In [4]:
```
import pandas as pd
from sklearn.model_selection import train_test_split
from sklearn.feature_extraction.text import CountVectorizer
from sklearn.linear_model import LogisticRegression
from sklearn.metrics import accuracy_score, precision_score, recall_score, f1_score
```

In [5]:
```
import kagglehub

# Download latest version
path = kagglehub.dataset_download("venky73/spam-mails-dataset")

print("Path to dataset files:", path)
```

```
Path to dataset files: /home/anurag/.cache/kagglehub/datasets/venky73/spam-mails-dataset/versions/1
```

In [6]:
```
path
```

Out[6]:
```
'/home/anurag/.cache/kagglehub/datasets/venky73/spam-mails-dataset/versions/1'
```

In [7]:
```
df = pd.read_csv(f"{path}/spam_ham_dataset.csv")
df
```

Out[7]:

	Unnamed: 0	label	text	label_num
0	605	ham	Subject: enron methanol ; meter # : 988291\r\n...	0
1	2349	ham	Subject: hpl nom for january 9 , 2001\r\n(see...	0
2	3624	ham	Subject: neon retreat\r\nho ho ho , we ' re ar...	0
3	4685	spam	Subject: photoshop , windows , office . cheap ...	1
4	2030	ham	Subject: re : indian springs\r\nthis deal is t...	0
...	...	...	...	...
5166	1518	ham	Subject: put the 10 on the ft\r\nthe transport...	0
5167	404	ham	Subject: 3 / 4 / 2000 and following noms\r\nhp...	0
5168	2933	ham	Subject: calpine daily gas nomination\r\n>\r\n...	0
5169	1409	ham	Subject: industrial worksheets for august 2000...	0
5170	4807	spam	Subject: important online banking alert\r\ndea...	1

5171 rows × 4 columns

In [8]:
```
df.tail()
```

Out[8]:

	Unnamed: 0	label	text	label_num
5166	1518	ham	Subject: put the 10 on the ft\r\nthe transport...	0
5167	404	ham	Subject: 3 / 4 / 2000 and following noms\r\nhp...	0
5168	2933	ham	Subject: calpine daily gas nomination\r\n>\r\n...	0
5169	1409	ham	Subject: industrial worksheets for august 2000...	0
5170	4807	spam	Subject: important online banking alert\r\ndea...	1

In [9]:
```
data = df[["text", "label_num"]]
data
```

Out[9]:

	text	label_num
0	Subject: enron methanol ; meter # : 988291\r\n...	0
1	Subject: hpl nom for january 9 , 2001\r\n(see...	0
2	Subject: neon retreat\r\nho ho ho , we ' re ar...	0
3	Subject: photoshop , windows , office . cheap ...	1
4	Subject: re : indian springs\r\nthis deal is t...	0
...	...	...
5166	Subject: put the 10 on the ft\r\nthe transport...	0
5167	Subject: 3 / 4 / 2000 and following noms\r\nhp...	0
5168	Subject: calpine daily gas nomination\r\n>\r\n...	0
5169	Subject: industrial worksheets for august 2000...	0
5170	Subject: important online banking alert\r\ndea...	1

5171 rows × 2 columns

In [10]:
```
data = data.rename(columns={"text": "message", "label_num": "label"})
data
```

Out[10]:

	message	label
0	Subject: enron methanol ; meter # : 988291\r\n...	0
1	Subject: hpl nom for january 9 , 2001\r\n(see...	0
2	Subject: neon retreat\r\nho ho ho , we ' re ar...	0
3	Subject: photoshop , windows , office . cheap ...	1
4	Subject: re : indian springs\r\nthis deal is t...	0
...	...	...
5166	Subject: put the 10 on the ft\r\nthe transport...	0
5167	Subject: 3 / 4 / 2000 and following noms\r\nhp...	0
5168	Subject: calpine daily gas nomination\r\n>\r\n...	0
5169	Subject: industrial worksheets for august 2000...	0
5170	Subject: important online banking alert\r\ndea...	1

5171 rows × 2 columns

In [11]:
```
X_train, X_test, Y_train, Y_test = train_test_split(data['message'], data['label'], test_size=0.2, random_state=
```

In [12]:
```
print(data.shape)
print(X_train.shape)
print(X_test.shape)
```

```
(5171, 2)
(4136,)
(1035,)
```

In [13]:
```
count_vector = CountVectorizer()
```

In [14]:
```
training_data = count_vector.fit_transform(X_train).toarray()
```

In [15]:
```
training_data
```

Out[15]:
```
array([[0, 0, 0, ..., 0, 0, 0],
       [0, 1, 0, ..., 0, 0, 0],
       [6, 0, 0, ..., 0, 0, 0],
       ...,
       [0, 0, 0, ..., 0, 0, 0],
       [0, 0, 0, ..., 0, 0, 0],
       [0, 0, 0, ..., 0, 0, 0]])
```

In [16]:
```
testing_data = count_vector.transform(X_test).toarray()
testing_data
```

Out[16]:
```
array([[0, 0, 0, ..., 0, 0, 0],
       [0, 0, 0, ..., 0, 0, 0],
       [0, 0, 0, ..., 0, 0, 0],
       ...,
       [2, 0, 0, ..., 0, 0, 0],
       [0, 0, 0, ..., 0, 0, 0],
       [0, 0, 0, ..., 0, 0, 0]])
```

In [17]:
```
frequency_matrix = pd.DataFrame(training_data, columns=count_vector.get_feature_names_out())
frequency_matrix.head()
```

Out[17]:

	00	000	0000	000000	000000000002858	000000000049773	000080	000099	0001	00020608	...	zyl	zynsdirnh	zynve	zyqt
0	0	0	0	0	0	0	0	0	0	0	...	0	0	0	
1	0	1	0	0	0	0	0	0	0	0	...	0	0	0	
2	6	0	0	0	0	0	0	0	0	0	...	0	0	0	
3	0	0	0	0	0	0	0	0	0	0	...	0	0	0	
4	0	0	0	0	0	0	0	0	0	0	...	0	0	0	

5 rows × 44759 columns

In [18]:
```
testing_data
```

Out[18]:
```
array([[0, 0, 0, ..., 0, 0, 0],
       [0, 0, 0, ..., 0, 0, 0],
       [0, 0, 0, ..., 0, 0, 0],
       ...,
       [2, 0, 0, ..., 0, 0, 0],
       [0, 0, 0, ..., 0, 0, 0],
       [0, 0, 0, ..., 0, 0, 0]])
```

In [19]:
```
clf = LogisticRegression(random_state=0).fit(training_data, Y_train)
```

In [20]:
```
predictions = clf.predict(testing_data)
predictions
```

Out[20]:
```
array([0, 0, 1, ..., 0, 0, 0])
```

In [21]:
```
print("Accuracy score: ", format(accuracy_score(Y_test, predictions)))
print("Precision score: ", format(precision_score(Y_test, predictions)))
print("Recall score: ", format(recall_score(Y_test, predictions)))
print("F1 Score: ", format(f1_score(Y_test, predictions)))
print("Confusion Matrix: ", format(accuracy_score(Y_test, predictions)))
```

```
Accuracy score:  0.9758454106280193
Precision score:  0.9573770491803278
Recall score:  0.9605263157894737
F1 Score:  0.9589490968801314
Confusion Matrix:  0.9758454106280193
```

In [ ]: