In [3]:
```
import pandas as pd
import seaborn as sns
```

In [4]:
```
df = pd.read_csv('diabetes.csv')
```

In [5]:
```
df
```

Out[5]:

	Pregnancies	Glucose	BloodPressure	SkinThickness	Insulin	BMI	Pedigree	Age	Ou
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 [6]:
```
# input data
x = df.drop('Outcome',axis = 1)

# output dta
y = df['Outcome']
```

In [7]:
```
y.value_counts()
```

Out[7]:
```
0    500
1    268
Name: Outcome, dtype: int64
```

In [8]:
```
# Feature scaling
from sklearn.preprocessing import MinMaxScaler as mms
scaler = mms()
```

In [9]:
```
x_scaled = scaler.fit_transform(x)
```

In [19]:
```
# cross-validation
from sklearn.model_selection import train_test_split as tts
```

```
xtrain, xtest, ytrain, ytest = tts(x_scaled, y, test_size = 0.25)
```

In [20]:
```
x.shape
```

Out[20]:
```
(768, 8)
```

In [21]:
```
xtrain.shape
```

Out[21]:
```
(576, 8)
```

In [22]:
```
xtest.shape
```

Out[22]:
```
(192, 8)
```

In [23]:
```
from sklearn.neighbors import KNeighborsClassifier as knc
knn = knc(n_neighbors = 5)
```

In [24]:
```
knn.fit(xtrain, ytrain)
```

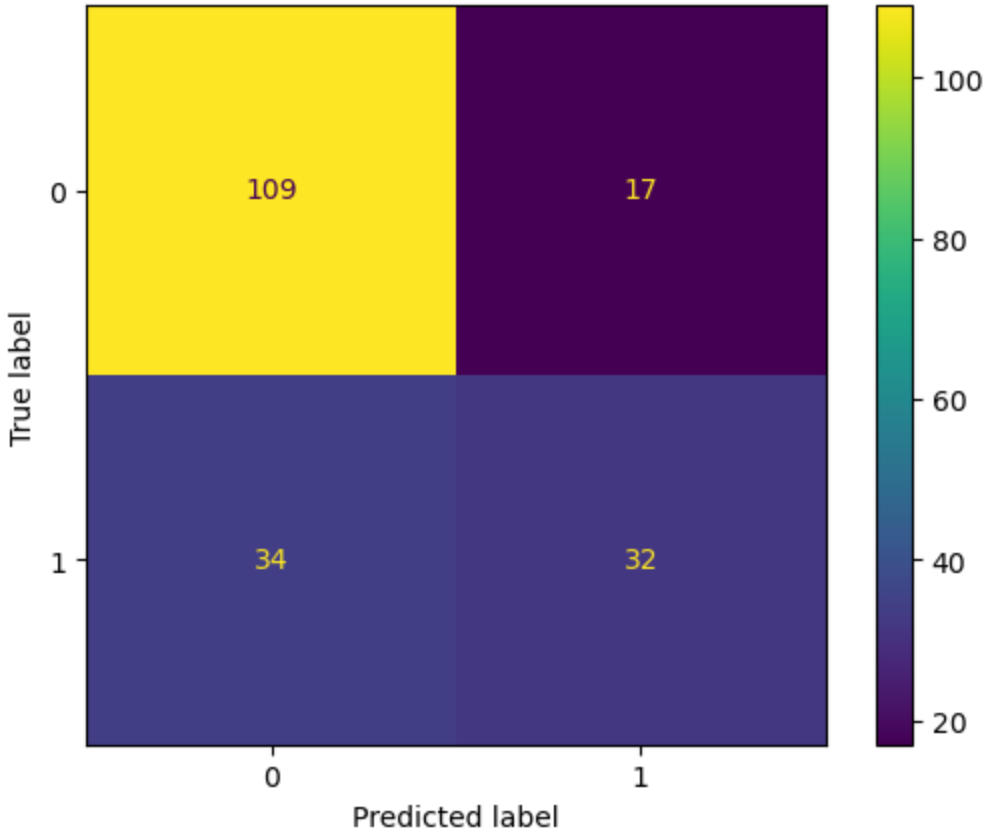
Out[24]:
```
▾ KNeighborsClassifier

KNeighborsClassifier()
```

In [25]:
```
from sklearn.metrics import accuracy_score as score
from sklearn.metrics import ConfusionMatrixDisplay as matrix
from sklearn.metrics import classification_report as report
```

In [26]:
```
y_pred = knn.predict(xtest)
```

In [27]:
```
matrix.from_predictions(ytest, y_pred)
```

Out[27]:
```
<sklearn.metrics._plot.confusion_matrix.ConfusionMatrixDisplay at 0x2a9a1da5330>
```



In [29]:
```
print(report(ytest, y_pred))
```

```
              precision    recall  f1-score   support

           0       0.76      0.87      0.81       126
           1       0.65      0.48      0.56        66

    accuracy                           0.73       192
   macro avg       0.71      0.67      0.68       192
weighted avg       0.72      0.73      0.72       192
```

In [30]:
```
score(ytest, y_pred)
```

Out[30]:
```
0.734375
```

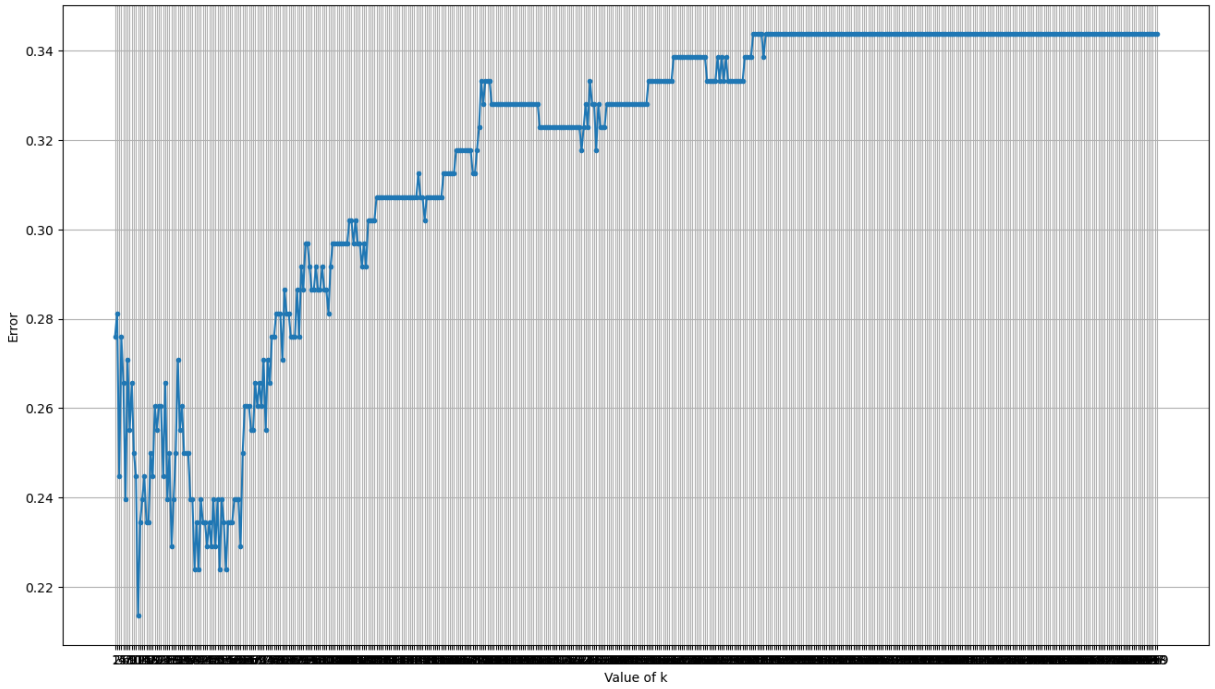
In [31]:
```
import matplotlib.pyplot as plt
import numpy as np
```

In [61]:
```
error = []
err = 1
i = 1
for k in range(1,500):
    knn = knc(n_neighbors = k)
    knn.fit(xtrain, ytrain)
    pred = knn.predict(xtest)
    error.append(np.mean(pred != ytest))
    if(np.mean(pred != ytest) < err):
        err = np.mean(pred != ytest)
```

```
        i = k
```

In [62]:
```
plt.figure(figsize=(16,9))
plt.xlabel('Value of k')
plt.ylabel('Error')
plt.grid()
plt.xticks(range(1,500))
plt.plot(range(1,500), error, marker='.')
```

Out[62]: [<matplotlib.lines.Line2D at 0x2a9a82a4ca0>]



In [46]:
```
print(err)
```

```
0.21354166666666666
```

In [47]:
```
i
```

Out[47]: 12

In [48]:
```
knn = knc(n_neighbors = i)
knn.fit(xtrain, ytrain)
y_pred = knn.predict(xtest)
```

In [49]:
```
score(ytest, y_pred)
```

Out[49]: 0.7864583333333334

In [63]:
```
print(report(ytest, y_pred))
```

	precision	recall	f1-score	support
0	0.78	0.94	0.85	126
1	0.82	0.48	0.61	66
accuracy			0.79	192
macro avg	0.80	0.71	0.73	192
weighted avg	0.79	0.79	0.77	192

In [ ]: