Fake News Detection Using NLP - Detailed Source Code and Output

```
# Fake News Detection using NLP - Complete Source Code

# Step 1: Install Libraries
# pip install pandas numpy scikit-learn nltk flask joblib

# Step 2: Import Libraries
import pandas as pd
import numpy as np
import re
import nltk
from nltk.corpus import stopwords
from nltk.stem import WordNetLemmatizer
from sklearn.model_selection import train_test_split
from sklearn.feature_extraction.text import TfidfVectorizer
from sklearn.linear_model import PassiveAggressiveClassifier
from sklearn.metrics import accuracy_score, confusion_matrix
import joblib

# Step 3: Download NLTK Resources
nltk.download('stopwords')
nltk.download('wordnet')

# Step 4: Preprocessing Function
def preprocess(text):
    text = re.sub(r'\W', ' ', text.lower())
    words = text.split()
    lemmatizer = WordNetLemmatizer()
    stop_words = set(stopwords.words('english'))
    words = [lemmatizer.lemmatize(w) for w in words if w not in stop_words]
    return ' '.join(words)

# Step 5: Load and Preprocess Dataset
df = pd.read_csv("fake_or_real_news.csv")  # Ensure this file contains 'text' and 'label' columns
df['text'] = df['text'].apply(preprocess)

# Step 6: Feature Extraction
X = df['text']
y = df['label']
tfidf = TfidfVectorizer(max_df=0.7)
X_tfidf = tfidf.fit_transform(X)

# Step 7: Train-Test Split
X_train, X_test, y_train, y_test = train_test_split(X_tfidf, y, test_size=0.2, random_state=42)

# Step 8: Model Training
model = PassiveAggressiveClassifier(max_iter=50)
model.fit(X_train, y_train)

# Step 9: Save the Model and Vectorizer
joblib.dump(model, 'model.pkl')
joblib.dump(tfidf, 'vectorizer.pkl')

# Step 10: Evaluation Output
y_pred = model.predict(X_test)
print("Accuracy:", accuracy_score(y_test, y_pred))
print("Confusion Matrix:\n", confusion_matrix(y_test, y_pred))
```

Sample Output:

Sample Output from Evaluation:

```
Accuracy: 0.96

Confusion Matrix:
[[946  29]
 [ 46 993]]
```