

1. Firstly, replace all Missing values with relevant figures.

```
import numpy as np
import pandas as pd
df = pd.read_csv('Dataset_Day7.csv')
print(df.info())
df.replace(0, np.nan, inplace=True)
missing_value_percent = df.isna().sum() / len(df)
* 100
print(missing_value_percent)
skewness = df.skew()
print(skewness)
df["Pregnancies"].fillna(df["Pregnancies"].median(),
inplace=True)
df["Glucose"].fillna(df["Glucose"].median(),
inplace=True)
df["BloodPressure"].fillna(df["BloodPressure"].me
an(), inplace=True)
df["BMI"].fillna(df["BMI"].median(),
inplace=True)
df["Outcome"].fillna(df["Outcome"].mean(),
inplace=True)
print(df.info())
```

```

1 C:\Users\tejas\PycharmProjects\pythonProject\venv\scripts\python.exe C:\Users\
tejas\PycharmProjects\pythonProject\START\Day7Q1.py
2 <class 'pandas.core.frame.DataFrame'>
3 RangeIndex: 768 entries, 0 to 767
4 Data columns (total 7 columns):
5 #   Column                Non-Null Count  Dtype
6 ---  ---
7 0   Pregnancies            768 non-null    int64
8 1   Glucose                 768 non-null    int64
9 2   BloodPressure           768 non-null    int64
10 3   BMI                    768 non-null    float64
11 4   DiabetesPedigreeFunction 768 non-null    float64
12 5   Age                    768 non-null    int64
13 6   Outcome                768 non-null    int64
14 dtypes: float64(2), int64(5)
15 memory usage: 42.1 KB
16 None
17 Pregnancies            14.453125
18 Glucose                 0.651042
19 BloodPressure           4.557292
20 BMI                    1.432292
21 DiabetesPedigreeFunction 0.000000
22 Age                    0.000000
23 Outcome                65.104167
24 dtype: float64
25 Pregnancies            0.885535
26 Glucose                 0.530989
27 BloodPressure           0.134153
28 BMI                    0.593970
29 DiabetesPedigreeFunction 1.919911
30 Age                    1.129597
31 Outcome                0.000000
32 dtype: float64
33 <class 'pandas.core.frame.DataFrame'>
34 RangeIndex: 768 entries, 0 to 767
35 Data columns (total 7 columns):
36 #   Column                Non-Null Count  Dtype
37 ---  ---
38 0   Pregnancies            768 non-null    float64
39 1   Glucose                 768 non-null    float64
40 2   BloodPressure           768 non-null    float64
41 3   BMI                    768 non-null    float64
42 4   DiabetesPedigreeFunction 768 non-null    float64
43 5   Age                    768 non-null    int64
44 6   Outcome                768 non-null    float64
45 dtypes: float64(6), int64(1)
46 memory usage: 42.1 KB
47 None
48
49 Process finished with exit code 0

```

2. Then remove all existing outliers and get the final data for classification.

```

import numpy as np
import pandas as pd
df = pd.read_csv('Dataset_Day7.csv')

```

```

print(df.info())
df.replace(0, np.nan, inplace=True)
missing_value_percent = df.isna().sum() / len(df) *
100
print(missing_value_percent)
skewness = df.skew()
print(skewness)
df["Pregnancies"].fillna(df["Pregnancies"].median(),
, inplace=True)
df["Glucose"].fillna(df["Glucose"].median(),
inplace=True)
df["BloodPressure"].fillna(df["BloodPressure"].mean
(), inplace=True)
df["BMI"].fillna(df["BMI"].median(), inplace=True)
df["Outcome"].fillna(df["Outcome"].mean(),
inplace=True)
print(df.info())
OutlierData = pd.DataFrame()
temp = df[["Pregnancies", "Glucose",
"BloodPressure", "BMI", "DiabetesPedigreeFunction" ,
"Age", "Outcome" ]]
for col in ["Pregnancies", "Glucose",
"BloodPressure", "BMI", "DiabetesPedigreeFunction" ,
"Age", "Outcome"]:
    Q1 = temp[col].quantile(0.25) # Gives 25th
Percentile or Q1
    Q3 = temp[col].quantile(0.75) # Gives 75th
Percentile or Q3

    IQR = Q3 - Q1

    UpperBound = Q3 + 1.5 * IQR
    LowerBound = Q1 - 1.5 * IQR

    OutlierData[col] = temp[col][(temp[col] <
LowerBound) | (temp[col] > UpperBound)]
    print(len(OutlierData))
    df_OutlierFree = df.drop(OutlierData.index,
axis=0)
    df_OutlierFree.info()

```

```

cejas\python\Projects\python\Project\STAT\day7\q2.py
2 <class 'pandas.core.frame.DataFrame'>
3 RangeIndex: 768 entries, 0 to 767
4 Data columns (total 7 columns):
5 #   Column                Non-Null Count  Dtype
6 ---  ---
7 0   Pregnancies           768 non-null   int64
8 1   Glucose               768 non-null   int64
9 2   BloodPressure         768 non-null   int64
10 3   BMI                  768 non-null   float64
11 4   DiabetesPedigreeFunction 768 non-null   float64
12 5   Age                  768 non-null   int64
13 6   Outcome              768 non-null   int64
14 dtypes: float64(2), int64(5)
15 memory usage: 42.1 KB
16 None
17 Pregnancies           14.453125
18 Glucose              0.651042
19 BloodPressure        4.557292
20 BMI                 1.432292
21 DiabetesPedigreeFunction 0.000000
22 Age                 0.000000
23 Outcome             65.104167
24 dtype: float64
25 Pregnancies           0.885535
26 Glucose              0.530989
27 BloodPressure        0.134153
28 BMI                 0.593970
29 DiabetesPedigreeFunction 1.919911
30 Age                 1.129597
31 Outcome             0.000000
32 dtype: float64
33 <class 'pandas.core.frame.DataFrame'>
34 RangeIndex: 768 entries, 0 to 767
35 Data columns (total 7 columns):
36 #   Column                Non-Null Count  Dtype
37 ---  ---
38 0   Pregnancies           768 non-null   float64
39 1   Glucose               768 non-null   float64
40 2   BloodPressure         768 non-null   float64
41 3   BMI                  768 non-null   float64
42 4   DiabetesPedigreeFunction 768 non-null   float64
43 5   Age                  768 non-null   int64
44 6   Outcome              768 non-null   float64
45 dtypes: float64(6), int64(1)
46 memory usage: 42.1 KB
47 None
48 14
49 <class 'pandas.core.frame.DataFrame'>
50 Index: 754 entries, 0 to 767
51 Data columns (total 7 columns):
52 #   Column                Non-Null Count  Dtype
53 ---  ---
54 0   Pregnancies           754 non-null   float64
55 1   Glucose               754 non-null   float64
56 2   BloodPressure         754 non-null   float64
57 3   BMI                  754 non-null   float64
58 4   DiabetesPedigreeFunction 754 non-null   float64

```

File - Day7Q2

```
59 5 Age 754 non-null int64
60 6 Outcome 754 non-null float64
61 dtypes: float64(6), int64(1)
62 memory usage: 47.1 KB
63 14
64 <class 'pandas.core.frame.DataFrame'>
65 Index: 754 entries, 0 to 767
66 Data columns (total 7 columns):
67 # Column Non-Null Count Dtype
68 ---
69 0 Pregnancies 754 non-null float64
70 1 Glucose 754 non-null float64
71 2 BloodPressure 754 non-null float64
72 3 BMI 754 non-null float64
73 4 DiabetesPedigreeFunction 754 non-null float64
74 5 Age 754 non-null int64
75 6 Outcome 754 non-null float64
76 dtypes: float64(6), int64(1)
77 memory usage: 47.1 KB
78 14
79 <class 'pandas.core.frame.DataFrame'>
80 Index: 754 entries, 0 to 767
81 Data columns (total 7 columns):
82 # Column Non-Null Count Dtype
83 ---
84 0 Pregnancies 754 non-null float64
85 1 Glucose 754 non-null float64
86 2 BloodPressure 754 non-null float64
87 3 BMI 754 non-null float64
88 4 DiabetesPedigreeFunction 754 non-null float64
89 5 Age 754 non-null int64
90 6 Outcome 754 non-null float64
91 dtypes: float64(6), int64(1)
92 memory usage: 47.1 KB
93 14
94 <class 'pandas.core.frame.DataFrame'>
95 Index: 754 entries, 0 to 767
96 Data columns (total 7 columns):
97 # Column Non-Null Count Dtype
98 ---
99 0 Pregnancies 754 non-null float64
100 1 Glucose 754 non-null float64
101 2 BloodPressure 754 non-null float64
102 3 BMI 754 non-null float64
103 4 DiabetesPedigreeFunction 754 non-null float64
104 5 Age 754 non-null int64
105 6 Outcome 754 non-null float64
106 dtypes: float64(6), int64(1)
107 memory usage: 47.1 KB
108 14
109 <class 'pandas.core.frame.DataFrame'>
110 Index: 754 entries, 0 to 767
111 Data columns (total 7 columns):
112 # Column Non-Null Count Dtype
113 ---
114 0 Pregnancies 754 non-null float64
115 1 Glucose 754 non-null float64
116 2 BloodPressure 754 non-null float64
117 3 BMI 754 non-null float64
```

File - Day7Q2

```
118 4 DiabetesPedigreeFunction 754 non-null float64
119 5 Age 754 non-null int64
120 6 Outcome 754 non-null float64
121 dtypes: float64(6), int64(1)
122 memory usage: 47.1 KB
123 14
124 <class 'pandas.core.frame.DataFrame'>
125 Index: 754 entries, 0 to 767
126 Data columns (total 7 columns):
127 # Column Non-Null Count Dtype
128 --- ---
129 0 Pregnancies 754 non-null float64
130 1 Glucose 754 non-null float64
131 2 BloodPressure 754 non-null float64
132 3 BMI 754 non-null float64
133 4 DiabetesPedigreeFunction 754 non-null float64
134 5 Age 754 non-null int64
135 6 Outcome 754 non-null float64
136 dtypes: float64(6), int64(1)
137 memory usage: 47.1 KB
138 14
139 <class 'pandas.core.frame.DataFrame'>
140 Index: 754 entries, 0 to 767
141 Data columns (total 7 columns):
142 # Column Non-Null Count Dtype
143 --- ---
144 0 Pregnancies 754 non-null float64
145 1 Glucose 754 non-null float64
146 2 BloodPressure 754 non-null float64
147 3 BMI 754 non-null float64
148 4 DiabetesPedigreeFunction 754 non-null float64
149 5 Age 754 non-null int64
150 6 Outcome 754 non-null float64
151 dtypes: float64(6), int64(1)
152 memory usage: 47.1 KB
153
154 Process finished with exit code 0
155
```

2. Split the data into 70% training and 30% testing data. Then, create a logistic regression model with target variable as 'Outcome'.
  - a. Print the default model performance metrics: Accuracy, Precision, Recall, F1Score & AIC

```
import numpy as np
import pandas as pd
from sklearn.model_selection import
train_test_split
from sklearn.linear_model import
LogisticRegression
from sklearn.metrics import accuracy_score,
```

```

precision_score, recall_score, f1_score,
roc_auc_score

df = pd.read_csv('Dataset_Day7.csv')
print(df.info())
missing_value_percent = df.isna().sum() / len(df)
* 100
print(missing_value_percent)
skewness = df.skew()
print(skewness)
df["Glucose"].fillna(df["Glucose"].median(),
inplace=True)
df["BloodPressure"].fillna(df["BloodPressure"].me
an(), inplace=True)
df["BMI"].fillna(df["BMI"].median(),
inplace=True)
df["Outcome"].fillna(df["Outcome"].mean(),
inplace=True)
print(df.info())
OutlierData = pd.DataFrame()
temp = df[["Pregnancies", "Glucose",
"BloodPressure", "BMI",
"DiabetesPedigreeFunction", "Age", "Outcome"]]
for col in ["Pregnancies", "Glucose",
"BloodPressure", "BMI",
"DiabetesPedigreeFunction", "Age", "Outcome"]:
    Q1 = temp[col].quantile(0.25) # Gives 25th
Percentile or Q1
    Q3 = temp[col].quantile(0.75) # Gives 75th
Percentile or Q3

    IQR = Q3 - Q1

    UpperBound = Q3 + 1.5 * IQR
    LowerBound = Q1 - 1.5 * IQR

    OutlierData[col] = temp[col][(temp[col] <
LowerBound) | (temp[col] > UpperBound)]
    print(len(OutlierData))
    df_OutlierFree = df.drop(OutlierData.index,
axis=0)
    df_OutlierFree.info()

```

```

# scaling on the selected continuous columns by
# subtracting the minimum value and dividing by the
# range (maximum -
# minimum) for each column.
continuous_columns = ['Pregnancies', 'Glucose',
'BloodPressure', 'BMI',
'DiabetesPedigreeFunction', 'Age']
df_continuous =
df_OutlierFree[continuous_columns]
df_scaled = df_OutlierFree.copy()
df_scaled[continuous_columns] = (df_continuous -
df_continuous.min()) / (df_continuous.max() -
df_continuous.min())
X = df_OutlierFree.drop('Outcome', axis=1) # all
columns except 'Outcome'
y = df_OutlierFree['Outcome'] # target Variable
X_train, X_test, y_train, y_test =
train_test_split(X, y, test_size=0.3,
random_state=203)
unique_classes = np.unique(y)
print('Number of unique classes:',
len(unique_classes))
logreg = LogisticRegression(random_state=203)
logreg = logreg.fit(X_train, y_train)

y_pred = logreg.predict(X_test) # default
threshold value is 0.5

print("Model Accuracy is
{}%".format(accuracy_score(y_test, y_pred) *
100))
print("Model Precision is
{}%".format(precision_score(y_test, y_pred) *
100))
print("Model Recall is
{}%".format(recall_score(y_test, y_pred) * 100))

f1_score = 2 * precision_score(y_test, y_pred) *
recall_score(y_test, y_pred) / (
precision_score(y_test, y_pred) +
recall_score(y_test, y_pred))

```



```
print("Model F1-Score is {}".format(f1_score *  
100))  
print("Model AUC is:", roc_auc_score(y_test,  
y_pred))
```

```

tejas\PycharmProjects\pythonProject\START\Day7Q3.py
2 <class 'pandas.core.frame.DataFrame'>
3 RangeIndex: 768 entries, 0 to 767
4 Data columns (total 7 columns):
5 #   Column                Non-Null Count  Dtype
6 ---  ---
7 0   Pregnancies            768 non-null   int64
8 1   Glucose                768 non-null   int64
9 2   BloodPressure          768 non-null   int64
10 3   BMI                   768 non-null   float64
11 4   DiabetesPedigreeFunction 768 non-null   float64
12 5   Age                   768 non-null   int64
13 6   Outcome               768 non-null   int64
14 dtypes: float64(2), int64(5)
15 memory usage: 42.1 KB
16 None
17 Pregnancies            0.0
18 Glucose                0.0
19 BloodPressure          0.0
20 BMI                   0.0
21 DiabetesPedigreeFunction 0.0
22 Age                   0.0
23 Outcome               0.0
24 dtype: float64
25 Pregnancies            0.901674
26 Glucose                0.173754
27 BloodPressure          -1.843608
28 BMI                   -0.428982
29 DiabetesPedigreeFunction 1.919911
30 Age                   1.129597
31 Outcome               0.635017
32 dtype: float64
33 <class 'pandas.core.frame.DataFrame'>
34 RangeIndex: 768 entries, 0 to 767
35 Data columns (total 7 columns):
36 #   Column                Non-Null Count  Dtype
37 ---  ---
38 0   Pregnancies            768 non-null   int64
39 1   Glucose                768 non-null   int64
40 2   BloodPressure          768 non-null   int64
41 3   BMI                   768 non-null   float64
42 4   DiabetesPedigreeFunction 768 non-null   float64
43 5   Age                   768 non-null   int64
44 6   Outcome               768 non-null   int64
45 dtypes: float64(2), int64(5)
46 memory usage: 42.1 KB
47 None
48 4
49 <class 'pandas.core.frame.DataFrame'>
50 Index: 764 entries, 0 to 767
51 Data columns (total 7 columns):
52 #   Column                Non-Null Count  Dtype
53 ---  ---
54 0   Pregnancies            764 non-null   int64
55 1   Glucose                764 non-null   int64
56 2   BloodPressure          764 non-null   int64
57 3   BMI                   764 non-null   float64
58 4   DiabetesPedigreeFunction 764 non-null   float64

```

```

59 5 Age 764 non-null int64
60 6 Outcome 764 non-null int64
61 dtypes: float64(2), int64(5)
62 memory usage: 47.8 KB
63 4
64 <class 'pandas.core.frame.DataFrame'>
65 Index: 764 entries, 0 to 767
66 Data columns (total 7 columns):
67 # Column Non-Null Count Dtype
68 ---
69 0 Pregnancies 764 non-null int64
70 1 Glucose 764 non-null int64
71 2 BloodPressure 764 non-null int64
72 3 BMI 764 non-null float64
73 4 DiabetesPedigreeFunction 764 non-null float64
74 5 Age 764 non-null int64
75 6 Outcome 764 non-null int64
76 dtypes: float64(2), int64(5)
77 memory usage: 47.8 KB
78 4
79 <class 'pandas.core.frame.DataFrame'>
80 Index: 764 entries, 0 to 767
81 Data columns (total 7 columns):
82 # Column Non-Null Count Dtype
83 ---
84 0 Pregnancies 764 non-null int64
85 1 Glucose 764 non-null int64
86 2 BloodPressure 764 non-null int64
87 3 BMI 764 non-null float64
88 4 DiabetesPedigreeFunction 764 non-null float64
89 5 Age 764 non-null int64
90 6 Outcome 764 non-null int64
91 dtypes: float64(2), int64(5)
92 memory usage: 47.8 KB
93 4
94 <class 'pandas.core.frame.DataFrame'>
95 Index: 764 entries, 0 to 767
96 Data columns (total 7 columns):
97 # Column Non-Null Count Dtype
98 ---
99 0 Pregnancies 764 non-null int64
100 1 Glucose 764 non-null int64
101 2 BloodPressure 764 non-null int64
102 3 BMI 764 non-null float64
103 4 DiabetesPedigreeFunction 764 non-null float64
104 5 Age 764 non-null int64
105 6 Outcome 764 non-null int64
106 dtypes: float64(2), int64(5)
107 memory usage: 47.8 KB
108 4
109 <class 'pandas.core.frame.DataFrame'>
110 Index: 764 entries, 0 to 767
111 Data columns (total 7 columns):
112 # Column Non-Null Count Dtype
113 ---
114 0 Pregnancies 764 non-null int64
115 1 Glucose 764 non-null int64
116 2 BloodPressure 764 non-null int64
117 3 BMI 764 non-null float64

```

```

118 4 DiabetesPedigreeFunction 764 non-null float64
119 5 Age 764 non-null int64
120 6 Outcome 764 non-null int64
121 dtypes: float64(2), int64(5)
122 memory usage: 47.8 KB
123 4
124 <class 'pandas.core.frame.DataFrame'>
125 Index: 764 entries, 0 to 767
126 Data columns (total 7 columns):
127 # Column Non-Null Count Dtype
128 --- ---
129 0 Pregnancies 764 non-null int64
130 1 Glucose 764 non-null int64
131 2 BloodPressure 764 non-null int64
132 3 BMI 764 non-null float64
133 4 DiabetesPedigreeFunction 764 non-null float64
134 5 Age 764 non-null int64
135 6 Outcome 764 non-null int64
136 dtypes: float64(2), int64(5)
137 memory usage: 47.8 KB
138 4
139 <class 'pandas.core.frame.DataFrame'>
140 Index: 764 entries, 0 to 767
141 Data columns (total 7 columns):
142 # Column Non-Null Count Dtype
143 --- ---
144 0 Pregnancies 764 non-null int64
145 1 Glucose 764 non-null int64
146 2 BloodPressure 764 non-null int64
147 3 BMI 764 non-null float64
148 4 DiabetesPedigreeFunction 764 non-null float64
149 5 Age 764 non-null int64
150 6 Outcome 764 non-null int64
151 dtypes: float64(2), int64(5)
152 memory usage: 47.8 KB
153 Number of unique classes: 2
154 Model Accuracy is 76.52173913043478%
155 Model Precision is 72.58064516129032%
156 Model Recall is 54.87804878048781%
157 Model F1-Score is 62.50000000000014%
158 Model AUC is: 0.7169578114700066
159
160 Process finished with exit code 0
161

```

4. Plot a F1\_score vs threshold curve. Find the threshold for which f1-score is the highest.

```

import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
from sklearn.model_selection import
train_test_split
from sklearn.linear_model import LogisticRegression
from sklearn.metrics import precision_recall_curve

```

```

df = pd.read_csv('Dataset_Day7.csv')
print(df.info())
missing_value_percent = df.isna().sum() / len(df) *
100
print(missing_value_percent)
skewness = df.skew()
print(skewness)
df["Pregnancies"].fillna(df["Pregnancies"].median(),
, inplace=True)
df["Glucose"].fillna(df["Glucose"].median(),
inplace=True)
df["BloodPressure"].fillna(df["BloodPressure"].mean
(), inplace=True)
df["BMI"].fillna(df["BMI"].median(), inplace=True)
df["Outcome"].fillna(df["Outcome"].mean(),
inplace=True)
print(df.info())
OutlierData = pd.DataFrame()
temp = df[["Pregnancies", "Glucose",
"BloodPressure", "BMI", "DiabetesPedigreeFunction",
"Age", "Outcome"]]
for col in ["Pregnancies", "Glucose",
"BloodPressure", "BMI", "DiabetesPedigreeFunction",
"Age", "Outcome"]:
    Q1 = temp[col].quantile(0.25) # Gives 25th
Percentile or Q1
    Q3 = temp[col].quantile(0.75) # Gives 75th
Percentile or Q3

    IQR = Q3 - Q1

    UpperBound = Q3 + 1.5 * IQR
    LowerBound = Q1 - 1.5 * IQR

    OutlierData[col] = temp[col][(temp[col] <
LowerBound) | (temp[col] > UpperBound)]
    print(len(OutlierData))
    df_OutlierFree = df.drop(OutlierData.index,
axis=0)
    df_OutlierFree.info()
# scaling on the selected continuous columns by
subtracting the minimum value and dividing by the

```

```

range (maximum -
# minimum) for each column.
continuous_columns = ['Pregnancies', 'Glucose',
'BloodPressure', 'BMI', 'DiabetesPedigreeFunction',
'Age']
df_continuous = df_OutlierFree[continuous_columns]
df_scaled = df_OutlierFree.copy()
df_scaled[continuous_columns] = (df_continuous -
df_continuous.min()) / (df_continuous.max() -
df_continuous.min())
X = df_OutlierFree.drop('Outcome', axis=1) # all
columns except 'Outcome'
y = df_OutlierFree['Outcome'] # target Variable
X_train, X_test, y_train, y_test =
train_test_split(X, y, test_size=0.3,
random_state=203)
unique_classes = np.unique(y)
print('Number of unique classes:',
len(unique_classes))
logreg = LogisticRegression(random_state=203)
logreg = logreg.fit(X_train, y_train)
y_pred = logreg.predict(X_test) # default
threshold value is 0.5
y_scores = logreg.predict_proba(X_test)[: , 1]
# display(y_scores)

prec, rec, tre = precision_recall_curve(y_test,
y_scores)

plt.plot(tre, prec[:-1], 'r--', label='Precision')
plt.plot(tre, rec[:-1], 'b--', label='Recall')

f_score = (2 * prec * rec) / (prec + rec)

plt.plot(tre, f_score[:-1], 'g--', label='F1-
Score')

plt.xlabel('Threshold range')
plt.legend(loc='upper left')
plt.show()

index = np.where(f_score == max(f_score))

```

```
print("Optimum Threshold for max precision and  
recall is {}".format(tre[index]))
```

```

tejas\PycharmProjects\pythonProject\SIARI\Day7Q4.py
2 <class 'pandas.core.frame.DataFrame'>
3 RangeIndex: 768 entries, 0 to 767
4 Data columns (total 7 columns):
5 #   Column                Non-Null Count  Dtype
6 ---  -
7 0   Pregnancies            768 non-null    int64
8 1   Glucose                768 non-null    int64
9 2   BloodPressure          768 non-null    int64
10 3   BMI                   768 non-null    float64
11 4   DiabetesPedigreeFunction 768 non-null    float64
12 5   Age                   768 non-null    int64
13 6   Outcome               768 non-null    int64
14 dtypes: float64(2), int64(5)
15 memory usage: 42.1 KB
16 None
17 Pregnancies            0.0
18 Glucose                0.0
19 BloodPressure          0.0
20 BMI                   0.0
21 DiabetesPedigreeFunction 0.0
22 Age                   0.0
23 Outcome               0.0
24 dtype: float64
25 Pregnancies            0.901674
26 Glucose                0.173754
27 BloodPressure          -1.843608
28 BMI                   -0.428982
29 DiabetesPedigreeFunction 1.919911
30 Age                   1.129597
31 Outcome               0.635017
32 dtype: float64
33 <class 'pandas.core.frame.DataFrame'>
34 RangeIndex: 768 entries, 0 to 767
35 Data columns (total 7 columns):
36 #   Column                Non-Null Count  Dtype
37 ---  -
38 0   Pregnancies            768 non-null    int64
39 1   Glucose                768 non-null    int64
40 2   BloodPressure          768 non-null    int64
41 3   BMI                   768 non-null    float64
42 4   DiabetesPedigreeFunction 768 non-null    float64
43 5   Age                   768 non-null    int64
44 6   Outcome               768 non-null    int64
45 dtypes: float64(2), int64(5)
46 memory usage: 42.1 KB
47 None
48 4
49 <class 'pandas.core.frame.DataFrame'>
50 Index: 764 entries, 0 to 767
51 Data columns (total 7 columns):
52 #   Column                Non-Null Count  Dtype
53 ---  -
54 0   Pregnancies            764 non-null    int64
55 1   Glucose                764 non-null    int64
56 2   BloodPressure          764 non-null    int64
57 3   BMI                   764 non-null    float64
58 4   DiabetesPedigreeFunction 764 non-null    float64

```



```

59 5 Age 764 non-null int64
60 6 Outcome 764 non-null int64
61 dtypes: float64(2), int64(5)
62 memory usage: 47.8 KB
63 4
64 <class 'pandas.core.frame.DataFrame'>
65 Index: 764 entries, 0 to 767
66 Data columns (total 7 columns):
67 # Column Non-Null Count Dtype
68 ---
69 0 Pregnancies 764 non-null int64
70 1 Glucose 764 non-null int64
71 2 BloodPressure 764 non-null int64
72 3 BMI 764 non-null float64
73 4 DiabetesPedigreeFunction 764 non-null float64
74 5 Age 764 non-null int64
75 6 Outcome 764 non-null int64
76 dtypes: float64(2), int64(5)
77 memory usage: 47.8 KB
78 4
79 <class 'pandas.core.frame.DataFrame'>
80 Index: 764 entries, 0 to 767
81 Data columns (total 7 columns):
82 # Column Non-Null Count Dtype
83 ---
84 0 Pregnancies 764 non-null int64
85 1 Glucose 764 non-null int64
86 2 BloodPressure 764 non-null int64
87 3 BMI 764 non-null float64
88 4 DiabetesPedigreeFunction 764 non-null float64
89 5 Age 764 non-null int64
90 6 Outcome 764 non-null int64
91 dtypes: float64(2), int64(5)
92 memory usage: 47.8 KB
93 4
94 <class 'pandas.core.frame.DataFrame'>
95 Index: 764 entries, 0 to 767
96 Data columns (total 7 columns):
97 # Column Non-Null Count Dtype
98 ---
99 0 Pregnancies 764 non-null int64
100 1 Glucose 764 non-null int64
101 2 BloodPressure 764 non-null int64
102 3 BMI 764 non-null float64
103 4 DiabetesPedigreeFunction 764 non-null float64
104 5 Age 764 non-null int64
105 6 Outcome 764 non-null int64
106 dtypes: float64(2), int64(5)
107 memory usage: 47.8 KB
108 4
109 <class 'pandas.core.frame.DataFrame'>
110 Index: 764 entries, 0 to 767
111 Data columns (total 7 columns):
112 # Column Non-Null Count Dtype
113 ---
114 0 Pregnancies 764 non-null int64
115 1 Glucose 764 non-null int64
116 2 BloodPressure 764 non-null int64
117 3 BMI 764 non-null float64

```

File - Day7Q4

```
118 4 DiabetesPedigreeFunction 764 non-null float64
119 5 Age 764 non-null int64
120 6 Outcome 764 non-null int64
121 dtypes: float64(2), int64(5)
122 memory usage: 47.8 KB
123 4
124 <class 'pandas.core.frame.DataFrame'>
125 Index: 764 entries, 0 to 767
126 Data columns (total 7 columns):
127 # Column Non-Null Count Dtype
128 ---
129 0 Pregnancies 764 non-null int64
130 1 Glucose 764 non-null int64
131 2 BloodPressure 764 non-null int64
132 3 BMI 764 non-null float64
133 4 DiabetesPedigreeFunction 764 non-null float64
134 5 Age 764 non-null int64
135 6 Outcome 764 non-null int64
136 dtypes: float64(2), int64(5)
137 memory usage: 47.8 KB
138 4
139 <class 'pandas.core.frame.DataFrame'>
140 Index: 764 entries, 0 to 767
141 Data columns (total 7 columns):
142 # Column Non-Null Count Dtype
143 ---
144 0 Pregnancies 764 non-null int64
145 1 Glucose 764 non-null int64
146 2 BloodPressure 764 non-null int64
147 3 BMI 764 non-null float64
148 4 DiabetesPedigreeFunction 764 non-null float64
149 5 Age 764 non-null int64
150 6 Outcome 764 non-null int64
151 dtypes: float64(2), int64(5)
152 memory usage: 47.8 KB
153 Number of unique classes: 2
154 Optimum Threshold for max precision and recall is [0.26995921]
155
156 Process finished with exit code 0
157
```

Figure 1

