# Sentiment Analysis Project - Source Code

# Import required 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.feature\_extraction.text import TfidfVectorizer  
from sklearn.model\_selection import train\_test\_split  
from sklearn.naive\_bayes import MultinomialNB  
from sklearn.metrics import classification\_report, accuracy\_score  
import joblib  
  
# Download NLTK resources  
nltk.download('stopwords')  
nltk.download('wordnet')  
  
# Load dataset (you can replace this with your own CSV)  
df = pd.read\_csv("sentiment\_dataset.csv") # Make sure it has 'text' and 'label' columns  
  
# Preprocessing function  
def preprocess(text):  
 text = text.lower()  
 text = re.sub(r"[^a-zA-Z\s]", "", text)  
 words = text.split()  
 stop\_words = set(stopwords.words("english"))  
 words = [w for w in words if w not in stop\_words]  
 lemmatizer = WordNetLemmatizer()  
 words = [lemmatizer.lemmatize(w) for w in words]  
 return " ".join(words)  
  
# Apply preprocessing  
df['clean\_text'] = df['text'].apply(preprocess)  
  
# Vectorize text using TF-IDF  
vectorizer = TfidfVectorizer(max\_features=5000)  
X = vectorizer.fit\_transform(df['clean\_text'])  
y = df['label'] # Must be 0/1/2 or categories like Positive/Negative/Neutral  
  
# Train-test split  
X\_train, X\_test, y\_train, y\_test = train\_test\_split(X, y, test\_size=0.2, random\_state=42)  
  
# Model training  
model = MultinomialNB()  
model.fit(X\_train, y\_train)  
  
# Evaluation  
y\_pred = model.predict(X\_test)  
print("Accuracy:", accuracy\_score(y\_test, y\_pred))  
print(classification\_report(y\_test, y\_pred))  
  
# Save the model and vectorizer  
joblib.dump(model, "sentiment\_model.pkl")  
joblib.dump(vectorizer, "vectorizer.pkl")