In [14]:
```
import pandas as pd

# Load the dataset
try:
    df = pd.read_csv('spam.csv', encoding='latin-1')
    print("DataFrame successfully loaded!")
except FileNotFoundError:
    print("Error: The file 'spam.csv' was not found.")
    df = None

# Display the first few rows of the DataFrame and its information
if df is not None:
    print("\nDataFrame:")
    print(df)
    print("\nDataFrame info:")
    print(df.info())
```

```
DataFrame successfully loaded!

DataFrame:
        v1                                                 v2 Unnamed: 2  \
0      ham  Go until jurong point, crazy.. Available only ...         NaN
1      ham                      Ok lar... Joking wif u oni...         NaN
2     spam  Free entry in 2 a wkly comp to win FA Cup fina...         NaN
3      ham  U dun say so early hor... U c already then say...         NaN
4      ham  Nah I don't think he goes to usf, he lives aro...         NaN
...    ...                                                ...         ...
5567  spam  This is the 2nd time we have tried 2 contact u...         NaN
5568   ham              Will Ì_ b going to esplanade fr home?         NaN
5569   ham  Pity, * was in mood for that. So...any other s...         NaN
5570   ham  The guy did some bitching but I acted like i'd...         NaN
5571   ham                        Rofl. Its true to its name         NaN

     Unnamed: 3 Unnamed: 4
0           NaN        NaN
1           NaN        NaN
2           NaN        NaN
3           NaN        NaN
4           NaN        NaN
...         ...        ...
5567        NaN        NaN
5568        NaN        NaN
5569        NaN        NaN
5570        NaN        NaN
5571        NaN        NaN

[5572 rows x 5 columns]

DataFrame info:
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 5572 entries, 0 to 5571
Data columns (total 5 columns):
 #   Column      Non-Null Count  Dtype
---  ------      --------------  -----
 0   v1          5572 non-null   object
 1   v2          5572 non-null   object
 2   Unnamed: 2  50 non-null     object
 3   Unnamed: 3  12 non-null     object
 4   Unnamed: 4  6 non-null      object
dtypes: object(5)
memory usage: 217.8+ KB
None
```

In [15]:
```
import numpy as np
import pandas as pd
import nltk
from nltk.corpus import stopwords
import string
from sklearn.model_selection import train_test_split
from sklearn.feature_extraction.text import CountVectorizer
from sklearn.naive_bayes import MultinomialNB
```

```
from sklearn.metrics import classification_report

# Load the dataset
df = pd.read_csv('spam.csv', encoding='latin-1')

# Drop unnecessary columns
df = df.drop(columns=['Unnamed: 2', 'Unnamed: 3', 'Unnamed: 4'], axis=1)

# Rename columns for clarity
df = df.rename(columns={'v1': 'label', 'v2': 'message'})

# Check for any remaining missing values
print("Missing values after dropping columns:")
print(df.isnull().sum())

# Map 'spam' and 'ham' to numerical values (1 and 0)
df['label'] = df['label'].map({'ham': 0, 'spam': 1})

# Print the value counts to confirm the mapping
print("\nLabel value counts after mapping:")
print(df['label'].value_counts())
```

```
Missing values after dropping columns:
label      0
message    0
dtype: int64

Label value counts after mapping:
label
0    4825
1     747
Name: count, dtype: int64
```

In [21]:

```
import numpy as np
import pandas as pd
from sklearn.model_selection import train_test_split
from sklearn.feature_extraction.text import CountVectorizer
from sklearn.naive_bayes import MultinomialNB
from sklearn.metrics import classification_report

# Load the dataset
df = pd.read_csv('spam.csv', encoding='latin-1')

# Data cleaning and preparation
df = df.drop(columns=['Unnamed: 2', 'Unnamed: 3', 'Unnamed: 4'], axis=1)
df = df.rename(columns={'v1': 'label', 'v2': 'message'})
df['label'] = df['label'].map({'ham': 0, 'spam': 1})

# Split the data into training and testing sets
X_train, X_test, y_train, y_test = train_test_split(df['message'], df['label'], test_si

# Convert text to a matrix of token counts, removing common English stopwords
count_vectorizer = CountVectorizer(stop_words='english')
X_train_counts = count_vectorizer.fit_transform(X_train)
X_test_counts = count_vectorizer.transform(X_test)

# Create and train the Naive Bayes model
spam_detect_model = MultinomialNB()
spam_detect_model.fit(X_train_counts, y_train)

# Evaluate model performance on the test set
predictions = spam_detect_model.predict(X_test_counts)
print("Classification Report:")
print(classification_report(y_test, predictions))

# --- Prediction on a new message ---
new_message = "Free entry in a weekly competition to win FA Cup final tickets! Text FA

print("Message to be predicted:",new_message)

# Vectorize the new message using the existing CountVectorizer
vectorized_message = count_vectorizer.transform([new_message])
```

```
# Predict the class (spam or ham) using the trained model
prediction = spam_detect_model.predict(vectorized_message)

# Display the final prediction
print("\n--- Result ---")
if prediction[0] == 1:
    print(f"The model predicts that this message is: 'SPAM'")
else:
    print(f"The model predicts that this message is: 'HAM'")
```

```
Classification Report:
              precision    recall  f1-score   support

           0       0.99      0.99      0.99      1464
           1       0.94      0.93      0.94       208

    accuracy                           0.98      1672
   macro avg       0.97      0.96      0.96      1672
weighted avg       0.98      0.98      0.98      1672

Message to be predicted: Free entry in a weekly competition to win FA Cup final ticket
s! Text FA to 87121.

--- Result ---
The model predicts that this message is: 'SPAM'
```

In [ ]: