```
import pandas as pd
import numpy as np
from sklearn.feature_extraction.text import CountVectorizer, TfidfVectorizer
from sklearn.model_selection import train_test_split
from sklearn.naive_bayes import MultinomialNB
from sklearn.metrics import accuracy_score, confusion_matrix
import seaborn as sns
import pickle

# Load the dataset (replace with your actual dataset path)
df = pd.read_csv('amazon_cell_phone_reviews.csv')  # Replace with your dataset path

# Checking the structure of the dataset (optional)
print(df.head())

# Assuming the dataset has 'review' column for text data and 'label' column for sentiment (positive/negative)
X = df['review']  # Features (text data)
y = df['label']  # Labels (positive/negative sentiment)

# Split the dataset into training and testing sets (70% training, 30% testing)
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.3, random_state=42)

# 1. CountVectorizer: Convert text to Bag of Words representation
count_vectorizer = CountVectorizer(stop_words='english', max_features=5000)  # Limit to top 5000 features for performance
count_train = count_vectorizer.fit_transform(X_train)
count_test = count_vectorizer.transform(X_test)

# 2. TF-IDF Vectorizer: Convert text to TF-IDF representation
tfidf_vectorizer = TfidfVectorizer(stop_words='english', max_features=5000)  # Limit to top 5000 features for performance
tfidf_train = tfidf_vectorizer.fit_transform(X_train)
tfidf_test = tfidf_vectorizer.transform(X_test)

# 3. Initialize Naive Bayes Classifier
nb_classifier = MultinomialNB()  # Using the default alpha=1.0

# 4. Train the model with Count Vectorizer data
print("Training model with CountVectorizer...")
nb_classifier.fit(count_train, y_train)

# 5. Evaluate the model with CountVectorizer on test data
count_pred = nb_classifier.predict(count_test)
count_score = accuracy_score(y_test, count_pred)  # Accuracy score
print(f'Accuracy with CountVectorizer: {count_score:.4f}')

# Confusion matrix for CountVectorizer
count_cm = confusion_matrix(y_test, count_pred, labels=['positive', 'negative'])
print(f'Confusion Matrix (CountVectorizer):\n{count_cm}')
sns.heatmap(count_cm, annot=True, fmt='d', cmap='Blues')  # Visualize the confusion matrix as a heatmap
sns.plt.show()

# 6. Train the model with TF-IDF Vectorizer data
print("Training model with TF-IDF Vectorizer...")
nb_classifier.fit(tfidf_train, y_train)

# 7. Evaluate the model with TF-IDF on test data
tfidf_pred = nb_classifier.predict(tfidf_test)
tfidf_score = accuracy_score(y_test, tfidf_pred)  # Accuracy score
print(f'Accuracy with TF-IDF Vectorizer: {tfidf_score:.4f}')

# Confusion matrix for TF-IDF
tfidf_cm = confusion_matrix(y_test, tfidf_pred, labels=['positive', 'negative'])
print(f'Confusion Matrix (TF-IDF):\n{tfidf_cm}')
sns.heatmap(tfidf_cm, annot=True, fmt='d', cmap='Blues')  # Visualize the confusion matrix as a heatmap
sns.plt.show()

# 8. Hyperparameter Tuning (Trying different alpha values for Naive Bayes)
alphas = [0.1, 0.5, 1.0, 2.0, 5.0]  # Different alpha values to try
for alpha in alphas:
    print(f"Training with alpha = {alpha}...")
    nb_classifier = MultinomialNB(alpha=alpha)
    nb_classifier.fit(tfidf_train, y_train)  # Train with TF-IDF data
    pred = nb_classifier.predict(tfidf_test)  # Predict on the test data
    score = accuracy_score(y_test, pred)  # Calculate accuracy score
    print(f'Alpha: {alpha}, Accuracy with TF-IDF: {score:.4f}')

# 9. Saving the trained model and vectorizers for future use
```

```
# Save the trained Naive Bayes model (for later use in predictions)
with open('nb_classifier.pkl', 'wb') as model_file:
    pickle.dump(nb_classifier, model_file)
print("Naive Bayes model saved as 'nb_classifier.pkl'.")

# Save the trained CountVectorizer and TF-IDF Vectorizer
with open('count_vectorizer.pkl', 'wb') as count_vectorizer_file:
    pickle.dump(count_vectorizer, count_vectorizer_file)
print("CountVectorizer saved as 'count_vectorizer.pkl'.")

with open('tfidf_vectorizer.pkl', 'wb') as tfidf_vectorizer_file:
    pickle.dump(tfidf_vectorizer, tfidf_vectorizer_file)
print("TF-IDF Vectorizer saved as 'tfidf_vectorizer.pkl'.")
```

Start coding or generate with AI.

Start coding or generate with AI.

Start coding or generate with AI.

Start coding or generate with AI.

Start coding or generate with AI.