SENTIMENT ANALYSIS
IN SOCIAL MEDIA MONITORING
MANCHESTER UNITED
delivered solid results in a challenging market

Sentiment analysis to predict stock price

Sentiment analysis on twitter dataset for #ManUtd to predict stock price

Vignesh Hariharan

1 TABLE OF CONTENTS

2 INTRODUCTION

2.1.1 SENTIMENT ANALAYSIS OF STOCK

Stock market changes are extremely unpredictable and factors such as market sentiment, government policies, company announcements related to stock are highly uncertain. Technological advancement helps the researchers and market professionals to develop mathematical models to secure their profit returns and to check risk effectively. The main option in social platform predominantly is twitter which is mainly suitable to express opinions and reviews.

Twitter is an attractive platform for the user community to discuss company health, company announcements, major news, and government policies etc. These information's helps to predict sentimental stock analysis. Twitter helps to gather information about the organization and assist to price predicts of its security

The main research is predicting the stock market through Efficient Market Hypothesis (EMH) and the patterns which they follow and changes around the world. Studies and research helps to know the mutual relationship between the market sentiment and monthly or weekly returns and the main challenge is collecting the models from a large set of data and predicting the stock market.

Machine learning deals with a training data set against which new incoming data is compared to predict the result, process is known as supervised learning another category of machine learning algorithms is regression. There are two main algorithms used in sentimental analysis of the stock market namely Naive Bayes, and Support Vector Machines (SVM).

2.1.2 STOCK PREDICTION USING TWITTER SENTIMENT ANALYSIS

With the collected data from twitter I going to find correlation between public sentiment and stock price. Analyzing twitter data helps us identify public mood which can be used to find correlation with stock market.

Null Hypothesis Testing: There is no relationship between twitter sentiment and stock market for Manchester United.

Sentiment analysis is an important part for the solution since the output of this module is helpful to learn predictive model.

2.2 COLLECTION OF DATA

- Data is collected for #ManUtd, from twitter. I captured around 3200 tweets.
- Data collected from google finance for MANU in NYSE

2.2.1 Twitter API to collect the tweets

The Twitter platform offers access to that corpus of data, via our APIs. Each API represents a facet of Twitter and allows developers to build upon and extend their applications in new and creative ways. It's important to note that the Twitter APIs are constantly evolving and developing on the Twitter Platform. For data protection twitter has restriction on data collection. Data can be collected only for 9 previous days from the current date.

2.2.2 Google finance service to collect stock market data

Google finance was launched in the year 2006, Using google finance I can collect data for any stock in world. The stock price is updated up to last hour. There is missing data sometime but for us it was stable, and I had no null values. There are many services like google finance, but I found that this is the most stable and easy to use API in python.

2.2.3 Three dataset I will be working with are:

- Twitter Data set
- Financial Data set
- Training data from sentiment 140

3 MANCHESTER UNITED FC?

Manchester is a professional football club. The short name of Manchester United is MUFC. Manchester United football club was established in the year 1902. It is based on the old Trafford, Greater Manchester, England. The football club contends in English Premier club, FA Cup, UEFA Champions group, UEFA Europa League and FIFA World Cup. The RED DEVILS is the moniker of the club. MUFC was set is first stride in London stock trade 1991 and in 2012 the club was recorded in New York Stock Exchange (NYSE) in the name of Manchester United PLC (NYSE: MANU). MUFC has numerous adherents over the online networking like twitter, Facebook, Instagram and so on.

3.1 Why Manchester United FC

One of the most famous club in the world and our favorite, I were looking for an opportunity work on something which I can relate to. With this project I intend to understand how football sentiments can play a role in deciding the stock price for the club. MUFC Real time analysis has carried out and exceptional examination in to how observation via web-based networking media are impacted by occasions in a football club. Manchester United is right now the main Premier

League club to be recorded on any real stock trade and their proprietors have indicated benefit boosting inclinations. Manchester United is the main traded on publicly market Premier League club, there have been several English groups exchanged on stock trades. Joined twitter in year 2012 but there were tweets from years before also for it. But due to twitter restriction I cannot go beyond 9 days for current date for data collection.

4 MUFC TWITTER FOLLOWERS

Manchester United made it to having an incredible 17,855,308 Twitter devotees and to being positioned 110 for number of supporters among all Twitter clients. MUFC is by all accounts on its approach to Twitter-verse control! The plot thickens while considering Manchester United's adherent to-following proportion, which is 171685.65. As you can presumably accumulate, this metric place this record serenely inside superstar status. It appears like Manchester United is heard on Twitter, with a crowd of people mindfulness score of 29%, which comes from being followed on 19,722 Twitter records and standardized to their 17,855,308 adherents.

4.1 APPLICATION OF SENTIMENTAL ANALYSIS

The applications for sentiment analysis are perpetual. More I're seeing it utilized as a part of social media monitoring to track client audits, study reactions, contenders, and so forth. In any case, it is likewise viable for use in business investigation and circumstances in which message should be examined.

Sentiment analysis is sought after because of its productivity. A great many content archives can be prepared for opinion (and different highlights including named elements, subjects, topics, and so on.) in seconds, contrasted with the hours it would take a group of individuals to physically entire. Since it is so proficient numerous organizations are adopting text and sentiment analysis and consolidating it into their procedures.

5 NATURAL LANGUAGE PROGRAMMING:

Natural-language processing (NLP) is the interaction between human languages and computer is a field of artificial intelligence. It is concerned with how to write a program to dig into natural language data. There are several main challenges with NLP which are natural language understanding, speech recognition, and natural language generation.

When I applying NLP algorithms, it looks like teaching a language to a child. It means that some of the main tasks like what is words, what is sentences, and what is the correct grammatical format of a sentence are like human natural language. There are many challenges for computer to deal with the tasks of identifying the concepts of natural language such as tokenization, chunking, part of speech tagging, parsing, machine translation, speech recognition [NLTK Essential]. Here I will focus on practical aspects of NLP.

5.1 Why learn NLP?

NLP is considered as the rarest skills that is on demand in the industry. By advent of big data, a new challenge comes out that is semi or unstructured data. As I know, in weblogs like twitter, in social media networking website like Facebook, in messenger apps or through email platforms, I generate tones of data every moment. Every company nowadays collects all types of data to find out deep insights and pattern by digging into data. To process all these unstructured data source, I need people who understand NLP.

5.2 Application of NLP

Now, I living in age of information and data is considered as the new oil, so I cannot imagine a day of our life without Google, I use Siri for most basic stuff. I use spam filters for filtering spam emails. I need spell checker on our Word document. There are many examples of real world NLP applications around us.

There are many awesome examples of the application of NLP that I can use that I may not know that they are created on NLP:

Spell correction (MS Word/ any other editor)

Search engines (Google, Bing, Yahoo, wolfram alpha)

Speech engines (Siri, Google Voice)

Spam classifiers (All e-mail services)

News feeds (Google, Yahoo!, and so on)

Machine translation (Google Translate, and so on)

IBM Watson

6 LANGUAGE AND TOOLS TO BUILD NLP

To create the aforementioned application, I need very particular skills with a quite great knowledge of language tools to understand the language properly. To create some of above applications and other basic preprocessing for NLP, I can find many tools which are either open sourced or copyrighted. Here is a small list of available NLP tools:

- GATE
- Mallet
- Open NLP

- UIMA
- Stanford toolkit
- Genism
- Natural Language Tool Kit (NLTK)

Most them have the same structure with similar functionality because they are written by JAVA, some of them really powerful and robust like NLTK not only because of its robustness but also because of ease of use. NLTK is also a very good learning kit because the learning curve of Python (on which NLTK is written) is very fast. NLTK has incorporated most of the NLP tasks, it's very elegant and easy to work with. For all these reasons, NLTK has become one of the most popular libraries in the NLP community [NLTK Essentials].

7 TEXT WRANGLING AND CLEANSING

I will go over pre-processing steps like tokenization, stemming, lemmatization, and stop word removal in more detail. I will explore all the tools in NLTK for text wrangling. I will talk about all the pre-processing steps used in modern NLP applications, the different ways to achieve some of these tasks, as well as the general do's and don'ts. The idea is to give you enough information about these tools so that you can decide what kind of pre-processing tool you need for your application.

7.1 Text Pre-processing Framework

The text wrangling process should be engaged with three main steps:

- tokenization
- normalization
- substitution

A very clear pre-processing flow for some frequently used document type is shown in **Error! Reference source not found.**. For example, in the case of a csv file, Python's csv module is the most robust way of handling the csv file. It allows you to play with different splitters, different quote characters, and so on. The other most commonly used files are json.

I will then follow up with a practical implementation of these steps next time, in order to see how they would be carried out in the Python ecosystem (Figure 2).

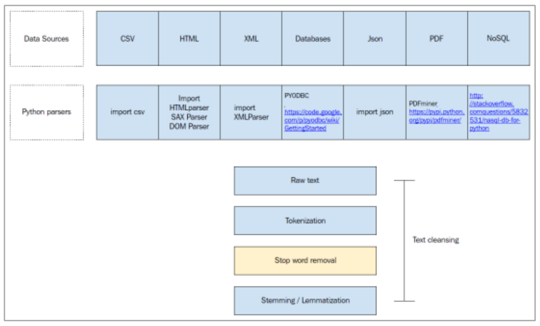

Data Sources
CSV
HTML
XML
Databases
Json
PDF
NoSQL
Python parsers
import csv
Import HTMLparser SAX Parser DOM Parser
import XMLParser
PYODBC https://code.google.com/p/pyodbc/wiki/GettingStarted
import json
PDFminer https://pypi.python.org/pypi/pdfminer/
http://stackoverflow.com/questions/5832531/nosql-db-for-python
Raw text
Tokenization
Stop word removal
Stemming / Lemmatization
Text cleansing

Figure 1: cleaning process and type of data sources

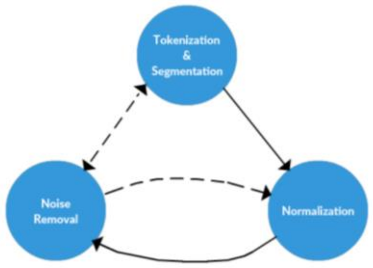


Figure 2 The Text pre-processing framework

7.1.1 Tokenization

Tokenization is a step which splits longer strings of text into smaller pieces, or tokens. Larger chunks of text can be tokenized into sentences, sentences can be tokenized into words, etc. This may sound like an easy process, but it is not.

7.1.2 Normalization

Normalization means a series of related process which make the text almost uniform like changing uppercase to lowercase, converting numbers into the word format, transforming words to their stems and should be considered as three different tasks (1) stemming, (2) lemmatization, and (3) everything else.

7.1.2.1 Stemming

Stemming is the process of eliminating affixes (suffixed, prefixes, infixes, circumfixes) from a word in order to obtain a word stem.

running ⟶ run

7.1.2.2 Lemmatization

Lemmatization is related to stemming, differing in that lemmatization is able to capture canonical forms based on a word's lemma.

For example, stemming the word "better" would fail to return its citation form (another word for lemma); however, lemmatization would result in the following:

better → good

It should be easy to see why the implementation of a stemmer would be the less difficult feat of the two.

7.1.2.3 Everything else

Although the stemming and lemmatization are the most important parts of the normalization there are numerous other steps that can make the text clean disparate signs in the text as listed below

1. set all characters to lowercase
2. remove numbers and convert to word format
3. remove punctuation (generally part of tokenization, but still worth keeping in mind at this stage, even as confirmation)
4. strip white space (also generally part of tokenization)
5. remove default stop words (like "I" "The", "me", etc)

7.1.3 Noise Removal

Noise removal continues the substitution tasks of the framework. For example, it is engaged with tasks as below:

1. remove text file headers, footers
2. remove HTML, XML, etc. mark-up and metadata

8 TEXT CLASSIFICATION

Text classification is a great use case of NLP, even NLP has own classifier algorithms, but I will exploit scikit-learn which has more reliable and more memory efficient algorithms for text mining (Hardeniya, 2015).

8.1 Machine learning

I all know that there are two types of machine learning approaches, supervised learning and unsupervised learning:

8.1.1 **Supervised learning**

In order to predict the future test data or dependent variable in supervised learning I have to use pre-tagged training data based on the following categories:

8.1.1.1 Classification

As it is obvious, in classification I need to predict if a test data belongs to one of the classes in dataset. I have binary or multiclass classification. For text mining I can use different types of classifiers like Naïve Bayes, SVM, Maximum Entropy, and etc. Here I will use Naïve Bayes classifier which has three types (1) GaussianNB: implements the Gaussian Naive Bayes algorithm for classification, (2) MultinomialNB implements the naive Bayes algorithm for multinomially distributed data, and is one of the two classic naive Bayes variants used in text classification, and (3) BernoulliNB implements the naive Bayes training and classification algorithms for data that is distributed according to multivariate Bernoulli distributions. This is used when I need to predict whether a test sample belongs to one of the classes (Pedregosa, 2011).

8.1.1.2 Regression

There are several types of regression in supervised learning. Based on the types of dependent variables I have to use the right type of it. Foe example for continuous variable, linear regression is frequently used, but for binary variable, logistic variable is implemented.

8.1.2 Unsupervised learning

When I don't have any labelled data and I still need to predict the class label, this kind of learning is called unsupervised learning. When I need to group items based on similarity between items, this is called a clustering problem. While if I need to represent high dimensional data in lower dimensions, this is more of a dimensionality reduction problem.

8.1.3 Semi-supervised learning

There is another technique in machine learning to build prediction model when the number of tagged data is really small in comparison with the unlabelled data. As the name suggests, it's more of a middle ground for supervised and unsupervised learning(Hardeniya, 2015).

8.2 Sklearn.feature_extraction

To extract features in a dataset includes texts, images, and etc. scikit-learn library has a specific module namely the **sklearn.feature_extraction** and it can be used for datasets consisting of formats such as text and image. This module has text feature extraction to transform text to numerical features useable for machine learning algorithms like CountVectorizer implements both tokenization and occurrence counting in a single class, in a large text corpus, some words will be very present (e.g. “the”, “a”, “is” in English) hence carrying very little meaningful information about the actual contents of the document. If I were to feed the direct count data directly to a classifier those very frequent terms would shadow the frequencies of rarer yet more interesting terms. In order to re-weight the count features into floating point values suitable for usage by a classifier it is very common to use the tf–idf transform (TfidfTransformer). Tf means **term-frequency** while idf means **inverse document-frequency** (Pedregosa, 2011)

9 IMPLEMENTATION

This project is mainly divided in to 5 major parts namely

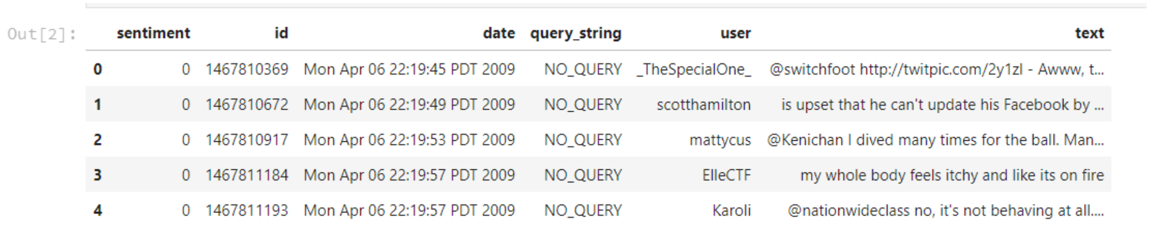
1.) Creating / Training the model
2.) Validating model
3.) Collecting twitter data and applying sentimental analysis
4.) Collecting stock market data
5.) Comparing stock values with sentiment for the day.

The broader idea behind the implementation is to create a model which can predict the sentiments in the text. NLTK, Pandas, Sklearn, Matplotlib and Tweepy packages are extensively used.

9.1 Creating / Training a model

9.1.1 Training data overview

The data used for training the model is available here. It can be downloaded in .csv format. I loaded data in data frame and added header to the dataset. Loaded dataset looks as follow.



Out[2]:

	sentiment	id	date	query_string	user	text
0	0	1467810369	Mon Apr 06 22:19:45 PDT 2009	NO_QUERY	_TheSpecialOne_	@switchfoot http://twitpic.com/2y1zl - Awww, t...
1	0	1467810672	Mon Apr 06 22:19:49 PDT 2009	NO_QUERY	scotthamilton	is upset that he can't update his Facebook by ...
2	0	1467810917	Mon Apr 06 22:19:53 PDT 2009	NO_QUERY	mattycus	@Kenichan I dived many times for the ball. Man...
3	0	1467811184	Mon Apr 06 22:19:57 PDT 2009	NO_QUERY	ElleCTF	my whole body feels itchy and like its on fire
4	0	1467811193	Mon Apr 06 22:19:57 PDT 2009	NO_QUERY	Karoli	@nationwideclass no, it's not behaving at all....

Figure 3 : Training data head

Dataset is designed to have 2 emotions, for our convenience I changed 0 to 0 (negative emotion) and 4 to 1 (positive emotion). To get better overview and understand how data is distributed between two sentiments I plotted it.

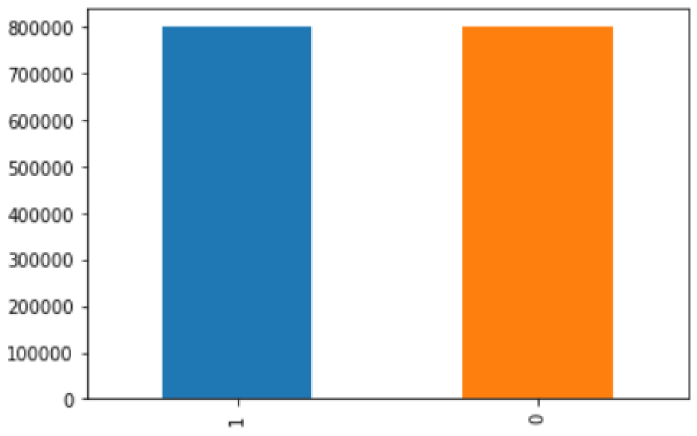


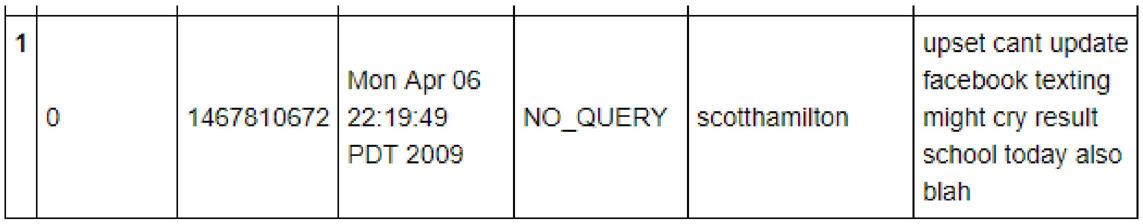
Figure 4 : Training Data distribution

I can see from Figure 4 , data is evenly distributed between both positive and negative sentiment.

9.1.2 Pre-processing training data

In Figure 3 , if I closely observe column ‘text’ I can see lot of stop words like “is, I, for “etc., unwanted characters like ‘@’, and URL. In pre-processing our intension is to remove all the irrelevant information and keep information which is useful for training a model. I also don’t need columns ‘id, query_string, and user’.

For pre-processing text data, I used regular expression in combination with lambda expression to get rid of unwanted text from ‘text’ column and removed unwanted columns using standard drop command of python pandas. I also used functions available in NLTK library to further clean this data.



1	0	1467810672	Mon Apr 06 22:19:49 PDT 2009	NO_QUERY	scotthamilton	upset cant update facebook texting might cry result school today also blah

Figure 5 : Data after removal of stop words, unwanted characters and URL

Figure 5 is one row from the processed dataset. It can be observed that the text has no stop words, no punctuation, no unwanted characters. It is important to remove them as these are unwanted noise and can impact performance of the mode.

Cleaned text data is plotted (Figure 6) in word cloud to understand if data is a good fit for training our model of text analysis or not.

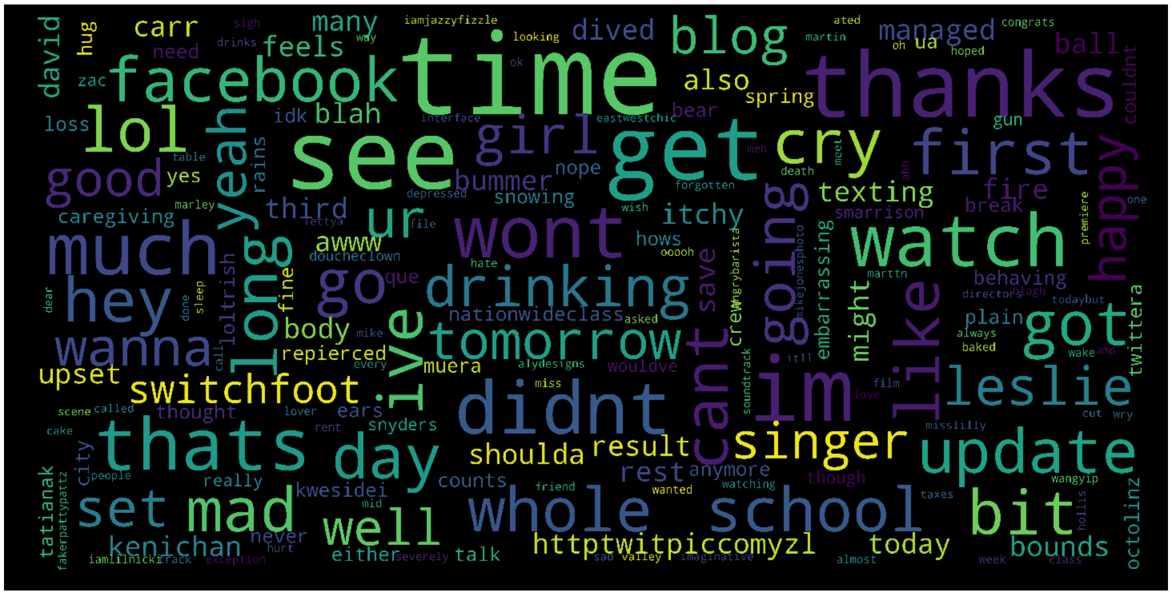


Figure 6 : Word cloud training data

From Figure 6 I see words like "Time thanks, watch" are having high frequency compared to others like school, girl etc. Due to limited processing power of our laptops I could not implement all the advanced techniques stemming and lemmatisation. But these steps are part of pre-processing of our test data which I collected from twitter.

To further our knowledge on frequency of words I can look at Figure 7. which shows top 30 frequent words.

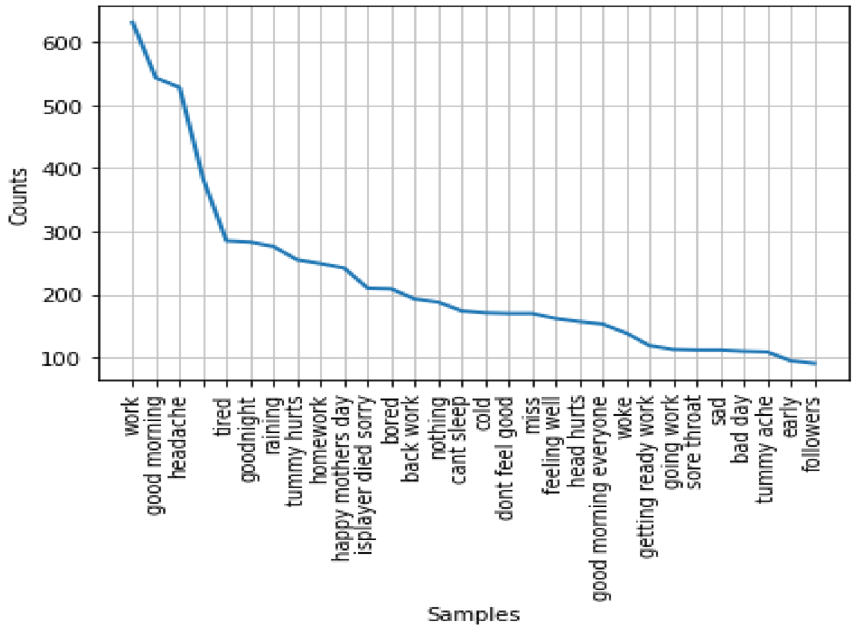


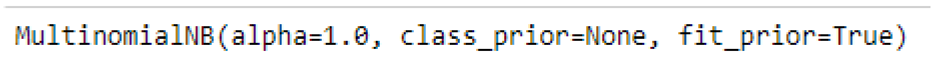
Figure 7 : Word frequency plot

Now the text is ready to train the model, so I go to next step which is training the model but before doing this, I will do stratified splitting of data in to train and validation dataset. Train dataset will be used for training the model and validation dataset will be used to check the accuracy of the model.

For this I will use method available in sklearn. For our project I have divided dataset in 4:1. i.e. 80% for training and 20% for validation.

9.1.3 Training the model

I are using Multinomial naïve bayes model, which is explained in the introduction. For every package data is required in specific format, in our case I using sklearn package and data required for MultinomialNB should be in vectorized form. I have used counter vectorizer to vectorize the data which is further transformed using tf-Idf format to reflect statistical importance of word in document. And finally, I train our model. It can take some time depending on amount of data. In our case it took around 5-10 minutes for the complete process of vectorization, tf-idf conversion and model training. I are using following parameters for MultinomialNB.



```
MultinomialNB(alpha=1.0, class_prior=None, fit_prior=True)
```

Figure 8 : MultinomialNB parameters

9.2 Validating model

Once the model is trained I can verify how it performed using the score. Score is simply the accuracy of prediction. For our mode I were able to achieve 76.38% accuracy. This is a very good accuracy considering I created our own technique and model. It can be further improved by adding few more memory hungry text cleaning techniques which big companies employ. Due to limited resources I must make peace with this score.

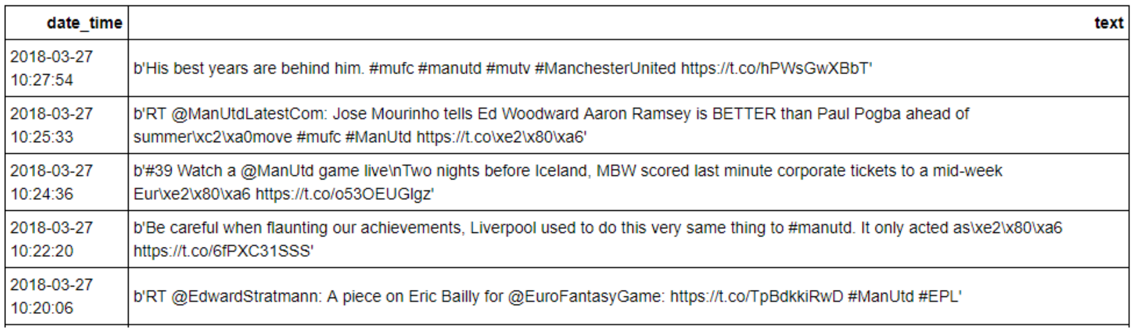
Now our model is ready for actual dataset. So, I go to next step which is collecting data from twitter.

9.3 Collecting twitter data and applying sentimental analysis

9.3.1 Collecting data

Collecting data from twitter is a straightforward task but require knowledge of python or R. For collecting data first step is to create a developer account on twitter from where I get consumer_key, Consumer_secret_key, Access_token and access_token_secret. These keys are passed as parameter in tweepy package's authentication function. After authentication I can use tweepy cursor (function) to get tweets based on hashtags. For our case I using hastag as "#ManUtd".

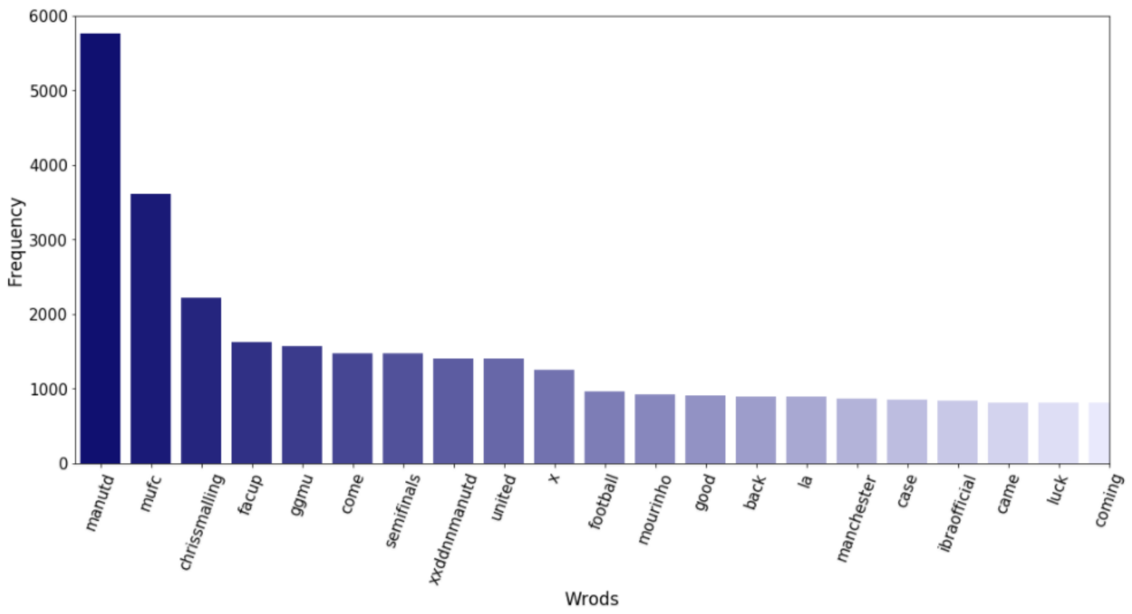
Note: There is a restriction of 9 days for which I can get twitter data from current date.



date_time	text
2018-03-27 10:27:54	b'His best years are behind him. #mufc #manutd #mutv #ManchesterUnited https://t.co/hPWsGwXBbT'
2018-03-27 10:25:33	b'RT @ManUtdLatestCom: Jose Mourinho tells Ed Woodward Aaron Ramsey is BETTER than Paul Pogba ahead of summer\xc2\xa0move #mufc #ManUtd https://t.co\xe2\x80\xa6'
2018-03-27 10:24:36	b'#39 Watch a @ManUtd game live\nTwo nights before Iceland, MBW scored last minute corporate tickets to a mid-week Eur\xe2\x80\xa6 https://t.co/o53OEUGlgz'
2018-03-27 10:22:20	b'Be careful when flaunting our achievements, Liverpool used to do this very same thing to #manutd. It only acted as\xe2\x80\xa6 https://t.co/6fPXC31SSS'
2018-03-27 10:20:06	b'RT @EdwardStratmann: A piece on Eric Bailly for @EuroFantasyGame: https://t.co/TpBdkkiRwD #ManUtd #EPL'

Figure 9 : Data from twitter

Data is converted into Pandas data frame and processed with all steps mentioned in section 9.1.2. I will be exploring twitter data to analyse if our text is cleaned or not and to understand what are most frequent words before finding sentiments in data.



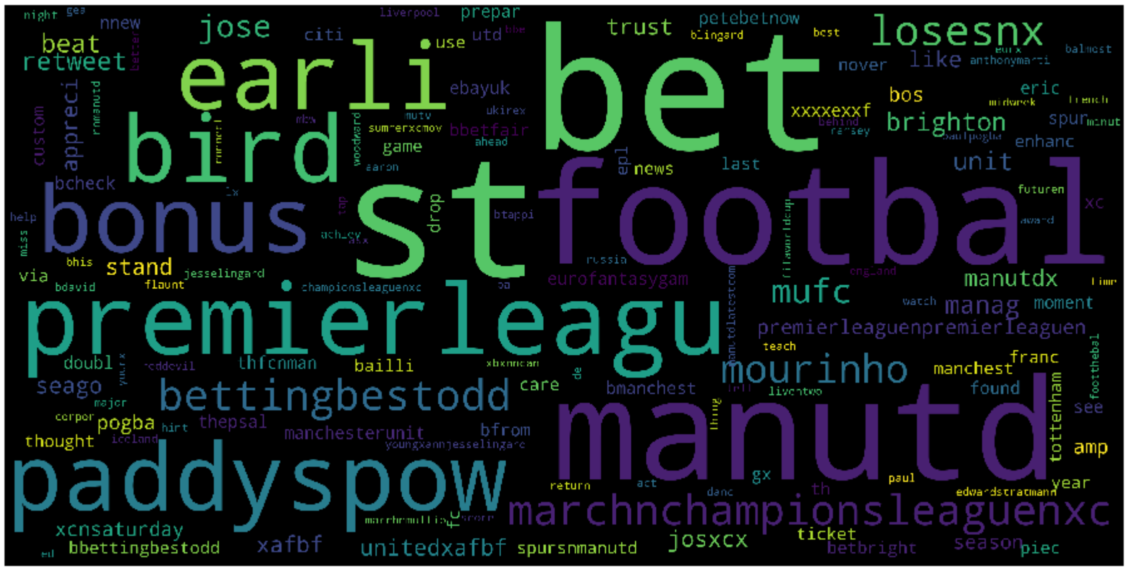


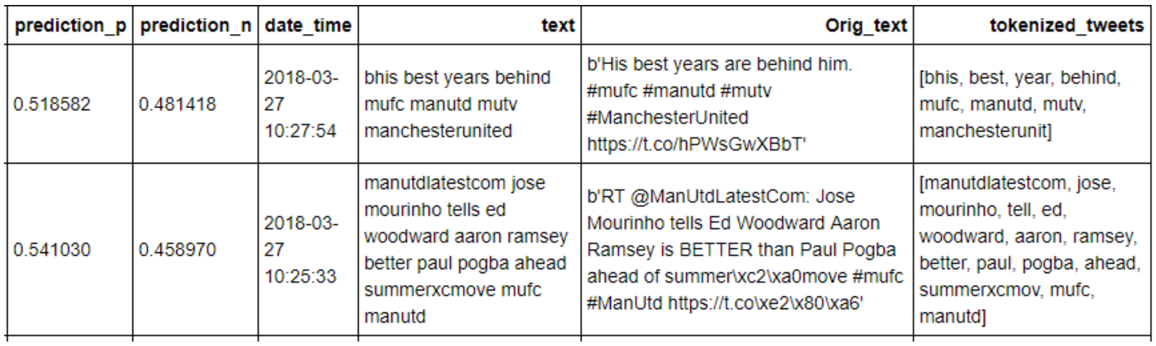
Figure 10 : Most Frequent words

From the Figure 10 I can see manutd and mufc are most frequent words which make sense since every tweet should be having # manutd and sometime #mufc. There are other words which I can see words like ‘bet’ ,’football’ etc., but to analyze the frequency of words bar graph is best option. Data is taken for the week when Manchester united was not performing well, it lost 2 matches that is why I see words like good and luck trending.

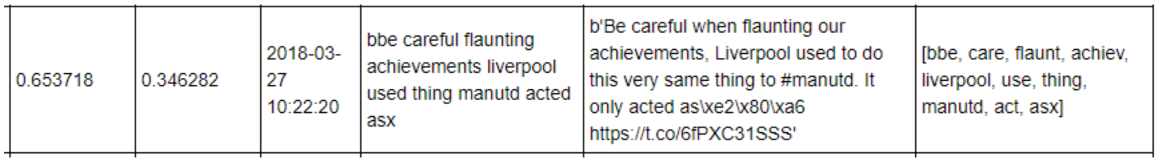
Now I sure that our data is clean and relevant. The next logical step is to predict the sentiments in the text.

9.3.2 Finding sentiments

Pre-processed data is given to model which finds the sentiments (positive or negative). Following is the screenshot of sentiments based on text.



prediction_p	prediction_n	date_time	text	Orig_text	tokenized_tweets
0.518582	0.481418	2018-03-27 10:27:54	bhis best years behind mufc manutd mutv manchesterunited	b'His best years are behind him. #mufc #manutd #mutv #ManchesterUnited https://t.co/hPWsGwXBbT'	[bhis, best, year, behind, mufc, manutd, mutv, manchesterunit]
0.541030	0.458970	2018-03-27 10:25:33	manutdlatestcom jose mourinho tells ed woodward aaron ramsey better paul pogba ahead summerxcmove mufc manutd	b'RT @ManUtdLatestCom: Jose Mourinho tells Ed Woodward Aaron Ramsey is BETTER than Paul Pogba ahead of summer\xc2\xa0move #mufc #ManUtd https://t.co\xe2\x80\xa6'	[manutdlatestcom, jose, mourinho, tell, ed, woodward, aaron, ramsey, better, paul, pogba, ahead, summerxcmov, mufc, manutd]



0.653718	0.346282	2018-03-27 10:22:20	bbe careful flaunting achievements liverpool used thing manutd acted asx	b'Be careful when flaunting our achievements, Liverpool used to do this very same thing to #manutd. It only acted as\xe2\x80\xa6 https://t.co/6fPXC31SSS'	[bbe, care, flaunt, achiev, liverpool, use, thing, manutd, act, asx]

Figure 11 : Sentiment prediction

Prediction_p and prediction_n means probability of positive and negative sentiment. The most challenging problem with twitter text is finding sentiments since most of the tweets are neutral in nature. It can be seen in Figure 11 where first two tweets are neutral in nature and thirds tweet is slightly more positive.

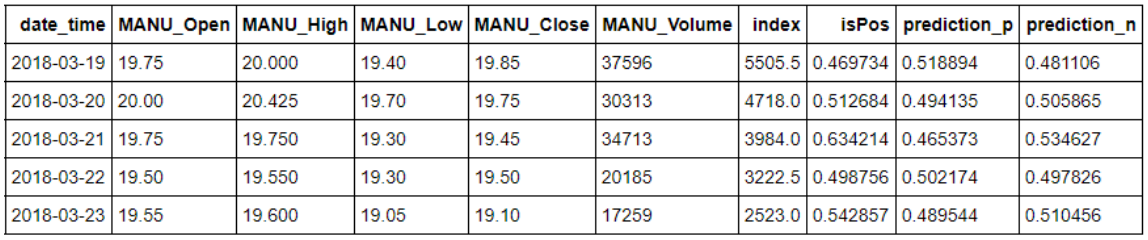
Now I have sentiments for the day. Next step I collect financial data.

9.4 Collecting financial data

For collecting financial data, I used googlefinance api, which take stock name, exchange, period for data and interval in input. For our case I collected data starting 3rd March 29, 2018 for every day. The data received from API has date_time, open, high, low, close price and volume. I are using high price for our analysis.

Stock Name = MANU

Stock Exchange = NYSE



date_time	MANU_Open	MANU_High	MANU_Low	MANU_Close	MANU_Volume	index	isPos	prediction_p	prediction_n
2018-03-19	19.75	20.000	19.40	19.85	37596	5505.5	0.469734	0.518894	0.481106
2018-03-20	20.00	20.425	19.70	19.75	30313	4718.0	0.512684	0.494135	0.505865
2018-03-21	19.75	19.750	19.30	19.45	34713	3984.0	0.634214	0.465373	0.534627
2018-03-22	19.50	19.550	19.30	19.50	20185	3222.5	0.498756	0.502174	0.497826
2018-03-23	19.55	19.600	19.05	19.10	17259	2523.0	0.542857	0.489544	0.510456

Figure 12 : Stock price for MANU

From the data I can observe price increased for initial two days and then dropped. I will compare this with sentiments and understand how they are corelated.

9.5 Result: comparing stock values with sentiment for the day

For merging the data, I must aggregate the sentiments per tweet to per day. For this I used groupby function. On aggregation I realized the sentiment were very close to neutral, this can be because of two major reasons first, limited amount of data and second, divided opinion of people. On close observation I observed slight changes impacted the stock price for our case as shown in Figure 13Figure 12.

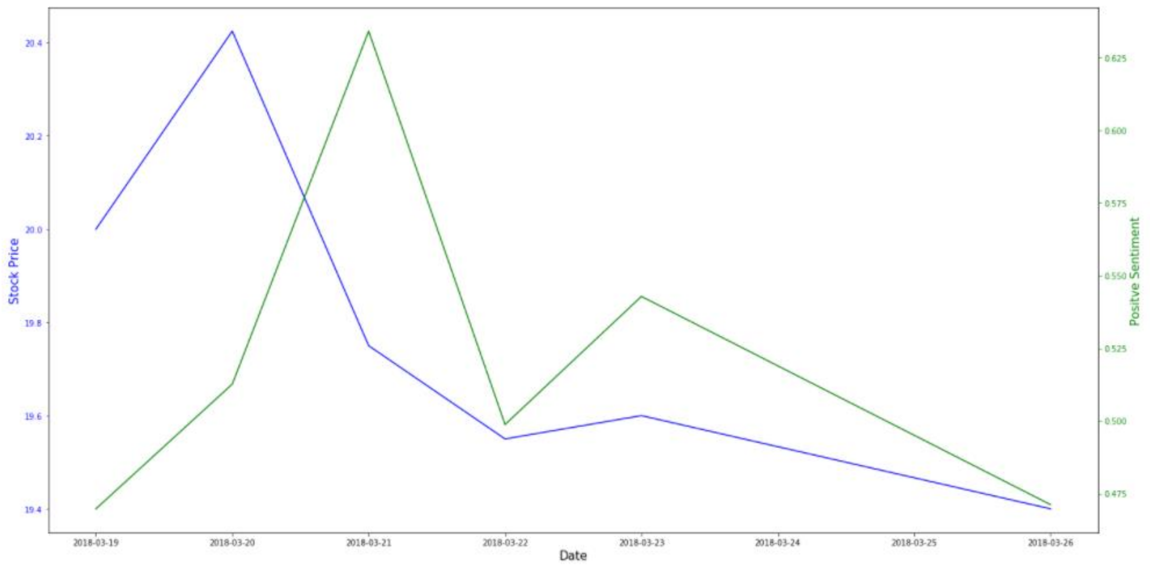


Figure 13 : Stock price vs Sentiment

In starting days with growth in positive sentiment stock price also increased even though the changes were very mild in sentiment throughout. After 21st march both positive sentiment and stock prices decreased because Manchester united lost matches back to back. The only problem I found was on day 20th march where our sentiments could not exactly match with stock price. Due to limited scope of research I have not analysed other factors which can directly impact stock price of all the stocks such as change in policy of reserve bank etc.

This research can be used as a prototype for a large-scale project since the result are in the right direction and point out the sentiments on twitter can help predict stock price. I are not assuming vice-versa because tweets are about the club and not stock, so stock price are not playing any role in sentiments on twitter for this research.

10 CONCLUSION

I investigate the relation between public sentiment and the stock price comes from google finance. For measuring sentiment towards ManUnited I collect tweets from twitter.com using #ManUtd. I have two-week tweets.

Firstly, our result shows even with small-scale Twitter feeds (40000 tweets) I can capture public mood using quick powerful and reliable natural language processing techniques.

Secondly, I have shown that a good correlation exists between rising and falling in stock values of a ManUtd to the public sentiment expressed through tweets. The tweets are classified into the categories: positive, negative. So I find out that the positive sentiment of public in twitter about a company reflect in its stock price and seems to have a promising future in research.

FUTURE WORK: I can consider forecasting model to predict one or two days ahead of stock values of every company and make a general model. Furthermore, I can apply clustering model to find different types of groups in our dataset, it is quite useful for companies to find out who are interested on their market.

11 BIBLIOGRAPHY

Hardeniya, N. (2015). *NLTK essentials.* Packt Publishing Ltd.

Pedregosa, F. (2011). Scikit-learn: Machine Learning in Python. *JMLR, 12*, 2825-2830.