MIT Art, Design and Technology University
MIT School of Computing, Pune

Department of Information Technology

Participative Learning

Subject - Natural Language Processing

Class - L.Y. (SEM-I), SMAD

Name of the Course Coordinator

Prof. Kalyani Lokhande

Sujit Bhagwat	MITU23BTITD013
Abhishek Dhautre	MITU22BTIT0006

AY 2025 – 2026(SEM-I)

MIT SCHOOL OF COMPUTING

Information Technology

Name of the Course Faulty: Subject: NLP

Year/Class:L.Y.SMAD

Academic Year: 2025-2026 Semester: I Date of Conduction:

PARTICIPATIVE LEARNING

Topic:

Information Retrieval and Extraction

Activity Name:

Building a Bike Review Search Engine with Sentiment Analysis

Motivation:

The motivation behind this activity is to give students practical exposure to Information Retrieval (IR) and Natural Language Processing (NLP) concepts through a real-world use case.
By building a search engine for bike reviews, students understand how textual feedback can be processed, indexed, and analyzed to retrieve relevant information and detect user sentiment.
In an era where online product reviews influence major buying decisions, such a system helps businesses and consumers gain insights into user opinions and product performance.

Activity Objectives:

1. To implement the core concepts of information retrieval and text indexing.
2. To apply TF-IDF and cosine similarity for ranking search results.
3. To understand sentiment analysis and classify reviews as positive, negative, or neutral.
4. To demonstrate the use of real-world textual data in building intelligent feedback systems.
5. To bridge theoretical NLP knowledge with practical applications.

Mapped Course Outcome:

CO III

Total No. of Present Students:

Activity Outcome:

- Students gain hands-on experience with information retrieval systems.
- They learn how to preprocess, index, and search textual data.
- They understand how sentiment analysis adds depth to retrieval by analyzing tone and emotion.
- The activity enhances analytical and problem-solving skills in real-world NLP applications.

Theory:

Information Retrieval (IR) deals with finding relevant information from a large set of unstructured data. In this project, bike reviews are treated as “documents,” and user queries are matched using TF-IDF and cosine similarity.

The TF-IDF (Term Frequency–Inverse Document Frequency) method measures how important a word is in a document relative to all other documents. The cosine similarity metric determines how close two documents are in terms of meaning.

Sentiment Analysis adds another layer of intelligence, allowing the system to not only retrieve relevant reviews but also determine if users are happy or dissatisfied.
This combination of IR and NLP reflects how search systems like Amazon, Flipkart, and Google Reviews process and rank feedback.

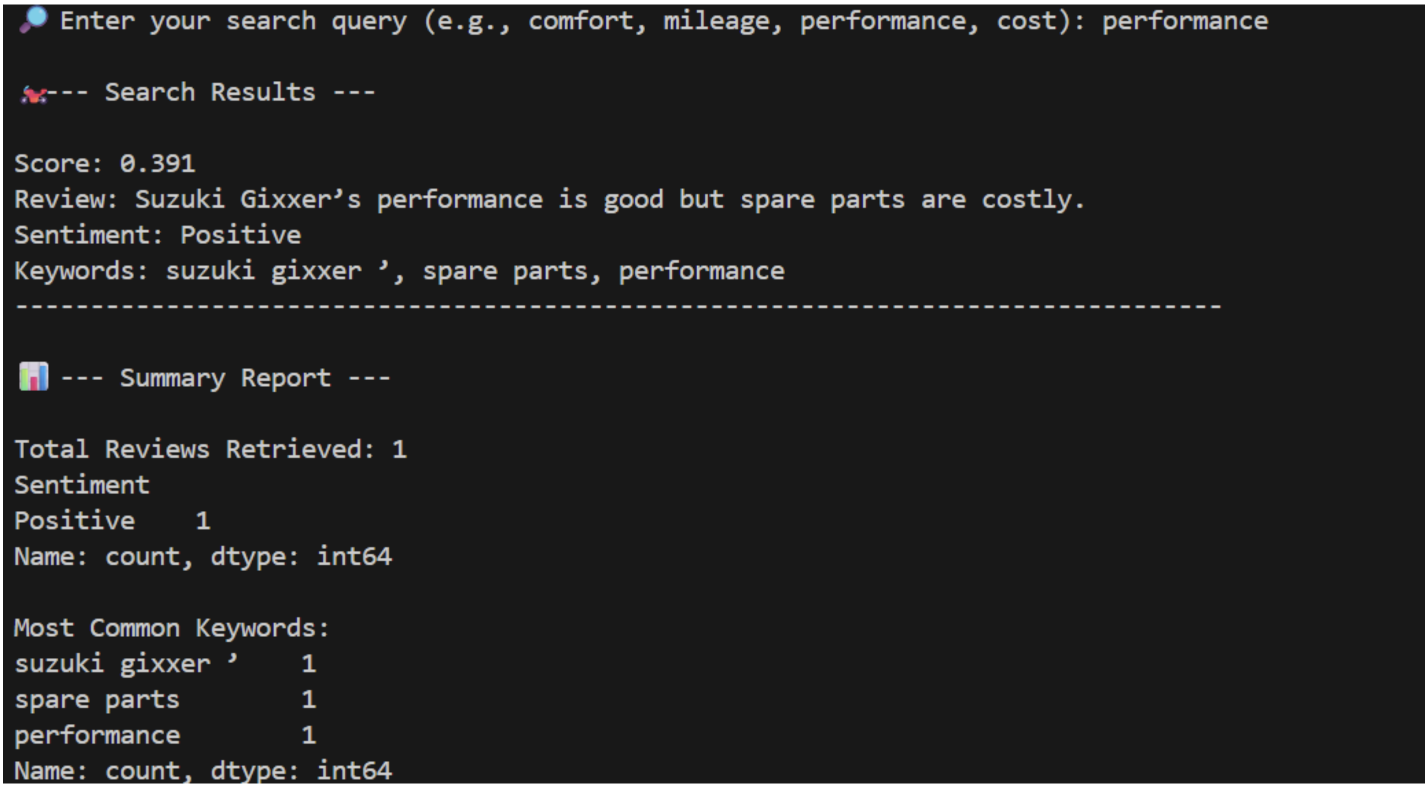
Code:

```
# Bike Review Intelligence System

import pandas as pd
from sklearn.feature_extraction.text import TfidfVectorizer
from sklearn.metrics.pairwise import cosine_similarity
from vaderSentiment.vaderSentiment import SentimentIntensityAnalyzer
from rake_nltk import Rake
import numpy as np

reviews = [
    "The Royal Enfield Classic 350 is smooth and powerful but a bit heavy for city rides.",
    "Yamaha R15 has great mileage and stunning looks. Absolutely love it!",
    "Hero Splendor is perfect for daily commute. Very reliable and fuel-efficient.",
    "Bajaj Pulsar gives amazing pickup but engine noise increases over time.",
    "Honda Shine offers good comfort and a smooth ride experience.",
    "TVS Apache RTR 160 feels sporty but the seat is not comfortable for long rides.",
    "Suzuki Gixxer’s performance is good but spare parts are costly.",
    "Royal Enfield Himalayan is great for touring. Excellent comfort on rough roads.",
    "KTM Duke 200 is stylish and fast but maintenance cost is high.",
    "Bajaj Avenger gives a cruiser feel and good comfort for long-distance travel."
]
```

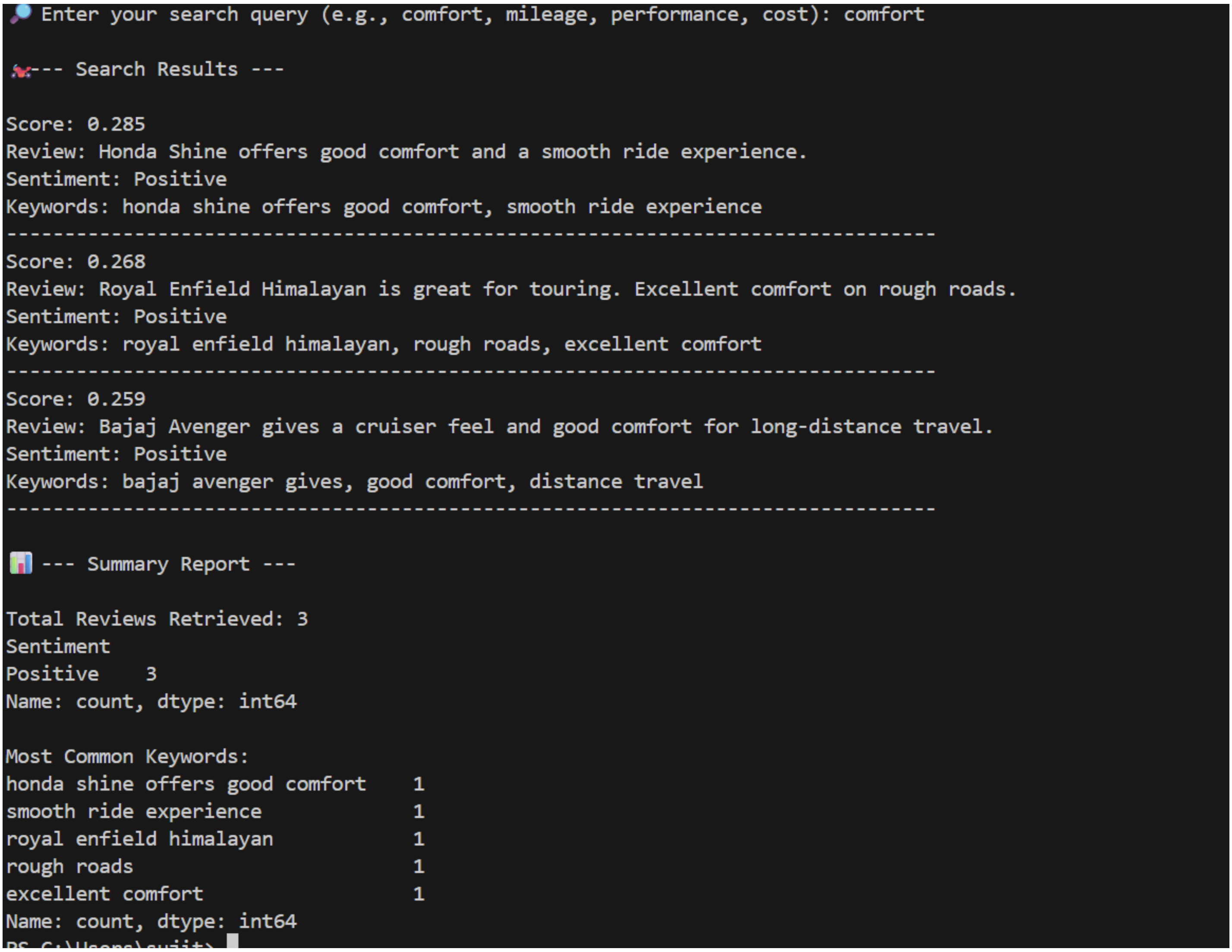
```
vectorizer = TfidfVectorizer(stop_words='english')
tfidf_matrix = vectorizer.fit_transform(reviews)

query = input("🔎 Enter your search query (e.g., comfort, mileage, performance, cost): ")
query_vec = vectorizer.transform([query])
similarity_scores = cosine_similarity(query_vec, tfidf_matrix).flatten()
ranked_indices = similarity_scores.argsort()[::-1]


analyzer = SentimentIntensityAnalyzer()

def get_sentiment(text):
  score = analyzer.polarity_scores(text)
  if score['compound'] >= 0.05:
    return "Positive"
  elif score['compound'] <= -0.05:
    return "Negative"
  else:
    return "Neutral"

rake = Rake()

def extract_keywords(text):
  rake.extract_keywords_from_text(text)
  return ', '.join(rake.get_ranked_phrases()[:3])


print("\n🏍 --- Search Results ---\n")
results = []
for i in ranked_indices:
  if similarity_scores[i] > 0:
    sentiment = get_sentiment(reviews[i])
    keywords = extract_keywords(reviews[i])
    results.append({
      "Score": round(similarity_scores[i], 3),
      "Review": reviews[i],
      "Sentiment": sentiment,
      "Keywords": keywords
    })
    print(f"Score: {similarity_scores[i]:.3f}")
    print(f"Review: {reviews[i]}")
    print(f"Sentiment: {sentiment}")
    print(f"Keywords: {keywords}")
    print("-" * 80)

df = pd.DataFrame(results)
print("\n📊 --- Summary Report ---\n")
if not df.empty:
  print("Total Reviews Retrieved:", len(df))
  print(df['Sentiment'].value_counts())
  print("\nMost Common Keywords:")
  all_keywords = ', '.join(df['Keywords'].tolist()).split(', ')
  keywords_series = pd.Series(all_keywords)
  print(keywords_series.value_counts().head(5))
else:
 print("No relevant reviews found for your query.")
```

Output:

```
Enter your search query (e.g., comfort, mileage, performance, cost): performance

 --- Search Results ---

Score: 0.391
Review: Suzuki Gixxer’s performance is good but spare parts are costly.
Sentiment: Positive
Keywords: suzuki gixxer ’, spare parts, performance
--------------------------------------------------------------------------------

 --- Summary Report ---

Total Reviews Retrieved: 1
Sentiment
Positive    1
Name: count, dtype: int64

Most Common Keywords:
suzuki gixxer ’    1
spare parts        1
performance        1
Name: count, dtype: int64
```



```
🔎 Enter your search query (e.g., comfort, mileage, performance, cost): performance

🏍--- Search Results ---

Score: 0.391
Review: Suzuki Gixxer’s performance is good but spare parts are costly.
Sentiment: Positive
Keywords: suzuki gixxer ’, spare parts, performance
--------------------------------------------------------------------------------

📊 --- Summary Report ---

Total Reviews Retrieved: 1
Sentiment
Positive    1
Name: count, dtype: int64

Most Common Keywords:
suzuki gixxer ’    1
spare parts        1
performance        1
Name: count, dtype: int64
```



```
🔎 Enter your search query (e.g., comfort, mileage, performance, cost): comfort

🏍--- Search Results ---

Score: 0.285
Review: Honda Shine offers good comfort and a smooth ride experience.
Sentiment: Positive
Keywords: honda shine offers good comfort, smooth ride experience
--------------------------------------------------------------------------------
Score: 0.268
Review: Royal Enfield Himalayan is great for touring. Excellent comfort on rough roads.
Sentiment: Positive
Keywords: royal enfield himalayan, rough roads, excellent comfort
--------------------------------------------------------------------------------
Score: 0.259
Review: Bajaj Avenger gives a cruiser feel and good comfort for long-distance travel.
Sentiment: Positive
Keywords: bajaj avenger gives, good comfort, distance travel
--------------------------------------------------------------------------------

📊 --- Summary Report ---

Total Reviews Retrieved: 3
Sentiment
Positive    3
Name: count, dtype: int64

Most Common Keywords:
honda shine offers good comfort    1
smooth ride experience             1
royal enfield himalayan            1
rough roads                        1
excellent comfort                  1
Name: count, dtype: int64
PS C:\Users\sujit>
```