Task-2 Text Classification With Naive Bayes

Importing Libraries

In [1]:
```
import pandas as pd
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, precision_score, recall_score, classification_report
import joblib
```

In [2]:
```
data = pd.read_csv('spam.csv', encoding='latin-1')
```

In [3]:
```
data = data[['v1', 'v2']]
```

In [4]:
```
data.columns = ['label', 'message']
```

In [5]:
```
data['label'] = data['label'].map({'spam': 1, 'ham': 0})
```

In [6]:
```
X = data['message']
y = data['label']
```

In [7]:
```
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)
```

In [8]:
```
vectorizer = TfidfVectorizer(stop_words='english', max_features=3000)
X_train_tfidf = vectorizer.fit_transform(X_train)
X_test_tfidf = vectorizer.transform(X_test)
```

In [9]:
```
nb_model = MultinomialNB()
nb_model.fit(X_train_tfidf, y_train)
```

Out[9]:
```
MultinomialNB()
```

In [10]:
```
y_pred = nb_model.predict(X_test_tfidf)
```

Evaluating The Model

In [11]:
```
accuracy = accuracy_score(y_test, y_pred)
precision = precision_score(y_test, y_pred)
recall = recall_score(y_test, y_pred)
classification_rep = classification_report(y_test, y_pred)

print(f"Accuracy: {accuracy:.4f}")
print(f"Precision: {precision:.4f}")
print(f"Recall: {recall:.4f}")
print("\nClassification Report:\n", classification_rep)
```

```
Accuracy: 0.9776
Precision: 1.0000
Recall: 0.8333

Classification Report:
              precision    recall  f1-score   support

           0       0.97      1.00      0.99       965
           1       1.00      0.83      0.91       150

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

In [12]:
```
joblib.dump(nb_model, 'naive_bayes_spam_model.pkl')
joblib.dump(vectorizer, 'tfidf_vectorizer_spam.pkl')
```

Out[12]:
```
['tfidf_vectorizer_spam.pkl']
```

Prediction

In [13]:
```
def predict_spam(new_message):

    nb_model = joblib.load('naive_bayes_spam_model.pkl')
    vectorizer = joblib.load('tfidf_vectorizer_spam.pkl')

    new_message_tfidf = vectorizer.transform([new_message])
```

```
    prediction = nb_model.predict(new_message_tfidf)

    if prediction[0] == 1:
        print(f"The message: '{new_message}' is **SPAM**")
    else:
        print(f"The message: '{new_message}' is **NOT SPAM**")
```

In [15]:
```
while True:
    new_input = input("Enter a new message to classify as Spam or Not Spam (or type 'exit' to stop):")
    if new_input.lower() == 'exit':
        break
    predict_spam(new_input)
```

```
Enter a new message to classify as Spam or Not Spam (or type 'exit' to stop):You are the lucky winner of a vacation pac
kage to the Bahamas. Click here to redeem.
The message: 'You are the lucky winner of a vacation package to the Bahamas. Click here to redeem.' is **SPAM**
Enter a new message to classify as Spam or Not Spam (or type 'exit' to stop):Congratulations! You've won a $1000 Walmar
t gift card. Click here to claim your prize.
The message: 'Congratulations! You've won a $1000 Walmart gift card. Click here to claim your prize.' is **SPAM**
Enter a new message to classify as Spam or Not Spam (or type 'exit' to stop):Happy Birthday! Hope you have a fantastic
day!
The message: 'Happy Birthday! Hope you have a fantastic day!' is **NOT SPAM**
Enter a new message to classify as Spam or Not Spam (or type 'exit' to stop):Your order has been shipped and will arriv
e on Friday.
The message: 'Your order has been shipped and will arrive on Friday.' is **NOT SPAM**
Enter a new message to classify as Spam or Not Spam (or type 'exit' to stop):exit
```

In [ ]: