```
                                                     ASSIGNMENT 5
#AIM:
1. Logistic Regression
2. Differentiate between Linear and Logistic Regression
3. Sigmoid Function
4. Types of LogisticRegression
5. Confusion Matrix Evaluation Metrics

import pandas as pd

import numpy as np

import seaborn as sns

import matplotlib.pyplot as plt

from sklearn.model_selection import train_test_split

from sklearn.preprocessing import StandardScaler

from sklearn.linear_model import LogisticRegression

from sklearn.metrics import accuracy_score, confusion_matrix,
classification_report

df=pd.read_csv("C:\\Users\\Welcome\\Desktop\\diabetes.csv")

print(df.head())

   Pregnancies  Glucose  BloodPressure  SkinThickness  Insulin
BMI  \
0            6      148             72             35        0  33.6

1            1       85             66             29        0  26.6

2            8      183             64              0        0  23.3

3            1       89             66             23       94  28.1

4            0      137             40             35      168  43.1


   DiabetesPedigreeFunction  Age  Outcome
0                     0.627   50        1
1                     0.351   31        0
2                     0.672   32        1
3                     0.167   21        0
4                     2.288   33        1

if df.select_dtypes(include=['object']).shape[1] > 0:
    df = pd.get_dummies(df, drop_first=True)

df.dropna(inplace=True)
```

```
cov_matrix = df.cov()
print("Covariance Matrix:\n", cov_matrix)

Covariance Matrix:
                           Pregnancies      Glucose  BloodPressure  \
Pregnancies                  11.354056    13.947131       9.214538
Glucose                      13.947131  1022.248314      94.430956
BloodPressure                 9.214538    94.430956     374.647271
SkinThickness                -4.390041    29.239183      64.029396
Insulin                     -28.555231  1220.935799     198.378412
BMI                           0.469774    55.726987      43.004695
DiabetesPedigreeFunction     -0.037426     1.454875       0.264638
Age                          21.570620    99.082805      54.523453
Outcome                       0.356618     7.115079       0.600697

                          SkinThickness       Insulin         BMI  \
Pregnancies                   -4.390041    -28.555231    0.469774
Glucose                       29.239183   1220.935799   55.726987
BloodPressure                 64.029396    198.378412   43.004695
SkinThickness                254.473245    802.979941   49.373869
Insulin                      802.979941  13281.180078  179.775172
BMI                           49.373869    179.775172   62.159984
DiabetesPedigreeFunction       0.972136      7.066681    0.367405
Age                          -21.381023    -57.143290    3.360330
Outcome                        0.568747      7.175671    1.100638

                          DiabetesPedigreeFunction         Age
Outcome
Pregnancies                              -0.037426   21.570620
0.356618
Glucose                                   1.454875   99.082805
7.115079
BloodPressure                             0.264638   54.523453
0.600697
SkinThickness                             0.972136  -21.381023
0.568747
Insulin                                   7.066681  -57.143290
7.175671
BMI                                       0.367405    3.360330
1.100638
DiabetesPedigreeFunction                  0.109779    0.130772
0.027472
Age                                       0.130772  138.303046
1.336953
Outcome                                   0.027472    1.336953
0.227483

 X = df.drop(columns=["Outcome"])
y = df["Outcome"]
```

```
xtrain, xtest, ytrain, ytest = train_test_split(X, y, test_size=0.2,
random_state=42)

scaler = StandardScaler()
xtrain = scaler.fit_transform(xtrain)
xtest = scaler.transform(xtest)

logreg = LogisticRegression()
logreg.fit(xtrain, ytrain)
```

```
LogisticRegression()
```

```
y_pred_train = logreg.predict(xtrain)
y_pred_test = logreg.predict(xtest)

train_accuracy = accuracy_score(ytrain, y_pred_train)
test_accuracy = accuracy_score(ytest, y_pred_test)
conf_matrix = confusion_matrix(ytest, y_pred_test)
class_report = classification_report(ytest, y_pred_test)

print("Training Accuracy:", train_accuracy)
print("Testing Accuracy:", test_accuracy)
print("Confusion Matrix:\n", conf_matrix)
print("Classification Report:\n", class_report)
```

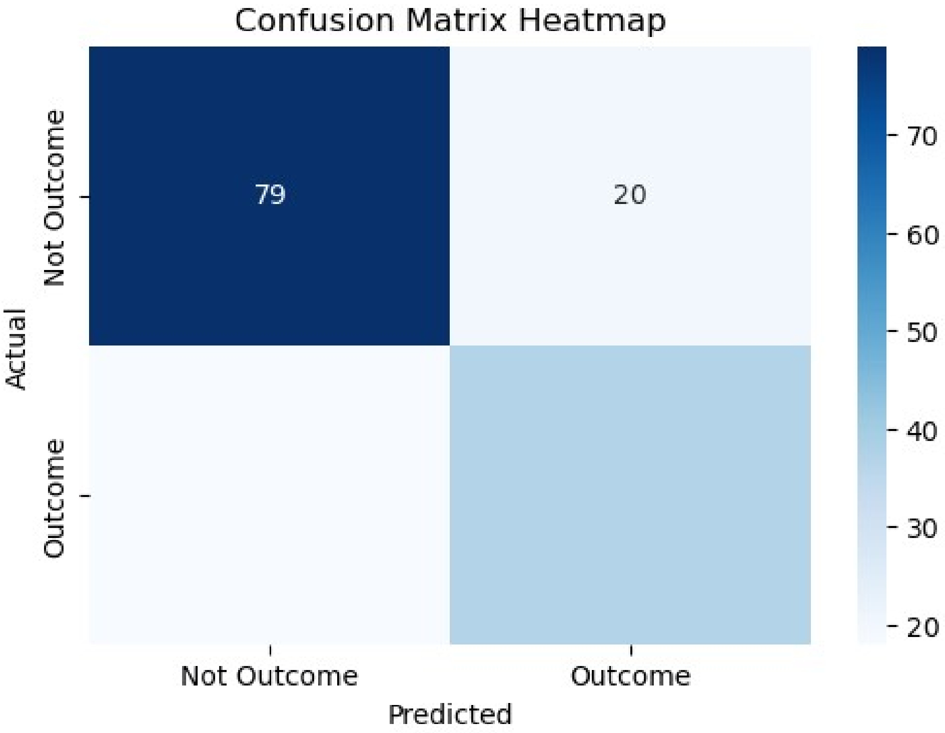
```
Training Accuracy: 0.7703583061889251
Testing Accuracy: 0.7532467532467533
Confusion Matrix:
 [[79 20]
 [18 37]]
Classification Report:
               precision    recall  f1-score   support

           0       0.81      0.80      0.81        99
           1       0.65      0.67      0.66        55

    accuracy                           0.75       154
   macro avg       0.73      0.74      0.73       154
weighted avg       0.76      0.75      0.75       154
```

```
plt.figure(figsize=(6,4))
sns.heatmap(conf_matrix, annot=True, fmt='d', cmap='Blues',
xticklabels=['Not Outcome', 'Outcome'], yticklabels=['Not Outcome',
'Outcome'])
plt.xlabel('Predicted')
plt.ylabel('Actual')
plt.title('Confusion Matrix Heatmap')
```

```
Text(0.5, 1.0, 'Confusion Matrix Heatmap')
```



```
name:pratik pate
roll no:13258
```