CHAPTER - 1

INTRODUCTION

YouTube is probably the most popular of them, with millions of videos uploaded by its users and billions of comments for all of these videos. YouTube was ranked as the second-most popular site by Alexa Internet, a web traffic analysis company, on December 2016. In order to increase the user's interaction, it allows their users can express themselves by rating the videos they watch objects (via clicking on the like/dislike buttons) and interacting with the other community members (via the comments feature) These activities (like, dislike, or the number of them) of views) of the users can serve as a global indicator of quality or popularity of a particular vide. Moreover, these Meta data (like/dislike/number of views) serve the purpose of helping the community filter relevant opinions more efficiently.

1.1. SENTIMENT ANALYSIS

Sentiment analysis is the process of determining whether a block of text is positive, negative, or, neutral. Sentiment analysis is the contextual mining of words that indicates the social sentiment of a brand and also helps the business determine whether the product that they are manufacturing is going to make a demand in the market or not. The goal that sentiment analysis tries to achieve is to analyze people's opinions in a way that can help businesses expand. It focuses not only on polarity (positive, negative and neutral) but also on emotions (happy, sad, angry, etc.). It uses various Natural Language Processing algorithms such as rule-based, automatic, and hybrid.

1.2. TYPES OF SENTIMENT ANALYSIS:

(i) Fine-grained sentiment analysis

(ii) Emotion detection

(iii) Aspect based sentiment analysis

(iv) Multilingual sentiment analysis

Fine-grained sentiment analysis: This depends on the polarity based. This category can be designed as very positive, positive, neutral, negative, very negative. The rating is done on the scale 1 to 5. If the rating is 5 then it is very positive, 2 then negative and 3 then neutral.

Emotion detection: The sentiment happy, sad, anger, upset, jolly, pleasant, and so on come under emotion detection. It is also known as a lexicon method of sentiment analysis.

Aspect based sentiment analysis: It focuses on a particular aspect like for instance, if a person wants to check the feature of the cell phone then it checks the aspect such as battery, screen, camera quality then aspect based is used.

Multilingual sentiment analysis: Multilingual consists of different languages where the classification needs to be done as positive, negative, and neutral. This is highly challenging and comparatively difficult.

In this project I have used **Aspect based sentiment analysis.**

1.3. WHY PERFORM SENTIMENT ANALYSIS?

According to the survey, 80% of the world's data is unstructured. The data needs to be analyzed and be in a structured manner whether it is in the form of emails, texts, documents, articles, and many more.

(i) Sentiment Analysis is required as it stores data in an efficient, cost-friendly.

(ii) Sentiment analysis solves real-time issues and can help you solve all the real-time scenarios.

1.4. HOW DOES SENTIMENT ANALYSIS WORK?

There are three approaches used:

Rule-based approach: Over here, the lexicon method, tokenization, parsing comes in the rule-based. The approach is that counts the number of positive and negative words in the given dataset. If the number of positive words is greater than the negative words then the sentiment is positive else vice-versa.

Automatic Approach: This approach works on the machine learning technique. Firstly, the datasets are trained and predictive analysis is done. The next process is the extraction of words from the text is done. This text extraction can be done using

different techniques such as Naive Bayes, Linear Regression, Support Vector, Deep Learning like this machine learning techniques are used.

Hybrid Approach: It is the combination of both the above approaches i.e. rule-based and automatic approach. The surplus is that the accuracy is high compared to the other two approaches.

In this project I have used **Rule-based approach.**

CHAPTER - 2

PROBLEM STATEMENT

A person who uploads videos to YouTube that are related to education, creativity, their profession, and many other things needs to analyze those videos to discover ways to enhance their quality and minimize errors by using user comments. The user comments can be categorized to show how the video is alike.

Use sentiment analysis to evaluate the user comments' tone and categorize them as positive, negative, or neutral using SentimentIntensityAnalyzer and SentiWordNet text analysis methods in the NLTK library.

CHAPTER - 3

BACKGROUND WORK

I have referred to various books, research papers and sites to do this project and went through many algorithms to get a clear idea about their usage in my work. To understand & learn the algorithms and methods, I read about them in books, research papers and surfed the internet. I installed Jupyter Notebook for my work. It is open-source software available on the internet for free.

After installing all the requirements, installing all the necessary libraries used for this project.

I made a research on collecting the YouTube comments for a particular video, which will acts as data for this project. After the research, I choose youtube_comment_scraper_python library to collect the comments from a particular video on YouTube.

For performing data pre-processing work, I referred to research papers and websites. The input is given to an NLP pipeline for data pre-processing. For the construction of an effective pipeline, I went through various concepts and took the best that was apt for my work.

I compared my work with various earlier types of research that have been done in this area and analyze the improvements that can be done on this project.

3.1. REVIEW OF LITERATURE

1. **Sentiment Analysis on YouTube: A Brief Survey**
   MAGNT Research Report (ISSN. 1444-8939),
   Vol.3 (1). PP: 1250-1257, Jan. 2015.

   Sentiment analysis or opinion mining is the field of study related to analyze opinions, sentiments, evaluations, attitudes, and emotions of users which they express on social media and other online resources. The revolution of social media sites has also attracted the users towards video sharing sites, such as YouTube. The online users express their opinions or sentiments on the videos that they watch on such sites. This paper presents a brief survey of techniques to analyze opinions posted by users about a particular video.

2. **Retrieving YouTube Video by Sentiment Analysis on User Comment**
   IEEE International Conference on Signal and Image Processing Applications (IEEE ICSIPA 2017), Malaysia, September 12-14, 2017.

   YouTube is one of the comprehensive video information source on the web where video is uploading continuously in real time. It is one of the most popular site in social media, where users interact with sharing, commenting and rating (like/views) videos. Generally the quality, relevancy and popularity of the video is maintained based on this rating. Sometimes irrelevant and low quality videos ranked higher in the search result due to the number of views or likes, which seems untenable. To minimize this issue, we present a Natural Language processing (NLP) based sentiment analysis approach on user comments. This analysis helps to find out the most relevant and popular video of YouTube according to the search. The effectiveness of the proposed scheme has been proved by a data driven experiment in terms of accuracy of finding relevant, popular and high quality video.

3.2. Python Libraries

- youtube_comment_scraper_python
- Natural language toolkit (nltk),
- Pandas,
- Matplotlib and
- Regular expression.

CHAPTER – 4

TECHNOLOGY ADOPTED

4.1. Natural Language Processing (NLP)

Natural Language Processing (NLP) is a process of manipulating or understanding the text or speech by any software or machine. NLTK (Natural Language Toolkit) is a suite that contains libraries and programs for statistical language processing. It is one of the most powerful NLP libraries, which contains packages to make machines understand human language and reply to it with an appropriate response.

NLP enables computers to understand natural language as humans do. Whether the language is spoken or written, natural language processing uses artificial intelligence to take real-world input, process it, and make sense of it in a way a computer can understand. Just as humans have different sensors such as ears to hear and eyes to see computers have programs to read and microphones to collect audio. And just as humans have a brain to process that input, computers have a program to process their respective inputs. At some point in processing, the input is converted to code that the computer can understand.

CHAPTER - 5

DETAILS OF TOOLS USED

5.1. Python

A general-purpose interpreted, interactive, object-oriented, and high-level programming language.

5.2. Jupyter Notebook

Jupyter Notebook (formerly IPython Notebook) is a web-based interactive computational environment for creating notebook documents. Jupyter Notebook is built using several open-source libraries, including IPython, ZeroMQ, Tornado, jQuery, Bootstrap, and MathJax. A Jupyter Notebook application is a browser-based REPL containing an ordered list of input/output cells which can contain code, text (using Github Flavored Markdown), mathematics, plots and rich media.

JupyterLab is a newer user interface for Project Jupyter, offering a flexible user interface and more features than the classic notebook UI. Jupyter Notebook is similar to the notebook interface of other programs such as Maple, Mathematica, and SageMath, a computational interface style that originated with Mathematica in the 1980s. Jupyter interest overtook the popularity of the Mathematica notebook interface in early 2018.

5.3. Libraries

6.3.1. Youtube_comment_scraper_python

Youtube_comment_scraper_python is a python library to fetch video comments on YouTube using browser automation. It currently runs only on windows. This library was released April 12, 2021, by DataKund. It fetch the video comments by using the video URL.

6.3.2. NLTK

The Natural Language Toolkit, or more commonly NLTK, is a suite of Libraries and programs for symbolic and statistical natural language processing for English written in the Python programming language**.**

6.3.3. Pandas

Pandas is defined as an open-source library that provides high-performance data manipulation in Python. It is used for data analysis in Python. Data analysis requires lots of processing, such as **restructuring, cleaning** or **merging**, etc. There are different tools are available for fast data processing, such as **Numpy, Scipy, Cython**, and **Panda**. Pandas is fast, simple and more expressive than other tools.

6.3.4. Matplotlib

Matplotlib is a plotting library for the Python programming language and its numerical mathematics extension NumPy. It provides an object-oriented API for embedding plots into applications using general-purpose GUI toolkits like Tkinter, wxPython, Qt, or GTK. There is also a procedural "pylab" interface based on a state machine (like OpenGL), designed to closely resemble that of MATLAB, though its use is discouraged. SciPy makes use of Matplotlib.

6.3.5. Regular Expression

A regular expression (or RE) specifies a set of strings that matches it; the functions in this module let you check if a particular string matches a given regular expression (or if a given regular expression matches a particular string, which comes down to the same thing). Regular expressions can be concatenated to form new regular expressions; if A and B are both regular expressions, then AB is also a regular expression. In general, if a string p matches A and another string q matches B, the string pq will match AB. This holds unless A or B contain low precedence operations; boundary conditions between A and B; or have numbered group references. Thus, complex expressions can easily be constructed from simpler primitive expressions like the ones described here.

CHAPTER – 6

METHODOLOGY

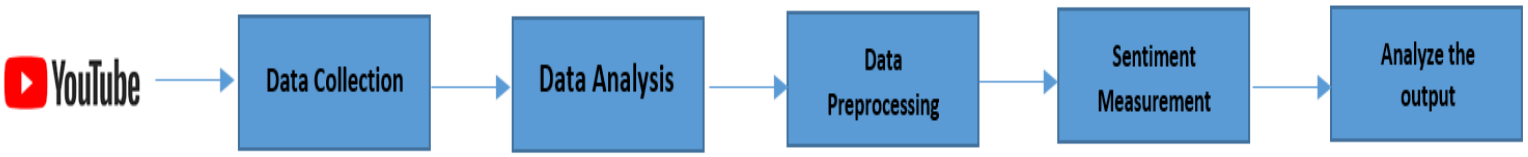


Fig. 6.1. Work process of sentiment analysis on YouTube video comment

6.1. Data Collection

The primary task of any project is to collect the data that is going to be used. The data, namely the user comments on the YouTube video, will be collected for this project. There are various Python libraries used to scrape the data from the YouTube video. Some of the libraries are

- Beautifulsoup,
- scraper,
- youtube_comment_scraper_python and
- Selenium.

These libraries are used for web scrapping, whereas, youtube_comment_scraper_python is a special library used to scrap the comments of the particular video using the video link. The scraped data can be loaded in the Pandas dataframe for further uses.

6.2. Data Analysis

It is an important task to analyze the collected data. The data is stored in the Pandas dataframe. The youtube_comment_scraper_python returns the data with five columns, namely, comment, likes, time, userlink, and user.

Some of the analyses done on the collected data are:

- Knowing the data types of the data,
- Total count of the data and
- Drop some unwanted columns in the data.

6.3. Data Pre-processing

Once the data is collected, it must undergo certain pre-processing techniques before being used in NLP algorithms.

Some of the pre-processing techniques are

- Converting them into lowercase,
- Removing new line characters,
- Punctuation removal,
- Stopwords removal,
- Removing references and hashtags,
- Removing multiple spaces,
- Removing special characters,
- Work tokenization,
- POS tagging and
- Word lemmatization.

6.4. Sentiment Measurement

After the data is pre-processed, it is given to NLP algorithms in nltk library. **NLTK is a standard python library** that provides a set of diverse algorithms for NLP. It is one of the most used libraries for NLP and Computational Linguistics. In this project, I used two text analysis methods.

1. **VADER Sentiment Analysis:**

VADER (Valence-Aware Dictionary and Sentiment Reasoner) is a lexicon and rule-based sentiment analysis tool that is specifically attuned to sentiments expressed in social media. **VADER** uses a combination of a sentiment lexicon is a list of lexical features (e.g., words) that are generally labelled according to their

semantic orientation as either positive or negative. VADER not only tells us the positivity and negativity scores but also whether a sentiment is positive or negative.

In the **VADER** model, the SentimentIntensityAnalyzer is used here.

VADER's **SentimentIntensityAnalyzer**() takes in a string and returns a dictionary of scores in each of four categories:

- negative
- neutral
- positive
- compound

The compound score is a metric that calculates the sum of all the lexicon ratings, which have been normalized between -1 (the most extreme negative) and +1 (the most extreme positive).

- Positive sentiment (compound score >= 0.03)
- (compound score > -0.03) and (compound score < 0.03) indicate Neutral sentiment.
- Negative sentiment: (compound score <= -0.03)

2. SentiWordNet

The most basic approach for Sentiment Analysis, i.e. Lexicon based Sentiment Analysis. The lexicon that I will be using is the SentiWordNet lexicon.

WordNet

1. WordNet is a lexical database composing English words, grouped as synonyms into what is known as synsets.
2. It is a freely available tool, that can be downloaded from its official website.
3. While WordNet can be loosely termed as a Thesaurus, it is said to be more semantically accurate, since it stores synonyms of words put together is specific contexts.
4. All the words are linked together by the ISA relationship (more commonly, generalization). For example, a car is a type of vehicle, just like a truck.

SentiWordNet

1. SentiWordNet operates on the database provided by WordNet.

2. The additional functionality that it provides is the measure of positivity, negativity, or neutrality required for Sentiment Analysis.

As a result, each synset s is assigned a Pos(s), or positivity score. Neg(s): a negativity score Obj(s): an objectivity (neutrality) score

Pos(s) + Neg(s) + Obj(s) = 1

The scores are very precise, pertaining to the word itself and its context. All three scores range between 0 and 1.

6.5. Analyze the Output

After measuring the sentiments of the comments, the comments should be classified as positive, negative or neutral based on the compound score of SentimentIntensityAnalyzer and the value obtained using the positive and negative scores from the SentiWordNet text analysis method.

The comment classification is based on the SentimentIntensityAnalyzer's compound score on the following conditions:

i) If the compound score is greater than 0.03, then the comment is positive.

Ii) If the compound score is less than -0.03, then the comment is negative.

iii) If the compound score is between 0.03 and -0.03, then the comment is neutral.

The comment classification is based on the value obtained from SentiWordNet under the following conditions:

i) If the value is greater than 0.3, then the comment is positive.

ii) If the value is less than -0.3, then the comment is negative.

iii) If the value is between 0.3 and -0.3, then the comment is neutral.

Once the comments are classified, it's time to analyze the sentiment of the YouTube video based on the classified user comments. In this project, I used a bar plot to visualize the outputs for quick understanding.

CHAPTER – 7

CONCLUSION AND FUTURE WORK

7.1. Conclusion

The "YouTube Comment Sentiment Analysis Using NLP" project uses SentimentIntensityAnalyzer and SentiWordNet text analysis techniques from the NLTK toolkit to analyze the tone of user comments and categorize them as positive, negative, or neutral. So, using the video URL and the youtube comment scraper python library, data is obtained. The data is then analyzed and preprocessed. To determine the sentiment of the comments, the preprocessed data is fed into the SentimentIntensityAnalyzer and SentiWordNet text analysis algorithms available in the NLTK toolkit. The result is bar charts that visually depict the classified comment counts by positive, negative, and neutral. The comments are categorized based on the relevant criteria.

7.2. Future Work

The users, who belong to various countries, languages, cultures, and so on, use the YouTube social platform. They also leave comments on the videos in various languages. This project is limited to comments that are in English. In the future, it will be possible to convert comments from various languages to English by using available APIs to translate the comments, which can then be used for analysis. Exploring many other text analysis methods for sentiment analysis that can be added to this project for effective analysis of the YouTube video.

REFERENCES

[1] Sentiment Analysis on YouTube: A Brief Survey
MAGNT Research Report (ISSN. 1444-8939),
Vol.3 (1). PP: 1250-1257, Jan. 2015.

[2] Retrieving YouTube Video by Sentiment Analysis on User Comment
IEEE International Conference on Signal and Image Processing Applications (IEEE ICSIPA 2017), Malaysia, September 12-14, 2017.

APPENDICES

Code:

```
#Libraries
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
%matplotlib inline
import os

# Import functions for data preprocessing & data preparation
from nltk.sentiment.vader import SentimentIntensityAnalyzer

import nltk
from nltk.tokenize import word_tokenize
from nltk.corpus import stopwords
import string
from string import punctuation
import re

from nltk.corpus import wordnet as wn
from nltk.corpus import sentiwordnet as swn
list(swn.senti_synsets('slow'))
from nltk.tag import pos_tag
from nltk.stem import WordNetLemmatizer
from youtube_comment_scraper_python import *


link = input("Youtube link: ")
saved = input("Output name: ")
saved = "D:/PG/SEM IV/" + saved + '.csv'

youtube.open(link)

response = youtube.video_comments()
all_data = []
for i in range(0, 10): # It will scroll 10 times
    response = youtube.video_comments()
    data = response['body']
```

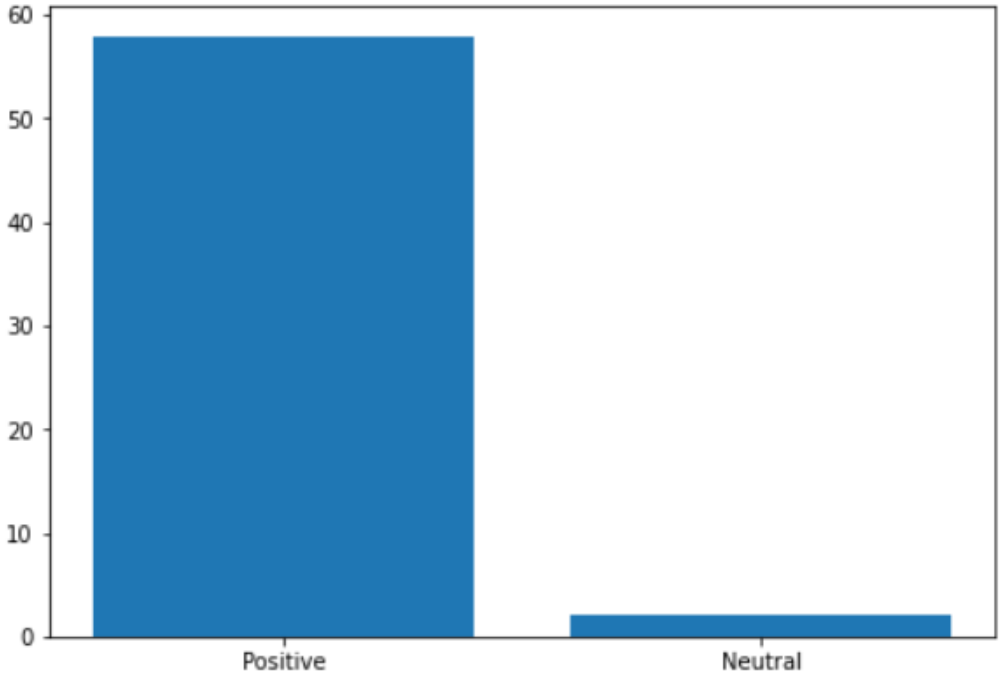
```
    all_data.extend(data)
df = pd.DataFrame(data)
print(df)

df.to_csv(saved)

print(df.columns)

# reading the csv file
data = df

# print the column names of the dataset
print(data.columns)

# print top of the data
print(data.head(10))

data = data.drop('UserLink', axis =1)
data = data.drop('user', axis =1)
print(data.head())

data.count()

data.dtypes

data["comment_text"] = data["Comment"].copy()

print(data.columns)

stop_words = stopwords.words('english')

def text_processing(text):
    # convert text into lowercase
    text = text.lower()

    # remove new line characters in text
    text = re.sub(r'\n',' ', text)

    # remove punctuations from text
    text = re.sub('[%s]' % re.escape(punctuation), "", text)
```

```
    # remove references and hashtags from text
    text = re.sub("^a-zA-Z0-9$,.", "", text)

    # remove multiple spaces from text
    text = re.sub(r'\s+', ' ', text, flags=re.I)

    # remove special characters from text
    text = re.sub(r'\W', ' ', text)


    text = ' '.join([word for word in word_tokenize(text) if word not in stop_words])


    return text

for i in range(0,len(data)):
    data["comment_text"][i] = text_processing(data['comment_text'][i])
sentiments = SentimentIntensityAnalyzer()
data["Positive"] = [sentiments.polarity_scores(i)["pos"] for i in data['Comment']]
data["Negative"] = [sentiments.polarity_scores(i)["neg"] for i in data['Comment']]
data["Neutral"] = [sentiments.polarity_scores(i)["neu"] for i in data['Comment']]
data['Compound'] = [sentiments.polarity_scores(i)["compound"] for i in
data['Comment']]
score = data["Compound"].values
sentiment = []
for i in score:
    if i >= 0.03:
        sentiment.append('Positive')

    elif i <= -0.03:
        sentiment.append('Negative')
    else:
        sentiment.append('Neutral')
data["Sentiment"] = sentiment
data.head()

def sentiwordnet_analysis(text):
    sentence= text
    token = nltk.word_tokenize(sentence)
```

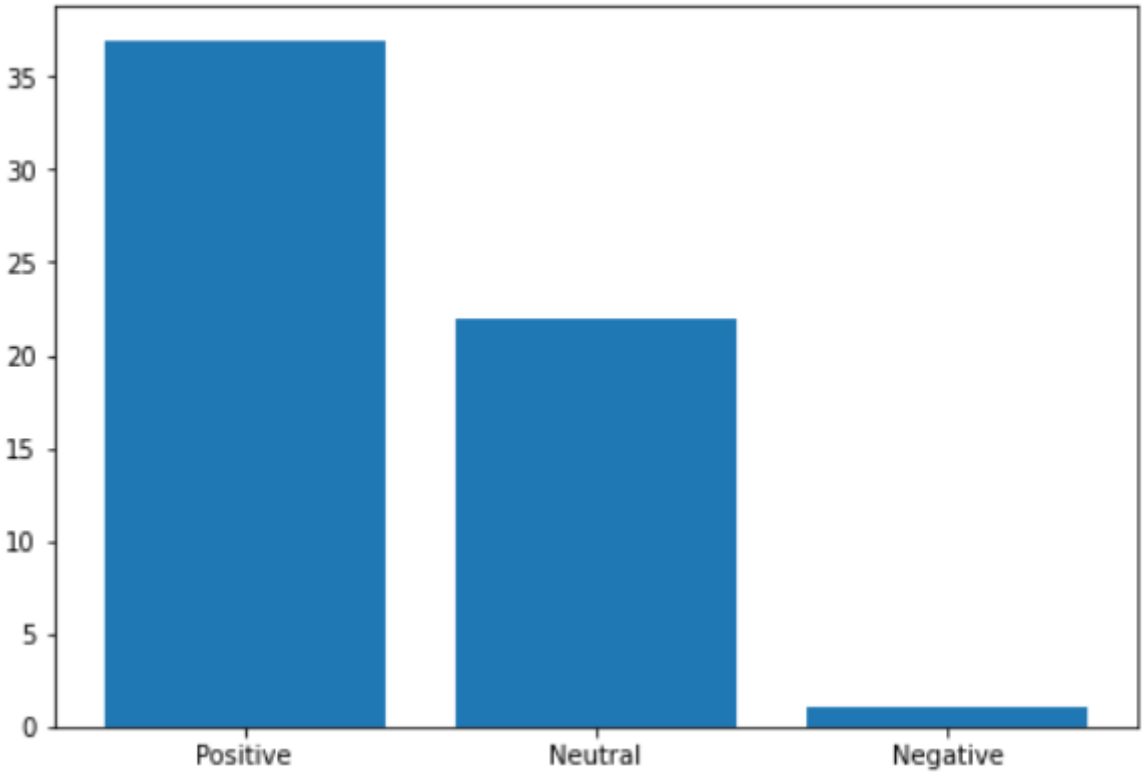
```
    after_tagging = nltk.pos_tag(token)
#     print (token)
#     print (after_tagging)
    def penn_to_wn(tag):
        if tag.startswith('J'):
            return wn.ADJ
        elif tag.startswith('N'):
            return wn.NOUN
        elif tag.startswith('R'):
            return wn.ADV
        elif tag.startswith('V'):
            return wn.VERB
        return None
    sentiment = 0.0
    tokens_count = 0
    lemmatizer = WordNetLemmatizer()
    for word, tag in after_tagging:
            wn_tag = penn_to_wn(tag)
            if wn_tag not in (wn.NOUN, wn.ADJ, wn.ADV):
                continue

            lemma = lemmatizer.lemmatize(word, pos=wn_tag)
            if not lemma:
                continue

            synsets = wn.synsets(lemma, pos=wn_tag)
            if not synsets:
                continue

            # Take the first sense, the most common
            synset = synsets[0]
            swn_synset = swn.senti_synset(synset.name())
#               print(swn_synset)

            sentiment += swn_synset.pos_score() - swn_synset.neg_score()
            tokens_count += 1
    return sentiment

data['Sentiwordnet_analyse'] = [sentiwordnet_analysis(i) for i in data['Comment']]
```

```
score = data["Sentiwordnet_analyse"].values
sentiment = []
for i in score:
    if i >= 0.3:
        sentiment.append('Positive')

    elif i <= -0.3:
        sentiment.append('Negative')
    else:
        sentiment.append('Neutral')
data["Sentiment_sentiwordnet"] = sentiment

data.to_csv('comment_analysed.csv')

count_df = data['Sentiment'].value_counts()
count_index = count_df.index
count_values = count_df.values
print(count_index, count_values)

count_list = count_df.values.tolist()
count_list

fig = plt.figure()
ax = fig.add_axes([0,0,1,1])
ax.bar(count_index,count_values)
plt.show()

count_df = data['Sentiment_sentiwordnet'].value_counts()
count_index = count_df.index
count_values = count_df.values
print(count_index, count_values)

count_list = count_df.values.tolist()
count_list

fig = plt.figure()
ax = fig.add_axes([0,0,1,1])
ax.bar(count_index,count_values)
plt.show()
```

Output:



Output Analyzed by SentimentIntensityAnalyzer



Output Analyzed by SentiWordNet