* **Data Cleaning:**

import re

import pdfplumber

from nltk.tokenize import word\_tokenize

from nltk.corpus import stopwords

from docx import Document

# Download required NLTK data

nltk.download('punkt')

nltk.download('stopwords')

# Define a regex pattern for the date format

date\_pattern = re.compile(r'(?:[A-Za-z]+, [A-Za-z]+ \d{1,2}, \d{4})')

# Function to insert newlines before and after each date in the text

def insert\_newlines(text):

# Add newlines before and after each date match

text\_with\_newlines = date\_pattern.sub(r'\n\n\n\g<0>\n', text)

return text\_with\_newlines

# Function to preprocess text

def preprocess\_text(text):

# Normalize text

text = text.lower()

text = re.sub(r'(?<!\w)([A-Za-z])\s+', r'\1', text)

text = re.sub(r'\s+', ' ', text)

text = re.sub(r'[^\w\s\d]', '', text)

# Tokenize text without stemming

tokens = word\_tokenize(text)

# Remove stop words

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

filtered\_tokens = [word for word in tokens if word not in stop\_words]

return ' '.join(filtered\_tokens)

# Read the PDF file using pdfplumber

def extract\_text\_from\_pdf(pdf\_path):

text\_content = []

with pdfplumber.open(pdf\_path) as pdf:

for page in pdf.pages:

text\_content.append(page.extract\_text())

return ' '.join(text\_content).replace('\n', ' ')

# Define the path for your PDF file and output file

pdf\_file\_path = r'C:\Users\DELL\Desktop\Project\Untitled Folder\Anne-Frank-The-Diary-Of-A-Young-Girl.pdf'

output\_word\_file = 'Filtered\_Anne-Frank-The-Diary-Of-A-Young-Girl.docx'

# Extract text from PDF

pdf\_text = extract\_text\_from\_pdf(pdf\_file\_path)

# Add newlines before and after dates

pdf\_text\_with\_newlines = insert\_newlines(pdf\_text)

# Now preprocess text excluding the dates

# Split the text by newlines, preprocess non-date lines, and rejoin the text

processed\_lines = [preprocess\_text(line) for line in pdf\_text\_with\_newlines.split('\n')]

preprocessed\_text = '\n'.join(processed\_lines)

# Create a Word document

doc = Document()

doc.add\_paragraph(preprocessed\_text)

# Save the Word document

doc.save(output\_word\_file)

print("Preprocessing complete. Output saved to", output\_word\_file)

* **Extracting emotions with counts , words associated with it and sentimental score :**

import csv

import docx

from nltk.corpus import stopwords

from nltk.tokenize import word\_tokenize

# Step 1: Preprocess the Text

diary\_file\_path = r'C:\Users\DELL\Desktop\Project\Untitled Folder\Filtered\_Anne-Frank-The-Diary-Of-A-Young-Girl.docx' # Replace with the path to your Word document

# Read the diary text from the document

doc = docx.Document(diary\_file\_path)

diary\_text = " ".join([paragraph.text for paragraph in doc.paragraphs])

# Tokenize and preprocess the text

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

tokens = word\_tokenize(diary\_text.lower())

preprocessed\_tokens = [token for token in tokens if token.isalpha() and token not in stop\_words]

# Step 2: Load the NRC Emotion Lexicon

nrc\_lexicon\_path = r'C:\Users\DELL\Desktop\Project\Untitled Folder\NRC-emotion-lexicon-wordlevel-alphabetized-v0.92.txt' # Replace with the path to your downloaded lexicon

# Create a dictionary to store the emotion mappings

nrc\_lexicon = {}

with open(nrc\_lexicon\_path, "r") as file:

for line in file:

word, emotion, value = line.strip().split("\t")

if word in nrc\_lexicon:

nrc\_lexicon[word][emotion] = int(value)

else:

nrc\_lexicon[word] = {emotion: int(value)}

# Step 3: Calculate Sentiment Score

emotions = {'joy': set(), 'sadness': set(), 'anger': set(), 'fear': set(), 'trust': set(), 'disgust': set(),

'surprise': set(), 'anticipation': set(), 'excitement': set()}

sentiment\_score = 0

for word in preprocessed\_tokens:

if word in nrc\_lexicon:

emotions\_found = nrc\_lexicon[word]

for emotion, sentiment in emotions\_found.items():

if sentiment == 1 and emotion in emotions:

emotions[emotion].add(word)

sentiment\_score += sentiment

# Step 4: Write Output to CSV

output\_file = 'output.csv'

with open(output\_file, 'w', newline='') as csvfile:

writer = csv.writer(csvfile)

writer.writerow(['Emotion', 'Count', 'Words', 'Sentiment Score'])

for emotion, words in emotions.items():

count = len(words)

distinct\_words = ", ".join(words)

emotion\_sentiment\_score = sum([nrc\_lexicon[word][emotion] for word in words])

writer.writerow([emotion.capitalize(), count, distinct\_words, emotion\_sentiment\_score])

writer.writerow(['Combined Sentiment Score:', sentiment\_score])

print(f"Output written to {output\_file} in CSV format.")

**Extracting dates and emotions accordingly**

import csv

import docx

import re

from nltk.corpus import stopwords

from nltk.tokenize import word\_tokenize

# Load the NRC Emotion Lexicon

nrc\_lexicon\_path = r'C:\Users\DELL\Desktop\Project\Untitled Folder\NRC-emotion-lexicon-wordlevel-alphabetized-v0.92.txt'

nrc\_lexicon = {}

with open(nrc\_lexicon\_path, "r") as file:

for line in file:

word, emotion, value = line.strip().split("\t")

if word in nrc\_lexicon:

nrc\_lexicon[word][emotion] = int(value)

else:

nrc\_lexicon[word] = {emotion: int(value)}

# Function to preprocess text and extract emotions

def extract\_emotions\_from\_text(text):

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

tokens = word\_tokenize(text.lower())

preprocessed\_tokens = [token for token in tokens if token.isalpha() and token not in stop\_words]

emotions = {'joy': set(), 'sadness': set(), 'anger': set(), 'fear': set(),

'trust': set(), 'disgust': set(), 'surprise': set(), 'anticipation': set(), 'excitement': set()}

for word in preprocessed\_tokens:

if word in nrc\_lexicon:

emotions\_found = nrc\_lexicon[word]

for emotion, sentiment in emotions\_found.items():

if sentiment == 1 and emotion in emotions:

emotions[emotion].add(word)

return emotions

# Function to find date pairs and extract emotions and words between them

def extract\_emotions\_between\_date\_pairs(diary\_text):

date\_entries = re.findall(r"([A-Za-z]+day [A-Za-z]+ \d{1,2} \d{4})", diary\_text)

emotions\_data\_between\_dates = []

for i in range(len(date\_entries) - 1):

date1 = date\_entries[i]

date2 = date\_entries[i + 1]

# Find the text between two dates

text\_between\_dates = re.search(f"{date1}(.+?){date2}", diary\_text, re.DOTALL)

if text\_between\_dates:

text\_between\_dates = text\_between\_dates.group(1).strip()

# Extract emotions from the text between dates

emotions\_data = extract\_emotions\_from\_text(text\_between\_dates)

# Append date range to the emotions data

emotions\_data['Date Range'] = f"{date1} to {date2}"

emotions\_data\_between\_dates.append(emotions\_data)

return emotions\_data\_between\_dates

# Specify the document path

docx\_path = r'C:\Users\DELL\Desktop\Project\Untitled Folder\Filtered\_Anne-Frank-The-Diary-Of-A-Young-Girl.docx'

# Read the diary text from the document

doc = docx.Document(docx\_path)

diary\_text = "\n".join([paragraph.text for paragraph in doc.paragraphs])

# Extract emotions between date pairs

emotions\_between\_dates = extract\_emotions\_between\_date\_pairs(diary\_text)

# Specify the CSV file path

csv\_file\_path = 'emotions\_between\_dates.csv'

# Write emotions data to CSV

with open(csv\_file\_path, mode='w', newline='', encoding='utf-8') as file:

writer = csv.writer(file)

# Write header

header = ['Date Range', 'Joy Count', 'Joy Words', 'Sadness Count', 'Sadness Words',

'Anger Count', 'Anger Words', 'Fear Count', 'Fear Words',

'Trust Count', 'Trust Words', 'Disgust Count', 'Disgust Words',

'Surprise Count', 'Surprise Words', 'Anticipation Count', 'Anticipation Words',

'Excitement Count', 'Excitement Words']

writer.writerow(header)

# Write data rows

for emotions\_data in emotions\_between\_dates:

row = [emotions\_data['Date Range']]

for emotion in emotions.keys():

count = len(emotions\_data[emotion])

distinct\_words = ", ".join(emotions\_data[emotion])

row.extend([count, distinct\_words])

writer.writerow(row)

print(f"Emotions data between date pairs saved to {csv\_file\_path}")

**Everything (Date , emotions, words , word count, sentiment score) with sentiment labels**

import csv

import docx

import re

from nltk.corpus import stopwords

from nltk.tokenize import word\_tokenize

# Load the NRC Emotion Lexicon

nrc\_lexicon\_path = r'C:\Users\DELL\Desktop\Project\Untitled Folder\NRC-emotion-lexicon-wordlevel-alphabetized-v0.92.txt'

nrc\_lexicon = {}

with open(nrc\_lexicon\_path, "r") as file:

for line in file:

word, emotion, value = line.strip().split("\t")

if word in nrc\_lexicon:

nrc\_lexicon[word][emotion] = int(value)

else:

nrc\_lexicon[word] = {emotion: int(value)}

# Function to preprocess text and extract emotions

def extract\_emotions\_from\_text(text):

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

tokens = word\_tokenize(text.lower())

preprocessed\_tokens = [token for token in tokens if token.isalpha() and token not in stop\_words]

emotions = {'joy': set(), 'sadness': set(), 'anger': set(), 'fear': set(),

'trust': set(), 'disgust': set(), 'surprise': set(), 'anticipation': set()}

for word in preprocessed\_tokens:

if word in nrc\_lexicon:

emotions\_found = nrc\_lexicon[word]

for emotion, sentiment in emotions\_found.items():

if sentiment == 1 and emotion in emotions:

emotions[emotion].add(word)

return emotions

# Function to calculate sentiment score

def calculate\_sentiment\_score(emotions\_data):

positive\_emotions = ['joy', 'trust', 'anticipation']

negative\_emotions = ['sadness', 'anger', 'fear', 'disgust']

positive\_score = sum(len(emotions\_data[emotion]) for emotion in positive\_emotions)

negative\_score = sum(len(emotions\_data[emotion]) for emotion in negative\_emotions)

sentiment\_score = positive\_score - negative\_score

return sentiment\_score

# Function to label sentiment based on sentiment score

def label\_sentiment(sentiment\_score):

if sentiment\_score > 10:

return "Hightly Positive"

elif sentiment\_score >= 1:

return "Positive"

elif sentiment\_score == 0:

return "Neutral"

elif sentiment\_score >= -10:

return "Negative"

else:

return "Hightly Negative"

# Function to find date pairs and extract emotions and words between them

def extract\_emotions\_between\_date\_pairs(diary\_text):

date\_entries = re.findall(r"([A-Za-z]+day [A-Za-z]+ \d{1,2} \d{4})", diary\_text)

emotions\_data\_between\_dates = []

for i in range(len(date\_entries) - 1):

date1 = date\_entries[i]

date2 = date\_entries[i + 1]

# Find the text between two dates

text\_between\_dates = re.search(f"{date1}(.+?){date2}", diary\_text, re.DOTALL)

if text\_between\_dates:

text\_between\_dates = text\_between\_dates.group(1).strip()

# Extract emotions from the text between dates

emotions\_data = extract\_emotions\_from\_text(text\_between\_dates)

# Calculate sentiment score

sentiment\_score = calculate\_sentiment\_score(emotions\_data)

# Label sentiment

sentiment\_label = label\_sentiment(sentiment\_score)

emotions\_data['Sentiment Score'] = sentiment\_score

emotions\_data['Sentiment Label'] = sentiment\_label

# Append date range to the emotions data

emotions\_data['Date Range'] = f"{date1} to {date2}"

emotions\_data\_between\_dates.append(emotions\_data)

return emotions\_data\_between\_dates

# Specify the document path

docx\_path = r'C:\Users\DELL\Desktop\Project\Untitled Folder\Filtered\_Anne-Frank-The-Diary-Of-A-Young-Girl.docx'

# Read the diary text from the document

doc = docx.Document(docx\_path)

diary\_text = "\n".join([paragraph.text for paragraph in doc.paragraphs])

# Extract emotions between date pairs

emotions\_between\_dates = extract\_emotions\_between\_date\_pairs(diary\_text)

# Specify the CSV file path

csv\_file\_path = 'emotions\_between\_dates\_with\_sentiment.csv'

# Write emotions data to CSV

with open(csv\_file\_path, mode='w', newline='', encoding='utf-8') as file:

writer = csv.writer(file)

# Write header

header = ['Date Range', 'Joy Count', 'Joy Words', 'Sadness Count', 'Sadness Words',

'Anger Count', 'Anger Words', 'Fear Count', 'Fear Words',

'Trust Count', 'Trust Words', 'Disgust Count', 'Disgust Words',

'Surprise Count', 'Surprise Words', 'Anticipation Count', 'Anticipation Words',

'Sentiment Score', 'Sentiment Label']

writer.writerow(header)

# Write data rows

for emotions\_data in emotions\_between\_dates:

row = [emotions\_data['Date Range']]

for emotion\_key in emotions\_data.keys():

if emotion\_key not in ['Date Range', 'Sentiment Score', 'Sentiment Label']:

count = len(emotions\_data[emotion\_key])

distinct\_words = ", ".join(emotions\_data[emotion\_key])

row.extend([count, distinct\_words])

row.extend([emotions\_data['Sentiment Score'], emotions\_data['Sentiment Label']])

writer.writerow(row)

print(f"Emotions data between date pairs with sentiment saved to {csv\_file\_path}")

* **With date Seperated**

import csv

import docx

import re

from nltk.corpus import stopwords

from nltk.tokenize import word\_tokenize

# Load the NRC Emotion Lexicon

nrc\_lexicon\_path = r'C:\Users\DELL\Desktop\Project\Untitled Folder\NRC-emotion-lexicon-wordlevel-alphabetized-v0.92.txt'

nrc\_lexicon = {}

with open(nrc\_lexicon\_path, "r") as file:

for line in file:

word, emotion, value = line.strip().split("\t")

if word in nrc\_lexicon:

nrc\_lexicon[word][emotion] = int(value)

else:

nrc\_lexicon[word] = {emotion: int(value)}

# Function to preprocess text and extract emotions

def extract\_emotions\_from\_text(text):

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

tokens = word\_tokenize(text.lower())

preprocessed\_tokens = [token for token in tokens if token.isalpha() and token not in stop\_words]

emotions = {'joy': set(), 'sadness': set(), 'anger': set(), 'fear': set(),

'trust': set(), 'disgust': set(), 'surprise': set(), 'anticipation': set()}

for word in preprocessed\_tokens:

if word in nrc\_lexicon:

emotions\_found = nrc\_lexicon[word]

for emotion, sentiment in emotions\_found.items():

if sentiment == 1 and emotion in emotions:

emotions[emotion].add(word)

return emotions

# Function to calculate sentiment score

def calculate\_sentiment\_score(emotions\_data):

positive\_emotions = ['joy', 'trust', 'anticipation']

negative\_emotions = ['sadness', 'anger', 'fear', 'disgust']

positive\_score = sum(len(emotions\_data[emotion]) for emotion in positive\_emotions)

negative\_score = sum(len(emotions\_data[emotion]) for emotion in negative\_emotions)

sentiment\_score = positive\_score - negative\_score

return sentiment\_score

# Function to label sentiment based on sentiment score

def label\_sentiment(sentiment\_score):

if sentiment\_score > 10:

return "Highly Positive"

elif sentiment\_score >= 1:

return "Positive"

elif sentiment\_score == 0:

return "Neutral"

elif sentiment\_score >= -10:

return "Negative"

else:

return "Highly Negative"

# Function to find date pairs and extract emotions and words between them

def extract\_emotions\_between\_date\_pairs(diary\_text):

date\_entries = re.findall(r"([A-Za-z]+day [A-Za-z]+ \d{1,2} \d{4})", diary\_text)

emotions\_data\_between\_dates = []

for i in range(len(date\_entries) - 1):

date1 = date\_entries[i]

date2 = date\_entries[i + 1]

# Find the text between two dates

text\_between\_dates = re.search(f"{date1}(.+?){date2}", diary\_text, re.DOTALL)

if text\_between\_dates:

text\_between\_dates = text\_between\_dates.group(1).strip()

# Extract emotions from the text between dates

emotions\_data = extract\_emotions\_from\_text(text\_between\_dates)

# Calculate sentiment score

sentiment\_score = calculate\_sentiment\_score(emotions\_data)

# Label sentiment

sentiment\_label = label\_sentiment(sentiment\_score)

emotions\_data['Sentiment Score'] = sentiment\_score

emotions\_data['Sentiment Label'] = sentiment\_label

# Append start and end dates to the emotions data

emotions\_data['Start Date'] = date1

emotions\_data['End Date'] = date2

emotions\_data\_between\_dates.append(emotions\_data)

return emotions\_data\_between\_dates

# Specify the document path

docx\_path = r'C:\Users\DELL\Desktop\Project\Untitled Folder\Filtered\_Anne-Frank-The-Diary-Of-A-Young-Girl.docx'

# Read the diary text from the document

doc = docx.Document(docx\_path)

diary\_text = "\n".join([paragraph.text for paragraph in doc.paragraphs])

# Extract emotions between date pairs

emotions\_between\_dates = extract\_emotions\_between\_date\_pairs(diary\_text)

# Specify the CSV file path

csv\_file\_path = 'emotions\_between\_dates\_with\_sentiment.csv'

# Write emotions data to CSV

with open(csv\_file\_path, mode='w', newline='', encoding='utf-8') as file:

writer = csv.writer(file)

# Write header

header = ['Start Date', 'End Date', 'Joy Count', 'Joy Words', 'Sadness Count', 'Sadness Words',

'Anger Count', 'Anger Words', 'Fear Count', 'Fear Words',

'Trust Count', 'Trust Words', 'Disgust Count', 'Disgust Words',

'Surprise Count', 'Surprise Words', 'Anticipation Count', 'Anticipation Words',

'Sentiment Score', 'Sentiment Label']

writer.writerow(header)

# Write data rows

for emotions\_data in emotions\_between\_dates:

row = [emotions\_data['Start Date'], emotions\_data['End Date']]

for emotion\_key in emotions\_data.keys():

if emotion\_key not in ['Start Date', 'End Date', 'Sentiment Score', 'Sentiment Label']:

count = len(emotions\_data[emotion\_key])

distinct\_words = ", ".join(emotions\_data[emotion\_key])

row.extend([count, distinct\_words])

row.extend([emotions\_data['Sentiment Score'], emotions\_data['Sentiment Label']])

writer.writerow(row)

print(f"Emotions data between date pairs with sentiment saved to {csv\_file\_path}")

import pandas as pd

import matplotlib.pyplot as plt

from matplotlib.dates import DateFormatter

# Replace 'your\_input.csv' with the actual path to your CSV file

csv\_file\_path = r'C:\Users\DELL\Desktop\Project\Untitled Folder\emotions\_between\_dates\_with\_sentiment.csv'

# Read the CSV file into a DataFrame

df = pd.read\_csv(csv\_file\_path)

df['Start Date'] = pd.to\_datetime(df['Start Date'])

# Sort DataFrame by date

df = df.sort\_values(by='Start Date')

# Define emotions and corresponding colors

emotions = ['Joy', 'Sadness', 'Anger', 'Fear', 'Trust', 'Disgust', 'Surprise', 'Anticipation']

colors = ['#1a75ff', '#00802b', '#b30000', '#cc6600', '#4d0099', '#66b2ff', '#ff9900', '#993366']

# Holocaust-related events during World War II

holocaust\_events = [

{'event': 'Wannsee Conference', 'date': '1942-01-20'},

{'event': 'Operation Reinhard (Aktion Reinhard)', 'date': '1942-03-17'},

{'event': 'Warsaw Ghetto Uprising', 'date': '1943-04-19'},

{'event': 'Liquidation of Auschwitz', 'date': '1944-01-27'},

# Add more events as needed

]

# Create a subplot with a line chart for the evolution of emotion counts over time

fig, ax = plt.subplots(figsize=(10, 6))

# Add traces for each emotion

for i, emotion in enumerate(emotions):

ax.plot(df['Start Date'], df[f'{emotion} Count'], label=emotion, color=colors[i])

# Add traces for Holocaust-related events

for event in holocaust\_events:

event\_date = pd.to\_datetime(event['date'])

ax.axvline(event\_date, color='red', linestyle='--', label=f'{event["event"]}')

# Update layout

ax.set\_title('Evolution of Emotion Counts over Time with Holocaust-related Events')

ax.set\_xlabel('Start Date')

ax.set\_ylabel('Emotion Count')

ax.legend(loc='upper left', bbox\_to\_anchor=(1, 1))

# Rotate date labels

ax.xaxis.set\_major\_formatter(DateFormatter("%Y-%m-%d"))

plt.xticks(rotation=45)

# Display the chart

plt.tight\_layout()

plt.show()