asg4-aniket-mittal-m24ds001

October 2, 2024

[2]:
```
!pip install selenium
```

```
Collecting selenium
  Downloading selenium-4.25.0-py3-none-any.whl.metadata (7.1 kB)
Requirement already satisfied: urllib3<3,>=1.26 in
/usr/local/lib/python3.10/dist-packages (from urllib3[socks]<3,>=1.26->selenium)
(2.2.3)
Collecting trio~=0.17 (from selenium)
  Downloading trio-0.26.2-py3-none-any.whl.metadata (8.6 kB)
Collecting trio-websocket~=0.9 (from selenium)
  Downloading trio_websocket-0.11.1-py3-none-any.whl.metadata (4.7 kB)
Requirement already satisfied: certifi>=2021.10.8 in
/usr/local/lib/python3.10/dist-packages (from selenium) (2024.8.30)
Requirement already satisfied: typing_extensions~=4.9 in
/usr/local/lib/python3.10/dist-packages (from selenium) (4.12.2)
Requirement already satisfied: websocket-client~=1.8 in
/usr/local/lib/python3.10/dist-packages (from selenium) (1.8.0)
Requirement already satisfied: attrs>=23.2.0 in /usr/local/lib/python3.10/dist-
packages (from trio~=0.17->selenium) (24.2.0)
Requirement already satisfied: sortedcontainers in
/usr/local/lib/python3.10/dist-packages (from trio~=0.17->selenium) (2.4.0)
Requirement already satisfied: idna in /usr/local/lib/python3.10/dist-packages
(from trio~=0.17->selenium) (3.10)
Collecting outcome (from trio~=0.17->selenium)
  Downloading outcome-1.3.0.post0-py2.py3-none-any.whl.metadata (2.6 kB)
Requirement already satisfied: sniffio>=1.3.0 in /usr/local/lib/python3.10/dist-
packages (from trio~=0.17->selenium) (1.3.1)
Requirement already satisfied: exceptiongroup in /usr/local/lib/python3.10/dist-
packages (from trio~=0.17->selenium) (1.2.2)
Collecting wsproto>=0.14 (from trio-websocket~=0.9->selenium)
  Downloading wsproto-1.2.0-py3-none-any.whl.metadata (5.6 kB)
Requirement already satisfied: pysocks!=1.5.7,<2.0,>=1.5.6 in
/usr/local/lib/python3.10/dist-packages (from urllib3[socks]<3,>=1.26->selenium)
(1.7.1)
Collecting h11<1,>=0.9.0 (from wsproto>=0.14->trio-websocket~=0.9->selenium)
  Downloading h11-0.14.0-py3-none-any.whl.metadata (8.2 kB)
Downloading selenium-4.25.0-py3-none-any.whl (9.7 MB)
                        9.7/9.7 MB
```

```
32.1 MB/s eta 0:00:00
Downloading trio-0.26.2-py3-none-any.whl (475 kB)
                           476.0/476.0 kB
27.5 MB/s eta 0:00:00
Downloading trio_websocket-0.11.1-py3-none-any.whl (17 kB)
Downloading wsproto-1.2.0-py3-none-any.whl (24 kB)
Downloading outcome-1.3.0.post0-py2.py3-none-any.whl (10 kB)
Downloading h11-0.14.0-py3-none-any.whl (58 kB)
                           58.3/58.3 kB
3.7 MB/s eta 0:00:00
Installing collected packages: outcome, h11, wsproto, trio, trio-
websocket, selenium
Successfully installed h11-0.14.0 outcome-1.3.0.post0 selenium-4.25.0
trio-0.26.2 trio-websocket-0.11.1 wsproto-1.2.0
```

Q1)Data Preparation

[5]:
```
!pip install webdriver-manager
```

```
Collecting webdriver-manager
  Downloading webdriver_manager-4.0.2-py2.py3-none-any.whl.metadata (12 kB)
Requirement already satisfied: requests in /usr/local/lib/python3.10/dist-
packages (from webdriver-manager) (2.32.3)
Collecting python-dotenv (from webdriver-manager)
  Downloading python_dotenv-1.0.1-py3-none-any.whl.metadata (23 kB)
Requirement already satisfied: packaging in /usr/local/lib/python3.10/dist-
packages (from webdriver-manager) (24.1)
Requirement already satisfied: charset-normalizer<4,>=2 in
/usr/local/lib/python3.10/dist-packages (from requests->webdriver-manager)
(3.3.2)
Requirement already satisfied: idna<4,>=2.5 in /usr/local/lib/python3.10/dist-
packages (from requests->webdriver-manager) (3.10)
Requirement already satisfied: urllib3<3,>=1.21.1 in
/usr/local/lib/python3.10/dist-packages (from requests->webdriver-manager)
(2.2.3)
Requirement already satisfied: certifi>=2017.4.17 in
/usr/local/lib/python3.10/dist-packages (from requests->webdriver-manager)
(2024.8.30)
Downloading webdriver_manager-4.0.2-py2.py3-none-any.whl (27 kB)
Downloading python_dotenv-1.0.1-py3-none-any.whl (19 kB)
Installing collected packages: python-dotenv, webdriver-manager
Successfully installed python-dotenv-1.0.1 webdriver-manager-4.0.2
```

[ ]:
```
from selenium import webdriver
from selenium.webdriver.common.by import By
from webdriver_manager.chrome import ChromeDriverManager
import time
import pandas as pd
```

[ ]:
```
def scroll_to_load_comments(driver):
    last_height = driver.execute_script("return document.documentElement.
 ↪scrollHeight")

    while True:
        # Scroll down to the bottom of the page
        driver.execute_script("window.scrollTo(0, document.documentElement.
 ↪scrollHeight);")

        # Wait for comments to load
        time.sleep(3)

        # Calculate new scroll height and compare with the last scroll height
        new_height = driver.execute_script("return document.documentElement.
 ↪scrollHeight")
        if new_height == last_height:
            break
        last_height = new_height

# Function to scrape comments
def scrape_youtube_comments(video_url):
    # Automatically install and use the appropriate ChromeDriver
    driver = webdriver.Chrome(ChromeDriverManager().install())
    driver.get(video_url)
    time.sleep(5)  # Wait for the page to load

    # Scroll down to load comments
    scroll_to_load_comments(driver)

    # Extract comments
    comments_section = driver.find_element(By.CSS_SELECTOR, '#contents␣
 ↪#contents #contents')
    comment_elements = comments_section.find_elements(By.CSS_SELECTOR,␣
 ↪'#content #content-text')

    comments = [comment.text.strip() for comment in comment_elements]

    driver.quit()  # Close the WebDriver
    return comments

# Function to save comments to CSV
def save_comments_to_csv(comments, filename):
    df = pd.DataFrame(comments, columns=["Comment"])
    df.to_csv(filename, index=False)
```

[ ]:
```
video_url = "https://youtu.be/QVKj3LADCnA?si=t0XprixE7NMVV461"
comments = scrape_youtube_comments(video_url)
save_comments_to_csv(comments, "Comments_youtube.csv")
```

[137]:
```
df = pd.read_csv("content/Comments_youtube.csv")
```

[138]:
```
df.drop_duplicates(inplace=True)
df.reset_index(drop=True, inplace=True)
df.shape
```

[138]:
```
(695, 1)
```

[140]:
```
import nltk
nltk.download('stopwords')
nltk.download('punkt')
from nltk.corpus import stopwords
from nltk.tokenize import word_tokenize
```

```
[nltk_data] Downloading package stopwords to /root/nltk_data…
[nltk_data]   Package stopwords is already up-to-date!
[nltk_data] Downloading package punkt to /root/nltk_data…
[nltk_data]   Package punkt is already up-to-date!
```

[141]:
```
stop_words = set(stopwords.words('english'))
negative_words = ['not', 'no', 'never', 'nor', 'neither']
new_stop_words = [word for word in stop_words if word not in negative_words]
def preprocess_text(text):
    if isinstance(text, (int, float)):
        return text
    text = str(text)
    tokens = word_tokenize(text)
    tokens = [word.lower() for word in tokens if word.isalpha()]
    tokens = [word for word in tokens if word not in new_stop_words]
    return " ".join(tokens)
```

[142]:
```
df['preprocessed_comment'] = df['Comment'].apply(preprocess_text)
df = df[df['preprocessed_comment'].str.strip().astype(bool)]
```

[143]:
```
df
```

[143]:

	Comment \
0	00:00 Lecture overview\n02:40 Elimination Succ…
1	I like this professor so much, listening his c…
2	there's something about his lectures that make…
3	This guy is amazing…When teachers taught thi…
4	This kind of lecture is only possible when you…
..	…

```
690                                          Goat !!!
691  As a teacher myself I am always shocked to see…
692  you are not a clear teacher because you not cl…
693                                       I hate math.
694   Not clear, lengthy, anticipate too much concepts

                                  preprocessed_comment
0    lecture overview elimination success eliminati…
1       like professor much listening course amusement
2    something lectures make easy concentrate follo…
3    guy amazing teachers taught college everything…
4    kind lecture possible deep passion work huge t…
..                                                 …
690                                            goat
691  teacher always shocked see lack participation …
692                  not clear teacher not clear step
693                                        hate math
694        not clear lengthy anticipate much concepts

[681 rows x 2 columns]
```

[144]:
```
df.dropna(inplace=True)
```

```
<ipython-input-144-c64f9f573c18>:1: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame

See the caveats in the documentation: https://pandas.pydata.org/pandas-
docs/stable/user_guide/indexing.html#returning-a-view-versus-a-copy
  df.dropna(inplace=True)
```

[ ]:
```

```

Q2) Sentiment Classification

[58]:
```
from nltk.sentiment import SentimentIntensityAnalyzer
nltk.download('vader_lexicon')
```

```
[nltk_data] Downloading package vader_lexicon to /root/nltk_data…
[nltk_data]   Package vader_lexicon is already up-to-date!
```

[58]:
```
True
```

[59]:
```
sia = SentimentIntensityAnalyzer()
```

[145]:
```
import pandas as pd
from nltk.sentiment import SentimentIntensityAnalyzer
import re
```

```
# Initialize VADER sentiment analyzer
sia = SentimentIntensityAnalyzer()

# Basic Sentiment Classification using VADER
def basic_sentiment_classification(text):
    score = sia.polarity_scores(text)

    if score['compound'] >= 0.05:
        return 'Positive'
    elif score['compound'] <= -0.05:
        return 'Negative'
    else:
        return 'Neutral'

# Fine-Grained Sentiment Classification
def fine_grained_sentiment_classification(text):
    score = sia.polarity_scores(text)

    # Detecting questions or inquiry
    if is_question(text):
        return 'Question/Inquiry'

    if score['compound'] >= 0.75:
        return 'Strong Positive'
    elif 0.05 < score['compound'] < 0.75:
        return 'General Positive'
    elif -0.75 < score['compound'] <= -0.05:
        return 'General Negative'
    elif score['compound'] <= -0.75:
        return 'Strong Negative'
    else:
        return 'Neutral'


def is_question(text):

    return bool(re.search(r'\?
 ↪|\b(who|what|when|where|why|how|which|can|could|would|should)\b', text.
 ↪lower()))


df['Basic Sentiment'] = df['preprocessed_comment'].
 ↪apply(basic_sentiment_classification)
df['Fine-Grained Sentiment'] = df['preprocessed_comment'].
 ↪apply(fine_grained_sentiment_classification)
df['is_question/'] = df['preprocessed_comment'].apply(is_question)
# Display the updated DataFrame
```

```
print(df)

# Optionally, save to a CSV file
df.to_csv('classified_comments.csv', index=False)
```

```
                                                Comment  \
0    00:00 Lecture overview\n02:40 Elimination Succ…
1    I like this professor so much, listening his c…
2    there's something about his lectures that make…
3    This guy is amazing…When teachers taught thi…
4    This kind of lecture is only possible when you…
..                                                 …
690                                          Goat !!!
691  As a teacher myself I am always shocked to see…
692  you are not a clear teacher because you not cl…
693                                       I hate math.
694   Not clear, lengthy, anticipate too much concepts

                                  preprocessed_comment Basic Sentiment  \
0    lecture overview elimination success eliminati…        Positive
1       like professor much listening course amusement        Positive
2    something lectures make easy concentrate follo…        Positive
3    guy amazing teachers taught college everything…        Positive
4    kind lecture possible deep passion work huge t…        Positive
..                                                 …             …
690                                           goat          Neutral
691  teacher always shocked see lack participation …        Negative
692                  not clear teacher not clear step        Negative
693                                        hate math        Negative
694        not clear lengthy anticipate much concepts        Negative

    Fine-Grained Sentiment  is_question/
0         General Positive         False
1         General Positive         False
2          Strong Positive         False
3         General Positive         False
4          Strong Positive         False
..                     …            …
690               Neutral         False
691        Strong Negative         False
692       General Negative         False
693       General Negative         False
694       General Negative         False

[680 rows x 5 columns]
```

```
<ipython-input-145-252377ac8159>:44: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
```

```
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-
docs/stable/user_guide/indexing.html#returning-a-view-versus-a-copy
  df['Basic Sentiment'] =
df['preprocessed_comment'].apply(basic_sentiment_classification)
<ipython-input-145-252377ac8159>:45: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-
docs/stable/user_guide/indexing.html#returning-a-view-versus-a-copy
  df['Fine-Grained Sentiment'] =
df['preprocessed_comment'].apply(fine_grained_sentiment_classification)
<ipython-input-145-252377ac8159>:46: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-
docs/stable/user_guide/indexing.html#returning-a-view-versus-a-copy
  df['is_question/'] = df['preprocessed_comment'].apply(is_question)
```

[146]:
```
df
```

[146]:

	Comment	preprocessed_comment	Basic Sentiment	
0	00:00 Lecture overview\n02:40 Elimination Succ…	lecture overview elimination success eliminati…	Positive	\
1	I like this professor so much, listening his c…	like professor much listening course amusement	Positive	\
2	there's something about his lectures that make…	something lectures make easy concentrate follo…	Positive	\
3	This guy is amazing…When teachers taught thi…	guy amazing teachers taught college everything…	Positive	\
4	This kind of lecture is only possible when you…	kind lecture possible deep passion work huge t…	Positive	\
..	…	…	…	…
690	Goat !!!	goat	Neutral	\
691	As a teacher myself I am always shocked to see…	teacher always shocked see lack participation …	Negative	\
692	you are not a clear teacher because you not cl…	not clear teacher not clear step	Negative	\
693	I hate math.			
694	Not clear, lengthy, anticipate too much concepts			

```
693                                          hate math        Negative
694          not clear lengthy anticipate much concepts        Negative

    Fine-Grained Sentiment  is_question/
0          General Positive         False
1          General Positive         False
2           Strong Positive         False
3          General Positive         False
4           Strong Positive         False
..                      …           …
690                Neutral         False
691         Strong Negative         False
692        General Negative         False
693        General Negative         False
694        General Negative         False

[680 rows x 5 columns]
```

Q3)Additional Sentiment Classification

[148]:
```
from transformers import pipeline
```

[149]:
```
# Initialize sarcasm classifier
sarcasm_classifier = pipeline("text-classification", model="helinivan/
 ↪english-sarcasm-detector")

# Function to analyze emotional vs. objective and subjective vs. objective
def analyze_sentiment(row):
    text = row['preprocessed_comment']

    # Sarcasm Detection
    sarcasm_prediction = sarcasm_classifier(text)[0]
    is_sarcastic = sarcasm_prediction['label'] == 'sarcastic'

    # Emotional vs. Objective
    sentiment_score = TextBlob(text).sentiment
    emotional = 'emotional' if abs(sentiment_score.polarity) > 0.5 else␣
 ↪'objective'

    # Subjective vs. Objective
    subjectivity = 'subjective' if sentiment_score.subjectivity > 0.5 else␣
 ↪'objective'

    return pd.Series({
        'is_sarcastic': is_sarcastic,
        'emotional_objective': emotional,
        'subjective_objective': subjectivity
```

```
    })
    df[['is_sarcastic', 'emotional_objective', 'subjective_objective']] = df.
 ↪apply(analyze_sentiment, axis=1)

# Display the results
print(df)
```

```
                                                Comment  \
0    00:00 Lecture overview\n02:40 Elimination Succ…
1    I like this professor so much, listening his c…
2    there's something about his lectures that make…
3    This guy is amazing…When teachers taught thi…
4    This kind of lecture is only possible when you…
..                                                 …
690                                        Goat !!!
691  As a teacher myself I am always shocked to see…
692  you are not a clear teacher because you not cl…
693                                      I hate math.
694   Not clear, lengthy, anticipate too much concepts

                                preprocessed_comment Basic Sentiment  \
0    lecture overview elimination success eliminati…        Positive
1       like professor much listening course amusement        Positive
2    something lectures make easy concentrate follo…        Positive
3    guy amazing teachers taught college everything…        Positive
4    kind lecture possible deep passion work huge t…        Positive
..                                                 …             …
690                                           goat         Neutral
691  teacher always shocked see lack participation …        Negative
692                  not clear teacher not clear step        Negative
693                                        hate math        Negative
694        not clear lengthy anticipate much concepts        Negative

    Fine-Grained Sentiment  is_question/
0          General Positive         False
1          General Positive         False
2           Strong Positive         False
3          General Positive         False
4           Strong Positive         False
..                      …             …
690                Neutral         False
691         Strong Negative         False
692        General Negative         False
693        General Negative         False
694        General Negative         False

[680 rows x 5 columns]
```

```
/usr/local/lib/python3.10/dist-
packages/transformers/tokenization_utils_base.py:1601: FutureWarning:
`clean_up_tokenization_spaces` was not set. It will be set to `True` by default.
This behavior will be depracted in transformers v4.45, and will be then set to
`False` by default. For more details check this issue:
https://github.com/huggingface/transformers/issues/31884
  warnings.warn(
```

[150]:
```
df
```

[150]:
```
                                                Comment  \
0    00:00 Lecture overview\n02:40 Elimination Succ…
1    I like this professor so much, listening his c…
2    there's something about his lectures that make…
3    This guy is amazing…When teachers taught thi…
4    This kind of lecture is only possible when you…
..                                                 …
690                                          Goat !!!
691  As a teacher myself I am always shocked to see…
692  you are not a clear teacher because you not cl…
693                                      I hate math.
694   Not clear, lengthy, anticipate too much concepts

                                preprocessed_comment Basic Sentiment  \
0    lecture overview elimination success eliminati…        Positive
1       like professor much listening course amusement        Positive
2    something lectures make easy concentrate follo…        Positive
3    guy amazing teachers taught college everything…        Positive
4    kind lecture possible deep passion work huge t…        Positive
..                                                 …             …
690                                             goat         Neutral
691  teacher always shocked see lack participation …        Negative
692                  not clear teacher not clear step        Negative
693                                        hate math        Negative
694        not clear lengthy anticipate much concepts        Negative

    Fine-Grained Sentiment  is_question/
0         General Positive         False
1         General Positive         False
2          Strong Positive         False
3         General Positive         False
4          Strong Positive         False
..                     …            …
690              Neutral         False
691        Strong Negative         False
692       General Negative         False
693       General Negative         False
```

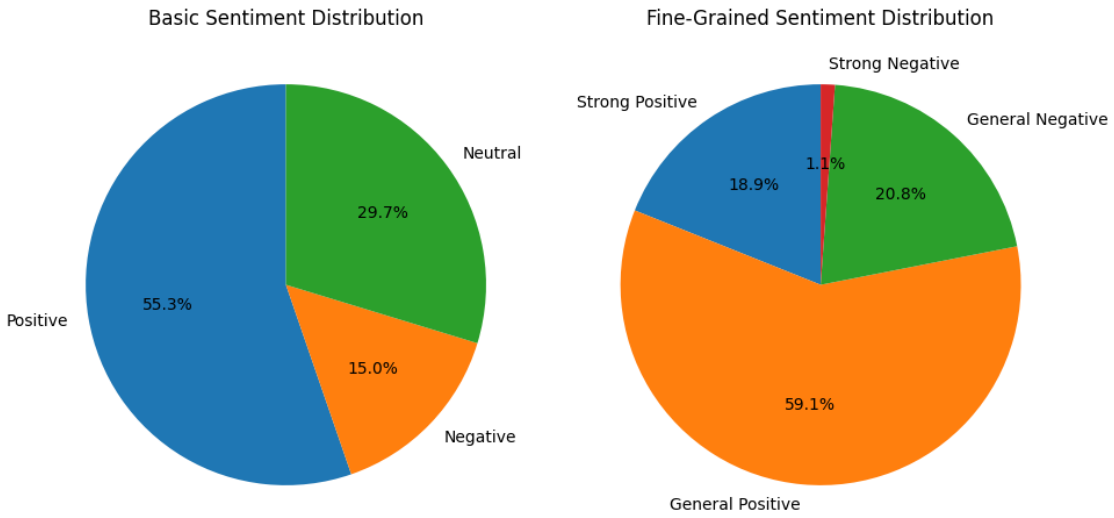
```
694       General Negative         False

[680 rows x 5 columns]
```

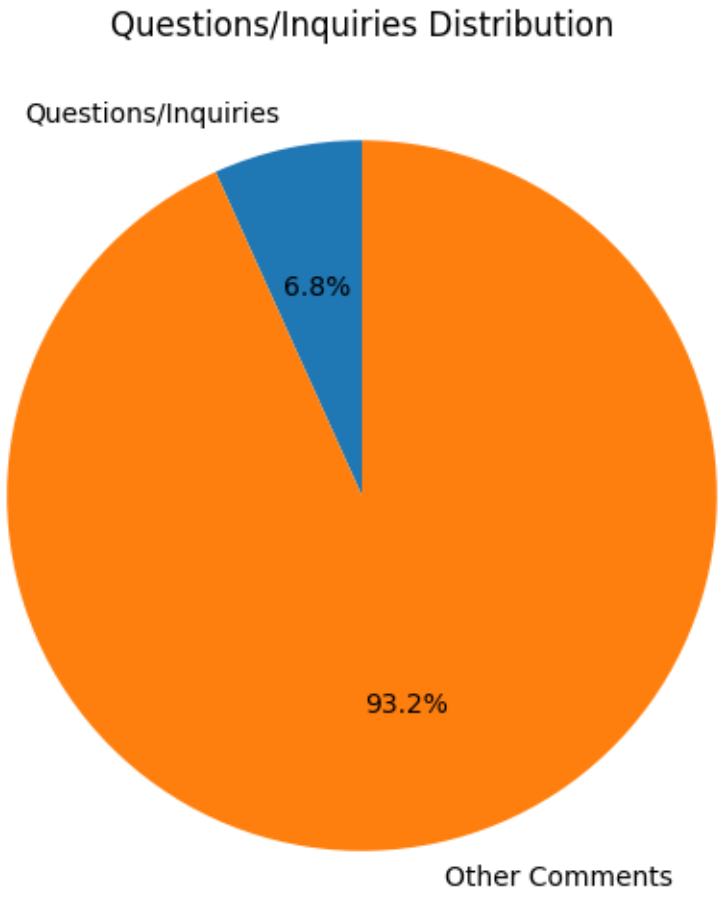
Q4) Sentiment Analysis Metrices

[151]:
```python
def sentiment_statistics(df):
    # Basic sentiment percentages
    total_comments = len(df)
    positive_count = df['Basic Sentiment'].value_counts().get('Positive', 0)
    negative_count = df['Basic Sentiment'].value_counts().get('Negative', 0)
    neutral_count = df['Basic Sentiment'].value_counts().get('Neutral', 0)

    positive_percentage = (positive_count / total_comments) * 100
    negative_percentage = (negative_count / total_comments) * 100
    neutral_percentage = (neutral_count / total_comments) * 100

    # Fine-grained sentiment percentages
    strong_positive_count = df['Fine-Grained Sentiment'].value_counts().
↪get('Strong Positive', 0)
    general_positive_count = df['Fine-Grained Sentiment'].value_counts().
↪get('General Positive', 0)
    general_negative_count = df['Fine-Grained Sentiment'].value_counts().
↪get('General Negative', 0)
    strong_negative_count = df['Fine-Grained Sentiment'].value_counts().
↪get('Strong Negative', 0)
    question_inquiry_count = df['Fine-Grained Sentiment'].value_counts().
↪get('Question/Inquiry', 0)

    # Print the statistics
    print(f"Percentage of Positive Comments: {positive_percentage:.2f}%")
    print(f"Percentage of Negative Comments: {negative_percentage:.2f}%")
    print(f"Percentage of Neutral Comments: {neutral_percentage:.2f}%")
    print(f"Percentage of Strong Positive Comments: {(strong_positive_count /␣
↪total_comments) * 100:.2f}%")
    print(f"Percentage of General Positive Comments: {(general_positive_count /␣
↪total_comments) * 100:.2f}%")
    print(f"Percentage of General Negative Comments: {(general_negative_count /␣
↪total_comments) * 100:.2f}%")
    print(f"Percentage of Strong Negative Comments: {(strong_negative_count /␣
↪total_comments) * 100:.2f}%")
    print(f"Percentage of Questions/Inquiries: {(question_inquiry_count /␣
↪total_comments) * 100:.2f}%")

    # Pie charts for sentiment distribution
    basic_sentiment_counts = [positive_count, negative_count, neutral_count]
```

```
    fine_grained_counts = [
        strong_positive_count,
        general_positive_count,
        general_negative_count,
        strong_negative_count
    ]

    # Basic Sentiment Pie Chart
    plt.figure(figsize=(10, 5))

    plt.subplot(1, 2, 1)
    plt.pie(basic_sentiment_counts, labels=['Positive', 'Negative', 'Neutral'],␣
↪autopct='%1.1f%%', startangle=90)
    plt.title('Basic Sentiment Distribution')

    # Fine-Grained Sentiment Pie Chart
    plt.subplot(1, 2, 2)
    plt.pie(fine_grained_counts, labels=['Strong Positive', 'General Positive',␣
↪'General Negative', 'Strong Negative'], autopct='%1.1f%%', startangle=90)
    plt.title('Fine-Grained Sentiment Distribution')

    plt.tight_layout()
    plt.show()

    # Pie chart for Questions/Inquiries
    plt.figure(figsize=(6, 6))
    inquiry_counts = [question_inquiry_count, total_comments -␣
↪question_inquiry_count]
    plt.pie(inquiry_counts, labels=['Questions/Inquiries', 'Other Comments'],␣
↪autopct='%1.1f%%', startangle=90)
    plt.title('Questions/Inquiries Distribution')
    plt.show()

# Call the function to get sentiment statistics and visualize results
sentiment_statistics(df)

# Optionally, save the updated DataFrame to a CSV file
df.to_csv('classified_comments.csv', index=False)
```

```
Percentage of Positive Comments: 55.29%
Percentage of Negative Comments: 15.00%
Percentage of Neutral Comments: 29.71%
Percentage of Strong Positive Comments: 12.21%
Percentage of General Positive Comments: 38.09%
Percentage of General Negative Comments: 13.38%
Percentage of Strong Negative Comments: 0.74%
Percentage of Questions/Inquiries: 6.76%
```


Basic Sentiment Distribution
Neutral
29.7%
55.3%
Positive
15.0%
Negative
Fine-Grained Sentiment Distribution
Strong Negative
Strong Positive
General Negative
1.1%
18.9%
20.8%
59.1%
General Positive


Questions/Inquiries Distribution
Questions/Inquiries
6.8%
93.2%
Other Comments

[152]:
```
from wordcloud import WordCloud
from sklearn.feature_extraction.text import CountVectorizer
```

[153]:
```
import matplotlib.pyplot as plt
%matplotlib inline
```

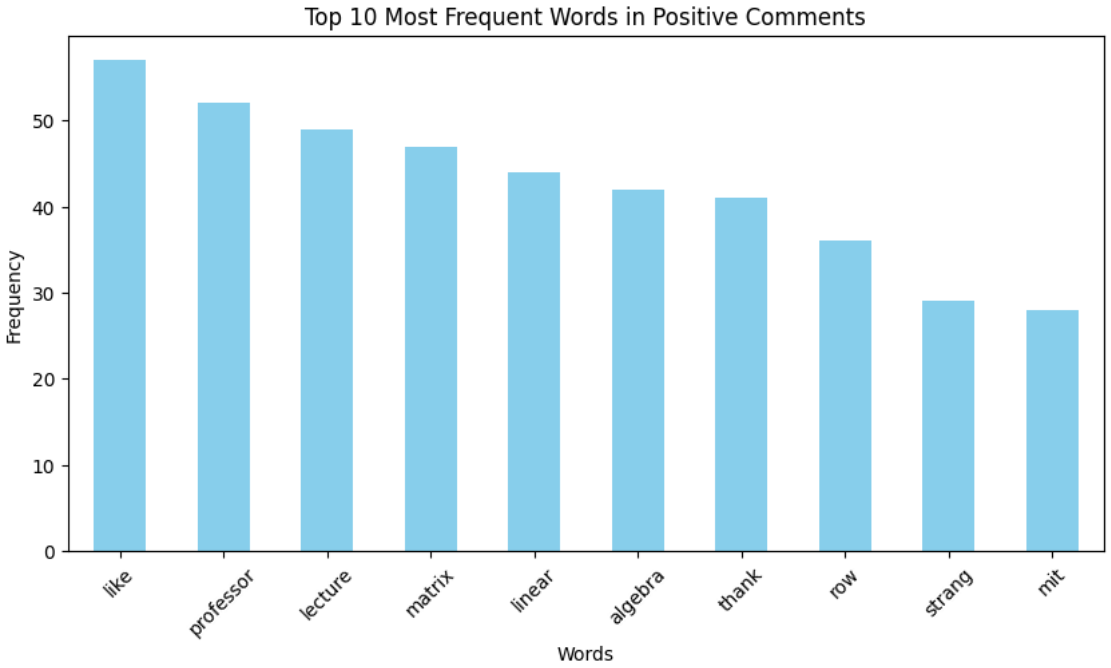
[155]:
```
def analyze_word_frequency(preprocessed_comments, title):
    vectorizer = CountVectorizer(stop_words='english')
    word_count = vectorizer.fit_transform(preprocessed_comments)
    word_freq = pd.DataFrame(word_count.toarray(), columns=vectorizer.
 ↪get_feature_names_out())
    total_word_freq = word_freq.sum().sort_values(ascending=False)

    print(f"\nMost Frequent Words in {title}:")
    print(total_word_freq.head(10))

    total_word_freq.head(10).plot(kind='bar', figsize=(10, 5), color='skyblue')
    plt.title(f"Top 10 Most Frequent Words in {title}")
    plt.xlabel('Words')
    plt.ylabel('Frequency')
    plt.xticks(rotation=45)
    plt.show()
```

[156]:
```
positive_comments = df[df['Basic Sentiment'] ==␣
 ↪'Positive']['preprocessed_comment']
negative_comments = df[df['Basic Sentiment'] ==␣
 ↪'Negative']['preprocessed_comment']
neutral_comments = df[df['Basic Sentiment'] ==␣
 ↪'Neutral']['preprocessed_comment']
positive_text = " ".join(positive_comments)
negative_text = " ".join(negative_comments)
neutral_text = " ".join(neutral_comments)
```

[160]:
```
analyze_word_frequency(positive_comments, "Positive Comments")
```

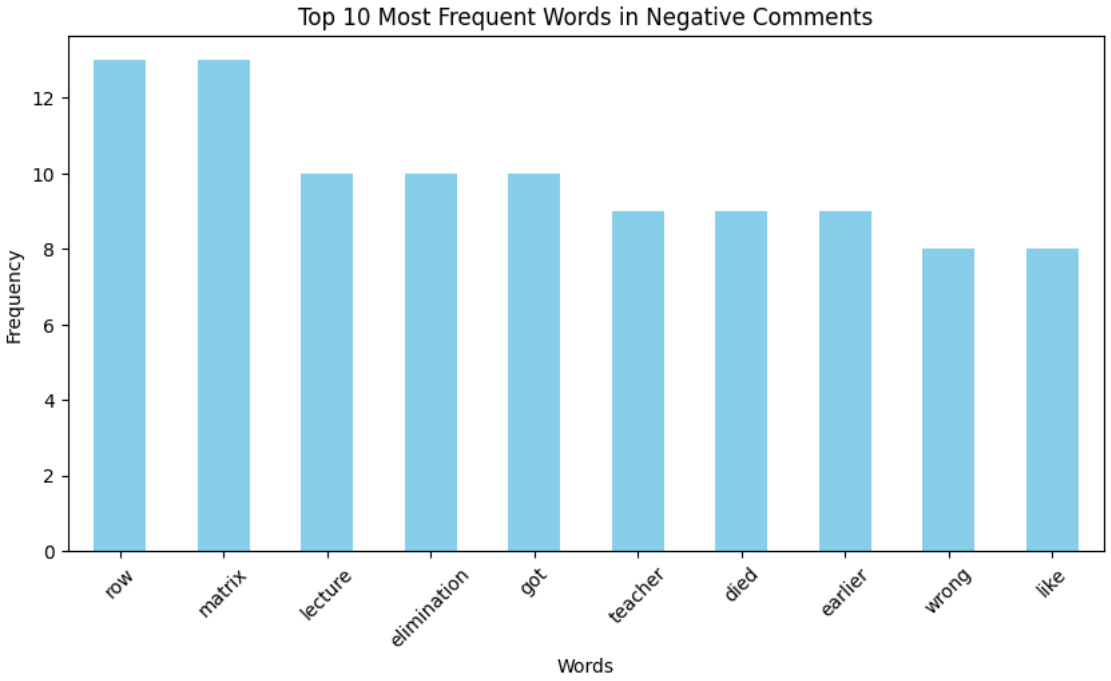
```
Most Frequent Words in Positive Comments:
like         57
professor    52
lecture      49
matrix       47
linear       44
algebra      42
thank        41
row          36
strang       29
mit          28
```

```
dtype: int64
```



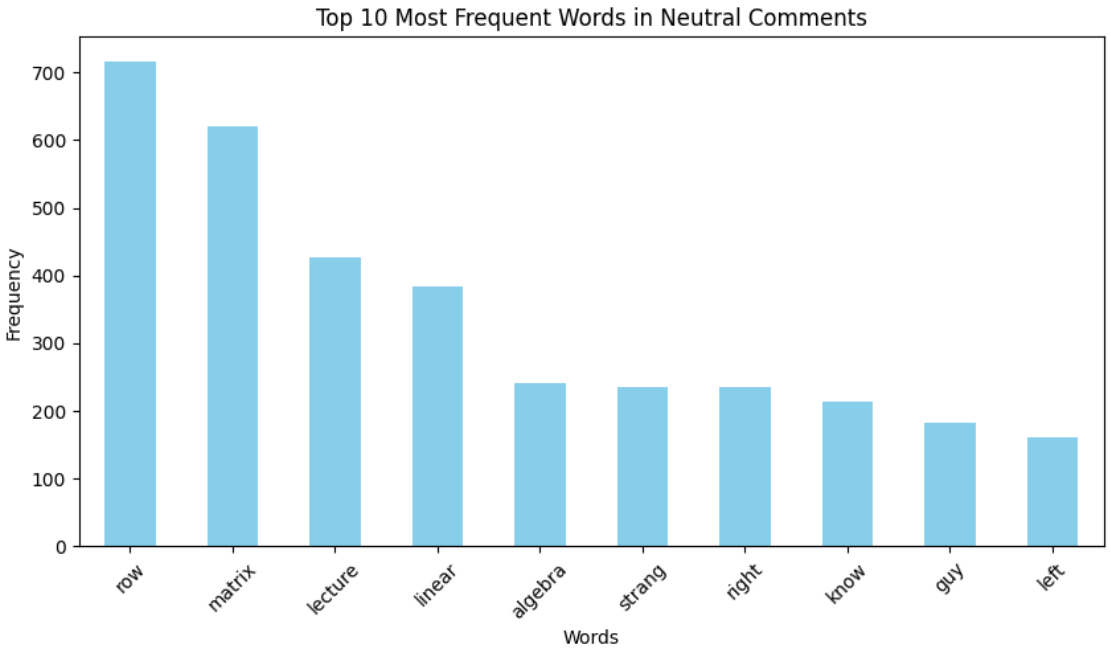
[163]:
```
analyze_word_frequency(negative_comments, "Negative Comments")
```

```
Most Frequent Words in Negative Comments:
row            13
matrix         13
lecture        10
elimination    10
got            10
teacher         9
died            9
earlier         9
wrong           8
like            8
dtype: int64
```



[76]:
```
analyze_word_frequency(neutral_comments, "Neutral Comments")
```

```
Most Frequent Words in Neutral Comments:
row         717
matrix      621
lecture     426
linear      384
algebra     241
strang      236
right       235
know        213
guy         182
left        161
dtype: int64
```



Q5)Summarization

[109]:
```
from transformers import pipeline
```

[110]:
```
import torch
device = torch.device("cuda" if torch.cuda.is_available() else "cpu")
print(f"Using device: {device}")
```

```
Using device: cpu
```

[111]:
```
summarizer = pipeline("summarization", model="facebook/bart-large-cnn",␣
↪device=0 if torch.cuda.is_available() else -1)
```

```
config.json:   0%|          | 0.00/1.58k [00:00<?, ?B/s]

model.safetensors:   0%|          | 0.00/1.63G [00:00<?, ?B/s]

generation_config.json:   0%|          | 0.00/363 [00:00<?, ?B/s]

vocab.json:   0%|          | 0.00/899k [00:00<?, ?B/s]

merges.txt:   0%|          | 0.00/456k [00:00<?, ?B/s]

tokenizer.json:   0%|          | 0.00/1.36M [00:00<?, ?B/s]

/usr/local/lib/python3.10/dist-
packages/transformers/tokenization_utils_base.py:1601: FutureWarning:
`clean_up_tokenization_spaces` was not set. It will be set to `True` by default.
This behavior will be depracted in transformers v4.45, and will be then set to
```

```
`False` by default. For more details check this issue:
https://github.com/huggingface/transformers/issues/31884
  warnings.warn(
```

[174]:
```python
def chunk_text(text, max_chunk_size=6000):
    return [text[i:i + max_chunk_size] for i in range(0, len(text),␣
 ↪max_chunk_size)]

def summarize_chunks(text_chunks):
    summarized_chunks = []


    for chunk in text_chunks:
        summary = summarizer(chunk, max_length=200, min_length=30,␣
 ↪do_sample=False)[0]['summary_text']
        summarized_chunks.append(summary)
    return " ".join(summarized_chunks)

def summarize_long_text(text):
    if len(text) <= 6000:
        return summarizer(text, max_length=200, min_length=30,␣
 ↪do_sample=False)[0]['summary_text']
    text_chunks = chunk_text(text)
    final_summary = summarize_long_text(text_chunks)
    return final_summary
```

[175]:
```python
def summarize_positive_comments(df):
    positive_comments = df[df['Basic Sentiment'] ==␣
 ↪"Positive"]['preprocessed_comment'].tolist()
    if not positive_comments:
        return "No positive comments available for summarization."

    positive_text = " ".join(positive_comments)
    return summarize_long_text(positive_text)
```

[176]:
```python
def summarize_negative_comments(df):
    negative_comments = df[df['Basic Sentiment'] ==␣
 ↪"Negative"]['preprocessed_comment'].tolist()
    if not negative_comments:
        return "No negative comments available for summarization."

    negative_text = " ".join(negative_comments)
    return summarize_long_text(negative_text)
```

[177]:
```python
def summarize_overall_comments(df):
    all_comments = df['preprocessed_comment'].tolist()
    overall_summary_text = summarize_long_text(" ".join(all_comments))
```

```
    total_comments = len(df)
    positive_count = len(df[df['basic_sentiment'] == 'Positive'])
    negative_count = len(df[df['basic_sentiment'] == 'Negative'])

    positive_percentage = (positive_count / total_comments) * 100
    negative_percentage = (negative_count / total_comments) * 100
    neutral_percentage = 100 - (positive_percentage + negative_percentage)

    majority_sentiment = "Positive" if positive_count > negative_count else␣
↪"Negative"

    overall_summary = {
        'majority_sentiment': majority_sentiment,
        'common_themes': {
            'positive': positive_percentage,
            'negative': negative_percentage,
            'neutral': neutral_percentage
        },
        'overall_summary_text': overall_summary_text
    }
    return overall_summary
```

[3]:
```
positive_summary = summarize_positive_comments(df)
positive_summary
```

'Every university linear algebra time begining everyone high school promise within month study engineering bit humbled introduction class man made lecture entertaining awesome wish knew lecture earlier professor like wanted learn math unlike classes meant eliminate many students going forward possible excellent lecture excellent instructor excellent courses comfort home.'

[ ]:
```
negative_summary = summarize_negative_comments(df)
negative_summary
```

'Teachers college knew lectures would used learn lecture instead useless professors could hardly understand hey closed caption disappear word showed continue appear next word change audio video not sync either many empty chairs would die opportunity listen video live got bit confused not understand point made maybe leave author prove author textbook using though hah lecture video timeline links lecture elimination pivots. way teaching linear algebra taking example solving linear equations not throwing formulas without incentive really great dumb mit lectures much simpler easy understand professor sjsu conundrum.'

[ ]:
```
overall_summary = summarize_overall_comments(df)
overall_summary
```

'{'majority_sentiment': 'Positive', 'common_themes': {'positive': 55.294117647058826, 'negative': 15.0, 'neutral': 29.705882352941174}, 'overall_summary_text': 'Every university linear algebra time begining everyone high school promise within month study engineering bit humbled introduc-

tion class man made lecture entertaining awesome wish knew lecture earlier professor like wanted learn math unlike classes meant eliminate many students going forward possible note row abc matrix abc represent elimination matrices conundrum.'}'

[ ]:

Q6)vizualization

[92]:
```
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
```

[ ]:

Sentiment distribution(Positive/Negative/Neutral)

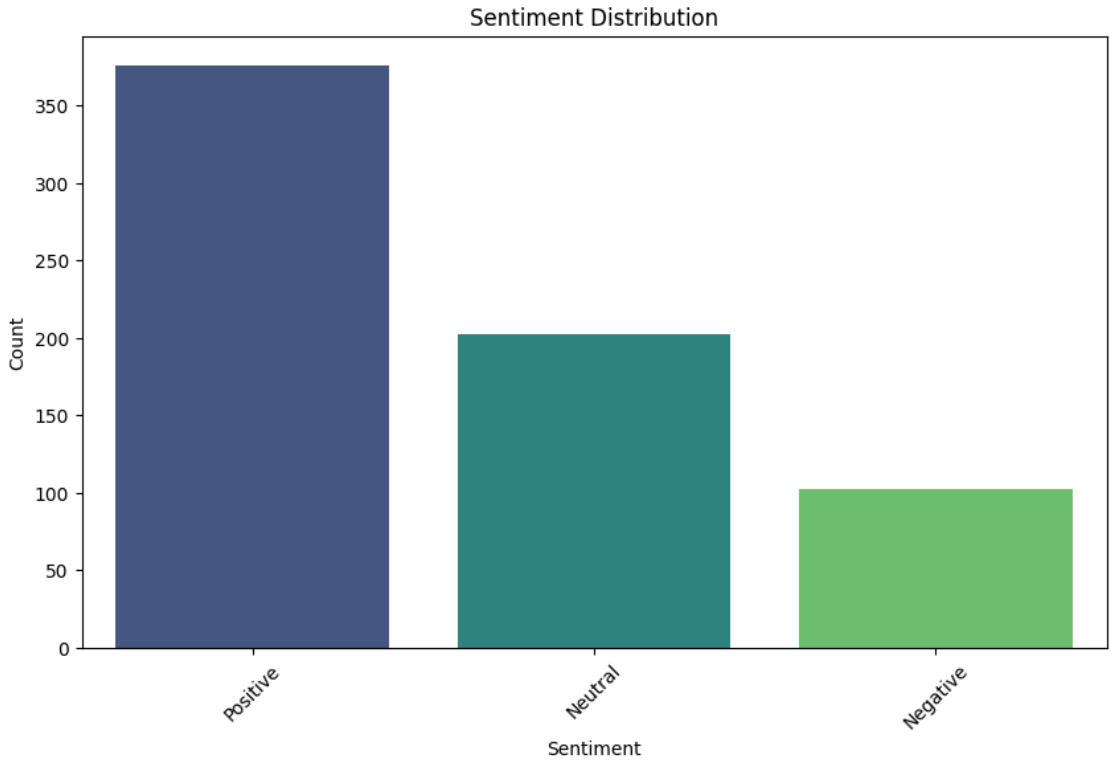
[166]:
```
sentiment_counts = df['Basic Sentiment'].value_counts()
```

[ ]:

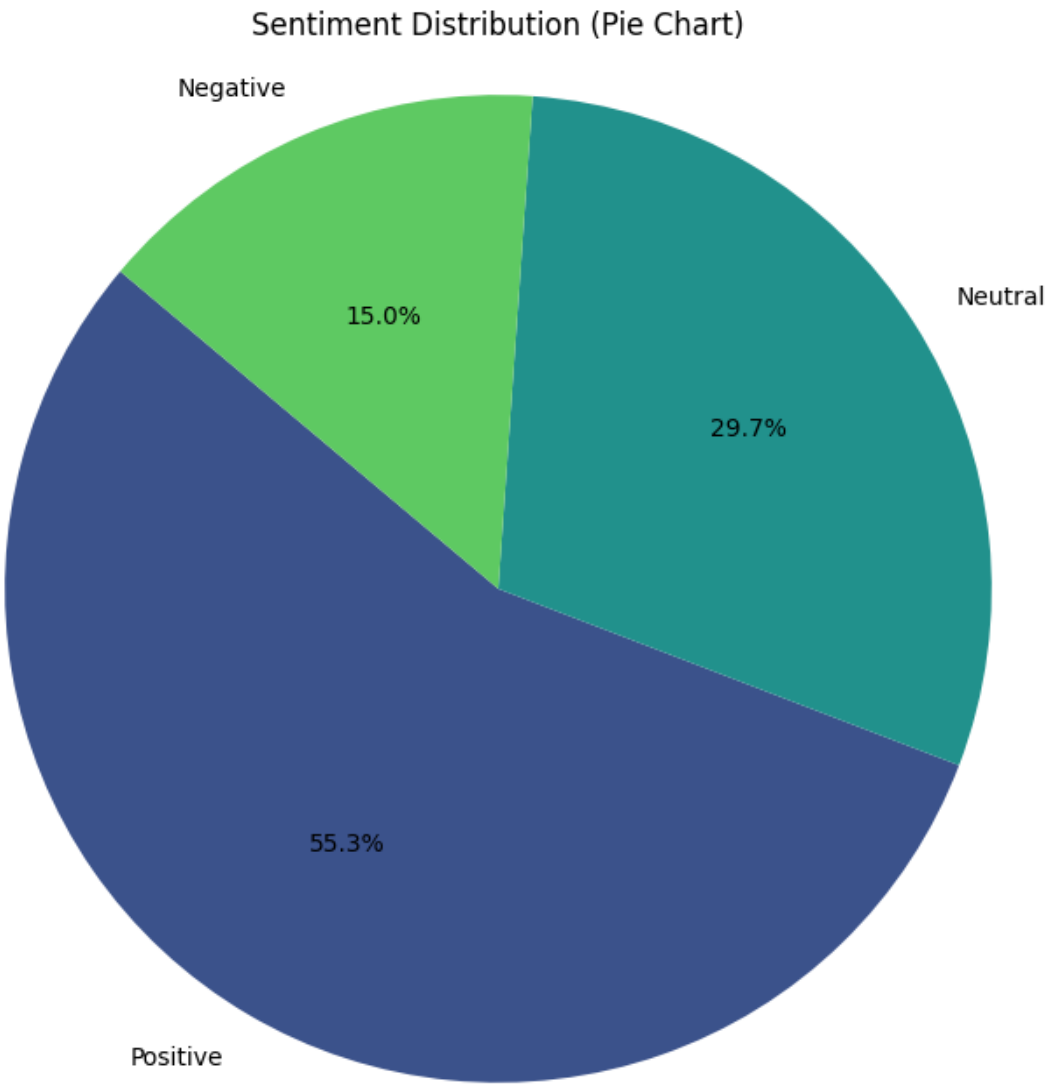
[180]:
```
plt.figure(figsize=(10, 6))
sns.barplot(x=sentiment_counts.index, y=sentiment_counts.values,␣
 ↪palette='viridis')
plt.title('Sentiment Distribution')
plt.xlabel('Sentiment')
plt.ylabel('Count')
plt.xticks(rotation=45)
plt.show()
```

```
<ipython-input-180-572ed898b2b9>:2: FutureWarning:


Passing `palette` without assigning `hue` is deprecated and will be removed in
v0.14.0. Assign the `x` variable to `hue` and set `legend=False` for the same
effect.


/usr/local/lib/python3.10/dist-packages/seaborn/_base.py:949: FutureWarning:

When grouping with a length-1 list-like, you will need to pass a length-1 tuple
to get_group in a future version of pandas. Pass `(name,)` instead of `name` to
silence this warning.

/usr/local/lib/python3.10/dist-packages/seaborn/_base.py:949: FutureWarning:

When grouping with a length-1 list-like, you will need to pass a length-1 tuple
to get_group in a future version of pandas. Pass `(name,)` instead of `name` to
```

```
silence this warning.

/usr/local/lib/python3.10/dist-packages/seaborn/_base.py:949: FutureWarning:

When grouping with a length-1 list-like, you will need to pass a length-1 tuple
to get_group in a future version of pandas. Pass `(name,)` instead of `name` to
silence this warning.
```



[179]:
```
plt.figure(figsize=(8, 8))
plt.pie(sentiment_counts, labels=sentiment_counts.index, autopct='%1.1f%%',␣
 ↪startangle=140, colors=sns.color_palette('viridis',␣
 ↪n_colors=len(sentiment_counts)))
plt.title('Sentiment Distribution (Pie Chart)')
plt.axis('equal')  # Equal aspect ratio ensures that pie chart is a circle.
plt.show()
```



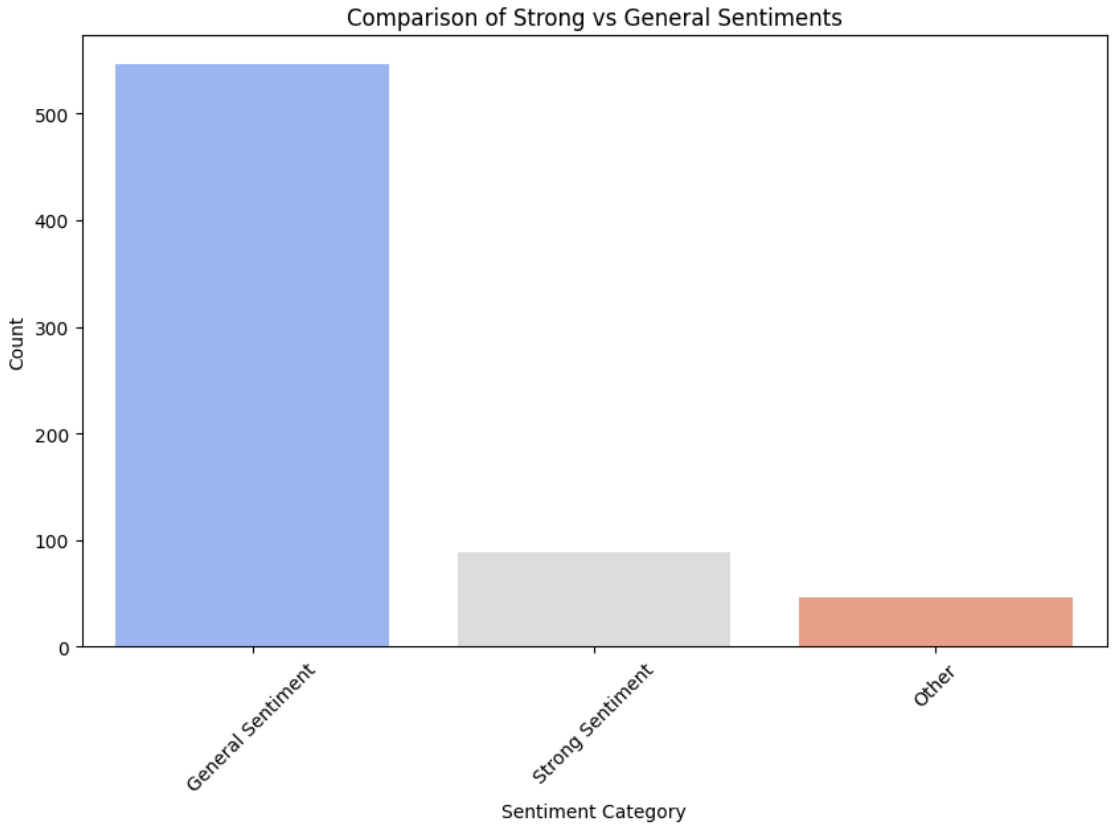
[ ]:

comparing the no. of strong vs Genral Sentiments

[168]:
```
def classify_sentiment(sentiment):
    if sentiment in ['Strong Positive', 'Strong Negative']:
        return 'Strong Sentiment'
    elif sentiment in ['General Positive', 'General Negative', 'Neutral']:
        return 'General Sentiment'
    return 'Other'

# Apply the classification
```

```
df['Sentiment Category'] = df['Fine-Grained Sentiment'].
 ↪apply(classify_sentiment)

# Count the occurrences of each category
sentiment_comparison = df['Sentiment Category'].value_counts()

# Plotting the comparison
plt.figure(figsize=(10, 6))
sns.barplot(x=sentiment_comparison.index, y=sentiment_comparison.values,␣
 ↪palette='coolwarm')
plt.title('Comparison of Strong vs General Sentiments')
plt.xlabel('Sentiment Category')
plt.ylabel('Count')
plt.xticks(rotation=45)
plt.show()
```

```
<ipython-input-168-3a30ee30a447>:9: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-
docs/stable/user_guide/indexing.html#returning-a-view-versus-a-copy
  df['Sentiment Category'] = df['Fine-Grained
Sentiment'].apply(classify_sentiment)
<ipython-input-168-3a30ee30a447>:16: FutureWarning:

Passing `palette` without assigning `hue` is deprecated and will be removed in
v0.14.0. Assign the `x` variable to `hue` and set `legend=False` for the same
effect.

  sns.barplot(x=sentiment_comparison.index, y=sentiment_comparison.values,
palette='coolwarm')
/usr/local/lib/python3.10/dist-packages/seaborn/_base.py:949: FutureWarning:
When grouping with a length-1 list-like, you will need to pass a length-1 tuple
to get_group in a future version of pandas. Pass `(name,)` instead of `name` to
silence this warning.
  data_subset = grouped_data.get_group(pd_key)
/usr/local/lib/python3.10/dist-packages/seaborn/_base.py:949: FutureWarning:
When grouping with a length-1 list-like, you will need to pass a length-1 tuple
to get_group in a future version of pandas. Pass `(name,)` instead of `name` to
silence this warning.
  data_subset = grouped_data.get_group(pd_key)
/usr/local/lib/python3.10/dist-packages/seaborn/_base.py:949: FutureWarning:
When grouping with a length-1 list-like, you will need to pass a length-1 tuple
to get_group in a future version of pandas. Pass `(name,)` instead of `name` to
silence this warning.
  data_subset = grouped_data.get_group(pd_key)
```



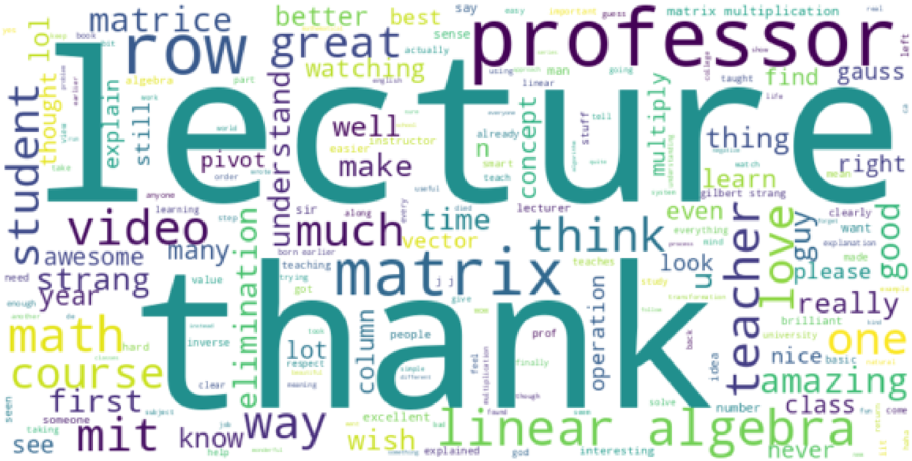
[ ]:

Words clouds for Positive/Negative/Neutral

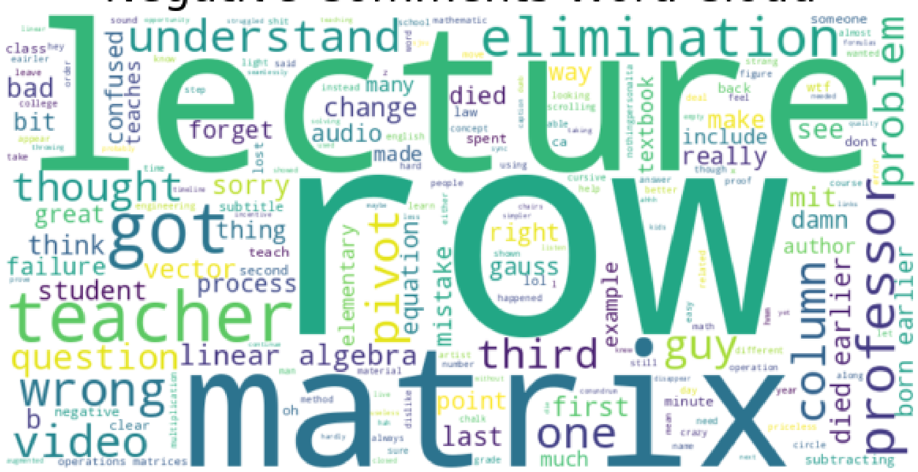
[181]:
```
def generate_wordcloud(data, ax, title):
    text = ' '.join(data)  # Join all the comments into a single string
    wordcloud = WordCloud(width=800, height=400, background_color='white').
 ↪generate(text)

    ax.imshow(wordcloud, interpolation='bilinear')
    ax.axis('off')  # Hide the axis
    ax.set_title(title, fontsize=20)

# Create a figure with subplots for each sentiment category
fig, axes = plt.subplots(3, 1, figsize=(10, 15))

# Generate word clouds for each sentiment category
positive_comments = df[df['Basic Sentiment'] ==␣
 ↪'Positive']['preprocessed_comment']
negative_comments = df[df['Basic Sentiment'] ==␣
 ↪'Negative']['preprocessed_comment']
```

```
neutral_comments = df[df['Basic Sentiment'] ==␣
 ↪'Neutral']['preprocessed_comment']

# Create word clouds in the respective subplots
generate_wordcloud(positive_comments, axes[0], 'Positive Comments Word Cloud')
generate_wordcloud(negative_comments, axes[1], 'Negative Comments Word Cloud')
generate_wordcloud(neutral_comments, axes[2], 'Neutral Comments Word Cloud')

# Adjust space between subplots
plt.subplots_adjust(hspace=0.5)  # Increase this value to create more space

# Show the plots
plt.show()
```

Positive Comments Word Cloud



Negative Comments Word Cloud



Neutral Comments Word Cloud

Clustering reviews with color coding

[130]:
```
!pip install sentence_transformers
```

```
Collecting sentence_transformers
  Downloading sentence_transformers-3.1.1-py3-none-any.whl.metadata (10 kB)
Requirement already satisfied: transformers<5.0.0,>=4.38.0 in
/usr/local/lib/python3.10/dist-packages (from sentence_transformers) (4.44.2)
Requirement already satisfied: tqdm in /usr/local/lib/python3.10/dist-packages
(from sentence_transformers) (4.66.5)
Requirement already satisfied: torch>=1.11.0 in /usr/local/lib/python3.10/dist-
packages (from sentence_transformers) (2.4.1+cu121)
Requirement already satisfied: scikit-learn in /usr/local/lib/python3.10/dist-
packages (from sentence_transformers) (1.5.2)
Requirement already satisfied: scipy in /usr/local/lib/python3.10/dist-packages
(from sentence_transformers) (1.13.1)
Requirement already satisfied: huggingface-hub>=0.19.3 in
/usr/local/lib/python3.10/dist-packages (from sentence_transformers) (0.24.7)
Requirement already satisfied: Pillow in /usr/local/lib/python3.10/dist-packages
(from sentence_transformers) (10.4.0)
Requirement already satisfied: filelock in /usr/local/lib/python3.10/dist-
packages (from huggingface-hub>=0.19.3->sentence_transformers) (3.16.1)
Requirement already satisfied: fsspec>=2023.5.0 in
/usr/local/lib/python3.10/dist-packages (from huggingface-
hub>=0.19.3->sentence_transformers) (2024.6.1)
Requirement already satisfied: packaging>=20.9 in
/usr/local/lib/python3.10/dist-packages (from huggingface-
hub>=0.19.3->sentence_transformers) (24.1)
Requirement already satisfied: pyyaml>=5.1 in /usr/local/lib/python3.10/dist-
packages (from huggingface-hub>=0.19.3->sentence_transformers) (6.0.2)
Requirement already satisfied: requests in /usr/local/lib/python3.10/dist-
packages (from huggingface-hub>=0.19.3->sentence_transformers) (2.32.3)
Requirement already satisfied: typing-extensions>=3.7.4.3 in
/usr/local/lib/python3.10/dist-packages (from huggingface-
hub>=0.19.3->sentence_transformers) (4.12.2)
Requirement already satisfied: sympy in /usr/local/lib/python3.10/dist-packages
(from torch>=1.11.0->sentence_transformers) (1.13.3)
Requirement already satisfied: networkx in /usr/local/lib/python3.10/dist-
packages (from torch>=1.11.0->sentence_transformers) (3.3)
Requirement already satisfied: jinja2 in /usr/local/lib/python3.10/dist-packages
(from torch>=1.11.0->sentence_transformers) (3.1.4)
Requirement already satisfied: numpy>=1.17 in /usr/local/lib/python3.10/dist-
packages (from transformers<5.0.0,>=4.38.0->sentence_transformers) (1.26.4)
Requirement already satisfied: regex!=2019.12.17 in
/usr/local/lib/python3.10/dist-packages (from
```

```
transformers<5.0.0,>=4.38.0->sentence_transformers) (2024.9.11)
Requirement already satisfied: safetensors>=0.4.1 in
/usr/local/lib/python3.10/dist-packages (from
transformers<5.0.0,>=4.38.0->sentence_transformers) (0.4.5)
Requirement already satisfied: tokenizers<0.20,>=0.19 in
/usr/local/lib/python3.10/dist-packages (from
transformers<5.0.0,>=4.38.0->sentence_transformers) (0.19.1)
Requirement already satisfied: joblib>=1.2.0 in /usr/local/lib/python3.10/dist-
packages (from scikit-learn->sentence_transformers) (1.4.2)
Requirement already satisfied: threadpoolctl>=3.1.0 in
/usr/local/lib/python3.10/dist-packages (from scikit-
learn->sentence_transformers) (3.5.0)
Requirement already satisfied: MarkupSafe>=2.0 in
/usr/local/lib/python3.10/dist-packages (from
jinja2->torch>=1.11.0->sentence_transformers) (2.1.5)
Requirement already satisfied: charset-normalizer<4,>=2 in
/usr/local/lib/python3.10/dist-packages (from requests->huggingface-
hub>=0.19.3->sentence_transformers) (3.3.2)
Requirement already satisfied: idna<4,>=2.5 in /usr/local/lib/python3.10/dist-
packages (from requests->huggingface-hub>=0.19.3->sentence_transformers) (3.10)
Requirement already satisfied: urllib3<3,>=1.21.1 in
/usr/local/lib/python3.10/dist-packages (from requests->huggingface-
hub>=0.19.3->sentence_transformers) (2.2.3)
Requirement already satisfied: certifi>=2017.4.17 in
/usr/local/lib/python3.10/dist-packages (from requests->huggingface-
hub>=0.19.3->sentence_transformers) (2024.8.30)
Requirement already satisfied: mpmath<1.4,>=1.1.0 in
/usr/local/lib/python3.10/dist-packages (from
sympy->torch>=1.11.0->sentence_transformers) (1.3.0)
Downloading sentence_transformers-3.1.1-py3-none-any.whl (245 kB)
                         245.3/245.3 kB
4.5 MB/s eta 0:00:00
Installing collected packages: sentence_transformers
Successfully installed sentence_transformers-3.1.1
```

[131]:
```
from sentence_transformers import SentenceTransformer
from sklearn.manifold import TSNE
import torch
```

[132]:
```
device = 'cuda' if torch.cuda.is_available() else 'cpu'

model = SentenceTransformer('paraphrase-MiniLM-L6-v2').to(device)
```

```
modules.json:   0%|          | 0.00/229 [00:00<?, ?B/s]
config_sentence_transformers.json:   0%|          | 0.00/122 [00:00<?, ?B/s]
README.md:   0%|          | 0.00/3.73k [00:00<?, ?B/s]
```

```
sentence_bert_config.json:   0%|          | 0.00/53.0 [00:00<?, ?B/s]

config.json:   0%|          | 0.00/629 [00:00<?, ?B/s]

model.safetensors:   0%|          | 0.00/90.9M [00:00<?, ?B/s]

tokenizer_config.json:   0%|          | 0.00/314 [00:00<?, ?B/s]

vocab.txt:   0%|          | 0.00/232k [00:00<?, ?B/s]

tokenizer.json:   0%|          | 0.00/466k [00:00<?, ?B/s]

special_tokens_map.json:   0%|          | 0.00/112 [00:00<?, ?B/s]

/usr/local/lib/python3.10/dist-
packages/transformers/tokenization_utils_base.py:1601: FutureWarning:
`clean_up_tokenization_spaces` was not set. It will be set to `True` by default.
This behavior will be depracted in transformers v4.45, and will be then set to
`False` by default. For more details check this issue:
https://github.com/huggingface/transformers/issues/31884
  warnings.warn(

1_Pooling/config.json:   0%|          | 0.00/190 [00:00<?, ?B/s]
```

[170]:
```
comments = df['preprocessed_comment'].tolist()

embeddings = model.encode(comments, show_progress_bar=True,␣
 ↪convert_to_tensor=True).to(device)

embeddings = embeddings.cpu().numpy()
```

```
Batches:   0%|          | 0/22 [00:00<?, ?it/s]
```

[172]:
```
from sklearn.cluster import KMeans
tsne = TSNE(n_components=3, random_state=42)
tsne_results = tsne.fit_transform(embeddings)

df['tsne_1'] = tsne_results[:, 0]
df['tsne_2'] = tsne_results[:, 1]
df['tsne_3'] = tsne_results[:, 2]
kmeans = KMeans(n_clusters=3, random_state=28)
df['cluster'] = kmeans.fit_predict(tsne_results)
```

```
<ipython-input-172-93712f35bf81>:4: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-
docs/stable/user_guide/indexing.html#returning-a-view-versus-a-copy
  df['cluster'] = kmeans.fit_predict(tsne_results)
```

[173]:
```
import plotly.express as px

fig = px.scatter_3d(
    df, x='tsne_1', y='tsne_2', z='tsne_3', color='cluster',
    hover_data={'preprocessed_comment': True, 'tsne_1': False, 'tsne_2': False,␣
 ↪'tsne_3': False},
    title="3D t-SNE Clustering of Comments Based on Sentiment"
)

fig.update_layout(
    scene = dict(
        xaxis_title="t-SNE Dimension 1",
        yaxis_title="t-SNE Dimension 2",
        zaxis_title="t-SNE Dimension 3"
    ),
    legend_title="Cluster"
)


fig.show()
```

[ ]:
```

```