In [4]:
```
import pandas as pd
import zipfile
import requests
from io import BytesIO
from sklearn.model_selection import train_test_split
from sklearn.feature_extraction.text import TfidfVectorizer
from sklearn.naive_bayes import MultinomialNB
from sklearn.metrics import accuracy_score, confusion_matrix, classification_rep
```

In [5]:
```
url = "https://archive.ics.uci.edu/ml/machine-learning-databases/00228/smss
```

In [6]:
```
response = requests.get(url)
zip_file = zipfile.ZipFile(BytesIO(response.content))
```

In [8]:
```
with zip_file.open('SMSSpamCollection') as file:
    data = pd.read_csv(file, sep='\t', header=None, names=['label', 'message'])
```

In [9]:
```
data['label'] = data['label'].map({'ham': 0, 'spam': 1})
```

In [10]:
```
from sklearn.model_selection import train_test_split
X_train, X_test, y_train, y_test = train_test_split(data['message'], data['label
```

In [11]:
```
vectorizer = TfidfVectorizer(stop_words='english')
X_train_tfidf = vectorizer.fit_transform(X_train)
X_test_tfidf = vectorizer.transform(X_test)
```

In [12]:
```
model = MultinomialNB()
model.fit(X_train_tfidf, y_train)
```

Out[12]:
```
▾ MultinomialNB

MultinomialNB()
```

In [13]:
```
y_pred = model.predict(X_test_tfidf)
```

In [14]:
```
accuracy = accuracy_score(y_test, y_pred)
conf_matrix = confusion_matrix(y_test, y_pred)
report = classification_report(y_test, y_pred)
```

In [15]:
```
print(f"Accuracy: {accuracy*100:.2f}%")
print("\nConfusion Matrix:\n", conf_matrix)
print("\nClassification Report:\n", report)
```

```
Accuracy: 97.85%

Confusion Matrix:
 [[966   0]
 [ 24 125]]

Classification Report:
               precision    recall  f1-score   support

           0       0.98      1.00      0.99       966
           1       1.00      0.84      0.91       149

    accuracy                           0.98      1115
   macro avg       0.99      0.92      0.95      1115
weighted avg       0.98      0.98      0.98      1115
```

In [20]:
```
test_email1 = [
    "Congratulations! You've won a $1000 gift card! Click the link below to clai
]
```

In [21]:
```
test_email_tfidf = vectorizer.transform(test_email)
```

In [22]:
```
prediction = model.predict(test_email_tfidf)
```

In [23]:
```
if prediction[0] == 1:
    print("The email is classified as SPAM.")
else:
    print("The email is classified as NOT SPAM.")
```

```
The email is classified as SPAM.
```

In [24]:
```
test_email2 = [
    "Hi, we would love to hear your feedback on your recent purchase. Please tak
]
```

In [25]:
```
test_email_tfidf = vectorizer.transform(test_email)
```

In [26]:
```
prediction = model.predict(test_email_tfidf)
```

In [29]:
```
if prediction[0] == 0:
    print("The email is classified as SPAM.")
else:
    print("The email is classified as NOT SPAM.")
```

```
The email is classified as NOT SPAM.
```

In [ ]: