```
import pandas as pd
import numpy as np
import string

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, f1_score, confusion_matrix, classification_report

import nltk
from nltk.corpus import stopwords
from nltk.stem import PorterStemmer
```

```
# Load dataset
df = pd.read_csv('/content/news.csv')

# Display first 5 rows
df.head()
```

	Unnamed: 0	title	text	label
0	8476	You Can Smell Hillary's Fear	Daniel Greenfield, a Shillman Journalism Fello...	FAKE
1	10294	Watch The Exact Moment Paul Ryan Committed Pol...	Google Pinterest Digg Linkedin Reddit Stumbleu...	FAKE
2	3608	Kerry to go to Paris in gesture of sympathy	U.S. Secretary of State John F. Kerry said Mon...	REAL
3	10142	Bernie supporters on Twitter erupt in anger ag...	— Kaydee King (@KaydeeKing) November 9, 2016 T...	FAKE
4	875	The Battle of New York: Why This Primary Matters	It's primary day in New York and front-runners...	REAL

Next steps: Generate code with df | New interactive sheet

```
# Dataset size
print("Dataset size:", df.shape)

# Column names
print(df.columns)
```

```
# Class distribution
print(df['label'].value_counts())
```

```
Dataset size: (6335, 4)
Index(['Unnamed: 0', 'title', 'text', 'label'], dtype='object')
label
REAL    3171
FAKE    3164
Name: count, dtype: int64
```

```
nltk.download('stopwords')

stop_words = set(stopwords.words('english'))
stemmer = PorterStemmer()

def preprocess_text(text):
    text = text.lower()
    text = text.translate(str.maketrans('', '', string.punctuation))
    words = text.split()
    words = [stemmer.stem(word) for word in words if word not in stop_words]
    return " ".join(words)

# Apply preprocessing
df['clean_text'] = df['text'].apply(preprocess_text)

df[['text', 'clean_text']].head()
```

```
[nltk_data] Downloading package stopwords to /root/nltk_data...
[nltk_data]   Package stopwords is already up-to-date!
```

	text	clean_text
0	Daniel Greenfield, a Shillman Journalism Fello...	daniel greenfield shillman journal fellow free...
1	Google Pinterest Digg Linkedin Reddit Stumbleu...	googl pinterest digg linkedin reddit stumbleup...
2	U.S. Secretary of State John F. Kerry said Mon...	us secretari state john f kerri said monday st...
3	— Kaydee King (@KaydeeKing) November 9, 2016 T...	— kayde king kaydeek novemb 9 2016 lesson toni...
4	It's primary day in New York and front-runners...	primari day new york frontrunn hillari clinton...

```
vectorizer = TfidfVectorizer(max_features=5000)

X = vectorizer.fit_transform(df['clean_text'])
y = df['label']

print("Feature matrix shape:", X.shape)
print("Sample feature names:", vectorizer.get_feature_names_out()[:10])
```

```
Feature matrix shape: (6335, 5000)
Sample feature names: ['10' '100' '1000' '10000' '100000' '11' '12' '1237' '13' '14']
```

```
X_train, X_test, y_train, y_test = train_test_split(
    X, y, test_size=0.2, random_state=42
)
```

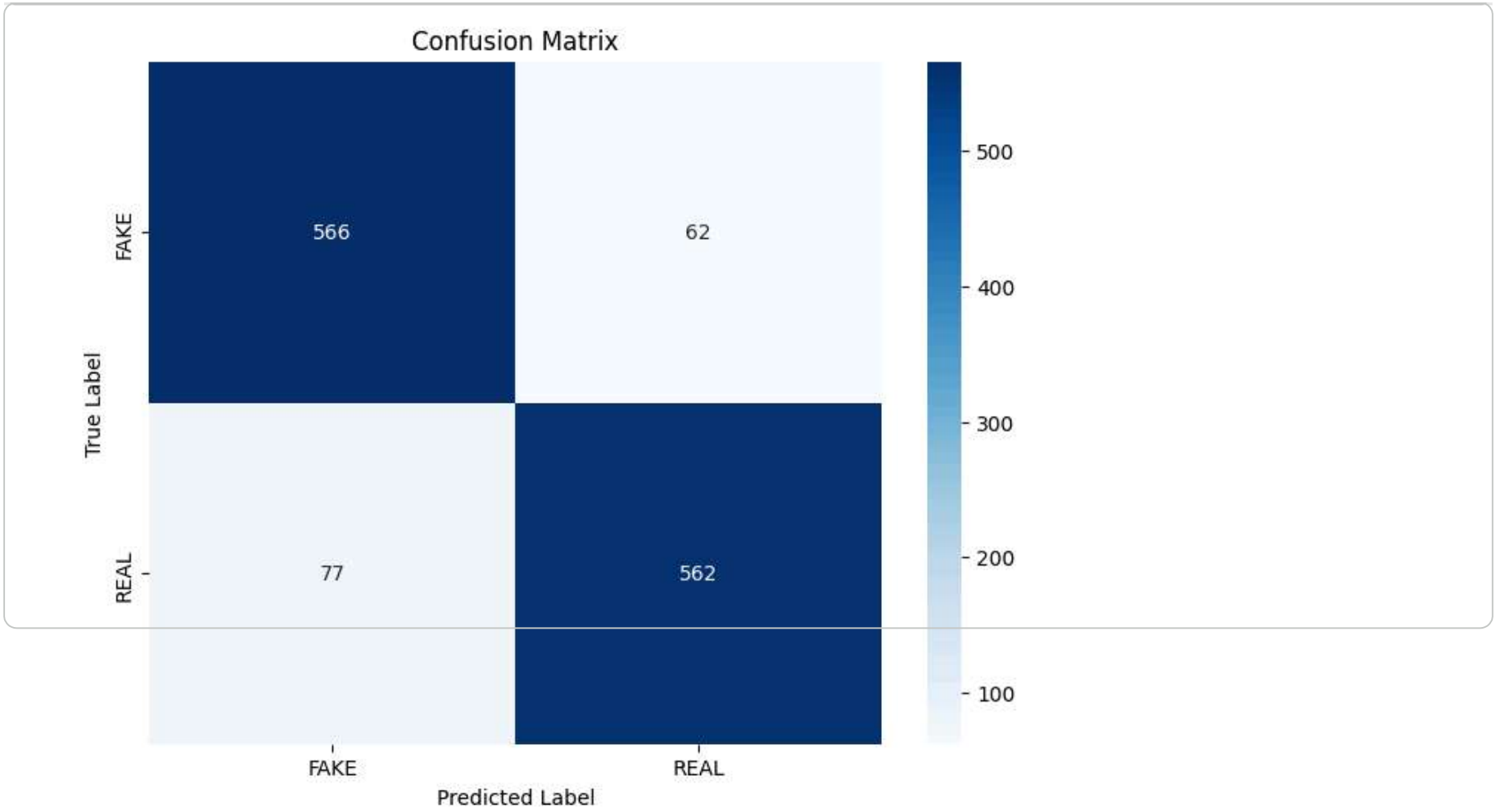
```
model = MultinomialNB()
model.fit(X_train, y_train)

print("Model trained successfully")
```

```
Model trained successfully
```

```
y_pred = model.predict(X_test)

accuracy = accuracy_score(y_test, y_pred)
precision = precision_score(y_test, y_pred, average='weighted')
recall = recall_score(y_test, y_pred, average='weighted')
f1 = f1_score(y_test, y_pred, average='weighted')

print("Accuracy:", accuracy)
print("Precision:", precision)
print("Recall:", recall)
print("F1-score:", f1)

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

```
Accuracy: 0.8902920284135754
Precision: 0.8905334513660044
Recall: 0.8902920284135754
F1-score: 0.8902879278784668

Confusion Matrix:
 [[566  62]
 [ 77 562]]

Classification Report:
              precision    recall  f1-score   support

        FAKE       0.88      0.90      0.89       628
        REAL       0.90      0.88      0.89       639

    accuracy                           0.89      1267
   macro avg       0.89      0.89      0.89      1267
weighted avg       0.89      0.89      0.89      1267
```

```
import matplotlib.pyplot as plt
import seaborn as sns

# Get the confusion matrix
cm = confusion_matrix(y_test, y_pred)

# Get class labels from y_test (or y_train if y_test is not available yet)
class_labels = np.unique(y_test)

plt.figure(figsize=(8, 6))
sns.heatmap(cm, annot=True, fmt='d', cmap='Blues',
            xticklabels=class_labels, yticklabels=class_labels)
plt.title('Confusion Matrix')
plt.xlabel('Predicted Label')
plt.ylabel('True Label')
plt.show()
```


Confusion Matrix
566
62
77
562
FAKE
REAL
True Label
FAKE
REAL
Predicted Label
500
400
300
200
100