codetech task4

December 9, 2025

[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, classification_report,␣
 ↪confusion_matrix
```

[2]:
```
url = "https://raw.githubusercontent.com/justmarkham/pycon-2016-tutorial/master/
 ↪data/sms.tsv"
data = pd.read_csv(url, sep='\t', header=None, names=['label','message'])
data.head()
```

[2]:

	label	message
0	ham	Go until jurong point, crazy.. Available only …
1	ham	Ok lar… Joking wif u oni…
2	spam	Free entry in 2 a wkly comp to win FA Cup fina…
3	ham	U dun say so early hor… U c already then say…
4	ham	Nah I don't think he goes to usf, he lives aro…

[3]:
```
data['label'] = data['label'].map({'ham':0, 'spam':1})
data.head()
```

[3]:

	label	message
0	0	Go until jurong point, crazy.. Available only …
1	0	Ok lar… Joking wif u oni…
2	1	Free entry in 2 a wkly comp to win FA Cup fina…
3	0	U dun say so early hor… U c already then say…
4	0	Nah I don't think he goes to usf, he lives aro…

[5]:
```
X_train, X_test, y_train, y_test = train_test_split(
    data['message'],
    data['label'],
    test_size=0.2,
    random_state=42
)
```

[6]:
```
tfidf = TfidfVectorizer(stop_words='english')
X_train_tfidf = tfidf.fit_transform(X_train)
X_test_tfidf = tfidf.transform(X_test)
```

[7]:
```
model = MultinomialNB()
model.fit(X_train_tfidf, y_train)
```

[7]:
```
MultinomialNB()
```

[8]:
```
predictions = model.predict(X_test_tfidf)

print("Accuracy:", accuracy_score(y_test, predictions))
print("\nClassification Report:\n", classification_report(y_test, predictions))
print("\nConfusion Matrix:\n", confusion_matrix(y_test, predictions))
```

```
Accuracy: 0.97847533632287

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


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

[9]:
```
def check_spam(text):
    vector = tfidf.transform([text])
    result = model.predict(vector)[0]
    return "SPAM" if result == 1 else "NOT SPAM"

# Example
check_spam("Congratulations! You won a free lottery ticket.")
```

[9]:
```
'NOT SPAM'
```

[ ]: