diabetes prediction using logistic regression

In [1]:
```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
```

In [2]:
```
ds = pd.read_csv('diabetes.csv')
```

In [3]:
```
ds.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 768 entries, 0 to 767
Data columns (total 9 columns):
 #   Column                    Non-Null Count  Dtype  
---  ------                    --------------  -----  
 0   Pregnancies               768 non-null    int64  
 1   Glucose                   768 non-null    int64  
 2   BloodPressure             768 non-null    int64  
 3   SkinThickness             768 non-null    int64  
 4   Insulin                   768 non-null    int64  
 5   BMI                       768 non-null    float64
 6   DiabetesPedigreeFunction  768 non-null    float64
 7   Age                       768 non-null    int64  
 8   Outcome                   768 non-null    int64  
dtypes: float64(2), int64(7)
memory usage: 54.1 KB
```

In [6]:
```
ds.describe()
```

Out[6]:

	Pregnancies	Glucose	BloodPressure	SkinThickness	Insulin	BMI	Dia
count	768.000000	768.000000	768.000000	768.000000	768.000000	768.000000	
mean	3.845052	120.894531	69.105469	20.536458	79.799479	31.992578	
std	3.369578	31.972618	19.355807	15.952218	115.244002	7.884160	
min	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	
25%	1.000000	99.000000	62.000000	0.000000	0.000000	27.300000	
50%	3.000000	117.000000	72.000000	23.000000	30.500000	32.000000	
75%	6.000000	140.250000	80.000000	32.000000	127.250000	36.600000	
max	17.000000	199.000000	122.000000	99.000000	846.000000	67.100000	

In [7]:
```
ds.isnull().sum()
```

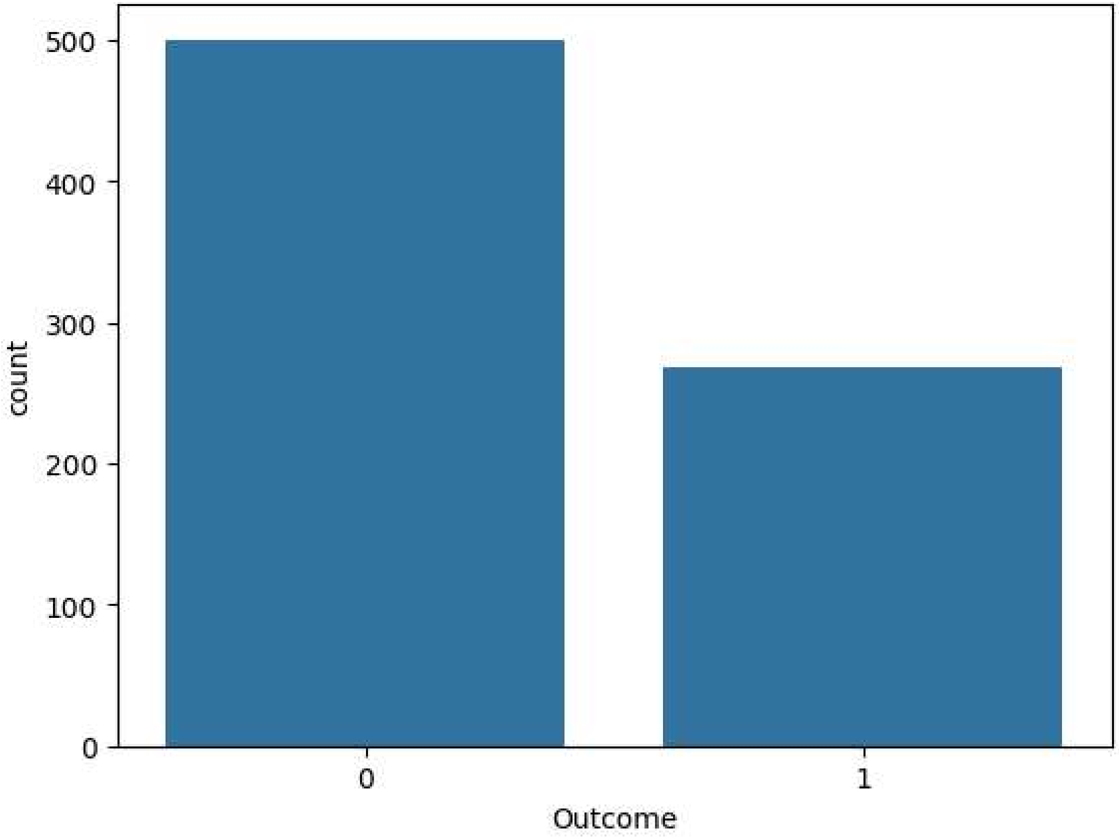
```
Out[7]: Pregnancies                 0
        Glucose                     0
        BloodPressure               0
        SkinThickness               0
        Insulin                     0
        BMI                         0
        DiabetesPedigreeFunction    0
        Age                         0
        Outcome                     0
        dtype: int64
```

```
In [8]: ds.duplicated().sum()
```

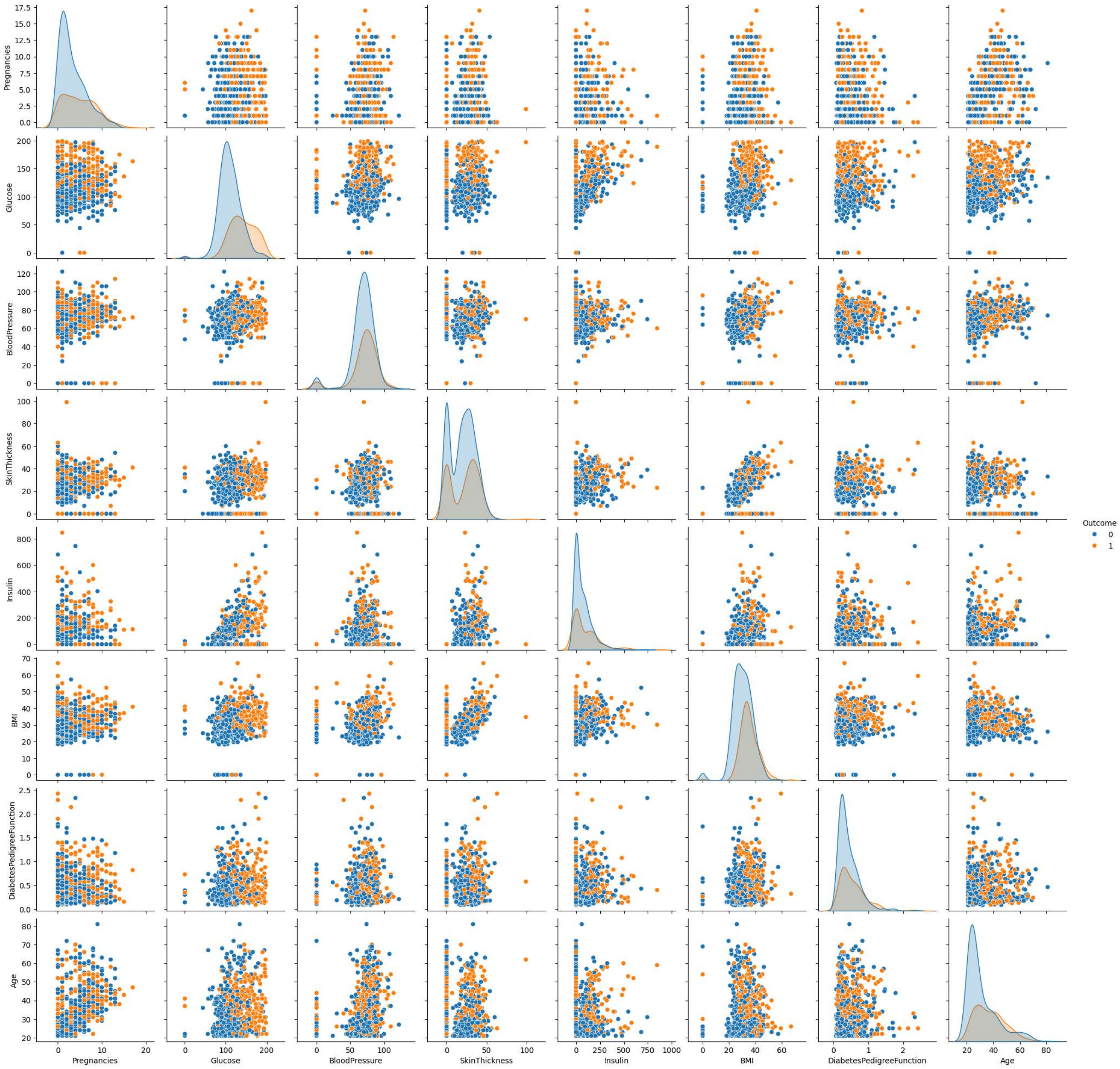
```
Out[8]: np.int64(0)
```

```
In [9]: sns.countplot(x="Outcome",data=ds)
```

```
Out[9]: <Axes: xlabel='Outcome', ylabel='count'>
```

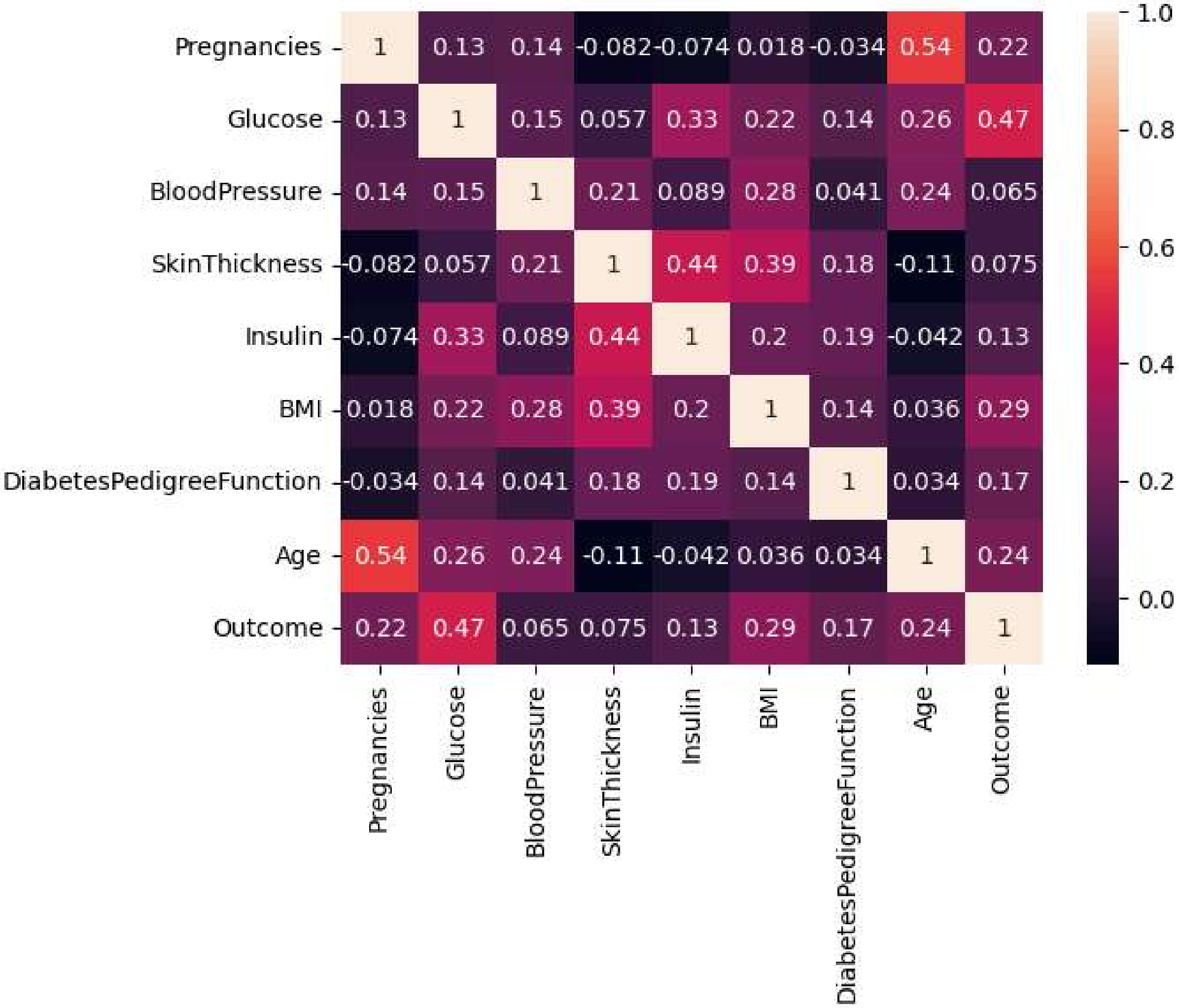


```
In [10]: sns.pairplot(data=ds,hue="Outcome")
         plt.show()
```



In [11]:

```
sns.heatmap(ds.corr(),annot = True)
plt.show()
```



	Pregnancies	Glucose	BloodPressure	SkinThickness	Insulin	BMI	DiabetesPedigreeFunction	Age	Outcome
Pregnancies	1	0.13	0.14	-0.082	-0.074	0.018	-0.034	0.54	0.22
Glucose	0.13	1	0.15	0.057	0.33	0.22	0.14	0.26	0.47
BloodPressure	0.14	0.15	1	0.21	0.089	0.28	0.041	0.24	0.065
SkinThickness	-0.082	0.057	0.21	1	0.44	0.39	0.18	-0.11	0.075
Insulin	-0.074	0.33	0.089	0.44	1	0.2	0.19	-0.042	0.13
BMI	0.018	0.22	0.28	0.39	0.2	1	0.14	0.036	0.29
DiabetesPedigreeFunction	-0.034	0.14	0.041	0.18	0.19	0.14	1	0.034	0.17
Age	0.54	0.26	0.24	-0.11	-0.042	0.036	0.034	1	0.24
Outcome	0.22	0.47	0.065	0.075	0.13	0.29	0.17	0.24	1

In [13]:

```
ds_new = ds
ds_new[["Glucose", "BloodPressure", "SkinThickness", "Insulin", "BMI"]] = ds_new[["
```

In [14]:

```
ds_new["Glucose"].fillna(ds_new["Glucose"].mean() , inplace = True)
ds_new["BloodPressure"].fillna(ds_new["BloodPressure"].mean() , inplace = True)
ds_new["Insulin"].fillna(ds_new["Insulin"].mean() , inplace = True)
ds_new["BMI"].fillna(ds_new["BMI"].mean() , inplace = True)
ds_new["SkinThickness"].fillna(ds_new["SkinThickness"].mean() , inplace = True)
```

```
C:\Users\Aryansh Pathak\AppData\Local\Temp\ipykernel_13604\1196353376.py:1: FutureWa
rning: A value is trying to be set on a copy of a DataFrame or Series through chaine
d assignment using an inplace method.
The behavior will change in pandas 3.0. This inplace method will never work because
the intermediate object on which we are setting values always behaves as a copy.

For example, when doing 'df[col].method(value, inplace=True)', try using 'df.method
({col: value}, inplace=True)' or df[col] = df[col].method(value) instead, to perform
the operation inplace on the original object.


  ds_new["Glucose"].fillna(ds_new["Glucose"].mean() , inplace = True)
C:\Users\Aryansh Pathak\AppData\Local\Temp\ipykernel_13604\1196353376.py:2: FutureWa
rning: A value is trying to be set on a copy of a DataFrame or Series through chaine
d assignment using an inplace method.
The behavior will change in pandas 3.0. This inplace method will never work because
the intermediate object on which we are setting values always behaves as a copy.

For example, when doing 'df[col].method(value, inplace=True)', try using 'df.method
({col: value}, inplace=True)' or df[col] = df[col].method(value) instead, to perform
the operation inplace on the original object.


  ds_new["BloodPressure"].fillna(ds_new["BloodPressure"].mean() , inplace = True)
C:\Users\Aryansh Pathak\AppData\Local\Temp\ipykernel_13604\1196353376.py:3: FutureWa
rning: A value is trying to be set on a copy of a DataFrame or Series through chaine
d assignment using an inplace method.
The behavior will change in pandas 3.0. This inplace method will never work because
the intermediate object on which we are setting values always behaves as a copy.

For example, when doing 'df[col].method(value, inplace=True)', try using 'df.method
({col: value}, inplace=True)' or df[col] = df[col].method(value) instead, to perform
the operation inplace on the original object.


  ds_new["Insulin"].fillna(ds_new["Insulin"].mean() , inplace = True)
C:\Users\Aryansh Pathak\AppData\Local\Temp\ipykernel_13604\1196353376.py:4: FutureWa
rning: A value is trying to be set on a copy of a DataFrame or Series through chaine
d assignment using an inplace method.
The behavior will change in pandas 3.0. This inplace method will never work because
the intermediate object on which we are setting values always behaves as a copy.

For example, when doing 'df[col].method(value, inplace=True)', try using 'df.method
({col: value}, inplace=True)' or df[col] = df[col].method(value) instead, to perform
the operation inplace on the original object.


  ds_new["BMI"].fillna(ds_new["BMI"].mean() , inplace = True)
C:\Users\Aryansh Pathak\AppData\Local\Temp\ipykernel_13604\1196353376.py:5: FutureWa
rning: A value is trying to be set on a copy of a DataFrame or Series through chaine
d assignment using an inplace method.
The behavior will change in pandas 3.0. This inplace method will never work because
the intermediate object on which we are setting values always behaves as a copy.

For example, when doing 'df[col].method(value, inplace=True)', try using 'df.method
({col: value}, inplace=True)' or df[col] = df[col].method(value) instead, to perform
```

```
the operation inplace on the original object.


  ds_new["SkinThickness"].fillna(ds_new["SkinThickness"].mean() , inplace = True)
```

In [15]:

```
ds_new.isnull().sum()
```

Out[15]:

```
Pregnancies                 0
Glucose                     0
BloodPressure               0
SkinThickness               0
Insulin                     0
BMI                         0
DiabetesPedigreeFunction    0
Age                         0
Outcome                     0
dtype: int64
```

In [16]:

```
y = ds_new['Outcome']
x = ds_new.drop('Outcome' , axis = 1)
```

In [17]:

```
from sklearn.model_selection import train_test_split
x_train , x_test , y_train , y_test = train_test_split(x, y , test_size = 0.20 , ra
```

In [18]:

```
from sklearn.linear_model import LogisticRegression
model = LogisticRegression()
model.fit(x_train , y_train)
y_predict = model.predict(x_test)
```

```
C:\Users\Aryansh Pathak\AppData\Local\Programs\Python\Python312\Lib\site-packages\sk
learn\linear_model\_logistic.py:469: ConvergenceWarning: lbfgs failed to converge (s
tatus=1):
STOP: TOTAL NO. of ITERATIONS REACHED LIMIT.

Increase the number of iterations (max_iter) or scale the data as shown in:
    https://scikit-learn.org/stable/modules/preprocessing.html
Please also refer to the documentation for alternative solver options:
    https://scikit-learn.org/stable/modules/linear_model.html#logistic-regression
  n_iter_i = _check_optimize_result(
```

In [19]:

```
y_predict
```

Out[19]:

```
array([1, 0, 0, 0, 0, 0, 0, 1, 0, 1, 0, 0, 0, 0, 0, 0, 1, 0, 1, 0, 0, 1,
       0, 1, 0, 0, 1, 0, 0, 0, 0, 0, 1, 1, 1, 0, 0, 0, 1, 1, 0, 0, 0, 0,
       0, 0, 0, 0, 1, 0, 1, 1, 0, 0, 0, 0, 1, 0, 1, 0, 1, 0, 0, 1, 0, 0,
       1, 0, 0, 1, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
       0, 0, 1, 1, 0, 0, 0, 1, 1, 1, 1, 0, 0, 0, 0, 0, 1, 0, 1, 0, 1, 0,
       1, 1, 0, 0, 0, 0, 0, 0, 1, 0, 1, 0, 0, 1, 0, 1, 1, 1, 0, 0, 0, 1,
       0, 1, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0])
```

In [20]:

```
from sklearn.metrics import confusion_matrix
cm = confusion_matrix(y_test , y_predict)
cm
```

Out[20]:

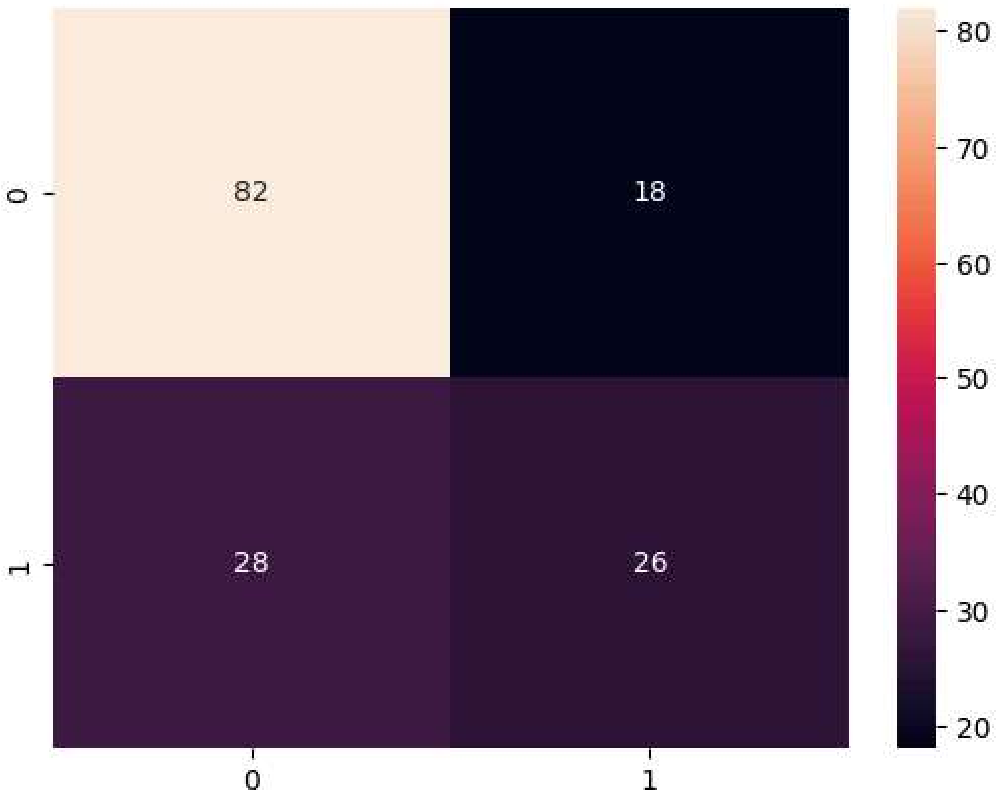
```
array([[82, 18],
       [28, 26]])
```

In [21]:

```
sns.heatmap(pd.DataFrame(cm), annot=True)
```

Out[21]:

```
<Axes: >
```



In [22]:

```
from sklearn.metrics import accuracy_score
```

In [23]:

```
accuracy = accuracy_score(y_test,y_predict)
accuracy
```

Out[23]:

```
0.7012987012987013
```

In [24]:

```
y_predict = model.predict([[1,148,72,38,71.789,13.6,0.927,10]])
print(y_predict)
if y_predict==1:
    print("Diabetic")
else:
    print("Non Diabetic")
```

```
[0]
Non Diabetic
```

```
C:\Users\Aryansh Pathak\AppData\Local\Programs\Python\Python312\Lib\site-packages\sk
learn\base.py:493: UserWarning: X does not have valid feature names, but LogisticReg
ression was fitted with feature names
  warnings.warn(
```

In [ ]: