Spam and Spammer Classification for Tweets Using Sentiment, Similarity Measure and Classifier Computation

Bindu P S[1], Rahul Shekar C[2], Puneeth Vishnukeerthy K[3], Savitha Vasanthan,Oshin Sharma[4], Anusha Sreedhar[5]
[1, 2, 3, 4, 5] Dept of Computer Science and Engineering
[1, 2, 3, 4, 5] Jain Global Campus, Karnataka, India

Abstract- *In today's world the numbers of ecommerce companies are increasing day by day and also huge number of products is coming into the market. When customers want to buy the products they generally just see a rating of the products and then purchase them. Later they come to know that products are not good. The rating is obtained from the reviews given by the users. There is no sentiment analysis performed on those reviews and if they are performed then the reviews which are not genuine are also taken into consideration. Twitter is a social media platform in which the number of users is increasing in millions which spans across different domains of users like college students, companies, politics, government programmers, film stars. In this paper first the tweets are collected for a set of hash tags related to various products. Once the tweets are collected the sentiment analysis is performed on those tweets as well as on products, sequence of data mining processes namely data cleaning, tokenization, frequency computation, feature vector computation are done. After that there is N*N comparison made in order to find the similarity between tweets. The rate variation of the sentiments and early time frame are measured for each user which are one of the important factors for spam user and tweet classification. The classifier will take into consideration weights and the factors namely Number Of Followers, Number of friends, Number of retweets, similarity measure, positive Rate Deviation, Negative Rate Deviation, Neutral Rate Deviation, Early Time Frame Ratio*

Keywords- Tokenization, Frequency computation, Feature Vector Computation, Sentiment Analysis, Data Cleaning, Similarity Measure

I. INTRODUCTION

Text mining is an approach which is responsible to find out short meaningful conclusions mining the data meaning the data it has it has lot of concepts like data processing data processing voice removal stop words removal tokenization frequency competition frequency computation this in this regard in this regard there is lot of work done in the literature. In this work first the list of hash tags are submitted for each of the products. Data Collection is performed using the twitter API. Once the tweets are Collected then set of trained positive and negative keywords are taken and then sentiment analysis is performed on tweets and also on products. The Tweets are taken and then set of words known as stop words are removed. Once each of the tweets has been cleaned then they are converted into set of words known as tokens. After that the repeated words per each tweet are combined and weight is assigned known as frequency so that the redundancy is removed. The Inverse Document Frequency and Feature Vector is computed for each of the tokens across all collected tweets. After that each tweet is compared against another tweet to measure similarity by making use of feature vector, intersection sum, union sum and then similarity measure. If the similarity measure of the tweet exceeds a certain threshold then the tweets are grouped as similar otherwise they are not grouped. For each of the tweet users the rate deviation is computed along with early time frame. Finally the classification measure will be based on tweet/user characteristics, similarity measure, rate deviations and time frame. Once we get the classification measure the tweets and user classification is performed.

II. BACKGROUND

As per the article [1] there are lot of fake reviews given for products and also The companies hire freelancers from places like the Philippines, Bangladesh, and Eastern Europe to write fake reviews for $1 to $10 each, and The Attorney General's Office slammed them with combined fines of more than $350,000.But this big reveal just scratched the surface.

1) Consumers' purchase decisions [2] are increasingly influenced by user-generated online reviews. Accordingly, there has been growing concern about the potential for posting deceptive opinion spam---fictitious reviews that have been deliberately written to sound authentic, to deceive the reader. But while this practice

has received considerable public attention and concern, relatively little is known about the actual prevalence, or rate, of deception in online review communities, and less still about the factors that influence.

2) Websites containing consumer reviews [3] are becoming targets of opinion spam fictitious opinions that have been deliberately written to sound authentic. Integrating work from psychology and computational linguistics

Spam campaigns spotted [4] in popular product review websites (e.g., amazon.com) have attracted mounting attention from both industry and academia, where a group of online posters are hired to collaboratively craft deceptive reviews for some target products. The goal is to manipulate perceived reputations of the targets for their best interests. Many efforts have been made to detect such colluders by extracting point wise features from individual reviewers/reviewer-groups, however, pairwise features which can potentially capture the underlying correlations among colluders and spammers and tweet users.

III. PROPOSED SYSTEM MODULES

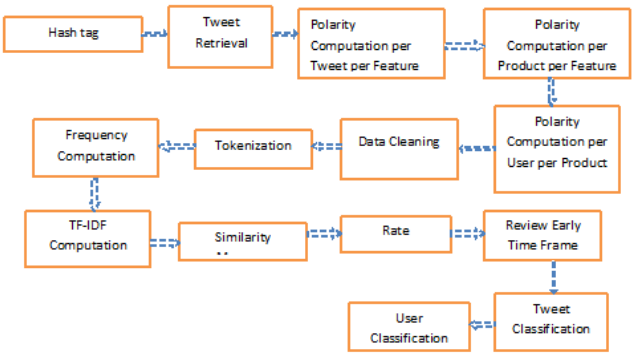


Fig1: Proposed System Modules

Fig1 shows the various modules used in the classification of tweets as spam and users as spam. In this section each of the modules are described in depth.

III.A Hashtag Collection

This module is responsible for taking input the hash tags and then save the hash tags in the format of (HashTagID, Hashtag and ProductID). HashTagID is an auto generated ID for the hash tag, Hash Tag is the actually hash tag in twitter on which the tweet has been performed. Product ID is the unique ID for the product to which the hashtag belongs.

III.B Tweet Retrieval

Twitter stores the reviews of the Products in the form of tweets which are associated with Hash Tags. This Module is responsible for collecting tweets from Twitter by Passing the Hash Tag, APPID and Secret Key. APPID and Secret Key are unique generated IDs by twitter when application is created. Hash tag is a concept under which the users will be able to Tweet. First a HTTP Request is sent from our application to the twitter application by passing the APPID, Secret Key, Oauth Key and OAuth Token. Once the authentication is successful then the list of hash tags are obtained. For each hash tag the tweets are collected and stored in the tweet matrix format which will contain the following

Parameter Name	Parameter Description
REVIEWID	Unique ID for Tweet
REVIEWDETAILS	The tweet description which the user has tweeted

Parameter Name	Parameter Description
LOOKUPID	Unique Id for the Product for which tweet has been performed
HASHTAG	Hashtag on which tweet is performed
TWEETSCREENNAME	Name of the screen
USERID	User who has performed tweet
LANGUAGE	This is the language code for the tweet, In this work it is always en
NOOFFOLLOWERS	Number of Followers for tweet
NOOFTWEETS	The number of tweets performed by the user
NOOFFRIENDS	The number of friends for the user
NOOFRETWEETS	Number of retweets performed by the user

III.C Polarity Computation per Tweet per Feature

This module is responsible for computing the sentiments of each tweet per feature. The positive sentiments, negative sentiments and neutral sentiments are found out per feature type. The feature types can be battery, memory, screen, touch. The sentiment analyzer module is trained with a set of positive, negative and neutral sentiments. The polarity computation matrix can be defined as below

Parameter Name	**Parameter Description**
REVIEWID	Unique Id for the tweet
POSITIVERATING	Positive Sentiment for tweet
NEGATIVERATING	Negative Sentiment for tweet
NEUTRALRATING	Neutral Sentiment for tweet
PRODUCTID	Unique ID for the product
FEATURETYPE	Feature Type for which sentiment has been computed. Ex- Battery, Memory, Camera etc

The algorithm used for computation of tweet wise sentiment is given in the Fig 2

Sentiment Analysis Per Feature Per Tweet

Input: Set of tweets collected

$\{T1, T2, \ldots\ldots\ldots\ldots\ldots, Tn\}$

Where,

$Ti = i^{th}\ tweet$

$n = number\ of\ tweets$

Output:

Tweet based sentiment matrix

$\{ST1, ST2, \ldots\ldots\ldots\ldots\ldots, STfn\}$

Where.,

$fn = n * f$

$f = number\ of\ features$

$STi = i^{th}\ sentiment$

$Each\ ST\ has\ the\ following$

$PS : positive\ sentiment$

$NS : negative\ sentiment$

$NuS : Nuetral\ Sentiment$

$T : tweet\ id$

$ft : feature\ type$

$P : product$

Details

1) Count the number of tweets Nc
2) for $k : 1 - - - > Nc$
   a) Obtain the k^{th} tweet
   b) Divide the review into a set of statements
   c) if the sentences can be treated as

$S_f - - - > \{s1, s2, \ldots\ldots\ldots\ldots, sm\}$

$si = i^{th}\ sentence$

$m = number\ of\ sentences$

d) for $j - - - > 1 : m$

1) Obtain the j^{th} sentence
2) Check whether the sentence sj has ft

$if\ sj \in ft$

$PS : Number\ of\ positive\ sentiments$

$NS : Number\ of\ negative\ sentiments$

$if\quad PS = NS = 0 \parallel sj \notin ft$

$Nus : Neutral\ Sentiments\ exist$

3) Repeat the above process for all sentences

$$PS_{Ti} = \sum_{j=1}^{m} PS_j$$

$$NS_{Ti} = \sum_{j=1}^{m} NS_j$$

$$NuS_{Ti} = \sum_{j=1}^{m} NuS_j$$

Where,

$PS_{Ti} = positive\ sentiments\ for\ i^{th}\ tweet$

$NS_{Ti} = negative\ sentiments\ for\ i^{th}\ tweet$

$NuS_{Ti} = nuetral\ sentiments\ for\ i^{th}\ review$

$PS_j = positive\ sentiment\ for\ j^{th}\ sentence$

$NS_j = negative\ sentiments\ for\ j^{th}\ sentence$

$NuS_j = nuetral\ sentiments\ for\ j^{th}\ sentence$

Fig2: Tweet Based Sentiments

$NS_j = negative\ sentiments\ for\ j^{th}\ sentence$

$NuS_j = nuetral\ sentiments\ for\ j^{th}\ sentence$

d) The sentiment set is a factor having the following

$ST_i = \{PS, NS, NuS, P, FT, Ti\}$

Fig2: Tweet Based Sentiments

Fig2 shows the Tweet Based Sentiment computation algorithm details.

III.D Polarity Computation per Product per Feature

This module is responsible for computing the polarity of each of the product per feature. For each of the product the reviews are found out and then the total polarity is computed based on summation of polarity across the reviews associated with the product.

The positive sentiment for the product is computed using the following

$$PS_{product,f_j} = \sum_{i=1}^{N_{tweets}} PST_i$$

Where,

$N_{tweets} = Number\ of\ tweets\ belonging\ to\ a\ product$

$PST_i = Positive\ Sentiment\ for\ i^{th}\ tweet$

$f_j = j^{th}\ feature$

$0 \le f \le N_f - 1$

$N_f = Number\ of\ features$

The negative sentiments for the product is computed using the following

$$NS_{product,f_j} = \sum_{i=1}^{N_{tweees}} NST_i$$

Where ,

$N_{tweets} = Number\ of\ tweets$

$NST_i = Negative\ Sentiment\ for\ the\ i^{th}\ tweet$

$f_j = j^{th}\ feature$

$0 \le f \le N_f - 1$

$N_f = Number\ of\ features$

The Neutral Sentiments are computed using the following equation

$$NP_{product,f_j} = \sum_{i=1}^{N_{tweets}} NPT_i$$

Where,

$NPT_i = Neutral\ Sentiments\ for\ i^{th}\ tweet$

$N_{tweets} = Number\ of\ tweets$

$f_j = j^{th}\ feature$

$0 \le j \le N_f$

$N_f = Number\ of\ features$

III.E Data Cleaning

The Data Cleaning algorithm is responsible for removal of stop words. Each of website is cleaned by removing the stop words from description. Stop words are the set of words which do not have any specific meaning. The data mining forum has defined set of keywords which do not have any meaning like *a, able, about, across, after, all, almost, also, am, among, an etc.*

III.F Tokenization

Tokenization is a process of converting the clean data into a set of words known as tokens.

III.G Frequency Computation

This is a process in which the frequency computation is performed. For each of the tweets the frequency is computed. Frequency is number of times a i^{th} token appears in j^{th} tweet. The frequency computation can be done using algorithm defined in fig3.

Frequency Computation Algorithm

Input : *set of tokens* $\{t1, t2, \ldots\ldots\ldots\ldots\ldots, tn\}$

Output : $\{w1, w2, \ldots\ldots\ldots\ldots wm\}$

$m <= n$

m = *number of words after freq computation*

n = *number of words before freq computation*

Details :

1) *measure the count of set of tokens* N_{token}

2) *find the unique set of tokens*

$\{u1, u2, \ldots\ldots uk\}$

$k < n$ *if tokens repeat*

$k <= n$ *if tokens are not repeated*

3) *measure the count of unique token ui in the set of* $\{t1, t2, \ldots\ldots\ldots\ldots\ldots, tn\}$ *call it as ci*

4)*Now a map is created with key as u and value as c*

Each c will contain{*tokenname*, *url*, *frequency*}

Fig3: Frequency Computation Algorithm

III.H TF-IDF Computation

The TF-IDF computation is performed by measuring the inverse document frequency which depends on frequency and number of tweets in which token is present.

The inverse document frequency is given by the formula

$$IDFT = \log(\frac{textFrequency}{NoOftweets})$$

The TF-IDF is defined as

$$v_i = tfi * idft_i$$

Where tf is the frequency of the i^{th} word and the IDFT is the inverse document frequency of the i^{th} word. The TF-IDF Computation is done using the algorithm used in fig4

Text Frequency – Inverse Document Frequency

$\{w1, w2, \ldots\ldots\ldots, wn\}$

set of words which belong to frequency set

each w has the following

wid : *unique id for word w*

f : *frequency of word w*

tweetId : *unique id for the tweet*

Output :

$\{o1, o2, \ldots\ldots ot\}$

o are se t of words

t total words

each $o \in \{fv1, fv2, \ldots\ldots\ldots, fvn\}$

fv is a feature vector

Details

1) *Find the set of tweets*

$\{t1, t2, t3, \ldots\ldots\ldots, tn\}$

2) *for each tweet*

a) *find the set of tokens* $\{t1, t2, \ldots\ldots tk\}$

b) *measure the count of tokens* N_{token}

c) *for* $k = 1 : N_{token}$

1. *obtain the kth token tk*
2. *measure number of websites in which tk is present*
3. *measure the IDF* = $\log(N / f)$
4. *obtain feature vector with* $fv = f * IDF$
5. *form the value o which will have* {*tokenname* , *freq* , *n*, *IDF* , *fv* , *tweetid* }

Fig4: TF-IDF Algorithm

III.I Similarity Measure

This module is responsible for measuring the similarity between tweets. It does a N*N Comparison between tweets and group them if they exceed a certain threshold. The similarity measure algorithm can be described using the following steps

1. Consider that there are 2 tweets to be compared t1 and t2
2. Find the list of unique words in tweet t1
3. Find the list of words in the tweet t2
4. Find the intersection word set between t1 and t2
5. Find the union word set between t1 and t2
6. For each of the word in the intersection set
   a. Obtain the word
   b. Find the TF-IDF for word in tweet t1
   c. Find the TF-IDF for word in tweet t2
   d. If TF-IDF(t1)>= TF-IDF(t2) then

Intersectionsum= intersectionsum+ TF-IDF(t1)

Else

Intersectionsum= intersectionsum+ TF-IDF(t2)

7. For each of the word in the union set
   a. Obtain the word
   b. Find the TF-IDF for word in tweet t1
   c. Find the TF-IDF for word in tweet t2

d. If TF-IDF(t1)<= TF-IDF(t2) then
unionsum= unionsum+
TF-IDF(t1)
Else
unionsum = unionsum +
TF-IDF(t2)

e. Compute the similarity measure using the following equation

$$similaritymeasure = \frac{intersectionsum}{unionsum}$$

III.J User Based Early Time Frame Ratio

User Based Early Time Frame Ratio provides the measure of how frequently the user performs the tweet. User Based is computed for all the users who have tweeted for the given hashtag.The user based early time frame ratio is given by the following equation

$$userbehaveweight = (1-(L-F)/T)*(L-F)$$

Where,

$L = LastTweet\ performed\ by\ user$

$F = FirstTweet\ performed\ by\ user$

$T = diffrence\ between\ first\ and\ last\ tweet\ performed$
$by\ user$

III.K Rate Deviation

The rate deviation will provide the variance of the sentiments performed by the user for the given product. If the variance is large then the tweet is closer to spam. The rate deviation computed for sentiments can be defined as below

$$RateDevPositive = positive - \frac{avgPositiv}{Ntweets}$$

Where,

$positive = positive\ sentiments\ total\ for\ user\ and\ producti$

$avgPositiv = Average\ positive\ sentiments$

$Ntweets = Number\ of\ tweets$

In a similar fashion the rate deviation would be computed for negative and neutral sentiments per product and per user.

III.L Classification of Tweets and Users

First the various user characteristics, similarity measure, early time frame and rate deviation are normalized with respect to total value across tweets/users. After that the total measure is computed for each of the tweets using the following equation

$$TM = w1 * N_{followers} + w2 * N_{friends} + w3 * N_{retweets} + w4 * P_{spam} + w4 * N_{tweets} + w5 * SM_{tweet} + w6 * PRD_{tweet} + w7 * NRD_{tweet} + w7 * NuRD_{tweet} + w8 * ETF_{user}$$

Where,

$w_i = adjustable\ weights\ for\ spam\ identification$

$N_{followers} = Number\ of\ followers$

$N_{friends} = Number\ of\ friends$

$N_{retweets} = Number\ of\ retweets$

$SM_{tweet} = similarity\ measure\ for\ tweet$

$PRD_{tweet} = positive\ rate\ deviation\ of\ user$
$who\ tweeted\ for\ specific\ product$

$NRD_{tweet} = negative\ rate\ deviation\ of\ user$
$who\ tweeted\ for\ specific\ product$

$NuRD_{tweet} = neutral\ rate\ deviation\ of\ user$
$who\ tweeted\ for\ specific\ product$

$ETF_{tweet} = early\ time\ frame\ for\ a\ tweet$

The threshold is found out. Once the total measure exceeds the threshold it is treated as spam otherwise it is treated as non-spam.

IV. EXPERIMENT RESULTS AND ANALYSIS

A complete web application has been created using Spring Framework along with ExtJS and Angular JS framework in order to execute the various stages for Spam Identification. This section describes the output of each section

Hash tag Submission

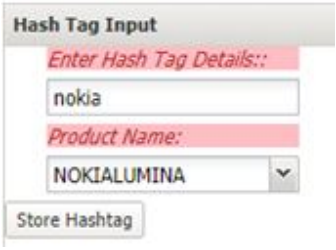
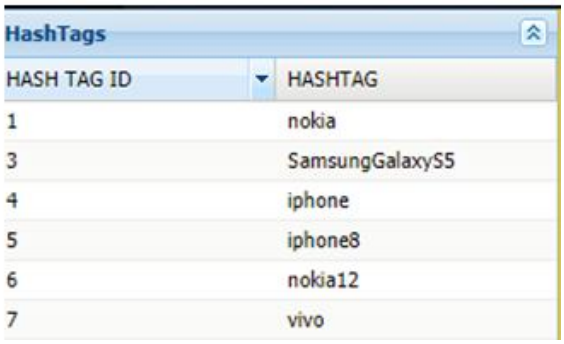


Fig5: Hashtag Submission Input

Asjhdhash Fig 5 shows the Hashtag Submission Screen in which the user will be able to select the product and then provide a hashtag in twitter which corresponds to the product.

View Hashtags



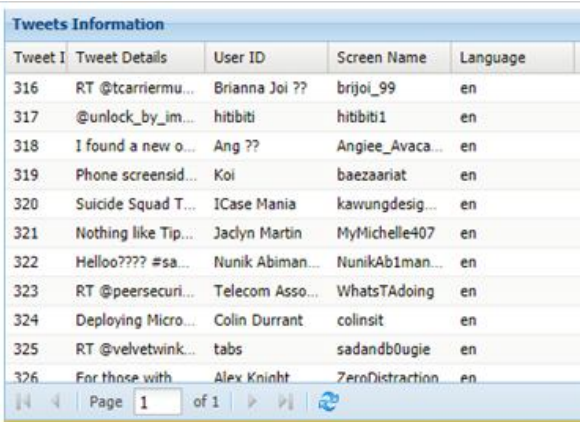
HashTags

HASH TAG ID	HASHTAG
1	nokia
3	SamsungGalaxyS5
4	iphone
5	iphone8
6	nokia12
7	vivo

Fig6: View Hashtags

Fig5 shows the hash tags which have been submitted in the application. Hashtag ID is the unique ID generated for the hashtag and Hashtag is the hash tag used in twitter application for peforming the tweet.

Data collection using Twitter



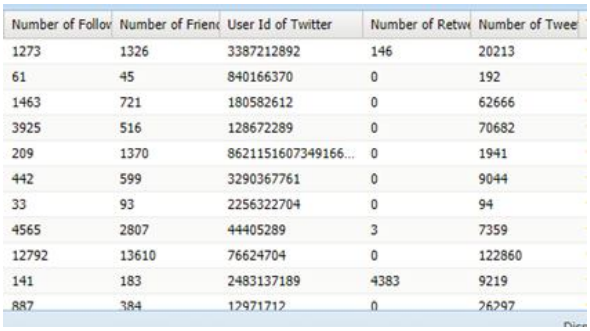
Tweets Information

Tweet I	Tweet Details	User ID	Screen Name	Language
316	RT @tcarriermu...	Brianna Joi ??	brijoi_99	en
317	@unlock_by_im...	hitibiti	hitibiti1	en
318	I found a new o...	Ang ??	Angiee_Avaca...	en
319	Phone screensid...	Koi	baezaariat	en
320	Suicide Squad T...	ICase Mania	kawungdesig...	en
321	Nothing like Tip...	Jaclyn Martin	MyMichelle407	en
322	Helloo???? #sa...	Nunik Abiman...	NunikAb1man...	en
323	RT @peersecuri...	Telecom Asso...	WhatsTAdoing	en
324	Deploying Micro...	Colin Durrant	colinsit	en
325	RT @velvetwink...	tabs	sadandb0ugie	en
326	For those with...	Alex Knight	ZeroDistraction	en

Page 1 of 1

Fig 7: Data Collection part1 Grid

Fig 7 shows the data collected from twitter for the set of hashtag. TweetID is the unique ID for the tweet. Tweet Details is the details of the statements which the user has tweeted.User ID is the unique User ID for tweet. Screen Name is the name of the screen on which the tweet has been performed. Language is the unique language code here the language code is en because only tweets related to english language are retrived.

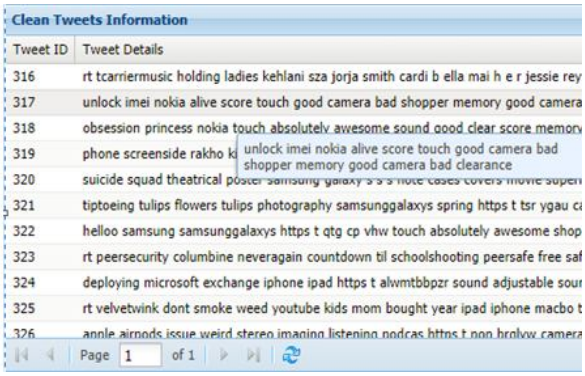


Number of Follov	Number of Frienc	User Id of Twitter	Number of Retw	Number of Twee
1273	1326	3387212892	146	20213
61	45	840166370	0	192
1463	721	180582612	0	62666
3925	516	128672289	0	70682
209	1370	8621151607349166...	0	1941
442	599	3290367761	0	9044
33	93	2256322704	0	94
4565	2807	44405289	3	7359
12792	13610	76624704	0	122860
141	183	2483137189	4383	9219
887	384	12971712	0	26297

Fig 8: Data Collection using Twitter Part2

Fig 8 shows the remaining columns of the data collection. The first column indicates the number of followers for tweet, The Second Column indicates Number of friends, User Id of Twitter is the unique ID for user in twitter application. Number of Retweets perfomed on the tweet as well as Number of Tweets have been given in the above matrix.

Data Cleaning Results



Clean Tweets Information

Tweet ID	Tweet Details
316	rt tcarriermusic holding ladies kehlani sza jorja smith cardi b ella mai h e r jessie rey
317	unlock imei nokia alive score touch good camera bad shopper memory good camera
318	obsession princess nokia touch absolutely awesome sound good clear score memory
319	phone screenside rakho k...
320	suicide squad theatrical poster samsung galaxy s 5 s 5 note cases covers movie super
321	tiptoeing tulips flowers tulips photography samsunggalaxys spring https t tsr ygau c
322	helloo samsung samsunggalaxys https t qtg cp vhw touch absolutely awesome shop
323	rt peersecurity columbine neveragain countdown til schoolshooting peersafe free saf
324	deploying microsoft exchange iphone ipad https t alwmtbbpzr sound adjustable sour
325	rt velvetwink dont smoke weed youtube kids mom bought year ipad iphone macbo t
326	apple airpods issue weird stereo imaging listening podcas https t pop brolvw camera

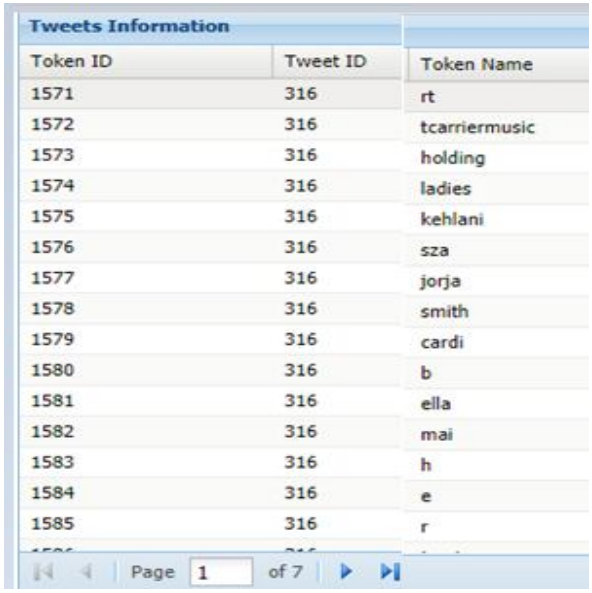
unlock imei nokia alive score touch good camera bad shopper memory good camera bad clearance

Page 1 of 1

Fig9: Data Cleaning Results

Fig shows the subpart of the data cleaning outout where the first column is the tweet id and the second column is the tweet details but in a clean format which will not contain even a single stop word

Tokenization Output



Tweets Information

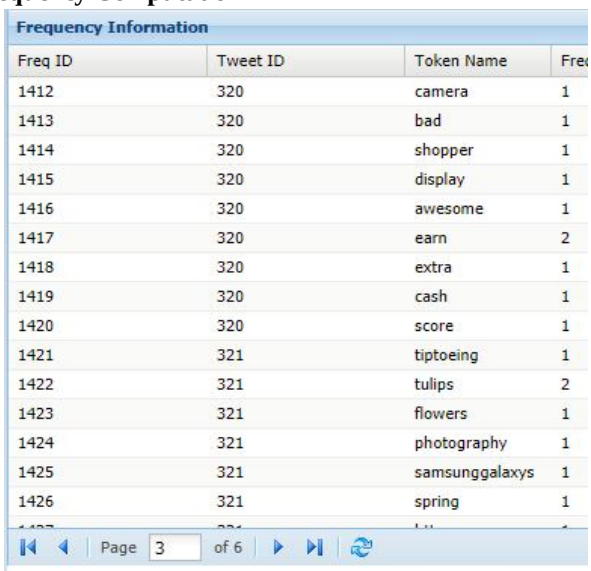
Token ID	Tweet ID	Token Name
1571	316	rt
1572	316	tcarriermusic
1573	316	holding
1574	316	ladies
1575	316	kehlani
1576	316	sza
1577	316	jorja
1578	316	smith
1579	316	cardi
1580	316	b
1581	316	ella
1582	316	mai
1583	316	h
1584	316	e
1585	316	r

Page 1 of 7

Fig 10: Tokenization output

Fig 10 shows the output for tokenization. As shown in the fig Token ID is the auto generated unique Id for each row of the tokenization matrix. Tweet ID is the unique ID for the tweet. Token Name is the name of the token.

Frequency Computation



Frequency Information

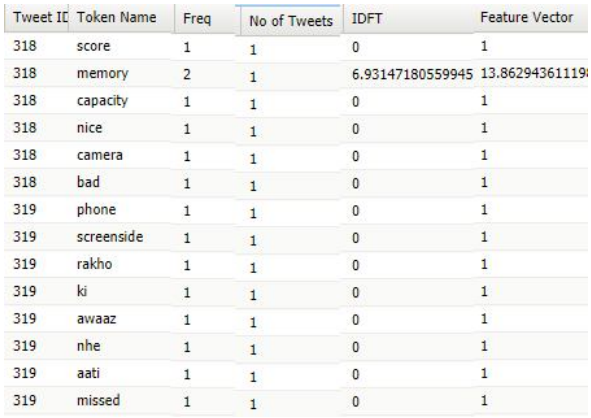
Freq ID	Tweet ID	Token Name	Fre
1412	320	camera	1
1413	320	bad	1
1414	320	shopper	1
1415	320	display	1
1416	320	awesome	1
1417	320	earn	2
1418	320	extra	1
1419	320	cash	1
1420	320	score	1
1421	321	tiptoeing	1
1422	321	tulips	2
1423	321	flowers	1
1424	321	photography	1
1425	321	samsunggalaxys	1
1426	321	spring	1

Page 3 of 6

Fig 11: Frequency Computation

Fig 11 shows the output of frequency computation. As shown in the frequency computation matrix the first column indicates FreqID which is the unique ID for each row of frequency computation. Tweet ID is the unique ID for the tweet. Token Name is the word which is present in the tweet and Frequency is number of time word is repeated in the tweet. For example display is repeated 1 in the tweet number 230 where as earn is present twice in the tweet number 320.

Feature Vector Computation



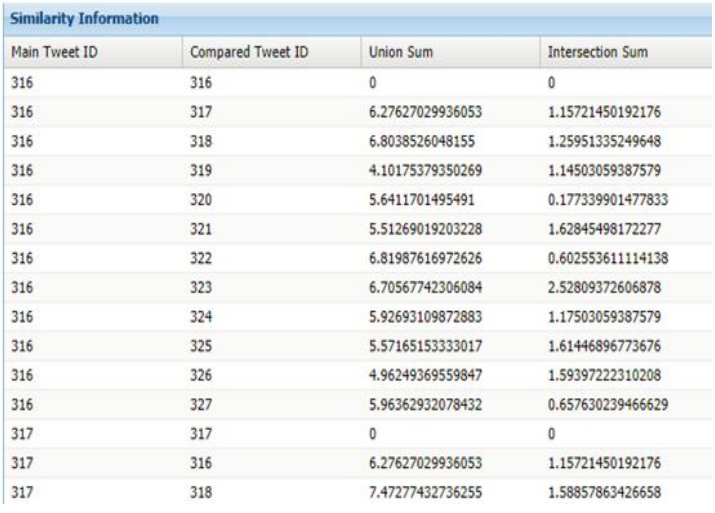
Tweet ID	Token Name	Freq	No of Tweets	IDFT	Feature Vector
318	score	1	1	0	1
318	memory	2	1	6.93147180559945	13.86294361119
318	capacity	1	1	0	1
318	nice	1	1	0	1
318	camera	1	1	0	1
318	bad	1	1	0	1
319	phone	1	1	0	1
319	screenside	1	1	0	1
319	rakho	1	1	0	1
319	ki	1	1	0	1
319	awaaz	1	1	0	1
319	nhe	1	1	0	1
319	aati	1	1	0	1
319	missed	1	1	0	1

Fig 12: Feature Vector Computation

Fig 12 shows the feature vector computation in which the first column indicates the tweet id which is the unique ID for tweet. The second column is the token name, the third column is the frequency of the token , Number of tweets are number of tweets in which word is present, IDFT is the inverse document frequency for word and Feature Vector is TF-IDF for the word in the tweet.

Similarity Measure

` Fig 13 shows the similarity measure. As shown in the fig there is Main Tweet ID is the tweet which is taken under consideration, Compared Tweet ID is the tweet with which the main tweet id is compared. Union Sum and Intersection Sum for each similarity measure are also shown.



Similarity Information

Main Tweet ID	Compared Tweet ID	Union Sum	Intersection Sum
316	316	0	0
316	317	6.27627029936053	1.15721450192176
316	318	6.8038526048155	1.25951335249648
316	319	4.10175379350269	1.14503059387579
316	320	5.6411701495491	0.177339901477833
316	321	5.51269019203228	1.62845498172277
316	322	6.81987616972626	0.602553611114138
316	323	6.70567742306084	2.52809372606878
316	324	5.92693109872883	1.17503059387579
316	325	5.57165153333017	1.61446896773676
316	326	4.96249369559847	1.59397222310208
316	327	5.96362932078432	0.657630239466629
317	317	0	0
317	316	6.27627029936053	1.15721450192176
317	318	7.47277432736255	1.58857863426658

Fig 13: Similarity Measure Part1

Similarity Measure	Group ID	Similarity
0	1	true
0.184379328283498	1	true
0.185117671656356	1	true
0.27915634421782	1	true
0.0314367226615222	1	true
0.295401142635667	1	true
0.0883525735832127	1	true
0.377007954091954	1	true
0.198252784502219	1	true
0.289764885344829	1	true
0.321203878710389	1	true
0.110273493554448	1	true
0	2	true
0.184379328283498	2	true
0.212582176936587	2	true

Fig 14: Similarity Measure Part2

Fig 14 shows the similarity measure parameters in which each of the tweet similarity measure is shown along with group id to which the tweet belongs and similarity status will be true if the tweets are similar otherwise it will be false.

Tweet Based Sentiments Measure

Tweet based sentiments measure contains the tweet id, positive, negative, nuetral sentiment along with feature type for each of the sentiments. Fig 15 shows the sentiment analysis for each tweet

Polarity Computation Output

Review ID	Product Name	Positive Rating	Negative Rating	Neutral Rating	Feature Type
316	NOKIALUMINA	0	0	1	BATTERY
316	NOKIALUMINA	1	1	0	MEMORY
316	NOKIALUMINA	4	0	0	TOUCH
316	NOKIALUMINA	0	0	1	SCREEN
316	NOKIALUMINA	1	1	0	CAMERA
317	NOKIALUMINA	0	0	1	BATTERY
317	NOKIALUMINA	1	0	0	MEMORY
317	NOKIALUMINA	2	0	0	TOUCH
317	NOKIALUMINA	0	0	1	SCREEN
317	NOKIALUMINA	0	2	0	CAMERA
318	NOKIALUMINA	0	0	1	BATTERY
318	NOKIALUMINA	2	0	0	MEMORY
318	NOKIALUMINA	4	0	0	TOUCH
318	NOKIALUMINA	0	0	1	SCREEN
318	NOKIALUMINA	0	1	0	CAMERA
319	SAMSUNGGALAXY1	0	0	1	BATTERY
319	SAMSUNGGALAXY1	0	1	0	MEMORY
319	SAMSUNGGALAXY1	5	0	0	TOUCH
319	SAMSUNGGALAXY1	0	0	1	SCREEN
319	SAMSUNGGALAXY1	1	0	0	CAMERA

Page 1 of 3

Fig 15: Tweet Based Sentiments

Product Based Sentiments

Total Polarity Output

Product Name	Positive Rating	Negative Ratin	Neutral Rating	Feature Typ
NOKIALUMINA	0	0	3	BATTERY
NOKIALUMINA	4	1	0	MEMORY
NOKIALUMINA	10	0	0	TOUCH
NOKIALUMINA	0	0	3	SCREEN
NOKIALUMINA	1	4	0	CAMERA
SAMSUNGGALAXY1	0	0	4	BATTERY
SAMSUNGGALAXY1	2	2	1	MEMORY
SAMSUNGGALAXY1	13	0	1	TOUCH
SAMSUNGGALAXY1	0	0	4	SCREEN
SAMSUNGGALAXY1	2	1	1	CAMERA
IPHONE6	0	0	4	BATTERY

Page 1 of 1

Fig16: Product Based Sentiments

Fig 16 shows the product based sentiments. As shown in the fig the first column is the product name, the second column is the total positive sentiments across all tweets , third column is the total negative sentiments across all tweets, neutral rating is the total neutral sentiments across all the tweets and finally feature type for which sentiments have been computed.

Sentiments Graph

ProductInfo

Product ID	Product Name
1	NOKIALUMINA
2	SAMSUNGGALAXY1
3	SAMSUNGGALAXY2
4	IPHONE6
5	IPHONE7
6	MICROMAX
7	ONE PLUS 5T

Fig17: Product Information

Fig 17 shows the product information which indicates the product name and an associated product id.

Find Graphs

Feature Types:

TOUCH

Fig18: Feature Type Selection

Fig 18 shows that the feature TOUCH has been selected

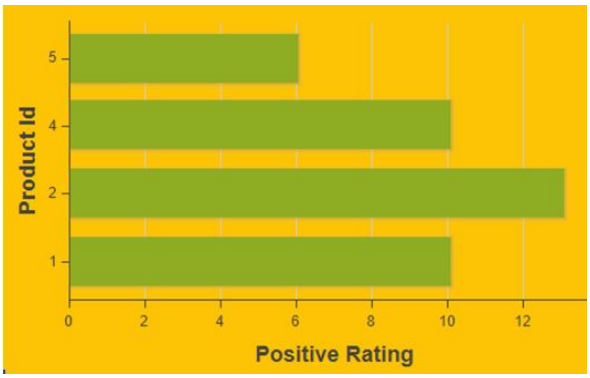


Fig 19: Sentiments Graph

Fig 19 shows the graph related to sentiments. As shown in the fig the yaxis indicates the product id and x axis represents the positive rating for each of the product. In a similar fashion negative and neutral graphs are obtained for each of the feature types.

Classification Measure

Classifier Information

Tweet ID	Followers Probability	Tweet Probability	Friends Probability
316	0.045976596359...	0.059426753652...	0.061077844311...
317	0.002203120485...	0.000564485069...	0.002072777521...
318	0.052838774920...	0.184239694472...	0.033210502072...
319	0.141758162380...	0.207806946106...	0.023767848917...
320	0.007548396417...	0.005706591245...	0.063104560110...
321	0.015963594336...	0.026589598774...	0.027590971902...
322	0.001191852065...	0.000276362481...	0.004283740211...
323	0.164872869112...	0.021635654288...	0.129295255642...
324	0.462005200809...	0.361211643680...	0.626900046061...
325	0.005092458826...	0.027104103394...	0.008429295255...
326	0.032035538861...	0.077313874278...	0.017687701520...
327	0.068513435423...	0.028124292556...	0.002579456471...

Fig 20: Classifier Part1

Fig 20 shows the part1 of classifier. As shown in the classifier tweet id is the unique id for the tweet. Followers Probability is the number of followers for the tweet to the number of followers across all the tweets. Tweet Probability is the number of tweets to the total number of tweets across all users. Friends Probability is the ratio of number of friends for the user who has performed tweet to the total number of friends across all tweets

Retweet Probability	Spam Probability	Similarity Probability
0.000429243854...	0.051257761077...	0.0982331257824843
0	0.169882865285...	0.0696636039578272
0	0.063254258351...	0.086034406114221
0	0.097473775163...	0.087518332864903
0	0.067567048693...	0.0187688706237814
0	0.058293140048...	0.114126408633914
0	0.078235530065...	0.0761373329340819
0.000008820079...	0.097473775163...	0.109877260169983
0	0.060672451887...	0.0755503296806349
0.012886135717...	0.042470716321...	0.111230473120075
0	0.162160916863...	0.0817664978162157
0	0.051257761077...	0.0710933583018973

Fig 21: Classifier Part2

Fig 21 shows the Classifer Matrix another 3 columns. Retweet Probability is the number of retweets for the tweet to the total number of retweets. Spam Probability is the ratio of number of spam probability for tweet to the total spam probability across all the tweets. Similarity Measure is the measure of similarity for the tweet to other tweets to the total similarity addition across all the tweets.

Rate Deviation Positive	Rate Deviation Negativ	Rate Deviation Nuetra
0.0555555555555558	0.0699708454810504	0.0833333333333333
0.138888888888892	0.0699708454810504	0.0833333333333333
0.0555555555555558	0.0816326530612243	0.0833333333333333
0.046875	0.0634110787172019	0.0989583333333333
0.151041666666667	0.0634110787172019	0.078125
0.046875	0.0634110787172019	0.0989583333333333
0.0885416666666667	0.0721574344023332	0.0572916666666667
0.09375	0.0481049562682221	0.0833333333333333
0.114583333333333	0.074344023323616	0.0833333333333333
0.0729166666666667	0.074344023323616	0.0833333333333333
0.0520833333333333	0.074344023323616	0.0833333333333333
0.0833333333333333	0.244897959183676	0.0833333333333333

Fig 22: Classifier Part3

Fig 22 shows the outout for classification measure with additional columns,Rate Deviation Positive which is the rate deviation for positive for the tweet to the total rate deviation positive acorss all the tweets. In a same fashion for negative and neutral rate deviation.

Early Time Frame Meas	Threshold	Total Measure	Is Spam
0	0.3997048984...	0.2663025582...	false
0	0.3997048984...	0.4817936922...	true
0	0.3997048984...	0.2910672316...	false
0	0.3997048984...	0.3093595875...	false
0	0.3997048984...	0.3092206166...	false
0	0.3997048984...	0.2701649142...	false
0	0.3997048984...	0.3199093281...	false
0	0.3997048984...	0.3402088646...	false
0	0.3997048984...	0.3195547676...	false
0	0.3997048984...	0.2813071093...	false
0	0.3997048984...	0.4145935308...	true
0	0.3997048984...	0.4996311231...	true

Fig 23: Classifier Part4

Fig 23 shows the classifier part4 with additional 4 columns. As shown in the fig early time frame is measured along with threshold and total measure for each tweet which is summation of all parameters with weight adjustability. If the total measure exceeds the thereshold then the tweet is regarded as spam otherwise it is regarded as non spam.

Classifier Graph- TWEET

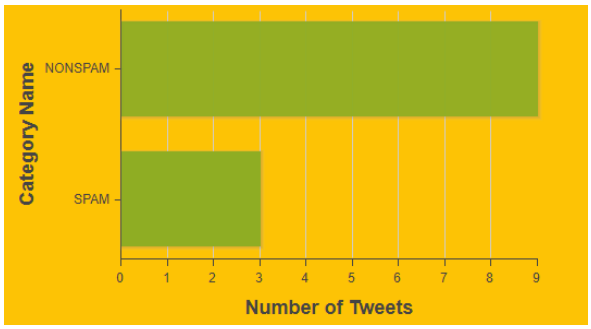
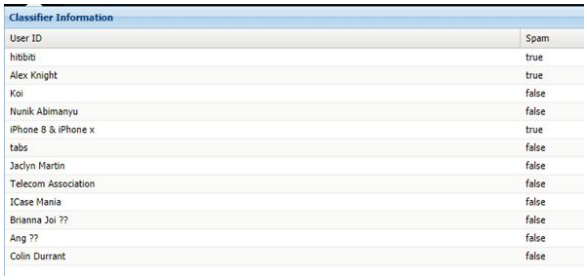


Fig 24: Classification Of Tweets

Fig 24 shows the graph of amoung total tweets how many of them are spam and how many are non spam. As shown in the graph 3 tweets are treated as spam and remaining are treated as non spam.

Classify User



Classifier Information

User ID	Spam
hitibiti	true
Alex Knight	true
Koi	false
Nunik Abimanyu	false
iPhone 8 & iPhone x	true
tabs	false
Jaclyn Martin	false
Telecom Association	false
ICase Mania	false
Brianna Joi ??	false
Ang ??	false
Colin Durrant	false

Fig 25: Classifier User

Fig 25 shows the user id and whether the user is a spammer or not. If user is a spammer it is marked as true otherwise it is marked as false.

V. CONCLUSION

In this paper a series of processes have been applied which will classify the tweet /user as a spam or non spam. Also the results are shown for each of the processes like data collection using twitter, data cleaning , tokenization, frequency computation, feature vector computation, tweet wise polarity computation, product wise polarity computation, rate deviation for positive, negative and nuetral polarity, early time frame measure and finally classification measure is done.

REFERENCES

[1] J. Donfro, A whopping 20 % of yelp reviews are fake. http://www.businessinsider.com/20-percent-of-yelp-reviews-fake-2013-9.Accessed: 2015-07-30.
[2] M. Ott, C. Cardie, and J. T. Hancock. Estimating the prevalence of deception in online review communities. In ACM WWW, 2012.
[3] M. Ott, Y. Choi, C. Cardie, and J. T. Hancock. Finding deceptive opinion spam by any stretch of the imagination.In ACL, 2011.
[4] Ch. Xu and J. Zhang. Combating product review spam campaigns via multiple heterogeneous pairwise features. In SIAM International Conference on Data Mining, 2014.
[5] N. Jindal and B. Liu, Opinion spam and analysis, in Proceedings of WSDM, 2008.
[6] Thorsten Joachims, Optimizing search engines using clickthrough data, in Proceedings of KDD, 2002.
[7] F. Li, M. Huang, Y. Yang, and X. Zhu, Learning to identify review spam, in Proceedings of IJCAI, 2011.
[8] E.P. Lim, V.A. Nguyen, N. Jindal, B. Liu, and H.W. Lauw, Detecting product review spammers using rating behaviors, in Proceedings of CIKM, 2010.