Spam Classifier

In [1]:

```
from src.homeworks.homework2.KNNClassifier import KNNClassifier
from src.homeworks.homework2.scalers import MinMaxScaler, MaxAbsScaler, S
from src.homeworks.homework2.score import MetricCalculator
from src.homeworks.homework2.train_test_split import train_test_split

import pandas as pd
import matplotlib.pyplot as plt
```

Read spam.csv:

In [2]:

```
data = pd.read_csv("src/homeworks/homework2/notebooks/spam.csv")
data.describe()
```

Out[2]:

	word_freq_make	word_freq_address	word_freq_all	word_freq_3d	word_freq_ou
count	4601.000000	4601.000000	4601.000000	4601.000000	4601.00000
mean	0.104553	0.213015	0.280656	0.065425	0.31222
std	0.305358	1.290575	0.504143	1.395151	0.67251
min	0.000000	0.000000	0.000000	0.000000	0.00000
25%	0.000000	0.000000	0.000000	0.000000	0.00000
50%	0.000000	0.000000	0.000000	0.000000	0.00000
75%	0.000000	0.000000	0.420000	0.000000	0.38000
max	4.540000	14.280000	5.100000	42.810000	10.00000

8 rows × 58 columns

Divide the dataset into X and y:

In [3]:

```
y = data["label"].to_numpy()
X = data.drop(columns=['label']).to_numpy()
X
```

Out[3]:

```
array([[0.000e+00, 6.400e-01, 6.400e-01, ..., 3.756e+00, 6.100e+01,
        2.780e+02],
       [2.100e-01, 2.800e-01, 5.000e-01, ..., 5.114e+00, 1.010e+02,
        1.028e+03],
       [6.000e-02, 0.000e+00, 7.100e-01, ..., 9.821e+00, 4.850e+02,
        2.259e+03],
       ...,
       [3.000e-01, 0.000e+00, 3.000e-01, ..., 1.404e+00, 6.000e+00,
        1.180e+02],
       [9.600e-01, 0.000e+00, 0.000e+00, ..., 1.147e+00, 5.000e+00,
        7.800e+01],
       [0.000e+00, 0.000e+00, 6.500e-01, ..., 1.250e+00, 5.000e+00,
        4.000e+01]], shape=(4601, 57))
```

Creat a function that performs normalization, splitting into train and test, and counts metrics.

In [4]:

```
def get_score(X, y, test_size: float = 0.15, shuffle: bool = True, random
    scaler = scaler()
    X_new = scaler.fit_transform(X)

    X_train, X_test, y_train, y_test = train_test_split(X_new, y, test_si

    accuracy = []
    f1_score = []

    for k in range(1, 21):
        knn = KNNClassifier(k, 5)
        knn.fit(X_train, y_train)
        y_pred = knn.predict(X_test)

        metric = MetricCalculator(y_pred, y_test)
        accuracy.append(metric.accuracy())
        f1_score.append(metric.f1_score())

    return accuracy, f1_score
```

MinMaxScaler

In [5]:

```
accuracy, f1_score = get_score(X, y, test_size=0.15, shuffle=True, random
```

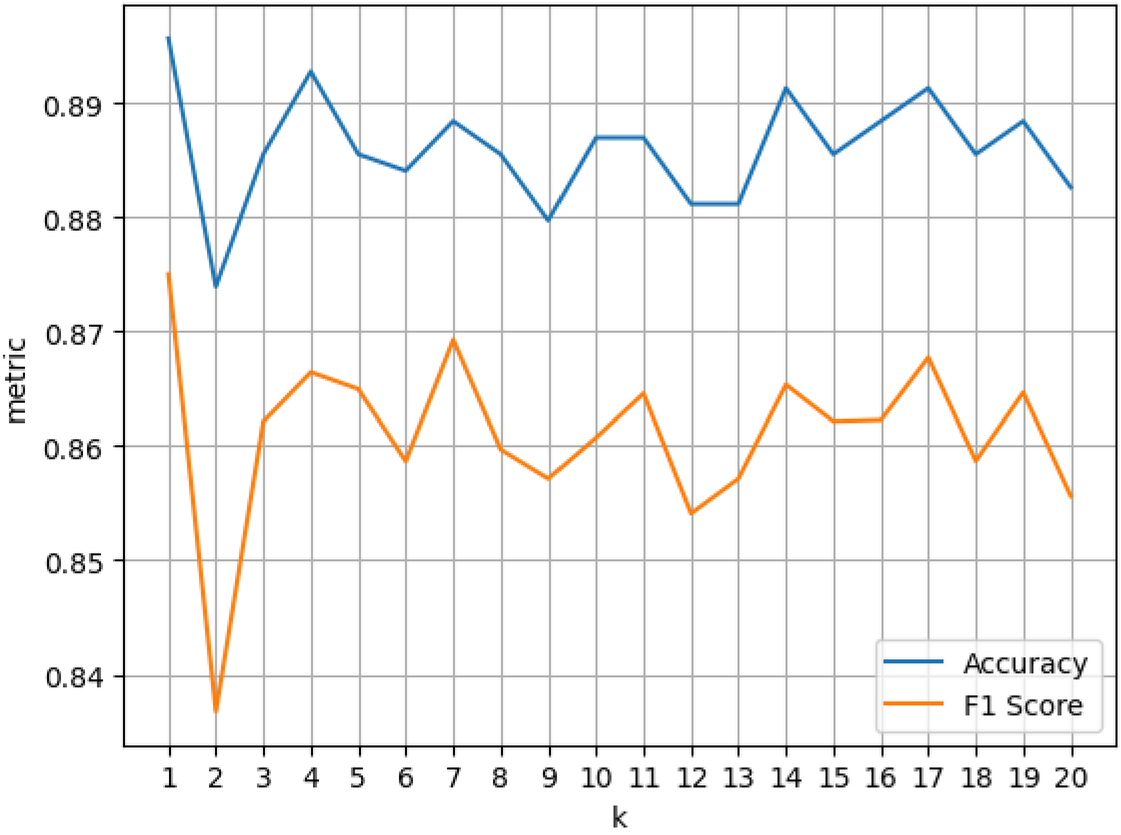
In [6]:

```
k_values = range(1, 21)
plt.plot(k_values, accuracy, label='Accuracy', linestyle='-')
plt.plot(k_values, f1_score, label='F1 Score', linestyle='-')

plt.xlabel('k')
plt.ylabel('metric')
plt.xticks(k_values)

plt.grid(True)
plt.legend()

plt.show()
```



MaxAbsScaler

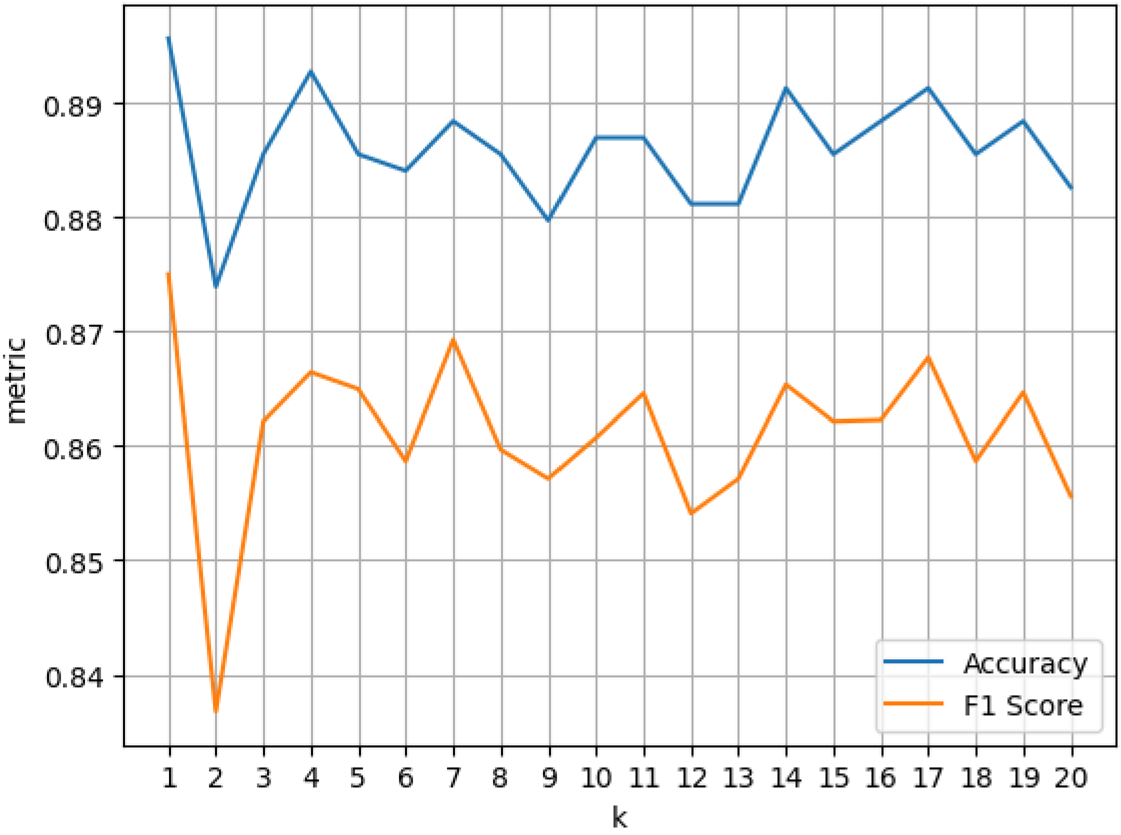
In [7]:
```
accuracy, f1_score = get_score(X, y, test_size=0.15, shuffle=True, random
```

In [8]:
```
k_values = range(1, 21)
plt.plot(k_values, accuracy, label='Accuracy', linestyle='-')
plt.plot(k_values, f1_score, label='F1 Score', linestyle='-')

plt.xlabel('k')
plt.ylabel('metric')
plt.xticks(k_values)

plt.grid(True)
plt.legend()

plt.show()
```



StandardScaler

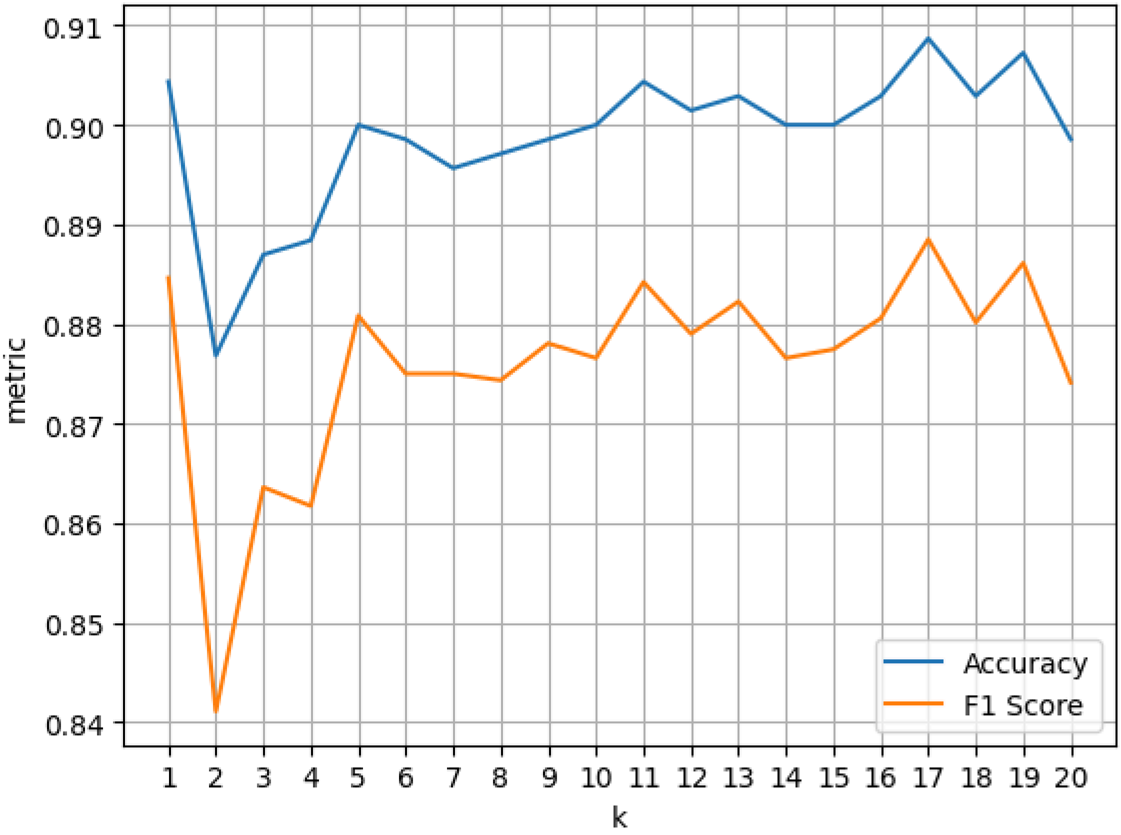
```
In [9]: accuracy, f1_score = get_score(X, y, test_size=0.15, shuffle=True, random
```

```
In [10]: k_values = range(1, 21)
plt.plot(k_values, accuracy, label='Accuracy', linestyle='-')
plt.plot(k_values, f1_score, label='F1 Score', linestyle='-')

plt.xlabel('k')
plt.ylabel('metric')
plt.xticks(k_values)

plt.grid(True)
plt.legend()

plt.show()
```


0.91
0.90
0.89
0.88
0.87
0.86
0.85
0.84
metric
1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
20
k
Accuracy
F1 Score