Configuration Manual

MSc Research Project
Masters of Science in Data Analytics
(MSCDAD_A_JAN24I)

Sayali Jadhav
Student ID: x23201665

School of Computing
National College of Ireland

Supervisor: Prof. Vladimir Milosavljevic

National College of Ireland

MSc Project Submission Sheet

School of Computing

Student Name:	Sayali Sunil Jadhav
Student ID:	x23201665
Programme:	Masters of Science in Data Analytics (MSCDAD_A_JAN24I) **Year:** 2024-2025
Module:	MSc Research Project
Lecturer:	Prof. Vladimir Milosavljevic
Submission Due Date:	12/12/2024
Project Title:	Sentiment Analysis of User Comments for a YouTube Educational Videos
Word Count:	845 **Page Count:** 9

Signature:	Sayali Sunil Jadhav
Date:	12/12/24

PLEASE READ THE FOLLOWING INSTRUCTIONS AND CHECKLIST

Attach a completed copy of this sheet to each project (including multiple copies)	☐
Attach a Moodle submission receipt of the online project submission, to each project (including multiple copies).	☐
You must ensure that you retain a HARD COPY of the project, both for your own reference and in case a project is lost or mislaid. It is not sufficient to keep a copy on computer.	☐

Assignments that are submitted to the Programme Coordinator Office must be placed into the assignment box located outside the office.

Office Use Only	
Signature:	
Date:	
Penalty Applied (if applicable):	

Configuration Manual

Sayali Jadhav
Student ID: x23201665

1 Introduction

This manual setup outlines the steps like specifications, tools required and steps required to configure the code. The manual helps achieving intended findings. It includes procedures and snippets of code for model implementation, testing and assessment. Machine learning model is used to analyse sentiments of the comments from YouTube e-video.

2 System Configuration

This research work is carried out on the computer system that meets the minimum requirements for tools and software as mentioned below:

Hardware Configuration:

. Operating System: Windows 11
. RAM: 16 GB
. Processor: AMD Ryzen 7 5700U with Radeon Graphics
. Hard Disk: 500 GB

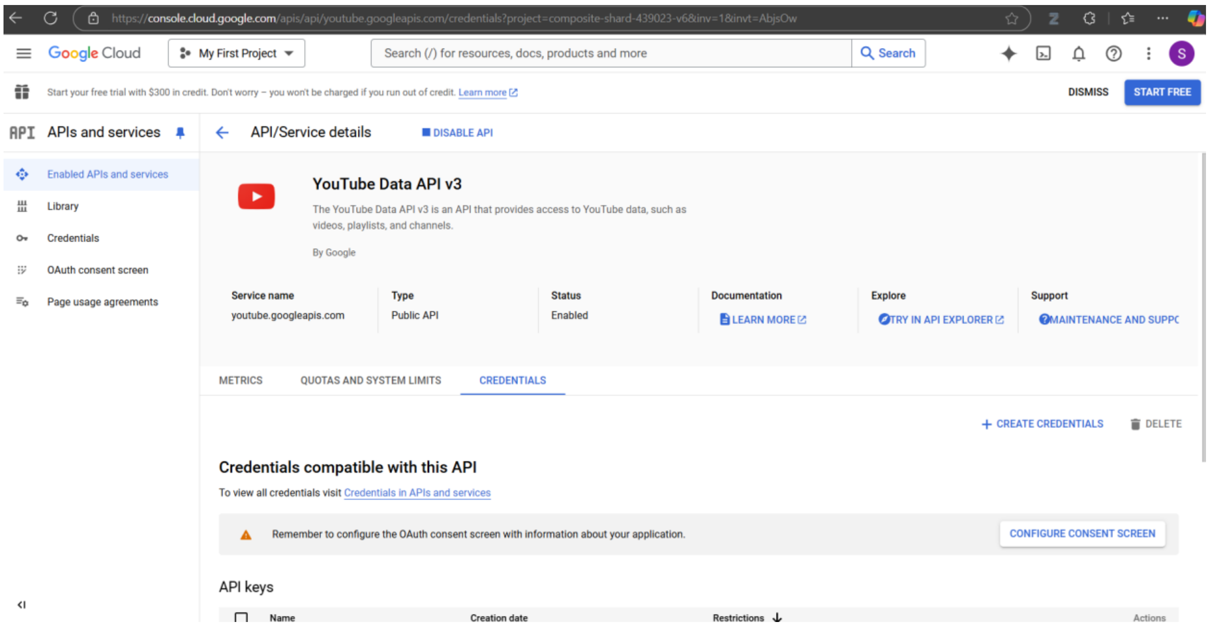
Software Configuration:

The software tools to implement this research project are as follows:
. Python
. Jupyter Notebook

3 Project Development

Development Stage:

- The first step is to finalize the video for retrieving comments from YouTube platform.
- Open Google Cloud by signing in your google account and create a folder and extract the YouTube API key from API Services on Google Cloud.
- Open Jupyter through Anaconda GUI.
- It redirects to Python installation done and then by using the API key and YouTube video ID; extract the comments.



Start your free trial with $300 in credit. Don't worry – you won't be charged if you run out of credit. Learn more

YouTube Data API v3

The YouTube Data API v3 is an API that provides access to YouTube data, such as videos, playlists, and channels.

By Google

Service name	Type	Status	Documentation	Explore	Support
youtube.googleapis.com	Public API	Enabled	LEARN MORE	TRY IN API EXPLORER	MAINTENANCE AND SUPPC

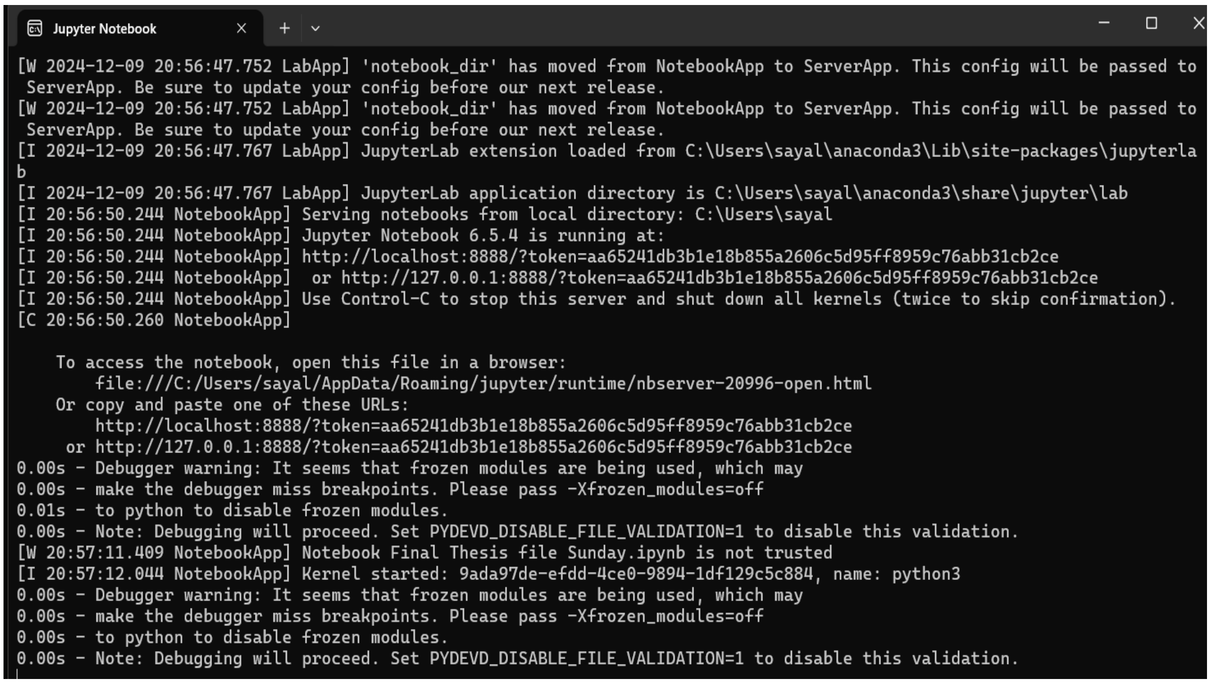
Credentials compatible with this API

To view all credentials visit Credentials in APIs and services

Remember to configure the OAuth consent screen with information about your application.

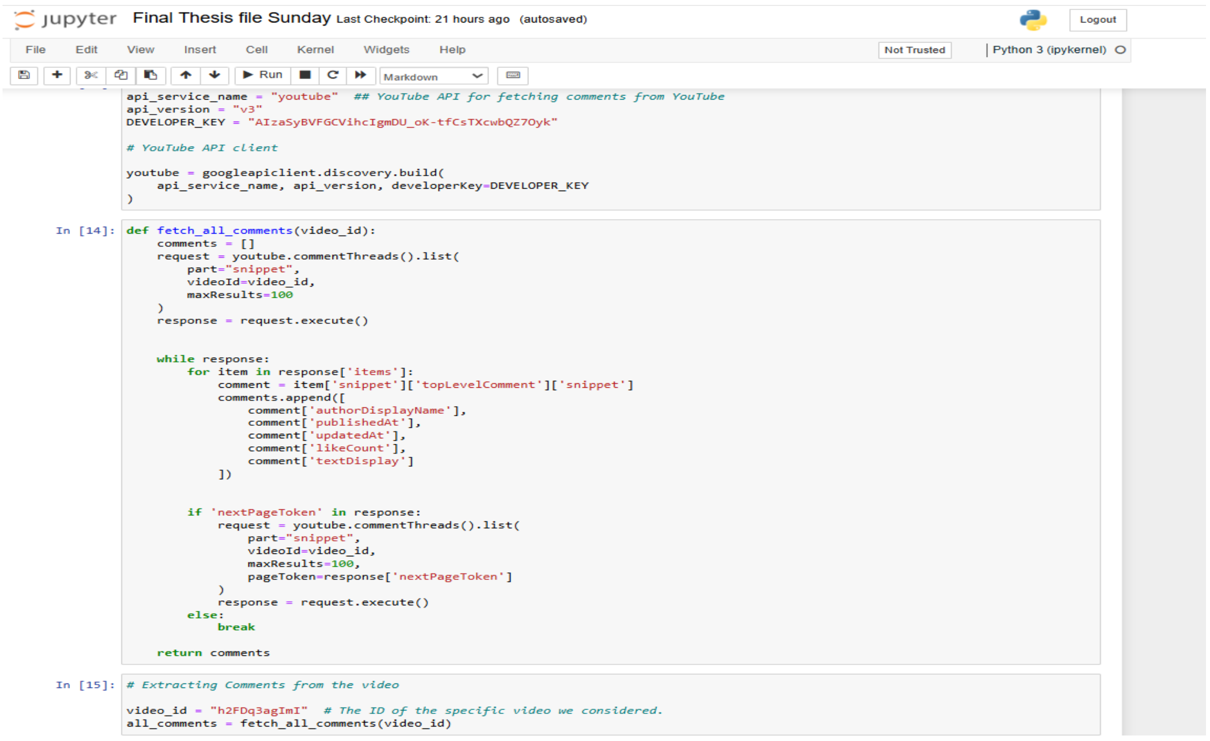
API keys

Fig.1. Google Cloud YouTube API key access



```
[W 2024-12-09 20:56:47.752 LabApp] 'notebook_dir' has moved from NotebookApp to ServerApp. This config will be passed to
 ServerApp. Be sure to update your config before our next release.
[W 2024-12-09 20:56:47.752 LabApp] 'notebook_dir' has moved from NotebookApp to ServerApp. This config will be passed to
 ServerApp. Be sure to update your config before our next release.
[I 2024-12-09 20:56:47.767 LabApp] JupyterLab extension loaded from C:\Users\sayal\anaconda3\Lib\site-packages\jupyterla
b
[I 2024-12-09 20:56:47.767 LabApp] JupyterLab application directory is C:\Users\sayal\anaconda3\share\jupyter\lab
[I 20:56:50.244 NotebookApp] Serving notebooks from local directory: C:\Users\sayal
[I 20:56:50.244 NotebookApp] Jupyter Notebook 6.5.4 is running at:
[I 20:56:50.244 NotebookApp] http://localhost:8888/?token=aa65241db3b1e18b855a2606c5d95ff8959c76abb31cb2ce
[I 20:56:50.244 NotebookApp]  or http://127.0.0.1:8888/?token=aa65241db3b1e18b855a2606c5d95ff8959c76abb31cb2ce
[I 20:56:50.244 NotebookApp] Use Control-C to stop this server and shut down all kernels (twice to skip confirmation).
[C 20:56:50.260 NotebookApp]

    To access the notebook, open this file in a browser:
        file:///C:/Users/sayal/AppData/Roaming/jupyter/runtime/nbserver-20996-open.html
    Or copy and paste one of these URLs:
        http://localhost:8888/?token=aa65241db3b1e18b855a2606c5d95ff8959c76abb31cb2ce
     or http://127.0.0.1:8888/?token=aa65241db3b1e18b855a2606c5d95ff8959c76abb31cb2ce
0.00s - Debugger warning: It seems that frozen modules are being used, which may
0.00s - make the debugger miss breakpoints. Please pass -Xfrozen_modules=off
0.01s - to python to disable frozen modules.
0.00s - Note: Debugging will proceed. Set PYDEVD_DISABLE_FILE_VALIDATION=1 to disable this validation.
[W 20:57:11.409 NotebookApp] Notebook Final Thesis file Sunday.ipynb is not trusted
[I 20:57:12.044 NotebookApp] Kernel started: 9ada97de-efdd-4ce0-9894-1df129c5c884, name: python3
0.00s - Debugger warning: It seems that frozen modules are being used, which may
0.00s - make the debugger miss breakpoints. Please pass -Xfrozen_modules=off
0.00s - to python to disable frozen modules.
0.00s - Note: Debugging will proceed. Set PYDEVD_DISABLE_FILE_VALIDATION=1 to disable this validation.
```

Fig.2. Jupyter Notebook Console



```
api_service_name = "youtube"  ## YouTube API for fetching comments from YouTube
api_version = "v3"
DEVELOPER_KEY = "AIzaSyBVFGCVihcIgmDU_oK-tfCsTXcwbQZ7Oyk"

# YouTube API client

youtube = googleapiclient.discovery.build(
    api_service_name, api_version, developerKey=DEVELOPER_KEY
)
```

```
In [14]: def fetch_all_comments(video_id):
             comments = []
             request = youtube.commentThreads().list(
                 part="snippet",
                 videoId=video_id,
                 maxResults=100
             )
             response = request.execute()


             while response:
                 for item in response['items']:
                     comment = item['snippet']['topLevelComment']['snippet']
                     comments.append([
                         comment['authorDisplayName'],
                         comment['publishedAt'],
                         comment['updatedAt'],
                         comment['likeCount'],
                         comment['textDisplay']
                     ])


                 if 'nextPageToken' in response:
                     request = youtube.commentThreads().list(
                         part="snippet",
                         videoId=video_id,
                         maxResults=100,
                         pageToken=response['nextPageToken']
                     )
                     response = request.execute()
                 else:
                     break

             return comments
```

```
In [15]: # Extracting Comments from the video

         video_id = "h2FDq3agImI"  # The ID of the specific video we considered.
         all_comments = fetch_all_comments(video_id)
```

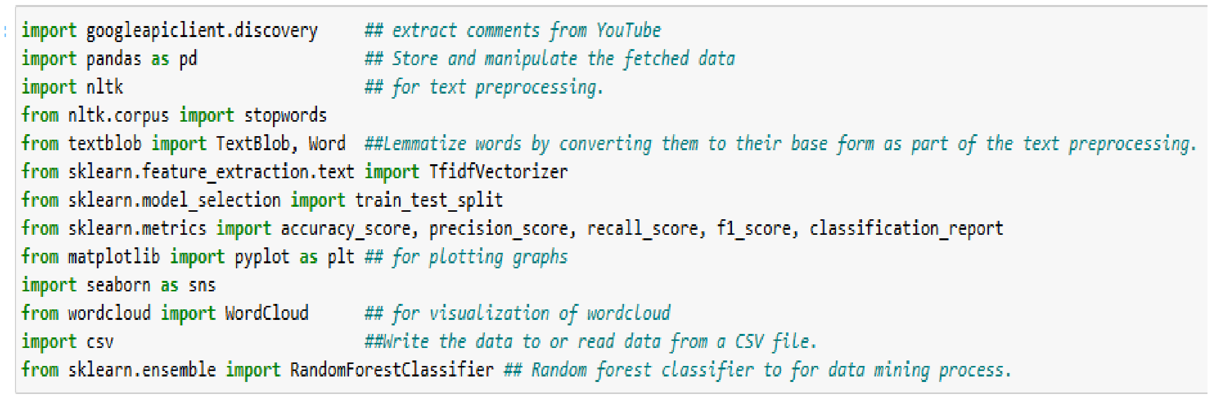
Fig.3. Python shell with API key and Video ID

4 Implementation

Import all the necessary python libraries that are used for implementing the project, they are as follows:

. Scikit-Learn
. Nltk
. Pandas
. Googleapiclient.discovery
. Textblob
. Matplotlib
. Seaborn
. WordCloud
. Csv

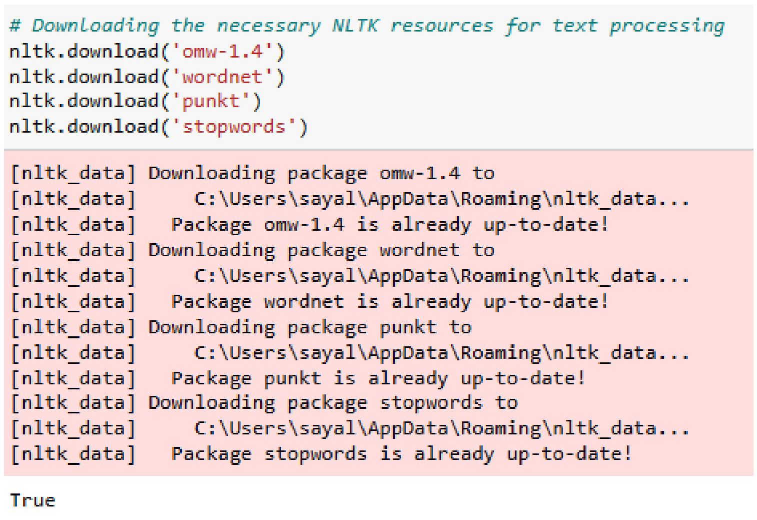
1. Importing Libraries



```
import googleapiclient.discovery     ## extract comments from YouTube
import pandas as pd                  ## Store and manipulate the fetched data
import nltk                          ## for text preprocessing.
from nltk.corpus import stopwords
from textblob import TextBlob, Word  ##Lemmatize words by converting them to their base form as part of the text preprocessing.
from sklearn.feature_extraction.text import TfidfVectorizer
from sklearn.model_selection import train_test_split
from sklearn.metrics import accuracy_score, precision_score, recall_score, f1_score, classification_report
from matplotlib import pyplot as plt ## for plotting graphs
import seaborn as sns
from wordcloud import WordCloud      ## for visualization of wordcloud
import csv                           ##Write the data to or read data from a CSV file.
from sklearn.ensemble import RandomForestClassifier ## Random forest classifier to for data mining process.
```

Fig.4. Libraries imported for research

Some of the text preprocessing NLTK packages are as below Figure 5.

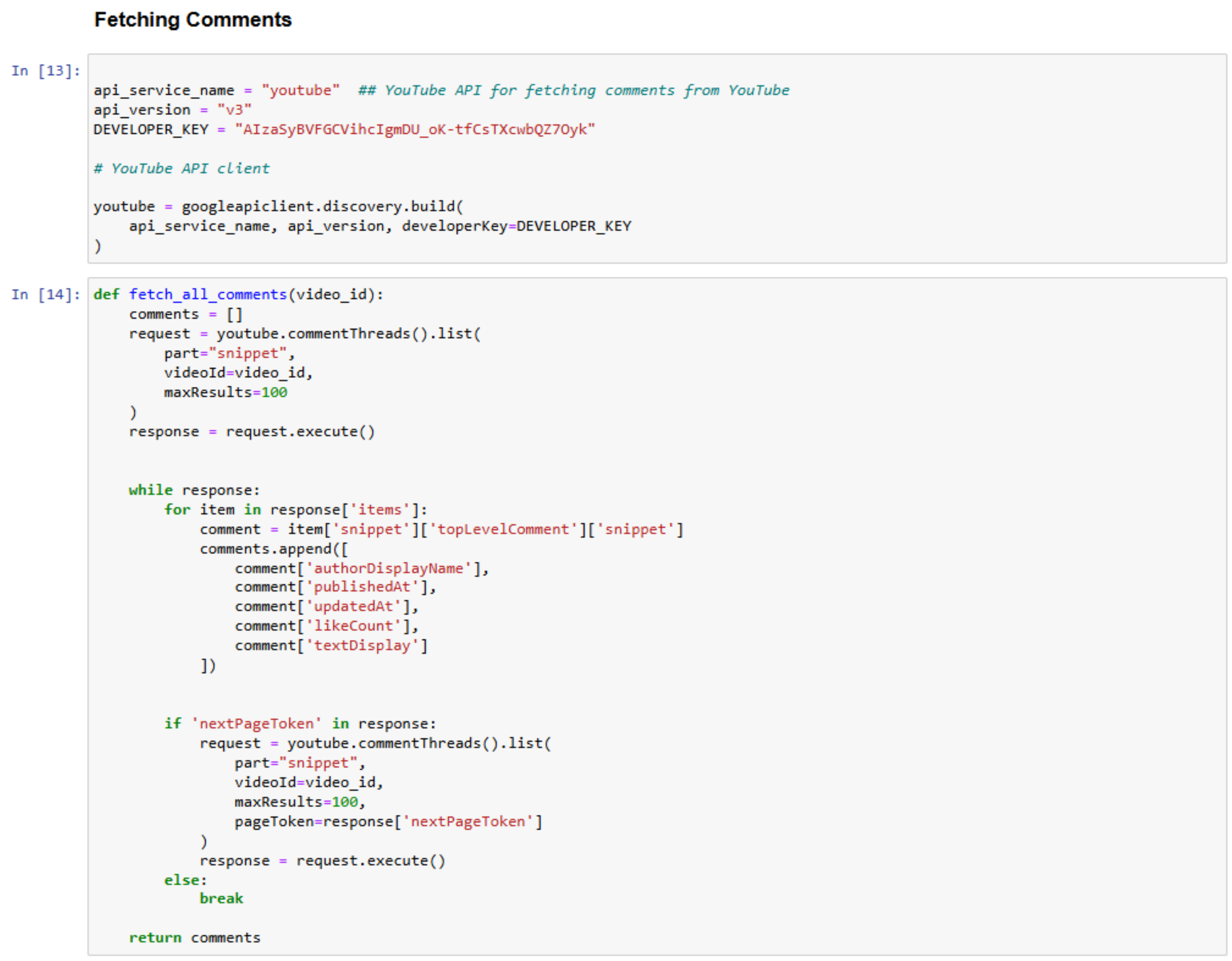


```
# Downloading the necessary NLTK resources for text processing
nltk.download('omw-1.4')
nltk.download('wordnet')
nltk.download('punkt')
nltk.download('stopwords')
```

```
[nltk_data] Downloading package omw-1.4 to
[nltk_data]     C:\Users\sayal\AppData\Roaming\nltk_data...
[nltk_data]   Package omw-1.4 is already up-to-date!
[nltk_data] Downloading package wordnet to
[nltk_data]     C:\Users\sayal\AppData\Roaming\nltk_data...
[nltk_data]   Package wordnet is already up-to-date!
[nltk_data] Downloading package punkt to
[nltk_data]     C:\Users\sayal\AppData\Roaming\nltk_data...
[nltk_data]   Package punkt is already up-to-date!
[nltk_data] Downloading package stopwords to
[nltk_data]     C:\Users\sayal\AppData\Roaming\nltk_data...
[nltk_data]   Package stopwords is already up-to-date!

True
```

Fig.5. NLTK resources for pre-processing

The next steps involved fetching comments from YouTube video by using API key and YouTube Video-ID. Figure 6 and 7 demonstrates the same:

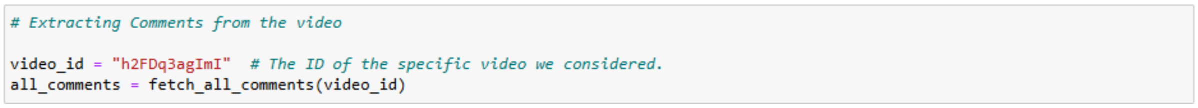


Fetching Comments

```
In [13]:
api_service_name = "youtube"   ## YouTube API for fetching comments from YouTube
api_version = "v3"
DEVELOPER_KEY = "AIzaSyBVFGCVihcIgmDU_oK-tfCsTXcwbQZ7Oyk"

# YouTube API client

youtube = googleapiclient.discovery.build(
    api_service_name, api_version, developerKey=DEVELOPER_KEY
)
```

```
In [14]:
def fetch_all_comments(video_id):
    comments = []
    request = youtube.commentThreads().list(
        part="snippet",
        videoId=video_id,
        maxResults=100
    )
    response = request.execute()


    while response:
        for item in response['items']:
            comment = item['snippet']['topLevelComment']['snippet']
            comments.append([
                comment['authorDisplayName'],
                comment['publishedAt'],
                comment['updatedAt'],
                comment['likeCount'],
                comment['textDisplay']
            ])


        if 'nextPageToken' in response:
            request = youtube.commentThreads().list(
                part="snippet",
                videoId=video_id,
                maxResults=100,
                pageToken=response['nextPageToken']
            )
            response = request.execute()
        else:
            break

    return comments
```

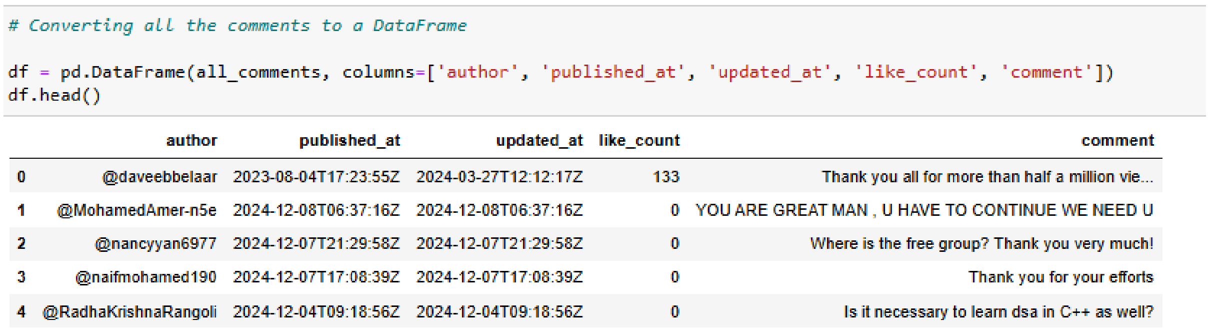
Fig.6. Fetching comments from YouTube video



```
# Extracting Comments from the video

video_id = "h2FDq3agImI"  # The ID of the specific video we considered.
all_comments = fetch_all_comments(video_id)
```

Fig.7. Extraction of Comments from desired video with Video-ID

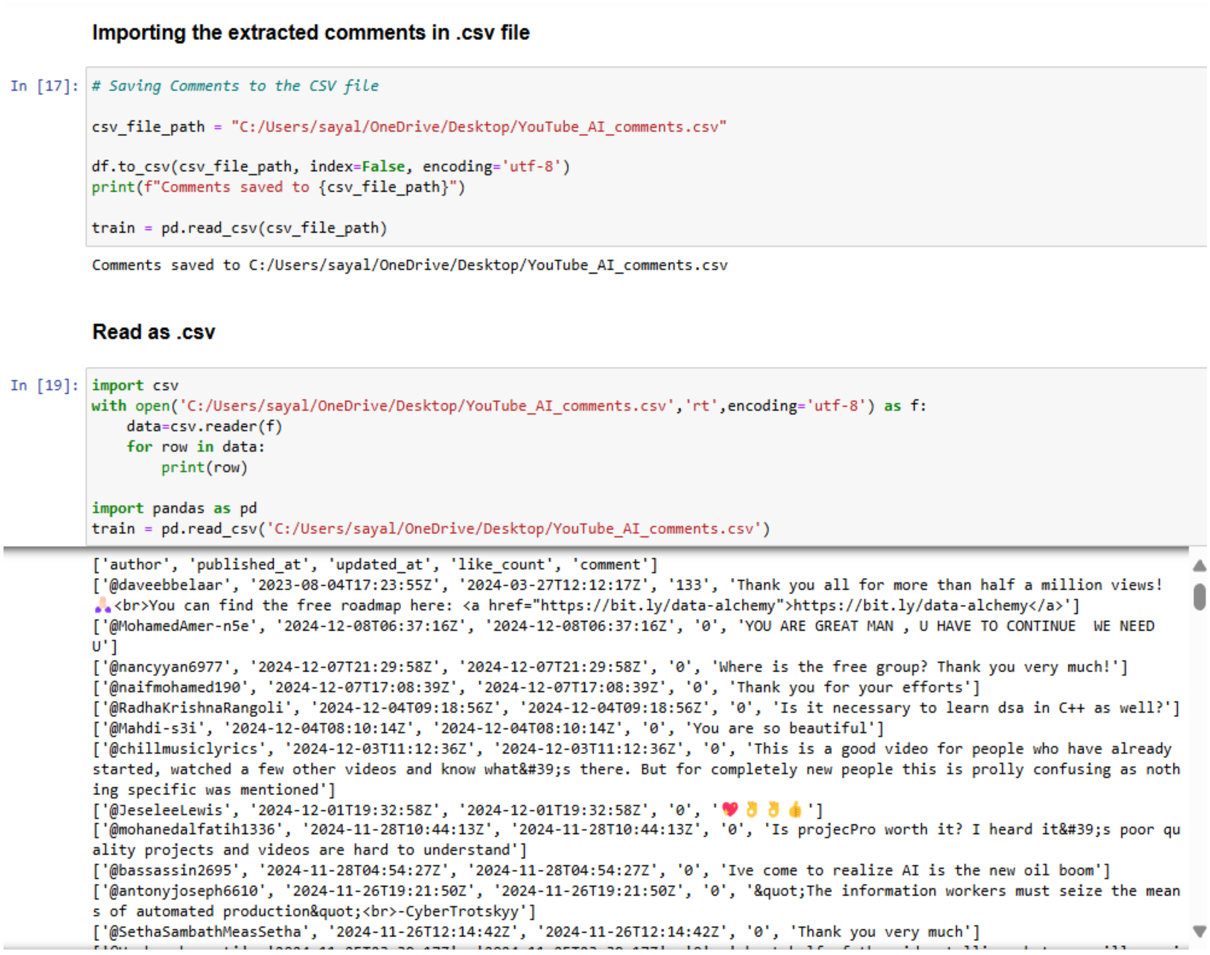
The below Figure 8 and 9 shows how the Data-Frame is created for the comments extracted and how it is saved in form of .csv file.



```
# Converting all the comments to a DataFrame

df = pd.DataFrame(all_comments, columns=['author', 'published_at', 'updated_at', 'like_count', 'comment'])
df.head()
```

	author	published_at	updated_at	like_count	comment
0	@daveebbelaar	2023-08-04T17:23:55Z	2024-03-27T12:12:17Z	133	Thank you all for more than half a million vie...
1	@MohamedAmer-n5e	2024-12-08T06:37:16Z	2024-12-08T06:37:16Z	0	YOU ARE GREAT MAN , U HAVE TO CONTINUE WE NEED U
2	@nancyyan6977	2024-12-07T21:29:58Z	2024-12-07T21:29:58Z	0	Where is the free group? Thank you very much!
3	@naifmohamed190	2024-12-07T17:08:39Z	2024-12-07T17:08:39Z	0	Thank you for your efforts
4	@RadhaKrishnaRangoli	2024-12-04T09:18:56Z	2024-12-04T09:18:56Z	0	Is it necessary to learn dsa in C++ as well?

Fig.8. Comments to a DataFrame



Importing the extracted comments in .csv file

```
In [17]: # Saving Comments to the CSV file

         csv_file_path = "C:/Users/sayal/OneDrive/Desktop/YouTube_AI_comments.csv"

         df.to_csv(csv_file_path, index=False, encoding='utf-8')
         print(f"Comments saved to {csv_file_path}")

         train = pd.read_csv(csv_file_path)

         Comments saved to C:/Users/sayal/OneDrive/Desktop/YouTube_AI_comments.csv
```

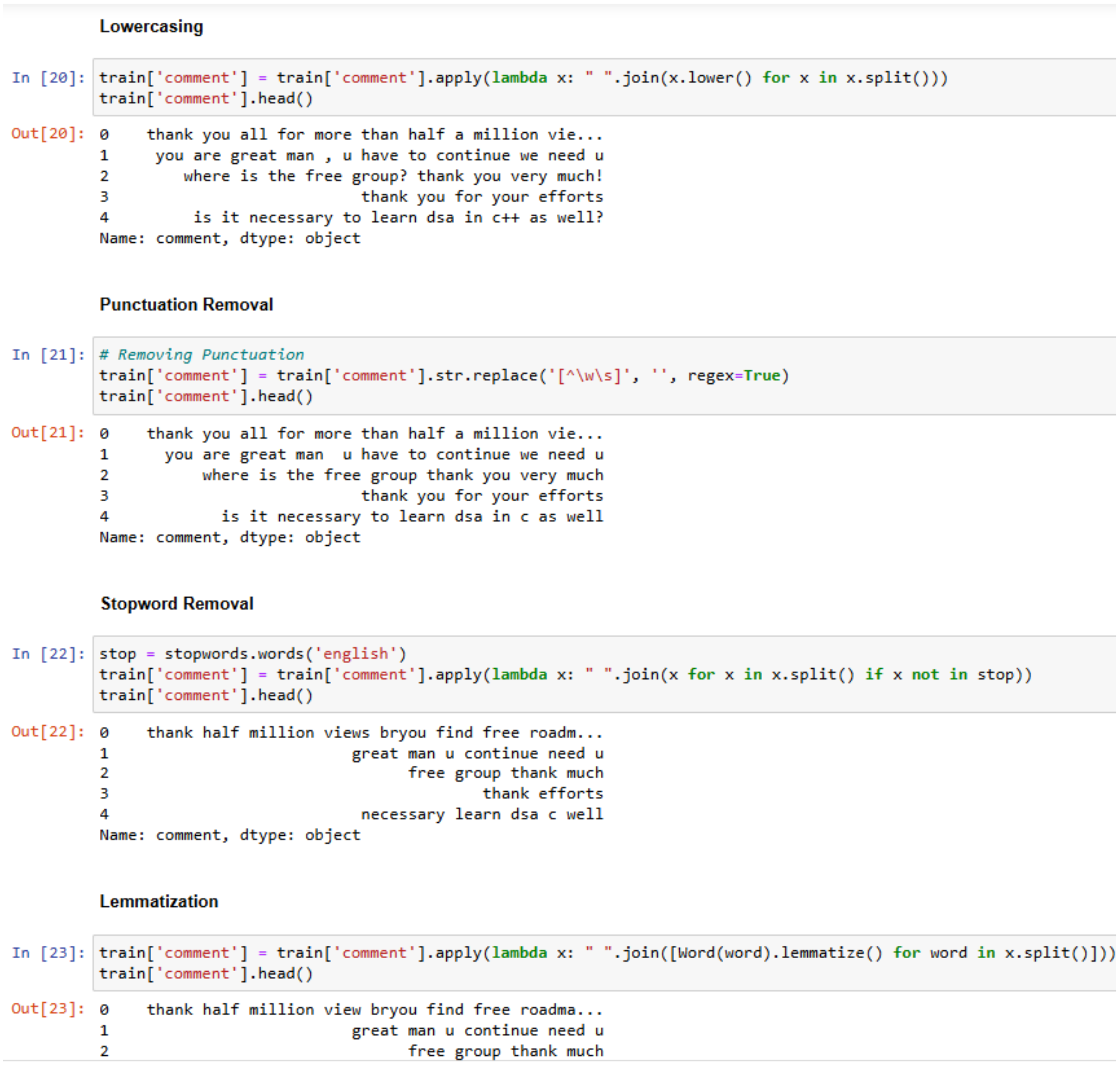
Read as .csv

```
In [19]: import csv
         with open('C:/Users/sayal/OneDrive/Desktop/YouTube_AI_comments.csv','rt',encoding='utf-8') as f:
             data=csv.reader(f)
             for row in data:
                 print(row)

         import pandas as pd
         train = pd.read_csv('C:/Users/sayal/OneDrive/Desktop/YouTube_AI_comments.csv')
```

```
['author', 'published_at', 'updated_at', 'like_count', 'comment']
['@daveebbelaar', '2023-08-04T17:23:55Z', '2024-03-27T12:12:17Z', '133', 'Thank you all for more than half a million views!
🙏<br>You can find the free roadmap here: <a href="https://bit.ly/data-alchemy">https://bit.ly/data-alchemy</a>']
['@MohamedAmer-n5e', '2024-12-08T06:37:16Z', '2024-12-08T06:37:16Z', '0', 'YOU ARE GREAT MAN , U HAVE TO CONTINUE  WE NEED
U']
['@nancyyan6977', '2024-12-07T21:29:58Z', '2024-12-07T21:29:58Z', '0', 'Where is the free group? Thank you very much!']
['@naifmohamed190', '2024-12-07T17:08:39Z', '2024-12-07T17:08:39Z', '0', 'Thank you for your efforts']
['@RadhaKrishnaRangoli', '2024-12-04T09:18:56Z', '2024-12-04T09:18:56Z', '0', 'Is it necessary to learn dsa in C++ as well?']
['@Mahdi-s3i', '2024-12-04T08:10:14Z', '2024-12-04T08:10:14Z', '0', 'You are so beautiful']
['@chillmusiclyrics', '2024-12-03T11:12:36Z', '2024-12-03T11:12:36Z', '0', 'This is a good video for people who have already
started, watched a few other videos and know what&#39;s there. But for completely new people this is prolly confusing as noth
ing specific was mentioned']
['@JeseleeLewis', '2024-12-01T19:32:58Z', '2024-12-01T19:32:58Z', '0', '💖👌👌👍']
['@mohanedalfatih1336', '2024-11-28T10:44:13Z', '2024-11-28T10:44:13Z', '0', 'Is projecPro worth it? I heard it&#39;s poor qu
ality projects and videos are hard to understand']
['@bassassin2695', '2024-11-28T04:54:27Z', '2024-11-28T04:54:27Z', '0', 'Ive come to realize AI is the new oil boom']
['@antonyjoseph6610', '2024-11-26T19:21:50Z', '2024-11-26T19:21:50Z', '0', '&quot;The information workers must seize the mean
s of automated production&quot;<br>-CyberTrotskyy']
['@SethaSambathMeasSetha', '2024-11-26T12:14:42Z', '2024-11-26T12:14:42Z', '0', 'Thank you very much']
```

Fig.9. Import and Read as .csv file

Then to all of these extracted comments stored in .csv file we applied pre-processing techniques. Cleaned and Transformed data necessary for modelling is ready after this step. Figure 10 and 11 illustrates the same.



Lowercasing

```
In [20]: train['comment'] = train['comment'].apply(lambda x: " ".join(x.lower() for x in x.split()))
         train['comment'].head()

Out[20]: 0    thank you all for more than half a million vie...
         1     you are great man , u have to continue we need u
         2        where is the free group? thank you very much!
         3                          thank you for your efforts
         4         is it necessary to learn dsa in c++ as well?
         Name: comment, dtype: object
```

Punctuation Removal

```
In [21]: # Removing Punctuation
         train['comment'] = train['comment'].str.replace('[^\w\s]', '', regex=True)
         train['comment'].head()

Out[21]: 0    thank you all for more than half a million vie...
         1      you are great man  u have to continue we need u
         2          where is the free group thank you very much
         3                          thank you for your efforts
         4            is it necessary to learn dsa in c as well
         Name: comment, dtype: object
```

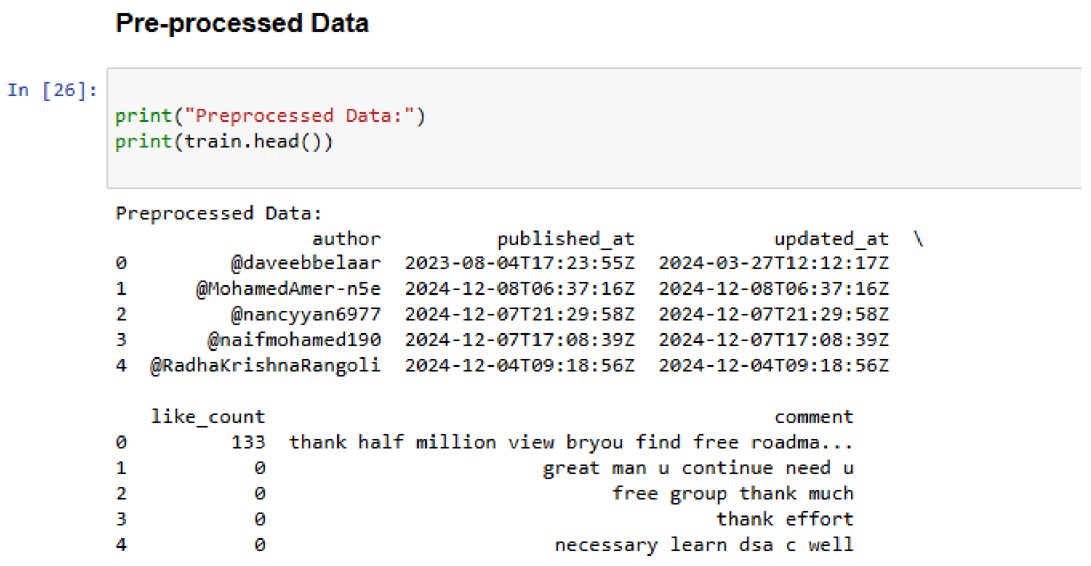
Stopword Removal

```
In [22]: stop = stopwords.words('english')
         train['comment'] = train['comment'].apply(lambda x: " ".join(x for x in x.split() if x not in stop))
         train['comment'].head()

Out[22]: 0    thank half million views bryou find free roadm...
         1                          great man u continue need u
         2                                free group thank much
         3                                        thank efforts
         4                           necessary learn dsa c well
         Name: comment, dtype: object
```

Lemmatization

```
In [23]: train['comment'] = train['comment'].apply(lambda x: " ".join([Word(word).lemmatize() for word in x.split()]))
         train['comment'].head()

Out[23]: 0    thank half million view bryou find free roadma...
         1                          great man u continue need u
         2                                free group thank much
```

Fig.10. Pre-processing steps

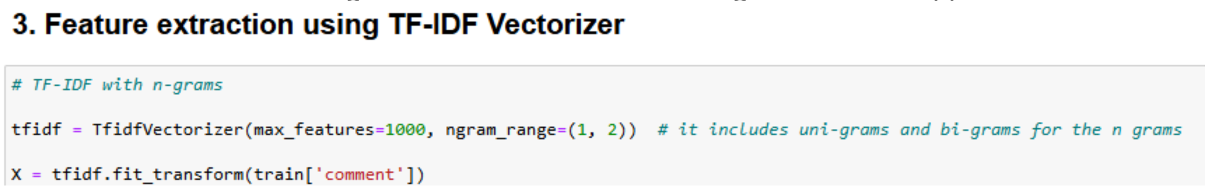

Pre-processed Data

```
In [26]:
print("Preprocessed Data:")
print(train.head())
```

```
Preprocessed Data:
                author           published_at             updated_at  \
0         @daveebbelaar  2023-08-04T17:23:55Z  2024-03-27T12:12:17Z
1      @MohamedAmer-n5e  2024-12-08T06:37:16Z  2024-12-08T06:37:16Z
2         @nancyyan6977  2024-12-07T21:29:58Z  2024-12-07T21:29:58Z
3       @naifmohamed190  2024-12-07T17:08:39Z  2024-12-07T17:08:39Z
4  @RadhaKrishnaRangoli  2024-12-04T09:18:56Z  2024-12-04T09:18:56Z

   like_count                                            comment
0         133  thank half million view bryou find free roadma...
1           0                        great man u continue need u
2           0                              free group thank much
3           0                                       thank effort
4           0                          necessary learn dsa c well
```

Fig.11. Pre-processed Data

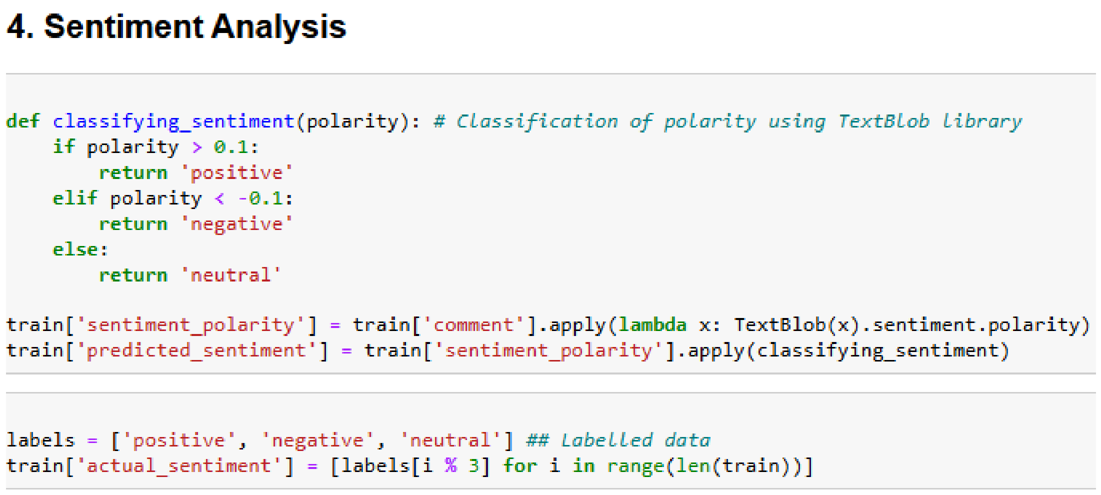
The feature extraction using TF-IDF vector is done through this code snippet as below:


3. Feature extraction using TF-IDF Vectorizer

```
# TF-IDF with n-grams

tfidf = TfidfVectorizer(max_features=1000, ngram_range=(1, 2))  # it includes uni-grams and bi-grams for the n grams

X = tfidf.fit_transform(train['comment'])
```

Fig.12. Feature Extraction using TF-IDF Vectorizer

After all this steps, sentiment analysis of the YouTube comments is done using Textblob and polarity of the sentiments is found out as in the following figure:


4. Sentiment Analysis

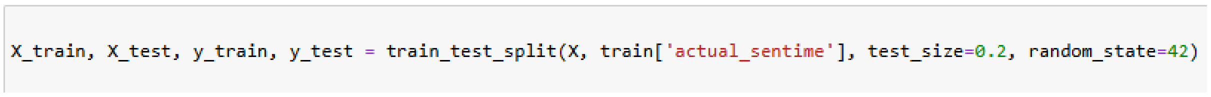
```
def classifying_sentiment(polarity): # Classification of polarity using TextBlob library
    if polarity > 0.1:
        return 'positive'
    elif polarity < -0.1:
        return 'negative'
    else:
        return 'neutral'

train['sentiment_polarity'] = train['comment'].apply(lambda x: TextBlob(x).sentiment.polarity)
train['predicted_sentiment'] = train['sentiment_polarity'].apply(classifying_sentiment)
```

```
labels = ['positive', 'negative', 'neutral'] ## Labelled data
train['actual_sentiment'] = [labels[i % 3] for i in range(len(train))]
```

Fig.13. Sentiment Analysis and Labelling Data

Splitting the pre-processed and labelled data into train and test as shown in Figure 14 below:

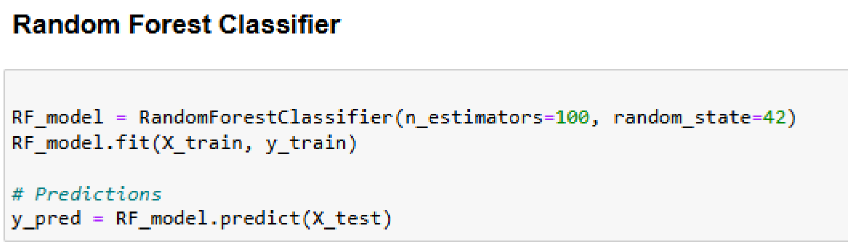
5. Data Mining Model



```
X_train, X_test, y_train, y_test = train_test_split(X, train['actual_sentime'], test_size=0.2, random_state=42)
```

Fig.14. Data Split into Train and Test

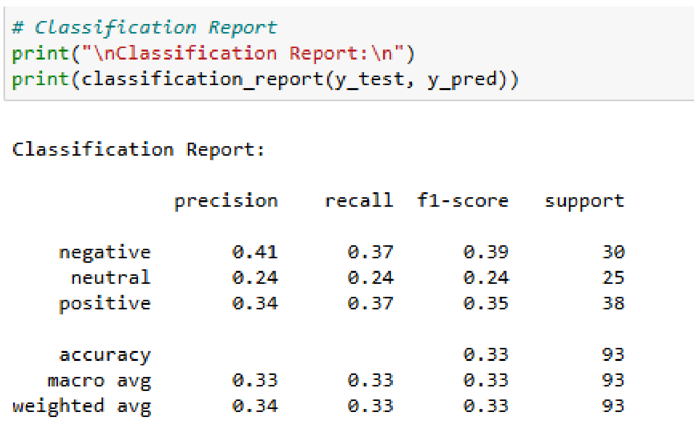
The Random Forest Classifier model is used to model the data as follows:



Random Forest Classifier

```
RF_model = RandomForestClassifier(n_estimators=100, random_state=42)
RF_model.fit(X_train, y_train)

# Predictions
y_pred = RF_model.predict(X_test)
```

Fig.15. Random Forest Model



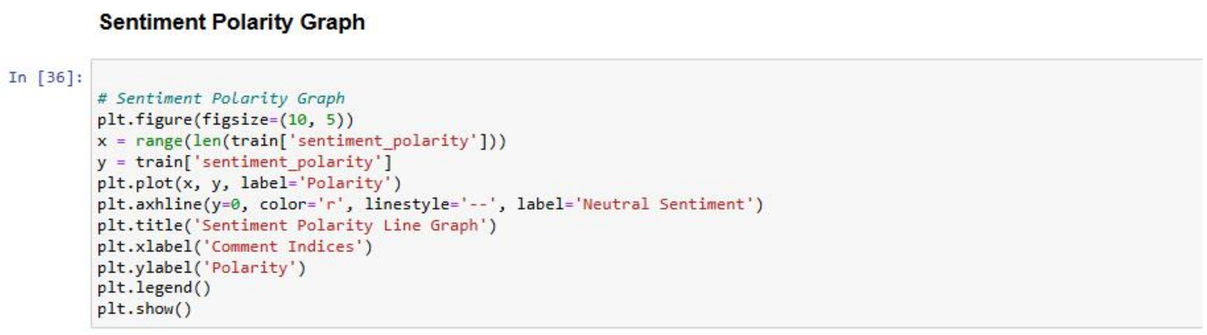
```
# Classification Report
print("\nClassification Report:\n")
print(classification_report(y_test, y_pred))
```

```
Classification Report:

              precision    recall  f1-score   support

    negative       0.41      0.37      0.39        30
     neutral       0.24      0.24      0.24        25
    positive       0.34      0.37      0.35        38

    accuracy                           0.33        93
   macro avg       0.33      0.33      0.33        93
weighted avg       0.34      0.33      0.33        93
```

Fig.16. Classification report with accuracy, F1-Score.

Visualization

The below Figure 17 Polarity graph represents sentiment polarity of comments from -1 to 1 (i.e. negative to positive cases) and some on neutral side marked with red dotted line.



Sentiment Polarity Graph

In [36]:

```
# Sentiment Polarity Graph
plt.figure(figsize=(10, 5))
x = range(len(train['sentiment_polarity']))
y = train['sentiment_polarity']
plt.plot(x, y, label='Polarity')
plt.axhline(y=0, color='r', linestyle='--', label='Neutral Sentiment')
plt.title('Sentiment Polarity Line Graph')
plt.xlabel('Comment Indices')
plt.ylabel('Polarity')
plt.legend()
plt.show()
```

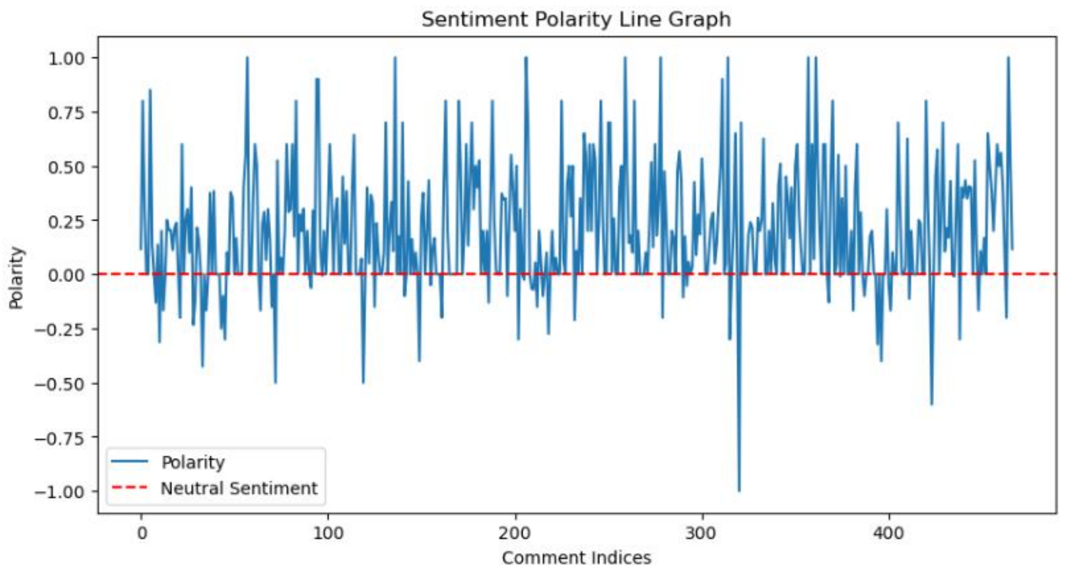


Fig.17. Sentiment Polarity Graph

References

Singh, R. and Tiwari, A., 2021. Youtube comments sentiment analysis. International Journal of Scientific Research in Engineering and Management (IJSREM), 5(5), pp.1-11.