



Sentiment Analysis on Stock Market using Twitter Data

Project proposal

Class: CSCE 5290 Section 002

Group #5

Team Members

Sweety Makwana

Gopi Krishna Sure

Gayathri Katukojwala

Nikhil Rangineni

GitHub Repository: <https://github.com/Smakwana30/CSCE-5290-NLP-Project>

Introduction

➤ Motivation

Twitter is a very well-known place where traders tweet about stocks and financial instruments they care about. Their tweets reveal their sentiment about the stocks they are tweeting. Which can help decide overall trend for the particular stock if analyzed correctly. The stock market which is seen relatively high volatile in recent years could provide chances for making money or losing money. There are various factors that affects the direction of the stock market, or a single stock as follows:

- fundamental of the publicly traded company
- technical indicators
- news and media release for the company
- public sentiment - social media posts such as Twitter, Facebook, etc.

While the field of AI has grew tremendously in recent times, an AI engineer could take advantage of available AI technologies to help predict the trend of stock market by analyzing sentiment of traders from twitter.

➤ Significance

As Abraham Lincoln said, “With public sentiment nothing can fail. Without it, nothing can succeed.” This also applies to the stock market. If public sentiment is positive about the stock market it means the stock market is expected to trend in a bullish manner, else if the public sentiment is negative, once could expect the stock market to trend in a bearish manner. If the sentiment is neutral, then stock market may not see much volatility and may trend neutral. Twitter data is linguistic and unstructured data which can be translated into structured and meaningful data using NLP models. Public sentiment could be classified into three categories as: positive, negative and neutral via text classification and text summarization tools available under the umbrella of Machine Learning field. Without the help of NLP model, the trader needs to read each and every tweet about the particular stock, analyze them manually and then decide the trend of the stock based on the sentiment score he/she comes up with. Not to forget here, the stock market data is constantly changing and so public sentiment about it. It is a very tedious process for anyone to manually keep track of all the tweets and analyze them constantly to help predict the stock prices.

➤ Objectives

To overcome the problems faced by traders and help them decide the trend of a particular stock price using twitter data, our team would like to propose a project to build an NLP based model to derive public sentiment from tweets. Which would help traders to decide the trend of the stock and prevent financial losses. We would acquire pre-labeled Twitter data for a particular financial instrument such as, Apple (AAPL), Tesla (TSLA), Amazon (AMZN), or S&P500 index from Kaggle to train and validate the NLP model. We would obtain historical data for an interested stock from Yahoo Finance to observe possible co-relation between the tweet sentiment analysis and stock prices. The historical data for stock would include daily price for Open, High, Low, Close, Volume and the date. We would yet to incorporate a type of the classifier to classify twitter

data. We may integrate appropriate neural network such as long Short-term Memory (LSTM) or Naïve Bayes algorithm.

➤ Background

One of the most significant areas of study in academic and professional circles is stock price prediction. Various data mining techniques are often used in research. But the machine learning/deep learning technique gives a more accurate, precise and easier way to solve such stock and market price questions.

Now-a-days, information about public opinions is vastly available on social media. This is acting as a platform where public shares their emotional ideas and feelings which can potentially impact overall stock market and/or particular individual stock price. Users of the microblogging service Twitter can follow and comment on the ideas of other users and express their own opinions in real time. More than a million users send more than 140 million tweets every day. Estimation investigation of twitter information and sentiment classification is the assignment of judging conclusion in a chunk of content as positive, negative or unbiased.

In general, the prediction of stock markets is a problem of interest as it changes frequently depending on the demand. The price of shares for the stock markets changes dynamically by the law of supply and demand in the real time situations. As we know if the demand increases the stocks values are increased. Even if we have the historical data of the stocks the price performance is affected by the mood of the people. We observe that the overall social mood on any product is considered as an important variable for the performance. Social mood includes online reviews, Wikipedia, Discussions on the organizations.

In this project a method for predicting stock prices is developed using Twitter tweets about various companies. There are many scholarly research done in this are to predict sentiment of mass people using their public posts from Twitter and other social media. To name some of these research articles:

[1]Go, Alec, Lei Huang, and Richa Bhayani. "Twitter sentiment analysis." *Entropy* 17 (2009): 252. This paper was presented in 2009 when Twitter was a new microblogging platform where users share their sentiments on any related topics including stock prices. They developed a machine learning algorithm to classify tweets for marketers and reviews for the company.

[2]Kouloumpis, Efthymios, Theresa Wilson, and Johanna Moore. "Twitter sentiment analysis: The good the bad and the omg!." *Proceedings of the international AAAI conference on web and social media*. Vol. 5. No. 1. 2011. This paper investigates usefulness of sentiment analysis on Twitter data as being inspired from other domains where strong link has been found between part-of-speech tagging and sentiment lexicons in other domains.

[3]Rakibul Hasan and Maisha Maliha . "Sentiment Analysis with NLP on Twitter Data." This paper was presented in 2019 to perform sentimental analysis on the twitter data using the bag of words and the TF-IDF vectorizer and achieved the 86 % of accuracy and the accuracy can be increased more by using the Tf-IDF vectorizer according to the authors.

[4]Rocha and Macedo.” Stock Price Prediction Using Neural Networks”. According to this paper the authors tested the Neural networks with 10 variations out of which 2 of them gave the best results with different hidden layers. They say that this is very important to predict the stock as that is the responsible for economic growth.

[5]Anupama B S, Rakshith D B, Rahul Kumar M, Navaneeth M.” Real Time Twitter Sentiment Analysis using Natural Language Processing”. This paper has used the naïve bayes algorithm to get the sentiment of people of a particular hashtag from the twitter, they have extracted the sentiment using the data related to the IMDB movie site.

[6]Zhang L .“Sentiment Analysis on Twitter with Stock Price and Significant Keyword Correlation”. In this paper, Zhang L worked on this sentimental analysis using twitter data and observed that working on gathering the keywords can be really helpful for the twitter data prediction. Once we have the keywords for different classification it will be easy for the Stock market predictions. They have used the SVM which gave them a high efficiency in predicting the stock movement.

[7]Johan Bollena, Huina Maoa, Xiaojun Zengb. “Twitter mood predicts the stock market”. According to this paper the public opinion or mood can be tracked from the twitter feed data using high dimensional methods rather than simple nlp text classification. Because of that the organizations are mainly focused on public mood swings as it is affecting the stock markets. Also they have used the Fuzzy neural network model to perform more analysis on the sentiment of stock

[8]Dev Shah, Haruna Isah, Farhana Zulkernine “Predicting the Effects of News Sentiments on the Stock Market”. A dictionary based sentimental analysis model is used in this paper to perform the nature of the stock in the twitter. A 70% accuracy is achieved through this sentimental dictionary model.

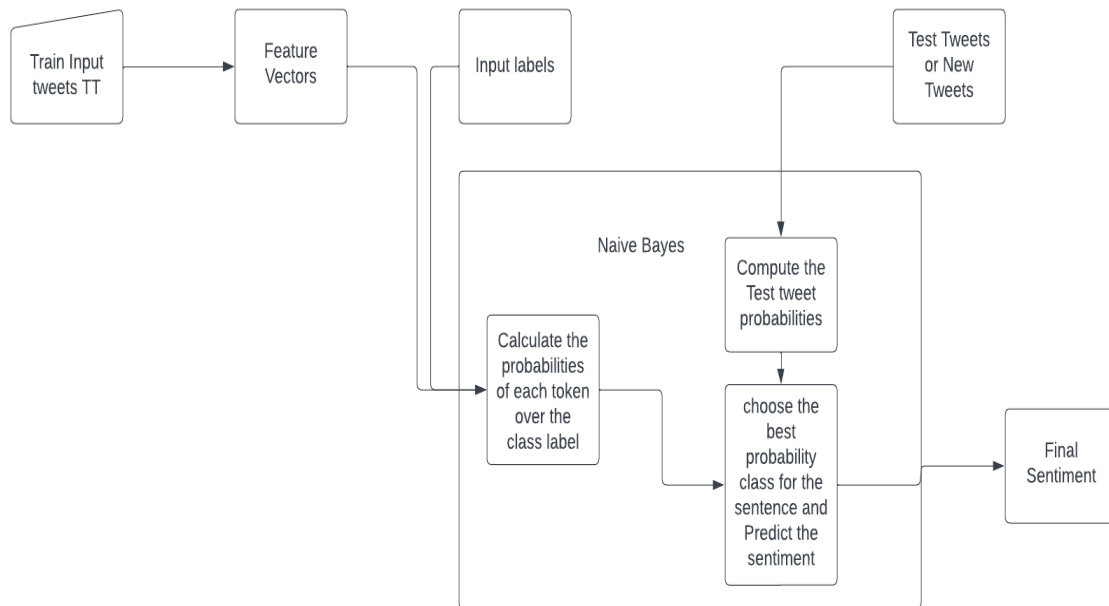
[9]Onam bharti, mrs. Monika malhotra. “sentiment analysis on twitter data”. This paper model has used the knn and naïve bayes combined to extract the sentiment out of the twitter data. They have introduced a four step model that is used for the above process where they have first classified the subjective and objective , later finding the sentiment in the subjective sentences is final goal of this paper.

[10]Cagla Balli, Mehmet Serdar Guzel, Erkan Bostanci, and Alok Mishra “Sentimental Analysis of Twitter Users from Turkish Content with Natural Language Processing”. This paper is related to the sentimental analysis, but it is more of extracting the nature of Turkish tweets , a LSTM model is used in this paper and Tf-IDF is also used for vectorization.

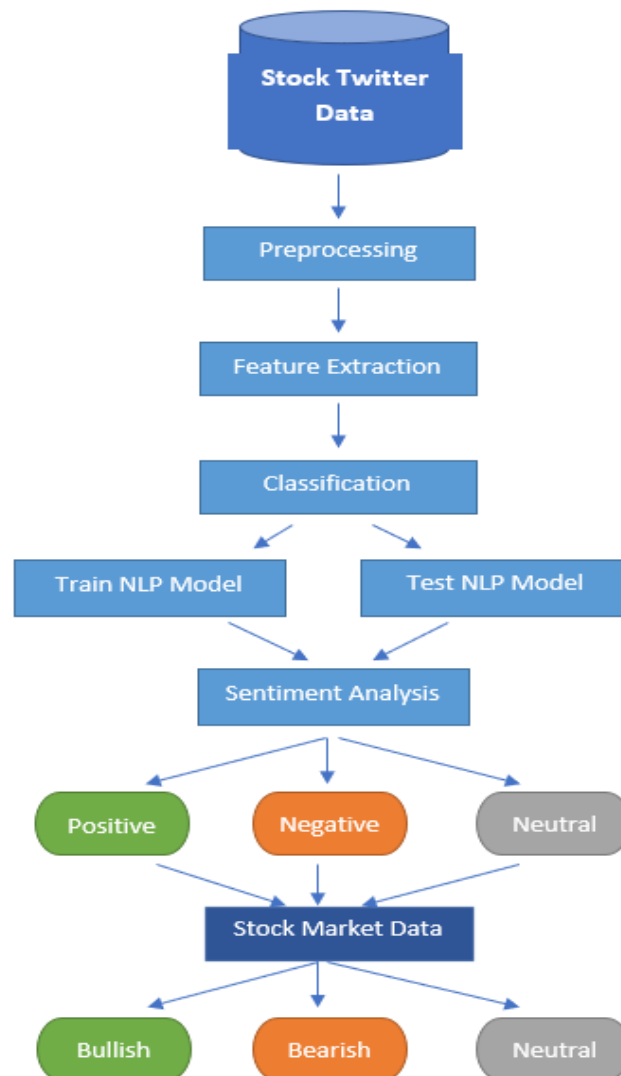
Your Model

➤ Architecture Diagram

The below diagram gives the idea of high level working model of naïve bayes.



➤ Workflow Diagram



The above workflow represents the basic design of our project. Firstly, we take the Twitter data for specific organization and perform some pre-processing techniques to the raw data and then extract the features from the pre-processed data and we build the classification model and classify the data into Positive, Negative and Neutral tweets.

➤ Dataset:

We are using the Twitter API to collect our datasets for the purpose of categorizing the tweets into positive, neutral, and negative tweets later using the NLP models we build along the process. We have taken the keys to access the twitter APIs and then search the tweets based on specific

keywords and then add them to a text file to feed into our model for pre-processing and perform sentimental analysis.

In the dataset, we have searched the keywords 'Tesla', '#TSLA', and combination of the words 'YahooFinance and tesla'. As we have seen that there are different types of tweets related to the Tesla stock, so we filtered the tweets based on the above keywords.

We have stored the filtered tweets in a text file to perform analysis and pre-processing later which is used to feed the model with the processed data.

As seen in Fig. 1 below, 3 code cells are used to fetch the tweets from the twitter, we have set the limit of 1000 and then used the 'search_tweets' method of the twitter to fetch the API and passed the parameter to the items method. Later it converted to the cursor object to iterate over the tweets. Later we have opened a file and written the tweets into the file line by line and after finishing we closed the file object.

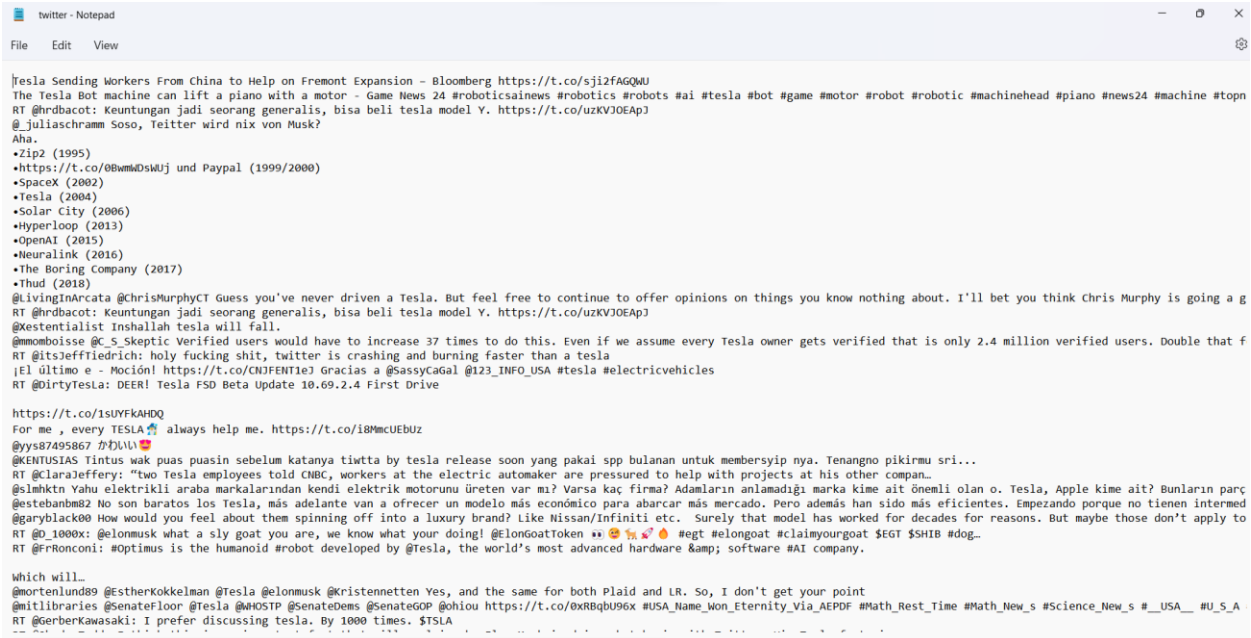
```
[6] #code to search the Tweets with keyword tesla and add them to a text file
keyword = 'Tesla'
limit=1000
tweets = tweep.Cursor(api.search_tweets, q=keyword, tweet_mode='extended').items(limit)
file = open('twitter.txt', 'w', encoding="utf-8")
for tweet in tweets:
    file.write(tweet.full_text+'\n')
file.close()

[7] #code to search the Tweets with keyword #TSLA and add them to a text file
keyword = '#TSLA'
limit=1000
tweets = tweep.Cursor(api.search_tweets, q=keyword, tweet_mode='extended').items(limit)
file = open('twitter.txt', 'a', encoding="utf-8")
for tweet in tweets:
    file.write(tweet.full_text+'\n')
file.close()

[8] #code to search the Tweets with keyword yahoofinance and Tesla and add them to a text file
keyword = 'yahoofinance and Tesla'
limit=1000
tweets = tweep.Cursor(api.search_tweets, q=keyword, tweet_mode='extended').items(limit)
file = open('twitter.txt', 'a', encoding="utf-8")
for tweet in tweets:
    file.write(tweet.full_text+'\n')
file.close()
```

Fig. 1: Codes for Data Collection

Below images Fig. 2. shows the snapshot of our datasets as originally abstracted tweets (twitter.txt) on Tesla company.



```

tesla Sending Workers From China to Help on Fremont Expansion - Bloomberg https://t.co/sjizfAGQMU
The Tesla Bot machine can lift a piano with a motor - Game News 24 #roboticsainews #robotics #robots #ai #tesla #bot #game #motor #robot #robotic #machinehead #piano #news24 #machine #topn
RT @hrdbacot: Keuntungan jadi seorang generalis, bisa beli tesla model Y. https://t.co/uzKVJOEApJ
@_juliaschramm Soso, Teitler wird nix von Musk?
Aha.
•Zip2 (1995)
•https://t.co/0BmmDskUj and Paypal (1999/2000)
•SpaceX (2002)
•Tesla (2004)
•Solar City (2006)
•Hyperloop (2013)
•OpenAI (2015)
•Neuralink (2016)
•The Boring Company (2017)
•Thud (2018)
@LivingInArcata @ChrisMurphyCT Guess you've never driven a Tesla. But feel free to continue to offer opinions on things you know nothing about. I'll bet you think Chris Murphy is going a g
RT @hrdbacot: Keuntungan jadi seorang generalis, bisa beli tesla model Y. https://t.co/uzKVJOEApJ
@Xestentialist Inshallah tesla will fall.
@momboboisie @C_Skeptic Verified users would have to increase 37 times to do this. Even if we assume every Tesla owner gets verified that is only 2.4 million verified users. Double that f
RT @itsjeffriedrich: holy fucking shit, twitter is crashing and burning faster than a tesla
¡El último e - Moción! https://t.co/CN9FENT1e3 Gracias a @SassyCaGal @123_INFO_USA #tesla #electricvehicles
RT @DirtyTesla: DEER! Tesla FSD Beta Update 10.69.2.4 First Drive

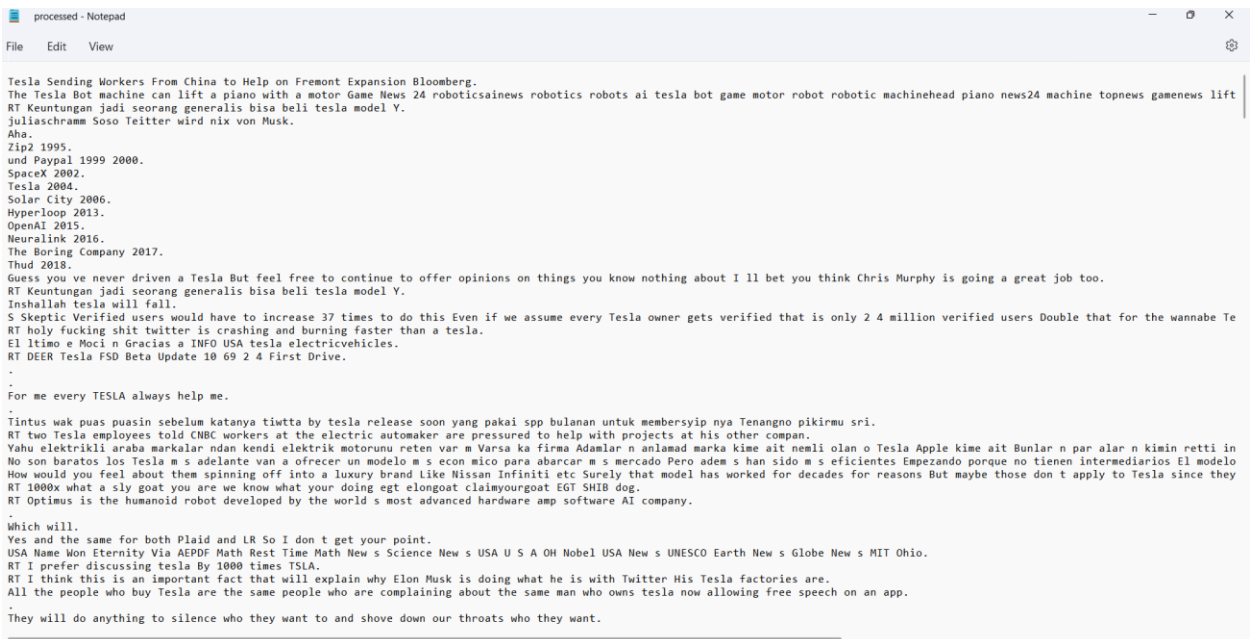
https://t.co/1sUYfKAHDQ
For me , every TESLA always help me. https://t.co/i8MmCUEBUZ
@yys87495867 かわい
@KENTUSIAS Tintus wak puas puasn sebelum katanya tiwta by tesla release soon yang pakai spp bulanan untuk membersyip nya. Tenangno pikirmu sri...
RT @ClaraJeffery: "two Tesla employees told CNBC, workers at the electric automaker are pressured to help with projects at his other compan...
@slmhtn Yahu elektrikli araba markalarından kendi elektrik motorunu üreten var mı? Varsa kaç firma? Adamların anlamadığı marka kime ait önemli olan o. Tesla, Apple kime ait? Bunların parç
@estebanbm82 No son baratos los Tesla, más adelante van a ofrecer un modelo más económico para abarcar más mercado. Pero además han sido más eficientes. Empezando porque no tienen intermed
@garyblack00 How would you feel about them spinning off into a luxury brand? Like Nissan/Infiniti etc. Surely that model has worked for decades for reasons. But maybe those don't apply to
RT @1000x: @elonmusk what a sly goat you are, we know what your doing! @ElonGoatToken 🐐 🐐 🐐 #egt #elongoat #claimyourgoat $EGT $SHIB #dog...
RT @FRonconi: #optimus is the humanoid robot developed by @tesla, the world's most advanced hardware & software #AI company.

Which will...
@mortenlund89 @EstherKokkelaan @Tesla @elonmusk @Kristennetten Yes, and the same for both Plaid and LR. So, I don't get your point
@miltlibraries @senatefloor @tesla @WHOSTP @senatedems @senategop @ohio https://t.co/0xRBqBU96x #USA_Name_Won_Eternity_Via_AEPDF #Math_Rest_Time #Math_New_s #Science_New_s #__USA__ #U_S_A
RT @Gerberkawasaki: I prefer discussing tesla. By 1000 times. $TSLA

```

Fig. 2: Snapshot of raw data collected in 'twitter.txt' file

The image below, Fig. 3. shows the snapshot of partially processed data from 'processed.txt' file, where we have removed special characters, URLs, Links and separated the sentences.



```

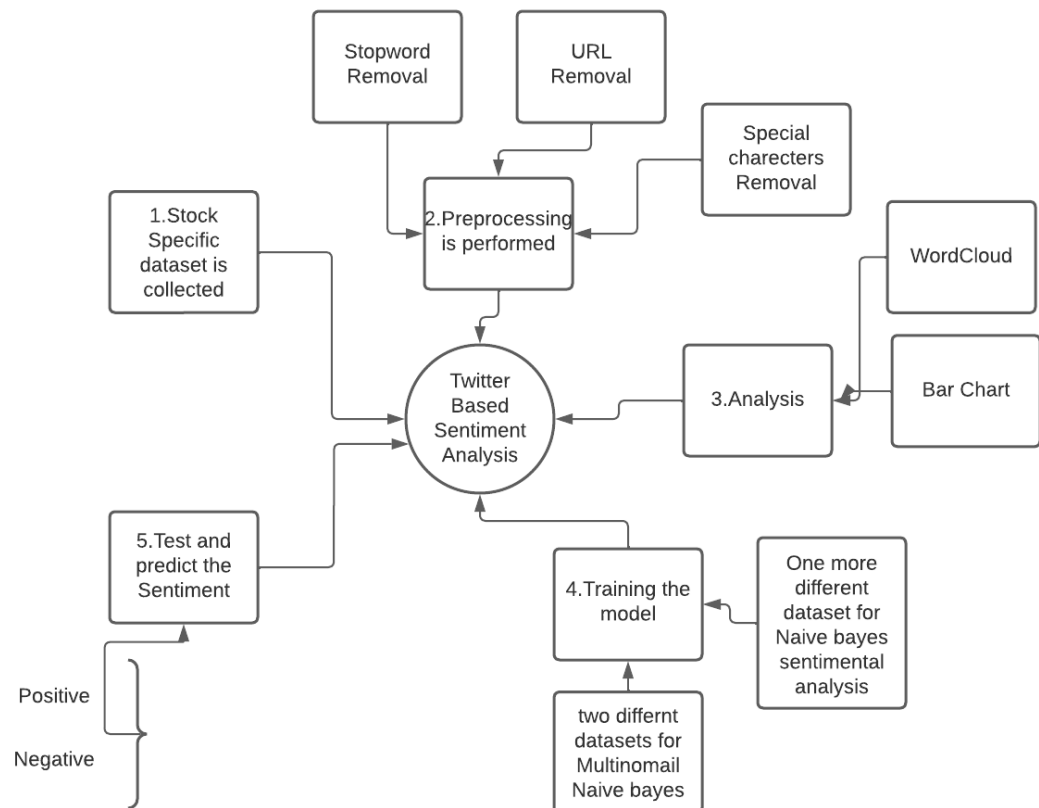
Tesla Sending Workers From China to Help on Fremont Expansion Bloomberg.
The Tesla Bot machine can lift a piano with a motor Game News 24 #roboticsainews robotics robots ai tesla bot game motor robot robotic machinehead piano news24 machine topnews gamenews lift
RT Keuntungan jadi seorang generalis bisa beli tesla model Y.
julaschramm Soso Teitler wird nix von Musk.
Aha.
Zip2 1995.
und Paypal 1999 2000.
SpaceX 2002.
Tesla 2004.
Solar City 2006.
Hyperloop 2013.
OpenAI 2015.
Neuralink 2016.
The Boring Company 2017.
Thud 2018.
Guess you ve never driven a Tesla But feel free to continue to offer opinions on things you know nothing about I ll bet you think Chris Murphy is going a great job too.
Inshallah tesla will fall.
S Skeptic Verified users would have to increase 37 times to do this Even if we assume every Tesla owner gets verified that is only 2 4 million verified users Double that for the wannabe Te
RT holy fucking shit twitter is crashing and burning faster than a tesla.
El ltimo e Moci n Gracias a INFO USA tesla electricvehicles.
RT DEER Tesla FSD Beta Update 10 69 2 4 First Drive.
.
.
For me every TESLA always help me.
.
Tintus wak puas puasn sebelum katanya tiwta by tesla release soon yang pakai spp bulanan untuk membersyip nya Tenangno pikirmu sri.
RT two Tesla employees told CNBC workers at the electric automaker are pressured to help with projects at his other compan.
Yahu elektrikli araba markalar ndan kendi elektrik motorunu reten var n Varsa ka firma Adamlar n anlamad marka kime ait nemli olan o Tesla Apple kime ait Bunlar n par alar n kimin retti in
No son baratos los Tesla m s adelante van a ofrecer un modelo m s econ mico para abarcar m s mercado Pero adem s han sido m s eficientes Empezando porque no tienen intermediarios El modelo
How would you feel about them spinning off into a luxury brand Like Nissan Infiniti etc Surely that model has worked for decades for reasons But maybe those don t apply to Tesla since they
RT 1000x what a sly goat you are we know what your doing egt elongoat claimyourgoat EGT SHIB dog.
RT Optimus is the humanoid robot developed by the world s most advanced hardware amp software AI company.
.
Which will.
Yes and the same for both Plaid and LR So I don t get your point.
USA Name Won Eternity Via AEPDF Math Rest Time Math New s Science New s USA U S A OH Nobel USA New s UNESCO Earth New s Globe New s MIT Ohio.
RT I prefer discussing tesla By 1000 times TSLA.
RT I think this is an important fact that will explain why Elon Musk is doing what he is with Twitter His Tesla factories are.
All the people who buy Tesla are the same people who are complaining about the same man who owns tesla now allowing free speech on an app.
.
They will do anything to silence who they want to and shove down our throats who they want.

```

Fig. 3: Partially Pre-processed data saved in to 'processed.txt' file

➤ Detail Design of Features

- Collect the dataset based on specific keywords related to a stock company from Twitter API
- Preprocess the data to remove special characters, URLs, links, stopwords, lemmatize, etc.
- Visualize and analyze Twitter data using “WordCloud”, Histogram, Unique words search, etc.
- Build NLP model for sentiment analysis
- classify Tweets using NLP model: Positive, Negative, Neutral
- Provide speed and scalability for financial decision to make profit and minimize losses
- Accuracy
- Visualize stock price chart along with Twitter Sentiment Polarity to predict the trend of the stock



In the analysis part we have visualized the text in different formats and did some calculations to get the frequency of most occurred words and the frequency of the length of sentences in the over Train data.

- ```
In [26]: #created the wordcloud method to display the words with the sentiment values
from wordcloud import WordCloud
from matplotlib import pyplot as plt

def DisplayWordCloud(input,bcol):
 plt.figure(figsize=(10,10))
 wocl=WordCloud(background_color=bcol,max_words=50, min_word_length=2, contour_width=1, contour_color='orange')
 wocl.generate(" ".join(input))
 plt.imshow(wocl)
 plt.axis("off")

In [27]: DisplayWordCloud(df[df.Sentiment=="positive"].Text, 'white')
```



2. In the below Fig. 5., we have formed the word cloud by filtering the negative sentiment tweets and displayed the words with more frequency with white background.

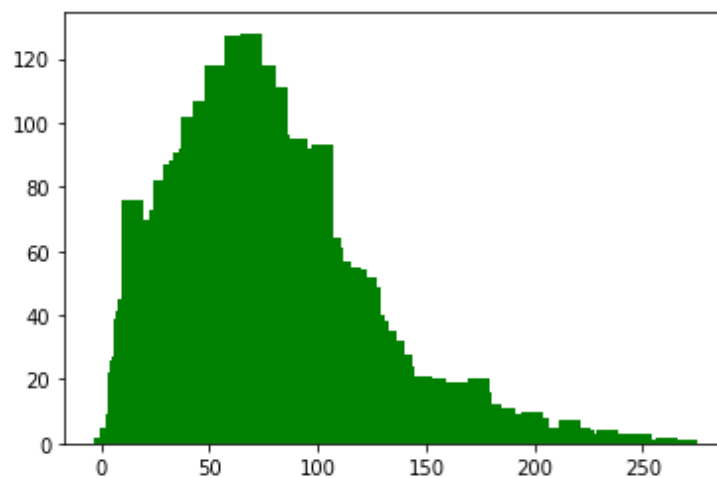
A word cloud visualization of financial news from March 2020. The most prominent words are AAPL, GOOG, EUR, short, user, company, stock, price, today, market, new, one, long, hit, target, support, year, mn, EUR, low, Weekly Triangle, see, lower, still, break, next, NFX, March, back, day, time, said, AMZN, Finnish volume, Operating profit high put share net profit sale, stop, trade, close, n't, and BAC.

3. In the below Fig. 6., we have displayed the word cloud with neutral sentiment text and displayed the words with more frequency and white back ground.

[illegible]

4. In the below Fig. 7., we have plotted a bar chart where it shows the frequency of the length of sentences in the whole training corpus. Using the for loop we have found the length of each sentence and then stored them in a dictionary, later the dictionary is ordered to sort the dictionary based on the items values. Finally using the 'plt.bar' method we have displayed the frequency and length of sentences.

```
[68] #frequency of length of sentences in the Train dataset
from collections import OrderedDict
freq = {}
for line in Train_X:
 l=len(line)
 if (l in freq):
 freq[l] += 1
 else:
 freq[l] = 1
final_dict = OrderedDict(sorted(freq.items()))
plt.bar(final_dict.keys(), final_dict.values(), 10, color='g')
plt.show()
```



*Fig. 7: Analysing length of sentences using Histogram*

5. In the below Fig. 8., we found the most occurred words and displayed the count and word name. We have downloaded the stopwords and the iterated over the corpus to form the corpus into list of words, later used the counter method to get the counts of the words and used the most\_common method to get the frequent occurred words and plotted them into a horizontal bar type graph.

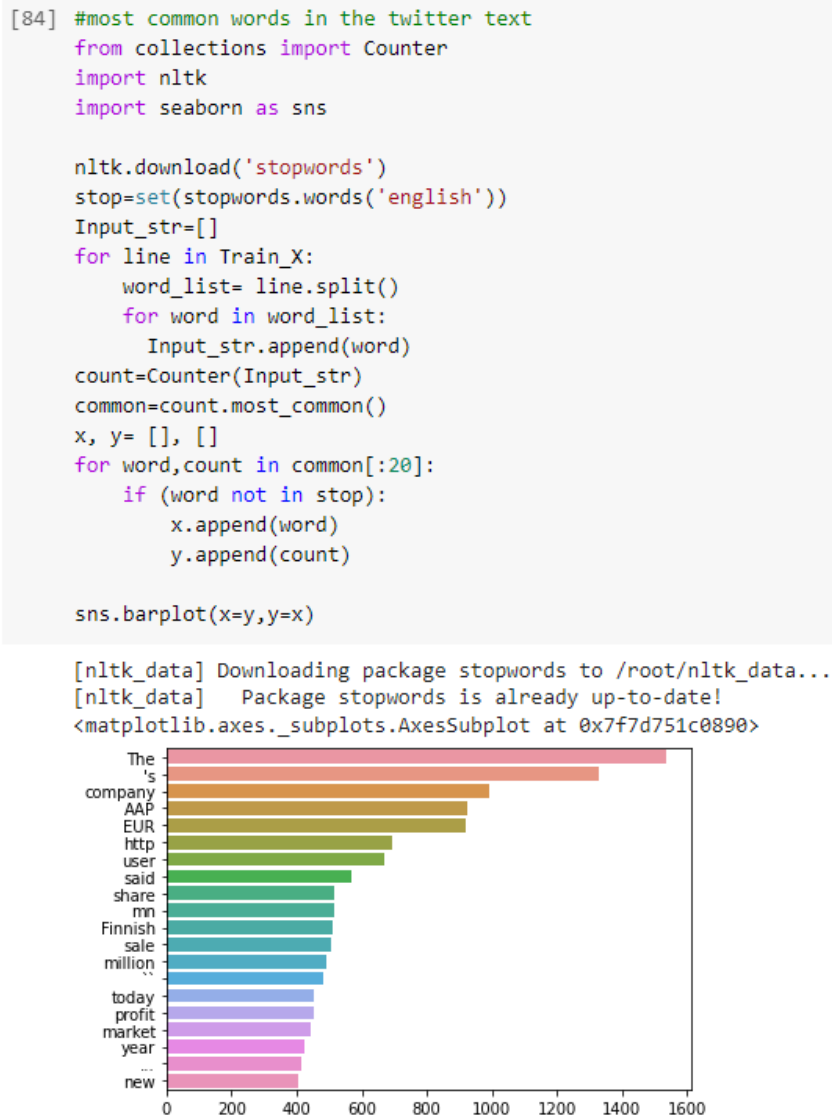


Fig. 8: Analysing Frequency of Unique words

## ➤ ADDITIONAL ANALYSIS

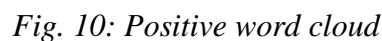
The below analysis is made on the train data that we used to train the model for the sentiment analysis purpose, we have done an additional 6 analysis on positive and negative tweets. We have used the word clouds and bar charts as they are very efficient in analyzing the data.

### Preprocessing

The below screenshot (Fig. 9) will preprocess our data, which means removal of the stopwords and removing the unnecessary links and words that are present in the tweet train data.

*Fig. 9: pre-processing*

```
[33] DisplayWordCloud(df[df.Sentiment=="positive"].Text, 'white')
```



Page 14 of 31

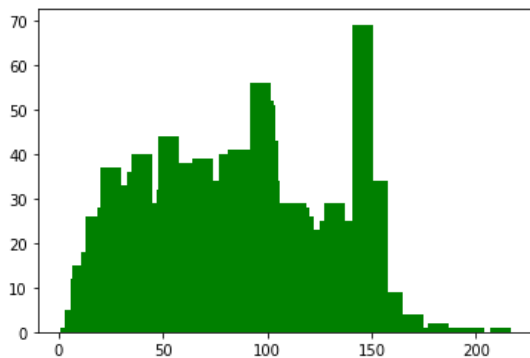
✓  
Dis



*Fig. 11: negative word cloud*

Below image Fig 12 gives us the frequency of length of sentences that are related to the positive sentiment tweets, which means the positive sentiment tweets has 150 length sentences are almost equal to the 70 and most of them are having a frequency of 30-40

```
[35] positive_tweets = df.loc[df['Sentiment']=='positive']
positive_tweets['Text']
#frequency of length of sentences in the positive text of dataset
from collections import OrderedDict
freq = {}
for line in positive_tweets['Text']:
 l=len(line)
 if (l in freq):
 freq[l] += 1
 else:
 freq[l] = 1
final_dict = OrderedDict(sorted(freq.items()))
plt.bar(final_dict.keys(), final_dict.values(), 10, color='g')
plt.show()
```

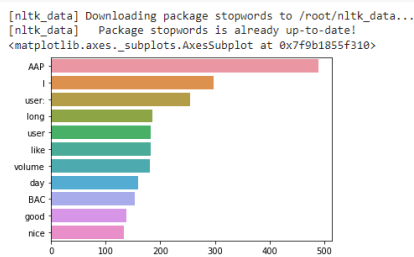


*Fig. 12: Analysing Frequency of Unique words*

```
[36] from scipy.stats.morstats import special
#most common words in the positive dataset text
from collections import Counter
import nltk
import seaborn as sns

nltk.download('stopwords')
stop=set(stopwords.words('english'))
Input_str=[]
for line in positive_tweets['Text']:
 word_list= line.split()
 for word in word_list:
 Input_str.append(word)
count=Counter(Input_str)
common=count.most_common()
x, y = [], []
symbols = {'.', ':', '"', '+', '[', '\\', '@', '^', '{', '%', '(', '-', '"', '*', '|', ',', '&', '<', "'", '}', '!', '>', ':', '?', '#', '$', ')', '/', ' '}
for word,count in common[:40]:
 if (word not in stop):
 if(word not in symbols):
 x.append(word)
 y.append(count)

sns.barplot(x=x,y=y,x)
```

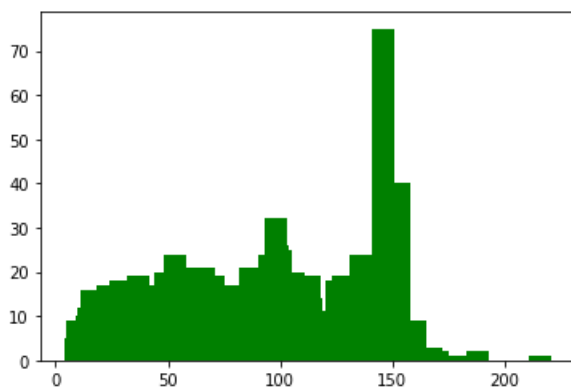


*Fig. 13: Analysing Frequency of Unique words*



Below image Fig. 14 gives us the sentences with length and their frequency in the negative sentiment text, most of them have frequency around 10-20.

```
[37] negative_tweets = df.loc[df['Sentiment']=='negative']
negative_tweets['Text']
#frequency of length of sentences in the negative text of dataset
from collections import OrderedDict
freq = {}
for line in negative_tweets['Text']:
 l=len(line)
 if (l in freq):
 freq[l] += 1
 else:
 freq[l] = 1
final_dict = OrderedDict(sorted(freq.items()))
plt.bar(final_dict.keys(), final_dict.values(), 10, color='g')
plt.show()
```



*Fig. 14: Analysing Frequency of Unique words*

The below image Fig. 15 gives us the words that are repeated a greater number of times in the negative sentiment text. Words such as short which means a negative word are repeated in the negative context.

There are few words that are repeated in both positive and negative, in the below image we are not splitting the words based on their polarity but the number of times they are repeated in the positive and negative context.



*Fig. 15: Analysing Frequency of Unique words*

## ➤ Implementation Algorithms

We have used the naïve bayes algorithm in our project for the classification and the sentimental analysis part. Multinomial naïve bayes along with naïve bayes algorithm are used in the project. For the Classification purpose we have used the multinomial Naïve bayes algorithm as there are multiple classes i.e., more than one class is being used in the classification model. Naïve bayes is one of the algorithms that is based on the bayes theorem.

The diagram shows the formula for Bayes' Theorem: 
$$P(A|B) = \frac{P(B|A) \cdot P(A)}{P(B)}$$
 Annotations with arrows pointing to the formula components: 

- Top-left: "Probability of B occurring given evidence A has already occurred" points to  $P(B|A)$ .
- Top-right: "Probability of A occurring" points to  $P(A)$ .
- Bottom-left: "Probability of A occurring given evidence B has already occurred" points to  $P(A|B)$ .
- Bottom-right: "Probability of B occurring" points to  $P(B)$ .

*Fig.16*

The above formula is the bayes formula from which we can be able to get the probabilities of the words or tokens from the sentences in a text data and then use that probabilities to find the exact label for each sentence. We will calculate the above probability assuming that we already know few probabilities. The main assumptions that naïve bayes makes that every feature is independent, and every feature makes the equal contribution in getting the higher accuracy. So, bayes theorem thinks that the features in our dataset are independent and they don't have any dependency on each other.

In our project we have also used the Tf-IDF vector component to find the most important words in a document. Which means it finds the most important positive and the negative words which helps to classify the tweets easily when we have a vector with the most important words.

## ➤ Implementation code Explanation

Installed the Tweepy package that is required to get the Tweets from twitter. (Fig.. 17)

```
In [1]: pip install tweepy

Requirement already satisfied: tweepy in c:\users\gopi.krishna\anaconda3\lib\site-packages (4.10.1)
Requirement already satisfied: requests<3,>=2.27.0 in c:\users\gopi.krishna\anaconda3\lib\site-packages (from tweepy) (2.28.1)
Requirement already satisfied: oauthlib<4,>=3.2.0 in c:\users\gopi.krishna\anaconda3\lib\site-packages (from tweepy) (3.2.0)
Requirement already satisfied: requests-oauthlib<2,>=1.2.0 in c:\users\gopi.krishna\anaconda3\lib\site-packages (from tweepy) (1.3.1)
Requirement already satisfied: idna<4,>=2.5 in c:\users\gopi.krishna\anaconda3\lib\site-packages (from requests<3,>=2.27.0->tweepy) (3.2)
Requirement already satisfied: urllib3<1.27,>=1.21.1 in c:\users\gopi.krishna\anaconda3\lib\site-packages (from requests<3,>=2.27.0->tweepy) (1.26.7)
Requirement already satisfied: certifi<=2017.4.17 in c:\users\gopi.krishna\anaconda3\lib\site-packages (from requests<3,>=2.27.0->tweepy) (2021.10.8)
Requirement already satisfied: charset-normalizer<3,>=2 in c:\users\gopi.krishna\anaconda3\lib\site-packages (from requests<3,>=2.27.0->tweepy) (2.0.4)
Note: you may need to restart the kernel to use updated packages.

In [2]: pip show tweepy

Name: tweepy
Version: 4.10.1
Summary: Twitter library for Python
Home-page: https://www.tweepy.org/
Author: Joshua Roesslein
Author-email: tweepy@googlegroups.com
License: MIT
Location: c:\users\gopi.krishna\anaconda3\lib\site-packages
Requires: oauthlib, requests, requests-oauthlib
Required-by:
Note: you may need to restart the kernel to use updated packages.
```

Fig. 17

In the below Fig. 18, In the first cell Imported the required packages.

In the second cell, copied the keys that are required to access the Twitter API to get the tweets.

In the third cell, Passed the keys for the authentication and created a API object to access the twitter API.

```
In [3]: import os
import tweepy as tw
import pandas as pd

In [4]: consumer_key= 'Y3RJeJFRHnA9QnpZu8z9S0Skb'
consumer_secret= 'mSHz1KdYkTnmYwuxMwX4fMzzGc9qv10ggyQxMU26pheaGRcLuE'
access_token= '1287803472805367808-Eed6VpYxgeKKILn12gt9HVJagcsmxx'
access_token_secret= 'K9jXNXr6i72odGz9XAiUnN5841eIU1DKcrj52RozR0743'

In [5]: auth = tw.OAuthHandler(consumer_key, consumer_secret)
auth.set_access_token(access_token, access_token_secret)
api = tw.API(auth, wait_on_rate_limit=True)
```

Fig. 18

In the below 3 cells (Fig. 19), Fetched the tweets that are related to the TESLA by passing the 3 types of arguments that are related to the tesla stock. After getting the tweets, the tweets are written into a file called twitter.txt and then later after the tweets are written into the file, the file object is closed. Given 10000 as parameter to fetch the tweets of that count.

```
In [6]: keyword = 'Tesla'
limit=10000
tweets = tw.Cursor(api.search_tweets, q=keyword, tweet_mode='extended').items(limit)
file = open('twitter.txt', 'w', encoding="utf-8")
for tweet in tweets:
 file.write(tweet.full_text+'\n')
file.close()

In [7]: keyword = '#TSLA'
limit=10000
tweets = tw.Cursor(api.search_tweets, q=keyword, tweet_mode='extended').items(limit)
file = open('twitter.txt', 'a', encoding="utf-8")
for tweet in tweets:
 file.write(tweet.full_text+'\n')
file.close()

In [8]: keyword = 'yahoofinance and Tesla'
limit=10000
tweets = tw.Cursor(api.search_tweets, q=keyword, tweet_mode='extended').items(limit)
file = open('twitter.txt', 'a', encoding="utf-8")
for tweet in tweets:
 file.write(tweet.full_text+'\n')
file.close()
```

*Fig.19*

In the below cell (Fig. 20) created a new text file with name processed.txt that contains the tweets from the twitter.txt after processing. All the special characters and the links are removed in this cell and then written into the processed.txt.

```
In [9]: file = open('processed.txt', 'w', encoding="utf-8")

In [10]: #preprocessing
import re
with open('twitter.txt','r', encoding="utf-8") as f:
 lines = f.readlines()
f.close()
for line in lines:
 content=' '.join(re.sub("([A-Za-z0-9+]|([^0-9A-Za-z \t])|(\w+:\w+\/\S+)", " ", line).split())
 file.write(content+'.\n')
file.close()
```

*Fig. 20*

In the below cells (Fig. 21), the processed text is changed into the list format and then added to the test variable which needs to be classified.

```
In [11]: # opening the file in read mode
my_file = open("processed.txt", "r")

reading the file
data = my_file.read()

replacing end splitting the text
when newline ('\n') is seen.
data_into_list = data.split("\n")
my_file.close()

In [12]: Test_X=[]
for x in data_into_list:
 Test_X.append(x)
```

*Fig. 21*

## Group #5

In the first cell (Fig. 22), the text is tokenized using the word tokenize method and then removed the stop words and then finally appened the remaining words into a sentence and then added back to the test\_data.

In the second cell removed all the duplicate tweets if there are any to avoid the wrong results.

```
[13]: import nltk
from nltk.stem import WordNetLemmatizer
from nltk.corpus import stopwords
from nltk.tokenize import word_tokenize
nltk.download('stopwords')
stopword = set(stopwords.words('english'))
Test_data=[]
for x in Test_X:
 tokens = word_tokenize(str(x))
 final_tokens = [w for w in tokens if w not in stopword]
 wordLemm = WordNetLemmatizer()
 finalwords=[]
 for w in final_tokens:
 if len(w)>1:
 word = wordLemm.lemmatize(w)
 finalwords.append(word)
 Test_data.append(' '.join(finalwords))

[nltk_data] Downloading package stopwords to C:\Users\Gopi
[nltk_data] krishna/nltk_data...
[nltk_data] Package stopwords is already up-to-date!
```

```
[14]: #Removing single words
Test_X=[]
for x in Test_data:
 if len(x)>10:
 Test_X.append(x)
Test_X = [*set(Test_X)]
```

*Fig. 22*

As seen in Fig. 23, to train the model we have collected 2 different sets from the Kaggle and then merged the 2 data sets. One is All-data.csv and other is stock\_data.csv. Merged the 2 datasets to form a dataset of size 10K. Collected the text into the Train\_X and then copied the sentiment to the Train\_Y variables to train the model. For one dataset changed the sentiment values from 1 to positive and -1 to negative.

```

In [15]: M twit = pd.read_csv("all-data.csv", encoding = "latin-1")

In [16]: M Train_Y=twit["Sentiment"]

In [17]: M Train_X=twit["Text"]

In [18]: M twit = pd.read_csv("stock_data.csv", encoding = "latin-1")

In [19]: M for ind in twit.index:
 if(twit['Sentiment'][ind]==-1):
 twit['Sentiment'][ind]="negative"
 else:
 twit['Sentiment'][ind]="positive"

C:\Users\GOPIKR~1\AppData\Local\Temp\ipykernel_44488\3332717168.py:5: 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
 twit['Sentiment'][ind]="positive"
C:\Users\Gopi krishna\anaconda3\lib\site-packages\pandas\core\indexing.py:1732: 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
 self._setitem_single_block(indexer, value, name)

In [20]: M Train_X=Train_X.append(twit["Text"])
 M Train_Y=Train_Y.append(twit["Sentiment"])

In [21]: M Train_X.shape

Out[21]: (10637,)

```

Fig. 23

In the below cell (Fig. 24) cleaned the Train\_X tweets by removing the stopwords, first downloaded the stopwords and then compared with the tokens in the text, if they are matched the tokens are removed and then appended to form a sentence and then added to the Train\_X variable.

```

In [22]: M import nltk
 M from nltk.stem import WordNetLemmatizer
 M from nltk.corpus import stopwords
 M from nltk.tokenize import word_tokenize
 M nltk.download('stopwords')
 M stopword = set(stopwords.words('english'))
 M Train_data=[]
 M for x in Train_X:
 M tokens = word_tokenize(str(x))
 M final_tokens = [w for w in tokens if w not in stopword]
 M wordLemm = WordNetLemmatizer()
 M finalwords=[]
 M for w in final_tokens:
 M if len(w)>1:
 M word = wordLemm.lemmatize(w)
 M finalwords.append(word)
 M Train_data.append(' '.join(finalwords))
 M Train_X= Train_data

[nltk_data] Downloading package stopwords to C:\Users\Gopi
[nltk_data] krishna\nltk_data...
[nltk_data] Package stopwords is already up-to-date!

```

Fig. 24

As seen in Fig. 25 below, the model is built with the pipeline having vector inputs as parameters and multinomial classification. And then the model is trained by using the Train\_X and Train\_Y variables. After training the model the Test data is passed which we collected from the twitter API and then the labels are predicted which can be classified as Positive, Negative and Neutral.

```
In [23]: from sklearn.feature_extraction.text import TfidfVectorizer
from sklearn.naive_bayes import MultinomialNB
from sklearn.pipeline import make_pipeline

model = make_pipeline(TfidfVectorizer(), MultinomialNB())

In [24]: model.fit(Train_X, Train_Y)
labels = model.predict(Test_X)
```

Fig. 25

## ➤ Continued Implementation

For the sentiment analysis we have imported the required packages from python library. We also downloaded the stop words and then read the input file for training the model from a csv files, and then displayed the top 2 records. (Fig. 26)

```
[29] import nltk
import numpy as np
import nltk.classify.util
from nltk.classify import NaiveBayesClassifier
from nltk.corpus import twitter_samples
from nltk.corpus import stopwords
from nltk.corpus import wordnet
import pandas as pd

[30] #nltk.download('twitter_samples')
nltk.download('stopwords')

[nltk_data] Downloading package stopwords to /root/nltk_data...
[nltk_data] Package stopwords is already up-to-date!
True

[31] df = pd.read_csv("stock_twitter.csv",encoding='latin-1',sep=',')

[32] df.head(2)
```

|   | Text                                              | Sentiment |
|---|---------------------------------------------------|-----------|
| 0 | Kickers on my watchlist XIDE TIT SOQ PNK CPW B... | positive  |
| 1 | user: AAP MOVIE. 55% return for the FEA/GEED i... | positive  |

Fig. 26



## Group #5

The below 2 methods are used for cleaning the tweet and the other to count the tweets and store in a frequency dictionary along with the class label. In the process\_tweet method we have removed the links and unnecessary characters from the text data. And the count\_tweets method we stored the word with class as key and count as value in the frequency dictionary. (Fig. 27)

```

✓ [39] #cleaning the dataset
0s
import re
import string
import numpy as np

from nltk.tokenize import TweetTokenizer
from nltk.corpus import stopwords

def process_tweet(tweet):

 # remove the hyperlinks
 tweet = re.sub(r'https?:\/\/\.[^\s]*', '', tweet)

 # remove the # symbol
 tweet = re.sub(r'#', '', tweet)

 tokenizer = TweetTokenizer(preserve_case=False, reduce_len=True, strip_handles=True)
 tweet_tokens = tokenizer.tokenize(tweet)
 tweet_clean = []

 return tweet_tokens

✓ [40] #counting the frequency
0s
def count_tweets(tweets, s):
 tweet_list = np.squeeze(s).tolist()
 frequency = {}

 for y, tweet in zip(tweet_list, tweets):
 for word in process_tweet(tweet):
 pair = (word, y)
 if pair in frequency:
 frequency[pair] += 1
 else:
 frequency[pair] = 1

 return frequency

```

Fig. 27

In the below screenshot (Fig. 28) we have divided the test and training dataset and assigned them to the appropriate variables.

```

[42] # splitting the data for training and testing
training_pos = positive_tweets.sample(frac=0.8, random_state=25)
testing_pos = positive_tweets.drop(training_pos.index)

training_neg = negative_tweets.sample(frac=0.8, random_state=25)
testing_neg = negative_tweets.drop(training_neg.index)

train_x = training_pos.append(training_neg)
test_x = testing_pos.append(testing_neg)

numpy array for the labels in the training set
train_y = np.append(np.ones((len(training_pos))), np.zeros((len(training_neg))))
test_y = np.append(np.ones((len(testing_neg))), np.zeros((len(testing_neg))))

```

Fig. 28

Finally, we have trained the model with the frequency that we calculated before and the train variables are given as parameters which would give is the prior probability and length of likelihood.

The next method which we used to predict the sentiment for the tweet data that we prepared and classified earlier. We calculated in way that if  $p$  is greater than 1 then it is classified a positive sentiment and less than 1 it is a negative sentiment. (Fig. 29)

```
[44] logprior, loglikelihood = train_naive_bayes(frequency, train_x, train_y)
 print(logprior)
 print(len(loglikelihood))

0.5593614105227651
9256
```

```
[45] def naive_bayes_predict(tweet, logprior, loglikelihood):
 word_l = process_tweet(tweet)
 p = 0
 p+=logprior

 for word in word_l:
 if word in loglikelihood:
 p+=loglikelihood[word]

 return p
```

*Fig. 29*

## ➤ Results:

The below screenshot (Fig. 30) gives us the final result of our project, the twitter data that was classified earlier using the classification model is now fed into the sentimental model to find the final sentiment of the tweets and also to find the accuracy of the model that was built.

So below graph shows there are more number of positive sentiment in the tweets which may indicate that the stock price may rise in the coming days or the next day. (Fig. 30)

```
[46] po=0
 n=0
 Final_labels=labels.tolist()
 pcount=Final_labels.count("positive")
 ncount=Final_labels.count("negative")
 for tweet,label in zip(tweets,labels):
 p=naive_bayes_predict(tweet,logprior,loglikelihood)
 if(p>1):
 if(label=="positive"):
 po=po+1
 else:
 if(label=="negative"):
 n=n+1
 Accuracy= ((n+po)/(pcount+ncount)) * 100
 print("Accuracy of model:"+str(Accuracy)+" %")
 #plotted a bar chart for the lables that are predicted for the input.
 import matplotlib.pyplot as plt
 fig = plt.figure()
 ax = fig.add_axes([0,0,1,1])
 Sentiment = ['Positive', 'Negative']
 Count = [po,n]
 ax.bar(Sentiment,Count)
 plt.show()
```

Accuracy of model:50.397350993377486 %

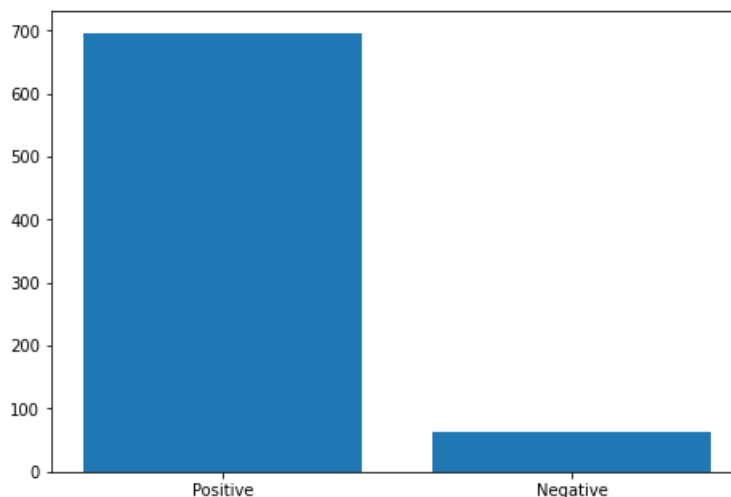


Fig. 30

The below two screenshots (Fig. 31 and Fig. 32) are just for the reference that the model is predicting the correct direction or not. The above results may change when we use the whole twitter data for Tesla in the project and we may get more accurate results using the whole twitter stock data. As there is a restriction on the Advanced twitter API that we used to get only limited number of tweets. Hence, we have only used limited number of tweets from the entire tweet data for Tesla. Our model result may change if we feed the whole twitter stock data. For now, with limited number of tweets, our NLP model predicts positive sentiment for Tesla stock which may indicate Bullish trend is coming. As we see in Fig. 31, which is 1-day line chart of Tesla stock price, the stock price has already increased. Further when we look at the 5 day price line chart as seen in Fig. 32, we can see that for past few days from Nov 23 the stock price is increasing which may be associated with positive sentiment of people about the stock.

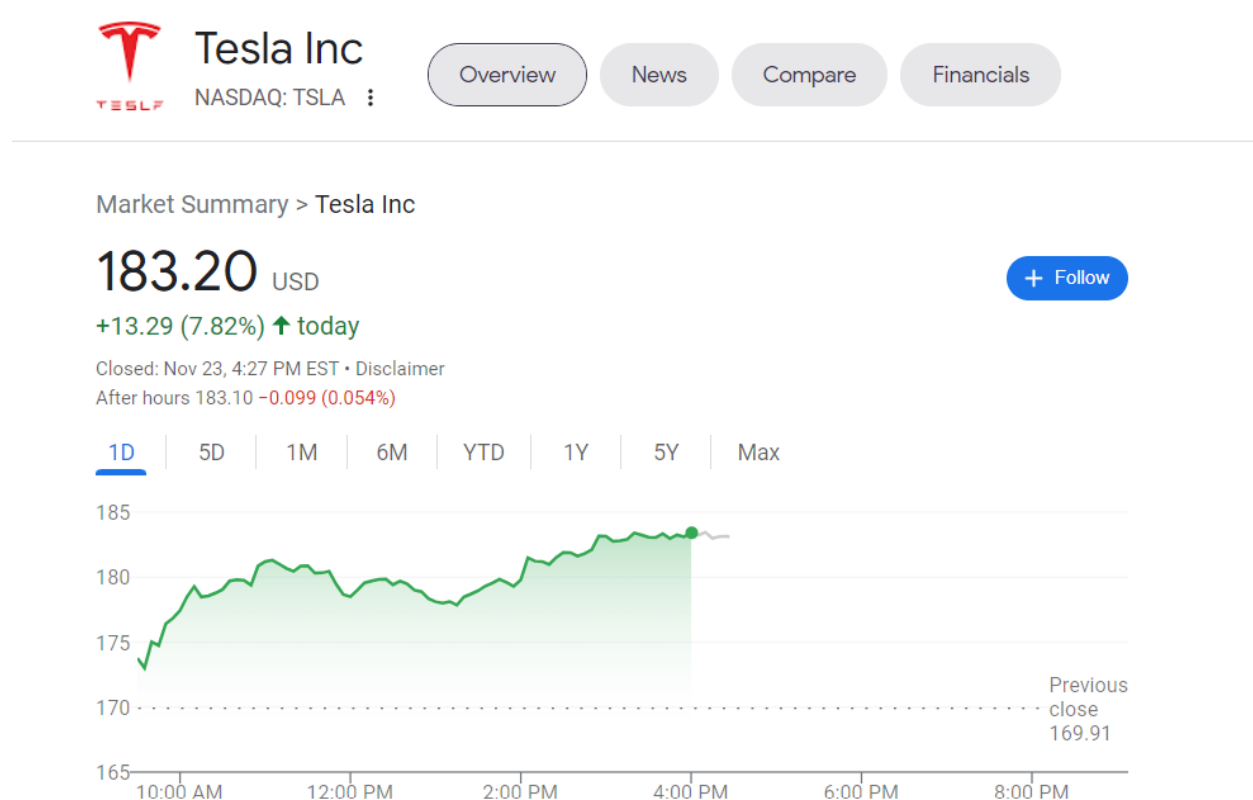


Fig. 31: 1-day chart of Tesla Stock



**Tesla Inc**  
NASDAQ: TSLA

Overview

Compare

Financials

Market Summary > Tesla Inc

**182.86** USD

+ Follow

-2.26 (-1.22%) ↓ past 5 days

Closed: Nov 25, 4:59 PM EST • Disclaimer

After hours 182.89 +0.030 (0.016%)

1D | 5D | 1M | 6M | YTD | 1Y | 5Y | Max



|      |        |           |         |            |        |
|------|--------|-----------|---------|------------|--------|
| Open | 185.06 | Mkt cap   | 572.99B | 52-wk high | 402.67 |
| High | 185.20 | P/E ratio | 56.33   | 52-wk low  | 166.18 |
| Low  | 180.63 | Div yield | -       |            |        |

*Fig. 32: 5-day chart of Tesla Stock*

## Project Management and Responsibility

To manage this project, we have divided our responsibilities amongst four team members as mentioned earlier in this project document. We communicated in person as well as via online medium such as WhatsApp and Zoom to cooperate with one another. We broke down the project into smaller tasks and worked on each task as assigned primarily. Each task was peer reviewed by all four of us as mentioned below in Fig. 33

| Task Description                | Primary Responsible (First Name) | % Of work done as Primary Responsible member | Peer Reviewed by (First Name)  |
|---------------------------------|----------------------------------|----------------------------------------------|--------------------------------|
| Brain Storming Ideas            | Gopi, Sweety, Nikhil, Gayathri   | 25% each member                              | Gopi, Sweety, Nikhil, Gayathri |
| Data Collection                 | Sweety, Gopi, Nikhil, Gayathri   | 25% each member                              | Nikhil, Gayatri                |
| Topic Research and Related work | Gopi, Sweety, Nikhil,            | 34% each member                              | Gopi, Sweety, Nikhil           |
| Workflow and Project Outline    | Sweety, Gopi, Nikhil             | 34% each member                              | Gopi, Nikhil, Sweety           |
| Feature Selection               | Gopi, Nikhil, Sweety             | 34% each member                              | Gayathri                       |
| Data Analysis                   | Gopi, Sweety, Nikhil, Gayathri   | 25% each member                              | Gopi, Sweety, Nikhil           |
| Implementation                  | Gopi, Sweety, Nikhil, Gayathri   | 25% each member                              | Gopi, Sweety, Nikhil           |
| Results                         | Gopi, Sweety, Nikhil             | 34% each member                              | Gopi, Sweety, Nikhil,          |
| Presentation                    | Gopi, Sweety, Nikhil             | 34% each member                              | Gopi, Sweety, Nikhil           |

*Fig. 33*

## Demos:

Video 1: Gopi Krishna Sure

[https://drive.google.com/file/d/1f17hOECQSC5CHC3JbZSrBuVOh\\_x-4xpd/view?usp=sharing](https://drive.google.com/file/d/1f17hOECQSC5CHC3JbZSrBuVOh_x-4xpd/view?usp=sharing)

Video 2: Sweety Makwana

[https://drive.google.com/file/d/12Ghms8lqK-OPScz8RgpyELZqvSEi7i\\_p/view?usp=sharing](https://drive.google.com/file/d/12Ghms8lqK-OPScz8RgpyELZqvSEi7i_p/view?usp=sharing)

Video 3: Nikhil Rangineni

[https://drive.google.com/file/d/1ZAD0JRu\\_ntfDOjS10KuvwOEWBfp\\_qZDK/view?usp=share\\_link](https://drive.google.com/file/d/1ZAD0JRu_ntfDOjS10KuvwOEWBfp_qZDK/view?usp=share_link)

Video 4: Gayathri Katukojwala

[https://drive.google.com/file/d/19ZN5XliwGPC0lf\\_dirnpu0ZiQX6bYHkz/view?usp=sharing](https://drive.google.com/file/d/19ZN5XliwGPC0lf_dirnpu0ZiQX6bYHkz/view?usp=sharing)

## References:

- <https://www.kaggle.com/>
- <https://finance.yahoo.com/>
- <https://data.world/crowdfunder/apple-twitter-sentiment>
- <https://www.ncbi.nlm.nih.gov/pmc/articles/PMC7959635/>
- <https://www.kaggle.com/datasets/ankurzing/sentiment-analysis-for-financial-news>
- <https://monkeylearn.com/blog/sentiment-analysis-of-twitter/>
- Go, Alec, Lei Huang, and Richa Bhayani. "Twitter sentiment analysis." *Entropy* 17 (2009): 252.
- Kouloumpis, Efthymios, Theresa Wilson, and Johanna Moore. "Twitter sentiment analysis: The good the bad and the omg!." *Proceedings of the international AAAI conference on web and social media*. Vol. 5. No. 1. 2011.
- <https://www.nltk.org/>
- <https://developer.twitter.com/en/docs/twitter-api>