	Bansilal Ramnath Agarwal Charitable Trust's Vishwakarma Institute of Information Technology **Department of** **Artificial Intelligence and Data Science**

Name: **Siddhesh Dilip Khairnar**		
Class: **TY**	Division: **B**	Roll No: **372028**
Semester: **6th**	Academic Year: **2023-24**	
Subject Name & Code: **Natural Language and Processing & ADUA32203**		
Title of Assignment: **Perform single-word, multi-word based and polarity- based sentiment analysis**		
Date of Performance: **06-03-2024**	Date of Submission: **13-03-2024**	

ASSIGNMENT NO: - 7

Aim: Perform single-word, multi-word based and polarity-based sentiment analysis

❖ **THEORY:**

Sentiment analysis is the process of identifying and extracting subjective information from textual data, and categorizing the sentiment as positive, negative, or neutral. There are several approaches to perform sentiment analysis, including:

1) **Lexicon-Based Approach:** This approach uses a pre-defined list of words with positive or negative polarity to determine the sentiment of a piece of text. Each word is assigned a score based on its polarity, and the overall sentiment of the text is calculated as the sum of the scores of all the words.
2) **Machine Learning Approach:** This approach involves training a machine learning model on a labelled dataset of positive and negative reviews, and then using the model to classify the sentiment of new text. The model can be trained using various algorithms such as Naive Bayes, Support Vector Machines (SVM), or Neural Networks.
3) **Hybrid Approach:** This approach combines both lexicon based and machine learning techniques to improve the accuracy of sentiment analysis. The lexicon- based approach can be used to pre-process the text by identifying and removing negations and amplifiers, and then the machine learning algorithm can be used to classify the sentiment.

Code:

In [8]:
```
from textblob import TextBlob
```

In [9]:
```
# Define some negative words
negative_words = ["not", "terrible", "bad", "awful", "disappointing", "gloomy", "depressing"]
```

In [10]:
```
def sentiment_analysis(sentence, unigram=True, bigram=False, negative_marking=False):

  blob = TextBlob(sentence)
  sentiment = blob.sentiment

  # Unigram analysis
  if unigram:
    if sentiment.polarity > 0:
      return "positive"
    elif sentiment.polarity < 0:
      return "negative"
    else:
      return "neutral"

  # Bigram analysis
  if bigram and not unigram:
    words = sentence.split()
    for i in range(len(words) - 1):
      bigram = f"{words[i]} {words[i+1]}"
      if bigram in negative_words:
        sentiment.polarity -= 0.5  # Adjust polarity for negative bigram
    if sentiment.polarity > 0:
      return "positive"
    elif sentiment.polarity < 0:
      return "negative"
    else:
      return "neutral"

  # Apply negative marking
  if negative_marking:
    for word in negative_words:
      if word in sentence.lower():
        sentiment.polarity -= 0.2  # Adjust polarity for negative word
    if sentiment.polarity > 0:
```

```
      if word in sentence.lower():
        sentiment.polarity -= 0.2  # Adjust polarity for negative word
    if sentiment.polarity > 0:
      return "positive"
    elif sentiment.polarity < 0:
      return "negative"
    else:
      return "neutral"

  return str(sentiment)
```

```
# Example usage
sentences = [
  "The movie was fantastic!",
  "I waited in line for an hour, and the service was terrible.",
  "This new restaurant is a hidden gem.",
  "The weather today is gloomy and depressing.",
  "It wasn't the best movie, but it was okay."
]
```

```
for sentence in sentences:
  print(f"Sentence: {sentence}")
  print(f"  Unigram: {sentiment_analysis(sentence)}")
  print(f"  Bigram: {sentiment_analysis(sentence, unigram=False, bigram=True)}")
  print(f"  Negative Marking: {sentiment_analysis(sentence, negative_marking=True)}")
  print("")
```

- **Output**:

```
Sentence: The movie was fantastic!
  Unigram: positive
  Bigram: positive
  Negative Marking: positive

Sentence: I waited in line for an hour, and the service was terrible.
  Unigram: negative
  Bigram: negative
  Negative Marking: negative

Sentence: This new restaurant is a hidden gem.
  Unigram: negative
  Bigram: negative
  Negative Marking: negative
```

- **Conclusion:** In above assignment, we successfully implemented sentimental analysis using single-word, multi-word and check after making sentence negative again check for both cases. Learn about how NLP concepts are used in sentimental analysis to get emotion of user regarding product. Able to clear all concepts regarding sentimental analysis and all.