In [2]:
```
import pandas as pd
import seaborn as sns
```

In [3]:
```
df=pd.read_csv("diabetes.csv")
df
```

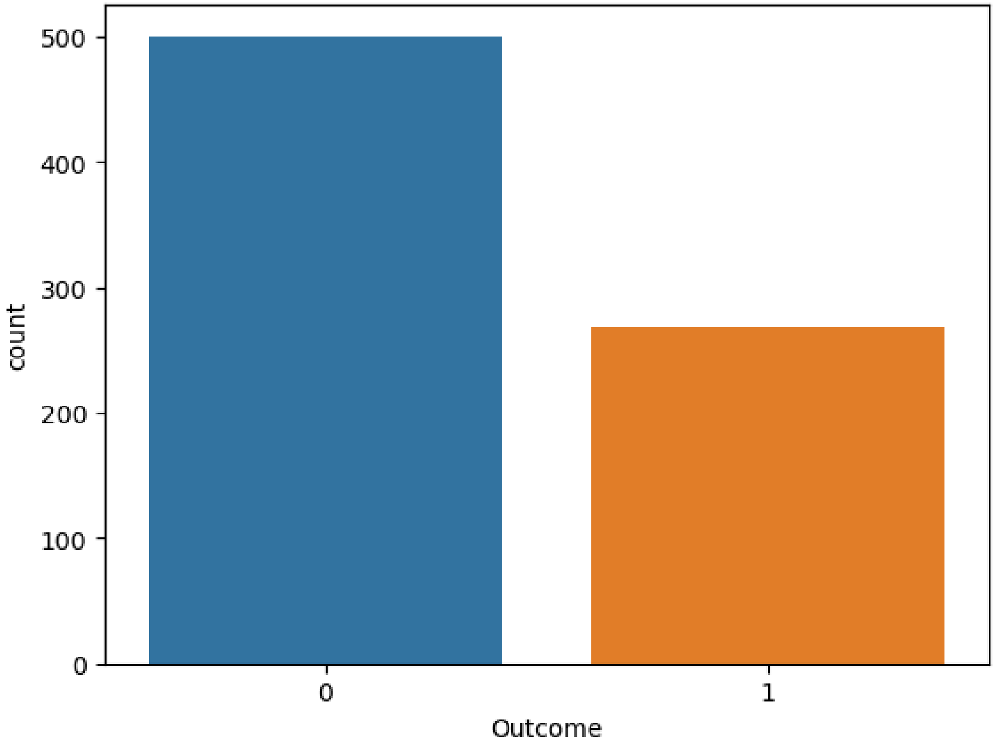
Out[3]:

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

768 rows × 9 columns

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

In [5]:
```
sns.countplot(x=y);
```



In [6]:
```
y.value_counts()
```

Out[6]:
```
Outcome
0    500
1    268
Name: count, dtype: int64
```

In [7]:
```
from sklearn.preprocessing import MinMaxScaler
scaler = MinMaxScaler()
x_scaled = scaler.fit_transform(x)
```

In [8]:
```
from sklearn.model_selection import train_test_split
x_train,x_test,y_train,y_test = train_test_split(x_scaled,y,random_state=42,test_size=0.3
```

In [9]:
```
x.shape
```

Out[9]:
```
(768, 8)
```

In [10]:
```
x_train.shape
```

Out[10]:
```
(537, 8)
```

In [11]:
```
x_test.shape
```

Out[11]:
```
(231, 8)
```

In [12]:
```
from sklearn.neighbors import KNeighborsClassifier
```

In [13]:
```
knn = KNeighborsClassifier(n_neighbors = 5)
```

In [14]:
```
knn.fit(x_train, y_train)
```

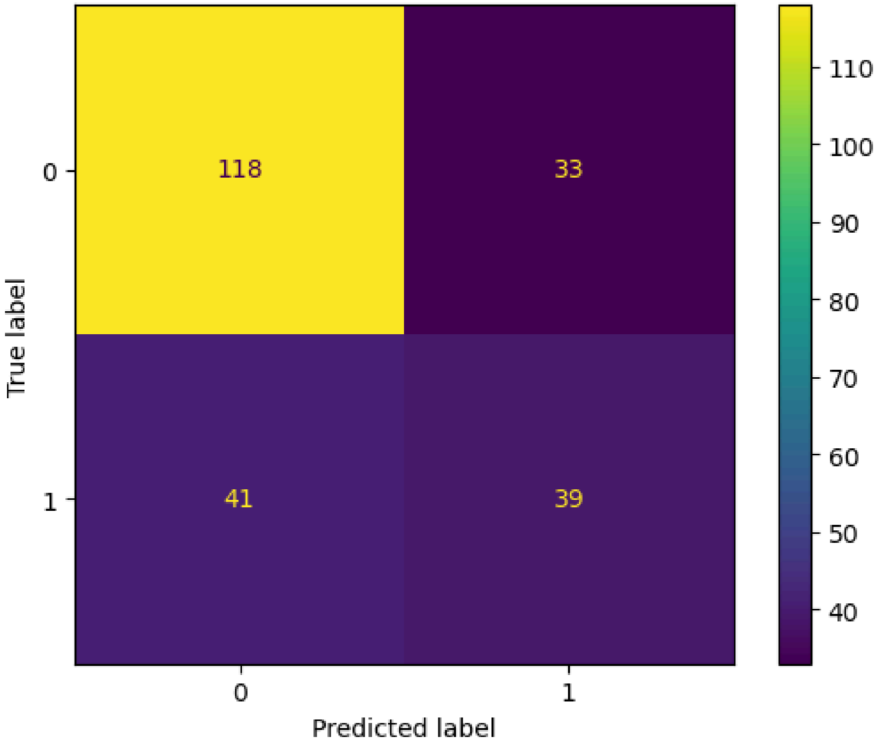
Out[14]:
```
▾  KNeighborsClassifier ⓘ ?
(https://scikit-
learn.org/1.4/modules/generated/sklearn.neighbors.KNeighborsClassifier.htm
KNeighborsClassifier()
```

In [15]:
```
from sklearn.metrics import accuracy_score , ConfusionMatrixDisplay
from sklearn.metrics import classification_report
```

In [16]:
```
y_pred = knn.predict(x_test)
```

```
In [17]: ConfusionMatrixDisplay.from_predictions(y_test,y_pred)
```

```
Out[17]: <sklearn.metrics._plot.confusion_matrix.ConfusionMatrixDisplay at 0x163253fca90>
```



```
In [18]: print(classification_report(y_test,y_pred))
```

```
              precision    recall  f1-score   support

           0       0.74      0.78      0.76       151
           1       0.54      0.49      0.51        80

    accuracy                           0.68       231
   macro avg       0.64      0.63      0.64       231
weighted avg       0.67      0.68      0.68       231
```

```
In [19]: import matplotlib.pyplot as plt
         import numpy as np
```

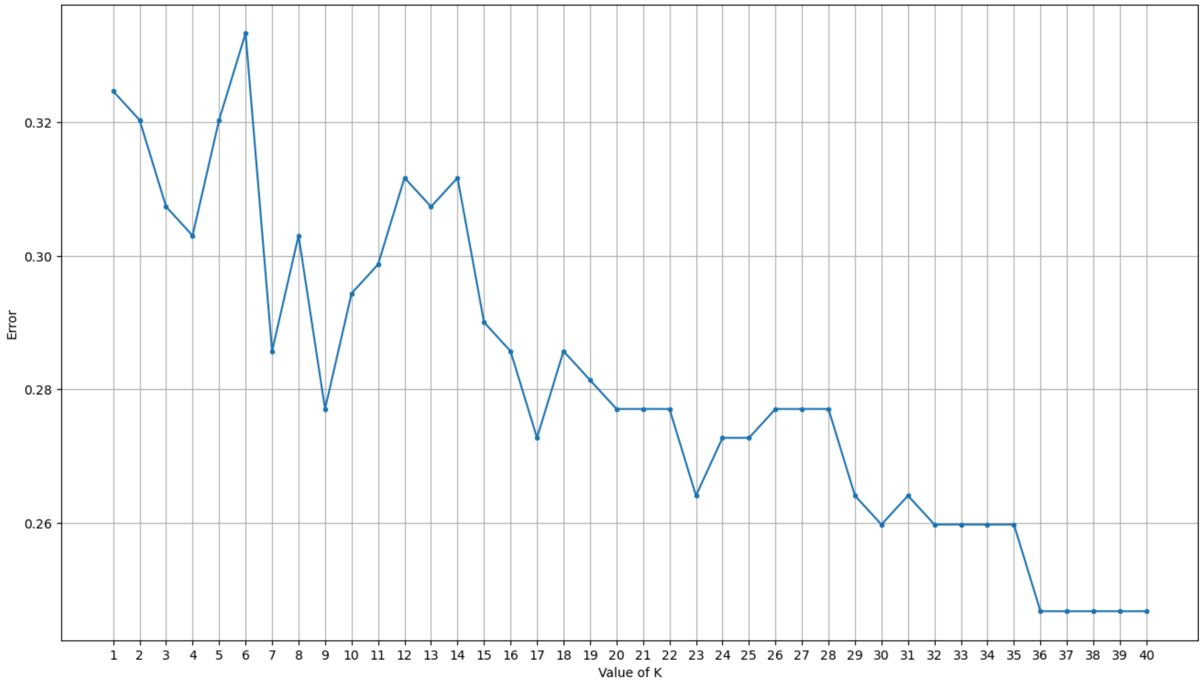
```
In [20]: error = []
         for k in range (1,41):
             knn = KNeighborsClassifier(n_neighbors = k)
             knn.fit(x_train, y_train)
             pred=knn.predict(x_test)
             error.append(np.mean(pred!=y_test))
```

In [21]:

```
plt.figure(figsize=(16,9))
plt.xlabel('Value of K')
plt.ylabel('Error')
plt.grid()
plt.xticks(range(1,41))
plt.plot(range(1,41),error,marker='.')
```

Out[21]:

```
[<matplotlib.lines.Line2D at 0x1632bc8b9d0>]
```



In [22]:

```
knn = KNeighborsClassifier(n_neighbors = 33)
```

In [23]:

```
knn.fit(x_train, y_train)
```

Out[23]:

```
▾      KNeighborsClassifier      ⓘ ?
                                 (https://scikit-
                                 learn.org/1.4/modules/generated/sklearn.neighbors.KNeighborsClassi
KNeighborsClassifier(n_neighbors=33)
```

In [24]:

```
y_pred=knn.predict(x_test)
```

In [25]:

```
print(classification_report(y_test,y_pred))
```

```
              precision    recall  f1-score   support

           0       0.77      0.87      0.81       151
           1       0.67      0.50      0.57        80

    accuracy                           0.74       231
   macro avg       0.72      0.68      0.69       231
weighted avg       0.73      0.74      0.73       231
```

In [ ]: