Project Report

Stock Recommender System

CS 410: Text Information System

University of Illinois, Urbana-Champaign

Team Members

Name	NetID	Email ID
Ezra Schroeder	ezras2	ezras2@illinois.edu
Rasbihari Pal	pal9	pal9@illinois.edu
Sandeep Kumar	kumar64	kumar64@illinois.edu

Team captain marked in **BOLD.**

Project Overview

Abstract

Our project is a novel text-mining application which combines 3 different functionalities around rating stocks into one application. Namely, it consists of a base analysis which is a two-tiered summary of sentiment of stock analyst ratings who are well known in the industry and rate stock ticker symbols, a twitter sentiment analysis aspect which scrapes tweets off of Twitter and analyzes them for sentiment about a particular stock symbol, and a recommendation engine which recommends stock symbols similar to a user-provided query stock symbol. There are many conceivable use-cases for an app such as ours upon further embellishment and improvements, such as individual investors in the stock market who want automated and instantaneous advice and suggestions that incorporates both analyst ratings from well-known analysts across the internet, public sentiment as embodied by tweets, and that produces not only concise summaries of these analyst ratings and Twitter sentiment but recommends similar stocks to their stock symbol (e.g. AAPL) of interest. Imaginably there may well also be corporate interest in incorporating an application such as ours into a larger pipeline which could be serving a huge myriad of purposes but that leverages actionable knowledge about the stock market into a larger purpose. Although our application is alpha version, it is not inconceivable that it could springboard academic research into these areas.

Motivation

In the year 2020, there has been huge surge in securities trading, driven by retail investors. The increase in the trading activity can be attributed primarily to the easy-to-use mobile based trading apps, offered by several FinTech companies such as eToro, Robinhood and InteractiveBrokers etc. The retail investors of today might lack time for in-depth research and/or even lack the necessary knowledge to analyze the financial standing of a company. In such a situation, many of the retail investors either rely on word of mouth or blindly following other investors on such trading platforms. This leads to investment decisions beyond the risk profile and/or risk appetite of the investors.

Introduction

We propose "Stock recommender system" as a solution to enable retail investors easy access to information, relevant for informed investment decisions. Based on user's preference of a stock, the stock recommender proposes a cumulated rating for the stock and also proposes other stocks with similar ratings. The recommender system combines

- stock rating data from various market analyst,
- market sentiments and
- company profile

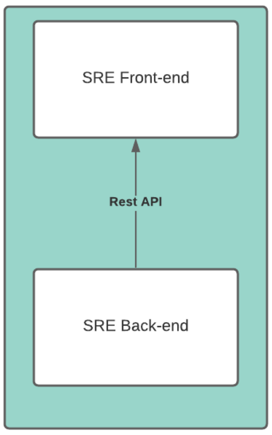
for determination of the cumulated rating and curation of the recommendation list.

High Level Design

Component View (Deployment Perspective)

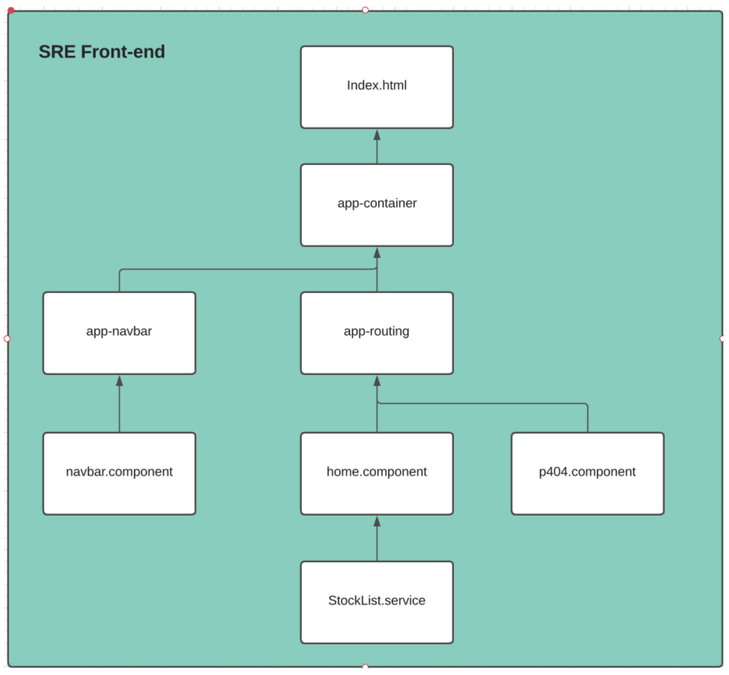
Stock Recommendation System consists of two components - SRE Front-end and SRE Back-end.

- The SRE Front-end is the UI component for user interaction, developed using Angular. The Front-end relies on SRE Back-end for all its data needs for providing various user centric functions.
- The SRE Back-end is the main component, which implements all the necessary algorithm and business rules and finally exposes the data related to ratings, recommendations and tweets via Rest APIs.



SRE Front-end (UI)

The SRE Front-end is developed in Angular framework and is a single page application. It consists of the following components:



A. app-container

The main component to render the Stock Recommendation System web page

B. app-navbar

The component responsible for top navigation bar

C. app-routing

The component dealing with url based routing and takes care of routing to the home page or canonical “page-not-found“ page.

D. home.component

This is the main component which takes user input, calls SRE backend services to fetch relevant data.

E. p404.component

This is the component for handling “page not found” scenario in case user enters incorrect URL not supported by SRE Front-end.

SRE Back-end (Rest API)

The SRE Back-end is a Flask based app, developed using Python and TinyDB which exposes various Rest APIs for the SRE Front-end.

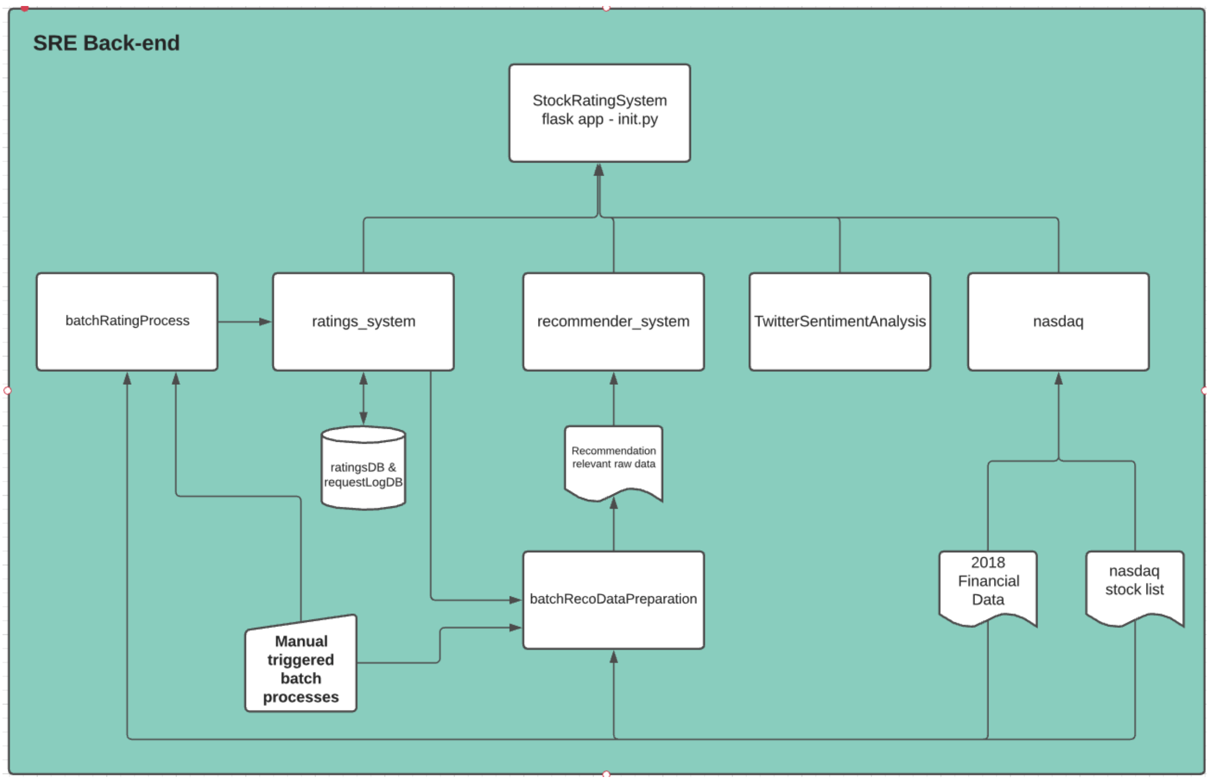
List of Rest APIs:

Rest API	Sample Response
Get list of all stocks in the corpus (output abridged for succinctness) `GET /stock/all`	`[ { "Symbol":"AAPL", "Security Name": "Apple Inc", "Market": "NASDAQ", "Sector": "Technology" } ]`

Get ratings for a given stock from a given market `GET /stock/ratings/<market>/<stock_symbol>`	`[{ "stockSymbol": "AAPL", "marketPlace": "NASDAQ", "refreshData": "2020-11-25", "overallRating": "HOLD", "analystsRatings": [{ "level_0": 0, "index": 1, "ratingDate": "2020-11-19", "ratingAgency": "The Goldman Sachs Group", "ratingAssigned": "Sell", "newRatings": -1, "scaledRatings": "SELL" }] }]`
Get a list of recommended stocks, similar to given stock `GET /stock/recommendation/<stock_symbol>`	`[ { "seq": 1, "stockSymbol": "HAFC", "stockName": "Hanmi Financial Corporation", "sector": "Finance", "rating": "SELL" }, ]`
Get Twitter sentiment for given stock 1: Positive, 0: Neutral, -1: Negative `GET /stock/sentiments/<market>/<stock_symbol>`	`{ "stockSymbol": "AAPL", "refreshDate": "2020-11-25", "sentiment": 1 }`
Get list of user request log `GET /requests/all`	`[ { "datetime": "2020-11-25T21:55:52.924706", "symbol": "TSLA", "market": "NASDAQ" } ]`

List of the Rest APIs exposed out of the backend can be found on GitHub project repository: https://github.com/MLwithSandy/CourseProject/tree/main/ProjectCode/StockRatingsSystem

The SRE Back-end is comprised of the following components and/or modules:



A. Flask app

This is the main component which exposes various Rest APIs for external consumers - in our case SRE Front-end. It integrates all other components such as ratings_system, recommender_system etc. to provide the necessary services. In addition, it also logs all requests to requestLogDB.

B. ratings_system

The ratings_system is the component which calculates the overall rating for the user selected (searched) stock, by scraping the required ratings data from a variety of market analyst websites and aggregating them. It also provides the list of our selected latest ten ratings from an assortment of various market analysts as a reference.

C. recommender_system

The recommender_system takes a stock symbol as input and recommends 5 stocks matching the profile of the user selected (searched) stock and a pre-defined similarity function.

D. TwitterSentimentAnalysis

TwitterSentimentAnalysis component fetches latest tweets for the user selected (searched) stock from Twitter which by making use of the Twitter API, performs sentiment analysis and returns overall sentiment for the given stock. It also provides the latest five tweets as a reference.

E. nasdaq

The nasdaq component is primarily used for preparation of data. It combines the nasdaq listed stocks with the dataset we used for identifying various feature parameters of a stock. By cleaning and combining our data sources, we were able to create a final dataset of 1462 stocks for use across our systems in our app (analyst ratings, twitter sentiment, recommender system).

F. batchRatingProcess

The batchRatingProcess is used to update ratings of all 1462 stocks in background. This is to ensure that the user does *not* face any significant delay in fetching the data from third party websites while interacting with our SRE app's UI. In addition, the ratings of a stock is not updated frequently and batch processing fits well from a requirements and solution design perspective.

G. batchRecoDataPreparation

The batchRecoDataPreparation is another batch process to prepare the data for the recommendation system as the recommendation engine relies on an assortment of analyst ratings of stocks as a feature in the similarity function.

Implementation details

Rating System

A. Approach

To calculate the ratings, the following algorithm is used:

- First of all, rating for the selected(searched) stock is searched in the ratings database.
    - Case 1: ratings data is already available in the ratings database. In such a case, ratings data is fetched from the database.
    - Case 2: ratings data is not available in ratings database. In this case, following step is executed to get the ratings.
        - Read the website marketbeat.com which lists the ratings from various market analysts and scrape the page section containing ratings relevant date
        - Data is cleansed, validated and structured as per requirement
        - In case of more than 10 ratings, only latest 10 ratings are selected.
        - Initially data from only current and previous months were considered but to increase the data quantity, this restriction is switched off in final product
        - Final list of ratings is stored in the database
- Once the rating data is available, ratings from various market analysts are scaled to following ratings scale: {-1: SELL, 0: HOLD, 1: BUY}
- Finally, overall rating is calculated based on the selected scaled ratings with possible outcome as SELL, HOLD or BUY.

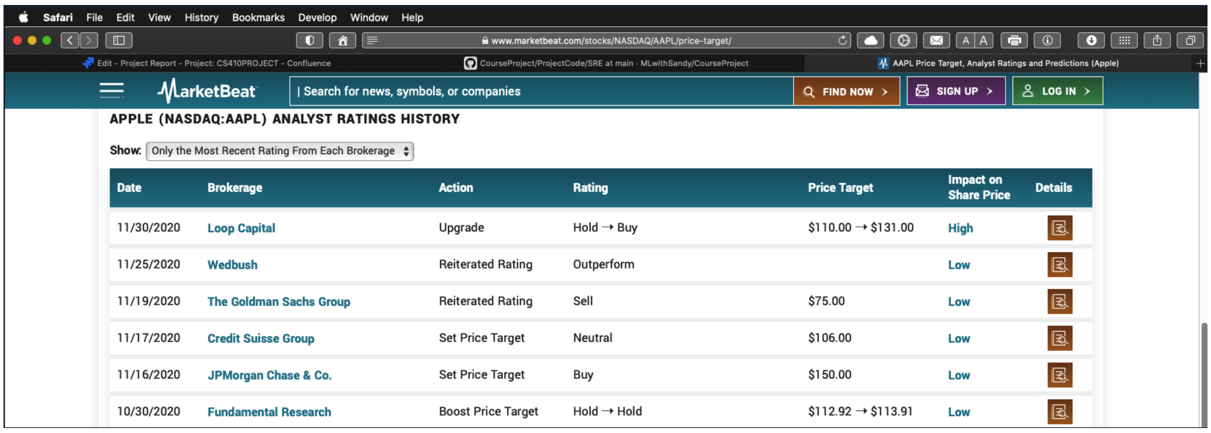
B. Webpage Scraper

Webpage scraper was developed using the learnings from MP 2.1 of the course CS 410: Text Information System. Python library BeautifulSoup and Chrome driver were primarily used to fetch the web document.

Following URL builder code were used to get URL for various stocks:

```
# MarketBeat URL builder

def get_mb_url(market, stock_symbol):
    base_url =
    'https://www.marketbeat.com/stocks/{market}/{stock_symbol}/price-target/'
    final_url = base_url.format(market=market, stock_symbol=stock_symbol)
    return final_url
```

e.g. https://www.marketbeat.com/stocks/NASDAQ/AAPL/price-target/



Safari File Edit View History Bookmarks Develop Window Help
www.marketbeat.com/stocks/NASDAQ/AAPL/price-target/
Edit - Project Report - Project: CS410PROJECT - Confluence
CourseProject/ProjectCode/SRE at main · MLwithSandy/CourseProject
AAPL Price Target, Analyst Ratings and Predictions (Apple)
MarketBeat
| Search for news, symbols, or companies
FIND NOW
SIGN UP
LOG IN

APPLE (NASDAQ:AAPL) ANALYST RATINGS HISTORY

Show: Only the Most Recent Rating From Each Brokerage

Date	Brokerage	Action	Rating	Price Target	Impact on Share Price	Details
11/30/2020	Loop Capital	Upgrade	Hold → Buy	\$110.00 → \$131.00	High	
11/25/2020	Wedbush	Reiterated Rating	Outperform		Low	
11/19/2020	The Goldman Sachs Group	Reiterated Rating	Sell	\$75.00	Low	
11/17/2020	Credit Suisse Group	Set Price Target	Neutral	\$106.00	Low	
11/16/2020	JPMorgan Chase & Co.	Set Price Target	Buy	\$150.00	Low	
10/30/2020	Fundamental Research	Boost Price Target	Hold → Hold	\$112.92 → \$113.91	Low	

As part of the exercise, following additional aspects were required to be taken care: 1. the time delay in reading the complete web document, and 2. the URL redirects - in case a particular stock was not found on the website, website would redirect the URL request to a default page.

```
# create a webdriver object and set options for headless browsing
def load_webdriver():
    if str(filePath).find("CS410") == -1:
        options = webdriver.ChromeOptions()
        options.add_argument('--no-sandbox')
        options.add_argument('--headless')
        options.add_argument("--disable-dev-shm-usage")
        driver = webdriver.Chrome(chrome_options=options)
    else:
        options = Options()
        options.headless = True
        driver = webdriver.Chrome(filePath / 'chromedriver', options=options)
    return driver
```

```
# read web document using beutifulsoup
def get_js_soup(url_web, driver):
    driver.get(url_web)
    time.sleep(5)
    if url_web != redirected_url:
        print("redirected URL : " + redirected_url)
        new_url = redirected_url + "price-target/"
        print("new URL for Analysts rating : " + new_url)

        if new_url.find('NASDAQ/price-target/') == -1:
            driver.get(url_web)
        else:
            return
```

Using the html div class identifier, relevant data were extracted from the webpage data and subsequently processed in panda dataframe.

```
table = soup_obj.find("table",
      attrs={"class": "scroll-table sort-table fixed-left-column fixed-header"})

if table is None:
    table = soup_obj.find("table",
      attrs={"class": "scroll-table sort-table fixed-header"})

if table is None:
    table = soup_obj.find("table",
      attrs={"class": "scroll-table sort-table"})

if table is None:
    print(stock_symbol + ": No table found for market analyst rating")
else:
    table_body = table.find('tbody')
    rows = table_body.find_all('tr')
    for row in rows:
        cols = row.find_all('td')
        cols = [ele.text.strip() for ele in cols]
        data.append([ele.split("➞")[-1].strip() for ele in cols if ele])
```

C. Uniform Scaling of Ratings

Uniform scaling of ratings was required to unify the ratings from various market analysts and ensure that the calculation of overall rating is without any bias and is not affected by some higher or lower ratings from some market analysts, only due to the fact that they use different scales for rating a stock.

Various ratings were mapped to {-1: SELL, 0: HOLD, 1: BUY} using following mapping dictionary is used.

```
ratings_dict = {
    "SELL": -1, "STRONG SELL": -1, "BEARISH": -1, "UNDERPERFORM": -1,
    "SECTOR UNDERPERFORM": -1, "MODERATE SELL": -1, "WEAK HOLD": -1,
    "UNDERWEIGHT": -1, "REDUCE": -1,
    "HOLD": 0, "NEUTRAL": 0, "AVERAGE": 0,
    "MARKET PERFORM": 0, "SECTOR PERFORM": 0, "SECTOR WEIGHT": 0,
    "PEER PERFORM": 0, "EQUAL WEIGHT": 0, "IN-LINE": 0,
    "MARKET OUTPERFORM": 1,
    "OUTPERFORM": 1, "MODERATE BUY": 1, "ACCUMULATE": 1, "OVER-WEIGHT": 1,
    "OVERWEIGHT": 1, "STRONG-BUY": 1, "ADD": 1, "BULLISH": 1, "BUY": 1,
    "POSITIVE": 1, "STRONG BUY": 1, "TOP PICK": 1, "CONVICTION-BUY": 1,
    "OUTPERFORMER": 1
}
```

There were other options also considered e.g., scaling of all ratings on a scale of 1-5 or 1-3, weighted scale to reflect strong ratings e.g., strong buy or strong sell. However, based on our need to combine the analyst rating with twitter sentiment, scaling to {-1: SELL, 0: HOLD, 1: BUY} were selected for this exercise.

D. Overall Ratings Calculation

First of all, aggregated ratings of all ratings from various market analyst is calculated based on arithmetic mean of all selected scaled ratings. e.g., aggregated rating for following five scaled ratings {-1: SELL, 1: BUY, 0: HOLD, 1: BUY, -1: SELL} from various market analyst will be {0: HOLD}.

Once the aggregated rating is calculated, it is combined with Twitter Sentiment analysis result as per below table for Overall ratings.

Aggregated Rating	Twitter Sentiment	Overall Ratings
-1: SELL	POSITIVE	HOLD
-1: SELL	NEUTRAL	SELL
-1: SELL	NEGATIVE	SELL
0: HOLD	POSITIVE	BUY
0: HOLD	NEUTRAL	HOLD
0: HOLD	NEGATIVE	SELL
1: BUY	POSITIVE	BUY
1: BUY	NEUTRAL	BUY
1: BUY	NEGATIVE	HOLD

Sentiment Analysis

Sentiment Analysis, also known as Opinion Mining, refers to the use of Natural Language Processing to computationally determine opinions and emotions of an opinion holder for an opinion target. A common use for this technology is to discover how people feel about certain topics, particularly through users' textual posts in Social Media space. To perform Sentiment Analysis to provide stock recommendation, Twitter has been considered as the Social Media space where users post their opinion as their tweets. As most of the elements in the opinion representation such as the opinion holder (Twitter users in this case) and opinion target (Stock to be recommended in this case) and the content and the context of the opinion (financial context) are already known, the main task is to decide opinion sentiment. So, this is a case of just using sentiment classification for understanding opinion where the input is opinionated text object, the output is a sentiment label i.e., polarity analysis with predefined categories such as positive, negative, or neutral. For this project, the sentiment analysis has been done through the process outlined below.

Process Description

A. Preparing the Data Set

To build the model, training and testing data set is needed. Ideally for optimal performance the financial tweets need to be downloaded from tweeter and should be human evaluated to create the labels which can be used for training the model and later for testing the model. However, one needs to spend considerable amount of manual effort to build such data. To optimize time and resource for this project, a readily available downloadable training set (polarity dataset from Cornell university -) has been used. The data set contain 2000 processed down-cased text files used in Pang/Lee ACL 2004 [1]; the names of the two subdirectories in that folder, "pos" and "neg", indicate the true classification (sentiment) of the component files according to the automatic rating classifier the tweets of the data set have been all labeled as positive or negative, depending on the content. The data set have been persisted into pickle file (a binary representation of python structure) to optimize performance of the subsequent run to build the classifier.

```
import pickle
from sklearn.datasets import load_files

#nltk.download('stopwords')

#Import Dataset -> generate two classes one each for each sub directorties
dataset = load_files('txt_sentoken/')

X,y = dataset.data, dataset.target

#store as pickle file, these are byte type file

with open('X.pickle', 'wb') as f:
    pickle.dump(X,f)

with open('y.pickle', 'wb') as f:
    pickle.dump(y,f)
```

The same data has been split into training and testing data set following a 80-20 rule where 80% of the downloaded pre-labeled data has been used as training data set and 20% of the same downloaded data has been used as testing data set.

B. Preprocessing the Data Set

The downloaded data set has been preprocessed before feeding into the program to create the Classifier to remove all the non-word characters, to convert into lower case, to remove single characters (e.g. i, a etc.).

```
import re
import pickle

with open('X.pickle', 'rb') as f:
    X=pickle.load(f)

with open('y.pickle', 'rb') as f:
    y=pickle.load(f)

corpus = []

for i in range(0,len(X)):
    data = re.sub(r'\W', ' ', str(X[i]))
    data = data.lower()
    data = re.sub(r'\s+[a-z]\s+', ' ', data)
    data = re.sub(r'^[a-z]\s+', ' ', data)
    data = re.sub(r'\s+', ' ', data)
    corpus.append(data)
```

C. Building the BOW, TF-IDF and Logistic Regression Classifier

Scikit-learn library (a free machine learning library for Python) has been used to create the Classifier.

At first, the bag of words model has been created and later the bag of words model would be converted into TF-IDF model. To covert the data into bag of words model, classes from Scikit-learn has been used. First, a max feature has been set to 2000 which means 2000 most frequent words would be used as features. The min document frequency would ensure to exclude a word to be considered as 2000 features which appear 3 or less documents (to prevent a word to become a feature which is rare into the set but very popular within a certain document) and the max document frequency would ensure to exclude a word to be considered as 2000 features which appear 60% or more documents (to exclude the most common words like the, an etc.). Then Stop words has been removed that is defined in nltk corpus. Now to form the Bag of Words model the corpus from above steps has been used.

```
from nltk.corpus import stopwords
from sklearn.feature_extraction.text import CountVectorizer

from nltk.corpus import stopwords
nltk.download('stopwords')

vectorizer = CountVectorizer(stop_words=stopwords.words('english'), max_df = 0.6, min_df = 3,
max_features = len(X))

X = vectorizer.fit_transform(corpus).toarray()#will generate 2D array [len(X),len(X)], total
number of docs = number of features = len(x)
```

Finally, TfidfTransformer from sklearn is used to create TF-IDF model from Bag of Words model created earlier.

```
from sklearn.feature_extraction.text import TfidfTransformer

# convert BOW to TF-IDF
transformer = TfidfTransformer()
X = transformer.fit_transform(X).toarray()
```

Logistic Regression (a Discriminative Classifier) is a classification algorithm (learning algorithm) which is used for binary classification problem. The task of sentiment analysis for this project can be thought as a binary classification problem where the goal is to predict positive or negative sentiment from a given tweets (a sentence in particular). Hence for the purpose of this project, a logistics regression classifier has been built where negative and positive sentiments have been denoted as 0 and 1 respectively.

So, the Binary Response Variable Y Є {0,1} needs to be calculated from the predictor X where X = { x_1, x_2 ….. x_{2000}} (all the 2000 features)

Hence, for

$$Y = \begin{cases} 1 & \text{category}(d) = \theta_1 \\ 0 & \text{category}(d) = \theta_2 \end{cases}$$

Logistics Regression can be represented as below:

$$\log\frac{p(\theta_1 \mid d)}{p(\theta_2 \mid d)} = \log\frac{p(Y=1\mid X)}{p(Y=0\mid X)} = \log\frac{p(Y=1\mid X)}{1-p(Y=1\mid X)} = \beta_0 + \sum_{i=1}^{M} x_i\beta_i \quad \beta_i \in \Re$$

So essentially using training data T, , parameters (M=2000) needs to be estimated.

Hence the conditional likelihood estimate is:

$$p(Y=1\mid X) = \frac{e^{\beta_0+\Sigma_{i=1}^{M} x_i\beta_i}}{e^{\beta_0+\Sigma_{i=1}^{M} x_i\beta_i}+1}$$

and

$$p(Y=0\mid X) = \frac{1}{e^{\beta_0+\Sigma_{i=1}^{M} x_i\beta_i}+1}$$

The goal of Logistic Regression algorithm is to optimize the parameters using training data set. Once the optimal values of the parameters are found by the algorithm, y can be calculated for any unknown sentiment of a new sentence. If y>=0.5, then that sentiment is positive sentiment and if y<0.5, then that sentiment is negative sentiment.

For this project, LogisticRegression from sklearn has been used to build the classifier.

First the input data set has been split as - 80% of available data has been considered as training data and 20% of the available data has been considered as test data

```
from sklearn.model_selection import train_test_split

text_train,text_test,sentiment_train,sentiment_test =
train_test_split(X,y,test_size=0.2,random_state=0)#80% training and 20% testing data
```

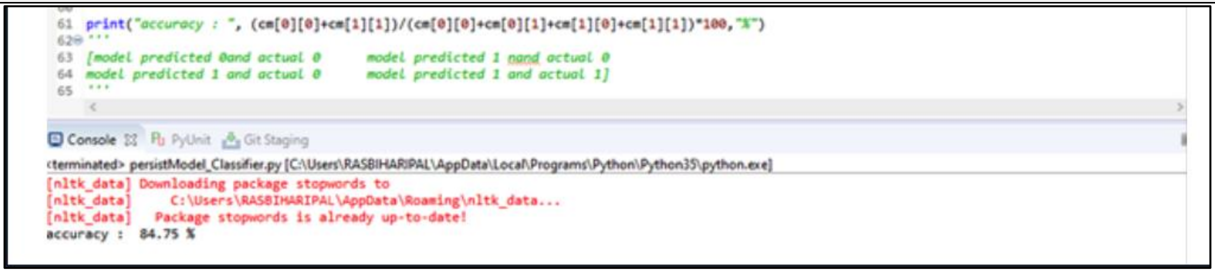
Then classifier is built:

```
from sklearn.linear_model import LogisticRegression

# create the classifier using logistic regression
classifier = LogisticRegression()
classifier.fit(text_train,sentiment_train)
```

Just to showcase the model performance, confusion_matrix class from sklearn has been used. With the input data, close to 85% accuracy has been achieved.

from sklearn.metrics import confusion_matrix

```
#testing model performance
sentiment_prediction = classifier.predict(text_test)
cm = confusion_matrix(sentiment_test,sentiment_prediction)# [[predicted as 0 and actually 0,
predicted as 0 and actually 1],
#[predicted as 1 and actually 0, predicted as 1 and actually 1]]


print("accuracy : ", (cm[0][0]+cm[1][1])/(cm[0][0]+cm[0][1]+cm[1][0]+cm[1][1])*100,"%")
'''
[model predicted 0and actual 0      model predicted 1 nand actual 0
model predicted 1 and actual 0      model predicted 1 and actual 1]
'''
```



```
print("accuracy : ", (cm[0][0]+cm[1][1])/(cm[0][0]+cm[0][1]+cm[1][0]+cm[1][1])*100,"%")
'''
[model predicted 0and actual 0      model predicted 1 nand actual 0
model predicted 1 and actual 0      model predicted 1 and actual 1]
'''
```

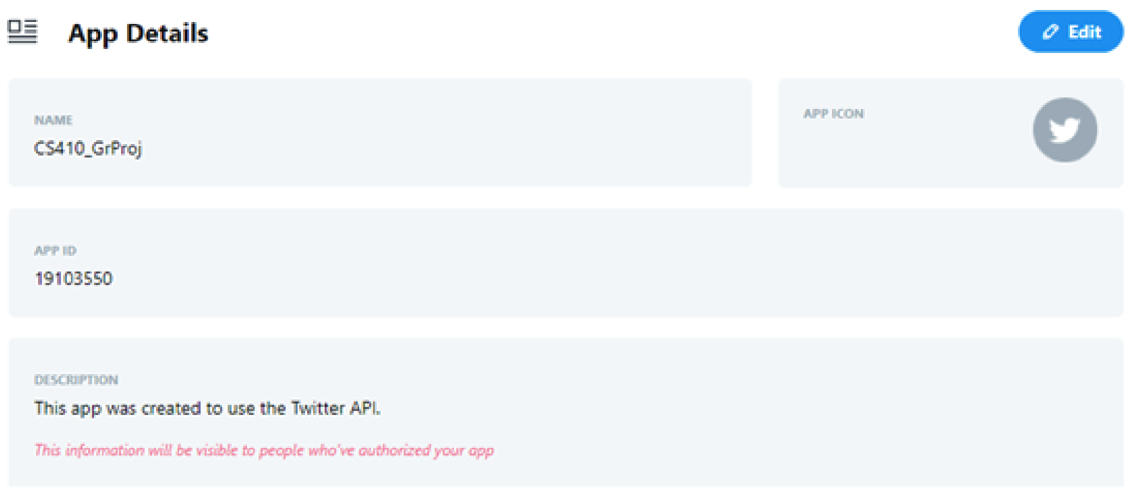
```
<terminated> persistModel_Classifier.py [C:\Users\RASBIHARIPAL\AppData\Local\Programs\Python\Python35\python.exe]
[nltk_data] Downloading package stopwords to
[nltk_data]     C:\Users\RASBIHARIPAL\AppData\Roaming\nltk_data...
[nltk_data]   Package stopwords is already up-to-date!
accuracy :  84.75 %
```

Finally, the model and vectorizer have been stored as pickle file (binary representation of python objects) so that while calculating the real time tweets, the saved model can be used as it is.

```
from sklearn.feature_extraction.text import TfidfVectorizer

#store the classifier ....pickle file

with open('classifier.pickle', 'wb') as f:
    pickle.dump(classifier,f)

#store the TFIDF vectorizer
vectorizerTFIDF = TfidfVectorizer(stop_words=stopwords.words('english'), max_df = 0.6, min_df
= 3, max_features = len(X))
X_TDIDF = vectorizerTFIDF.fit_transform(corpus).toarray()

with open('vectorizerTFIDF.pickle', 'wb') as f:
    pickle.dump(vectorizerTFIDF,f)
```

D. Fetching the real time tweets

A developer app has been created in Twitter for the purpose of this project.



NAME
CS410_GrProj

APP ID
19103550

DESCRIPTION
This app was created to use the Twitter API.
This information will be visible to people who've authorized your app

The OAuth2 mechanism has been used to make API calls to Twitter API for fetching the recent tweets.

ConsumerKey and ConsumerSecret from the created app are used to generate bearer token in the runtime API call using https://api.twitter.com/oauth2/token

Finally, the bearer token is used to call API to perform recent search (returns Tweets from the last 7 days that match a search query) using “recent search” API. https://developer.twitter.com/en/docs/twitter-api/tweets/search/api-reference/get-tweets-search-recent

For this project, Max result of 100 has been set to perform recent search. This is to ensure throttling of number of results fetched (as for the basic access, twitter account has 500000 search results/month limitation)

E. Performing Sentiment analysis of the fetched tweets

To perform sentiment analysis in real time, firstly the Logistic Regression Classifier model and TF-IDF vectorizer is loaded from the saved pickle file.

```
import pickle
import re

with open('classifier.pickle', 'rb') as f:
    clf=pickle.load(f)

with open('vectorizerTFIDF.pickle', 'rb') as f:
    vectorizer=pickle.load(f)
```

For a given stock the “recent search” API is used to search recent tweets. After fetching related tweets (max = 100), each tweet is preprocessed to create bag of words and to be represented as TF-IDF vectorizer. Then, prebuilt Logistic Regression Classifier is used to predict the sentiment of each tweets.

```
  for t in tweets_fetched:
        t = re.sub(r'^https://t.co/[a-zA-Z0-9]*\s',' ',t)
        t = re.sub(r'\s+https://t.co/[a-zA-Z0-9]*\s',' ',t)
        t = re.sub(r'\s+https://t.co/[a-zA-Z0-9]*$',' ',t)
        t = t.lower()
        t = re.sub(r"that's",'that is',t)
        t = re.sub(r"there's",'there is',t)
        t = re.sub(r"what's",'what is',t)
        t = re.sub(r"where's",'where is',t)
        t = re.sub(r"it's",'it is',t)
        t = re.sub(r"who's",'who is',t)
        t = re.sub(r"i'm",'i am',t)
        t = re.sub(r"she's",'she is',t)
        t = re.sub(r"he's",'he is',t)
        t = re.sub(r"they're",'they are',t)
        t = re.sub(r"who're",'who are',t)
        t = re.sub(r"shouldn't",'should not',t)
        t = re.sub(r"wouldn't",'would not',t)
        t = re.sub(r"couldn't",'could not',t)
        t = re.sub(r"can't",'can not',t)
        t = re.sub(r"won't",'will not',t)
        t = re.sub(r'\W', ' ', t)
        t = re.sub(r'\d', ' ', t)
        t = re.sub(r'\s+[a-z]\s+', ' ', t)
        t = re.sub(r'\s+[a-z]$', ' ', t)
        t = re.sub(r'^[a-z]\s+', ' ', t)
        t = re.sub(r'\s+', ' ', t)
        sentiment = clf.predict(vectorizer.transform([t]).toarray())
```

Finally, total positive and negative sentiments are calculated for total tweets fetched in runtime and the final sentiment of a particular stock has been calculated as positive = 1 (if the positive percentage > 65%), negative = -1 (if the positive percentage < 35%) or neutral = 0 (if the positive percentage is in between 35% and 65%).

```
    if (tot_positive+tot_negetive)>0 :
        positive_percentage = tot_positive/(tot_positive+tot_negetive)
    print("positive_percentage :",positive_percentage*100, "%")

    if positive_percentage>0.65 :
        print("stock is buy")
        return 1
    elif positive_percentage<0.35 and positive_percentage>0:
        print("stock is sell")
        return -1
    else:
        print("stock is neutral")
        return 0
```

Following end point has been built to provide twitter sentiment analysis result to a stock.

Definition

- Get Twitter sentiment for given stock - 1: Positive, 0: Neutral, -1: Negative

  `GET /stock/sentiments/<market>/<stock_symbol>`

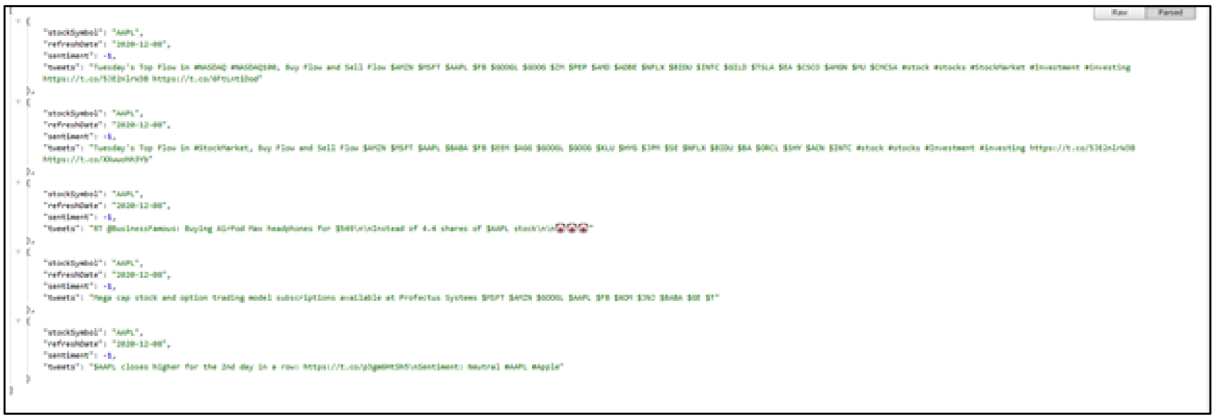
e.g. http://localhost:5000/stock/sentiments/NASDAQ/AAPL

Response

- 200 OK on success

```
json { "stockSymbol": "AAPL", "refreshDate": "2020-11-25", "sentiment": 1 }
```

Sample run:



```
[
  {
    "stockSymbol": "AAPL",
    "refreshDate": "2020-12-08",
    "sentiment": -1,
    "tweets": "Tuesday's Top Flow in #NASDAQ #NASDAQ100, Buy Flow and Sell Flow $AMZN $MSFT $AAPL $FB $GOOGL $GOOG $ZM $PEP $AMD $ADBE $NFLX $BIDU $INTC $GILD $TSLA $BA $CSCO $AMGN $MU $CMCSA #stock #stocks #StockMarket #Investment #investing
https://t.co/[illegible] https://t.co/[illegible]"
  },
  {
    "stockSymbol": "AAPL",
    "refreshDate": "2020-12-08",
    "sentiment": -1,
    "tweets": "Tuesday's Top Flow in #StockMarket, Buy Flow and Sell Flow $AMZN $MSFT $AAPL $BABA $FB $BRK $AGG $GOOGL $GOOG $XLU $HYG $JPM $GE $NFLX $BIDU $BA $ORCL $SHY $ACN $INTC #stock #stocks #Investment #investing https://t.co/[illegible]
https://t.co/[illegible]"
  },
  {
    "stockSymbol": "AAPL",
    "refreshDate": "2020-12-08",
    "sentiment": -1,
    "tweets": "RT @BusinessFamous: Buying AirPod Max headphones for $549\n\nInstead of 4.4 shares of $AAPL stock\n\n😭😭😭"
  },
  {
    "stockSymbol": "AAPL",
    "refreshDate": "2020-12-08",
    "sentiment": -1,
    "tweets": "Mega cap stock and option trading model subscriptions available at Profectus Systems $MSFT $AMZN $GOOGL $AAPL $FB $NKE $[illegible] $BABA $GE $T"
  },
  {
    "stockSymbol": "AAPL",
    "refreshDate": "2020-12-08",
    "sentiment": -1,
    "tweets": "$AAPL closes higher for the 2nd day in a row: https://t.co/[illegible]\nSentiment: Neutral #AAPL #Apple"
  }
]
```

Recommender System

Approach

The basic approach to the recommendation engine is to use a similarity function to compare pre-identified features of all stocks in our corpus against user-provided stock symbol features. While doing so, we are implementing one of the methods for building recommendation engine - Content based recommendation or Item similarity, which was part of Week 6 lecture of course CS 410 Text Information System.

Recommender System recommends to the user the top 5 stock symbols which are most similar to the user provided stock symbol in terms of those underlying features and the similarity function. For the initial, ***Cosine similarity*** is used as similarity function and the following financial/economic features were selected as underlying features for similarity calculation.

1. company is in S&P 500

2. company profitability over last three years
3. revenue growth for last three years
4. current market analyst ratings
5. sector
6. gross profit per market cap

Analyst ratings are calculated by rating_system and same is reused also for recommendation system. Other features for the stock are calculated or derived using the financial data of the companies from year 2018. Source: Kaggle dataset available at https://www.kaggle.com/cnic92/200-financial-indicators-of-us-stocks-20142018.

As part of the exercise, significant amount of effort was used to identify a single data source for financial or economic data and also to understand various financial terms to find the right feature. Based on our current understanding of a company performance and its relation to stock price, we used above listed features. The features might be further optimized with right guidance from a financial analyst or person with insight of stock market and factors influencing the investor decisions.

Libraries and Datasets

Python Libraries

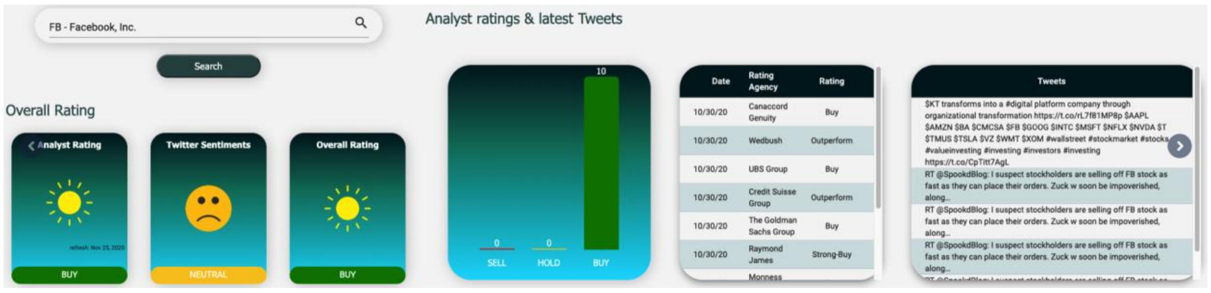
```
Flask
flask_cors
markdown
beautifulsoup4
selenium
pandas
pymongo
tinydb
misaka
nltk
requests
sklearn
flask-jsonpify
```

Datasets

- https://www.marketbeat.com/
- https://www.cs.cornell.edu/people/pabo/movie-review-data/
- https://www.kaggle.com/cnic92/200-financial-indicators-of-us-stocks-20142018?select=2018_Financial_Data.csv
- https://developer.twitter.com

Verification

The result of the recommender system was verified based on manual verification process by selecting some stocks at random and checking the result for individual section e.g., ratings, sentiments and overall ratings. For example, Overall rating for Facebook (FB) stock is BUY. Looking at individual ratings component, the result seems reasonable.



Date	Rating Agency	Rating
10/30/20	Canaccord Genuity	Buy
10/30/20	Wedbush	Outperform
10/30/20	UBS Group	Buy
10/30/20	Credit Suisse Group	Outperform
10/30/20	The Goldman Sachs Group	Buy
10/30/20	Raymond James	Strong-Buy
	Monness	

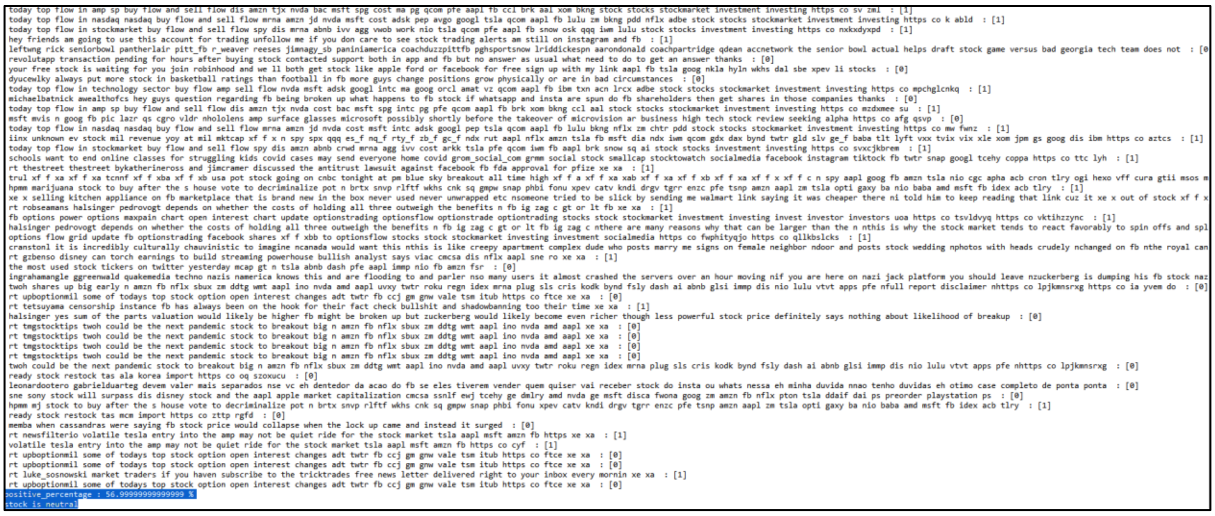
Tweets
$KT transforms into a #digital platform company through organizational transformation https://t.co/rL7f81MP8p $AAPL $AMZN $BA $CMCSA $FB $GOOG $INTC $MSFT $NFLX $NVDA $T $TMUS $TSLA $VZ $WMT $XOM #wallstreet #stockmarket #stocks #valueinvesting #investing #investors #investing https://t.co/CpTitt7AgL
RT @SpookdBlog: I suspect stockholders are selling off FB stock as fast as they can place their orders. Zuck w soon be impoverished, along...
RT @SpookdBlog: I suspect stockholders are selling off FB stock as fast as they can place their orders. Zuck w soon be impoverished, along...
RT @SpookdBlog: I suspect stockholders are selling off FB stock as fast as they can place their orders. Zuck w soon be impoverished, along...

Analyst Ratings: BUY

Verification: In the list of market analyst ratings, all 10 analysts have rated it as BUY. Therefore, the aggregated rating is BUY.

Tweet Sentiments: Neutral

Verification: From the latest 100 tweets fetched (as shown in below screenshot), it seems reasonable to have a NEUTRAL sentiment



```
today top flow in amp sp buy flow and sell flow dis amzn tjx nvda bac msft spg cost ma pg qcom pfe aapl fb ccl brk aal xom bkng stock stocks stockmarket investment investing https co sv zml  : [1]
today top flow in nasdaq nasdaq buy flow and sell flow mrna amzn jd nvda msft cost adsk pep avgo googl tsla qcom aapl fb lulu zm bkng pdd nflx adbe stock stocks stockmarket investment investing https co k abld  : [1]
today top flow in stockmarket buy flow and sell flow spy dis mrna abnb ivv agg vwob work nio tsla qcom pfe aapl fb snow osk qqq iwm lulu stock stocks investment investing https co nxkxdyxpd  : [1]
hey friends am going to use this account for trading unfollow me if you don care to see stock trading alerts am still on instagram and fb  : [1]
leftwng rick seniorbowl pantherlair pitt_fb r_weaver reeses jimnagy_sb paniniamerica coachduzzpittfb pghsportsnow lriddickespn aarondonald coachpartridge qdean accnetwork the senior bowl actual helps draft stock game versus bad georgia tech team does not  : [0
revolutapp transaction pending for hours after buying stock contacted support both in app and fb but no answer as usual what need to do to get an answer thanks  : [0]
your free stock is waiting for you join robinhood and we ll both get stock like apple ford or facebook for free sign up with my link aapl fb tsla goog nkla hyln wkhs dal sbe xpev li stocks  : [0]
dyucewlky always put more stock in basketball ratings than football in fb more guys change positions grow physically or are in bad circumstances  : [0]
today top flow in technology sector buy flow amp sell flow nvda msft adsk googl intc ma goog orcl amat vz qcom aapl fb ibm txn acn lrcx adbe stock stocks stockmarket investment investing https co mpchglcnkq  : [1]
michaelbatnick awealthofcs hey guys question regarding fb being broken up what happens to fb stock if whatsapp and insta are spun do fb shareholders then get shares in those companies thanks  : [0]
today top flow in amp sp buy flow and sell flow dis amzn tjx nvda cost bac msft spg intc pg pfe qcom aapl fb brk xom bkng ccl aal stock stocks stockmarket investment investing https co mzdxmee su  : [1]
msft mvis n goog fb pic lazr qs cgro vldr nhololens amp surface glasses microsoft possibly shortly before the takeover of microvision ar business high tech stock review seeking alpha https co afg qsvp  : [0]
today top flow in nasdaq nasdaq buy flow and sell flow mrna amzn jd nvda cost msft intc adsk googl pep tsla qcom aapl fb lulu bkng nflx zm chtr pdd stock stocks stockmarket investment investing https co mw fwnz  : [1]
iinx unknown ev stock mil revenue yoy at mil mktcap xf f x n spy spx qqq es_f nq_f rty_f zb_f gc_f ndx rut aapl nflx amzn tsla fb msft dia ndx iwm qcom gdx dax bynd twtr gld slv ge_f baba tlt lyft vxx tvix vix xle xom jpm gs goog dis ibm https co aztcs  : [1]
today top flow in stockmarket buy flow and sell flow spy dis amzn abnb crwd mrna agg ivv cost arkk tsla pfe qcom iwm fb aapl brk snow sq ai stock stocks investment investing https co svxcjkbrem  : [1]
schools want to end online classes for struggling kids covid cases may send everyone home covid grom_social_com grmm social stock smallcap stocktowatch socialmedia facebook instagram tiktock fb twtr snap googl tcehy coppa https co ttc lyh  : [1]
rt thestreet thestreet bykatherineross and jimcramer discussed the antitrust lawsuit against facebook fb fda approval for pfize xe xa  : [1]
trul xf f xa xf f xa tcnnf xf f xba xf f xb usa pot stock going on cnbc tonight at pm blue sky breakout all time high xf f a xf f xa xab xf f xa xf f xb xf f xa xf f x xf f c n spy aapl goog fb amzn tsla nio cgc apha acb cron tlry ogi hexo vff cura gtii msos m
hpmm marijuana stock to buy after the s house vote to decriminalize pot n brtx snvp rlftf wkhs cnk sq gmpw snap phbi fonu xpev catv kndi drgv tgrr enzc pfe tsnp amzn aapl zm tsla opti gaxy ba nio baba amd msft fb idex acb tlry  : [1]
xe x selling kitchen appliance on fb marketplace that is brand new in the box never used never unwrapped etc nsomeone tried to be slick by sending me walmart link saying it was cheaper there ni told him to keep reading that link cuz it xe x out of stock xf f x
rt robseamans halsinger pedrovogt depends on whether the costs of holding all three outweigh the benefits n fb ig zag c gt or lt fb xe xa  : [1]
fb options power options maxpain chart open interest chart update optionstrading optionsflow optionstrade optiontrading stocks stock stockmarket investment investing invest investor investors uoa https co tsvldvyq https co vktihzzync  : [1]
halsinger pedrovogt depends on whether the costs of holding all three outweigh the benefits n fb ig zag c gt or lt fb ig zag c nthere are many reasons why that can be larger than the n nthis is why the stock market tends to react favorably to spin offs and spl
options flow grid update fb optionstrading facebook shares xf f xbb to optionsflow stocks stock stockmarket investing investment socialmedia https co fwphityqjo https co qllkbslcks  : [1]
cranstonl it is incredibly culturally chauvinistic to imagine ncanada would want this nthis is like creepy apartment complex dude who posts marry me signs on female neighbor ndoor and posts stock wedding nphotos with heads crudely nchanged on fb nthe royal can
rt gzbenso disney can torch earnings to build streaming powerhouse bullish analyst says viac cmcsa dis nflx aapl sne ro xe xa  : [1]
the most used stock tickers on twitter yesterday mcap gt n tsla abnb dash pfe aapl immp nio fb amzn fsr  : [0]
ingrahamangle ggreenwald quakemedia techno nazis namerica knows this and are flooding to and parler nso many users it almost crashed the servers over an hour moving nif you are here on nazi jack platform you should leave nzuckerberg is dumping his fb stock naz
twoh shares up big early n amzn fb nflx sbux zm ddtg wmt aapl ino nvda amd aapl uvxy twtr roku regn idex mrna plug sls cris kodk bynd fsly dash ai abnb glsi immp dis nio lulu vtvt apps pfe nfull report disclaimer nhttps co lpjkmnsrxg https co ia yvem do  : [0]
rt upboptionmil some of todays top stock option open interest changes adt twtr fb ccj gm gnw vale tsm itub https co ftce xe xa  : [0]
rt tetsuyama censorship instance fb has always been on the hook for their fact check bullshit and shadowbanning too their time xe xa  : [1]
halsinger yes sum of the parts valuation would likely be higher fb might be broken up but zuckerberg would likely become even richer though less powerful stock price definitely says nothing about likelihood of breakup  : [0]
rt tmgstocktips twoh could be the next pandemic stock to breakout big n amzn fb nflx sbux zm ddtg wmt aapl ino nvda amd aapl xe xa  : [0]
rt tmgstocktips twoh could be the next pandemic stock to breakout big n amzn fb nflx sbux zm ddtg wmt aapl ino nvda amd aapl xe xa  : [0]
rt tmgstocktips twoh could be the next pandemic stock to breakout big n amzn fb nflx sbux zm ddtg wmt aapl ino nvda amd aapl xe xa  : [0]
rt tmgstocktips twoh could be the next pandemic stock to breakout big n amzn fb nflx sbux zm ddtg wmt aapl ino nvda amd aapl xe xa  : [0]
twoh could be the next pandemic stock to breakout big n amzn fb nflx sbux zm ddtg wmt aapl ino nvda amd aapl uvxy twtr roku regn idex mrna plug sls cris kodk bynd fsly dash ai abnb glsi immp dis nio lulu vtvt apps pfe nhttps co lpjkmnsrxg  : [0]
ready stock restock tas ala korea import https co oq szoxucu  : [0]
leonardootero gabrielduarteg devem valer mais separados nse vc eh dentedor da acao do fb se eles tiverem vender quem quiser vai receber stock do insta ou whats nessa eh minha duvida nnao tenho duvidas eh otimo case completo de ponta ponta  : [0]
sne sony stock will surpass dis disney stock and the aapl apple market capitalization cmcsa ssnlf ewj tcehy ge dmlry amd nvda ge msft disca fwona goog zm amzn fb nflx pton tsla ddaif dai ps preorder playstation ps  : [0]
hpmm mj stock to buy after the s house vote to decriminalize pot n brtx snvp rlftf wkhs cnk sq gmpw snap phbi fonu xpev catv kndi drgv tgrr enzc pfe tsnp amzn aapl zm tsla opti gaxy ba nio baba amd msft fb idex acb tlry  : [1]
ready stock restock tas mcm import https co zttp rgfd  : [0]
memba when cassandras were saying fb stock price would collapse when the lock up came and instead it surged  : [0]
rt newsfilterio volatile tesla entry into the amp may not be quiet ride for the stock market tsla aapl msft amzn fb https xe xa  : [1]
volatile tesla entry into the amp may not be quiet ride for the stock market tsla aapl msft amzn fb https co cyf  : [1]
rt upboptionmil some of todays top stock option open interest changes adt twtr fb ccj gm gnw vale tsm itub https co ftce xe xa  : [0]
rt upboptionmil some of todays top stock option open interest changes adt twtr fb ccj gm gnw vale tsm itub https co ftce xe xa  : [0]
rt luke_sosnowski market traders if you haven subscribe to the tricktrades free news letter delivered right to your inbox every mornin xe xa  : [1]
rt upboptionmil some of todays top stock option open interest changes adt twtr fb ccj gm gnw vale tsm itub https co ftce xe xa  : [0]
positive_percentage : 56.99999999999999 %
stock is neutral
```

Overall Rating: BUY

Verification: Combining Analyst Ratings “Buy” with Twitter Sentiments “NEUTRAL”, as per listed decision table, the result is BUY.

Conclusion and Future Work

Overall, the Stock recommending system as described in this project report has yielded satisfactory result in recommending rating (e.g., buy, hold, sell) for the user entered Stock symbol and recommending five similar stocks. The result has been validated by some popular stocks. One of key challenges faced in building the system is the access on right financial data as well as good training and testing set to train and test the model for sentiment analysis. The choice we had to manually create the required data. However, to optimize the time and resource available, it has been decided to use readily available data in the web which might have not yielded the perfect recommendation.

There are several interesting directions for the future version of recommender system. First, the overall functionality can be improved by considering user input of sectors and providing recommendation and trends specific to that sector. A second direction involves defining the test and train data and possibly human labeling twitter feed (for sentiment analysis) just for tweets related to stock market and use the same to train and test the model. Given the current trend of machine learning algorithms, a third interesting research direction is to explore the timeseries data for stock adjacency. Finally, overall score can be improved further to consider more analysts' reports and microblogging websites and come up with more recommender categories i.e., Strong Buy, Buy, Hold, Sell, Strong Sell etc.

At the end, it has been a great journey of ideation and learning in a collaborative manner to design and develop the stock recommender system. We thank different analysts' websites to make their ratings available publicly and twitter to grant access of real time tweets through their public API and last but not the least, we thank professor ChengXiang Zhai and all our TAs and all the reviewers to give the direction needed and to provide the valuable feedback.

References

[1] Bo Pang and Lillian Lee. 2004. A sentimental education: sentiment analysis using subjectivity summarization based on minimum cuts. In Proceedings of the 42nd Annual Meeting on Association for Computational Linguistics (ACL '04). Association for Computational Linguistics, USA, 271–es. DOI:https://doi.org/10.3115/1218955.1218990

[2] Carbone, N. (2020, January 18). 200+ Financial Indicators of US stocks (2014-2018). Retrieved December 11, 2020, from https://www.kaggle.com/cnic92/200-financial-indicators-of-us-stocks-20142018