**EX NO:07**

**DATE:**

**TEXT ANALYSIS USING NLP**

**AIM:**

To preprocess textual data, analyze word frequency, apply TF-IDF vectorization, and

visualize important words using WordCloud for text mining and sentiment analysis.

**ALGORITHM:**

STEP 01: Install and import required libraries (NLTK, scikit-learn, pandas, matplotlib,

wordcloud).

STEP 02: Load the sample dataset into a Pandas DataFrame.

STEP 03: Preprocess the text:

 Convert to lowercase.

 Remove punctuation and special characters.

 Tokenize text.

 Remove stopwords.

STEP 04: Perform word frequency analysis to find the most common words.

STEP 05: Apply TF-IDF vectorization to extract important words from each document.

STEP 06: Display top TF-IDF words for each document.

STEP 07: Generate and visualize a WordCloud of all documents.

STEP 08: Display the ouput.

**CODING:**

# Step 1: Install Required Libraries

!pip install nltk scikit-learn pandas matplotlib wordcloud --quiet

# Step 2: Import Libraries

import pandas as pd

import re

import nltk

from nltk.corpus import stopwords

from sklearn.feature\_extraction.text import TfidfVectorizer

import matplotlib.pyplot as plt

from wordcloud import WordCloud

# Download NLTK stopwords

nltk.download('stopwords')

# Step 3: Load Updated Dataset

data = {

'text': [

"The delivery was super quick and smooth!",

"Worst packaging I have ever seen",

"The product quality exceeded my expectations",

"I would not recommend this to anyone",

"Absolutely worth the price",

"Terrible customer service experience",

"I am delighted with the purchase",

"The item broke after one use",

"One of the best deals I got online",

"Not satisfied with the durability",

"Extremely happy with the performance",

"Totally a waste of money",

"I enjoyed the overall shopping process",

"Very disappointing build quality",

"Excellent support team and guidance",

"The color was completely different from the picture",

"Highly recommend for everyone",

"I regret spending money on this"

]

}

df = pd.DataFrame(data)

print("Original Data:")

print(df)

# Step 4: Text Preprocessing

stop\_words = set(stopwords.words('english'))

def preprocess(text):

text = text.lower() # lowercase

text = re.sub(r'[^a-z\s]', '', text) # remove punctuation/numbers

tokens = text.split() # tokenize

tokens = [word for word in tokens if word not in stop\_words] # remove stopwords

return " ".join(tokens)

df['clean\_text'] = df['text'].apply(preprocess)

print("\nCleaned Text:")

print(df[['text', 'clean\_text']])

# Step 5: Word Frequency Analysis

all\_words = ' '.join(df['clean\_text']).split()

word\_freq = pd.Series(all\_words).value\_counts()

print("\nTop 10 Most Frequent Words:")

print(word\_freq.head(10))

# Step 6: TF-IDF Vectorization

vectorizer = TfidfVectorizer()

X = vectorizer.fit\_transform(df['clean\_text'])

feature\_names = vectorizer.get\_feature\_names\_out()

# Show top 5 TF-IDF words per document

print("\nTop 5 TF-IDF Words per Document:")

for i, doc in enumerate(df['clean\_text']):

tfidf\_scores = X[i].toarray()[0]

top\_indices = tfidf\_scores.argsort()[-5:][::-1]

top\_words = [(feature\_names[idx], tfidf\_scores[idx]) for idx in top\_indices if tfidf\_scores[idx] > 0]

print(f"Document {i+1}: {top\_words}")

# Step 7: WordCloud Visualization

text\_combined = ' '.join(df['clean\_text'])

wordcloud = WordCloud(width=800, height=400, background\_color='white').generate(text\_combined)

plt.figure(figsize=(12, 6))

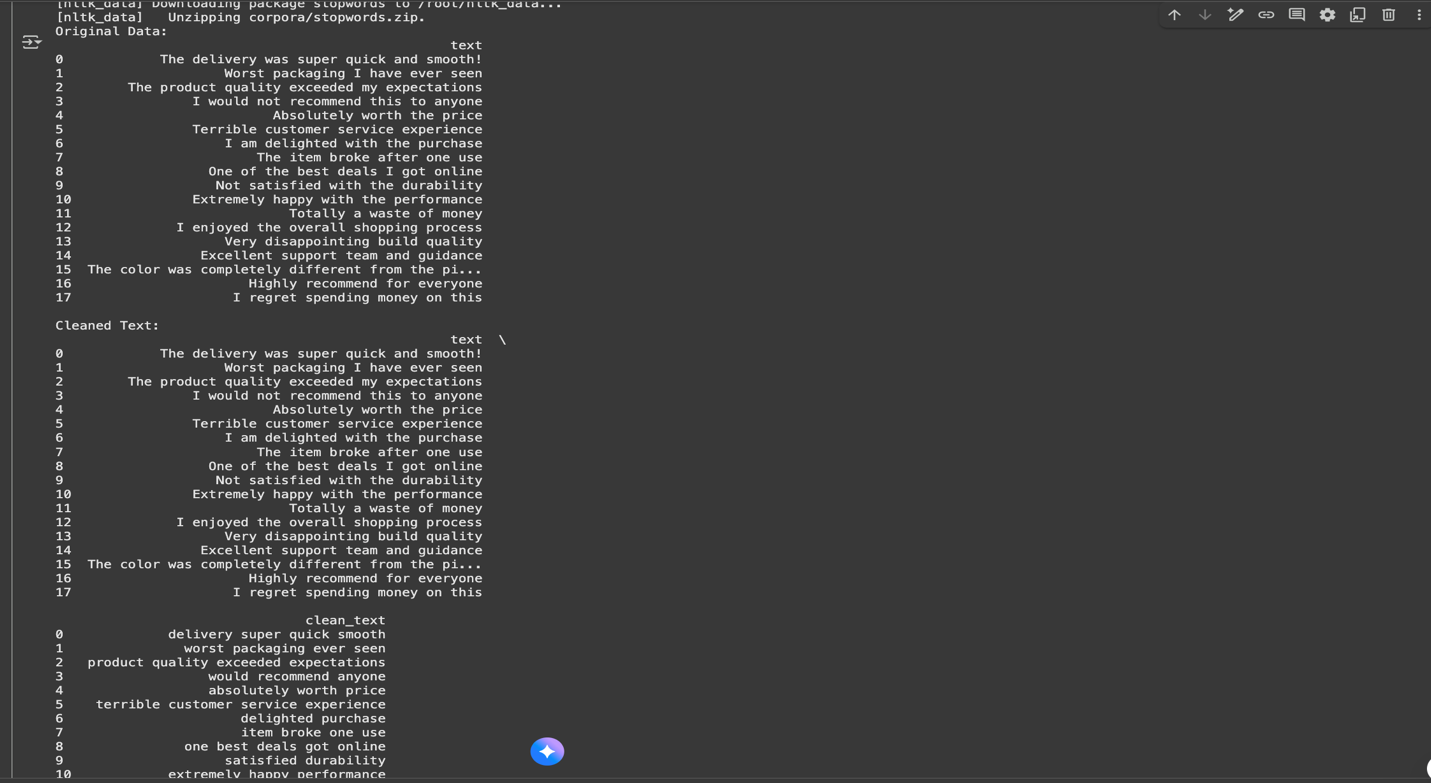
plt.imshow(wordcloud, interpolation='bilinear')

plt.axis('off')

plt.title("WordCloud of All Documents", fontsize=16)

plt.show()

**Output:**







|  |  |
| --- | --- |
| COE (20) |  |
| Record (20) |  |
| VIVA (10) |  |
| Total (50) |  |

**RESULT:**

The program successfully cleaned the text dataset, displayed frequent words, extracted

top TF-IDF keywords for each document, and visualized the dataset using a WordCloud.