```
import pandas as pd
import numpy as np
import seaborn as sns
import matplotlib.pyplot as plt
import warnings
warnings.filterwarnings("ignore")
pd.set_option("display.max_rows", None)
pd.set_option("display.max_columns", None)
from statsmodels.stats.outliers_influence import variance_inflation_factor
from statsmodels.tools.tools import add_constant
from sklearn.model_selection import train_test_split,cross_val_score
from xgboost import XGBClassifier
from sklearn.metrics import confusion_matrix, classification_report, accuracy_score,roc_curve, auc
```

```
df=pd.read_csv("/content/drive/MyDrive/Colab Notebooks/Data Science Projects & Resources/September Placement Project/I
```

```
df.head()
```

	EventId	DER_mass_MMC	DER_mass_transverse_met_lep	DER_mass_vis	DER_pt_h	DER_deltaeta_jet_jet	DER_mass_jet_jet
0	100000	138.470	51.655	97.827	27.980	0.91	124.711
1	100001	160.937	68.768	103.235	48.146	-999.00	-999.000
2	100002	-999.000	162.172	125.953	35.635	-999.00	-999.000
3	100003	143.905	81.417	80.943	0.414	9.00	-999.000
4	100004	175.864	16.915	134.805	16.405	-999.00	-999.000

```
df.shape
```

```
(250000, 33)
```

```
df=df.drop(columns=["EventId"],axis=1)
```

```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 250000 entries, 0 to 249999
Data columns (total 32 columns):
 #   Column                      Non-Null Count   Dtype  
---  ------                      --------------   -----  
 0   DER_mass_MMC                250000 non-null  float64
 1   DER_mass_transverse_met_lep  250000 non-null  float64
 2   DER_mass_vis                250000 non-null  float64
 3   DER_pt_h                    250000 non-null  float64
 4   DER_deltaeta_jet_jet        250000 non-null  float64
 5   DER_mass_jet_jet            250000 non-null  float64
 6   DER_prodeta_jet_jet         250000 non-null  float64
 7   DER_deltar_tau_lep          250000 non-null  float64
 8   DER_pt_tot                  250000 non-null  float64
 9   DER_sum_pt                  250000 non-null  float64
 10  DER_pt_ratio_lep_tau        250000 non-null  float64
 11  DER_met_phi_centrality      250000 non-null  float64
 12  DER_lep_eta_centrality      250000 non-null  float64
 13  PRI_tau_pt                  250000 non-null  float64
 14  PRI_tau_eta                 250000 non-null  float64
 15  PRI_tau_phi                 250000 non-null  float64
 16  PRI_lep_pt                  250000 non-null  float64
 17  PRI_lep_eta                 250000 non-null  float64
 18  PRI_lep_phi                 250000 non-null  float64
 19  PRI_met                     250000 non-null  float64
 20  PRI_met_phi                 250000 non-null  float64
 21  PRI_met_sumet               250000 non-null  float64
 22  PRI_jet_num                 250000 non-null  int64  
 23  PRI_jet_leading_pt          250000 non-null  float64
 24  PRI_jet_leading_eta         250000 non-null  float64
 25  PRI_jet_leading_phi         250000 non-null  float64
 26  PRI_jet_subleading_pt       250000 non-null  float64
 27  PRI_jet_subleading_eta      250000 non-null  float64
 28  PRI_jet_subleading_phi      250000 non-null  float64
```

```
 29  PRI_jet_all_pt               250000 non-null  float64
 30  Weight                       250000 non-null  float64
 31  Label                        250000 non-null  object
dtypes: float64(30), int64(1), object(1)
memory usage: 61.0+ MB
```

```
df.duplicated().sum()
```

```
np.int64(0)
```

```
df["Label"].value_counts()/df.shape[0]*100
```

Label	count
b	65.7332
s	34.2668

dtype: float64

```
df["Label"]=df["Label"].map({"s":1,"b":0})
```

```
for i in df.describe().keys():
  print(i,"------",df[i].nunique())
```

```
DER_mass_MMC ------ 108338
DER_mass_transverse_met_lep ------ 101637
DER_mass_vis ------ 100558
DER_pt_h ------ 115563
DER_deltaeta_jet_jet ------ 7088
DER_mass_jet_jet ------ 68366
DER_prodeta_jet_jet ------ 16593
DER_deltar_tau_lep ------ 4692
DER_pt_tot ------ 59042
DER_sum_pt ------ 156098
DER_pt_ratio_lep_tau ------ 5931
DER_met_phi_centrality ------ 2829
DER_lep_eta_centrality ------ 1002
PRI_tau_pt ------ 59639
PRI_tau_eta ------ 4971
PRI_tau_phi ------ 6285
PRI_lep_pt ------ 61929
PRI_lep_eta ------ 4987
PRI_lep_phi ------ 6285
PRI_met ------ 87836
PRI_met_phi ------ 6285
PRI_met_sumet ------ 179740
PRI_jet_num ------ 4
PRI_jet_leading_pt ------ 86590
PRI_jet_leading_eta ------ 8558
PRI_jet_leading_phi ------ 6285
PRI_jet_subleading_pt ------ 42464
PRI_jet_subleading_eta ------ 8628
PRI_jet_subleading_phi ------ 6286
PRI_jet_all_pt ------ 103559
Weight ------ 104094
Label ------ 2
```

`This looks like the Higgs Boson challenge dataset from Kaggle so they explicitly state "-999 means missing value`

```
df = df.replace(-999, np.nan)
```

```
cols_to_drop = []

for col in df.describe().columns:
    null_percent = df[col].isna().mean() * 100
    corr_with_label = df[col].corr(df['Label'])

    if null_percent > 0:
```

```
        print(f"{col}    {round(null_percent, 2)}%   And Correlation: {round(corr_with_label, 3)}")
        cols_to_drop.append(col)
```

```
DER_mass_MMC    15.25%   And Correlation: 0.012
DER_deltaeta_jet_jet    70.98%   And Correlation: 0.328
DER_mass_jet_jet    70.98%   And Correlation: 0.317
DER_prodeta_jet_jet    70.98%   And Correlation: -0.294
DER_lep_eta_centrality    70.98%   And Correlation: 0.308
PRI_jet_leading_pt    39.97%   And Correlation: 0.109
PRI_jet_leading_eta    39.97%   And Correlation: 0.0
PRI_jet_leading_phi    39.97%   And Correlation: -0.0
PRI_jet_subleading_pt    70.98%   And Correlation: -0.023
PRI_jet_subleading_eta    70.98%   And Correlation: 0.001
PRI_jet_subleading_phi    70.98%   And Correlation: -0.006
```

```
df = df.drop(columns=cols_to_drop)
```

```
df.shape
```

```
(250000, 21)
```

```
df.describe().T[["min","25%","50%","75%","max"]]
```

	min	25%	50%	75%	max
DER_mass_transverse_met_lep	0.000000	19.241000	46.524000	73.598000	690.075000
DER_mass_vis	6.329000	59.388750	73.752000	92.259000	1349.351000
DER_pt_h	0.000000	14.068750	38.467500	79.169000	2834.999000
DER_deltar_tau_lep	0.208000	1.810000	2.491500	2.961000	5.684000
DER_pt_tot	0.000000	2.841000	12.315500	27.591000	2834.999000
DER_sum_pt	46.104000	77.550000	120.664500	200.478250	1852.462000
DER_pt_ratio_lep_tau	0.047000	0.883000	1.280000	1.777000	19.773000
DER_met_phi_centrality	-1.414000	-1.371000	-0.356000	1.225000	1.414000
PRI_tau_pt	20.000000	24.591750	31.804000	45.017000	764.408000
PRI_tau_eta	-2.499000	-0.925000	-0.023000	0.898000	2.497000
PRI_tau_phi	-3.142000	-1.575000	-0.033000	1.565000	3.142000
PRI_lep_pt	26.000000	32.375000	40.516000	53.390000	560.271000
PRI_lep_eta	-2.505000	-1.014000	-0.045000	0.959000	2.503000
PRI_lep_phi	-3.142000	-1.522000	0.086000	1.618000	3.142000
PRI_met	0.109000	21.398000	34.802000	51.895000	2842.617000
PRI_met_phi	-3.142000	-1.575000	-0.024000	1.561000	3.142000
PRI_met_sumet	13.678000	123.017500	179.739000	263.379250	2003.976000
PRI_jet_num	0.000000	0.000000	1.000000	2.000000	3.000000
PRI_jet_all_pt	0.000000	0.000000	40.512500	109.933750	1633.433000
Weight	0.001502	0.018636	1.156188	2.404128	7.822543
Label	0.000000	0.000000	0.000000	1.000000	1.000000

```
X, y = df.drop(columns=["Label"]), df[["Label"]]
X_train, X_test, y_train, y_test = train_test_split(
    X, y, test_size=0.3, random_state=42,stratify=y
)
```

```
for col in X_train.select_dtypes(include=[np.number]).columns:
    if col not in ["PRI_jet_num"]:
```

```
    Q1 = X_train[col].quantile(0.25)
    Q3 = X_train[col].quantile(0.75)
    IQR = Q3 - Q1
    lower = Q1 - 1.5 * IQR
    upper = Q3 + 1.5 * IQR

    # Cap outliers in train set
    X_train[col] = X_train[col].clip(lower, upper)

    # Cap outliers in test set using same limits
    X_test[col] = X_test[col].clip(lower, upper)
```

- `> 1` → highly right skewed
- `0.5 – 1` → moderately right skewed
- `-0.5 – 0.5` → approximately symmetric
- `< -1` → highly left skewed

```
skew_vals = X_train.skew(numeric_only=True).sort_values(ascending=False)
print(skew_vals)
```

```
PRI_jet_all_pt                1.165093
PRI_tau_pt                    1.104698
DER_sum_pt                    1.099283
DER_pt_h                      1.060041
PRI_lep_pt                    1.041495
DER_pt_tot                    1.022211
Weight                        0.937619
PRI_met_sumet                 0.868196
PRI_met                       0.853049
DER_pt_ratio_lep_tau          0.770962
PRI_jet_num                   0.610937
DER_mass_vis                  0.592298
DER_mass_transverse_met_lep   0.512645
DER_met_phi_centrality        0.151360
PRI_lep_eta                   0.021327
PRI_tau_eta                   0.016492
PRI_tau_phi                   0.013794
PRI_met_phi                   0.010342
PRI_lep_phi                  -0.045434
DER_deltar_tau_lep           -0.242922
dtype: float64
```

```
from sklearn.preprocessing import PowerTransformer

pt = PowerTransformer(method='yeo-johnson')
cols = ["PRI_jet_all_pt", "PRI_tau_pt", "DER_sum_pt", "DER_pt_h",
        "PRI_lep_pt", "DER_pt_tot", "Weight", "PRI_met_sumet",
        "PRI_met", "DER_pt_ratio_lep_tau", "DER_mass_vis",
        "DER_mass_transverse_met_lep"]

X_train[cols] = pt.fit_transform(X_train[cols])
X_test[cols]= pt.transform(X_test[cols])
```

```
skew_vals = X_train.skew(numeric_only=True).sort_values(ascending=False)
print(skew_vals)
```

```
PRI_jet_num                   0.610937
DER_met_phi_centrality        0.151360
PRI_tau_pt                    0.128725
Weight                        0.124396
PRI_lep_pt                    0.098881
DER_sum_pt                    0.061295
PRI_lep_eta                   0.021327
DER_mass_vis                  0.017782
PRI_tau_eta                   0.016492
PRI_tau_phi                   0.013794
DER_pt_ratio_lep_tau          0.012877
PRI_met_phi                   0.010342
```

```
PRI_met_sumet                -0.015714
PRI_met                      -0.028938
PRI_lep_phi                  -0.045434
DER_pt_tot                   -0.061279
DER_pt_h                     -0.132481
PRI_jet_all_pt               -0.156553
DER_mass_transverse_met_lep  -0.176934
DER_deltar_tau_lep           -0.242922
dtype: float64
```

```
X_train.corr()
```

	DER_mass_transverse_met_lep	DER_mass_vis	DER_pt_h	DER_deltar_tau_lep	DER_pt_tot	DER_
DER_mass_transverse_met_lep	1.000000	0.111364	-0.299641	0.054264	-0.026058	-0.
DER_mass_vis	0.111364	1.000000	-0.071886	0.654415	-0.015656	0.
DER_pt_h	-0.299641	-0.071886	1.000000	-0.467085	0.414660	0.
DER_deltar_tau_lep	0.054264	0.654415	-0.467085	1.000000	-0.120137	-0.
DER_pt_tot	-0.026058	-0.015656	0.414660	-0.120137	1.000000	0.
DER_sum_pt	-0.219592	0.144891	0.790776	-0.371284	0.240220	1.
DER_pt_ratio_lep_tau	0.358568	0.017382	-0.043620	0.100828	-0.002907	-0.
DER_met_phi_centrality	-0.455711	-0.087564	0.661162	-0.208123	0.211046	0.
PRI_tau_pt	-0.232022	0.351841	0.239296	-0.140872	0.062360	0.
PRI_tau_eta	-0.003637	0.001560	0.006448	0.002102	0.005489	0.
PRI_tau_phi	0.001987	-0.008568	0.004138	-0.012424	-0.000112	0.
PRI_lep_pt	0.280440	0.416752	0.197853	-0.025904	0.060846	0.
PRI_lep_eta	-0.010176	0.000453	0.017246	-0.001127	0.010912	0.
PRI_lep_phi	0.001117	-0.000938	-0.005041	-0.000162	-0.003427	-0.
PRI_met	0.240189	-0.166505	0.437393	-0.405327	0.168302	0.
PRI_met_phi	-0.015153	0.001353	0.010950	-0.001376	0.000908	0.
PRI_met_sumet	-0.230505	0.047497	0.777961	-0.370524	0.462306	0.
PRI_jet_num	-0.241887	-0.055816	0.720344	-0.349733	0.296466	0.
PRI_jet_all_pt	-0.281707	-0.058360	0.826528	-0.394538	0.219320	0.
Weight	0.493117	0.015138	-0.463001	0.155949	-0.190426	-0.

```
X_train_vif = X_train.loc[:, X_train.nunique() > 1].copy()

target = 'Label'
if target in X_train_vif.columns:
    X_train_vif = X_train_vif.drop(columns=[target])

# VIF calculation
vif_data = pd.DataFrame()
vif_data["Feature"] = X_train_vif.columns
vif_data["VIF"] = [variance_inflation_factor(X_train_vif.values, i) for i in range(X_train_vif.shape[1])]

print(vif_data)
```

```
                       Feature        VIF
0   DER_mass_transverse_met_lep   2.004419
1                  DER_mass_vis   2.380202
2                      DER_pt_h   6.061215
3             DER_deltar_tau_lep   8.060528
4                    DER_pt_tot   1.637768
5                    DER_sum_pt  21.133212
6           DER_pt_ratio_lep_tau  18.427979
7         DER_met_phi_centrality   2.135599
8                    PRI_tau_pt  12.649882
9                   PRI_tau_eta   1.454335
```

```
10                 PRI_tau_phi   1.047010
11                  PRI_lep_pt  10.098926
12                 PRI_lep_eta   1.455220
13                 PRI_lep_phi   1.046974
14                     PRI_met   1.794596
15                 PRI_met_phi   1.002365
16               PRI_met_sumet   4.256301
17                 PRI_jet_num  13.166819
18              PRI_jet_all_pt  16.053289
19                      Weight   1.811286
```

```
X_train=X_train.drop(columns=["DER_sum_pt","DER_pt_ratio_lep_tau","PRI_jet_num"],axis=1)
X_test=X_test.drop(columns=["DER_sum_pt","DER_pt_ratio_lep_tau","PRI_jet_num"],axis=1)
```

```
X_train_vif = X_train.loc[:, X_train.nunique() > 1].copy()

# VIF calculation
vif_data = pd.DataFrame()
vif_data["Feature"] = X_train_vif.columns
vif_data["VIF"] = [variance_inflation_factor(X_train_vif.values, i) for i in range(X_train_vif.shape[1])]

print(vif_data)
```

```
                       Feature       VIF
0   DER_mass_transverse_met_lep  1.974754
1                  DER_mass_vis  1.702645
2                      DER_pt_h  5.864412
3            DER_deltar_tau_lep  1.105035
4                    DER_pt_tot  1.558858
5        DER_met_phi_centrality  2.120195
6                    PRI_tau_pt  1.625269
7                   PRI_tau_eta  1.454325
8                   PRI_tau_phi  1.046944
9                    PRI_lep_pt  1.613037
10                  PRI_lep_eta  1.455176
11                  PRI_lep_phi  1.046948
12                      PRI_met  1.659424
13                  PRI_met_phi  1.002346
14                PRI_met_sumet  3.704738
15              PRI_jet_all_pt  4.159834
16                      Weight  1.790855
```

```
model = XGBClassifier(random_state=42)
model.fit(X_train, y_train.values.ravel())
y_pred = model.predict(X_test)
y_train_pred = model.predict(X_train)
y_test_pred = model.predict(X_test)
print("Training Score",round(accuracy_score(y_train, y_train_pred),2))
print("Testing Score",round(accuracy_score(y_test, y_test_pred),2))
```

```
Training Score 1.0
Testing Score 1.0
```

```
cvs=cross_val_score(model,X_train,y_train,cv=5)

print([f"{round(Score,2)}" for Score in cvs])#scores at each fold
print("Testing Mean",round(cvs.mean(),2))
```

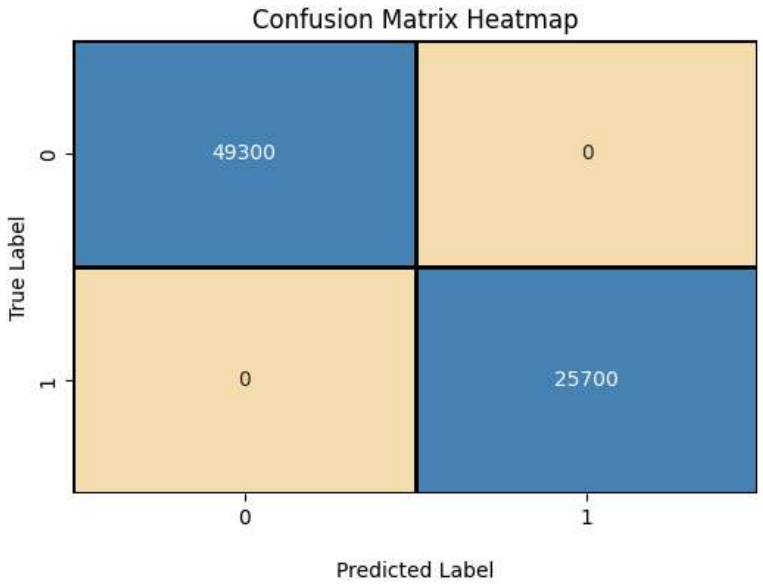
```
['1.0', '1.0', '1.0', '1.0', '1.0']
Testing Mean 1.0
```

```
print("Classification Report:\n", classification_report(y_test, y_pred))
```

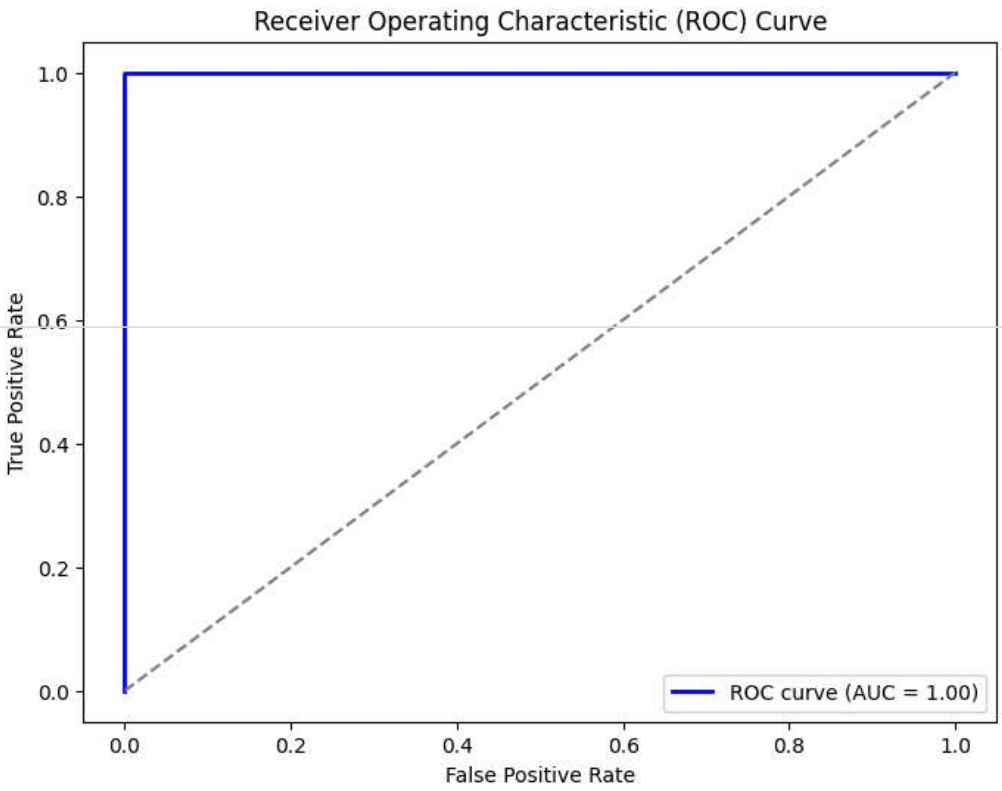
```
Classification Report:
               precision    recall  f1-score   support

           0       1.00      1.00      1.00     49300
           1       1.00      1.00      1.00     25700

    accuracy                           1.00     75000
   macro avg       1.00      1.00      1.00     75000
```

```
weighted avg       1.00      1.00      1.00     75000
```

```
cm = confusion_matrix(y_test, y_test_pred)
colors = ["#F5DEB3", "#4682B4"]
cmap = sns.color_palette(colors, as_cmap=True)

plt.figure(figsize=(6, 4))
sns.heatmap(cm, annot=True, fmt="d", cmap=cmap, cbar=False, linewidths=1, linecolor='black')
plt.xlabel("\nPredicted Label")
plt.ylabel("\nTrue Label")
plt.title("Confusion Matrix Heatmap")
plt.show()
```




```
y_true = y_test
y_pred_proba = model.predict_proba(X_test)[:, 1]

fpr, tpr, thresholds = roc_curve(y_true, y_pred_proba)


roc_auc = auc(fpr, tpr)



plt.figure(figsize=(8, 6))
plt.plot(fpr, tpr, color='b', lw=2, label='ROC curve (AUC = %0.2f)' % roc_auc)
plt.plot([0, 1], [0, 1], color='gray', linestyle='--')
plt.xlabel('False Positive Rate')
plt.ylabel('True Positive Rate')
plt.title('Receiver Operating Characteristic (ROC) Curve')
plt.legend(loc='lower right')
plt.show()
```


Receiver Operating Characteristic (ROC) Curve
True Positive Rate
False Positive Rate
0.0
0.2
0.4
0.6
0.8
1.0
ROC curve (AUC = 1.00)