Comment Sentilizer

Project Report submitted in partial fulfillment of
The requirements for the degree of

MASTER OF COMPUTER APPLICATION

Of

Maulana Abul Kalam Azad University of Technology

By

SHUVODEEP TALUKDAR, Roll No: 29171022013

RUMI HAZRA, Roll No: 29171022009

MOHAMMED SOHEB ALAM, Roll No: 29171022015

RANJIT JANA, Roll No: 29171022021

Under the guidance of

Dr. MOUJHURI PATRA

DEPARTMENT OF MASTERS OF COMPUTER APPLICATION

NETAJI SUBHASH ENGINEERING COLLEGE
TECHNO CITY, GARIA, KOLKATA – 700 152

Academic year of pass out 2022-2024

CERTIFICATE

This is to certify that this project report titled Comment Sentilizer submitted in partial fulfillment of requirements for award of the degree Master of Computer Application of West Bengal University of Technology is a faithful record of the original work carried out by,

SHUVODEEP TALUKDAR, Roll No: 29171022013 Reg No:222910510016

RUMI HAZRA, Roll No: 29171022009 Reg No:222910510020

MOHAMMED SOHEB ALAM, Roll No: 29171022015 Reg No:222910510008

RANJIT JANA, Roll No: 291710220 Reg No:222910510018

under my guidance and supervision.

It is further certified that it contains no material, which to a substantial extent has been submitted for the award of any degree/diploma in any institute or has been published in any form, except the assistances drawn from other sources, for which due acknowledgement has been made.

Date:……….. Guide's signature

Dr. MOUJHURI PATRA

Sd/____________________

Head of the Department

MASTER OF COMPUTER APPLICATION

NETAJI SUBHASH ENGINEERING COLLEGE
TECHNO CITY, GARIA, KOLKATA – 700 152

DECLARATION

We hereby declare that this project report titled

Comment Sentilizer is our own original work carried out as a under graduate student in Netaji Subhash Engineering College except to the extent that assistances from other sources are duly acknowledged.

All sources used for this project report have been fully and properly cited. It contains no material which to a substantial extent has been submitted for the award of any degree/diploma in any institute or has been published in any form, except where due acknowledgement is made.

Student’s names:	Signatures:	Dates:
………………………..	………………………..	……………………
………………………..	………………………..	……………………
………………………..	………………………..	……………………
………………………..	………………………..	……………………

CERTIFICATE OF APPROVAL

We hereby approve this dissertation titled

Comment Sentilizer

carried out by

SHUVODEEP TALUKDAR, Roll No: 29171022013 Reg No:222910510016

RUMI HAZRA, Roll No: 29171022009 Reg No:222910510020

MOHAMMED SOHEB ALAM, Roll No: 29171022015 Reg No:222910510008

RANJIT JANA, Roll No: 291710220 Reg No:222910510018

under the guidance of

Dr. MOUJHURI PATRA

of Netaji Subhash Engineering College, Kolkata in partial fulfillment of requirements for award of the Master of Computer Application of West Bengal University of Technology

Date:………..

Examiners' signatures:

1. ………………………………………….
2. ………………………………………….
3. ………………………………………….

Acknowledgement and/or Dedication

It is indeed a great pleasure to express our thanks and gratitude to all those who helped us during this period. This project would not have been materialized without the help from many quarters. Sincerely thank to all the persons who ever played a vital role in the successful completion of our project. Sincerely thank all the people who co-operate and encourage us throughout the semester and make our project work successful. Thankful to our project's major guide Dr Moujhuri Patra who has constantly remained helpful in suggesting directions and providing us guidance throughout this project.

RUMI HAZRA
RANJIT JANA
MOHAMMED SOHEB ALAM
SHUVODEEP TALUKDAR

Dated:…………………

ABSTRACT

Over time, textual information has increased exponentially, resulting to the potential research within the field of machine learning (ML) and natural language processing (NLP). Sentiment analysis of youtube comments is a very interesting topic now a days. While many of these videos have a significant number of user comments and reviews, little work has been done so far in extracting trends from these comments due to their low information consistency and quality. In this paper, we perform sentiment analysis on the YouTube comments related to popular topics using machine-learning techniques/algorithms. We demonstrate that an analysis of the sentiments to spot their trends, seasonality and forecasts can provide a transparent picture of the influence of real-world events on public sentiments. Results show that the trends in users' sentiments are well correlated to the real world events associated with the respective keywords. The main purpose of this research is to facilitate researchers to identify quality research papers on their sentiment analysis. In this research, sentiment analysis of you-tube comments using citation sentences is carried out using an existing constructed annotated corpus.

This corpus is consisted of 1500 citation sentences. The noise was cleaned from data using different data normalization rules in order to clean the comments from the corpus. To perform classification on this data set we developed a system in which six different machine learning algorithms including Naïve-Bayes (NB), Support Vector Machine (SVM), Logistic Regression (LR), Decision Tree (DT), K-Nearest Neighbour (KNN) and Random Forest (RF) are implemented. Then the accuracy of the system is evaluated using different evaluation metrics e.g. F-score and Accuracy score.

Keywords— Sentimental analysis; citations; machine learning; classification.

INTRODUCTION

Sentiment analysis (or opinion mining) is how social media listening tools analyze and understand the emotions expressed in a text. Generally, the text refers to brand mentions, posts, comments, and reviews of your brand on various social channels.

Sentiment analysis, also referred to as opinion mining, is an approach to natural language processing (NLP) that identifies the emotional tone behind a body of text. This is a popular way for organizations to determine and categorize opinions about a product, service or idea.

Using sentiment analysis, these users' opinions and emotions can be extracted and quantified. This study examines sentiment analysis on Youtube comments and how well the number of comments classified as positive, neutral and negative can be useful in predicting the like proportion of a Youtube video.

OBJECTIVE: -

Sentiment analysis of YouTube comments using AI/ML serves several objectives:

1. **Understanding Audience Sentiment:** It helps in comprehending the emotions, opinions, and reactions of viewers towards specific content, topics, or creators.
2. **Content Optimization:** By analyzing sentiments, creators can refine their content strategy based on audience preferences, improving engagement and relevance.
3. **Creator-Viewer Relationship:** It can assist creators in fostering a better relationship with their audience by understanding their sentiments and responding accordingly.
4. **Identifying Trends and Patterns:** Analyzing sentiments across comments can reveal trends, patterns, and shifts in public opinion, which can be invaluable for marketers, content creators, or researchers.
5. **Improving User Experience:** YouTube can use sentiment analysis to enhance user experience by identifying and addressing common concerns or issues raised by viewers.
6. **Moderation and Filtering:** Sentiment analysis can aid in moderating comments by identifying and filtering out potentially offensive or harmful content.
7. **Insights for Decision-Making:** Businesses and advertisers can leverage sentiment analysis to make informed decisions about collaborations, marketing strategies, and product development based on audience reactions.

MOTIVATION: -

Analysing sentimental comments on YouTube can be motivating for several reasons. Here are a few:

Understanding Emotional Impact:

- **Connection:** Reading heartfelt comments can remind us of the deep emotional connections people forge with content.
- **Inspiration:** Positive sentiments can inspire creators by showing how their work impacts lives.
- **Gratitude:** Seeing gratitude in comments can motivate creators, validating their efforts and dedication.

Improving Content:

- **Insight:** Sentiment analysis helps understand what resonates with viewers, aiding in creating more engaging content.
- **Feedback:** Negative comments can provide constructive criticism, offering avenues for improvement.
- **Adaptation:** Recognizing changing sentiments helps in adapting content to audience preferences.

Human Connection:

- **Empathy:** Sentimental analysis reminds us of the human aspect behind online interactions, fostering empathy.
- **Community Building:** Engaging with sentiments fosters a sense of community, bringing people together over shared emotions.
- **Encouragement:** Responding to comments—especially emotional ones—can create a supportive environment, encouraging more interactions.

Personal Growth:

- **Reflection:** Analyzing sentiments prompts reflection, enabling creators to evaluate their impact and growth.
- **Motivation:** Positive sentiments act as a motivator to keep creating, knowing they're making a difference.
- **Resilience:** Dealing with both positive and negative sentiments builds resilience, crucial in the online space.

Ultimately, sentiment analysis of YouTube comments can serve as a potent reminder of the impact of content, providing insights, fostering connections, and driving personal and creative growth.

HIGHLIGHTS ABOUT WHATS NEW AND INNOVATIVE IN OUR YOUTUBE SENTIMENTAL ANALYSE PROJECT

sentiment analysis of YouTube comments has seen several advancements and potential innovations aimed at enhancing the accuracy, depth, and application of this analysis. Here are some trends and innovations that could have emerged or developed further:

1. **Fine-grained Sentiment Analysis:**
   - **Aspect-based Sentiment Analysis:** Going beyond overall sentiment, this approach dissects comments to understand sentiments towards specific aspects of a video or topic, providing more granular insights.
2. **Multimodal Analysis:**
   - **Integration of Visual and Textual Data:** Leveraging advancements in computer vision to analyse visual components in video comments alongside text for a more comprehensive understanding of sentiment.
   - **Emotion Recognition in Videos:** Utilizing facial expression and emotion recognition techniques to assess sentiment in video comments.
3. **Contextual Understanding:**
   - **Conversation Context:** Enhancing algorithms to understand sentiments in the context of ongoing conversations within comment threads, capturing changes in sentiment as discussions evolve.
   - **Sarcasm and Irony Detection:** Improved models capable of detecting subtle nuances like sarcasm or irony in comments, leading to more accurate sentiment interpretation.
4. **User-Centric Analysis:**
   - **User Behavior Modeling:** Analyzing user behavior and historical interactions to personalize sentiment analysis, providing tailored insights based on individual preferences.
   - **Sentiment Trends across User Groups:** Identifying sentiment patterns within different user demographics or communities to understand varying perspectives.
5. **Real-time and Dynamic Analysis:**
   - **Live Streaming Sentiment Analysis:** Providing real-time sentiment analysis during live streams or trending events, enabling immediate response or content adjustment based on viewer reactions.

- **Temporal Analysis:** Tracking sentiment changes over time to identify trends and patterns, aiding in content strategy adjustments.

6. **Ethical Considerations and Bias Mitigation:**
    - **Ethical AI Practices:** Efforts to ensure fairness, transparency, and ethical considerations in sentiment analysis models to mitigate biases in interpretations.
    - **Bias Detection and Mitigation:** Techniques aimed at identifying and minimizing biases in sentiment analysis algorithms to provide more balanced results.
7. **Interactive and Engaging Platforms:**
    - **User-Generated Sentiment Annotations:** Platforms allowing users to contribute feedback or corrections to sentiment analysis results, fostering engagement and improving accuracy over time.
    - **Visualizing Sentiment Data:** Tools and features for visualizing sentiment data in intuitive ways, making it easier for content creators to understand and act upon insights.

These trends indicate a movement towards more sophisticated, context-aware, and inclusive sentiment analysis methods aimed at providing deeper insights into audience perceptions and behaviors on YouTube. Continual advancements in AI and natural language processing are likely to further refine these techniques for better understanding and utilization of sentiment data from YouTube comments.

In the upcoming chapters, we would see the problem solving methodology, the hardware and software requirements. Software and Hardware Requirements. All computer software needs certain hardware components or other software resources to be present on a computer. These prerequisites are known as (computer) system requirements and are often used as a guideline as opposed to an absolute rule.

METHODOLOGY

Relevant theoretical aspects:

Functional and Operational Requirements:

The primary task in the system development life cycle is the preliminary investigation to determine the feasibility of the proposed system. Feasibility is defined as the practical extent to which a project can be performed successfully. To evaluate feasibility, a feasibility study is performed, which determines whether the solution considered to accomplish the requirements is practical and workable in the software. Information such as resource availability, cost estimation for software development, benefits of the software to the organization after it is developed and cost to be incurred on its maintenance are considered during the feasibility study.

The objective of a software feasibility study, as the name implies, is to assess from the operational, technical, economic and organizational point of view whether the project is viable.

Various other objectives of feasibility study are listed below:

- To analyse whether the software will meet organization requirements.
- To determine whether the software can be implemented using the current technology and within the specified budget and schedule.
- To determine whether the software can be integrated with other existing software.

Types of Feasibility :

Various types of feasibility that are commonly include technical feasibility, operational feasibility, economic feasibility, legal feasibility and schedule feasibility.

1.Technical Feasibility :

Technical feasibility assesses the current resources

(such as hardware and software) and technology, which are required to accomplish user requirements in the software development team within the allocated time and budget. For this, the software development team ascertains whether the current resources and technology can be upgraded or added in the software to accomplish specified user requirements. Technical feasibility also performs the following tasks.

- Analyses the technical skills and capabilities of the software development team members.
- Determines whether the relevant technology is stable and established.
- Ascertain that the technology chosen for software development has a large number of users so that they can be consulted when problem arise or improvement are required.

2. Operational Feasibility :

Operational feasibility assesses the extent to which the required software performs a series of steps to solve business problems user requirement. This feasibility is dependent on human resources (software development team) and involves visualizing whether the software will operate after it is developed and be operative once it is installed. Operational feasibility also performs the following tasks.

- Determines whether the problems anticipated in user requirement are of high priority.
- Determines whether the solution suggested by the software development team is acceptable.
- Analyses whether user will adapt to new software.
- Determines whether the organization is satisfied by the alternative solution proposed by the software development team.

3. Economic Feasibility :

Economic feasibility determines whether the required software is capable of generating financial gains for an organization. It involves the cost incurred on the software development team, estimated cost of hardware and software, cost of hardware and software, cost of performing feasibility study,and so no. For this, it is essential to consider expenses made on purchases (such as hardware purchase) and activities required to carry out softwaredevelopment. In addition, it is necessary to consider the benefits that can be achieved by developing the software. Software is said to be economically feasible ifit focuses on the issues listed below.

- Cost incurred in software development to produce long-term gains for an organization.
- Cost required to conduct full software investigation (such as requirement elicitationand
requirement analysis)
- Cost of hardware, software, development term, and training.

4.Legal Feasibility :

In legal feasibility study project analyzed in legality point ofview. This includes analyzing barriers of legal implementation of project, data protection acts or social media laws, project certificate, license, copyright etc. Overall it can be said that Legal Feasibility Study is study to know if proposed project conform legal and ethical requirements.

5.Schedule Feasibility :

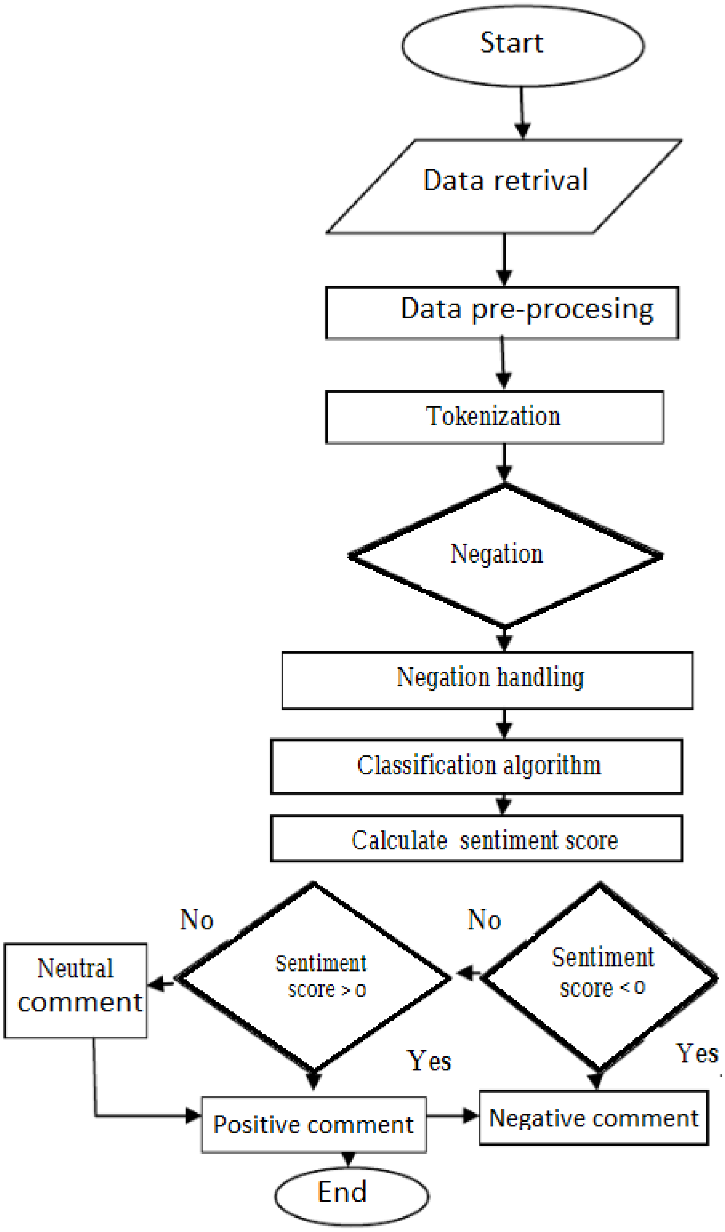
In Schedule Feasibility study mainly timelines/deadlines isanalyzed for proposed project which includes how many times teams will take tocomplete final project which has a great impact on the organization as purpose of the project may fail if it can’t be completed on time.

6.Organizational viability:

it is related to how much the solution benefits the organization. It is verified if there will be adherence to the use of the solution by the users due to the organizational culture and the perception of those involved; whether the solution is aligned with the organization’s strategic objectives; whether there is understanding and support from the organization’s top management in relation to the project, etc.

DESCRIPTION OF FLOWCHART

Youtube comment sentiment analyser FlowChart



Briefly Description:-

Data Retrieval:
To perform sentiment analysis, data can be retrieved from youtube(video) API Real-time sentiment analysis allows for the identification of potential PR crises and the ability to take immediate action before they become serious issues.

Data pre-processing: involves cleaning the text, removing irrelevant information, and transforming the text into a format suitable for analysis. This step also includes feature extraction procedures using algorithms such as the modified inverse class frequency algorithm (LFMI) based on log term.

Tokenization: is the process of breaking down text into individual words or tokens. This step is crucial for further analysis as it allows the text to be processed at a granular level, enabling the extraction of meaningful insights from the data.

Negation: In sentiment analysis, it's important to consider negation handling. This involves identifying and handling negations in the text to accurately determine the sentiment expressed. For example, the presence of "not" changes the sentiment of a word from positive to negative.

Negation Handling: In sentiment analysis, it's important to consider negation handling. This involves identifying and handling negations in the text to accurately determine the sentiment expressed. For example, the presence of "not" changes the sentiment of a word from positive to negative.

Classification Algorithm: A classification algorithm is used to categorize the text into different sentiment classes such as positive, negative, or neutral. This can be achieved through various techniques, including dictionary-based approaches and machine learning models.

Calculate Sentiment Score: Once the classification algorithm has been applied, the sentiment score for the text can be calculated. This score indicates the overall sentiment expressed in the text and can be used to determine whether the sentiment is positive, negative, or neutral.

Output(Positive,Negative,Neutral): Based on the calculated sentiment score, the output of the sentiment analysis can be categorized into three main classes: *positive, **negative, or **neutral*. This categorization provides valuable insights into the sentiment expressed in the analyzed text, which can be used for various applications such as understanding customer feelings and opinions, identifying potential PR crises, and making informed business decisions.

SYSTEM SPECIFICATIONS :

SOFTWARE USED:

Operating System	Windows 10, Ubuntu 20.04 LTS
Programming Language	PYTHON
Client-Side Validation	STREAMLIT
Web Browser	Google Chrome, Microsoft Edge, Firefox
Web Server	STREAMLIT COMMUNITY CLOUD

And, also a dedicated server to run all the above softwares.

HARDWARE USED:

Microprocessor	I7 12th gen (3.5 GHz base clock, up to 4.7GHz maxboost clock, 12MB L3 cache, 10 cores)
Chipset	Intel Integrated SoC
Memory standard	16GB DDR4-2400 MHz RAM(1 x 16 GB)
Hard drive	512 GB PCIe® NVMe™ M.2 SSD
Display	39.6 cm (15.6") diagonal, FHD (1920 x 1080), micro-edge, anti-glare, 220 nits, 45%NTSC

Communication Interface :

The software is in development for a client/server-based setup with a Local Area Network (using the Ethernet interface, one to one connection & TCP/IP protocols) on a stand-alone machine whereby client and server components reside on the same machine.

Description of the components of the system & their interaction:

INRODUCTION TO PYTHON:

Python is a high-level, versatile programming language known for its simplicity and readability. Created by Guido van Rossum and first released in 1991, Python has gained immense popularity due to its ease of learning, clear syntax, and vast community support.

Here are some key features of Python:

1. **Readability**: Python emphasizes code readability and uses indentation to define code blocks, making it easy to understand and write.
2. **Versatility**: It's a general-purpose language used for web development, data analysis, artificial intelligence, scientific computing, scripting, automation, and more.
3. **Large Standard Library**: Python comes with a comprehensive standard library that provides modules and functions for various tasks, reducing the need for external libraries for many common programming tasks.
4. **Interpreted Language**: Python is an interpreted language, meaning the code is executed line by line. This leads to easier debugging and prototyping.
5. **Community and Ecosystem**: Python has a vast and active community contributing to libraries (like NumPy, Pandas, TensorFlow) and frameworks (Django, Flask) that extend its capabilities for various purposes.
6. **Object-Oriented**: Python supports object-oriented programming paradigms, allowing for the creation and usage of classes and objects.
7. **Platform Independence**: Python is platform-independent, meaning code written in Python can run on various platforms without modifications.
8. **Open Source**: Python is free to use and distribute. Its open-source nature encourages collaboration and continual improvement.

Python's simplicity and versatility make it a preferred choice for beginners and professionals alike in various fields, from web development and data science to machine learning and automation. Its vast ecosystem of libraries and frameworks makes it adaptable to a wide range of applications.

INRODUCTION TO PYTHON MODULE:

1.PANDAS:

Pandas, a pivotal Python library, specializes in data manipulation and analysis. Its centerpiece, the DataFrame, organizes data into tables for streamlined handling. With an array of functions, it excels in data cleaning, transformation, and exploration. Pandas effortlessly handles various file formats and integrates seamlessly with NumPy, making it a cornerstone for data scientists, analysts, and developers. Its versatility and powerful tools facilitate tasks ranging from basic data cleaning to complex analytics, playing a crucial role in diverse industries and research domains.

2.NLTK:

NLTK (Natural Language Toolkit) is a Python library designed for natural language processing tasks. It offers tools for tokenization, stemming, tagging, parsing, and semantic reasoning, aiding tasks like sentiment analysis, machine translation, and information extraction. With various corpora, lexical resources, and algorithms, NLTK serves as an educational resource and a practical tool for researchers and developers in linguistics, machine learning, and AI, enabling exploration and analysis of human language data for diverse applications.

3.PLOTLY:

Plotly is a Python library known for creating interactive and visually appealing visualizations. It offers a range of chart types, including scatter plots, bar charts, and heatmaps, with interactive capabilities like zooming and hovering. With support for various platforms and frameworks, Plotly enables seamless integration across web-based applications and data science environments. Its versatility and user-friendly interface make it a go-to choice for generating dynamic, publication-quality visualizations for data analysis, presentations, and web applications.

4.STREAMLIT:

Streamlit is a Python library revolutionizing rapid web app development. It simplifies creating interactive web applications for data science and machine learning projects. With its intuitive API, developers transform scripts into shareable apps effortlessly. Streamlit handles complex tasks like data visualization, model deployment, and user interaction seamlessly, offering widgets for user inputs and dynamic updates. Its real-time preview and automatic reruns streamline development, making it a favored choice for quick prototyping and deploying data-centric apps. Streamlit's emphasis on simplicity and speed accelerates the creation of powerful, user-friendly applications without the need for extensive web development expertise.

5.YOUTUBECOMMENTSCRAPER:

The "youtubecommentscrapper" module is a Python tool designed for extracting and analyzing comments from YouTube videos. It enables users to retrieve comments, usernames, timestamps, and other metadata from videos via YouTube's API. This module simplifies the process of collecting and storing comments for research, sentiment analysis, or content moderation purposes. By providing an interface to access and process YouTube comments programmatically, it offers convenience for developers and analysts seeking to extract insights or work with comment data from YouTube videos.

6. COLORMA:

Colorama, a Python library, simplifies cross-platform colored terminal text printing. It offers easy text colorization and styling for command-line interfaces, enhancing readability and user experience. Colorama abstracts ANSI escape character sequences, enabling developers to apply foreground and background colors, styles, and text formatting effortlessly. With its simplicity and compatibility across various operating systems, Colorama serves as a go-to solution for developers aiming to add colorful and styled text output to their command-line applications, making interactions more visually appealing and intuitive.

7.GOOGLEAPICLIENT:

The **googleapiclient** module, part of Google's client library for Python, facilitates interaction with various Google APIs. It simplifies the process of making requests to Google services like Gmail, Google Drive, YouTube, and more, streamlining API usage and data retrieval. With easy-to-use methods and authentication mechanisms, developers can access and manipulate Google services programmatically, enabling tasks such as data retrieval, updates, and integration of Google functionalities into their applications with simplicity and efficiency

8.PROTOBUF:

The Protobuf Python module allows seamless integration of Protocol Buffers in Python applications. It offers functionalities to serialize and deserialize data, following predefined message formats specified in .proto files. This module generates Python classes from these definitions, enabling efficient data encoding and decoding, maximizing speed and minimizing space usage. With its simplicity and performance benefits, the Protobuf Python module provides a robust solution for data serialization, facilitating streamlined communication and data exchange across various systems and platforms.

Data Collection Method:

Automated Data Collection: The data collection process for YouTube sentiment analysis can be automated using web scraping tools, allowing for the extraction of comments from YouTube videos and exporting them as a CSV or Excel file.

Integration with Apps: Tools like Monkey Learn offer integrations with apps such as Zen desk, Fresh desk, Survey Monkey, Gmail, and Zapier, making it convenient to gather YouTube data and perform sentiment analysis without the need for coding.

Other Methods: In addition to web scraping and integrations with apps, there are also mentions of sentiment analysis using big data from YouTube videos metadata and the use of sentiment analysis tools for data extraction and processing.

Data analysis and interpretation :

Sentiment analysis in the context of YouTube involves the interpretation and classification of emotions within text and voice data, allowing for the understanding of the sentiment (positive, negative, or neutral) expressed in the content. This analysis is crucial for businesses and organizations to gauge public opinion, understand customer feedback, and make data-driven decisions.

Analyzing sentiment on YouTube involves examining the sentiment expressed in the comments and content of videos. This process can be performed using text mining and sentiment analysis techniques, which allow for the evaluation of sentiment without human judgment, reducing subjectivity and potential bias.

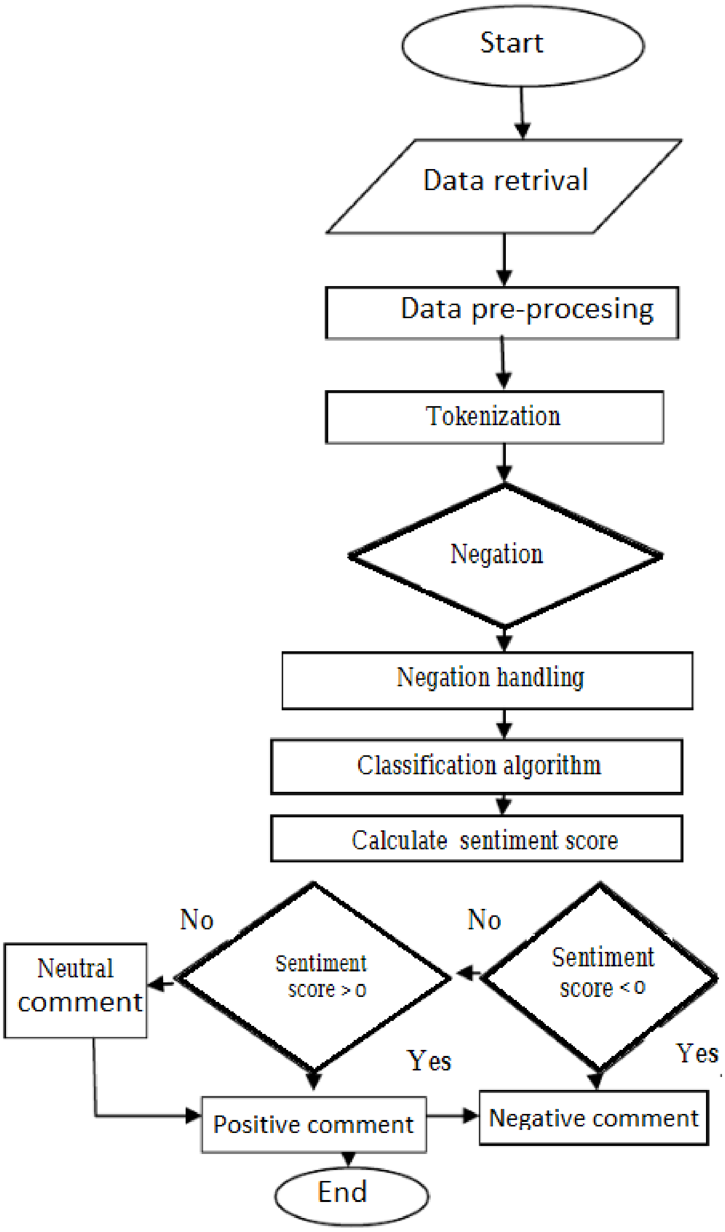
Various tools and platforms, such as R for text analysis, Power BI with cognitive services, and Google Cloud's Natural Language API, offer capabilities for sentiment analysis, entity analysis, content classification, and syntax analysis, enabling users to gain insights from unstructured data, including text and voice content.

System Design

FUNCTIONAL FLOWCHART :

A system consists of many different activities or processes. We know the relation between the processes that process will contain several individual processes. Weoften show these relations in terms of process charts.

Youtube comment sentiment analyser FlowChart



IMPLEMENTATION

Modular Pseudo Code:

```
1. FUNCTION get_channel_id WITH PARAMETER video_id
   SET response TO CALL youtube.videos().list(part='snippet', id=video_id).execute()
   SET channel_id TO response['items'][0]['snippet']['channelId']
   RETURN channel_id
END FUNCTION


2. FUNCTION save_video_comments_to_csv WITH PARAMETER video_id
   SET comments TO an empty list
   SET results TO CALL youtube.commentThreads().list() WITH PARAMETERS part='snippet', videoId=video_id, textFormat='plainText'.execute()
      WHILE results is not empty:
      FOR EACH item IN results['items']:
         SET comment TO item['snippet']['topLevelComment']['snippet']['textDisplay']
         SET username TO item['snippet']['topLevelComment']['snippet']['authorDisplayName']
         APPEND [username, comment] TO comments
      IF 'nextPageToken' is in results:
         SET nextPage TO results['nextPageToken']
         SET results TO CALL youtube.commentThreads().list() WITH PARAMETERS part='snippet', videoId=video_id, textFormat='plainText', pageToken=nextPage.execute()
      ELSE:
         BREAK
   SET filename TO video_id + '.csv'
   OPEN csvfile WITH filename, 'w', newline='', encoding='utf-8'
   SET writer TO csv.writer(csvfile)
```

```
    CALL writer.writerow() WITH PARAMETERS ['Username', 'Comment']
    FOR EACH comment IN comments:
        CALL writer.writerow() WITH PARAMETERS [comment[0], comment[1]]
    RETURN filename
END FUNCTION

3. FUNCTION get_video_stats WITH PARAMETER video_id
    TRY
        SET response TO CALL youtube.videos().list() WITH PARAMETERS part='statistics', id=video_id.execute()
        RETURN response['items'][0]['statistics']
    EXCEPT HttpError AS error
        PRINT 'An error occurred: ' CONCATENATE error
        RETURN None
END FUNCTION

4. FUNCTION get_channel_info WITH PARAMETERS youtube, channel_id
    TRY
        SET response TO CALL youtube.channels().list() WITH PARAMETERS part='snippet,statistics,brandingSettings', id=channel_id.execute()
        SET channel_title TO response['items'][0]['snippet']['title']
        SET video_count TO response['items'][0]['statistics']['videoCount']
        SET channel_logo_url TO response['items'][0]['snippet']['thumbnails']['high']['url']
        SET channel_created_date TO response['items'][0]['snippet']['publishedAt']
        SET subscriber_count TO response['items'][0]['statistics']['subscriberCount']
        SET channel_description TO response['items'][0]['snippet']['description']
        SET channel_info TO {
            'channel_title': channel_title,
            'video_count': video_count,
            'channel_logo_url': channel_logo_url,
            'channel_created_date': channel_created_date,
            'subscriber_count': subscriber_count,
            'channel_description': channel_description
        }
        RETURN channel_info
    EXCEPT HttpError AS error
        PRINT 'An error occurred: ' CONCATENATE error
        RETURN None
END FUNCTION

5. FUNCTION extract_video_id WITH PARAMETER youtube_link
    SET video_id_regex TO r"^(?:https?:\/\/)?(?:www\.)?(?:youtube\.com\/watch\?v=|youtu.be\/)([a-zA-Z0-9_-]{11})"
    SET match TO CALL re.search() WITH PARAMETERS video_id_regex, youtube_link
    IF match is not None:
        SET video_id TO match.group(1)
        RETURN video_id
    ELSE:
        RETURN None
END FUNCTION

6. FUNCTION analyze_sentiment WITH PARAMETER csv_file
    # Initialize the sentiment analyzer
    sid = SentimentIntensityAnalyzer()
    # Read in the YouTube comments from the CSV file
    comments = []
    OPEN csvfile WITH csv_file, 'r', encoding='utf-8-sig'
    SET reader TO csv.DictReader(csvfile)
    FOR EACH row IN reader:
        APPEND row['Comment'] TO comments
    # Count the number of neutral, positive, and negative comments
    SET num_neutral TO 0
    SET num_positive TO 0
    SET num_negative TO 0
    FOR EACH comment IN comments:
        SET sentiment_scores TO sid.polarity_scores(comment)
        IF sentiment_scores['compound'] EQUALS 0.0:
            INCREMENT num_neutral BY 1
        ELSE IF sentiment_scores['compound'] GREATER THAN 0.0:
            INCREMENT num_positive BY 1
        ELSE:
            INCREMENT num_negative BY 1
    # Return the results as a dictionary
    SET results TO {'num_neutral': num_neutral, 'num_positive': num_positive, 'num_negative': num_negative}
    RETURN results
END FUNCTION


7. FUNCTION bar_chart WITH PARAMETER csv_file OF TYPE str AND RETURN TYPE None
    # Call analyze_sentiment function to get the results
    results: Dict[str, int] = CALL analyze_sentiment() WITH PARAMETER csv_file
    # Get the counts for each sentiment category
    num_neutral = results['num_neutral']
    num_positive = results['num_positive']
    num_negative = results['num_negative']
    # Create a Pandas DataFrame with the results
    df = pd.DataFrame({
        'Sentiment': ['Positive', 'Negative', 'Neutral'],
        'Number of Comments': [num_positive, num_negative, num_neutral]
    })
    # Create the bar chart using Plotly Express
    fig = px.bar(df, x='Sentiment', y='Number of Comments', color='Sentiment',
                 color_discrete_sequence=['#87CEFA', '#FFA07A', '#D3D3D3'],
                 title='Sentiment Analysis Results')
    fig.update_layout(title_font=dict(size=20))
    # Show the chart
    st.plotly_chart(fig, use_container_width=True)
END FUNCTION


8. FUNCTION plot_sentiment WITH PARAMETER csv_file OF TYPE str AND RETURN TYPE None
    # Call analyze_sentiment function to get the results
    results: Dict[str, int] = CALL analyze_sentiment() WITH PARAMETER csv_file
    # Get the counts for each sentiment category
    num_neutral = results['num_neutral']
    num_positive = results['num_positive']
    num_negative = results['num_negative']
    # Plot the pie chart
    labels = ['Neutral', 'Positive', 'Negative']
    values = [num_neutral, num_positive, num_negative]
    colors = ['yellow', 'green', 'red']
    fig = go.Figure(data=[go.Pie(labels=labels, values=values, textinfo='label+percent',
                                 marker=dict(colors=colors))])
    fig.update_layout(title={'text': 'Sentiment Analysis Results', 'font': {'size': 20, 'family': 'Arial', 'color': 'grey'},
                             'x': 0.5, 'y': 0.9},
                    font=dict(size=14))
    st.plotly_chart(fig)
END FUNCTION


9. FUNCTION create_scatterplot WITH PARAMETERS csv_file OF TYPE str,
x_column OF TYPE str, y_column OF TYPE str AND RETURN TYPE None
    # Load data from CSV
    data = pd.read_csv(csv_file)
    # Create scatter plot using Plotly
    fig = px.scatter(data, x=x_column, y=y_column, color='Category')
    # Customize layout
    fig.update_layout(
        title='Scatter Plot',
        xaxis_title=x_column,
        yaxis_title=y_column,
        font=dict(size=18)
    )
    # Display plot in Streamlit
    st.plotly_chart(fig, use_container_width=True)
END FUNCTION


10. FUNCTION print_sentiment WITH PARAMETER csv_file OF TYPE str AND
RETURN TYPE None
    # Call analyze_sentiment function to get the results
    results: Dict[str, int] = CALL analyze_sentiment() WITH PARAMETER
csv_file
    # Get the counts for each sentiment category
    num_neutral = results['num_neutral']
    num_positive = results['num_positive']
    num_negative = results['num_negative']
    # Determine the overall sentiment
    IF num_positive > num_negative:
        overall_sentiment = 'POSITIVE'
        color = Fore.GREEN
    ELSE IF num_negative > num_positive:
        overall_sentiment = 'NEGATIVE'
        color = Fore.RED
    ELSE:
        overall_sentiment = 'NEUTRAL'
        color = Fore.YELLOW
    # Print the overall sentiment in color
    PRINT '\n'+ Style.BRIGHT+ color + overall_sentiment.upper().center(50, ' ') + Style.RESET_ALL
END FUNCTION


11. FUNCTION delete_non_matching_csv_files WITH PARAMETERS directory_path, video_id
    FOR EACH file_name IN os.listdir(directory_path):
        IF NOT file_name.endswith('.csv'):
            CONTINUE
        IF file_name EQUALS f'{video_id}.csv':
            CONTINUE
        os.remove(os.path.join(directory_path, file_name))
END FUNCTION


st.set_page_config(page_title='Sentiment Analysis', page_icon = 'LOGO.png', initial_sidebar_state = 'auto')
st.sidebar.title("Sentimental Analsis")
st.sidebar.header("Enter YouTube Link")
youtube_link = st.sidebar.text_input("Link")
directory_path = os.getcwd()
hide_st_style = """
            <style>
            #MainMenu {visibility: hidden;}
            footer {visibility: hidden;}
            </style>
            """
st.markdown(hide_st_style, unsafe_allow_html=True)


IF youtube_link:
    video_id = extract_video_id(youtube_link)
    channel_id = get_channel_id(video_id)
    IF video_id:
        st.sidebar.write("The video ID is:", video_id)
        csv_file = save_video_comments_to_csv(video_id)
        delete_non_matching_csv_files(directory_path,video_id)
        st.sidebar.write("Comments saved to CSV!")
        st.sidebar.download_button(label="Download Comments",
data=open(csv_file, 'rb').read(), file_name=os.path.basename(csv_file),
mime="text/csv")


        channel_info = get_channel_info(youtube,channel_id)


        col1, col2 = st.columns(2)


        WITH col1:
           channel_logo_url = channel_info['channel_logo_url']
           st.image(channel_logo_url, width=250)


        WITH col2:
           channel_title = channel_info['channel_title']
           st.title(' ')
           st.text(" YouTube Channel Name ")
           st.title(channel_title)
           st.title("  ")
```

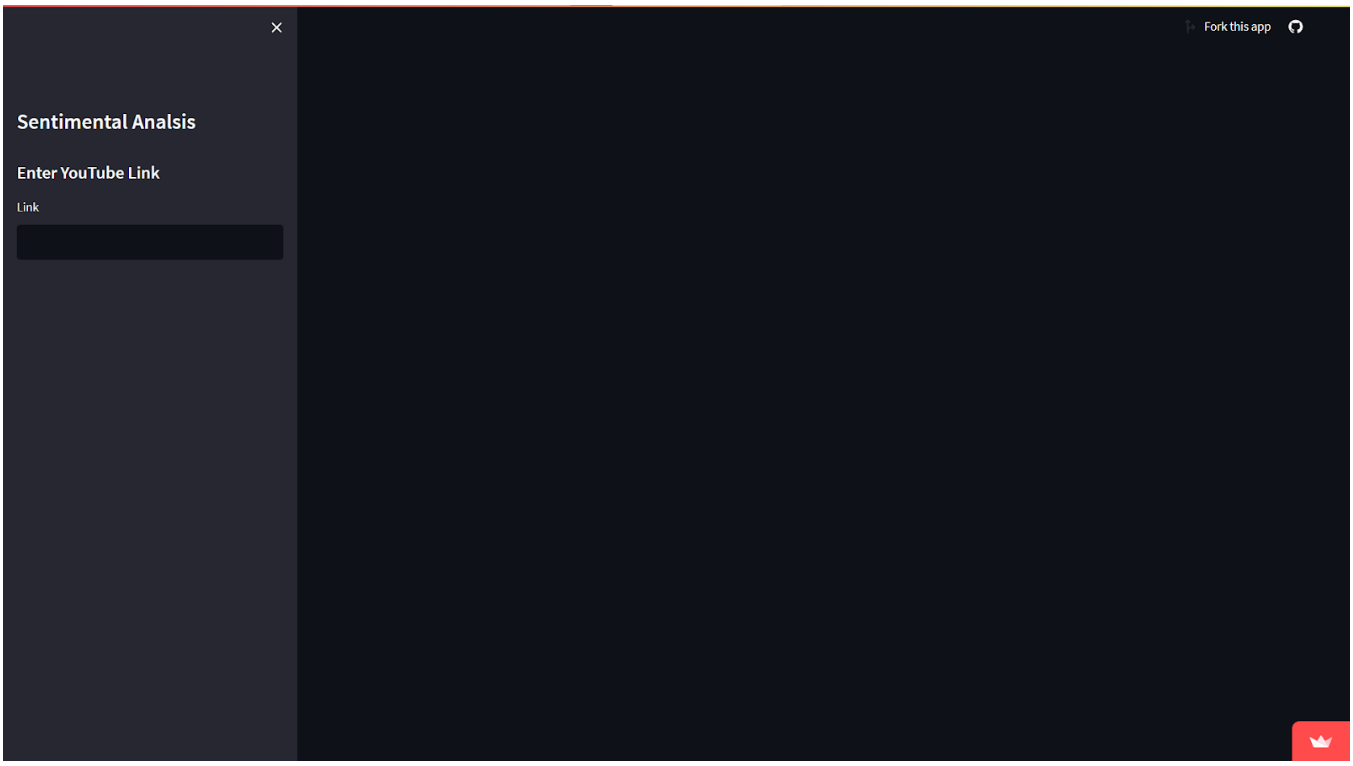
```
  st.title(" ")
  st.title(" ")
col3, col4 ,col5 = st.columns(3)
WITH col3:
  video_count=channel_info['video_count']
  st.header(" Total Videos ")
  st.subheader(video_count)
WITH col4:
  channel_created_date= channel_info['channel_created_date']
  created_date = channel_created_date[:10]
  st.header("Channel Created ")
  st.subheader(created_date)
WITH col5:
   st.header(" Subscriber_Count ")
   st.subheader(channel_info["subscriber_count"])
stats = get_video_stats(video_id)
st.title("Video Information :")
col6, col7 ,col8 = st.columns(3)
WITH col6:
   st.header(" Total Views ")
   st.subheader(stats["viewCount"])
WITH col7:
  st.header(" Like Count ")
  st.subheader(stats["likeCount"])

WITH col8:
   st.header(" Comment Count ")
   st.subheader(stats["commentCount"])
_, container, _ = st.columns([10, 80, 10])
container.video(data=youtube_link)
results = analyze_sentiment(csv_file)
col9, col10 ,col11 = st.columns(3)
```

```
    WITH col9:
        st.header(" Positive Comments ")
        st.subheader(results['num_positive'])
    WITH col10:
       st.header(" Negative Comments ")
       st.subheader( results['num_negative'])
    WITH col11:
        st.header(" Neutral Comments ")
        st.subheader(results['num_neutral'])
    bar_chart(csv_file)
    plot_sentiment(csv_file)
    st.subheader("Channel Description ")
    channel_description = channel_info['channel_description']
    st.write(channel_description)
ELSE:
    st.error("Invalid YouTube link")
```

Testing results

Manual Testing:

HOME PAGE:

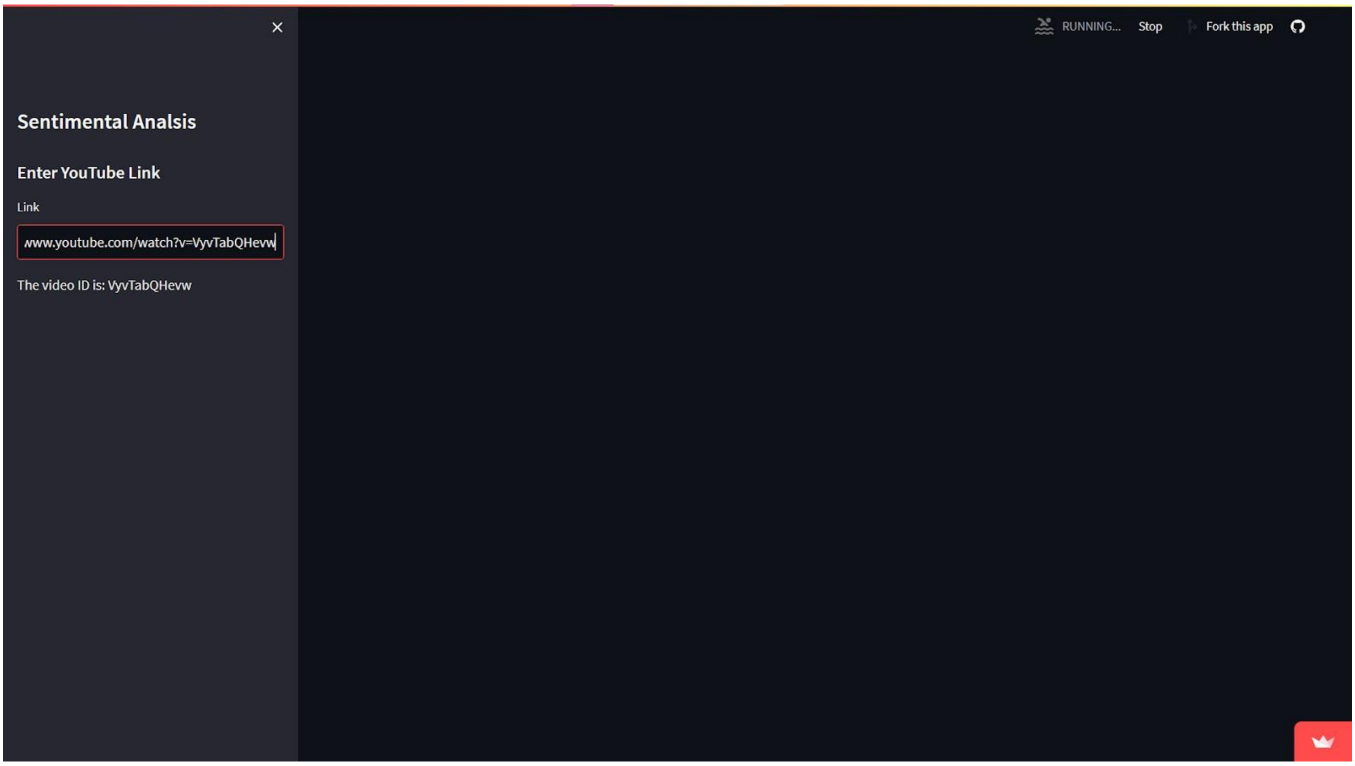


Sentimental Analsis

Enter YouTube Link

Link

PROVIDING THE LINK:



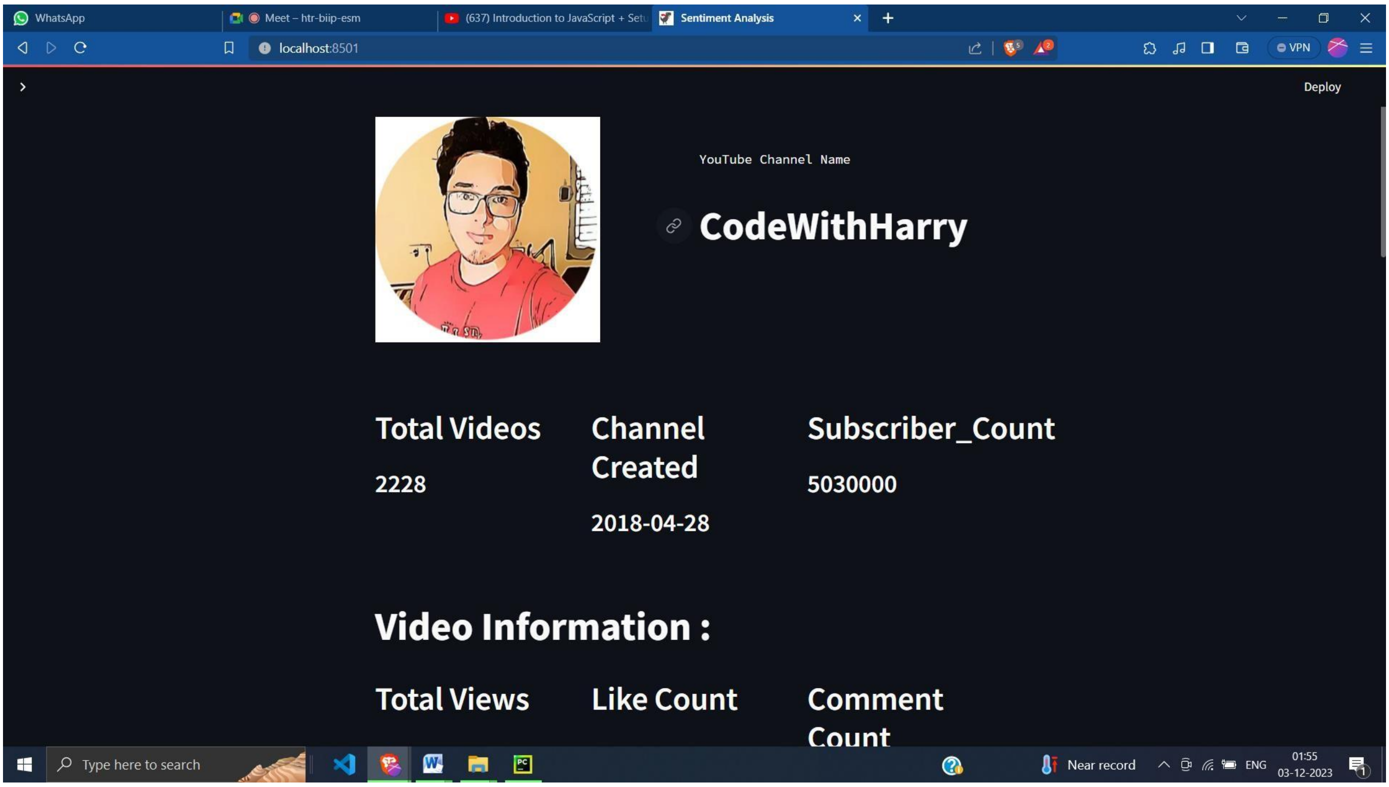
Sentimental Analsis

Enter YouTube Link

Link: www.youtube.com/watch?v=VyvTabQHevw

The video ID is: VyvTabQHevw

ANALYSED RESULT:



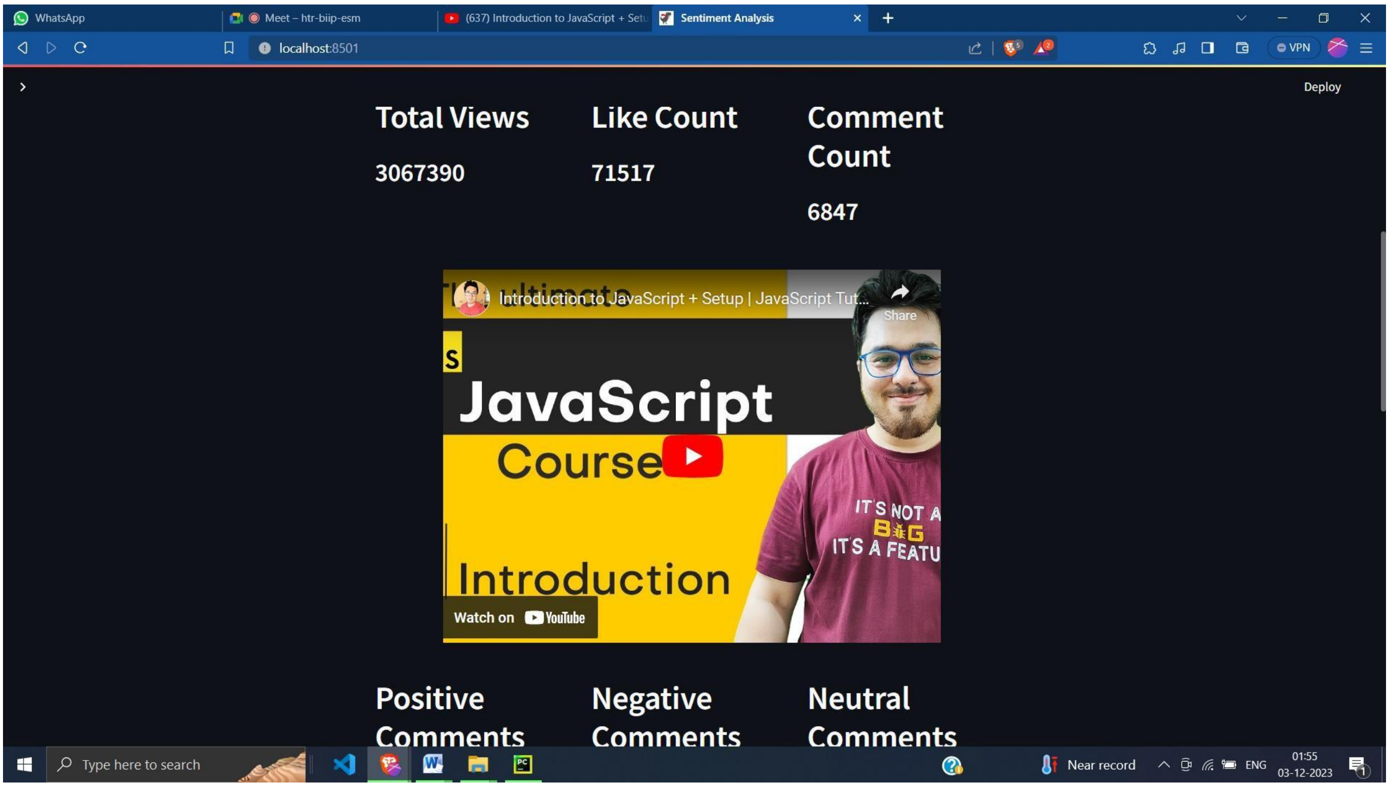
YouTube Channel Name

CodeWithHarry

Total Videos	Channel Created	Subscriber_Count
2228	2018-04-28	5030000

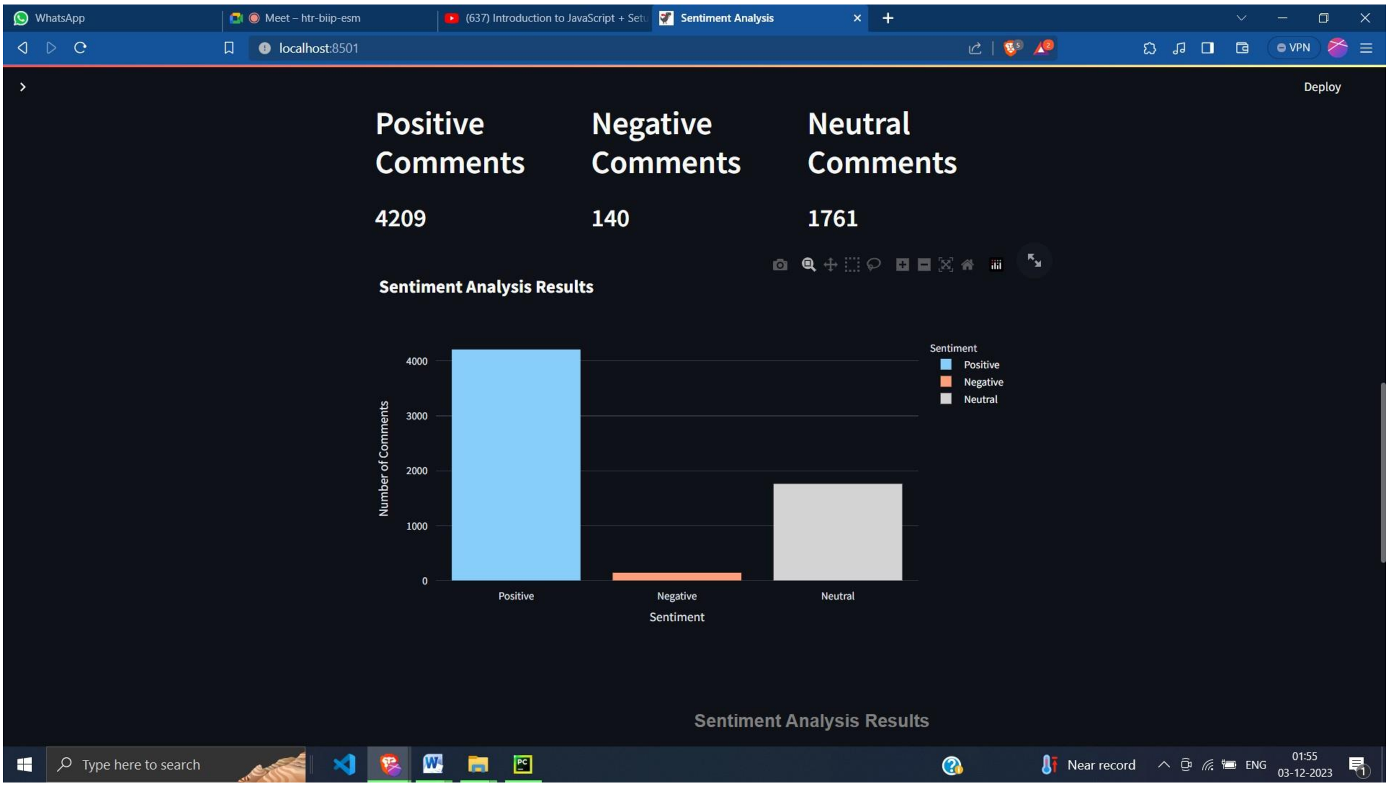
Video Information :

Total Views	Like Count	Comment Count



Total Views	Like Count	Comment Count
3067390	71517	6847

Positive Comments	Negative Comments	Neutral Comments



Positive Comments

4209

Negative Comments

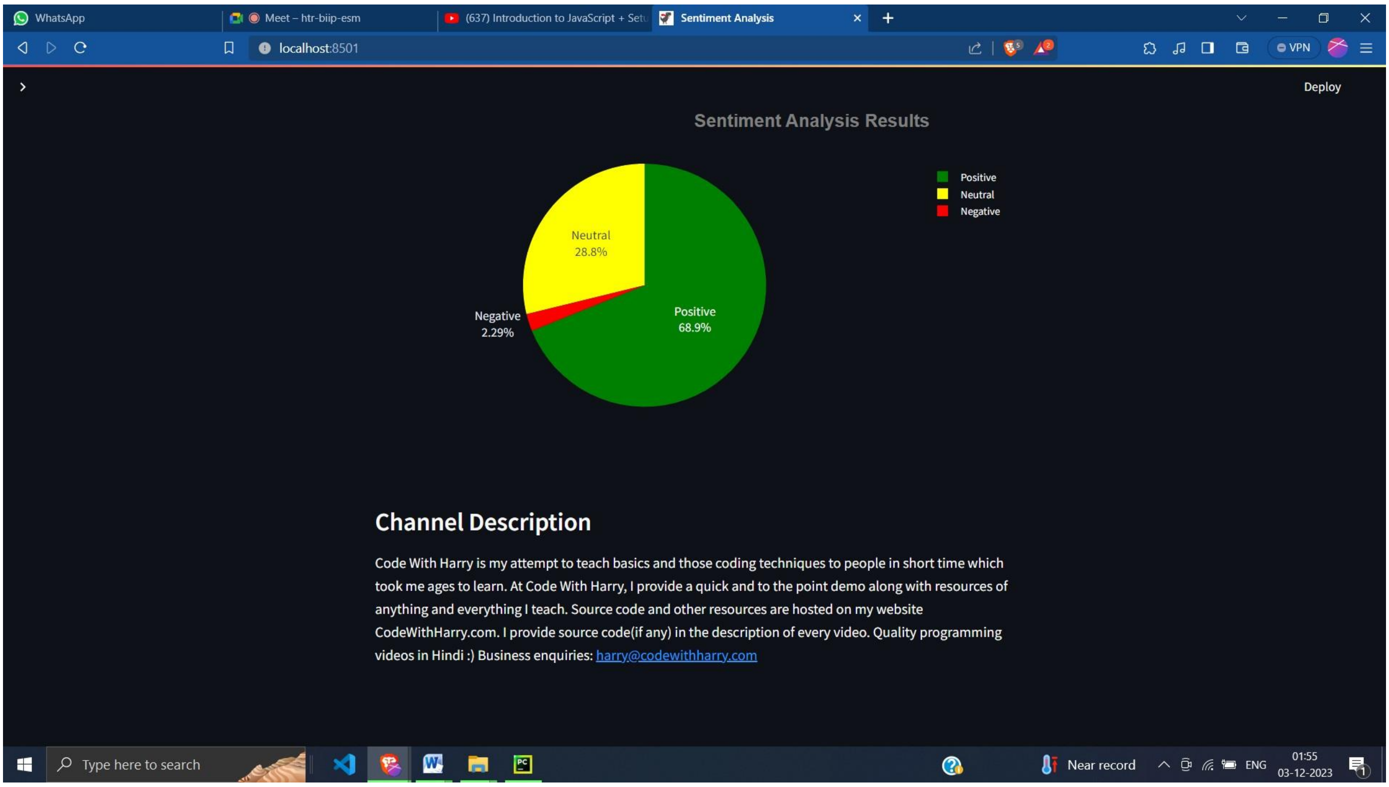
140

Neutral Comments

1761

Sentiment Analysis Results

Sentiment Analysis Results



Sentiment Analysis Results

Channel Description

Code With Harry is my attempt to teach basics and those coding techniques to people in short time which took me ages to learn. At Code With Harry, I provide a quick and to the point demo along with resources of anything and everything I teach. Source code and other resources are hosted on my website CodeWithHarry.com. I provide source code(if any) in the description of every video. Quality programming videos in Hindi :) Business enquiries: harry@codewithharry.com

SCOPE

The scope of sentiment analysis of YouTube comments is quite broad and diverse, and it can serve various purposes:

1. **Audience Engagement**: Understanding the sentiment of comments helps creators gauge audience reactions to their content. Positive sentiment indicates appreciation, while negative sentiment might highlight areas for improvement.
2. **Content Optimization**: Analyzing sentiments helps creators identify the type of content that resonates most with their audience. They can tailor future videos based on the sentiments expressed in comments.
3. **Brand Reputation Management**: For businesses or brands using YouTube, sentiment analysis can provide insights into how their audience perceives their products or services. This information can guide marketing strategies and help in managing brand reputation.
4. **Trend Analysis**: Analyzing sentiments across a range of videos or topics can reveal trends and patterns in audience preferences or opinions on particular subjects.
5. **Community Insights**: Sentiment analysis can help in understanding the community dynamics within the comment sections, identifying key influencers or active participants.
6. **Content Moderation**: It aids in content moderation by identifying and flagging potentially offensive or harmful comments, allowing for a more positive and constructive environment.
7. **Market Research**: For market researchers, sentiment analysis on YouTube comments can provide valuable insights into consumer opinions, preferences, and trends related to various products or services.
8. **Policy and Decision Making**: Governments or organizations might analyze sentiments in comments related to social issues or public policies to gauge public opinion and sentiment towards certain decisions or events.

The scope extends to various industries and purposes, ranging from content creation to marketing, customer feedback analysis, and even sociological or psychological studies. However, it's important to note that sentiment analysis might not always accurately capture the nuances of human emotions and context, so combining it with other analytical methods can provide a more comprehensive understanding.

Limitations of the Project

Limitations of the Project in YouTube Comment Sentiment Analysis
The project involving YouTube comment sentiment analysis may encounter several limitations, as highlighted in various sources:

1.**Data Gathering Limitations**: The process of gathering YouTube comments and associated sentiment analysis may be limited by the availability and accessibility of the data. The project may face challenges in retrieving and processing a comprehensive set of comments for analysis.

2.**Exploratory Nature of Social Media Analytics**: The generic limitations of social media analytics, including YouTube comment sentiment analysis, stem from their fundamentally exploratory multi-method nature. This suggests that the project may face inherent limitations in the nature of the data and the methods used for analysis .

3.**Polarity Classification Challenges:** The project may encounter challenges in accurately determining the polarity of comments, especially in cases where comments exhibit nuanced or ambiguous sentiment. This could impact the precision of the sentiment analysis .

4.**Volume of Comments:** YouTube videos can attract thousands of comments, posing a challenge in efficiently processing and analyzing a large volume of comments. The scalability of sentiment analysis algorithms and techniques may be a limitation for the project .

5.**API and Data Access Constraints**: The project may face limitations related to accessing YouTube data through the YouTube Data API. This could include constraints on the volume of data that can be accessed, API rate limits, and authentication requirements .
Accuracy of Sentiment Analysis: The accuracy of sentiment analysis techniques used in the project may be limited by the complexity of natural language, sarcasm, and context-dependent sentiment expressions in YouTube comments.

CONCLUSION

This project is successfully completed and works properly according to the need. The frontend of this project has been developed using Streamlit and the backend has been developed using python. The system maintains data consistency by avoiding manual error and documents are maintained accurately which reduces the losses that can be made due to various environment features. All the requirement regarding to this problem are solved the needs specified in the problem definition are fulfilled. This project will help all end users as a user-friendly.

Future Scope :

1.Currently our project is web based only. In future we aim to launch it as a desktop and android application for ease of our users.

2. With increase in our user base we would take steps to improve data security and add review system and further exploring the sentiments for sensitivity.

3. Built-in tools like video conferencing, automated chat-bots, etc to facilitate better communication between both the parties, to be added.

4.In future, we can add twitter, amazom , facebook , Instagram, etc comment analyser

References:

i) **PYTHON PROGRAMMING** by Atanu Das (Author), Rajkumar Patra (Author)

ii) The Hundred-Page Machine Learning Book Andriy Burkov (Author)

Bibliography

The sources related to YouTube comment sentiment analysis in a project include:

1."Having problems in displaying youtube comment sentiment analysis"
2."Sentiment Analysis of Positive and Negative of YouTube Comments"
3."(PDF) Sentiment Analysis on Youtube Comments: A brief study"
4."Sentiment Analysis for YouTube Comments in Roman Urdu"
5."youtube-comment-sentiment-analysis GitHub Topics GitHub"
6."Sentiment Analysis on YouTube: A Brief Survey"
7."chiaszu/youtube-comments-sentiment-analysis: YouTube..."
8."Master Thesis Project Sentiment Analysis of YouTube Public Videos"
9."Using text mining and sentiment analysis to analyse YouTube Italian videos concerning vaccination"
10."Text Data Analysis By Python (Youtube Case Study)"

Project Code

app.py:

```
import streamlit as st
import os
from Senti import extract_video_id,analyze_sentiment,bar_chart,plot_sentiment
from YoutubeCommentScrapper import save_video_comments_to_csv,get_channel_info,youtube,get_channel_id,get_video_stats
def delete_non_matching_csv_files(directory_path, video_id):
    for file_name in os.listdir(directory_path):
        if not file_name.endswith('.csv'):
            continue
        if file_name == f'{video_id}.csv':
            continue
        os.remove(os.path.join(directory_path, file_name))
st.set_page_config(page_title='Sentiment Analysis', page_icon = 'LOGO.png', initial_sidebar_state = 'auto')
#st.set_page_config(page_title=None, page_icon=None, layout="centered", initial_sidebar_state="auto", menu_items=None)
st.sidebar.title("Sentimental Analsis")
st.sidebar.header("Enter YouTube Link")
youtube_link = st.sidebar.text_input("Link")
directory_path = os.getcwd()
hide_st_style = """
            <style>
            #MainMenu {visibility: hidden;}
            footer {visibility: hidden;}
            </style>
            """
st.markdown(hide_st_style, unsafe_allow_html=True)
if youtube_link:
    video_id = extract_video_id(youtube_link)
    channel_id = get_channel_id(video_id)
    if video_id:
        st.sidebar.write("The video ID is:", video_id)
        csv_file = save_video_comments_to_csv(video_id)
        delete_non_matching_csv_files(directory_path,video_id)
    st.sidebar.write("Comments saved to CSV!")
    st.sidebar.download_button(label="Download Comments",
data=open(csv_file, 'rb').read(),
file_name=os.path.basename(csv_file), mime="text/csv")

    #using fn
    channel_info = get_channel_info(youtube,channel_id)

    col1, col2 = st.columns(2)

    with col1:
      channel_logo_url = channel_info['channel_logo_url']
      st.image(channel_logo_url, width=250)

    with col2:
      channel_title = channel_info['channel_title']
      st.title(' ')
      st.text(" YouTube Channel Name ")
      #st.markdown('** YouTube Channel Name **')
      st.title(channel_title)
      st.title("  ")
      st.title(" ")
      st.title(" ")


    #Using fn



    st.title(" ")
    col3, col4 ,col5 = st.columns(3)


    with col3:
      video_count=channel_info['video_count']
      st.header(" Total Videos ")
      #st.subheader("Total Videos")
      st.subheader(video_count)

    with col4:
      channel_created_date=
channel_info['channel_created_date']
      created_date = channel_created_date[:10]
      st.header("Channel Created ")
      st.subheader(created_date)

    with col5:

         st.header(" Subscriber_Count ")
         st.subheader(channel_info["subscriber_count"])

    st.title(" ")

    stats = get_video_stats(video_id)

    st.title("Video Information :")
    col6, col7 ,col8 = st.columns(3)


    with col6:
         st.header(" Total Views ")
      #st.subheader("Total Videos")
      st.subheader(stats["viewCount"])

    with col7:
      st.header(" Like Count ")
      st.subheader(stats["likeCount"])


    with col8:

         st.header(" Comment Count ")
         st.subheader(stats["commentCount"])

    st.header(" ")


    _, container, _ = st.columns([10, 80, 10])
    container.video(data=youtube_link)





    results = analyze_sentiment(csv_file)


    col9, col10 ,col11 = st.columns(3)
```

```
        with col9:
            st.header(" Positive Comments ")
           #st.subheader("Total Videos")
           st.subheader(results['num_positive'])

        with col10:
           st.header(" Negative Comments ")
           st.subheader( results['num_negative'])


        with col11:

            st.header(" Neutral Comments ")
            st.subheader(results['num_neutral'])


        bar_chart(csv_file)

        plot_sentiment(csv_file)


        st.subheader("Channel Description ")
        channel_description = channel_info['channel_description']
        st.write(channel_description)

    else:
        st.error("Invalid YouTube link")
```

YoutubeCommentScrapper.py:

```
import csv
from googleapiclient.discovery import build
from collections import  Counter
import streamlit as  st
from Senti import  extract_video_id
from googleapiclient.errors import  HttpError

import warnings
warnings.filterwarnings('ignore')

#  Replace with your own API  key
DEVELOPER_KEY = "AIzaSyDZnkEibEZhbU9nITXRI-ed1eirNYEzI_U"
YOUTUBE_API_SERVICE_NAME = 'youtube'
YOUTUBE_API_VERSION = 'v3'
# Create a client object to interact with the YouTube API
youtube = build(YOUTUBE_API_SERVICE_NAME,
YOUTUBE_API_VERSION,  developerKey=DEVELOPER_KEY)

#video_id=extract_video_id(youtube_link) def

get_channel_id(video_id):
    response = youtube.videos().list( part='snippet',
id=video_id).execute()
    channel_id = response['items'][0]['snippet']['channelId']
    return channel_id

#channel_id=get_channel_id(video_id)


def save_video_comments_to_csv(video_id):
    # Retrieve comments for the specified video using the
comments().list() method
    comments = []
    results = youtube.commentThreads().list(
        part='snippet',
        videoId=video_id,
        textFormat='plainText'
    ).execute()
    # Extract the text content of each comment and add it to the
comments list
    while results:
        for item in results['items']:
            comment =
item['snippet']['topLevelComment']['snippet']['textDisplay']
            username =
item['snippet']['topLevelComment']['snippet']['authorDisplayName']
            comments.append([username,comment])
        if 'nextPageToken' in  results:
            nextPage = results['nextPageToken']
            results =  youtube.commentThreads().list(
                part='snippet',
                videoId=video_id,
                textFormat='plainText',
                pageToken=nextPage
            ).execute()
        else:
            break

    # Save the comments to a CSV file with the video ID as the filename
    filename = video_id + '.csv'
    with open(filename, 'w', newline='', encoding='utf-8') as csvfile: writer
        = csv.writer(csvfile)
        writer.writerow(['Username','Comment'])
        for comment in  comments:
            writer.writerow([comment[0],comment[1]])

    return filename

def get_video_stats(video_id):
    try:
        response = youtube.videos().list(
            part='statistics',
            id=video_id
        ).execute()

        return response['items'][0]['statistics']

    except HttpError as  error:
        print(f'An error occurred: {error}')
        return None
```

```
def get_channel_info(youtube, channel_id): try:
        response = youtube.channels().list(
            part='snippet,statistics,brandingSettings',
            id=channel_id
        ).execute()

        channel_title = response['items'][0]['snippet']['title'] video_count =
        response['items'][0]['statistics']['videoCount'] channel_logo_url =
response['items'][0]['snippet']['thumbnails']['high']['url']
        channel_created_date =
response['items'][0]['snippet']['publishedAt']
        subscriber_count =
response['items'][0]['statistics']['subscriberCount']
        channel_description =
response['items'][0]['snippet']['description']


        channel_info = {
            'channel_title': channel_title,
            'video_count': video_count,
            'channel_logo_url': channel_logo_url,
            'channel_created_date': channel_created_date,
            'subscriber_count': subscriber_count,
            'channel_description': channel_description
        }

        return channel_info

    except HttpError as  error:
        print(f'An error occurred:  {error}')
        return None
```

Senti.py:

```
import csv
import re
import pandas as pd
import nltk
nltk.download('vader_lexicon')
from nltk.sentiment.vader import SentimentIntensityAnalyzer
import plotly.express as  px
import plotly.graph_objects as go
from colorama import Fore, Style
from typing import  Dict
import streamlit as  st

def extract_video_id(youtube_link):
    video_id_regex =
r"^(?:https?:\/\/)?(?:www\.)?(?:youtube\.com\/watch\?v=|youtu.be\/)([a- zA-Z0-
9_-]{11})"
    match = re.search(video_id_regex, youtube_link) if
    match:
        video_id = match.group( 1)
        return video_id
    else:
        return None

def analyze_sentiment(csv_file):
    # Initialize the sentiment analyzer sid
    = SentimentIntensityAnalyzer()

    # Read in the YouTube comments from the CSV file
    comments = []
    with open(csv_file, 'r', encoding='utf-8-sig') as csvfile:
        reader = csv.DictReader(csvfile)
        for row in reader:
            comments.append(row['Comment'])

    # Count the number of neutral, positive, and negative comments
    num_neutral =  0
    num_positive = 0
    num_negative =  0
    for comment in  comments:
        sentiment_scores = sid.polarity_scores(comment)
        if sentiment_scores['compound'] ==  0.0:
            num_neutral += 1
        elif sentiment_scores['compound'] > 0.0:
            num_positive += 1
        else:
            num_negative += 1

    # Return the results as a dictionary
    results = {'num_neutral': num_neutral, 'num_positive':
num_positive, 'num_negative': num_negative}
    return results

def bar_chart(csv_file: str) -> None:
    # Call analyze_sentiment function to get the results
    results: Dict[str, int] = analyze_sentiment(csv_file)

    # Get the counts for each sentiment category
    num_neutral = results['num_neutral']
    num_positive = results['num_positive']
    num_negative = results['num_negative']

    # Create a Pandas DataFrame with the results
    df = pd.DataFrame({
        'Sentiment': ['Positive', 'Negative', 'Neutral'],
        'Number of Comments': [num_positive, num_negative,
num_neutral]
    })

    # Create the bar chart using Plotly Express
    fig = px.bar(df, x='Sentiment', y='Number of Comments',
color='Sentiment',
                 color_discrete_sequence=['#87CEFA', '#FFA07A',
'#D3D3D3'],
                 title='Sentiment Analysis Results')
    fig.update_layout(title_font=dict(size=20))


    # Show the chart
    st.plotly_chart(fig, use_container_width=True)

def plot_sentiment(csv_file: str) -> None:
    # Call analyze_sentiment function to get the results
    results: Dict[str, int] = analyze_sentiment(csv_file)

    # Get the counts for each sentiment category
    num_neutral = results['num_neutral']
    num_positive = results['num_positive']
    num_negative = results['num_negative']

    # Plot the pie  chart
    labels = ['Neutral', 'Positive', 'Negative']
    values = [num_neutral, num_positive, num_negative]
    colors = ['yellow', 'green', 'red']
    fig = go.Figure(data=[go.Pie(labels=labels, values=values,
textinfo='label+percent',
                                marker=dict(colors=colors))])
    fig.update_layout(title={'text': 'Sentiment Analysis Results', 'font':
{'size': 20, 'family': 'Arial', 'color': 'grey'},
                             'x': 0.5, 'y':  0.9},
                      font=dict(size=14))
    st.plotly_chart(fig)



def create_scatterplot(csv_file: str, x_column: str, y_column: str) -> None:
    # Load data from  CSV
    data = pd.read_csv(csv_file)

    # Create scatter plot using Plotly
    fig = px.scatter(data, x=x_column, y=y_column,  color='Category')

    # Customize layout
    fig.update_layout(
        title='Scatter Plot',
        xaxis_title=x_column,
        yaxis_title=y_column,
        font=dict(size=18)
    )

    # Display plot in  Streamlit
    st.plotly_chart(fig, use_container_width=True)



def print_sentiment(csv_file: str) -> None:
    # Call analyze_sentiment function to get the results
    results: Dict[str, int] = analyze_sentiment(csv_file)
    # Get the counts for each sentiment category
    num_neutral = results['num_neutral']
    num_positive = results['num_positive']
    num_negative = results['num_negative']


    # Determine the overall sentiment if
    num_positive > num_negative:
        overall_sentiment = 'POSITIVE'
        color = Fore.GREEN
    elif num_negative > num_positive:
        overall_sentiment = 'NEGATIVE'
        color = Fore.RED
    else:
        overall_sentiment = 'NEUTRAL'
        color = Fore.YELLOW

    # Print the overall sentiment in color
    print('\n'+ Style.BRIGHT+ color +
overall_sentiment.upper().center( 50, ' ') + Style.RESET_ALL)
```

Appendices

In the context of YouTube comment sentiment analysis, appendices can contain various types of information, such as source code, sentiment analysis results, and other relevant data. Here are some snippets from search results that provide insights into the content of appendices related to YouTube comment sentiment analysis:

*mentions: "See Appendix A for the source code that we used. 3.1 Data Gathering. Labeled data (positive/negative/neutral) was needed to train our classifiers us-."

*states: "The ten terms associating most strongly with positive sentiment (see online appendix) overall were: Please; nice; wow; beautiful..."

*provides: "Appendix 4 provides a..."

*discusses: "Sentiment analysis is a trending topic among YouTube video comments studies.....Appendix 1).

These snippets indicate that appendices in YouTube comment sentiment analysis can include source code, sentiment analysis results, and other relevant data such as terms associated with positive sentiment and individual sentiment scores for specific entities.

If you need more specific information or details about a particular aspect of YouTube comment sentiment analysis, feel free to ask!