



VIT[®]
Vellore Institute of Technology
(Deemed to be University under section 3 of UGC Act, 1956)

SENTIMENTAL ANALYSIS AND OPINION MINING

YOUTUBE MINING

Group Members

Samridhi Jhijaria – 19BCE2470

Diva Bhatia – 19BCE2452

SLOT – C1+TC1

**Artificial Intelligence – CSE3013(Project
Component)**

Abstract:

This project deals with the extraction of the data of any particular video on Youtube platform. It uses the concept of Youtube API keys facilitated by Google to help developers extract data for rightful use. We obtained my own personal key, which has been used for this project. Here I extract the data of any Youtube video, given its video id and extract all the data such as no. of likes, dislikes, comments and views.

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. It involves the use of data mining, machine learning (ML) and artificial intelligence (AI) to mine text for sentiment and subjective information.

Literature survey:

S.No .	Title of the Paper And year	Algorithm used	Data set being used	Performance measures	Gap identified	Scope for future work
1.	Sentiment Analysis – Mhd. Majid Akhtar June 2019	TextBlob Sentiment (python) To check polarity.	“Liverpool FC- The road to Madrid- UCL2019” by channel MRCLFCompilations. ID of the video is Y-XHMLaJL-s. Download Dataset of Youtube Video comments via running a script Convert Json file to CSV file via convert_json_to_csv.ipynb file Run 'sentinment_analysis' python file and the results showing positive, negative and neutral response & knowing the accuracy of model.	ACCURACY: 70% PRECISION: 100% RECALL: 75% In finding and categorizing positive, negative and neutral comments.	The dataset is not too clean.	The dataset is not too clean. The proper way should be to tokenize it, exclude stop words and generate a bag of words from it. These steps were ignored in this project.

2.	Polarity trend analysis of public sentiment on YouTube - Amar Krishna 2019	Naive Bayes classification technique. Bayesian probability	Internet Movie Database(IMDb) dataset designed and developed by Pang and Lee	Results show that the trends in users' sentiments is well correlated to the real-world events associated with the respective keywords. Using the Weka forecasting tool, they were able to predict the possible sentiment scores for 26 weeks into the future with a confidence interval of 95%.	focused on the comment ratings and their dependencies to topics	Can study the sentiment trends in YouTube comments, with focus on the popular/trending keywords
3.	Naïve Multi-label classification of YouTube comments using comparative opinion mining -Asad Ullah Rafiq Khan* , Madiha Khan, Mohammad Badruddin Khan 30 march 2018	Naïve Bayes Probabilistic classifier. MEKA was used for multi-label classification. In order to gain further insights, experiments were performed in WEKA with different settings.	java program to fetch all the comments of "IPHONE VS ANDROID" video and used YouTube API and GSON library to parse JSON. stored only comparative comments in MySQL database	comparative opinion mining of YouTube comments was performed on comments with contents containing comparison of Android and iOS. The results in terms of different performance measures are not satisfactory but the naïve assumption regarding neighbourhood words of keywords performed well as compare to other experimental setting	difficult to understand the polarity of some comments due to ambiguity, spelling errors, missing punctuation in the sentences and grammatical errors.	work further on data processing side and use lexicon to make the comments more comprehensible for machine.
4.	Sentiment Analysis on YouTube: A Brief	Lexicon oriented techniques WordNet	Data collected in 3-month period from YouTube.	Detected User Filter New Unrated Comments	Detection of Sentiment Polarity -	Improving the social lexicon and to validate it

	Survey -Muhammad Zubair Asghar , Shakeel Ahmad , Afsana Marwat , Fazal Masud Kundi 2014	and SentiWord Net. Breadth-first Search. Fusion Method. Centroid classifier.		Polarity of comments High Accuracy on Emotions and Valence Annotation	lexicon based sentiment analysis, negation detection and Social Media Aware Phrase Detection	statistically and proper event classification can help to increase the performance to predict rating of comments.
5.	Social Network Analysis and Mining - Makazhanov et al. 2014	Hybrid SVM Logic Regression Model Sentistrength	Twitter Election Commission of Tajikistan Twitter API	Users political preference was predicted from the knowledge derived from the content generated and users behaviour during the campaign.	users political preference prediction problem on Twitter network	Can include contextual and behavioural features based prediction models.
6.	Opinion mining on marglish and Devanagari comments of youtube cookery channels – S.Shah, A.Kaushik, S.Sharma, J.Shah 17march 2020	Parametric and non parametric learning models, Count vectorizer with multilayer perception and Bernoulli Naïve Bayes	Marglish (Marathi+English) and Devanagari comments of top marathi youtube cookery channels	Count vectorizer with BNB works best with accuracy of 60.60%	Emoticons and sarcasm is not considered in the datasets	Clustering algo like density based spatial clustering of applications with noise DBSCAN and K-means can be used to simplify labelling process.
7.	Polarity trend analysis of public sentiment on youtube – Amar Krishna 2017	Decision ntrees, naïve bayes, RF, Python scripts with youtube API,	Youtube comments related to popular topics. Collecting 1000 comments per video(max3000 videos) using keywords like 'Nadal', 'Obama', 'Federer' etc.	Results show that the trends in users' sentiments is well correlated to the real world events associated with the respective keywords.	Keywords used can be expanded to a larger vocabulary. K-nearest neighbour classifier could have been used.	Topics like social media text classification using different classification techniques (higher accuracy rate) is a prominent area for future research

8.	Sentiment analysis model based on youtube comments – F.I. Tanaseb, Irwan S., H.D. Purnomo August 2018	Support vector machine SVN, lexicon based, confusion matrix	Youtube comments of a particular person from 2017 to 2018	Results – Accuracy: 84% Precision: 91% Recall: 80% TruePositive rate 91.1 TrueNegative rate 44.8%	Less data recordings might show more TPR or TNR in the given period of time	Data recordings can be taken in higher numbers to achieve the accuracy of results and conclusions on the opinion mining analysis
9.	Opinion mining, a comparison on hybrid approaches – Alex M.G. Almeida, Emerson C. Paraiso, Stela N. Moriguchi 2019	Lexicon based OM Crowd Explicit Sentiment Analysis, Cognitive Based Polarity Identification, nlp	Datasets that provide a more informal use of language are suitable for hybrid approaches. Social media content available at cyberemotions consortium. SentiStrength dataset.	Emoticons attribution can improve accuracy while combined with crowd explicit sentiment analysis and cognitive based polarity identification approaches. Hybrid approaches achieve better precision in case of neutral sentences.	Knowledge related to sarcasm is hard to categorise.	Focusing on the computational cost and on a deeper research into sarcasm.
10.	Naïve Multi-label classification using comparative opinion mining- A. U. Khan, M. Khan, M. Khan	Naïve Bayes Probabilistic classifier. MEKA and WEKA were used.	comments of “IPHONE VS ANDROID” video from youtube.	The results in terms of different performance measures are not satisfactory but the naïve assumption regarding neighbourhood words of keywords performed well as compared to other experimental settings	difficult to understand the polarity of some comments due to ambiguity, spelling errors, missing punctuations.	Perform better data processing side and use lexicon.
11.	Opinion Mining -A. Severyn , A. Moschitti ,O.	FVEC , STRUCT.	Youtube videos of technical reviews and advertisements of automobiles (AUTO) and	The metrics used are precision, recall and accuracy for each	Tough to work with detailed and well-argument	joint model to classify a the comments of a given

	Uryupina , B. Plank		tablets (TABLETS).	individual class and the general accuracy of the multi-class classifier (Acc).	criticism or irony.	video can be used.
12.	Sentiment analysis— F. Tanaseb, H.D. Purnomo 2018	SVN	A Youtube user's comments from 2017 to 2018	Precise results were obtained showing positive and negative comments	Low comment rate.	A group of users' comments can be targeted to get a better result
13.	Social Network Mining - Makazhano v et 2014	Hybrid SVM LRM	Election Commission of Tajikistan Twitter API	Users' political preference was predicted from the knowledge derived from the content generated and users' behaviour during the campaign.	Irony and sarcastic comments were tough to classify	prediction models can be used to include contextual and behavioural features
14.	Sentiment Analysis or Opinion Mining: A Review - Bilal Saber, Saidah Saad	The existing techniques including machine learning (supervised and unsupervised), and lexical- based approaches	Clippings from Blogs, Review Sites, News articles, Social Networks.	The performance of the approaches; (SVM, NB and ME) with SA and classification are highly successful. The other approaches include ID3, CentroidClassifier, Winnow Classifier, KNN and Association Rules mining approach.	The negative sentiment may sometimes be represented in a sentence without using any notable negative words	tackling language generalization and negations.
15.	Opinion Mining on YouTube -Aliaksei Severyn , Alessandro Moschitti , Olga Uryupina , Barbara Plank ,	FVEC bag of words model. STRUCT model to induce structural patterns of sentiment.	Youtube videos limited to (technical reviews and advertisings of automobiles (AUTO) and tablets (TABLETS). To collect the videos, they compiled a list of products	results are reported for sentiment, type and full classification tasks. The metrics used are precision, recall and accuracy for each individual	The largest group of errors are implicit sentiments. some comments do not contain any	Can work on a joint model to classify and the comments of a given video, such that it is possible to exploit latent dependencies

	Katja Filippova Jan 2017		and queried the YouTube gData API6 to retrieve the videos. Manually excluded irrelevant videos. For each video, they extracted 1K comments and manually annotated each comment with its type and polarity	class and the general accuracy of the multi-class classifier (Acc).	explicit positive or negative opinions, but provide detailed and well-argument criticism or irony.	s between entities and the sentiments of the comment thread.
--	--------------------------------	--	---	---	--	--

Gap Identified:

In general, many algorithms and projects depict the use and extraction of comments or likes (hardly both) and also sometimes avoid taking emoticons in comments, which plays a huge part in the sentiment analysis of a particular comment. Here we have taken all these missing data into account and extracted using our API key accordingly. Unlike other projects we hereby have also shown the proper visualisation for our results which is very user friendly and helpful to understand the results at a greater depth.

We have simply used "Youtube Data API" which is available on "Google Developers Console" to scrap youtube comments of a particular video and download them in a pkl format. Then we have made use of python library called "NLTK" (Natural Language Toolkit), a platform for building python programs to work with Human language data. More specifically, what I have used is called VADER (Valence Aware Dictionary and Sentiment Reasoner) which is a lexicon and rule-based sentiment analysis tool that is specifically attuned to sentiments expressed on social media. We have combined this vader lexicon and youtube data api to give a machine generated report on sentiments of comments that are posted (Expressed) on a particular video.

Outcomes:

This project gives us the emotional tone behind a comment in YouTube videos. We have different existing models which can give the amount of likes, dislikes, comments in a YouTube video but our project gives the emotional tone behind the comments on the YouTube videos along with those features. This additional feature which we have added in the project makes it outcomes. Applications: Sentiment analysis tools can be used by organizations for a variety of applications, including:

Identifying brand awareness, reputation and popularity at a specific moment or over time.

Tracking consumer reception of new products or features.

Evaluating the success of a marketing campaign.

Pinpointing the target audience or demographics.

Collecting customer feedback from social media, websites or online forms.

Conducting market research.

Categorizing customer service requests

Schedule of this project:

This project deals with the extraction of the data of any particular video on Youtube platform. It uses the concept of Youtube API keys facilitated by Google to help developers extract data for rightful use. I obtained my own personal key, which has been used for this project. Here I extract the data of any Youtube video, given its video id and extract all the data such as no. of likes, dislikes, comments and views. A list of dataset of emojis also has been used. Further I run sentimental analysis on these comments using a naive bayes classifier under the “nltk” package of Python. I thereafter show the data in form of proper visualisations such as pie chart and Wordcloud. In addition to identifying sentiment, opinion mining can extract the polarity (or the amount of positivity and negativity), subject and opinion holder within the text. Furthermore, sentiment

Technology Used in this Project:

Naïve Bayes Model:

It is a classification technique based on Baye's Theorem with an assumption of independence among predictors. In simple terms, a Naive Bayes classifier assumes that the presence of a particular feature in a class is unrelated to the presence of any other feature. For example, a fruit may be considered to be an apple if it is red, round, and about 3 inches in diameter. Even if these features depend on each other or upon the existence of the other features, all of these properties independently contribute to the probability that this fruit is an apple and that is why it is known as 'Naive'.

Naive Bayes model is easy to build and particularly useful for very large data sets. Along with simplicity, Naive Bayes is known to outperform even highly sophisticated classification methods.

Bayes theorem provides a way of calculating posterior probability $P(c|x)$ from $P(c)$, $P(x)$ and $P(x|c)$. Look at the equation below:

The diagram shows the equation $P(c|x) = \frac{P(x|c)P(c)}{P(x)}$ with arrows pointing from labels to the corresponding parts of the equation. 'Likelihood' points to $P(x|c)$, 'Class Prior Probability' points to $P(c)$, 'Posterior Probability' points to $P(c|x)$, and 'Predictor Prior Probability' points to $P(x)$.

$$P(c|X) = P(x_1|c) \times P(x_2|c) \times \dots \times P(x_n|c) \times P(c)$$

Above,

- $P(c|x)$ is the posterior probability of class (c, target) given predictor (x, attributes).
- $P(c)$ is the prior probability of class.
- $P(x|c)$ is the likelihood which is the probability of predictor given class.
- $P(x)$ is the prior probability of predictor.

How Technology is transforming?

Let's understand it using an example. Below I have a training data set of weather and corresponding target variable 'Play' (suggesting possibilities of playing). Now, we need to classify whether players will play or not based on weather condition. Let's follow the below steps to perform it.

Step 1: Convert the data set into a frequency table.

Step 2: Create Likelihood table by finding the probabilities like Overcast probability = 0.29 and probability of playing is 0.64.

Weather	Play
Sunny	No
Overcast	Yes
Rainy	Yes
Sunny	Yes
Sunny	Yes
Overcast	Yes
Rainy	No
Rainy	No
Sunny	Yes
Rainy	Yes
Sunny	No
Overcast	Yes
Overcast	Yes
Rainy	No

Frequency Table		
Weather	No	Yes
Overcast		4
Rainy	3	2
Sunny	2	3
Grand Total	5	9

Likelihood table				
Weather	No	Yes		
Overcast		4	=4/14	0.29
Rainy	3	2	=5/14	0.36
Sunny	2	3	=5/14	0.36
All	5	9		
	=5/14	=9/14		
	0.36	0.64		

Step 3: Now, use Naive Bayesian equation to calculate the posterior probability for each class. The class with the highest posterior probability is the outcome of prediction.

Problem: Players will play if weather is sunny. Is this statement is correct?

We can solve it using above discussed method of posterior probability.

$$P(\text{Yes} | \text{Sunny}) = P(\text{Sunny} | \text{Yes}) * P(\text{Yes}) / P(\text{Sunny})$$

Here we have $P(\text{Sunny} | \text{Yes}) = 3/9 = 0.33$, $P(\text{Sunny}) = 5/14 = 0.36$, $P(\text{Yes}) = 9/14 = 0.64$ Now,

$$P(\text{Yes} | \text{Sunny}) = 0.33 * 0.64 / 0.36 = 0.60, \text{ which has higher probability.}$$

Naive Bayes uses a similar method to predict the probability of different class based on various attributes. This algorithm is mostly used in text classification and with problems having multiple classes.

What are the Pros and Cons of Naive Bayes?

Pros:

- It is easy and fast to predict class of test data set. It also perform well in multi class prediction
- When assumption of independence holds, a Naive Bayes classifier performs better compare to other models like logistic regression and you need less training data.
- It perform well in case of categorical input variables compared to numerical variable(s). For numerical variable, normal distribution is assumed (bell curve, which is a strong assumption).

Cons:

- If categorical variable has a category (in test data set), which was not observed in training data set, then model will assign a 0 (zero) probability and will be unable to make a prediction. This is often known as “Zero Frequency”. To solve this, we can use the smoothing technique. One of the simplest smoothing techniques is called Laplace estimation.
- On the other side naive Bayes is also known as a bad estimator, so the probability outputs from `predict_proba` are not to be taken too seriously.
- Another limitation of Naive Bayes is the assumption of independent predictors. In real life, it is almost impossible that we get a set of predictors which are completely independent

How to build a basic model using Naive Bayes in Python?

Again, scikit learn (python library) will help here to build a Naive Bayes model in Python. There are three types of Naive Bayes model under scikit learn library:

- **Gaussian:** It is used in classification and it assumes that features follow a normal distribution.
- **Multinomial:** It is used for discrete counts. For example, let's say, we have a text classification problem. Here we can consider bernoulli trials which is one step further and instead of “word occurring in the document”, we have “count how often word occurs in the document”, you can think of it as “number of times outcome number x_i is observed over the n trials”.
- **Bernoulli:** The binomial model is useful if your feature vectors are binary (i.e. zeros and ones). One application would be text classification with ‘bag of words’ model where the 1s & 0s are “word occurs in the document” and “word does not occur in the document” respectively. Based on your data set, you can choose any of above discussed model. Below is the example of Gaussian model.

Based on your data set, you can choose any of above discussed model. Below is the example of Gaussian model.

Bigram Model:

A bigram or diagram is a sequence of two adjacent elements from a string of tokens, which are typically letters, syllables, or words. A bigram is an n -gram for $n=2$. The frequency distribution of every bigram in a string is commonly used for simple statistical analysis of text in many applications, including in computational linguistics, cryptography, speech recognition, and so on.

Gappy bigrams or skipping bigrams are word pairs which allow gaps (perhaps avoiding connecting words, or allowing some simulation of dependencies, as in a dependency grammar).

Head word bigrams are gappy bigrams with an explicit dependency relationship. Bigrams help provide the conditional probability of a token given the preceding token, when the relation of the conditional probability is applied:

That is, the probability $P()$ of a token W_n given the preceding token W_{n-1} is equal to the probability of their bigram, or the co-occurrence of the two tokens $P(W_{n-1}, W_n)$, divided by the probability of the preceding token.

Prototype :

Dataset Description:

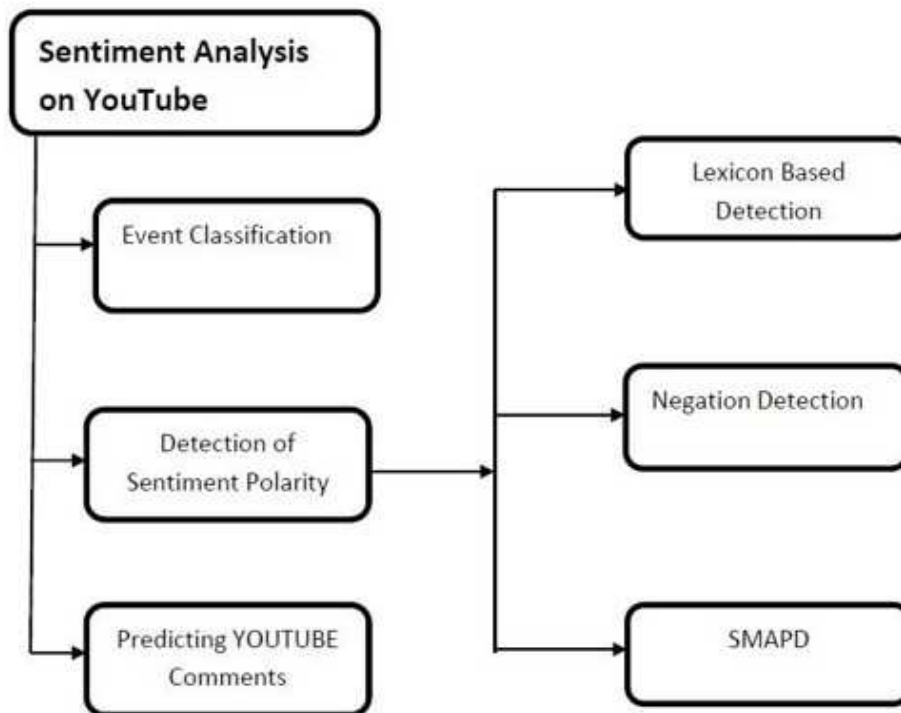
This project outputs data relating to the sentiment of the comments of a YouTube video.

There are 2 files used to achieve this currently:

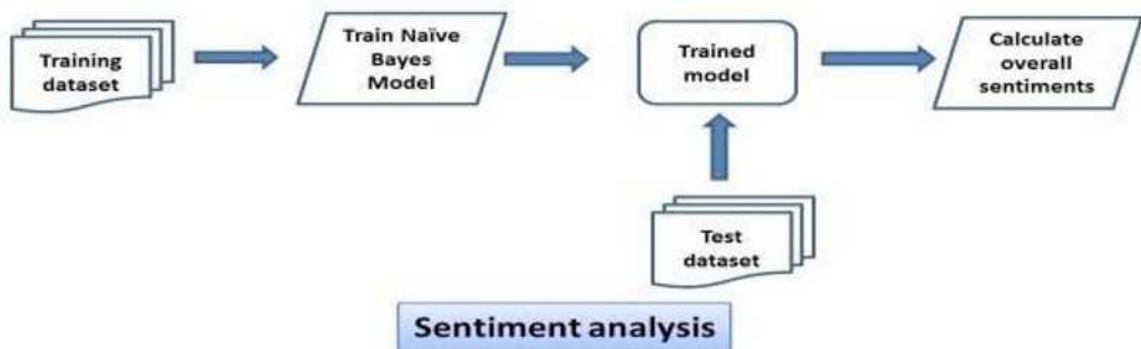
- `download_comments.py` : used to download the comments (threads and individual replies) on a YouTube video
- `analyze_comments.py` : analyzes the resultant data

So far, this script can download the comments of a YouTube video and generate an average of the sentiments of all the comments. This gives us an overall view of how positive/neutral/negative the comments are, which (usually) provides a more accurate metric of the reception of an individual video outside of just likes/dislikes/views.

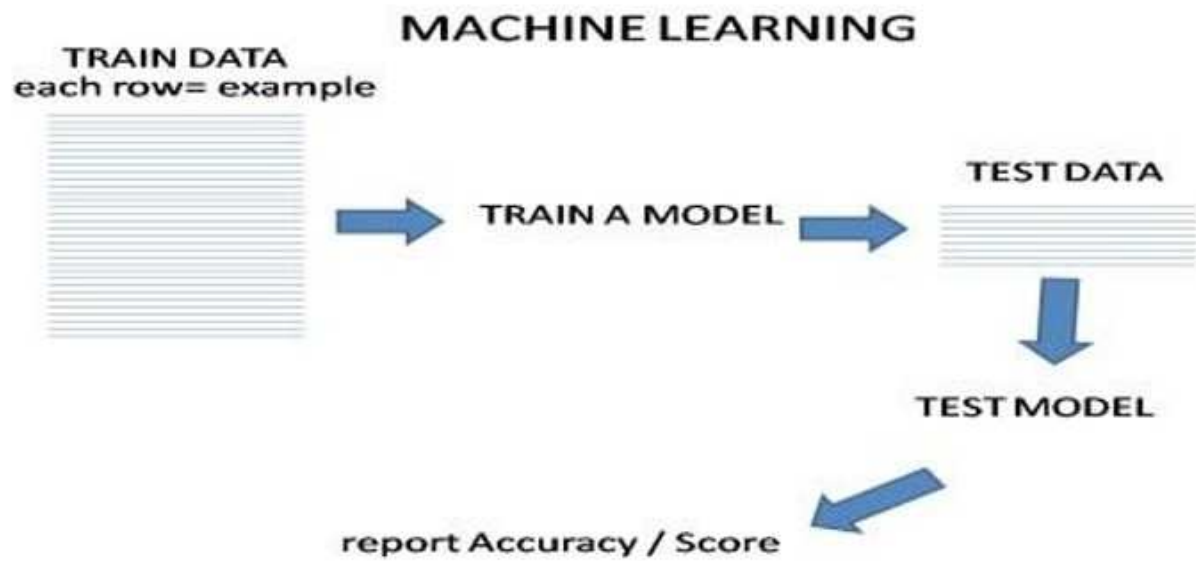
Architecture Diagram:



Module Diagram:



Overview of entire sentimental analysis process:



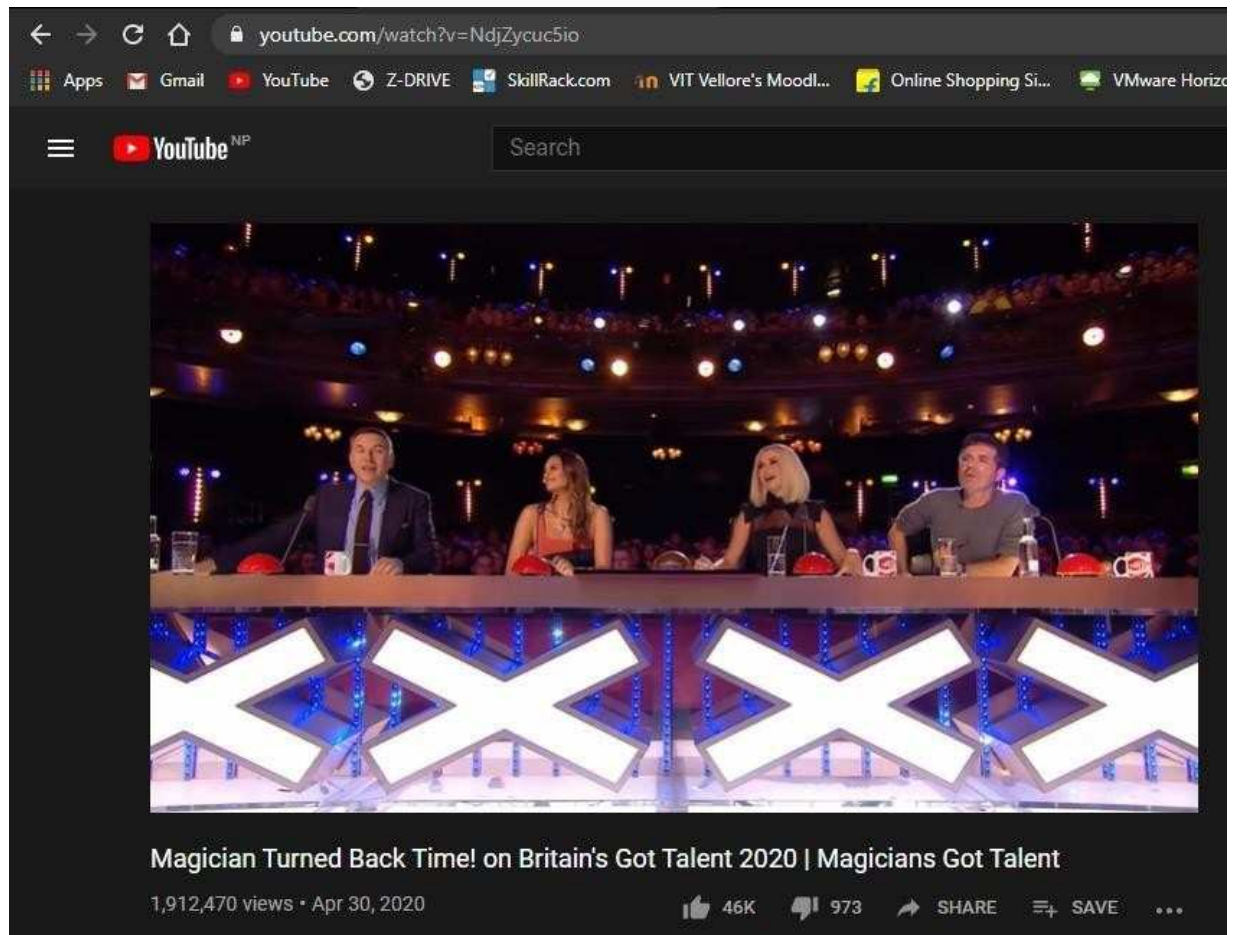
Experimental Setup:

APIs/Dependencies Used:

- sklearn
- nltk
- pandas
- YouTube API

Implementation aspects and verification:

Video ID: NdjZycuc5io



Output:

```
comp:0.0634399151436032, pos:0.10415567885117502, neu:0.8173932767624014, neg:0.07452774151436035
Comment      Compound  Positive  Neutral  Negative
0   This is a simple trick. He uses Kostya Kimlats... -0.4019    0.000    0.901    0.099
1   I want to go back to my childhood. Pls help me...  0.5106    0.398    0.602    0.000
2       it's evolving, just backwards                0.0000    0.000    1.000    0.000
3   Plot twist : the actual time at that time is 1...  0.0000    0.000    1.000    0.000
4       scripted                                     0.0000    0.000    1.000    0.000
...          ...          ...          ...          ...
3059          Duka atuh  0.0000    0.000    1.000    0.000
3060 @Rupen Maharjan If you watched the first episo...  0.0000    0.000    1.000    0.000
3061     it was shooted before the pandemic.. i guess  0.0000    0.000    1.000    0.000
3062          ah, the alpha c***  0.0000    0.000    1.000    0.000
3063          MACHA OFFICIAL nope  0.0000    0.000    1.000    0.000

[3064 rows x 5 columns]

#####Generating Report#####

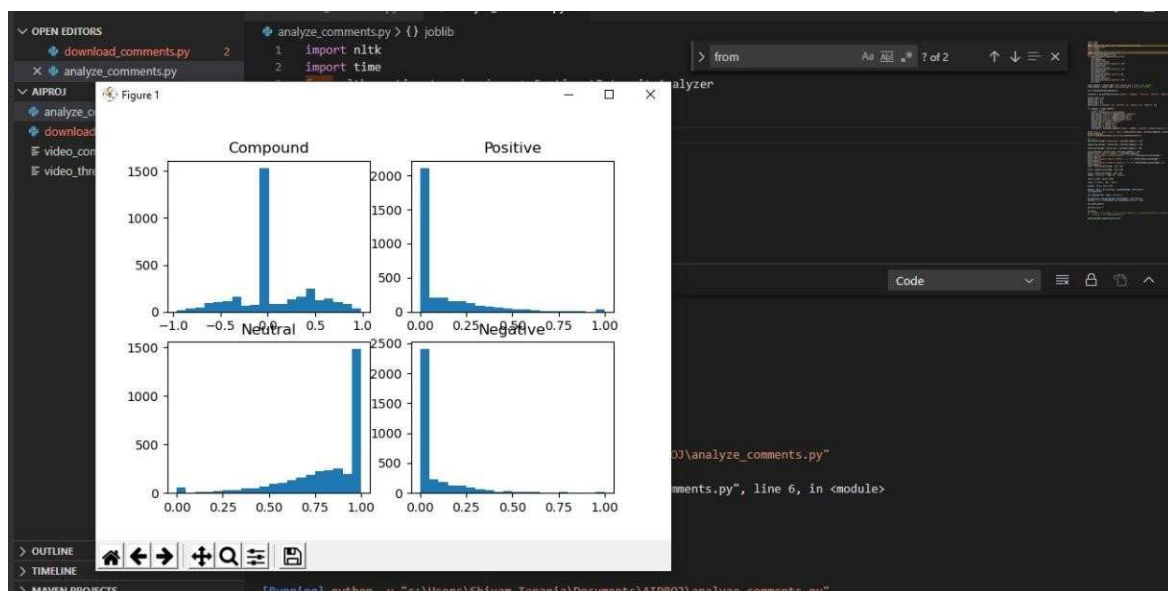
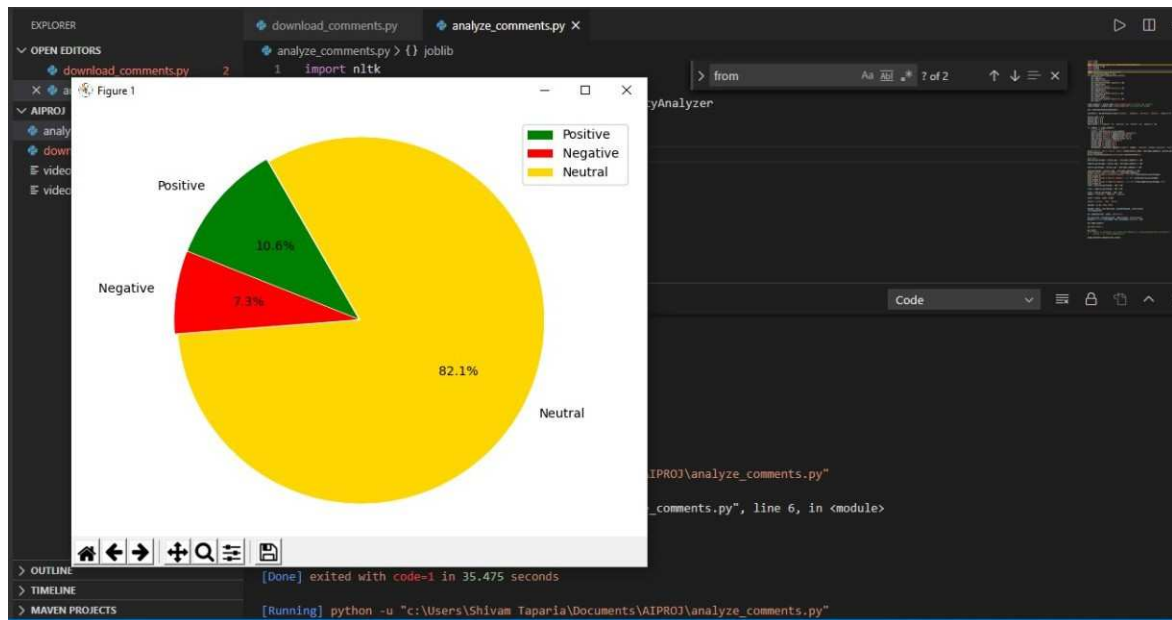
ToTal number of Comments:  3064

Percentage of Positive Comments:  10.42

Percentage of Neutral Comments:  81.74

Percentage of Negative Comments:  7.45
```

The basic idea behind sentiment analysis using vader lexicon is that it contains a dictionary of words with some value assigned to it. Eg a word like "Good" or "Amazing" would have some Positive value assigned to it and a word like "Bad" or "sad" would have a Negative value assigned to it. So what I did is that I made a program that reads through the lines from a CSV file that contains all the comments on a particular youtube video and then calculate Compound Score for each line



Challenges with sentiment analysis:

Challenges associated with sentiment analysis typically revolve around inaccuracies

in training models. Objectivity, or comments with a neutral sentiment, tend to pose a problem for systems and are often misidentified. For example, if a customer received the wrong color item and submitted a comment "The product was blue," this would be identified as neutral when in fact it should be negative.

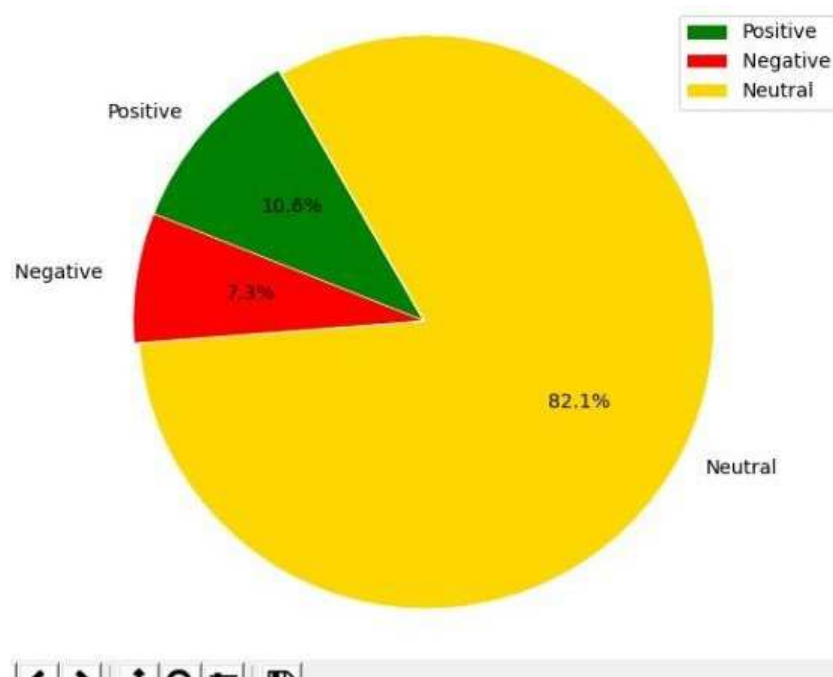
Sentiment can also be challenging to identify when systems cannot understand the context or tone. Answers to polls or survey questions like "nothing" or "everything" are hard to categorize when the context is not given, as they could be labeled as positive or negative depending on the question. Similarly, irony and sarcasm often cannot be explicitly trained and lead to falsely labeled sentiments.

Computer programs also have trouble when encountering emojis and irrelevant information. Special attention needs to be given to training models with emojis and neutral data so as to not improperly flag texts.

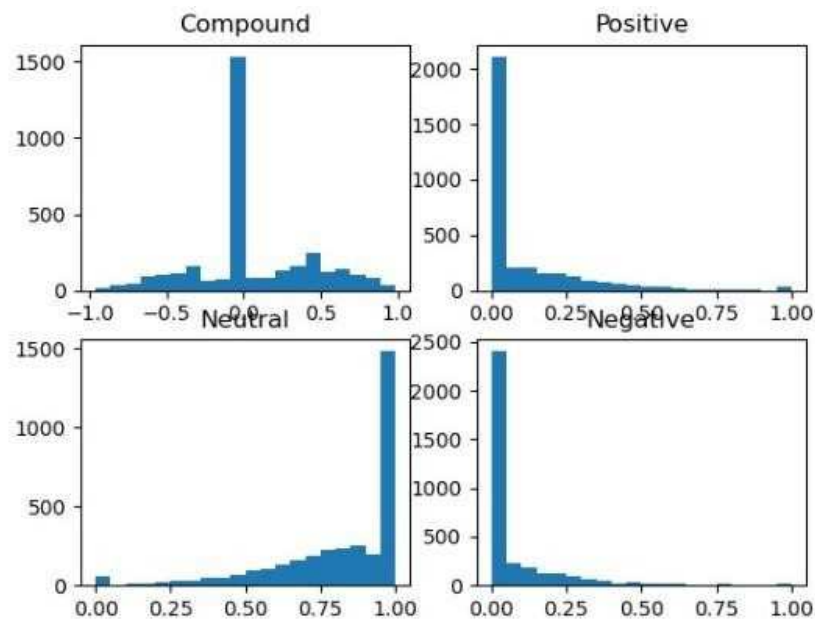
Finally, people can be contradictory in their statements. Most reviews will have both positive and negative comments, which is somewhat manageable by analyzing sentences one at a time. However, the more informal the medium (Twitter or blog posts, for example), the more likely people are to combine different opinions in the same sentence and the more difficult it will be for a computer to parse.

Result and obtain validation:

Pie Chart:



Sentiment Subplots Graph:



Conclusion:

Considering the above reflection on the project's planning, it can be concluded that the scope of the project was met within the timeline set. As such, the desired classification accuracy of 80% or more is reached. The reporting is performed in real time and the graphical user interface offers an intuitive, yet comprehensive user interaction. Overall, the project was a good opportunity to further hone my programming skills and expand my knowledge in the fields of natural language processing and machine learning.

References:

1. H. Soong, N. B. A. Jalil, R. Kumar Ayyasamy and R. Akbar, "The Essential of Sentiment Analysis and Opinion Mining in Social Media : Introduction and Survey of the Recent Approaches and Techniques," 2019 IEEE 9th Symposium on Computer Applications & Industrial Electronics (ISCAIE), Malaysia, 2019, pp. 272-277, doi: 10.1109/ISCAIE.2019.8743799.
2. Jandail, R.R., Sharma, P., & Agrawal, C. (2014). A Survey on Sentiment Analysis and Opinion Mining: A need for an Organization and Requirement of a customer.
3. Asghar, Dr. Muhammad & Ahmad, Shakeel & Marwat, Afsana & Kundi, Fazal. (2015). Sentiment Analysis on YouTube: A Brief Survey. Brist. 3. 1250-1257.
4. Ravi, Kumar. (2015). A survey on opinion mining and sentiment analysis:

Tasks, approaches and applications. Knowledge-Based Systems. 89. 14-46. 10.1016/j.knosys.2015.06.015.

5. Bhargava, M.G., & Rao, D.R. (2018). Sentimental analysis on social media data using R programming. International journal of engineering and technology, 7, 80.
6. Patel, Vishakha & Prabhu, Gayatri & Bhowmick, Kiran. (2015). A Survey of Opinion Mining and Sentiment Analysis. International Journal of Computer Applications. 131. 24-27. 10.5120/ijca2015907218.
7. Liu, B., & Zhang, L.J. (2012). A Survey of Opinion Mining and Sentiment Analysis. Mining Text Data.
8. Z. Nanli, Z. Ping, L. Weiguo and C. Meng, "Sentiment analysis: A literature review," 2012 International Symposium on Management of Technology (ISMOT), Hangzhou, 2012, pp. 572576, doi: 10.1109/ISMOT.2012.6679538.
9. Tang, H., Tan, S., & Cheng, X. (2009). A survey on sentiment detection of reviews. Expert Syst. Appl., 36, 10760-10773.

10. Jain, S.K., & Singh, P. (2018). Systematic Survey on Sentiment Analysis. 2018 First International Conference on Secure Cyber Computing and Communication (ICSCCC), 561-565.

<https://www.analyticsvidhya.com/blog/2017/09/naive-bayes-explained/>

<https://en.wikipedia.org/wiki/Bigram>

<https://developers.google.com/youtube/v3/docs/comments/list>

CODE

`download_comment.py`

`# coding: utf-8 from apiclient.discovery`

`import build`

`#from apiclient.errors import HttpError`

`#from oauth2client.tools import argparser`

`import pickle`

`#from sklearn.externals import joblib`

`import config`

`DEVELOPER_KEY = "API Daal Edhar"`

`YOUTUBE_API_SERVICE_NAME = "youtube"`

`YOUTUBE_API_VERSION = "v3"`

`#nltk.download('vader_lexicon')`

`youtube = build(YOUTUBE_API_SERVICE_NAME, YOUTUBE_API_VERSION,
developerKey=DEVELOPER_KEY)`

`def get_video_comments(video_id): # returns a list of comments given a video id`

`threads = []`

`comments = []`

`comment_results = youtube.commentThreads().list(part='snippet',
videoId=video_id, textFormat='plainText', maxResults=100,).execute()`

`for thread in comment_results['items']:`

`threads.append(thread)`

`comments.append(thread['snippet']['topLevelComment']['snippet']['textDisplay'])`

```

while 'nextPageToken' in comment_results: # as long as another page is
accessible, keep adding to threads and comments

    comment_results = youtube.commentThreads().list( part='snippet',
videoId=video_id, textFormat='plainText', maxResults=100,
pageToken=comment_results['nextPageToken'] ).execute()

    for thread in comment_results['items']:

        threads.append(thread)

        comments.append(thread['snippet']['topLevelComment']['snippet']['textDisplay'])

    for thread in threads: # get comments out of top level comment threads (replies)
        if(thread['snippet']['totalReplyCount'] > 0):

            replies = youtube.comments().list( part='snippet', parentId =
thread['id'] ).execute()

            for reply in replies['items']: # print(reply) comments.append(reply['snippet']
['textDisplay'])

                print('Found {} threads with a total of {} comments'.format(len(threads),
len(comments)))

            return comments, threads

video_comments, video_threads = get_video_comments('NdjZycuc5io')
joblib.dump(video_comments, 'video_comments.pkl') # pickle the comments
joblib.dump(video_threads, 'video_threads.pkl') # pickle the threads
print('PICKLING IS COMPLETE')

```

analyze_comment.py

```
import nltk
import time

from nltk.sentiment.vader import SentimentIntensityAnalyzer

import pandas as pd

import pickle

from sklearn.externals import joblib

import matplotlib.pyplot as plt

def graph_sentiment_subplots(sent_scores):

    plt.figure()

    plt.subplot(221)

    plt.title('Compound')

    plt.hist(sent_scores['compound'])

    plt.subplot(222)

    plt.title('Positive')

    plt.hist(sent_scores['positive'])

    plt.subplot(223)

    plt.title('Neutral')

    plt.hist(sent_scores['neutral'])

    plt.subplot(224)

    plt.title('Negative')

    plt.hist(sent_scores['negative'])

    plt.show()

video_comments = joblib.load('video_comments.pkl') # pickle the comments

video_threads = joblib.load('video_threads.pkl') # pickle the threads

sid = SentimentIntensityAnalyzer()

sentiments = pd.DataFrame(columns=['Comment', 'Compound', 'Positive', 'Neutral', 'Negative'])

overall_comp = 0.0
```

```

overall_pos = 0.0
overall_neu = 0.0
overall_neg = 0.0
sent_scores = {'compound': [], 'positive': [], 'neutral': [], 'negative': []}
for comment in video_comments:
    # print(comment) scores =
    sid.polarity_scores(comment)
    sent_scores['compound'].append(scores['compound'])
    sent_scores['positive'].append(scores['pos'])
    sent_scores['neutral'].append(scores['neu'])
    sent_scores['negative'].append(scores['neg'])
    overall_comp += scores['compound']
    overall_pos += scores['pos']
    overall_neu += scores['neu']
    overall_neg += scores['neg']

    sentiments = sentiments.append({'Comment': comment, 'Compound':
scores['compound'], 'Positive': scores['pos'], 'Neutral': scores['neu'], 'Negative':
scores['neg']}, ignore_index=True)

print('comp:{}, pos:{}, neu:{}, neg:{}'.format(overall_comp / len(video_comments),
overall_pos / len(video_comments), overall_neu / len(video_comments), overall_neg
/ len(video_comments))) print(sentiments)

print("\n#####Generating Report#####\n") #pie chart
positive_percentage = overall_pos / len(video_comments) * 100
negative_percentage = overall_neg / len(video_comments) * 100
neutral_percentage = overall_neu / len(video_comments) * 100
comp_percentage = overall_comp / len(video_comments) * 100
print("\nTotal number of Comments: ",len(video_comments))
print("\nPercentage of Positive Comments: ",positive_percentage)
print("\nPercentage of Neutral Comments: ",neutral_percentage)
print("\nPercentage of Negative Comments: ",negative_percentage)

```



```
size1 = positive_percentage / 100 * 360
size2 = negative_percentage / 100 * 360
size3 = neutral_percentage / 100 * 360
labels = 'Positive', 'Negative ', 'Neutral'
sizes = [size1, size2, size3]
colors = ['Green', 'Red', 'gold']
explode = (0.1, 0.01, 0.01)
plt.pie(sizes, explode=explode, colors=colors ,startangle=120)
plt.show()
graph_sentiment_subplots(sent_scores)
```

THANK YOU



SENTIMENTAL ANALYSIS AND OPINION MINING YOUTUBE DATA

Group Members

Samridhi Jhijaria – 19BCE2470

Diva Bhatia – 19BCE2452

SLOT – C1+TC1

Artificial Intelligence – CSE3013(Project Component)

Abstract:

This project deals with the extraction of the data of any particular video on Youtube platform. It uses the concept of Youtube API keys facilitated by Google to help developers extract data for rightful use. We obtained my own personal key, which has been used for this project. Here I extract the data of any Youtube video, given its video id and extract all the data such as no. of likes, dislikes, comments and views.

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. It involves the use of data mining, machine learning (ML) and artificial intelligence (AI) to mine text for sentiment and subjective

Literature survey:

S.No.	Title of the Paper And year	Algorithm used	Data set being used	Performance measures	Gap identified	Scope for future work
1.	Sentiment Analysis - Mhd. Majid Akhtar June 2019	TextBlob Sentiment (python) To check polarity.	"Liverpool FC- The road to Madrid- UCL2019" by channel MRCLFCompilations. ID of the video is Y-XHmlaJL-s. Download Dataset of Youtube Video comments via running a script Convert Json file to CSV file via convert_json_to_csv.ipynb file Run 'sentiment_analysis' python file and the results showing positive, negative and neutral response & knowing the accuracy of model.	ACCURACY: 70% PRECISION: 100% RECALL: 75% In finding and categorizing positive, negative and neutral comments.	The dataset is not too clean.	The dataset is not too clean. The proper way should be to tokenize it, exclude stop words and generate a bag of words from it. These steps were ignored in this project.
2.	Polarity trend analysis of public sentiment on YouTube - Amar Krishna 2019	Naive Bayes classification technique. Bayesian probability	Internet Movie Database(IMDb) dataset designed and developed by Pang and Lee	Results show that the trends in users' sentiments is well correlated to the real-world events associated with the respective keywords. Using the Weka forecasting tool, they were able to predict the possible sentiment scores for 26 weeks into the future with a confidence interval of 95%.	focused on the comment ratings and their dependencies to topics	Can study the sentiment trends in YouTube comments, with focus on the popular/trending keywords

Gap Identified:

In general, many algorithms and projects depict the use and extraction of comments or likes (hardly both) and also sometimes avoid taking emoticons in comments, which plays a huge part in the sentiment analysis of a particular comment. Here we have taken all these missing data into account and extracted using our API key accordingly. Unlike other projects we hereby have also shown the proper visualisation for our results which is very user friendly and helpful to understand the results at a greater depth. We have simply used "Youtube Data API" which is available on "Google Developers Console" to scrap youtube comments of a particular video and download them in a pkl format. Then we have made use of python library called "NLTK" (Natural Language Toolkit), a platform for building python programs to work with Human language data. More specifically, what I have used is called VADER (Valence Aware Dictionary and Sentiment Reasoner) which is a lexicon and rule-based sentiment analysis tool that is

Outcomes:

This project gives us the emotional tone behind a comment in YouTube videos. We have different existing models which can give the amount of likes, dislikes, comments in a YouTube video but our project gives the emotional tone behind the comments on the YouTube videos along with those features. This additional feature which we have added in the project makes it outcomes. Applications: Sentiment analysis tools can be used by organizations for a variety of applications, including:

- Identifying brand awareness, reputation and popularity at a specific moment or over time.

- Tracking consumer reception of new products or features.

- Evaluating the success of a marketing campaign.

- Pinpointing the target audience or demographics.

Schedule of this project:

This project deals with the extraction of the data of any particular video on Youtube platform. It uses the concept of Youtube API keys facilitated by Google to help developers extract data for rightful use. I obtained my own personal key, which has been used for this project. Here I extract the data of any Youtube video, given its video id and extract all the data such as no. of likes, dislikes, comments and views. A list of dataset of emojis also has been used. Further I run sentimental analysis on these comments using a naive bayes classifier under the “nltk” package of Python. I thereafter show the data in form of proper visualisations such as pie chart and Wordcloud. In addition to identifying sentiment, opinion mining can extract the polarity (or the amount of positivity and negativity), subject and opinion holder within the text. Furthermore, sentiment

Technology Used in this Project:

Naïve Bayes Model:

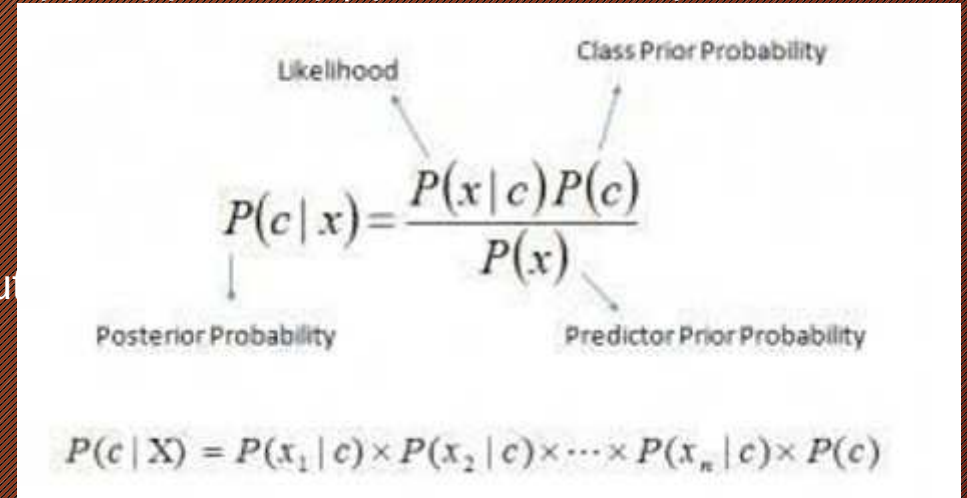
It is a classification technique based on Baye's Theorem with an assumption of independence among predictors. In simple terms, a Naive Bayes classifier assumes that the presence of a particular feature in a class is unrelated to the presence of any other feature. For example, a fruit may be considered to be an apple if it is red, round, and about 3 inches in diameter. Even if these features depend on each other or upon the existence of the other features, all of these properties independently contribute to the probability that this fruit is an apple and that is why it is known as 'Naive'.

Naive Bayes model is easy to build and particularly useful for very large data sets. Along with simplicity, Naive Bayes is known to outperform even highly sophisticated classification methods.

Bayes theorem provides a way of calculating posterior probability $P(c|x)$ from $P(c)$, $P(x)$ and $P(x|c)$. Look at the equation below:

Here,

- $P(c|x)$ is the posterior probability of class (c , target) given predictor (x , attribute).
- $P(c)$ is the prior probability of class.
- $P(x|c)$ is the likelihood which is the probability of predictor given class.
- $P(x)$ is the prior probability of predictor.


$$P(c|x) = \frac{P(x|c)P(c)}{P(x)}$$
$$P(c|X) = P(x_1|c) \times P(x_2|c) \times \dots \times P(x_n|c) \times P(c)$$

How Technology is transforming?

Let's understand it using an example. Below I have a training data set of weather and corresponding target variable 'Play' (suggesting possibilities of playing). Now, we need to classify whether players will play or not based on weather condition. Let's follow the below steps to perform it.

Step 1: Convert the data set into a frequency table.

Step 2: Create Likelihood table by finding the probabilities like Overcast probability = 0.29 and probability of playing is 0.64.

Step 3: Now, use Naive Bayesian equation to calculate the posterior probability for each class. The class with the highest posterior probability is the outcome of prediction.

Problem: Players will play if weather is sunny. Is this statement is correct?

Weather	Play
Sunny	No
Overcast	Yes
Rainy	Yes
Sunny	Yes
Sunny	Yes
Overcast	Yes
Rainy	No
Rainy	No
Sunny	Yes
Rainy	Yes
Sunny	No
Overcast	Yes
Overcast	Yes
Rainy	No

Frequency Table		
Weather	No	Yes
Overcast		4
Rainy	3	2
Sunny	2	3
Grand Total	5	9

Likelihood table				
Weather	No	Yes		
Overcast		4	=4/14	0.29
Rainy	3	2	=5/14	0.36
Sunny	2	3	=5/14	0.36
All	5	9		
	=5/14	=9/14		
	0.36	0.64		

What are the Pros and Cons of Naive Bayes?

Pros:

- It is easy and fast to predict class of test data set. It also perform well in multi class prediction
- When assumption of independence holds, a Naive Bayes classifier performs better compare to other models like logistic regression and you need less training data.
- It perform well in case of categorical input variables compared to numerical variable(s). For numerical variable, normal distribution is assumed (bell curve, which is a strong assumption).

Cons:

- If categorical variable has a category (in test data set), which was not observed in training data set, then model will assign a 0 (zero) probability and will be unable to make a prediction. This is often known as “Zero Frequency”. To solve this, we can use the smoothing technique. One of the simplest smoothing techniques is called Laplace estimation.
- On the other side naive Bayes is also known as a bad estimator, so the probability outputs from `predict_proba` are not to be taken too seriously.
- Another limitation of Naive Bayes is the assumption of independent predictors. In real life, it is almost impossible that we get a set of predictors which are completely independent

How to build a basic model using Naive Bayes in Python?

Again, scikit learn (python library) will help here to build a Naive Bayes model in Python. There are three types of Naive Bayes model under scikit learn library:

- **Gaussian:** It is used in classification and it assumes that features follow a normal distribution.
- **Multinomial:** It is used for discrete counts. For example, let's say, we have a text classification problem. Here we can consider bernoulli trials which is one step further and instead of "word occurring in the document", we have "count how often word occurs in the document", you can think of it as "number of times outcome number x_i is observed over the n trials".
- **Bernoulli:** The binomial model is useful if your feature vectors are binary (i.e. zeros and ones). One application would be text classification with 'bag of words' model where the 1s & 0s are "word occurs in the document" and "word does not occur in the document" respectively. Based on your data set, you can choose any of above discussed model.

Bigram Model:

A bigram or digram is a sequence of two adjacent elements from a string of tokens, which are typically letters, syllables, or words. A bigram is an n -gram for $n=2$. The frequency distribution of every bigram in a string is commonly used for simple statistical analysis of text in many applications, including in computational linguistics, cryptography, speech recognition, and so on.

Gappy bigrams or skipping bigrams are word pairs which allow gaps (perhaps avoiding connecting words, or allowing some simulation of dependencies, as in a dependency grammar).

Head word bigrams are gappy bigrams with an explicit dependency relationship. Bigrams help provide the conditional probability of a token given the preceding token, when the relation of the conditional probability is applied:

That is, the probability $P()$ of a token W_n given the preceding token W_{n-1} is equal to the probability of their bigram, or the co-occurrence of the two tokens $P(W_{n-1}, W_n)$, divided by the probability of the preceding token.

Prototype :

Dataset Description:

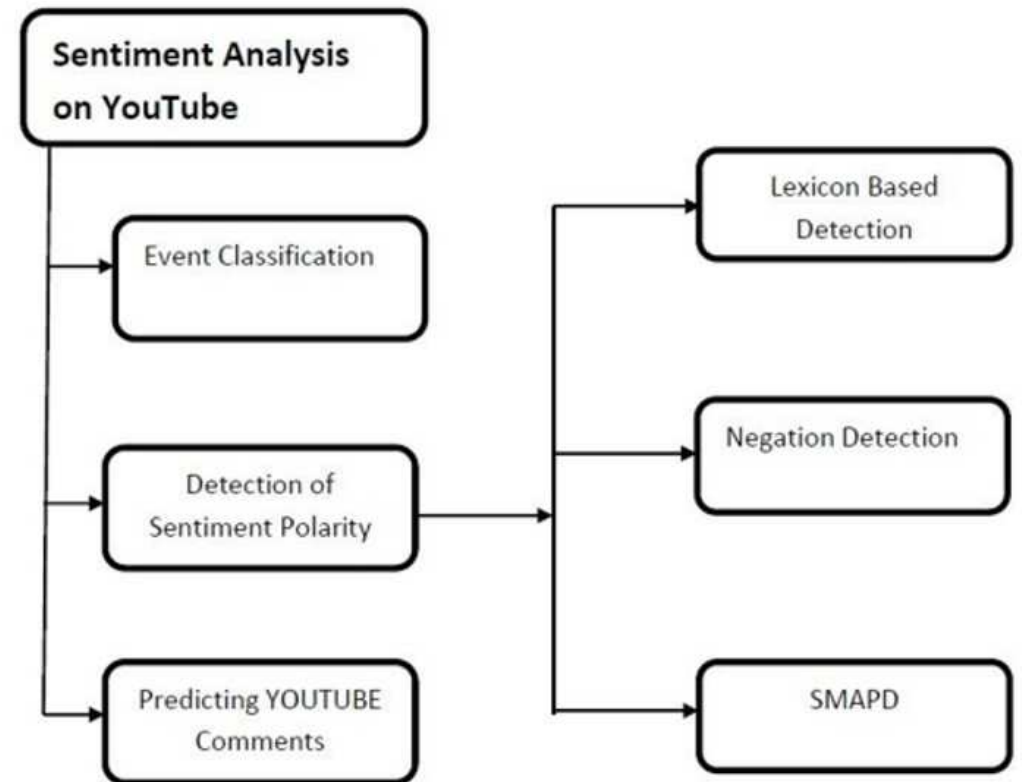
This project outputs data relating to the sentiment of the comments of a YouTube video.

There are 2 files used to achieve this currently:

- `download_comments.py` : used to download the comments (threads and individual replies) on a YouTube video
- `analyze_comments.py` : analyzes the resultant data

So far, this script can download the comments of a YouTube video and generate an average of the sentiments of all the comments. This gives us an overall view of how positive/neutral/negative the comments are, which (usually) provides a more accurate metric of the reception of an individual video outside of just likes/dislikes/views.

Architecture Diagram:

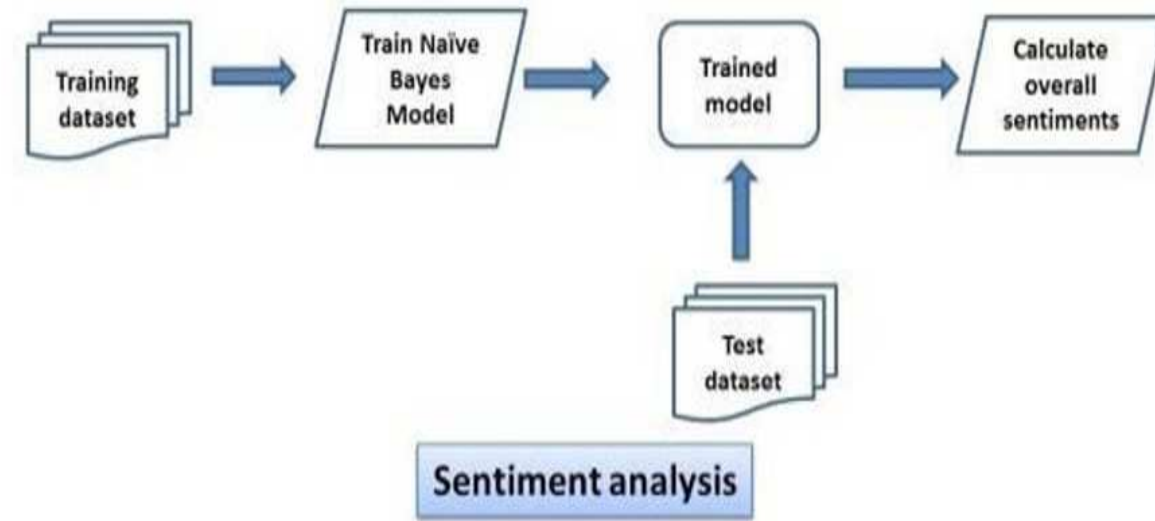


Experimental Setup:

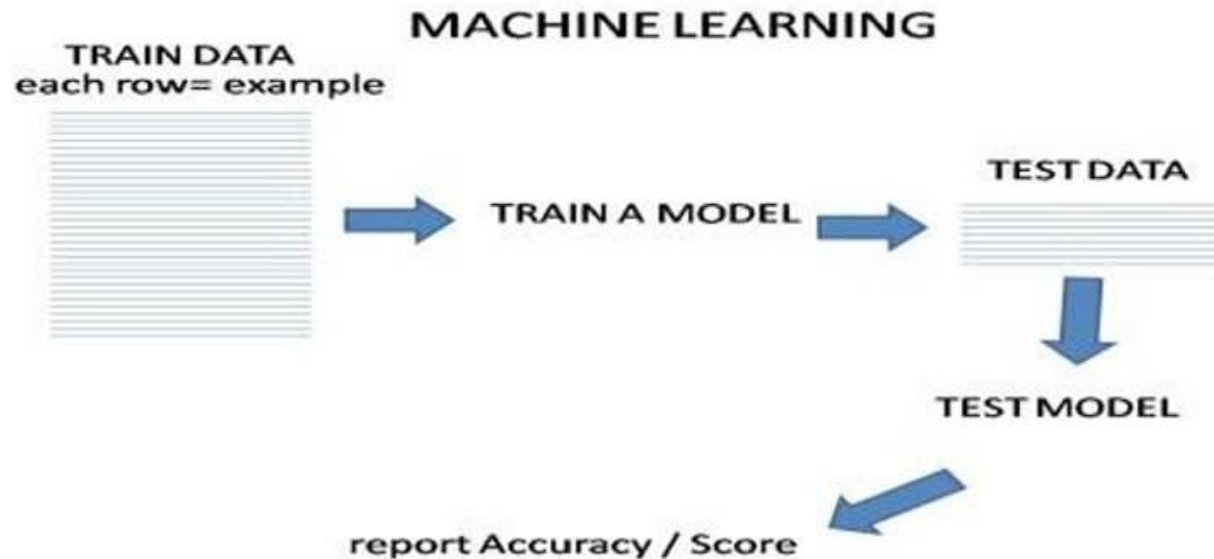
APIs/Dependencies Used:

- sklearn
- nltk
- pandas
- YouTube API

Module Diagram:



Overview of entire sentimental analysis process:



Implementation aspects and verification:

Video ID: NdjZycuc5io



Magician Turned Back Time! on Britain's Got Talent 2020 | Magicians Got Talent

1,912,470 views · Apr 30, 2020



46K



973



SHARE



SAVE



OUTPUT

comp:0.0634399151436032, pos:0.10415567885117502, neu:0.8173932767624014, neg:0.07452774151436035

	Comment	Compound	Positive	Neutral	Negative
0	This is a simple trick. He uses Kostya Kimlats...	-0.4019	0.000	0.901	0.099
1	I want to go back to my childhood. Pls help me...	0.5106	0.398	0.602	0.000
2	it's evolving, just backwards	0.0000	0.000	1.000	0.000
3	Plot twist : the actual time at that time is 1...	0.0000	0.000	1.000	0.000
4	scripted	0.0000	0.000	1.000	0.000
...
3059	Duka atuh	0.0000	0.000	1.000	0.000
3060	@Rupen Maharjan If you watched the first episo...	0.0000	0.000	1.000	0.000
3061	it was shooted before the pandemic.. i guess	0.0000	0.000	1.000	0.000
3062	ah, the alpha c***	0.0000	0.000	1.000	0.000
3063	MACHA OFFICIAL nope	0.0000	0.000	1.000	0.000

[3064 rows x 5 columns]

#####Generating Report#####

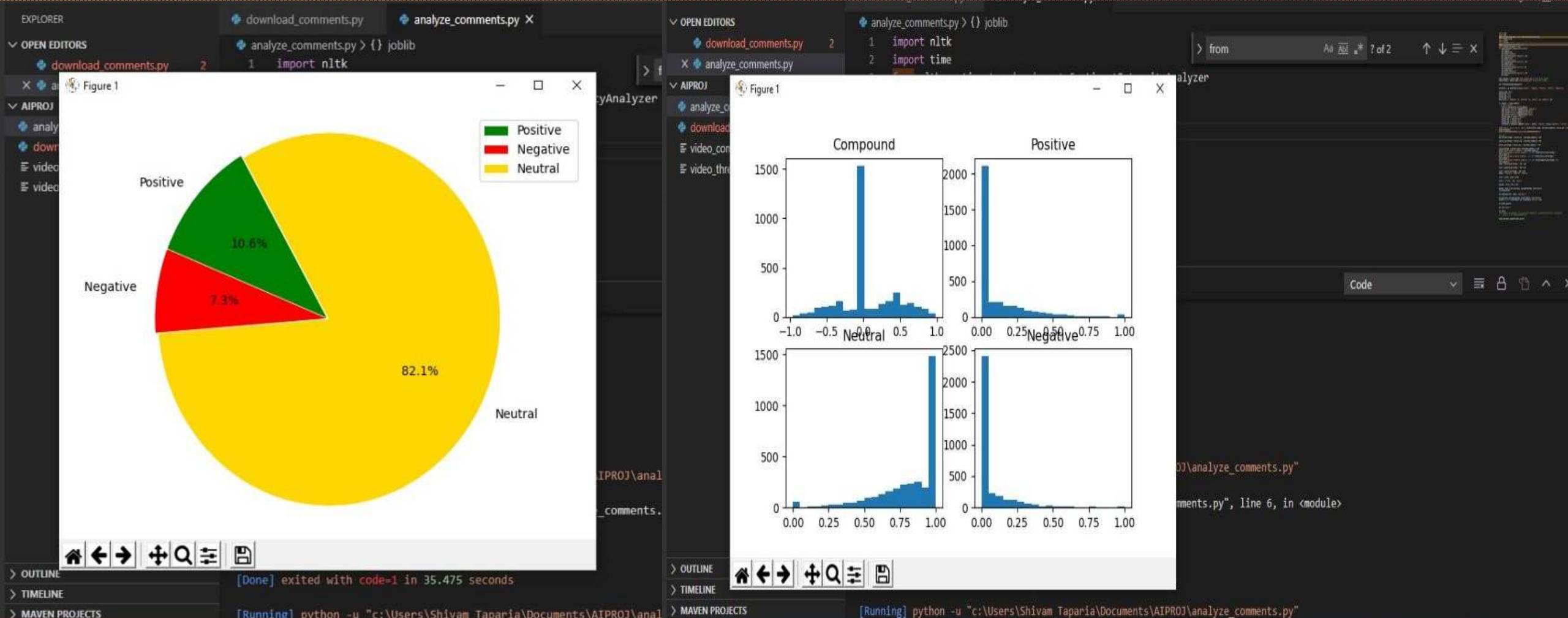
Total number of Comments: 3064

Percentage of Positive Comments: 10.42

Percentage of Neutral Comments: 81.74

Percentage of Negative Comments: 7.45

The basic idea behind sentiment analysis using vader lexicon is that it contains a dictionary of words with some value assigned to it. Eg a word like "Good" or "Amazing" would have some Positive value assigned to it and a word like "Bad" or "sad" would have a Negative value assigned to it. So what I did is that I made a program that reads through the lines from a CSV file that contains all the comments on a particular youtube video and then calculate



Challenges with sentiment analysis:

Challenges associated with sentiment analysis typically revolve around inaccuracies in training models. Objectivity, or comments with a neutral sentiment, tend to pose a problem for systems and are often misidentified. For example, if a customer received the wrong color item and submitted a comment "The product was blue," this would be identified as neutral when in fact it should be negative.

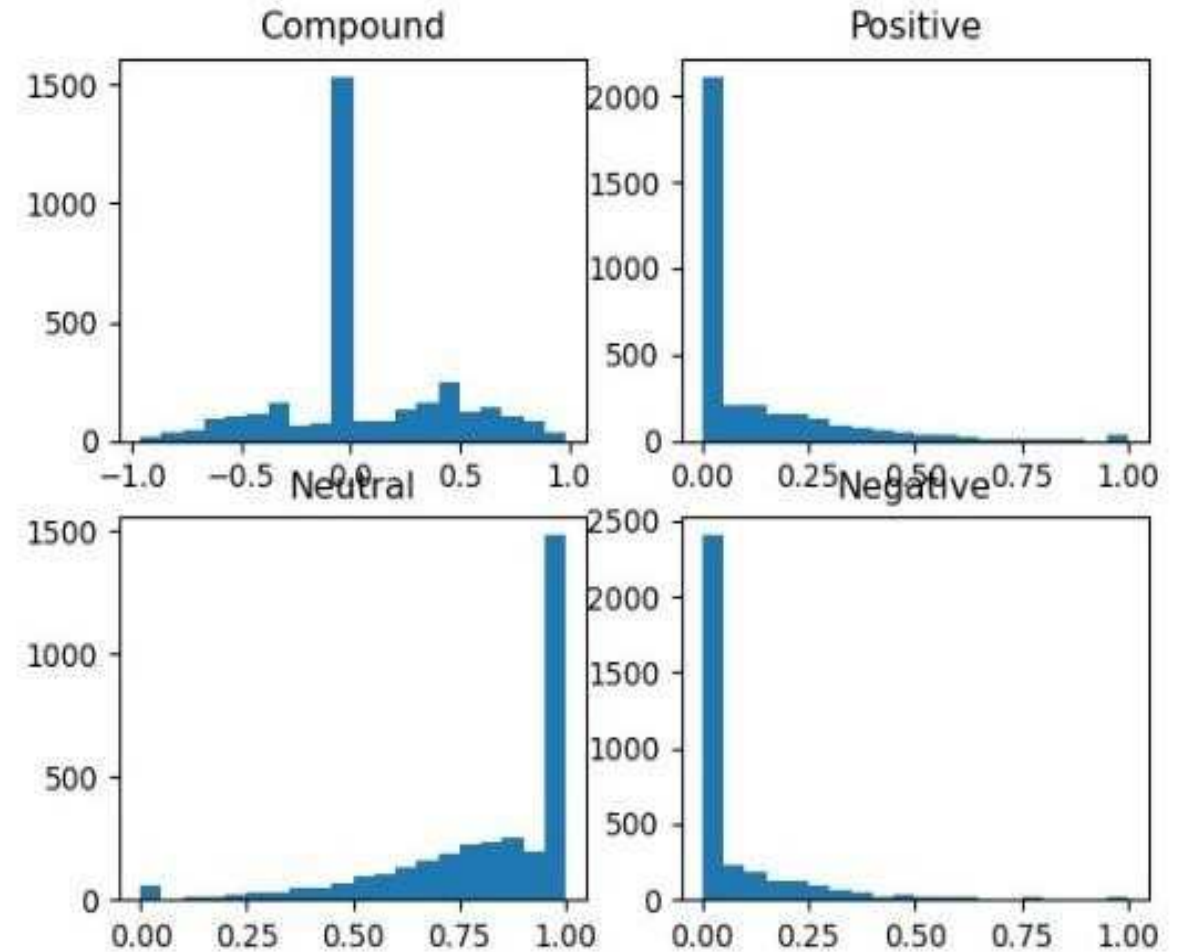
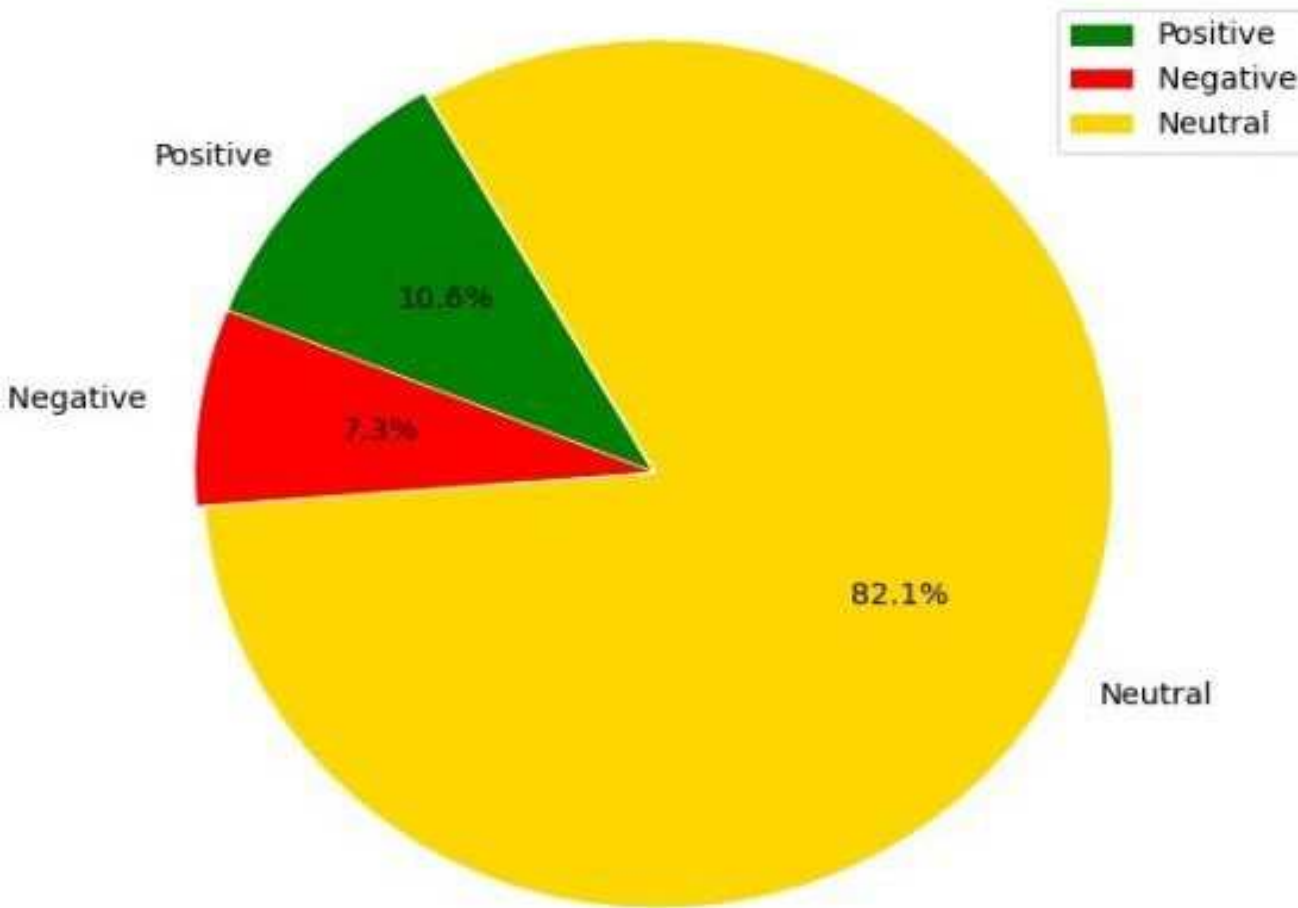
Sentiment can also be challenging to identify when systems cannot understand the context or tone. Answers to polls or survey questions like "nothing" or "everything" are hard to categorize when the context is not given, as they could be labeled as positive or negative depending on the question. Similarly, irony and sarcasm often cannot be explicitly trained and lead to falsely labeled sentiments.

Computer programs also have trouble when encountering emojis and irrelevant information. Special attention needs to be given to training models with emojis and neutral data so as to not improperly flag texts.

Finally, people can be contradictory in their statements. Most reviews will have both positive and negative comments, which is somewhat manageable by analyzing sentences one at a time. However, the more informal the medium (Twitter or blog

Result and obtain validation:

Pie Chart AND Sentiment Subplots Graph



Conclusion:

Considering the above reflection on the project's planning, it can be concluded that the scope of the project was met within the timeline set. As such, the desired classification accuracy of 80% or more is reached. The reporting is performed in real time and the graphical user interface offers an intuitive, yet comprehensive user interaction. Overall, the project was a good opportunity to further hone my programming skills and expand my knowledge in the fields of natural language processing and machine learning.

References:

H. Soong, N. B. A. Jalil, R. Kumar Ayyasamy and R. Akbar, "The Essential of Sentiment Analysis and Opinion Mining in Social Media : Introduction and Survey of the Recent Approaches and Techniques," 2019 IEEE 9th Symposium on Computer Applications & Industrial Electronics (ISCAIE), Malaysia, 2019, pp. 272-277, doi: 10.1109/ISCAIE.2019.8743799.

Jandail, R.R., Sharma, P., & Agrawal, C. (2014). A Survey on Sentiment Analysis and Opinion Mining: A need for an Organization and Requirement of a customer.

Asghar, Dr. Muhammad & Ahmad, Shakeel & Marwat, Afsana & Kundi, Fazal. (2015). Sentiment Analysis on YouTube: A Brief Survey. Brist. 3. 1250-1257.

Ravi, Kumar. (2015). A survey on opinion mining and sentiment analysis: Tasks, approaches and applications. Knowledge-Based Systems. 89. 14-46. 10.1016/j.knosys.2015.06.015.

Bhargava, M.G., & Rao, D.R. (2018). Sentimental analysis on social media data using R programming. International journal of engineering and technology, 7, 80.

Patel, Vishakha & Prabhu, Gayatri & Bhowmick, Kiran. (2015). A Survey of Opinion Mining and Sentiment Analysis. International Journal of Computer Applications. 131. 24-27. 10.5120/ijca2015907218.

Liu, B., & Zhang, L.J. (2012). A Survey of Opinion Mining and Sentiment Analysis. Mining Text Data.

Z. Nanli, Z. Ping, L. Weiguo and C. Meng, "Sentiment analysis: A literature review," 2012 International Symposium on Management of Technology (ISMOT), Hangzhou, 2012, pp. 572576, doi: 10.1109/ISMOT.2012.6679538.

Tann H, Tan S, & Cheng X (2009). A survey on sentiment detection of reviews. Expert Syst Appl. 36. 10760-10773