Code Listings

1. Code to Fetch and Store Comments (CommentStorageTest.py)

```
#code adapted from GeeksforGeeks, 2023


from googleapiclient.discovery import build


API_KEY = 'AIzaSyB2K7fsqS846BU61_9ENVIAquhqyFL-oK0'
youtube = build('youtube', 'v3', developerKey=API_KEY)


# Taking input from the user and slicing for video id
video_id = input('Enter Youtube Video URL: ')[-11:]
print("video id: " + video_id)


# Getting the channelId of the video uploader
video_response = youtube.videos().list(
    part='snippet',
    id=video_id
).execute()
print (video_response)


# Splitting the response for channelID
video_snippet = video_response['items'][0]['snippet']
uploader_channel_id = video_snippet['channelId']
print("channel id: " + uploader_channel_id)
```

```
# Fetch comments
print("Fetching Comments...")
comments = []
nextPageToken = None
while len(comments) < 75:
    print (len(comments))
    request = youtube.commentThreads().list(
        part='snippet',
        videoId=video_id,
        maxResults=75,  # You can fetch up to 100 comments per request
        pageToken=nextPageToken
    )
    response = request.execute()
    for item in response['items']:
        comment = item['snippet']['topLevelComment']['snippet']
        # Check if the comment is not from the video uploader
        if comment['authorChannelId']['value'] != uploader_channel_id:
            comments.append(comment['textDisplay'])
    nextPageToken = response.get('nextPageToken')

    if not nextPageToken:
```

```
            break


    # Printing comments
    print(comments[:75])



    # Storing YouTube Comments


    f = open(".venv/Euphoria-AlgeeSmithComments7.txt", 'w',
    encoding='utf-8')
    for idx, comment in enumerate(comments):
        f.write(str(comment)+"\n")
    f.close()
    print("Comments stored successfully!")


    #end of adapted code
```

2. Code for VADER (VADERSentiment.py)

```
from vaderSentiment.vaderSentiment import
SentimentIntensityAnalyzer


#Code adapted from GeeksforGeeks, 2023
def sentiment_scores(comment, polarity):
    # Creating a SentimentIntensityAnalyzer object.
```

```
    sentiment_object = SentimentIntensityAnalyzer()

    sentiment_dict = sentiment_object.polarity_scores(comment)
    polarity.append(sentiment_dict['compound'])

    return polarity

polarity = []
positive_comments = []
negative_comments = []
neutral_comments = []


f = open("Euphoria-AlgeeSmithComments20.txt", 'r',
encoding='`utf-8')
comments = f.readlines()
f.close()
print("Analysing Comments...")
for index, items in enumerate(comments):
    polarity = sentiment_scores(items, polarity)

    if polarity[-1] > 0.05:
        positive_comments.append(items)
    elif polarity[-1] < -0.05:
        negative_comments.append(items)
```

```
    else:
        neutral_comments.append(items)

# Print polarity
#print the sentiments for all of the comments in the .txt file
print(polarity[:75])

#overall polarity

avg_polarity = sum(polarity) / len(polarity)
print("Average Polarity:", avg_polarity)
if avg_polarity > 0.05:
    print("The Video has got a Positive response")
elif avg_polarity < -0.05:
    print("The Video has got a Negative response")
else:
    print("The Video has got a Neutral response")

print("The comment with most positive sentiment:",
comments[polarity.index(max(
    polarity))], "with score", max(polarity), "and length",
len(comments[polarity.index(max(polarity))]))
print("The comment with most negative sentiment:",
comments[polarity.index(min(
```

```
    polarity))], "with score", min(polarity), "and length",
len(comments[polarity.index(min(polarity))]))


#end of adapted code
```

3. Code for TextBlob (TEXTBLOBSentimentAnalysisFinal.py)

```
from textblob import TextBlob


#code adapted from Pranayteja, 2023


#define textblob object
def get_sentiment_scores(blob):
    #sentiment = [] - was doing this because polarity was
defined in VADER
    positive_scores = []
    negative_scores = []
    neutral_scores = []


    sentiment_score = blob.sentiment


    if sentiment_score[-1] > 0:
            positive_scores.append(sentiment_score)
            print("The comment has got a Positive response")
    elif sentiment_score[-1] < 0:
            negative_scores.append(sentiment_score)
```

```
            print("The comment has got a Negative response")
    else:
            neutral_scores.append(sentiment_score)
            print("The comment has got a Neutral response")

    #return positive_scores, negative_scores, neutral_scores
    return sentiment_score

#end of adapted code

polarity = []

#code adapted from Whalen, 2020
#code to open and read .txt files with comments inside


with open("Euphoria-ZendayaComments1.txt", encoding='`utf-8') as
f:
            comment = f.readlines()

#end of adapted code

# print the sentiments for all the comments in the .txt file

            print("Analysing Comments...")
```

```
            for index, items in enumerate(comment):
                blob = TextBlob(items)
                polarity = get_sentiment_scores(blob)
                print(polarity[-1])

#code adapted from GeeksforGeeks, 2023
#How to get the overall polarity of the comments in the .txt
file

avg_polarity = sum(polarity) / len(polarity)
print("Average Polarity:", avg_polarity)
if avg_polarity > 0.05:
    print("The Video has got a Positive response")
elif avg_polarity < -0.05:
    print("The Video has got a Negative response")
else:
    print("The Video has got a Neutral response")

#end of adapted code
```

4. Code for Instruct ChatGPT model (CTest3.py)

```
#code adapted from Matthews, 2023
import openai
import time
import pandas as pd
```

```
#Set OpenAI API Key
openai.api_key = "sk-proj-FENSZDQ9JLR2eUbZy-kd6mE1UPUo-
zTda583svfj4aOjVT-FBrxiEvdCACT3BlbkFJVroeZJKbqcQ91J9J6LW-
zz2Epo0ds6r1Z8HYPQyI6NC-9h0IiQJHKqp4gA"
# Read the file contents into a list of rows

#end of adapted code

#open file in read mode
with open("Euphoria-AlgeeSmithComments20.txt", encoding='`utf-
8') as f:
    comment = f.readlines()
print("Analysing Comments...")
for index, items in enumerate(comment):
 print(items)
prompt = f"Please analyze the sentiment of the following
text:{items}"

#code adapted from Matthews, 2023

# Call the sentiment analysis API with the prompt
response = openai.Completion.create(
    engine="text-davinci-002",
    prompt=prompt,
```

```
    temperature=0,
    max_tokens=128,  # Increase max_tokens to retrieve more than
one token
    n=1,
    stop=None,
    timeout=10,
    )
# Extract the sentiment from the API response
sentiment = response.choices[0].text.strip().replace("The
sentiment of the text is ", "").rstrip('.')
# Map the sentiment to a more concise label
if "Positive" in sentiment:
    sentiment = "Positive"
elif "Negative" in sentiment:
    sentiment = "Negative"
elif "Neutral" in sentiment:
        sentiment = "Neutral"

# Print the text and its corresponding sentiment
print(f"{items} -> {sentiment}")

# Pause for 0.5 seconds to avoid hitting API rate limits
time.sleep(0.5)

#end of adapted code
```

5. Code for Evaluation Metrics (EvaluationMetrics.py)

```
#code adapted from GeeksforGeeks, 2023



from sklearn.metrics import accuracy_score, precision_score,
recall_score, f1_score


# True labels (example from code-not sure what to put here)
y_true = [1, 1, 1, 1, 1, 1, 1, 0, 0, 1, 1, 1, 1, 1, 1, 1, 1, 1,
1, 1]
# Predicted labels
y_pred = [0, 1, 0, 1, 0, 1, -1, 0, 0, 0, 0, 0, 0, 1, 0, 1, 0, 1,
1, 0]


# Accuracy
accuracy = accuracy_score(y_true, y_pred)
# Precision
precision = precision_score(y_true, y_pred, average="macro")
# Recall
recall = recall_score(y_true, y_pred, average="macro")
# F1-Score
f1 = f1_score(y_true, y_pred, average="macro")


print("Accuracy:", accuracy)
```

```
print("Precision:", precision)

print("Recall:", recall)

print("F1-Score:", f1)


#end of adapted code
```

References

Classification Metrics Using Sklearn. (2023, October 16). GeeksforGeeks. https://www.geeksforgeeks.org/sklearn-classification-metrics/

Matthews, J. (2023, April 12). Performing Sentiment Analysis on Text Data Using OpenAI API in Python. *Medium*. https://medium.com/@jacobcmatthews/performing-sentiment-analysis-on-text-data-using-openai-api-in-python-4a37816d11df

Pranayteja. (2023, September 7). YouTube Comments Sentiment Analysis Using YouTube Data API v3. *Medium*. https://medium.com/@pranayteja270/youtube-comments-sentiment-analysis-using-youtube-data-api-v3-bf4a2a041144

Sentiment Analysis of YouTube Comments. (2023, October 25). GeeksforGeeks. https://www.geeksforgeeks.org/sentiment-analysis-of-youtube-comments/

Whalen, Z. (2020). *Getting Started with TextBlob*. YouTube. https://www.youtube.com/watch?v=zoxRhs1Tnuo