Sentiment Analysis

Introduction:

This task demonstrates performing sentiment analysis on textual data using Natural Language Processing (NLP) techniques. Sentiment analysis classifies text data, such as tweets or reviews, into sentiment categories like positive, negative, or neutral. This helps organizations and researchers understand opinions expressed in large volumes of text efficiently.

The goal was to build a complete sentiment analysis pipeline including data preprocessing, model training, and performance evaluation. We used a sample dataset with labeled text samples, applied text cleaning techniques, converted text into numerical features using TF-IDF vectorization, and trained a Logistic Regression classifier to predict sentiments.

Code Implementation:

```
# Step 1: Import Required Libraries
import pandas as pd
import re
from nltk.corpus import stopwords
from nltk.stem.porter import PorterStemmer
from sklearn.model_selection import train_test_split
from sklearn.feature_extraction.text import TfidfVectorizer
from sklearn.linear_model import LogisticRegression
from sklearn.metrics import accuracy_score, classification_report, confusion_matrix
import nltk

# Download stopwords
nltk.download('stopwords')

# Step 2: Create Sample Dataset
data = {
    'text': [
        'I love this product! It is amazing.',
        'This is the worst experience I ever had.',
        'I feel okay about this.',
        'Absolutely fantastic service!',
        'Not good, not bad, just average.',
        'I am very happy with the quality.',
        'Terrible! I will never buy this again.',
        'It is fine, nothing special.',
        'Great value for money.',
        'The product broke after two days.'
    ],
    'sentiment': [
```

```
        'positive',
        'negative',
        'neutral',
        'positive',
        'neutral',
        'positive',
        'negative',
        'neutral',
        'positive',
        'negative'
    ]
}
df = pd.DataFrame(data)

# Step 3: Text Preprocessing Function
stop_words = set(stopwords.words('english'))
stemmer = PorterStemmer()

def preprocess_text(text):
    text = text.lower()
    text = re.sub(r'http\S+|www\S+|https\S+', '', text)  # Remove URLs
    text = re.sub(r'\@\w+|\#','', text)                  # Remove mentions and hashtags
    text = re.sub(r'[^a-z\s]', '', text)                 # Remove punctuation and numbers
    tokens = text.split()
    tokens = [stemmer.stem(word) for word in tokens if word not in stop_words]
    return " ".join(tokens)

# Apply preprocessing
df['clean_text'] = df['text'].apply(preprocess_text)

# Step 4: Feature Extraction with TF-IDF Vectorizer
tfidf = TfidfVectorizer(max_features=500)
X = tfidf.fit_transform(df['clean_text']).toarray()
y = df['sentiment']

# Step 5: Train-Test Split
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.3, random_state=42)

# Step 6: Model Training - Logistic Regression
model = LogisticRegression(max_iter=1000)
model.fit(X_train, y_train)

# Step 7: Predictions and Evaluation
y_pred = model.predict(X_test)
print("Accuracy:", accuracy_score(y_test, y_pred))
print("\nClassification Report:\n", classification_report(y_test, y_pred))
print("Confusion Matrix:\n", confusion_matrix(y_test, y_pred))
```

Conclusion:

In this task, a sentiment analysis pipeline was successfully built and tested on sample textual data. The preprocessing steps cleaned and normalized the raw text, improving the quality of input features. TF-IDF vectorization effectively converted text into numerical form suitable for classification.

The Logistic Regression model achieved an accuracy of approximately 67% on the test set, showing a moderate ability to classify sentiment. The evaluation metrics and confusion matrix revealed some misclassifications, highlighting challenges in distinguishing neutral sentiments and limited data size.

Overall, this task provided hands-on experience in applying NLP techniques for sentiment classification, including data cleaning, feature extraction, model implementation, and performance analysis. These foundational skills are essential for real-world applications involving opinion mining and text analytics.