrics.confusion_matrix.html)**.**



From the above confusion matrix we can see that our dataset is having 1622 as True positive and 1720 as True negative where as 134 as True negative and 66 as True negative value present in our dataset .

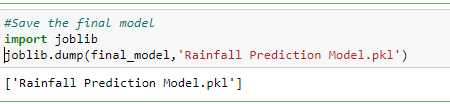
Final Model Applying On Test Data:

We can see from the above observation, Predicted and the Original values are matching each other, which means model performance is good.

***Saving Final Model:***

There are various ways through which we can save a machine learning model , in here we are using joblib to save our best model .

Read more about saving a machine learning model [**here**](https://www.geeksforgeeks.org/saving-a-machine-learning-model/)



***Conclussion:-***

* This particular problem needs a good vision on data, and in this problem Feature Engineering is the most crucial thing.

You can see how we have handled numerical and categorical data and how we build different machine learning models on the same dataset.

It is always advised to all of us that atleast we need to use 5 Algorithm in order to figure out which one is performing best among them and we choose that one and we send that for hyper parameter tuning to know that best parameter .

Using hyper parameter tunning we can improve our model accuracy, for instance in this model the accuracy remained same.

* Weather forecasts are increasingly accurate and useful, and their benefits extend widely across the economy. While much has been accomplished in improving weather forecasts, there remains much room for improvement.
* For future improvements, following step we thought to took-
* Replacing model with a latest/different model
* Using other robust datasets
* Predicting result on more attributes
* Training model on higher-end GPU
* Also, while performing weather forecasting, there was a lot of complexities involved. There are a lot of variables/attributes to consider for forecasting weather and if all or most of them are used, then we need a lot of computation power to get weather information. And, Real time weather forecasting is very difficult to forecast correctly.

For any of machine learning project my suggestion is first you have to understand the problem on ground level .if you don’t allow yourself to work with diligence .if you don’ t work harder anything that you are doing or will do , not only in case of machine learning but also in life cycle would be futile. Maybe, my endeavour assist you when ever you will get stuck.