**Creating Data set :**

import pandas as pd

# Creating the dataset

data = {

'Tweet': [

'I absolutely love the new features in this update!',

'Why is this product so expensive? Not worth it at all.',

'The event was amazing, had a great time!',

'Customer service is terrible, I’m very disappointed.',

'Just tried the new app, it’s pretty smooth and user-friendly.',

'The latest update broke everything, very frustrating.',

'Had a great experience with this brand, highly recommended!',

'Not sure what all the hype is about, it’s just okay.',

'This product has changed my life for the better!',

'Unreliable service, not going to use it again.'

],

'Timestamp': [

'01-08-2024 10:15', '02-08-2024 14:23', '03-08-2024 09:30',

'04-08-2024 16:45', '05-08-2024 11:00', '06-08-2024 19:20',

'07-08-2024 08:00', '08-08-2024 12:45', '09-08-2024 17:30',

'10-08-2024 13:15'

],

'User': [

'@user1', '@user2', '@user3', '@user4', '@user5',

'@user6', '@user7', '@user8', '@user9', '@user10'

]

}

df = pd.DataFrame(data)

# Convert Timestamp to datetime

df['Timestamp'] = pd.to\_datetime(df['Timestamp'], format='%d-%m-%Y %H:%M')

# Display the DataFrame

df.head()

**Data Preprocessing:**

import re

import nltk

from nltk.corpus import stopwords

# Download NLTK resources

nltk.download('stopwords')

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

# Function to clean tweet text

def clean\_tweet(tweet):

tweet = re.sub(r'http\S+', '', tweet) # Remove URLs

tweet = re.sub(r'@\w+', '', tweet) # Remove mentions

tweet = re.sub(r'#', '', tweet) # Remove hashtags

tweet = re.sub(r'\W', ' ', tweet) # Remove special characters

tweet = tweet.lower() # Convert to lowercase

tweet = ' '.join([word for word in tweet.split() if word not in stop\_words]) # Remove stopwords

return tweet

# Apply cleaning function

df['Cleaned\_Tweet'] = df['Tweet'].apply(clean\_tweet)

df.head()

from nltk.sentiment.vader import SentimentIntensityAnalyzer

# Initialize VADER sentiment analyzer

nltk.download('vader\_lexicon')

sid = SentimentIntensityAnalyzer()

# Function to get sentiment score

def get\_sentiment\_score(tweet):

return sid.polarity\_scores(tweet)['compound']

# Apply sentiment analysis to each cleaned tweet

df['Sentiment\_Score'] = df['Cleaned\_Tweet'].apply(get\_sentiment\_score)

# Classify sentiment based on the score

df['Sentiment'] = df['Sentiment\_Score'].apply(lambda score: 'Positive' if score > 0 else ('Negative' if score < 0 else 'Neutral'))

df.head()

import matplotlib.pyplot as plt

import seaborn as sns

# Set the size of the plot

plt.figure(figsize=(10, 5))

# Plot the sentiment trends

sns.lineplot(x=daily\_sentiment.index, y=daily\_sentiment.values, marker='o')

plt.title('Sentiment Trend Over Time')

plt.xlabel('Date')

plt.ylabel('Average Sentiment Score')

plt.xticks(rotation=45)

plt.grid(True)

plt.show()

df[['Tweet', 'Timestamp', 'User', 'Cleaned\_Tweet', 'Sentiment\_Score', 'Sentiment']]