```
# Data handling
import pandas as pd
import numpy as np

# Text preprocessing
import string
import nltk
from nltk.corpus import stopwords

# Feature extraction
from sklearn.feature_extraction.text import TfidfVectorizer

# Model and evaluation
from sklearn.model_selection import train_test_split
from sklearn.naive_bayes import MultinomialNB
from sklearn.metrics import accuracy_score, classification_report, confusion

# Visualization
import matplotlib.pyplot as plt
import seaborn as sns

# Download stopwords
nltk.download('stopwords')
```

```
[nltk_data] Downloading package stopwords to /root/nltk_data...
[nltk_data]   Unzipping corpora/stopwords.zip.
True
```

```
# Load dataset
data = pd.read_csv("/content/news.csv")

# Display first 5 rows
data.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 `data` New interactive sheet

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

# Column names
print("Columns:", data.columns)

# Class distribution
print("\nClass distribution:")
print(data['label'].value_counts())
```

```
Dataset size: (6335, 4)
Columns: Index(['Unnamed: 0', 'title', 'text', 'label'], dtype='object')

Class distribution:
label
REAL    3171
FAKE    3164
Name: count, dtype: int64
```

```
# Drop index column
data.drop(columns=['Unnamed: 0'], inplace=True)

data.head()
```

	title	text	label
0	You Can Smell Hillary’s Fear	Daniel Greenfield, a Shillman Journalism Fello...	FAKE
1	Watch The Exact Moment Paul Ryan Committed Pol...	Google Pinterest Digg Linkedin Reddit Stumbleu...	FAKE
2	Kerry to go to Paris in gesture of sympathy	U.S. Secretary of State John F. Kerry said Mon...	REAL
3	Bernie supporters on Twitter erupt in anger ag...	— Kaydee King (@KaydeeKing) November 9, 2016 T...	FAKE
4	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 `data` New interactive sheet

```
# Combine title and text into one column
data['content'] = data['title'] + " " + data['text']

data[['content', 'label']].head()
```

	content	label
0	You Can Smell Hillary’s Fear Daniel Greenfield...	FAKE
1	Watch The Exact Moment Paul Ryan Committed Pol...	FAKE
2	Kerry to go to Paris in gesture of sympathy U....	REAL
3	Bernie supporters on Twitter erupt in anger ag...	FAKE
4	The Battle of New York: Why This Primary Matte...	REAL

```
# Load stopwords
stop_words = set(stopwords.words('english'))

def preprocess_text(text):
    # Convert to lowercase
    text = text.lower()

    # Remove punctuation
    text = text.translate(str.maketrans('', '', string.punctuation))

    # Remove stopwords
    words = text.split()
    words = [word for word in words if word not in stop_words]
```

```
    return " ".join(words)

# Apply preprocessing
data['clean_content'] = data['content'].apply(preprocess_text)

data[['clean_content', 'label']].head()
```

	clean_content	label
0	smell hillary’s fear daniel greenfield shillma...	FAKE
1	watch exact moment paul ryan committed politic...	FAKE
2	kerry go paris gesture sympathy us secretary s...	REAL
3	bernie supporters twitter erupt anger dnc trie...	FAKE
4	battle new york primary matters primary day ne...	REAL

```
# Input feature
X_text = data['clean_content']

# Target labels
y = data['label']
```

```
# Initialize TF-IDF vectorizer
tfidf = TfidfVectorizer(max_df=0.7)

# Convert text to numerical features
X = tfidf.fit_transform(X_text)

# Feature matrix shape
print("Feature matrix shape:", X.shape)

# Sample feature names
print("Sample features:", tfidf.get_feature_names_out()[:10])
```

```
Feature matrix shape: (6335, 84496)
Sample features: ['00' '000' '0000' '0000000031' '000000031' '0000035' '00001
 '00011' '00017b2908ff9fa45188d243fd49aaeeb2dhrcofficecom']
```

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

print("Training samples:", X_train.shape[0])
print("Testing samples:", X_test.shape[0])
```

```
Training samples: 5068
Testing samples: 1267
```

```
# Initialize Multinomial Naive Bayes
nb_model = MultinomialNB()

# Train the model
nb_model.fit(X_train, y_train)

print("Model trained successfully!")
```

```
Model trained successfully!
```

```
# Predict test data
y_pred = nb_model.predict(X_test)

# Show first 10 predictions
y_pred[:10]
```

```
array(['REAL', 'FAKE', 'REAL', 'FAKE', 'REAL', 'REAL', 'REAL', 'REAL',
       'FAKE', 'REAL'], dtype='<U4')
```

```
# Accuracy
accuracy = accuracy_score(y_test, y_pred)
print("Accuracy:", accuracy)
```

```
Accuracy: 0.8326756116811366
```

```
# Classification report
print("\nClassification Report:\n")
print(classification_report(y_test, y_pred))
```

```

Classification Report:

              precision    recall  f1-score   support

        FAKE       0.99      0.67      0.80       633
        REAL       0.75      0.99      0.86       634

    accuracy                           0.83      1267
   macro avg       0.87      0.83      0.83      1267
weighted avg       0.87      0.83      0.83      1267
```

```
# Confusion Matrix
cm = confusion_matrix(y_test, y_pred)
print(cm)
```

```
[[426 207]
 [  5 629]]
```

```
# Plot confusion matrix
sns.heatmap(cm, annot=True, fmt='d', cmap='YlGnBu',
            xticklabels=['FAKE', 'REAL'],
            yticklabels=['FAKE', 'REAL'])
plt.xlabel("Predicted")
plt.ylabel("Actual")
plt.title("Confusion Matrix")
plt.show()
```