AY: 2025-26

Class:	BE	**Semester:**	VII
Course Code:	CSDOL7011	**Course Name:**	Natural Language Processing

Name of Student:	Konisha Jayesh Thakare
Roll No. :	71
Experiment No.:	10
Title of the Experiment:	Design and Development of a Real-World NLP Application
Date of Performance:	16.09.2025
Date of Submission:	23.09.2025

Evaluation

Performance Indicator	Max. Marks	Marks Obtained
Performance	5	
Understanding	5	
Journal work and timely submission	10	
Total	20	

Performance Indicator	Exceed Expectations (EE)	Meet Expectations (ME)	Below Expectations(BE)
Performance	4-5	2-3	1
Understanding	4-5	2-3	1
Journal work and timely submission	8-10	5-8	1-4

Checked by

Name of Faculty : Dr. Tatwadarshi P. Nagarhalli

Signature :

Date :

Aim: To design and develop a real-world NLP application, implementing NLP techniques learned in previous experiments, and to evaluate its performance and usability.

Dataset Details:

Source: [IMDB Dataset of 50K Movie Reviews]

Size: 50,000 movie reviews

Columns:

- review: Text of the movie review
- sentiment: Label (positive or negative)

Train/Test Split: 80% training (40,000 reviews), 20% testing (10,000 reviews)

Theory:

Sentiment analysis is a classic NLP task to determine the sentiment expressed in text. In this experiment:

1. Text Preprocessing removes noise from raw reviews: lowercasing, removing punctuation, stopwords removal, and stemming using the Porter Stemmer.

2. Feature Extraction converts text into numerical vectors using TF-IDF, capturing importance of words relative to the document and corpus.

3. Model Training uses Multinomial Naïve Bayes, suitable for text classification with count-based features.

4. Evaluation is done using metrics:

   - Accuracy: Overall correctness of predictions
   - Precision, Recall, F1-score: Performance per class
   - Confusion Matrix: Correct vs. misclassified examples

Procedure:

1. Load IMDB dataset and map positive → 1, negative → 0.
2. Preprocess reviews:
   - Convert to lowercase
   - Remove punctuation

- Tokenize
- Remove stopwords
- Apply Porter Stemming

3. Convert processed reviews into TF-IDF vectors.
4. Split dataset into training (80%) and testing (20%) sets.
5. Train Multinomial Naïve Bayes classifier on the training set.
6. Predict sentiment on the test set.
7. Evaluate model performance using accuracy, classification report, and confusion matrix.
8. Build a simple interface function to predict sentiment of new user input.

Implementation:

```
import nltk
import pandas as pd
from sklearn.model_selection import train_test_split
from sklearn.feature_extraction.text import TfidfVectorizer
from sklearn.naive_bayes import MultinomialNB
from sklearn.metrics import accuracy_score, classification_report
import string
from nltk.corpus import stopwords
from nltk.stem import PorterStemmer

nltk.download('stopwords')
stop_words = set(stopwords.words('english'))
ps = PorterStemmer()

data = pd.read_csv("IMDB_Reviews.csv")
data['sentiment'] = data['sentiment'].map({'positive':1, 'negative':0})

def preprocess(text):
    text = text.lower()
    text = text.translate(str.maketrans('', '', string.punctuation))
    tokens = nltk.word_tokenize(text)
    tokens = [ps.stem(word) for word in tokens if word not in stop_words]
    return " ".join(tokens)

data['clean_text'] = data['review'].apply(preprocess)
tfidf = TfidfVectorizer()
X = tfidf.fit_transform(data['clean_text'])
y = data['sentiment']
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2,
random_state=42)
```

```
model = MultinomialNB()
model.fit(X_train, y_train)
y_pred = model.predict(X_test)
print("Accuracy:", accuracy_score(y_test, y_pred))
print("\nClassification Report:\n", classification_report(y_test, y_pred))

def predict_sentiment(text):
    text = preprocess(text)
    vector = tfidf.transform([text])
    pred = model.predict(vector)
    return "Positive" if pred[0] == 1 else "Negative"

user_input = "I loved the movie, it was fantastic!"
print("Sentiment Prediction:", predict_sentiment(user_input))
```

Results:

```
PS C:\Users\Konisha Thakare\OneDrive\Desktop\Python> python imdb_sentiment_classifier.py
[nltk_data] Downloading package stopwords to C:\Users\Konisha
[nltk_data]     Thakare\AppData\Roaming\nltk_data...
[nltk_data]   Package stopwords is already up-to-date!
Accuracy: 0.8641

Classification Report:
              precision    recall  f1-score   support

           0       0.85      0.88      0.86      4961
           1       0.88      0.85      0.86      5039

    accuracy                           0.86     10000
   macro avg       0.86      0.86      0.86     10000
weighted avg       0.86      0.86      0.86     10000

Sentiment Prediction: Positive
```

Example User Input:

```
"I loved the movie, it was fantastic!" → Predicted Sentiment: Positive
```

Interpretation:

- High accuracy (~86%) demonstrates the model can reliably distinguish positive and negative reviews.
- Balanced precision and recall indicate consistent performance across classes.
- The application can predict sentiment for unseen movie reviews effectively.

Conclusion:

This experiment successfully demonstrated the design and development of a real-world NLP application for sentiment analysis using the IMDB 50K movie reviews dataset. By combining preprocessing, TF-IDF feature extraction, and Multinomial Naïve Bayes classification, the model achieved 86.4% accuracy on the test set. The application can predict the sentiment of new reviews in real-time, demonstrating the practical utility of NLP techniques for real-world scenarios like movie review analysis. Proper experimental methodology, including data preprocessing, training/testing split, and performance evaluation, ensures reliable and reproducible results.