SENTIMENT ANALYSIS

OBJECTIVE:

The objective of this project is to perform sentiment analysis on a textual dataset using SQL. The dataset, sourced from Kaggle, comprises statements of movie reviews. The analysis focuses on classifying the textual data into positive and negative sentiments using a keyword-based approach.

INTRODUCTION:

Sentiment analysis is a widely used technique for understanding and categorizing human emotions from textual data. This project explores a basic, keyword-based approach to sentiment analysis using SQL.

The primary goal of this beginner-level project is to classify these statements into positive and negative sentiments by identifying patterns using predefined keywords. The project employs SQL for simple data exploration and pattern-based classification, making it an accessible starting point for learning about sentiment analysis techniques.

The analysis involves:

- Exploring the distribution of sentiments within the dataset.
- Extracting subsets of data classified as positive or negative based on keywords.
- Evaluating the accuracy of the keyword-based classification.

This project provides a foundational understanding of sentiment analysis, offering a stepping stone for beginners to build upon and explore more advanced techniques such as Natural Language Processing (NLP) in the future.

TECHNOLOGY USED:

1. Database Management System

PostgreSQL: Used to store, manage, and query the dataset. Enabled efficient execution of SQL queries for data exploration, sentiment extraction, and analysis.

2. Query Language

SQL (Structured Query Language): Served as the primary tool for performing sentiment analysis. Facilitated keyword-based searches, data filtering, and classification.

3. Dataset (link of dataset)

Kaggle Dataset: A pre-labelled dataset containing textual statements with sentiments (positive or negative). Provided a structured format for applying SQL-based methods.

SCHEMA OF THE TABLE:

Table name: **sentiment**

Attributes:

1. id (int): serves as a primary key
2. statement (varchar): includes textual statements
3. sentiment (text): positive or negative

ANALYSIS THROUGH SQL:

```
--Getting overall idea of the data
select * from sentiment;
select count(sentiment) from sentiment;  --50000
select count(sentiment) from sentiment where sentiment='positive';  --25000
select count(sentiment) from sentiment where sentiment='negative';  --25000

--Extracting information based on some positive expressions and saved the data as "positive"
select * from sentiment where sentence like '%amazed%' or sentence like '%brilliant%' or sentence like '%fantastic%' or sentence like '%stunning%' or sentence like '%altruistic%' or sentence like '%omnibus%' or sentence like '%fond%' or sentence like '%Harry%' or sentence like '%master%' or sentence like '%pure%' or sentence like '%iconic%' or sentence like '%praise%' or sentence like '%realistic%' or sentence like '%rocked%' or sentence like '%Robert Downey%' or sentence like '%marvel%' or sentence like '%excite%' or sentence like  '%amazing%' or sentence like '%applause%' or sentence like '%recommend%' or sentence like '%Scarlette Johansson%' or sentence like '%Tom Cruise%' or sentence like '%perfect%' or sentence like '%wonderful%' or sentence like '%Will Smith%' or sentence like '%fulfilling%';

--Extracting information based on some negative expressions and saved the data as "negative"
select * from sentiment where sentence like '%awful%' or sentence like '%disturb%' or sentence like '%disappoint%' or sentence like '%garbage%' or sentence like '%grip%' or sentence like '%terrible%' or sentence like '%fake%' or sentence like '%oppress%' or sentence like '%pathetic%' or sentence like '%reluctant%' or sentence like '%pathetic%' or sentence like '%horrible%' or sentence like '%dull%' or sentence like '%ludicrous%' or sentence like '%corny%' or sentence like '%sucks%' or sentence like '%depress%' or sentence like '%frustrate%';

--Getting idea about positive data
select * from positive;
select count(sentiment) from positive; --19205
select count(sentiment) from positive where sentiment='positive';  --12523
select count(sentiment) from positive where sentiment='negative';  --6682

--Getting idea about negative data
select * from negative;
select count(sentiment) from negative; --15362
select count(sentiment) from negative where sentiment='positive';  --4557
select count(sentiment) from negative where sentiment='negative';  --10805
```

RESULT:
The project processes text data by categorizing it into positive or negative sentiments using predefined keywords or a sentiment scoring system stored in SQL tables. The results are stored and queried to analyse sentiment trends, such as the percentage of each sentiment type or the most common words associated with each sentiment.

- **Percentage of each sentiment type (accuracy):**

Positive sentiments correctly classified = **65.2%**
Correct positive sentiment rate = 12523/19205 ≈ 0.652

Negative sentiments correctly classified = **70.3%**
Correct negative sentiment rate = 10805/15362 ≈ 0.703

INSIGHTS:
The accuracy is decent for a dictionary-based approach but leaves rooms for improvement with advance techniques. And it misses some correct classification, especially for complex and sarcastic reviews.

CONCLUSION:

This project demonstrated the use of SQL for performing sentiment analysis on a movie review dataset. By leveraging a dictionary-based approach, the pipeline successfully classified reviews as positive or negative achieving an accuracy of 65-70%.

SQL proved to be an effective tool for text preprocessing and basic sentiment scoring.

Challenges such as handling sarcasm, negations, and limited vocabulary were identified, underscoring the limitations of rule-based methods for nuanced text analysis.

This project served as a valuable learning experience in implementing sentiment analysis using SQL, providing insights into text data handling, and setting a foundation for more sophisticated NLP techniques in future projects.