from sklearn.feature\_extraction.text import CountVectorizer, TfidfVectorizer

from nltk.tokenize import word\_tokenize

from gensim.models import Word2Vec

# Sample data

data = [

"This is the first document.",

"This document is the second document.",

"And this is the third one.",

"Is this the first document?"

]

# Bag-of-Words Approach: Count Occurrence

count\_vectorizer = CountVectorizer()

count\_matrix = count\_vectorizer.fit\_transform(data)

count\_feature\_names = count\_vectorizer.get\_feature\_names\_out()

count\_array = count\_matrix.toarray()

print("Count Occurrence:")

print(count\_feature\_names)

print(count\_array)

# Bag-of-Words Approach: Normalized Count Occurrence

normalized\_count\_array = count\_array / count\_array.sum(axis=1, keepdims=True)

print("\nNormalized Count Occurrence:")

print(normalized\_count\_array)

# TF-IDF

tfidf\_vectorizer = TfidfVectorizer()

tfidf\_matrix = tfidf\_vectorizer.fit\_transform(data)

tfidf\_feature\_names = tfidf\_vectorizer.get\_feature\_names\_out()

tfidf\_array = tfidf\_matrix.toarray()

print("\nTF-IDF:")

print(tfidf\_feature\_names)

print(tfidf\_array)

# Word Embeddings using Word2Vec

tokenized\_data = [word\_tokenize(sentence.lower()) for sentence in data]

word2vec\_model = Word2Vec(sentences=tokenized\_data, vector\_size=100, window=5, min\_count=1, workers=4)

word\_vectors = [word2vec\_model.wv[word] for sentence in tokenized\_data for word in sentence]

print("\nWord Embeddings using Word2Vec:")

print(word\_vectors)