Analyzing Stock Market Movements Using Twitter Sentiment Analysis

Nirali Patel
Computer Science
Illinois Institute of Technology
Chicago
npate104@hawk.iit.edu

Jaimin Sanghvi
Computer Science
Illinois Institute of Technology
Chicago
jsanghv2@hawk.iit.edu

***Abstract*— In contemporary era, the use of social media has reached unprecedented levels. Among all social media, Twitter is such a popular micro-blogging service, which enables users to share short messages in real time about events or express own opinion. In this paper, we examine the effectiveness of various machine learning techniques on retrieved tweet corpus. We apply machine learning model to predict tweet sentiment as well as find the correlation between twitter sentiment and stock prices. We accomplish this by mining tweets using Twitter's search API and process it for further analysis. To determine tweet sentiment, we test the effective two machine learning techniques: Naïve Bayes classification and Support vector machines. By evaluating each model, we discovered that support vector machine gives higher accuracy though cross validation. After predicting tweet sentiment, we have mined stock historical data using Yahoo finance API. We have designed feature matrix for stock market prediction using positive, negative, neutral and total sentiment score and stock price for each day. We have applied same machine learning algorithm to determine correlation between tweet sentiments and stock market prices and analyzed how tweet sentiments directly correlates with stock market prices.**

Keywords—stock market analysis; sentiment analysis; twitter; microblogging; prediction

I. INTRODUCTION

In modern era, vast amount of data transmitted online through different social media channels. Data contains information about virtually every topic. Twitter is microblogging platform where 100 million user login daily and more than 500 million tweets are sent per day. Each tweet has maximum of 140 characters hence more than 70 billion characters generated per day only on twitter. Though each tweet may not valuable, we can extract important data that provide valuable insight about public mood and sentiment score on certain topic.

Recently, in field of computer science it is an interesting topic to collect vast amount of data from social website, process it and running data analysis on retrieved data to extract relevant information. In case of twitter, we have collected large amount live data, process tweets and generated feature matrix to apply machine learning algorithm.

In the stated application, historical finance data is also required to make prediction about stock prices. There are bunch of factors that are involved in prediction of stock prices as well as tweet sentiments. In this paper, we take look at two different machine learning algorithms that mostly use for classification and examine their effectiveness on twitter corpus. We have used implemented techniques in two different ways: 1) To predict tweet sentiment score based on classification co-efficient and 2) The correlation between tweet sentiment and stock market prices.

II. PROBLEM STATEMENT

- How to utilize social media to assess market sentiment and predict the behavior of stock of certain company, market and stock indexes to identify an opportunity for trading?
- In order to design feasible system for identifying an opportunity for trading based on user sentiment. We have used twitter as our social media and analyze it to determine behavior of following stocks: Google, Yahoo, Apple, Microsoft and Goldman Sachs.
- We have chosen twitter as our social media because of its real time facet and importance of real time decision in trading. We have preferred to mine finance historical data from yahoo finance.
- The five stocks were determined based on opinion and mindshare on twitter, ensuring that we will have enough data model how their social data affect their stock.
- Last is to learn and investigate how machine learning techniques can be used to identify trends.

III. PROBLEM SOLUTION

- In above section, we have briefly discussed about our problem statement of the project. In this section we will expand our objective based on it, and derive proposed solution. There are mainly six goal of our project:

1. Collect social data from twitter and finance data from yahoo finance
2. Process and clean tweet corpus
3. Generate feature set and data set from processed tweet
4. Analyze and predict tweet sentiment using Naïve Bayes classification and Support vector machine techniques
5. Compute tweet mood and generate data set for stock prediction
6. Predict stock market trend from twitter sentiment statistics using Naïve Bayes support vector machine technique.

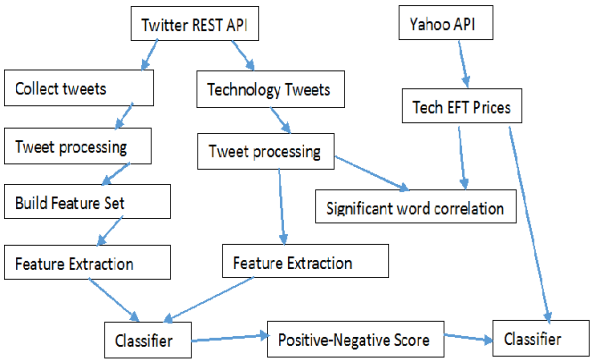


FIGURE I. MODEL

- For tweet collection, Twitter provides robust API. There are two possible ways to gather tweets: The streaming API or the search API. To overcame, the limitation of streaming API, we have used search API. The search API is REST API which allow users to request specific query of recent tweets.
- The search API allow more fine tuning queries filtering based on time, region, language etc.
- The request of JSON object contains the tweet and their metadata. It includes variety of information including username, time, location, retweets. We have focused on time and tweet text for further analysis purpose.
- An API requires the user have an API key authentication. After authenticating using key, we were able to access through python library called Tweepy.
- The text of each tweet contains too much extraneous words that do not consider to its sentiment. Tweets include URLs, tags to others and many other symbols that do not have any meaning. To accurately obtain tweet's sentiment we need to filter noise from its original state.

An example of tweet characters and formats can be seen in below table:

TABLE I. TWITTER EXAMPLE

Company Name	Tweet status/Text
$AAPL	@JeffMacke Stunning that they never did this. Or $AAPL, or $TWX- these guys all should have been trying to buy $NFLX.
$GOOG	Oracle, #Google fail to settle Android lawsuit before retrial. Read more: https://t.co/oYX31p9CDZ $GOOG
$YHOO	Investors hope Yahoos sale puts check in the mail $YHOO https://t.co/S6DviHMrfm https://t.co/4mUngBgHzt https://t.co/czUcEPLrjO
$MSFT	StockTwits: EARNINGS Mon - $IBM $PEP $NFLX Tue - $JNJ $GS $YHOO Wed - $KO $AXP Thu - $UA $GOOG
	$SBUX $MSFT $GM Frâ€¦ https://t.co/eaFaIcF9Wr
$GS	Oil Prices to Fall Asr Producers Fail to Reach Deal https://t.co/IvYFfYIeR7 via @WSJ Commodities head Jeff Currie $GS called it last week

- First step is to split the text by space, forming a list of individual words per text which is called as list of words. We will use each word in tweet as feature to train our classifier.
- Next step is to remove stop words from list of total words. I have used python's Natural Language Toolkit (NLT) library to remove stop words.
- Stop words contains articles, punctuation and some extra words which do not have any sentiment value. It contains stop word dictionary which check each word in list of tokenized words against dictionary. It word is stop word then it filters it out.
- Tweet contains extra symbols like "@", "#" and URLs. The word next to "@" symbol is always a username which does not add any sentiment value to text. Words following "#" are kept as they contain information about tweet. URLs are filtered out as they do not add any sentiment meaning to text. To accomplish all there process we use regex that matches these symbols.
- After gathering large tweet corpus, we have built and train classifier for tweet sentiment analysis. We examine mainly two classifiers: Naïve Bayes and Support Vector Machine.
- For each classifier we extract the same features from the tweets to classify on it.
- To build feature set, we process each tweet and extract meaningful feature and create feature matrix by unigram technique.
- A unigram is simply an N-gram size single word. For each word in a tweet, a unigram feature is created for classifier. For example, if positive tweet contains word "sad", a feature for classification would be whether or not a tweet contains the word "sad". If feature came from negative tweet, the classifier would more likely to classify other tweets containing the same word as negative.
- As explained the method above, the feature set grow larger and larger as dataset increases. After certain point, it is become difficult to handle larger dataset. In this case it is unnecessary to use every unigram as feature vector to train Naïve Bayes classifier and Support Vector machine. To avoid critical situation, we decided to use n mostly significant feature for training.
- We have determined the n best features from larger set using chi-squared test. It scores each word of training data and separate n best feature to classify model. For ease of implementation, we have used Python's Natural Language Toolkit (NLT) which allow us to calculate with conditional frequency and frequency of each feature.
- After calculating feature score, we rank the feature in order of score and choose top n feature for training and

classification. A feature reduction helps us to improve speed of classification.

- We find tweet sentiment value using Python's AFIINN library for training datasets labels. For example, the processed tweet is "Oracle, #Google fail to settle Android lawsuit before retrial"
- In this example, we have two important keywords with negative keyword which results whole tweet sentiment negative.

TABLE II. TWITTER SENTIMENT CALCULATION

Keyword	Fail	Lawsuit	Tweet
Sentiment	-2	-2	-4

- Generated data used as training dataset to classify two different model named as Naïve Bayes classifier and Support vector machine. We receive tweet sentiment labels as an output for test dataset. We will use this dataset for stock market prediction.
- We calculate total available stock tweets regarding each company and generate another dataset which contains positive, negative, neutral as well as total tweets of each day as a feature matrix. On other side we have taken stock market historical data for each day and calculate market up as well as down direction and took it as label for dataset.
- In case of stock market historical data, we have used Python's yahoo-finance library.
- After training our classifier, we decide to move on an application to look at correlation between tweet sentiment and stock market prices on each day scale. To do so, we have collected stock data as well as tweet data for same timeline. In addition, we focus on specific company stocks and gathered day by day data for each.
- We collect data using Yahoo finance API to gather data on 5 EFT's throughout trading day which was 8:30 to 3:00 PM central time. We used each gathered price and timestamp for stock prediction analysis.

IV. IMPLEMENTATION DETAILS

Instruction

The application is clubbed in single file. Hence user do not need to run different file to run different tasks. The program will execute in following manner.

1. Get twitter data and yahoo finance data for given company keyword
2. Collect tweet for given company stock and store it in text file
3. Generate csv file which separates tweet time and twit text in different column
4. Process tweet, remove stop words and create feature set for training data
5. Classify training data and test classified model on test data for tweet sentiment analysis
6. Calculate total number of tweets, positive tweet, negative tweet and neutral tweets
7. Generate dataset for stock prediction which contain feature set as tweet sentiment statistics and target set as stock up or down direction
8. Classify stock prediction dataset and predict upcoming stock direction

A. Design Issues

- During twitter based stock prediction analysis, I have found some of design and implementation issues at different levels.
- We do not have any ready dataset to classify our model. We have learned twitter API parameters and instruction to download live data to train classifier. It can be downloaded using Tweepy library but for that we do need authentication key and token.
- We refer twitter instruction and created an account on twitter developer site which provided secret key and token for twitter API response.
- Next challenge is to download vast amount of twitter data and create feature set for further analysis. It grows more and more as data increases. After certain point, it became difficult to handle large amount of data with larger dataset. To shrink our feature set, we have used chi-squared test. It shrinks our feature set and limits our feature set with n most important features.
- To correlate twitter and finance data set we need to download it in same timeline manner. It is challenging to find proper lag and collect data for same timeline. We have downloaded twitter and yahoo finance data for same timeline to sync for same timeline.
- The whole procedure includes download tweets, create feature set, design feature vector, design label, apply classifier, predict sentiment, correlate data for same timeline and predict stock market price. To process on live data, this whole chain takes a lot time. We have improved model in multiple ways to reduce processing time.
- As per rule, twitter do not provide historical data for finance tweets. We are restricted with limited dataset. We tried a lot but we are unable to fetch bunch of historical data.

B. Instruction to run

I have implemented above mentioned application with stated techniques in PyCharm IDE. Instruction to run an application:

- Load project in PyCharm IDE
- Run **DownloadTwitterData.py** file: It contains whole application chain such as download twitter data, download yahoo data, create feature set, design feature matrix, analyze and predict tweet sentiment using Naïve Bayes classification and Support vector machine
- Run **DownloadTwitterData.py** file: It contains basic code for stock prediction analysis. We have predicted stock market movement using Support vector machine model

- The application run on live dataset hence it must need an internet connectivity.
- In addition, it must need twitter authentication parameters to collect twitter data. Hence configuration file is required to run an application

V. ALGORITHM

Classification is an interesting problem in machine learning. In our case we need to construct classifier that is trained our "positive", "negative" or "neutral" labeled tweet corpus.

By using implemented classifier, we have labeled future tweets as either "positive", "negative" or "neutral" based on trained tweets features. In this project, we examine two common classifier used for text classification: Naïve Bayes Bernoulli and Support Vector Machines models.

In addition, we have implemented Naïve Bayes Classifier. We use it for stock market prediction. We have correlated stock tweets statics and stock prices and classify it for certain day. Train classifier will use to predict stock prices for specific company for next upcoming days.

A. Naïve Bayes Classifier

- A Naïve Bayes classifier is based on Bayes rule and simple form of Bayesian network. It is a simple probabilistic model relies on the assumption of feature independent in order to classify data.
- The algorithm assumes that each feature is independent of presence or absence of other feature in input data as this assumption is known as 'naïve'.
- Bayes classifier use Bayes theorem, which is

$$P(c|F) = \frac{P(F/c\ )P(c)}{P(F)}$$

P(c/F) = probability of feature F being in class c
P(F/c) = probability of generating feature F given by class c
P(c) = probability of occurrence of class c
P(F) = probability of feature F occurring

- In above context, we are looking for class c, so we find probable class given for given feature, F. Denominator does not depend on class so we treat it as a constant. Numerator depends on class so we focus to determine the value of P(F/c). For a class c_j, the features are conditionally independent of each other, hence

$$P\left(f1, f2 \ldots \frac{fn}{cj}\right) = \prod_i P(\frac{fi}{cj})$$

From these we classify tweet statistics with label c* with a maximum posterior decision taking the most probable label from all labels C.

$$c* = arg \max_{cj \in C} P(cj) \prod_i P(fi/cj)$$

The Naïve Bayes is very simple and its conditional independence assumptions are not realistic in real world. However, we have performed it multiple times and it gives better accuracy for stock prediction.

B. Naïve Bayes Bernoulli

- There are two different type of NB classifier named as Multinomial model and Bernoulli model. In this section, we have used Bernoulli model.
- It is equivalent to binary independence model which generates an indicator for each feature of tweet, either 1 indicating presence of feature in tweet or 0 indication the absence of feature in tweet. The Bernoulli model has the same time complexity as the multinomial model.
- In our scenario, we have extract feature set from twitter corpus. The extracted feature will be check in twitter and generates feature data for tweet sentiment analysis.
- If feature exist in tweet, then it is considering as 1 and if feature does not exist in tweet then it is considering as 0. The generated train data is used for classification and trained classify model will be used to predict tweet sentiment.

TRAINBERNOULLINB(C,T)
1. V ← EXTRACTFEATURE(T)
2. N ← COUNT TWEETS(T)
3. **FOR EACH** C ∈ C
4. **DO** N ← COUNTTWEETSINCLASS(T,C)
5. prior[c] ← Nc/N
6. **FOR EACH** T ∈ V
7. **DO** N_{CT} ← COUNTTWEETSINCLASSCONTAINSTERM(T,C, T)
8. CONDPROB[T][C] ← (N_{CT} +1)/ (N_C +2)
9. **RETURN** V, PRIOR, CONDPROB

APPLYBERNOULLINB(C,V, PRIOR, CONDPROB)
1. V_D ← EXTRACTTERMFROMTWEET(T)
2. **FOR EACH** C ∈ C
3. **DO** SCORE[C] ← LOG PRIOR[C]
4. **FOR EACH** T ∈ V
5. **DO** IF T ∈ V_D
6. **THEN** SCORE[C] += LOG CONDPROB[T][C]
7. **ELSE** SCORE[C] += LOG (1-CONDPROB[T][C])
9. **RETURN** $arg \max c \in C$ SCORE[C]

The Bernoulli model estimates P(t/c) as the fraction of class c that contains term t. When classifying a test tweets, the Bernoulli uses binary occurrence information, ignoring number of occurrences.

C. Multiclass support vector machine

Multiclass SVMs classify an input vector $x \in R^d$ into one of k classes using the following simple rule:

$$\hat{y} = \underset{m \in [k]}{\operatorname{argmax}} \boldsymbol{w}_m^{\mathrm{T}} \boldsymbol{x}.$$

Equation [1]

Each vector $\omega_m \in R^d$ can be thought as prototype representing m^{th} class and the inner product as the score of the m^{th} class with respect to x. Hence above equation choose the class for highest score. Given n training instances $x \in R^d$ and associates labels $y \in [k]$, the multiclass SVM formulation estimates $\omega_{1,\ldots}\ \omega_k$ by solving following problem.

$$\underset{\boldsymbol{w}_1,\ldots,\boldsymbol{w}_k}{\text{minimize}} \ \frac{1}{2}\sum_{m=1}^{k} \|\boldsymbol{w}_m\|^2 + C\sum_{i=1}^{n}\left[1 + \max_{m \neq y_i} \boldsymbol{w}_m^{\mathrm{T}}\boldsymbol{x}_i - \boldsymbol{w}_{y_i}^{\mathrm{T}}\boldsymbol{x}_i\right]$$

Equation [2]

where C>0 is regularization parameter and [u] = 0 if u >0 and u otherwise. Intuitively Equation (2) means that, for each training instance, we suffer no loss if the score of the correct class is larger than the score of "closest" class by least 1. In this paper, we assume that $\|x_i\| > 0$ since x_i with $\|x_i\| = 0$. The dual of equation is given by [3], [5]

$$\begin{aligned} \underset{\boldsymbol{\alpha}}{\text{minimize}} \quad & f(\boldsymbol{\alpha}) = \frac{1}{2}\sum_{m=1}^{k}\|\boldsymbol{w}_m(\boldsymbol{\alpha})\|^2 + \sum_{i=1}^{n}\sum_{m=1}^{k}\Delta_i^m\alpha_i^m \\ \text{subject to} \quad & \alpha_i^m \le C_i^m \quad \forall i \in [n] \quad \forall m \in [k] \\ & \sum_{m=1}^{k}\alpha_i^m = 0 \quad \forall i \in [n], \end{aligned}$$

Equation [3]

The primal-dual relationship is given by

$$\boldsymbol{w}_m(\boldsymbol{\alpha}) = \sum_{i=1}^{n}\alpha_i^m \boldsymbol{x}_i \quad \forall m \in [k].$$

The gradient of f plays an important role and is given by

$$g_i^m = \frac{\partial f}{\partial \alpha_i^m} = \boldsymbol{w}_m(\boldsymbol{\alpha})^{\mathrm{T}}\boldsymbol{x}_i + \Delta_i^m \quad \forall i \in [n]\ \forall m \in [k].$$

Equation [4]

Following [3], we have an optimal solution if and only if $v_i = 0\ \forall i \in [n]$, where

$$v_i = \max_{m \in [k]} g_i^m - \min_{m \in [k]:\alpha_i^m < C_i^m} g_i^m \quad \forall i \in [n].$$

Equation [5]

The larger v_i, the more $\alpha^{1,}.. \alpha^k$ violate the optimality conditions. In general, give tolerance parameter ∈, we can stop an optimization algorithm if $v_i < \in, \forall i \in [n]$.

DUAL COMPOSTION FOR MULTICLASS SVMs

The main idea of dual decomposition methods is to update at every iteration a small subset of dual variable, keeping all others fixed. Since the variables $\alpha^{1,}.. \alpha^k$ associated with x_i are tied by an equality constraint, a natural choice for decomposition methods is to update these variables as a block. Our goal is to find $\delta_I \in R^k$ such that the update $\alpha i \leftarrow \alpha i + \delta_I$ maximizes the decrease of the dual objective. Then restricted problem is

$$\begin{aligned} \underset{\boldsymbol{\delta}_i}{\text{minimize}} \quad & \hat{f}(\boldsymbol{\delta}_i) = \frac{\|\boldsymbol{x}_i\|^2}{2}\|\boldsymbol{\delta}_i\|^2 + \boldsymbol{g}_i^{\mathrm{T}}\boldsymbol{\delta}_i \\ \text{subject to} \quad & \boldsymbol{\delta}_i \le \boldsymbol{C}_i - \boldsymbol{\alpha}_i \\ & \boldsymbol{\delta}_i^{\mathrm{T}}\mathbf{1} = 0, \end{aligned}$$

Algorithm: Dual decomposition for multiclass SVMs

Input: {(xi, yi)} n i=1, C > 0, > 0
Initialize α ← 0 and wm ← 0 ∀m ∈ [k]
Shuffle data
repeat
  vmax ← −∞
  for i ∈ [n] **do**
    Compute violation vi by Eq. (5)
    vmax ← max(vmax, vi)
    if vi > 0 **then**
      Find δi by solving Eq. (6)
      αi ← αi + δi and wm ← wm + δ m i
    xi ∀m ∈ [k]
    end if
  end for
until vmax ≤ ∈
Output: α and w1, . . . , wk

VI. Result and Discussion

We have implemented application in number of phases. We would like to describe our results and discuss about results in following manner.

In first phase of task, we have collected stock tweets using twitter API. It responses tweets in JSON format. It is mandatory to keep configuration file as an authentication to download tweet data live.

The indispensable configuration parameters are:

```
{
  "consumer_key":"",
  "consumer_secret":"",
  "access_token":"",
  "access_token_secret":""
}
```

After fetching twitter corpus, we have store it in text file. In parallel we process our tweets for further analysis purpose. To process tweet, we have follow some basic steps:

1. Tokenized tweets

2. Remove extra keyword from tweets
3. Remove stop words from tweet
4. Replace URLs and meaningful notation with meaning full keywords

To prepare training set we have used AFINN library. It computes tweet keywords positive and negative scores and result tweet sentiment with positive, negative or neutral score. We have map tweet score with positive, negative or neutral keyword. If total tweet score is greater than 0 then it considers as positive keyword. If total tweet score is less than 0 then it considers as negative. If total tweet score is 0 then it is considered as neutral.

For example, the processed tweet is "Oracle, #Google fail to settle Android lawsuit before retrial". The total sentiment score of this tweet -4. Hence it is considered as negative tweet.

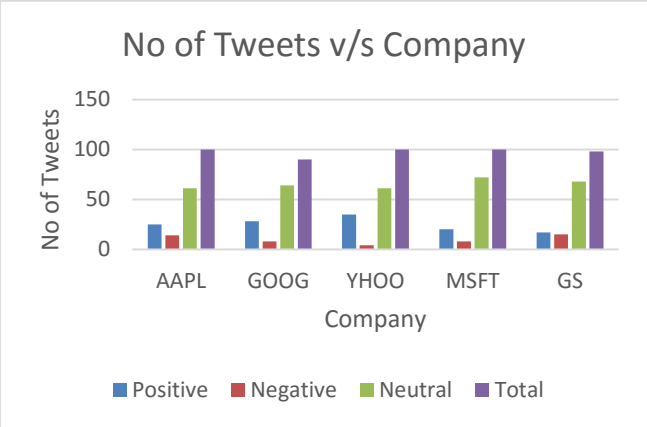
The generated tweet corpus csv file contains tweet date, tweet text, tweet sentiment keyword and total tweet score. We have prepared dataset using generated tweet corpus csv file.

Total available tweets for five different companies (15th April):

TABLE III. COMPANY TWEET COUNTS

COMPANY	POSITIVE	NEGATIVE	NEUTRAL	TOTAL
AAPL	25	14	61	100
GOOG	28	8	64	90
YHOO	35	4	61	100
MSFT	20	8	72	100
GS	17	15	68	98

FIGURE II. NUMBER OF TWEETS VS COMPANY

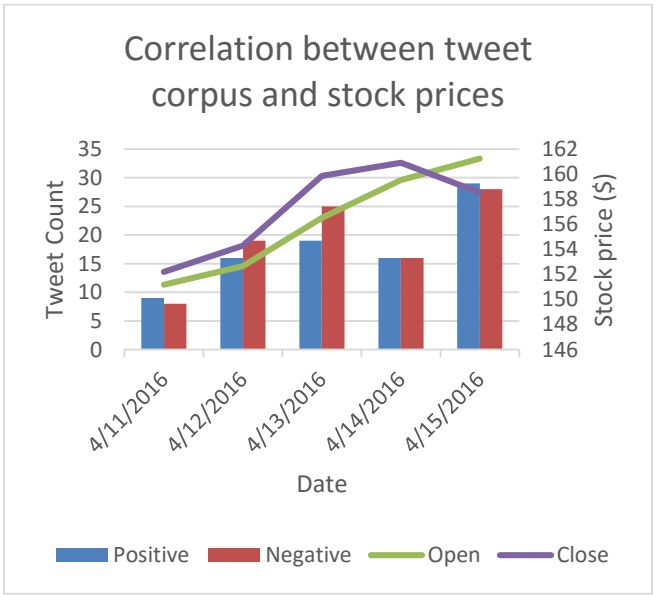


Approximately, we can fetch around 90-10 tweet of each stock company for a single day. The above graph represents relation between company stock name and total number of tweets (positive, negative, neutral and total)

Using Python's csv library, we have read tweet corpus data which will converted into model input dataset. Tweet corpus file provide data in four different columns. We read tweet text from csv file and create feature set by choosing important keyword from whole dataset. The feature set will compare with each tweet and found present feature in tweet. If feature is presented, then it will consider as feature 1. If feature is absent, then it will considered as feature 0.

Using stated, approach we have built feature matrix. It considers feature matrix in form of 0 and 1 and target matrix in form of positive, negative and neutral. For example, tweet is "Oracle, #Google fail to settle Android lawsuit before retrial". The important keywords are fail and lawsuit so they are considered as 1 and other keywords are considered as 0.
We have also captured tweet statistics and stock prices for last eight days. For more accuracy we do not process weekend dataset.
For Apple company stock prediction, we have collected tweet corpus using $APPL keyword and one other hand we have collected yahoo finance data for APPL stock prices. The general correlation between apple tweet corpus and stock price is displayed in following chart.
The relation between APPL tweet corpus and stock market price is as below:

FIGURE III. RELATION BETWEEN STOCK TWEETS AND MARKET PRICE (AAPL)



The relation between GS (Goldman Sache) tweet corpus and stock prices are as below:

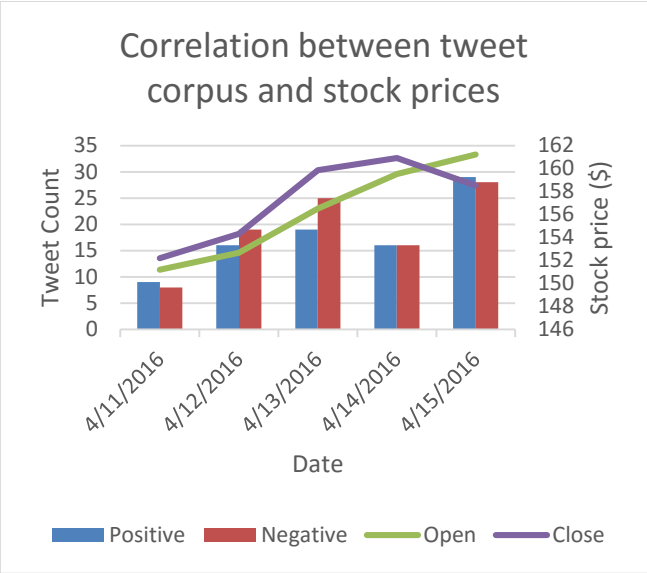


FIGURE IV. RELATION BETWEEN STOCK TWEETS AND MARKET PRICE(GS)

The above graphs represent the relation between tweet corpus and stock prices. We can observe how stock prices are correlated with tweet corpus. These relation inspires to predict the stock prices based on corresponding positive and negative tweet corpus.

Our next step is to predict tweet sentiment using Naïve Bayes classifier and Support vector machine model. We have considered three different classes named as positive (0), negative (1) and neutral (2).

The model accuracy and error statistics are as below:

Naïve Bayes
Accuracy: 0.8039 (80.39 %)

Error/Class	**Class 0**	**Class 1**	**Class2**
Precision	0.4615	0.4500	0.9710
Recall	0.75	1.0	0.7882
F-Measure	0.5714	0.6207	0.8701

Support Vector Machine
Accuracy: 0.8431 (84.31 %)

Error/Class	**Class 0**	**Class 1**	**Class2**
Precision	0.6923	0.6500	0.9275
Recall	0.5625	1.0	0.8767
F-Measure	0.6206	0.7878	0.9014

The above results conclude that the SVM model accuracy is higher than Naïve Bayes in all cases. After analyzing model on various datasets, we observed that the accuracy is varied from 75-85%. The variation happens depends on live datasets. The accuracy is increased if we receive less features. On opposite side, an accuracy is decreased as we receive large dataset as well as large feature matrix.

In addition, we have compared our model accuracy with inbuilt scikit learn and NLTK libraries for each evaluation. An output of libraries is 3-4% better than implemented models.

The output of tweet sentiment provides tweet statistics. The tweet statistics are considered as features for stock prediction. Tweet statistics and stock price movement of each day are use as stock prediction dataset.

For stock prediction we have used classifier named as Support vector machines. Here, we are not provided more than one-week old data. Due to lack of dataset we could not justify our model at best. Though we have designed one dummy database and classify our mode for it.

The accuracy and errors statistics of implemented model is as below:

Accuracy: 0.7239 (72.39 %)

The 72% model accuracy means the prediction is around 70-75%. With this more complicate strategy, we seek to explore knowledge of the direction in which the market will change. This allows us to take a decision whether it should be purchased or not.

We have compared calculated accuracy with prebuilt scikit-learn and NLTK libraries. The accuracy difference is around 4-5 %.

Hence we can conclude that the model is accurate more an over 70%. That mean the probability of market movement is predicted around 70%.

VII. CONCLUSION AND FUTURE WORK

After analyzing tweet sentiment, we have predicted stock market movement with 75% accuracy. The implemented model works much better for tweet sentiment analysis. Performing whole task, I come to conclude that we should work more on twitter and yahoo dataset which help us to improve our model.

We faced restriction to download historical data for twitter corpus. Hence we need to get official permission with twitter to retrieve more data for better analysis.

VIII. REFERENCES

[1] Machine learning in prediction of stock market indicators based on historical data and data from Twitter sentiment analysis.
http://ieeexplore.ieee.org.ezproxy.gl.iit.edu/stamp/stamp.jsp?tp=&arnumber=6753954&tag=1

[2] Stock Prediction Using Twitter Sentiment Analysis
http://cs229.stanford.edu/proj2011/GoelMittal-StockMarketPredictionUsingTwitterSentimentAnalysis.pdf

[3] Large-scale Multiclass Support Vector Machine Training via Euclidean Projection onto the Simplex
http://www.mblondel.org/publications/mblondel-icpr2014.pdf

[4] Sentiment Analysis on Twitter with Stock Price and Significant Keyword Correlation
http://apps.cs.utexas.edu/tech_reports/reports/tr/TR-2124.pdf

[5] https://github.com/ravikiranj/twitter-sentiment-analyzer

[6] https://gist.github.com/mblondel/97cffbea574a5890f0d7

[7] https://gist.github.com/mblondel/97cffbea574a5890f0d7