Importing Libraries

```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
%matplotlib inline
import seaborn as sns
import plotly.express as px
from sklearn import preprocessing
import scipy.stats as stats
from sklearn.model_selection import train_test_split
from collections import Counter
from imblearn.over_sampling import SMOTE
from sklearn.metrics import accuracy_score,confusion_matrix,classification_report
from sklearn import metrics
from sklearn.ensemble import RandomForestClassifier
from xgboost import XGBClassifier
from sklearn.svm import SVC
from sklearn.linear_model import LogisticRegression
from sklearn.naive_bayes import GaussianNB
from sklearn.neighbors import KNeighborsClassifier
import joblib
import warnings
warnings.filterwarnings("ignore")
```

```
from google.colab import drive
drive.mount('/content/drive')
```

```
Mounted at /content/drive
```

```
df=pd.read_csv("/content/drive/MyDrive/Rainfall weather.csv.crdownload")
```

```
df.describe()
```

```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 4150 entries, 0 to 4149
Data columns (total 23 columns):
 #   Column         Non-Null Count  Dtype  
---  ------         --------------  -----  
 0   Date           4150 non-null   object 
 1   Location       4150 non-null   object 
 2   MinTemp        4128 non-null   float64
 3   MaxTemp        4135 non-null   float64
 4   Rainfall       4079 non-null   float64
 5   Evaporation    0 non-null      float64
 6   Sunshine       0 non-null      float64
 7   WindGustDir    4080 non-null   object 
 8   WindGustSpeed  4080 non-null   float64
 9   WindDir9am     3423 non-null   object 
 10  WindDir3pm     4058 non-null   object 
 11  WindSpeed9am   4114 non-null   float64
 12  WindSpeed3pm   4111 non-null   float64
 13  Humidity9am    4121 non-null   float64
 14  Humidity3pm    4117 non-null   float64
 15  Pressure9am    4128 non-null   float64
 16  Pressure3pm    4117 non-null   float64
 17  Cloud9am       1289 non-null   float64
 18  Cloud3pm       1427 non-null   float64
 19  Temp9am        4121 non-null   float64
 20  Temp3pm        4117 non-null   float64
 21  RainToday      4078 non-null   object 
 22  RainTomorrow   4078 non-null   object 
dtypes: float64(16), object(7)
memory usage: 745.8+ KB
```

```
df.sum
```

```
<bound method NDFrame._add_numeric_operations.<locals>.sum of             Date
Location  MinTemp  MaxTemp  Rainfall  Evaporation  \
0     2008-12-01         Albury     13.4     22.9       0.6          NaN
1     2008-12-02         Albury      7.4     25.1       0.0          NaN
2     2008-12-03         Albury     12.9     25.7       0.0          NaN
3     2008-12-04         Albury      9.2     28.0       0.0          NaN
4     2008-12-05         Albury     17.5     32.3       1.0          NaN
...          ...            ...      ...      ...       ...          ...
4145  2012-02-10  BadgerysCreek     16.6     26.3      48.2          NaN
4146  2012-02-11  BadgerysCreek     16.8     25.4       7.0          NaN
4147  2012-02-12  BadgerysCreek     13.2     27.0       6.6          NaN
4148  2012-02-13  BadgerysCreek     16.4     26.2       0.2          NaN
4149  2012-02-14  BadgerysCreek     16.2     26.3       5.2          NaN
```

```
      Sunshine WindGustDir  WindGustSpeed WindDir9am  ... Humidity9am  \
0          NaN           W           44.0          W  ...        71.0
1          NaN         WNW           44.0        NNW  ...        44.0
2          NaN         WSW           46.0          W  ...        38.0
3          NaN          NE           24.0         SE  ...        45.0
4          NaN           W           41.0        ENE  ...        82.0
...        ...         ...            ...        ...  ...         ...
4145       NaN           S           28.0        SSE  ...       100.0
4146       NaN           W           63.0         SW  ...        91.0
4147       NaN         WSW           41.0         SW  ...        91.0
4148       NaN          SE           30.0        ESE  ...        76.0
4149       NaN           E           31.0        SSE  ...         NaN

      Humidity3pm  Pressure9am  Pressure3pm  Cloud9am  Cloud3pm  Temp9am  \
0            22.0       1007.7       1007.1       8.0       NaN     16.9
1            25.0       1010.6       1007.8       NaN       NaN     17.2
2            30.0       1007.6       1008.7       NaN       2.0     21.0
3            16.0       1017.6       1012.8       NaN       NaN     18.1
4            33.0       1010.8       1006.0       7.0       8.0     17.8
...           ...          ...          ...       ...       ...      ...
4145         69.0       1012.1       1009.3       NaN       NaN     17.8
4146         96.0       1011.6       1010.8       NaN       NaN     18.5
4147         88.0       1015.5       1013.5       NaN       NaN     18.8
4148         75.0       1018.0       1016.2       NaN       NaN     21.3
4149          NaN          NaN          NaN       NaN       NaN      NaN

      Temp3pm  RainToday  RainTomorrow
0        21.8         No            No
1        24.3         No            No
2        23.2         No            No
3        26.5         No            No
4        29.7         No            No
...       ...        ...           ...
4145     24.4        Yes           Yes
4146     17.9        Yes           Yes
4147     20.0        Yes            No
4148     23.1         No           Yes
4149      NaN        NaN           NaN

[4150 rows x 23 columns]>
```

Handling Missing Values

```
df.isnull().sum()
```

```
Date                0
Location            0
MinTemp            22
MaxTemp            15
Rainfall           71
Evaporation      4150
Sunshine         4150
WindGustDir        70
```

```
WindGustSpeed      70
WindDir9am        727
WindDir3pm         92
WindSpeed9am       36
WindSpeed3pm       39
Humidity9am        29
Humidity3pm        33
Pressure9am        22
Pressure3pm        33
Cloud9am         2861
Cloud3pm         2723
Temp9am            29
Temp3pm            33
RainToday          72
RainTomorrow       72
dtype: int64
```

```
msno.matrix(df,color=(0.55,0.255,0.255),fontsize=16)
```

```
df_c=df[["RainToday","WindGustDir","WindDir9am","WindDir3pm"]]
```

```
df.drop(columns=["Evaporation","Sunshine","Cloud9am","Cloud3pm"],axis=1,inplace=True)
df.drop(columns=["RainToday","WindGustDir","WindDir9am","WindDir3pm"],axis=1,inplace=True)
```

```
c_names=df_c.columns
```

```
from sklearn.impute import SimpleImputer
```

```
imp_mode=SimpleImputer(missing_values=np.nan,strategy="most_frequent")
df_c=imp_mode.fit_transform(df_c)
df_c=pd.DataFrame(df_c,columns=c_names)
```

```
df_c.tail()
```

```
df_c.head()
```

Split data into dependent and independent variable

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

```
df = df[df['RainTomorrow'].notnull()]
df['Pressure9am'].fillna(df['Pressure9am'].mean(),inplace=True)
df['Pressure3pm'].fillna(df['Pressure3pm'].mean(),inplace=True)
```

```
X=df.drop('RainTomorrow',axis=1)
Y=df['RainTomorrow']
```

```
set(Y)
```

```
{'No', 'Yes'}
```

```
X=X.drop('Date',axis=1)
```

```
names=X.columns
names
```

```
Index(['Location', 'MinTemp', 'MaxTemp', 'Rainfall', 'WindGustSpeed',
       'WindSpeed9am', 'WindSpeed3pm', 'Humidity9am', 'Humidity3pm',
       'Pressure9am', 'Pressure3pm', 'Temp9am', 'Temp3pm'],
      dtype='object')
```

```
from sklearn.preprocessing import LabelEncoder, MinMaxScaler
sc=StandardScaler()
```

```
print(len(X),len(Y))
```

```
4078 4078
```

Encode Label

```
from zmq.constants import XPUB_VERBOSE
LE = LabelEncoder()
X['Location'] = LE.fit_transform(X['Location'])
X.head()
```

```
LE = LabelEncoder()
```

```
Y=pd.DataFrame(Y)
Y= LE.fit_transform(Y)
print(len(X),len(Y))
```

```
4078 4078
```

```
sc=StandardScaler()
```

```
X=sc.fit_transform(X)
X[:7]
```

```
array([[-0.59528759,  0.56837949, -0.0085919 , -0.22426047,  0.78583207,
         1.6959896 ,  1.2723515 , -0.22201428, -1.37796131, -1.47074286,
        -1.22154605,  0.32115973,  0.02625225],
       [-0.59528759, -0.42682132,  0.28860911, -0.32021537,  0.78583207,
        -0.64056306,  1.00035747, -1.7954242 , -1.22486289, -1.07041563,
        -1.12232734,  0.3701459 ,  0.37441575],
       [-0.59528759,  0.48544609,  0.36966393, -0.32021537,  0.93712123,
         1.54995506,  1.54434552, -2.14507085, -0.96969886, -1.48454725,
        -0.99476042,  0.99063741,  0.22122381],
       [-0.59528759, -0.12826107,  0.68037408, -0.32021537, -0.72705948,
         0.38167873, -0.76760369, -1.73714976, -1.68415815, -0.10410851,
        -0.41362225,  0.51710442,  0.68079963],
       [-0.59528759,  1.24843337,  1.26126698, -0.16029055,  0.55889834,
        -0.20245943,  0.72836345,  0.41900458, -0.81660044, -1.04280686,
        -1.37746117,  0.46811825,  1.12644891],
       [-0.59528759,  0.76741965,  0.91002941, -0.2882304 ,  1.693567  ,
         1.54995506,  1.2723515 , -1.15440534, -1.3269285 , -1.26367705,
        -1.46250579,  0.92532252,  1.01503659],
       [-0.59528759,  0.71765961,  0.27509997, -0.32021537,  1.23969954,
         1.6959896 ,  1.2723515 , -1.50405199, -1.53105973, -1.2084595 ,
        -1.06563093,  0.51710442,  0.41619537]])
```

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

Train and Test the Model And Model Evaluation

```
from sklearn import model_selection
```

```
import sklearn.metrics as metrics
```

```
!pip3 install catboost
from catboost import CatBoostClassifier
cat = CatBoostClassifier(iterations=2000, eval_metric = "AUC")
cat.fit(X_train_res, y_train_res)
```

```
Looking in indexes: https://pypi.org/simple, https://us-python.pkg.dev/colab-wheels/p
Collecting catboost
```

```
  Downloading catboost-1.1.1-cp37-none-manylinux1_x86_64.whl (76.6 MB)
     |████████████████████████████████| 76.6 MB 1.2 MB/s
Requirement already satisfied: matplotlib in /usr/local/lib/python3.7/dist-packages (
Requirement already satisfied: graphviz in /usr/local/lib/python3.7/dist-packages (fr
Requirement already satisfied: scipy in /usr/local/lib/python3.7/dist-packages (from 
Requirement already satisfied: plotly in /usr/local/lib/python3.7/dist-packages (from
Requirement already satisfied: six in /usr/local/lib/python3.7/dist-packages (from ca
Requirement already satisfied: numpy>=1.16.0 in /usr/local/lib/python3.7/dist-package
Requirement already satisfied: pandas>=0.24.0 in /usr/local/lib/python3.7/dist-packag
Requirement already satisfied: python-dateutil>=2.7.3 in /usr/local/lib/python3.7/dis
Requirement already satisfied: pytz>=2017.3 in /usr/local/lib/python3.7/dist-packages
Requirement already satisfied: kiwisolver>=1.0.1 in /usr/local/lib/python3.7/dist-pac
Requirement already satisfied: cycler>=0.10 in /usr/local/lib/python3.7/dist-packages
Requirement already satisfied: pyparsing!=2.0.4,!=2.1.2,!=2.1.6,>=2.0.1 in /usr/local
Requirement already satisfied: typing-extensions in /usr/local/lib/python3.7/dist-pac
Requirement already satisfied: tenacity>=6.2.0 in /usr/local/lib/python3.7/dist-packa
Installing collected packages: catboost
Successfully installed catboost-1.1.1
Learning rate set to 0.050311
0:      total: 102ms    remaining: 3m 23s
1:      total: 150ms    remaining: 2m 30s
2:      total: 203ms    remaining: 2m 15s
3:      total: 258ms    remaining: 2m 8s
4:      total: 305ms    remaining: 2m 1s
5:      total: 355ms    remaining: 1m 57s
6:      total: 408ms    remaining: 1m 56s
7:      total: 458ms    remaining: 1m 53s
8:      total: 508ms    remaining: 1m 52s
9:      total: 557ms    remaining: 1m 50s
10:     total: 605ms    remaining: 1m 49s
11:     total: 658ms    remaining: 1m 49s
12:     total: 713ms    remaining: 1m 49s
13:     total: 766ms    remaining: 1m 48s
14:     total: 815ms    remaining: 1m 47s
15:     total: 868ms    remaining: 1m 47s
16:     total: 916ms    remaining: 1m 46s
17:     total: 965ms    remaining: 1m 46s
18:     total: 1.02s    remaining: 1m 46s
19:     total: 1.07s    remaining: 1m 46s
20:     total: 1.12s    remaining: 1m 45s
21:     total: 1.17s    remaining: 1m 45s
22:     total: 1.22s    remaining: 1m 44s
23:     total: 1.27s    remaining: 1m 44s
24:     total: 1.32s    remaining: 1m 44s
25:     total: 1.37s    remaining: 1m 43s
26:     total: 1.43s    remaining: 1m 44s
27:     total: 1.48s    remaining: 1m 44s
28:     total: 1.53s    remaining: 1m 43s
29:     total: 1.57s    remaining: 1m 43s
30:     total: 1.62s    remaining: 1m 42s
31:     total: 1.67s    remaining: 1m 42s
32:     total: 1.72s    remaining: 1m 42s
33:     total: 1.78s    remaining: 1m 42s
34:     total: 1.83s    remaining: 1m 42s
35:     total: 1.88s    remaining: 1m 42s
```

```
X_train,X_test,Y_train,Y_test=model_selection.train_test_split(X,Y,test_size=0.2,random_state
```

```
from sklearn.ensemble import RandomForestClassifier
from sklearn.ensemble import GradientBoostingClassifier
```

```
RFC=RandomForestClassifier()
```

```
GBC=GradientBoostingClassifier()
```

```
np.any(np.isnan(X))
```

```
True
```

```
Y_pred = cat.predict(X_test)
print(confusion_matrix(Y_test,Y_pred))
print(accuracy_score(Y_test,Y_pred))
print(classification_report(Y_test,Y_pred))
```

```
[[21459  1258]
 [ 2702  3673]]
0.8638801044960814
              precision    recall  f1-score   support

           0       0.89      0.94      0.92     22717
           1       0.74      0.58      0.65      6375

    accuracy                           0.86     29092
   macro avg       0.82      0.76      0.78     29092
weighted avg       0.86      0.86      0.86     29092
```

```
metrics.plot_roc_curve(cat, X_test, Y_test)
metrics.roc_auc_score(Y_test, Y_pred, average=None)
```

```
RFC=RandomForestClassifier()
RFC.fit(X_train_res,Y_train_res)
```

```
RandomForestClassifier()
```

```
Y_pred1 = rf.predict(X_test)
print(confusion_matrix(Y_test,Y_pred1))
print(accuracy_score(Y_test,Y_pred1))
print(classification_report(Y_test,Y_pred1))
```

```
[[20595  2122]
 [ 2356  4019]]
0.8460745222054173
              precision    recall  f1-score   support

           0       0.90      0.91      0.90     22717
           1       0.65      0.63      0.64      6375

    accuracy                           0.85     29092
   macro avg       0.78      0.77      0.77     29092
weighted avg       0.84      0.85      0.85     29092
```

```
metrics.plot_roc_curve(rf, X_test, Y_test)
metrics.roc_auc_score(Y_test, Y_pred1, average=None)
```

```
logreg = LogisticRegression()
logreg.fit(X_train_res, Y_train_res)
```

```
LogisticRegression()
```

```
Y_pred2 = logreg.predict(X_test)
print(confusion_matrix(Y_test,Y_pred2))
print(accuracy_score(Y_test,Y_pred2))
print(classification_report(Y_test,Y_pred2))
```

```
[[17601  5116]
 [ 1515  4860]]
0.772067922452908
              precision    recall  f1-score   support

           0       0.92      0.77      0.84     22717
           1       0.49      0.76      0.59      6375

    accuracy                           0.77     29092
   macro avg       0.70      0.77      0.72     29092
weighted avg       0.83      0.77      0.79     29092
```

Save The Model

```
import pickle


pickle.dump(RFC,open('rainfall.pkl','wb'))
pickle.dump(LE,open('encoder.pkl','wb'))
pickle.dump(imp_mode,open('imputer.pkl','wb'))
pickle.dump(sc,open('scale.pkl','wb'))
```

Colab paid products - Cancel contracts here