

## Experiment 2

**Aim: - Implement a Sentiment Analysis on linguistic data to study different feature extraction techniques like bag of words, TF\_IDF, word to vector and compare their performance.**

### Theory:

Sentiment analysis, also known as opinion mining, is a natural language processing task that involves determining the sentiment expressed in a piece of text, such as a review, comment, or tweet. The goal of sentiment analysis is to classify the text into different sentiment categories, such as positive, negative, neutral, or even more fine-grained sentiments.

To implement sentiment analysis and study different feature extraction techniques like bag of words, TF-IDF, and word embeddings (Word2Vec), we need to follow these general steps:

### Data Collection and Preprocessing:

Collect the linguistic data (text) along with their corresponding sentiment labels (e.g., positive, negative). Preprocess the text data by removing punctuation, converting to lowercase, removing stop words, and performing lemmatization or stemming.

### Different Feature Extraction Techniques:

Implement three different feature extraction techniques: bag of words, TF-IDF, and word embeddings (Word2Vec).

### Bag of Words (BoW) Feature Extraction:

Represent each document (piece of text) as a fixed-length vector, where each element corresponds to the frequency of a word in the document.

Create a vocabulary of unique words from the training data and assign an index to each word. Convert each document into a vector representation by counting the occurrences of each word from the vocabulary.

### TF-IDF Feature Extraction:

TF-IDF (Term Frequency-Inverse Document Frequency) is a numerical representation of a document's importance in a corpus of documents.

Calculate the TF-IDF value for each word in each document, which reflects its importance in the document relative to the entire corpus.

Here's how you can calculate TF-IDF for a term in a document:

**Department of Computer**

1. **Term Frequency (TF):** Calculate how often a term appears in a document.
2. **TF (term, document) = (Number of times the term appears in the document) / (Total number of terms in the document)**
3. **Inverse Document Frequency (IDF):** Calculate the logarithmically scaled inverse fraction of the documents that contain the term.
4. **IDF (term, corpus) =  $\log ((\text{Total number of documents in the corpus}) / (\text{Number of documents containing the term}))$**
5. **TF-IDF Score:** Multiply the TF value by the IDF value for a specific term in a specific document.
6. **TF-IDF (term, document, corpus) = TF (term, document) \* IDF (term, corpus)**

Here's a simple example with some numbers:

Let's say you have a corpus of 100 documents, and you want to calculate the TF-IDF score for the term "apple" in document #5.

- The term "apple" appears 10 times in document #5.
- Document #5 contains a total of 500 terms.
- The term "apple" appears in 30 out of the 100 documents in the corpus.

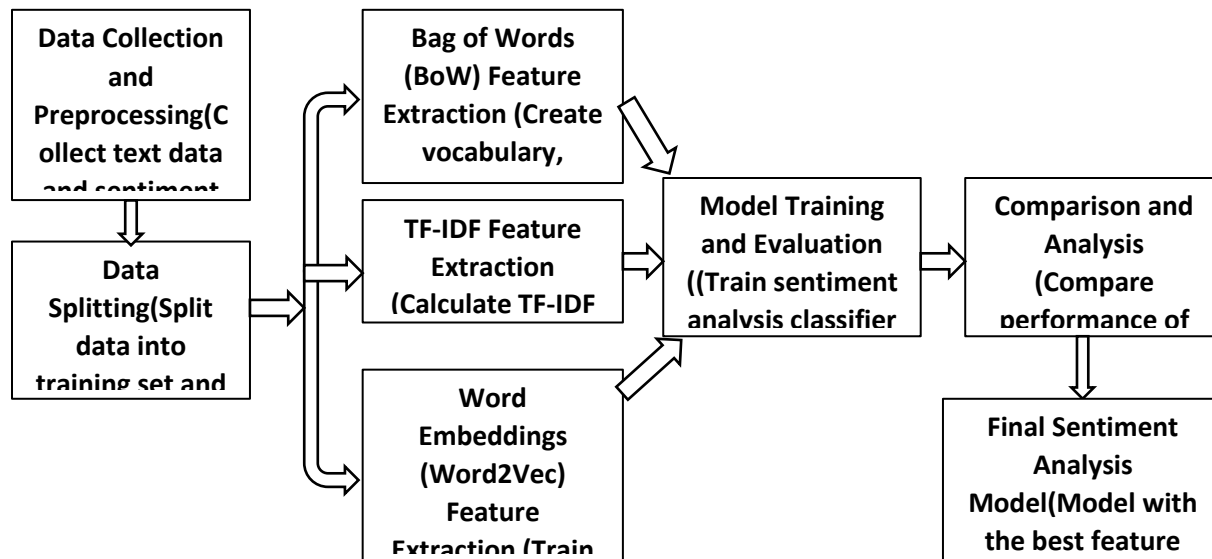
The calculations would be:

7. **TF (apple, document #5) =  $10 / 500 = 0.02$**
8. **IDF (apple, corpus) =  $\log (100 / 30) \approx 0.52$**
9. **TF-IDF (apple, document #5, corpus) =  $0.02 * 0.52 \approx 0.01$**

**Word Embeddings (Word2Vec) Feature Extraction:**

Word embeddings are dense vector representations of words that capture semantic meaning and relationships between words.

Train a Word2Vec model on the training data to create word embeddings for each word in the vocabulary.



### Lab Experiments to be Performed in This Session: -

**Exercise 1: Perform Sentiment Analysis to compare the feature extraction methods like bag of words, TF-IDF, Word to Vector**

**Data Set: Amazon Product Reviews Dataset**

1. **Dataset Selection:** Start by selecting an appropriate sentiment analysis dataset that contains text data labeled with positive or negative sentiments. Popular datasets include IMDB Movie Reviews, Twitter Sentiment Analysis Dataset, or Amazon Product Reviews.
2. **Preprocessing:** Clean and preprocess the text data by removing punctuation, converting text to lowercase, removing stop words, and performing tokenization. Ensure that the data is ready for analysis.
3. **Feature Extraction:**
  - a. **Bag of Words:** Convert the preprocessed text into a numerical vector representation using the Bag of Words model. Each document will be represented as a vector of word frequencies.
  - b. **TF-IDF:** Compute the Term Frequency-Inverse Document Frequency (TF-IDF) representation for the text data. TF-IDF accounts for the importance of words by considering both their frequency in a document and their rarity in the entire dataset.
  - c. **Word2Vec:** Train a Word2Vec model using the preprocessed text data to convert words into dense word embeddings.

4. Sentiment Analysis Model: Choose a suitable sentiment analysis model (e.g., Naive Bayes, Support Vector Machine, or a deep learning-based model) to classify the sentiment of the text data.
5. Experiment Setup: Split the dataset into training and testing sets. Ensure a consistent and fair evaluation by using the same train-test split across all three feature representations (Bag of Words, TF-IDF, and Word2Vec).
6. Model Training and Evaluation: Train the sentiment analysis model on each of the three feature representations separately and evaluate its performance on the test set using appropriate evaluation metrics such as accuracy, precision, recall, and F1-score.

**Exercise 2: Analyze and compare the performance of each feature representation on the sentiment analysis task. Consider factors such as accuracy, computational efficiency, and interpretability.**