```
"""
Created on Sun Oct 20 11:46:45 2024

@author: pooja
"""
import pandas as pd
df = pd.read_csv(r"C:\Users\pooja\Downloads\IMDB Dataset.csv.zip")
print(df.head())
print(df.isnull().sum())

import re
import nltk
from nltk.corpus import stopwords
from nltk.tokenize import word_tokenize
from nltk.stem import WordNetLemmatizer
nltk.download('punkt')
nltk.download('stopwords')
nltk.download('wordnet')
def clean_text(text):
    text = re.sub(r'[^A-Za-z\s]', '', text)
    text = text.lower()
    tokens = word_tokenize(text)
    stop_words = set(stopwords.words('english'))
    tokens = [word for word in tokens if word not in stop_words]
    lemmatizer = WordNetLemmatizer()
    tokens = [lemmatizer.lemmatize(word) for word in tokens]
    cleaned_text = ' '.join(tokens)
    return cleaned_text
df['cleaned_reviews'] = df['review'].apply(clean_text)
print(df['cleaned_reviews'].head())
from sklearn.feature_extraction.text import TfidfVectorizer
vectorizer = TfidfVectorizer(max_features=3000)
X = vectorizer.fit_transform(df['cleaned_reviews'])
print(vectorizer.get_feature_names_out())
from sklearn.preprocessing import LabelEncoder
encoder = LabelEncoder()
y = encoder.fit_transform(df['sentiment'])
print(y[:5])
from sklearn.model_selection import train_test_split
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.25, random_state=42)
print("Training data size:", X_train.shape)
print("Testing data size:", X_test.shape)
from sklearn.linear_model import LogisticRegression
from sklearn.metrics import accuracy_score, classification_report
model = LogisticRegression()
model.fit(X_train, y_train)
y_pred = model.predict(X_test)
accuracy = accuracy_score(y_test, y_pred)
print(f"Accuracy: {accuracy * 100:.2f}%")
print(classification_report(y_test, y_pred, target_names=encoder.classes_))
from sklearn.naive_bayes import MultinomialNB
nb_model = MultinomialNB()
nb_model.fit(X_train, y_train)
y_pred_nb = nb_model.predict(X_test)
accuracy_nb = accuracy_score(y_test, y_pred_nb)
print(f"Naive Bayes Accuracy: {accuracy_nb * 100:.2f}%")
```

```
from sklearn.metrics import confusion_matrix
y_pred = model.predict(X_test)
print("Accuracy:", accuracy_score(y_test, y_pred))
print("Classification Report:\n", classification_report(y_test, y_pred))
print("Confusion Matrix:\n", confusion_matrix(y_test, y_pred))
from sklearn.model_selection import GridSearchCV
param_grid = {'C': [0.1, 1, 10, 100]}
grid_model = GridSearchCV(LogisticRegression(), param_grid, cv=5)
grid_model.fit(X_train, y_train)
from sklearn.model_selection import cross_val_score
cross_val_scores = cross_val_score(LogisticRegression(C=grid_model.best_params_['C']), X, y, cv=5)
print("Cross-validation scores:", cross_val_scores)
print("Average cross-validation score:", cross_val_scores.mean())
import joblib
joblib.dump(grid_model.best_estimator_, 'sentiment_analysis_model.pkl')
```