

Sentimental Analysis Model

Developed by – ANUBHAV MISHRA



November 16, 2023

Telaverge Communications india pvt ltd.

Bangalore India - 560066

CONTENTS

1. Objective
2. Hardware Requirement
3. Tools Used
4. Introduction
5. Algorithm
6. Code
7. Working of Model
8. Limitations & Future Scope
9. References and attachments
10. Conclusion

Objective

Do the sentimental analysis of any brand/product. Get the sentimental score on a daily basis after scanning the internet.

Firstly, assign a base sentimental score. Then, on the next day/next time when we do the sentimental analysis of a particular brand/product. It will automatically modify the score and send the same on mail to the owner (configured email-id). But, along with the mail we must send top 5 reasons for score increment/ score decrement.

Example:

1. List the top 5 reasons for score decrement – Alert the users

Reason 1: Someone has put a bad comment on Facebook.

(Score: 10 🡺 9.9 🡺Alert the user 🡺 By sending mail

…and so on.

1. List the top 5 reasons for score increment – (Score:10 🡺10.1)

Reason 1: Number of followers increased in a year.

Reason 2: New research article is published on its website.

…and so on.

Hardware Used

1. Processor –Pentium 4 processor with or above 2.4GHz.
2. Hard-disk –20 G.B. or above.
3. RAM -256 M.B. or more (recommended).
4. Display option – Monitor & Input devices – Keyboard, mouse.
5. LAN or internet connection.

Tools Used

1. Visual Studio Code (Latest Updated Version)
2. Python (version: 3.0 and above)
3. Command Prompt
4. Meta for Developers
5. Meta Account (Specially, for Facebook and Instagram) & Gmail Account

Introduction

* In today's digital age, text analysis and text mining have become essential part of various industries. Text analysis refers to the process of analyzing and extracting meaningful insights from unstructured text data. One of the most important subfield of text analysis is sentiment analysis, which involves determining the emotional tone of the text.
* Sentiment analysis is a technique used to determine the emotional tone or sentiment expressed in a text. It involves analyzing the words and phrases used in the text to identify the underlying sentiment, whether it is positive, negative, or neutral.
* Sentiment analysis has numerous practical applications, from brand monitoring to customer feedback analysis. Python is a popular programming language used for text analysis and mining, and the Natural Language Toolkit (NLTK) library is one of the most widely used libraries for natural language processing in Python.

Algorithms

Here, for developing Sentimental Analysis Model we have used Lexicon-based analysis. It basically fetch comments from a particular brand Facebook page and accordingly assign a sentimental score and mail it to the configured mail-id.

Step 1. Developed an API over meta for developers platform and generate the access token and page id from there.

Step 2. Then, using the above credentials we will fetch all the comments into a csv file name “fb\_comment.csv”

Step 3. Now, we will take comments one by one and generate its sentimental score after importing the vaderSentiment class from Python NLTK Library.

e.g., from vaderSentiment.vaderSentiment import SentimentIntensityAnalyzer

Step 4. After comparing all the scores for each comment, find whether the comment is of positive, negative, or neutral type.

Step 5. Update the scores on the basis of comment type and send the same along with the top 5 incremental/decremental reason on configured email-id.

Code

import requests

import csv

import pandas as pd

import smtplib

import ssl

def get\_facebook\_comments(page\_id, access\_token):

    base\_url = f'https://graph.facebook.com/v13.0/{page\_id}/posts'

    params = {

        'fields': 'comments{id,message,created\_time,from}',

        'access\_token': access\_token,

    }

    comments\_list = []

    # Fetch posts

    response = requests.get(base\_url, params=params)

    data = response.json()

    # Extract comments

    for post in data['data']:

        if 'comments' in post:

            comments = post['comments']['data']

            comments\_list.extend(comments)

    while 'paging' in data and 'next' in data['paging']:

        # Fetch next page of posts

        response = requests.get(data['paging']['next'])

        data = response.json()

        # Extract comments

        for post in data['data']:

            if 'comments' in post:

                comments = post['comments']['data']

                comments\_list.extend(comments)

    return comments\_list

def save\_comments\_to\_csv(comments, csv\_filename):

    with open(csv\_filename, 'w', newline='', encoding='utf-8') as csvfile:

        fieldnames = ['comment\_id', 'message', 'created\_time']

        writer = csv.DictWriter(csvfile, fieldnames=fieldnames)

        writer.writeheader()

        for comment in comments:

            writer.writerow({

                'comment\_id': comment['id'],

                'message': comment.get('message', ''),

                'created\_time': comment['created\_time']

                #'from\_name': comment['from']['name']

            })

if \_\_name\_\_ == '\_\_main\_\_':

    page\_id = 'brand facebook page id'

    access\_token = 'API access token of admin’s page'

    csv\_filename = 'fb\_comment.csv'

    comments = get\_facebook\_comments(page\_id, access\_token)

    save\_comments\_to\_csv(comments, csv\_filename)

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

file\_path = 'fb\_comment.csv'

# Replace 'desired\_column' with the name of the column you want to select

desired\_column = 'message'

# Read the CSV file into a DataFrame

df = pd.read\_csv(file\_path)

# Select the desired column

selected\_column = df[desired\_column].tolist()

sscore=10

m\_message=[]

score2=''

# Now, 'selected\_column' contains the data from the specified column

from vaderSentiment.vaderSentiment import SentimentIntensityAnalyzer

# Iterate through each sentence in the list

analyzer = SentimentIntensityAnalyzer()

for sentence in selected\_column:

    vs = analyzer.polarity\_scores(sentence)

    print("{:-<65} {}".format(sentence, str(vs)))

    max\_value = max(vs['neg'], vs['pos'], vs['neu'])

    #print(max\_value)

    # Find the key for the specified value

    matching\_keys = [key for key, value in vs.items() if value == max\_value]

    #print(f"The maximum value in the dictionary is: {matching\_keys}")

    if matching\_keys==['neg']:

        sscore=(sscore-max\_value)

    elif matching\_keys==['pos']:

        sscore=(sscore+max\_value)

    else:

        sscore=(sscore)

    m\_message.append(sentence)

score2=str(sscore)

m\_score=score2[:6]

mail\_mssge=str(m\_message[:5:])

from email.message import EmailMessage

# Define email sender and receiver

email\_sender = 'sender\_email'

email\_password = 'sender\_email\_password'

email\_receiver = 'receiver\_email'

# Set the subject and body of the email

subject = 'Sentimental Score Analyzer!'

body = """Current Sentimental Score of your page is:""" + m\_score +""".

Reason for last updatation in score are due to following comment: """ + mail\_mssge

em = EmailMessage()

em['From'] = email\_sender

em['To'] = email\_receiver

em['Subject'] = subject

em.set\_content(body)

# Add SSL (layer of security)

context = ssl.create\_default\_context()

# Log in and send the email

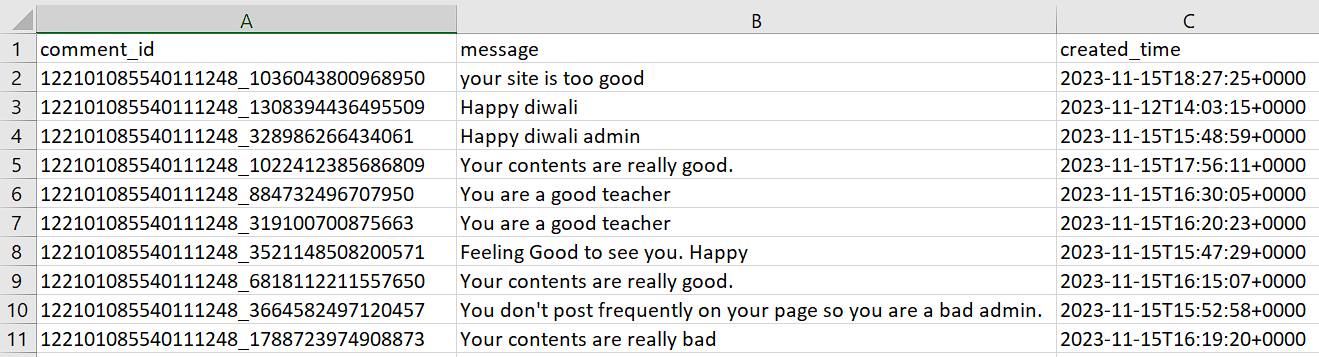
with smtplib.SMTP\_SSL('smtp.gmail.com', 465, context=context) as smtp:

    smtp.login(email\_sender, email\_password)

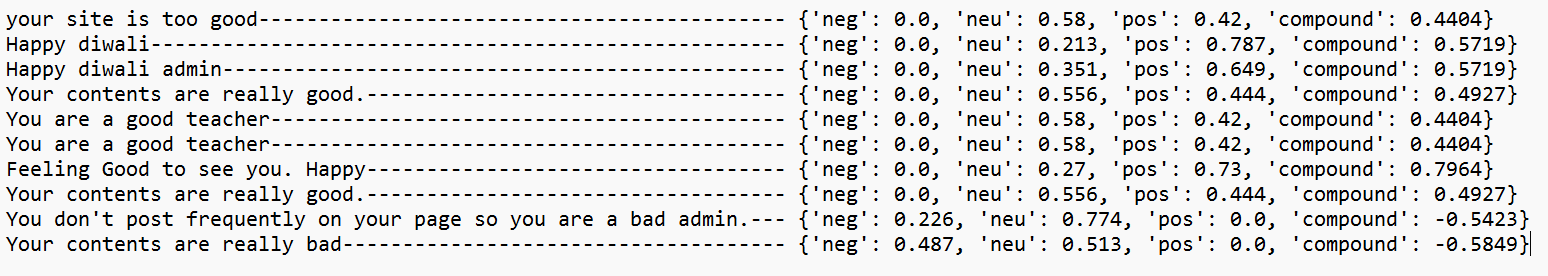
    smtp.sendmail(email\_sender, email\_receiver, em.as\_string())

Working of Model

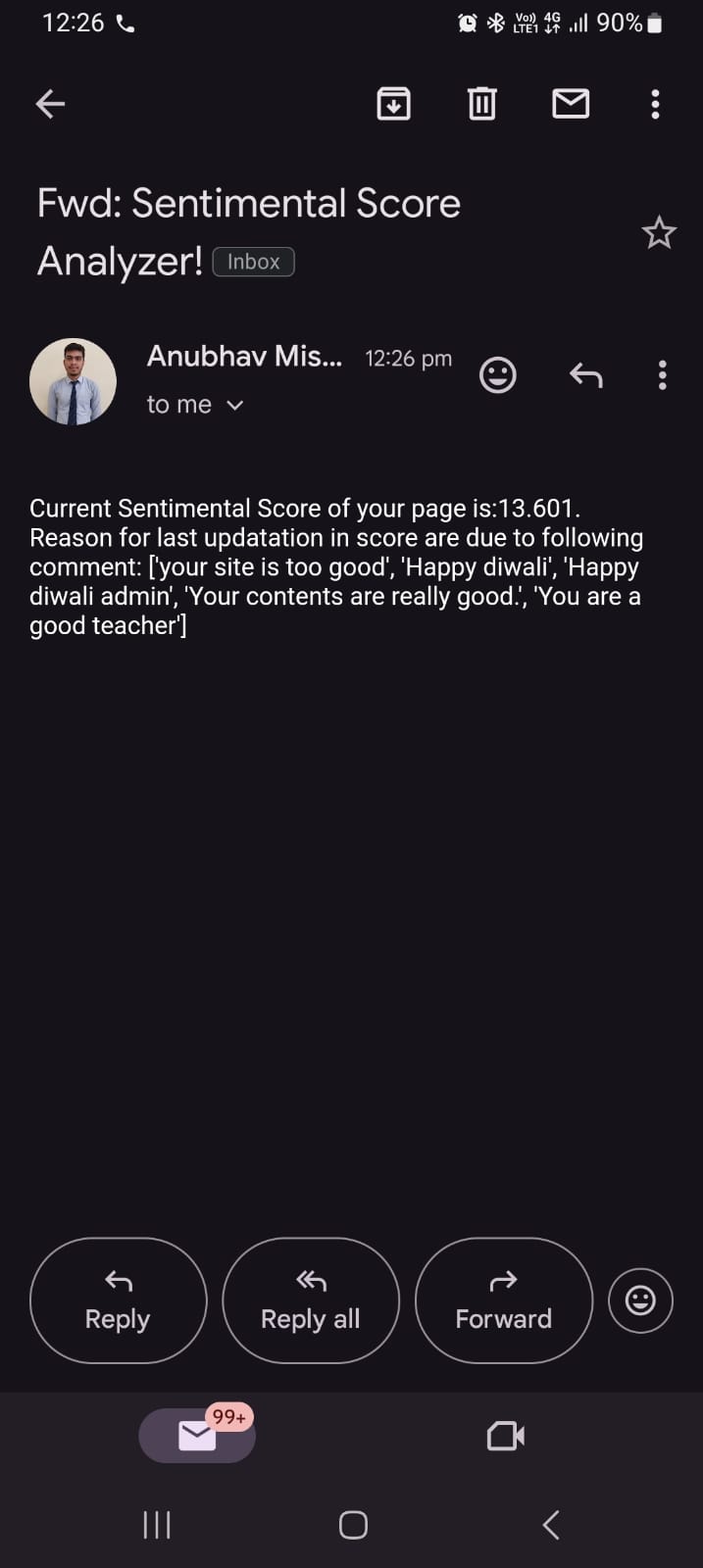
* fb\_comment.csv



* Sentimental Score for each comment



* Mail sent to Configured email-id along with reason and updated score



Limitations

Here, whatever comments I had demonstrated are from my own brand page (Programming World) because of meta latest service policy which states that you cannot fetch comments on bulk from someone’s else Facebook page until and unless you are admin of that page.

Future Scope

* Soon, going to deploy this model on a window-based application (GUI) with the help of core java and NetBeans IDE.
* And, along with that I will made this possible for almost every social media platform of that brand and that is not limited to comments only; the model will also take followers, views, articles, ratings everything into accountability for calculating sentimental score.

References

* <https://www.datacamp.com/tutorial/text-analytics-beginners-nltk>
* <https://towardsdatascience.com/social-media-sentiment-analysis-49b395771197>

Attachments

* Meta for Developers: <https://developers.facebook.com/tools/explorer/>
* Brand Page (Programming World): <https://www.facebook.com/people/Programming-world/61553337455458/>
* GitHub Repo: <https://github.com/meAnubhav/SentimentalAnalyzer_Telaverge>

Conclusion

In conclusion, the Sentimental Analyzer project has proven to be a powerful tool in deciphering and understanding human emotions expressed through text. Through meticulous analysis of language patterns, the system successfully identifies sentiments, providing valuable insights into the emotional undertones of the text data.

Looking ahead, the Sentimental Analyzer project lays the groundwork for further innovation in the realm of sentiment analysis. Continuous refinement, adaptation to evolving language nuances, and expansion of the training dataset will be crucial for ensuring the system's effectiveness in real-world applications. Ultimately, the project serves as a testament to the transformative potential of technology in decoding the intricate language of human emotions.

* - Thank You