In [1]:
```
import pandas as pd
import seaborn as sns
```

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

In [3]:
```
df
```

Out[3]:

	Pregnancies	Glucose	BloodPressure	SkinThickness	Insulin	BMI	Pedigree	Age	Outcom
0	6	148	72	35	0	33.6	0.627	50	
1	1	85	66	29	0	26.6	0.351	31	
2	8	183	64	0	0	23.3	0.672	32	
3	1	89	66	23	94	28.1	0.167	21	
4	0	137	40	35	168	43.1	2.288	33	
...	...	...	...	...	...	...	...	...	
763	10	101	76	48	180	32.9	0.171	63	
764	2	122	70	27	0	36.8	0.340	27	
765	5	121	72	23	112	26.2	0.245	30	
766	1	126	60	0	0	30.1	0.349	47	
767	1	93	70	31	0	30.4	0.315	23	

768 rows × 9 columns

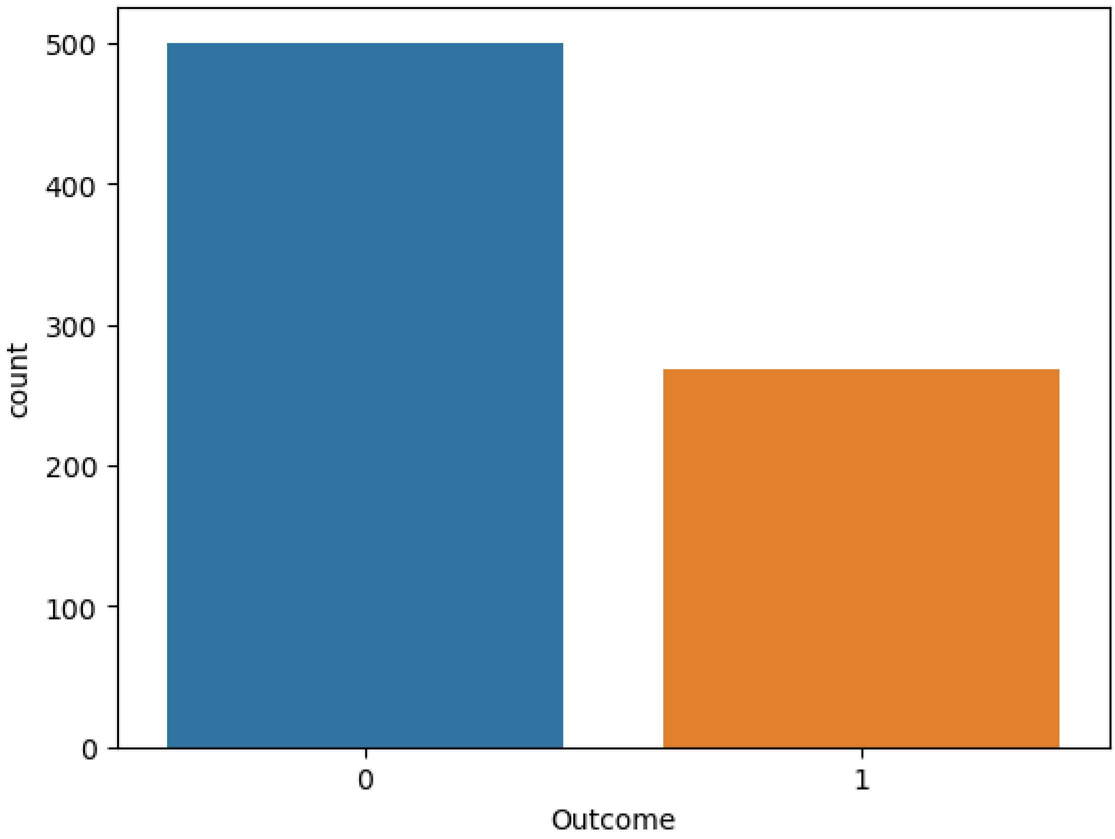
In [4]:
```
x=df.drop('Outcome',axis=1)
y=df['Outcome']
```

In [5]:

```
sns.countplot(x=y)
```

Out[5]:

```
<Axes: xlabel='Outcome', ylabel='count'>
```



In [6]:

```
from sklearn.preprocessing import scale
x=scale(x)
```

In [7]:

```
from sklearn.model_selection import train_test_split
x_train, x_test, y_train, y_test = train_test_split(
    x,y, test_size=0.25, random_state=0)
```

In [ ]:

In [ ]:

In [8]:

```
from sklearn.neighbors import KNeighborsClassifier
```

In [9]:

```
knn=KNeighborsClassifier(n_neighbors=5)
```

In [10]:
```
knn.fit(x_train,y_train)
```

Out[10]:
```
▾ KNeighborsClassifier
KNeighborsClassifier()
```

In [11]:
```
y_pred=knn.predict(x_test)
```

In [12]:
```
from sklearn import metrics
```

In [13]:
```
cs=metrics.confusion_matrix(y_test,y_pred)
print(cs)
```

```
[[115  15]
 [ 22  40]]
```

In [14]:
```
print("Accuracy",metrics.accuracy_score(y_test,y_pred))
```

```
Accuracy 0.8072916666666666
```

In [15]:
```
total_misclassified=cs[0,1]+cs[1,0]
print(total_misclassified)
total_examples=cs[0,0]+cs[0,1]+cs[1,0]+cs[1,1]
print(total_examples)
print("Error rate",total_misclassified/total_examples)
print("Error rate",1-metrics.accuracy_score(y_test,y_pred))
```

```
37
192
Error rate 0.19270833333333334
Error rate 0.19270833333333337
```

In [16]:
```
print("Precision score",metrics.precision_score(y_test,y_pred))
```

```
Precision score 0.7272727272727273
```

In [17]:
```
print("Recall score",metrics.recall_score(y_test,y_pred))
```

```
Recall score 0.6451612903225806
```

In [21]:
```
print("Classification Report",metrics.classification_report(y_test,y_pred))
```

```
Classification Report               precision    recall  f1-score   support

           0       0.84      0.88      0.86       130
           1       0.73      0.65      0.68        62

    accuracy                           0.81       192
   macro avg       0.78      0.76      0.77       192
weighted avg       0.80      0.81      0.80       192
```

In [ ]: