In [3]:
```
import pandas as pd
import numpy as np
import seaborn as sns
import matplotlib.pyplot as plt
from sklearn.metrics import classification_report, confusion_matrix, ConfusionMa
```

In [4]:
```
data = pd.read_csv("C:\\Users\\Omkar\\Downloads\\emails.csv.zip")
```

In [5]:
```
data = data.drop('Email No.', axis=1)
```

In [6]:
```
print(data.shape)
```

```
(5172, 3001)
```

In [7]:
```
print(data.info())
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 5172 entries, 0 to 5171
Columns: 3001 entries, the to Prediction
dtypes: int64(3001)
memory usage: 118.4 MB
None
```

In [8]:
```
print(data.describe())
```

	the	to	ect	and	for	\
count	5172.000000	5172.000000	5172.000000	5172.000000	5172.000000	
mean	6.640565	6.188128	5.143852	3.075599	3.124710	
std	11.745009	9.534576	14.101142	6.045970	4.680522	
min	0.000000	0.000000	1.000000	0.000000	0.000000	
25%	0.000000	1.000000	1.000000	0.000000	1.000000	
50%	3.000000	3.000000	1.000000	1.000000	2.000000	
75%	8.000000	7.000000	4.000000	3.000000	4.000000	
max	210.000000	132.000000	344.000000	89.000000	47.000000	

	of	a	you	hou	in	...	\
count	5172.000000	5172.000000	5172.000000	5172.000000	5172.000000	...	
mean	2.627030	55.517401	2.466551	2.024362	10.600155	...	
std	6.229845	87.574172	4.314444	6.967878	19.281892	...	
min	0.000000	0.000000	0.000000	0.000000	0.000000	...	
25%	0.000000	12.000000	0.000000	0.000000	1.000000	...	
50%	1.000000	28.000000	1.000000	0.000000	5.000000	...	
75%	2.000000	62.250000	3.000000	1.000000	12.000000	...	
max	77.000000	1898.000000	70.000000	167.000000	223.000000	...	

	connevey	jay	valued	lay	infrastructure	\
count	5172.000000	5172.000000	5172.000000	5172.000000	5172.000000	
mean	0.005027	0.012568	0.010634	0.098028	0.004254	
std	0.105788	0.199682	0.116693	0.569532	0.096252	
min	0.000000	0.000000	0.000000	0.000000	0.000000	
25%	0.000000	0.000000	0.000000	0.000000	0.000000	
50%	0.000000	0.000000	0.000000	0.000000	0.000000	
75%	0.000000	0.000000	0.000000	0.000000	0.000000	
max	4.000000	7.000000	2.000000	12.000000	3.000000	

	military	allowing	ff	dry	Prediction
count	5172.000000	5172.000000	5172.000000	5172.000000	5172.000000
mean	0.006574	0.004060	0.914733	0.006961	0.290023
std	0.138908	0.072145	2.780203	0.098086	0.453817
min	0.000000	0.000000	0.000000	0.000000	0.000000
25%	0.000000	0.000000	0.000000	0.000000	0.000000
50%	0.000000	0.000000	0.000000	0.000000	0.000000
75%	0.000000	0.000000	1.000000	0.000000	1.000000
max	4.000000	3.000000	114.000000	4.000000	1.000000

[8 rows x 3001 columns]

In [9]:
```
print(data['Prediction'].value_counts())
```

```
Prediction
0    3672
1    1500
Name: count, dtype: int64
```

In [10]:
```
X = data.drop('Prediction', axis=1)
y = data['Prediction']
```

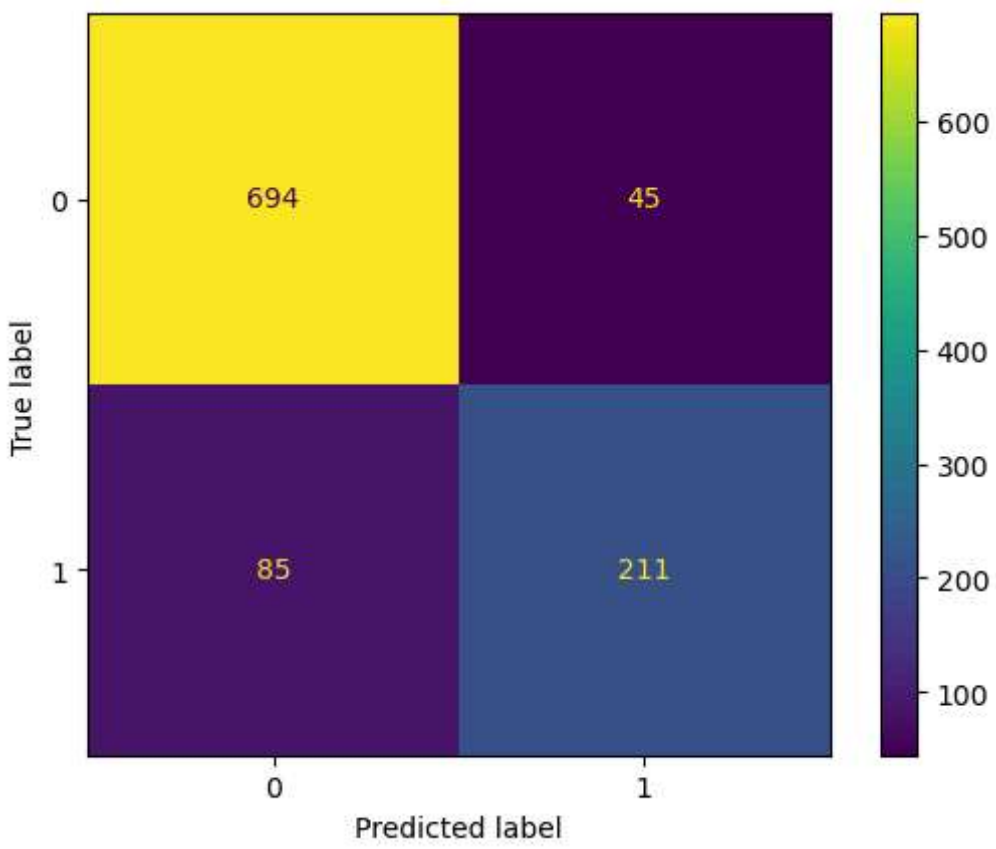
In [11]:
```
from sklearn.model_selection import train_test_split
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.20, random
```

In [12]:
```
from sklearn.neighbors import KNeighborsClassifier
neigh = KNeighborsClassifier(n_neighbors=2)
neigh.fit(X_train, y_train)
y_pred = neigh.predict(X_test)
```

In [13]:
```
print("Confusion Matrix:")
cm = confusion_matrix(y_test, y_pred)
print(cm)
```

```
Confusion Matrix:
[[694  45]
 [ 85 211]]
```

In [14]:
```
mat = ConfusionMatrixDisplay(confusion_matrix=cm)
mat.plot()
plt.show()
```



In [15]:
```
print(classification_report(y_test, y_pred))
```

```
              precision    recall  f1-score   support

           0       0.89      0.94      0.91       739
           1       0.82      0.71      0.76       296

    accuracy                           0.87      1035
   macro avg       0.86      0.83      0.84      1035
weighted avg       0.87      0.87      0.87      1035
```

In [16]:
```
print("Accuracy Score:")
print(accuracy_score(y_test, y_pred))
```

```
Accuracy Score:
0.8743961352657005
```

In [17]:
```
print("Precision Score:")
print(precision_score(y_test, y_pred))
```

```
Precision Score:
0.82421875
```

In [18]:
```
print("Recall Score:")
print(recall_score(y_test, y_pred))
```

```
Recall Score:
0.7128378378378378
```

In [19]:
```
print("Error:")
print(1 - accuracy_score(y_test, y_pred))
```

```
Error:
0.12560386473429952
```

In [21]:
```
from sklearn.svm import SVC
SVM = SVC(gamma='auto')
SVM.fit(X_train, y_train)
```

Out[21]:

SVC

SVC(gamma='auto')

In [22]:
```
SVM.score(X_train, y_train)
SVM.score(X_test, y_test)
```

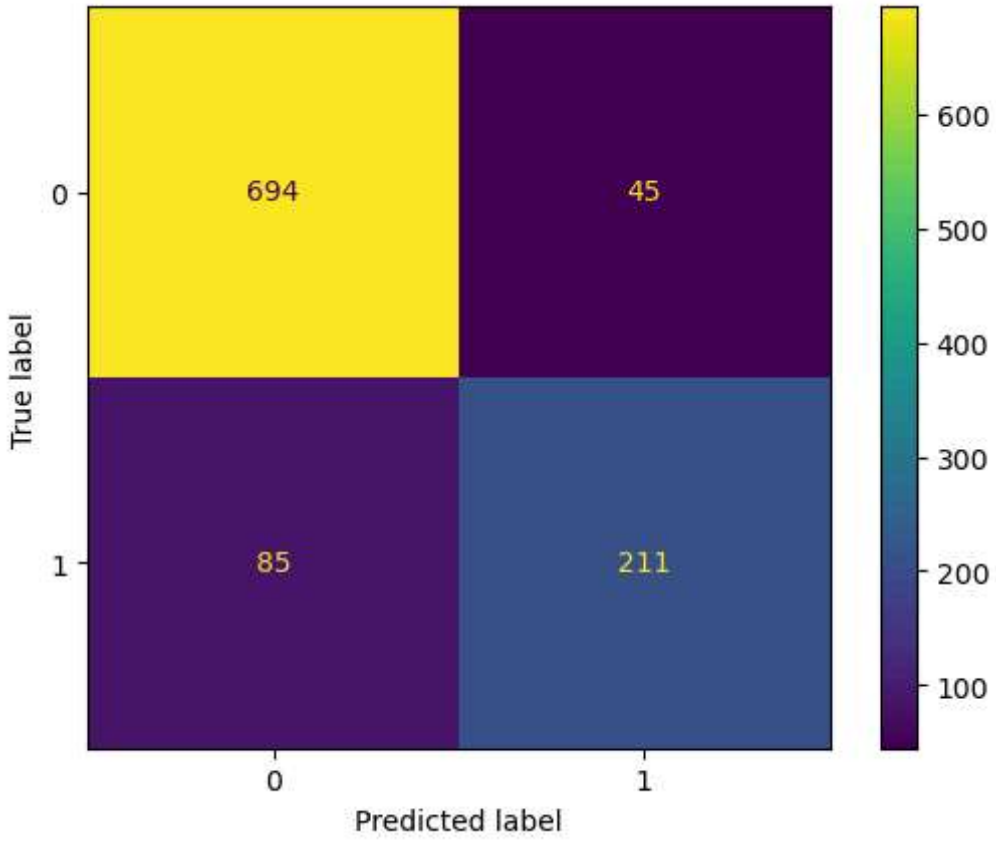
Out[22]: 0.9120772946859903

In [23]:
```
print("Confusion Matrix: ")
cm = confusion_matrix(y_test, y_pred)
cm
```

```
Confusion Matrix:
```

Out[23]:
```
array([[694,  45],
       [ 85, 211]], dtype=int64)
```

In [24]:
```
mat = ConfusionMatrixDisplay(confusion_matrix = cm)
mat.plot()
plt.show()
```



In [25]:
```
print(classification_report(y_test, y_pred))
```

```
              precision    recall  f1-score   support

           0       0.89      0.94      0.91       739
           1       0.82      0.71      0.76       296

    accuracy                           0.87      1035
   macro avg       0.86      0.83      0.84      1035
weighted avg       0.87      0.87      0.87      1035
```