Fall 10-15-2017

Presidential Job Approval Rating Analysis Through Social Media

Subramanian Venkataraman
vsubu1@gmail.com

Subramanian Venkataraman
Harrisburg University of Science and Technology, vsubu1@gmail.com

HARRISBURG UNIVERSITY OF SCIENCE AND TECHNOLOGY

Presidential Job Approval Rating Analysis Through Social Media

IMPLEMENTATION AND ANALYSIS

SUBRAMANIAN VENKATARAMAN - 171383

Abstract

The aim of this study is to identify patterns in President Trump's approval in the Twitter universe through Social Media and Sentiment Analysis, and compare against scientific polling to get meaningful insights on the limitations of Social Media Analytics. For the purposes for this exercise, results from scientific polling will be considered the true measure of approval, and will be used as control. In order to perform sentiment analysis, we have used supervisory learning using Naive Bayes Classifier algorithm which produced 0.862667 accuracy levels.

1. Introduction

Social Media Analytics is now a fast-growing technique to measure sentiment of a product, or company or a public figure. As such, there are many assumptions behind the validity of this technique. The most important of which, by the sheer volume and velocity of data, the sample is considered demographically representative of the population by the law of large numbers. Any insight gained from Social Media Analytics derives its validity from this assumption. And therefore, it would be the responsibility of the analysts to check this assumption against another sophisticated analytical technique, scientific polling.

The reason scientific polling is being used to measure the accuracy of sentiment analysis is because of the long history of this methodology providing consistent and reasonably accurate predictions across multiple countries for more than half a century. There have been a few famous misses, but considering the number of times the polls have gotten it right, it would be reasonable to consider scientific polling as the best tool available to measure sentiment.

This study captures Tweets from Twitter Social media , finds public sentiments from the tweets and categorizes them into positive and negative. The final aggregated result for each day is compared against the results of approval rating companies results to find out how effective is the social media in predicting President job approval rating.

2. Method

2.1 Approval Rating Procedure

In United States, the presidential job approval rating was introduced by George Gallup in 1937 to collect public support for the President of the US during his presidential power. An approval rating is a percentage calculated by a polling which indicates the percentage of elected persons who approve or disapprove of a person or a program.

The approval rating companies generally collects the database of 800 to 1200 registered voters, The selected interviewers are called through cellphones, landline phones or through online polling web site. They are given set of questions to share their view on President Trump's policies to answer if the approve or disapprove. The polling results data from the set of interviewers are collected and consolidated and aggregated. In order to give best accurate results, the data is weighted based on the methodological standards and historical accuracy. The polling results are adjusted for house effects to make it consistent with polling consensus. The data collected is accounted for uncertainty and wide range is estimated within which the Trump's approval rating could vary. It is typically arrived through scientific polling by measuring the percentage of a sample that approves of the President. A sample of a few hundred to a few thousand demographically representative adults are chosen to answer questions about Presidential performance, etc. and is compiled and adjusted based on demographic attributes to provide an extrapolated view of Approval among the electorate at large.

There are 3 types are polling takes place with respect to the President's Approval Rating which are discussed below.

2.1.1 All adults (A)

This poll a sample of adults regardless of their voter registration or likelihood of voting. This is useful for measuring how much the nation at large approves of the President and the Administration

2.1.2 Registered Voters (RV)

This polls only registered voters regardless of their likelihood of voting. This measure theoretical voter outcomes assuming 100% turn out

2.1.3 Likely Voters (LV)

This polls only registered voters who have a history of voting regularly and are likely to vote on Election Day. This has the most immediate impact for politicians are closely followed.

2.2 Types of polls by methodology

The following are the type of polls carried out based on pre-and post-election seasons

2.2.1 Benchmark Poll

This is typically the first poll before start of the campaign season. This is considered the benchmark against which all effects of campaigning activity are measured

2.2.2 Brushfire Poll

This typically happens once campaign kickoff and through the campaign. This is used to test messaging and where the candidate is gaining support or losing support. This poll tells which message or issue is gaining most traction with the populace.

2.2.3 Tracking Poll

These polls are taken through time to measure support and discount daily volatility. The sample is repeatedly polled through a week and a moving average is published to show a less volatile trend in support

To make the polling data more robust, calculating and adjusting the data for "house effects" through a systematic tendency for polling which will favor either a Democratic or Republican candidate.

Poll results adjustments as per 'house effects' is merely a reflection of the tendencies. By doing so, the rating companies do not have to change the voter's preferences instead showing the candidate by up by 3-4 points. The results generally indicate the different strategies and assumptions that the polling firms have used.

3. Data Collection from Twitter

For performing Trump Approval rating analysis, we have considered 3 weeks of data starting from Sep-26-2017 to Oct-17-2017 and for each day we considered around 20,000 tweets. The Twitter data for this duration is downloaded using Spark-Scala programming into Cassandra Data base for each day. The key-word used for downloading tweets from Twitter are Trump. Hence, all tweets talk about President Trump’s actions, statements are downloaded into local data base.

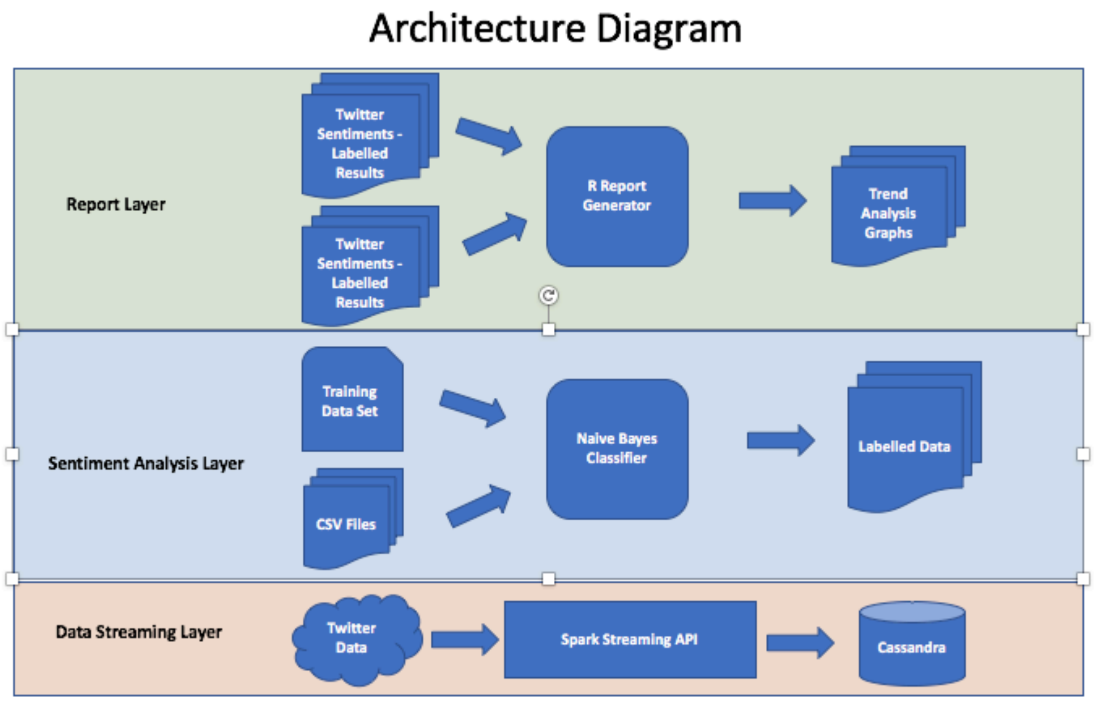
The data collected from Twitter includes, Twitter Id, Tweet Create Time, GeoLocation, Place, Tweet text, User name, Language, Location and Time Zone.

The data from Cassandra is exported into CSV file for each day in order to process them separately. We wrote a Sentiment Analyzer tool using Python which reads the CSV input data file and performs sentiment analysis using Naive Bayes Classifier. Based on the overall number of tweets, we calculate the percentage of tweets with positive and negative sentiments for each day.

Data Collection from Approval Rating Companies were captured manually from http://fivethirtyeight.com/ for the major approval rating companies.

The data from approval rating companies are compared against aggregated twitter sentiments for each day and correlation graph is generated to study the pattern of data for each day through a graphical representation and the results are analyzed and concluded to report the findings.

Figure 1



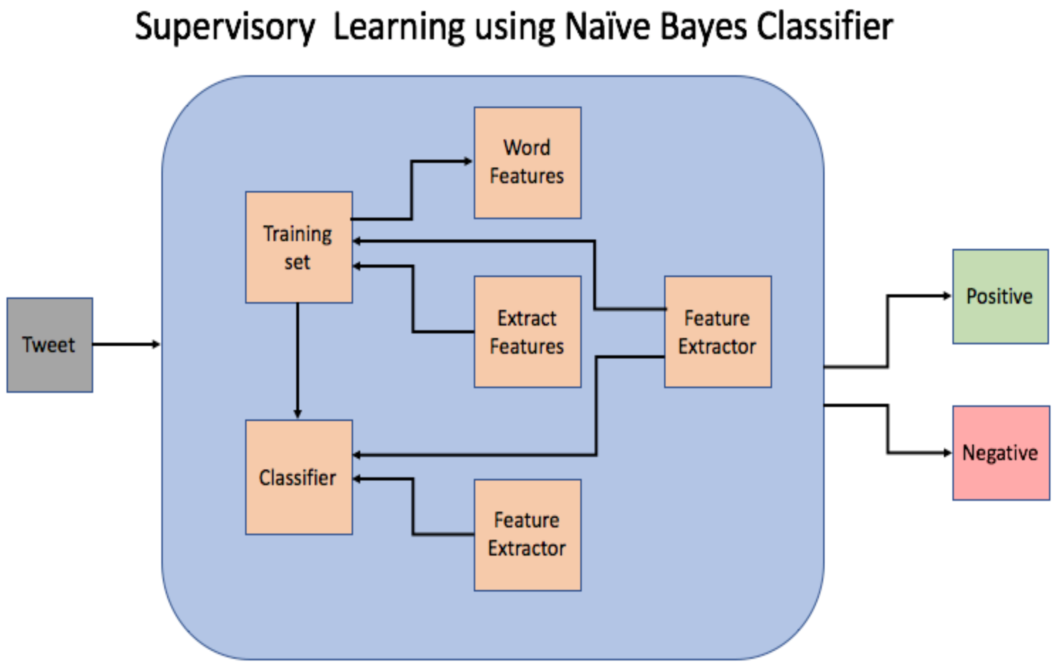
Architecture Diagram

4. Data Processing

1. The Twitter Search Program is written in Spark-Scala to download the data from Twitter based on search keywords and USA geo location.
2. This program takes few parameters such as "Trump", "Donald Trump", or any other hash tags based on government policies.

3. The data from Cassandra data base is exported into CSV file for each date. We considered 20,000 records per day from Sep-26-2017 to Oct-17-2017.
4. Python code is developed for performing Sentiment Analysis on Twitter Data using Naïve Bayes Classifier approach.
5. This Python program is run against each file for generating sentiment results for each record in the CSV file which generates 'positive' or 'negative' for each record.
6. The final sentiment results for each day is aggregated to percentage level based on the total number of tweets processed for each day.
7. R program is generated to develop graphs to perform trend analysis which reads the consolidated csv file for Twitter Sentiment results and approval rating company's results.
8. This R-Program generates trend graphs for Twitter sentiment results and approval rating company's results. For Twitter sentiment results, we have considered 3 days moving average value to smoothen the curves and spikes.

Figure 2



Sentiment Analysis Model Diagram

5. Sentiment Analysis

This module is implemented for preparing classifier through the following set of steps.

1. Removes the URLs from the twitter text
2. Removes the words that start with @. This is a kind of key word hence removed.
3. Removes Stop words from the tweets (Eg. 'an', 'be', 'some', 'for', 'do'
4. Removes the words that start with #Tags
5. Removes multiple white spaces
6. Removes & quot that appear in the text
7. Removes The key word 'RT :' which appears for tweet text.

8. Removes any numbers from the text
9. Removes non-English letters
10. Removes the special characters,'|', ',', '*', '#',
11. Removes repeating characters such as ‘!!!!’
12. Converts tweets to lower case
13. Adds Verb, Adjectives and Adverbs as they qualify the statements.
14. Labels the statement with positive and negative
15. Using NLTK sentiment analyzer finds the polarity of the tweets for the classifications of positive, negative, neutral and compound.
16. Generates the output in the CSV file with classification for each tweet.
17. The positive and negative label tweets are aggregated to percentage based on the number of tweets in the data set for the corresponding date.

6. Results

The Consolidated sentiment analysis results and data from approval rating companied are given below for the duration 9/27/2017 to 10/17/2017.

Table 1

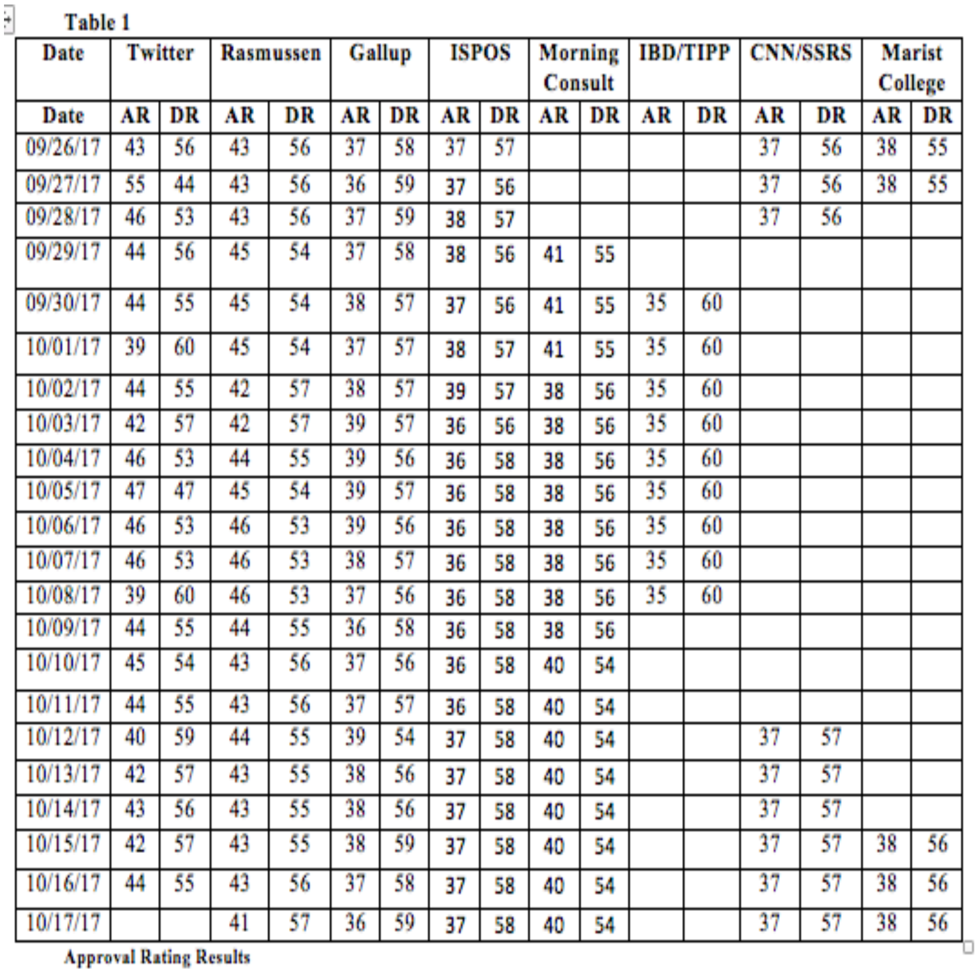


Table 1

Date	Twitter		Rasmussen		Gallup		ISPOS		Morning Consult		IBD/TIPP		CNN/SSRS		Marist College	
Date	AR	DR	AR	DR	AR	DR	AR	DR	AR	DR	AR	DR	AR	DR	AR	DR
09/26/17	43	56	43	56	37	58	37	57					37	56	38	55
09/27/17	55	44	43	56	36	59	37	56					37	56	38	55
09/28/17	46	53	43	56	37	59	38	57					37	56		
09/29/17	44	56	45	54	37	58	38	56	41	55						
09/30/17	44	55	45	54	38	57	37	56	41	55	35	60				
10/01/17	39	60	45	54	37	57	38	57	41	55	35	60				
10/02/17	44	55	42	57	38	57	39	57	38	56	35	60				
10/03/17	42	57	42	57	39	57	36	56	38	56	35	60				
10/04/17	46	53	44	55	39	56	36	58	38	56	35	60				
10/05/17	47	47	45	54	39	57	36	58	38	56	35	60				
10/06/17	46	53	46	53	39	56	36	58	38	56	35	60				
10/07/17	46	53	46	53	38	57	36	58	38	56	35	60				
10/08/17	39	60	46	53	37	56	36	58	38	56	35	60				
10/09/17	44	55	44	55	36	58	36	58	38	56						
10/10/17	45	54	43	56	37	56	36	58	40	54						
10/11/17	44	55	43	56	37	57	36	58	40	54						
10/12/17	40	59	44	55	39	54	37	58	40	54			37	57		
10/13/17	42	57	43	55	38	56	37	58	40	54			37	57		
10/14/17	43	56	43	55	38	56	37	58	40	54			37	57		
10/15/17	42	57	43	55	38	59	37	58	40	54			37	57	38	56
10/16/17	44	55	43	56	37	58	37	58	40	54			37	57	38	56
10/17/17			41	57	36	59	37	58	40	54			37	57	38	56

Approval Rating Results

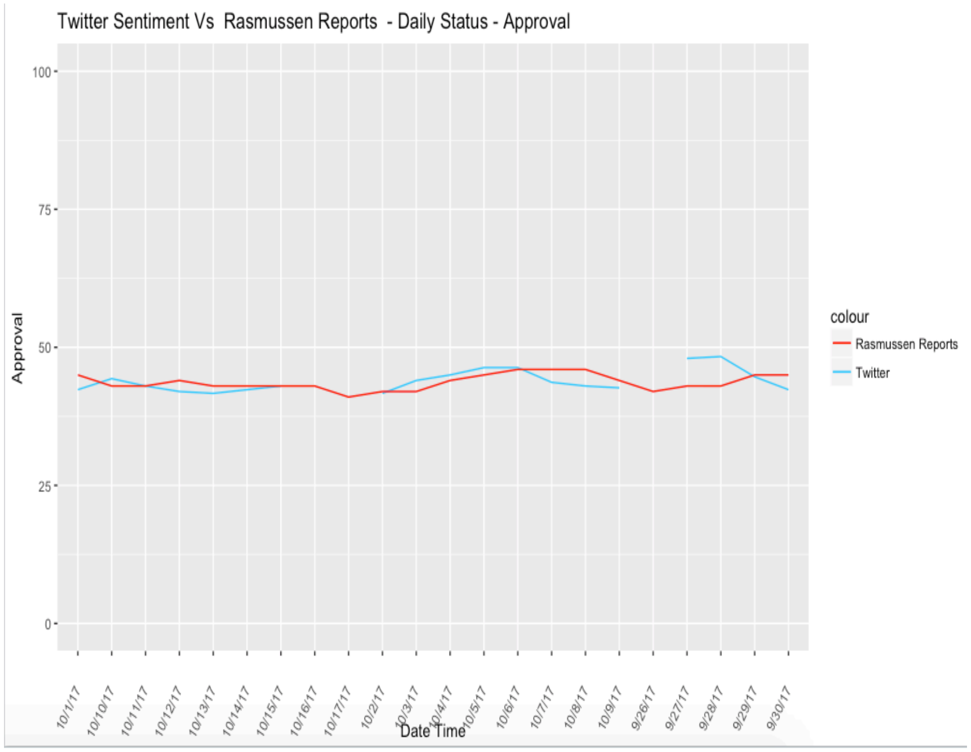
AR – Approval Rating **DR** - Disapproval Rating

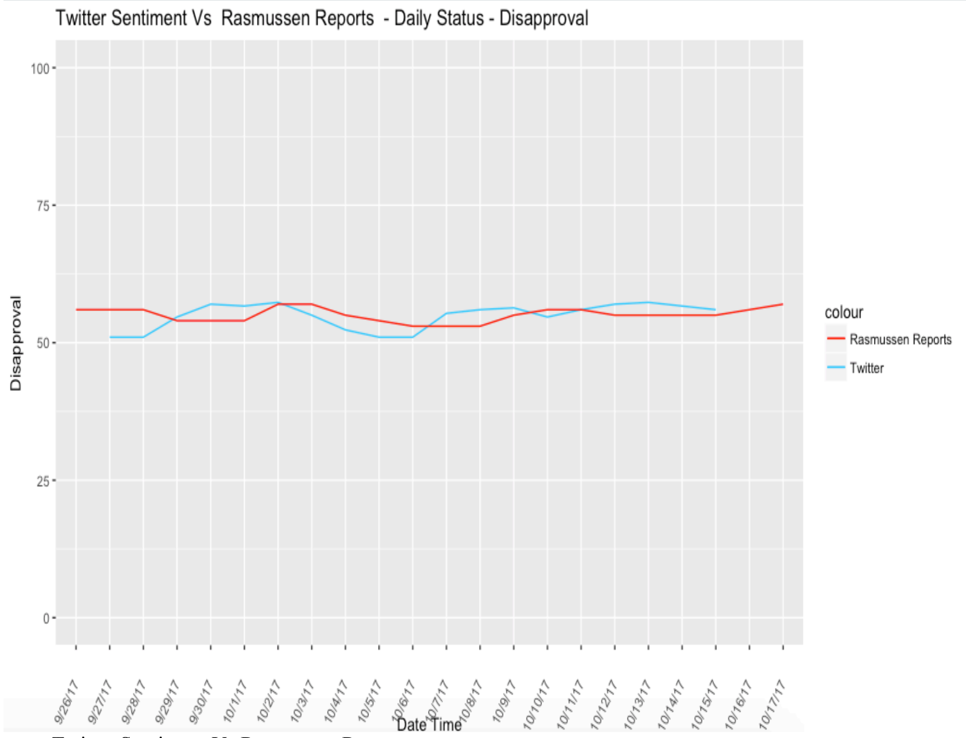
7. Results from Analytical Reports

7.1 Twitter Vs Rasmussen Reports

The following analytical reports provides the trend analysis between Twitter Approval Rating and Rating from Approval Rating companies. These reports have been generated using R programming by reading data from CSV file. These reports have been taken for 3 days moving average value in order to adjust the spikes in the trends.

Figure 3





Twitter Sentiment Vs Rasmussen Reports

7.1.1 Analysis

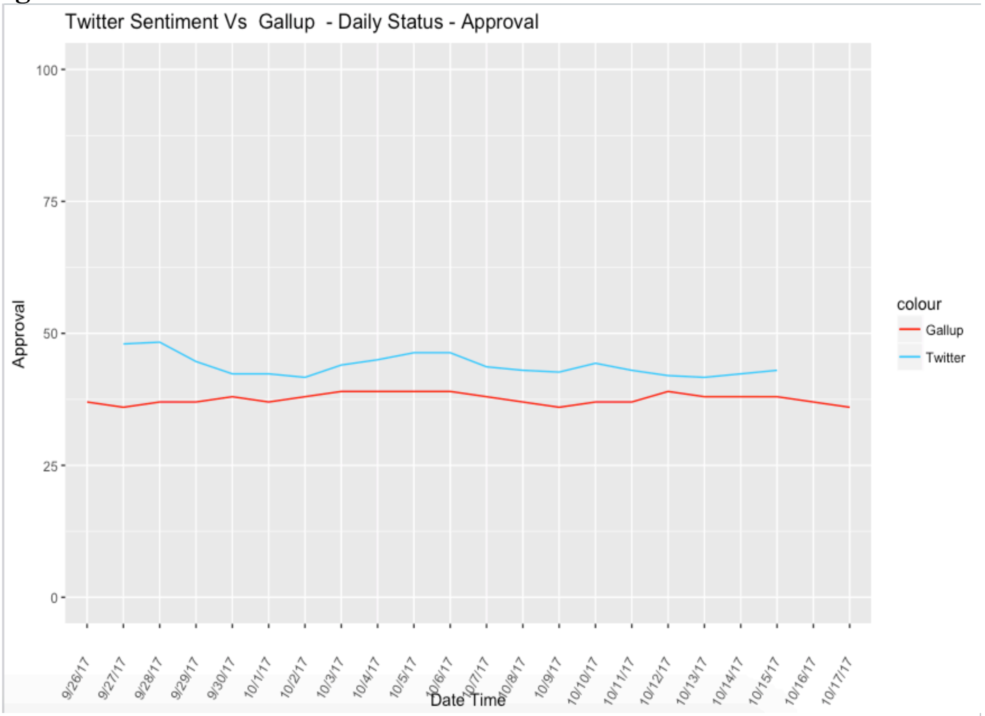
The Twitter sentiments data is trending in line with Rasmussen Reports for the President Job approval rating results for ‘Disapproved’ results. The data for Twitter is adjusted with 3 days moving average values.

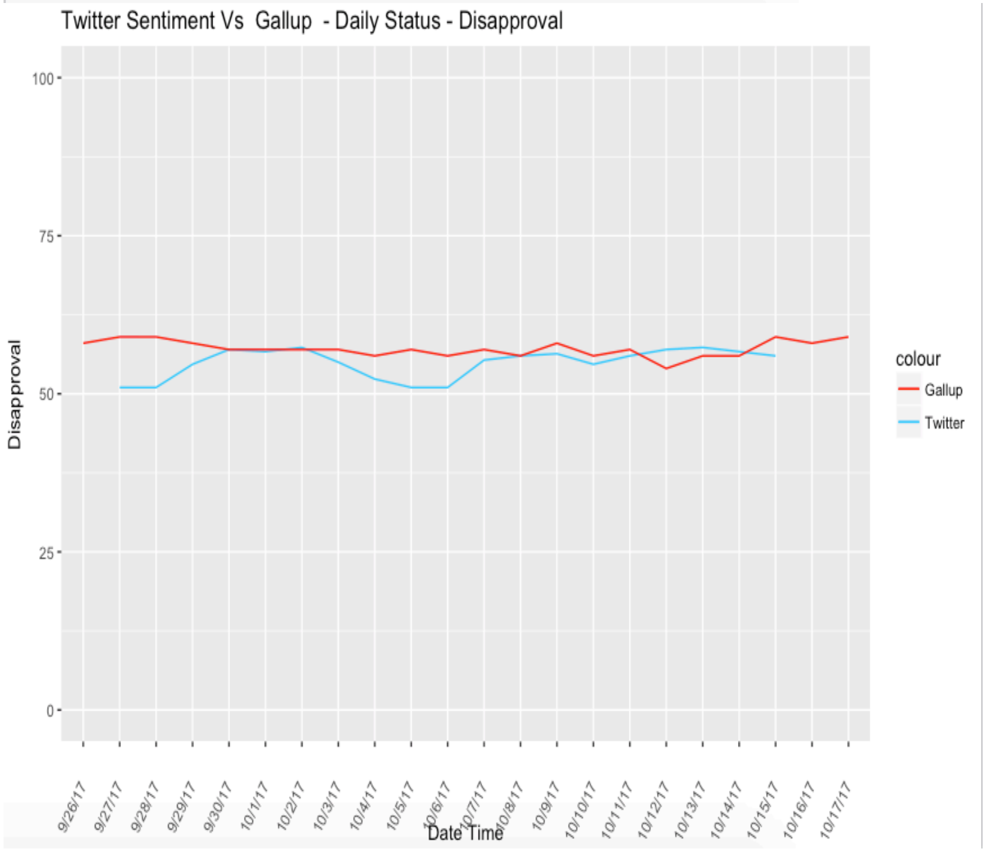
You can notice from the twitter sentiments that there is a downfall from OCT-04-2017 to OCT-06-2017. During this period the hurricane Maria shattered Puerto Rico without power supply , drinking water, food and shelter for 2 million Americans. Comments and counter arguments from the Republicans, public of Puerto Rico and San Juan, Mayor of Puerto Rico which created lots of negative sentiments against President Trump. This is reflecting very well in Twitter Sentiments as well as from Rasmussen Approval Rating Reports.

Conclusion: The Twitter Sentiments are reflecting President Job approval rating results well in advance and in real time.

7.2 Twitter Vs Gallup Reports

Figure 4





Twitter Sentiment Vs Gallup Reports

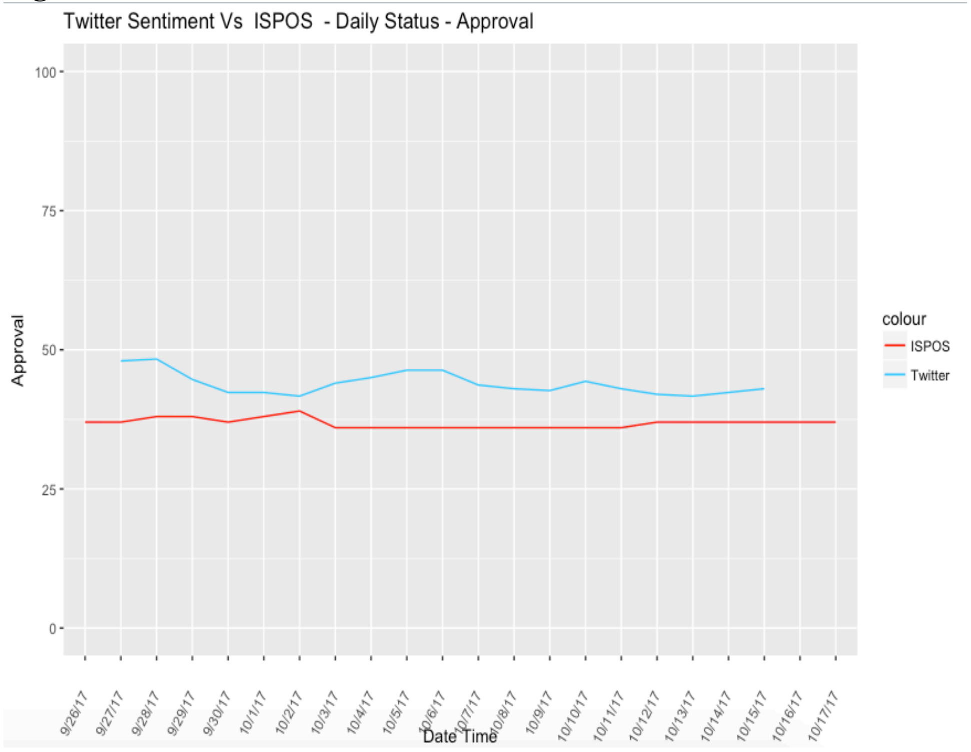
7.2.1 Analysis

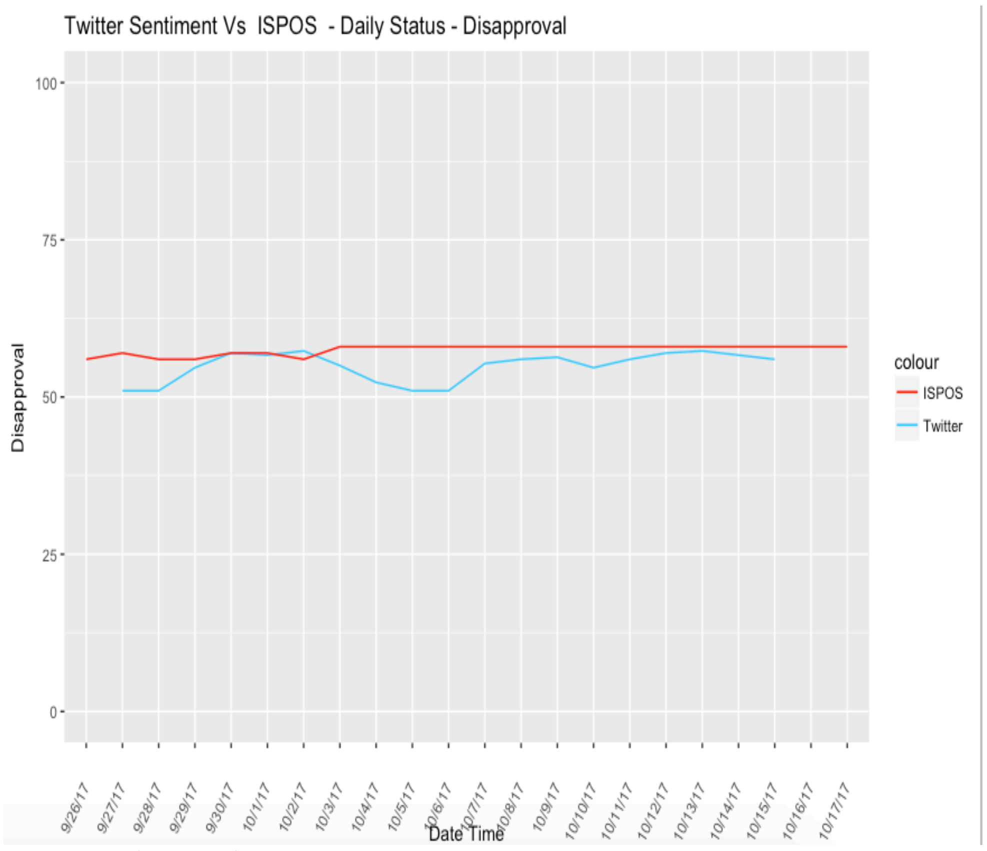
The Gallup approval rating results are at the low end when compared with Rasmussen Reports. But, while looking at the trend between Twitter Sentiments and Gallup Poll results, there is a positive correlation for high and low values though values have more differences.

The negative sentiments touches the common poll results at various points. We can notice that the negative poll results from Gallup matches with Rasmussen poll results also the trend coincides with Twitter sentiments on several dates.

7.3 Twitter Vs ISPOS Report

Figure 5





Twitter Sentiment Vs ISPOS Reports

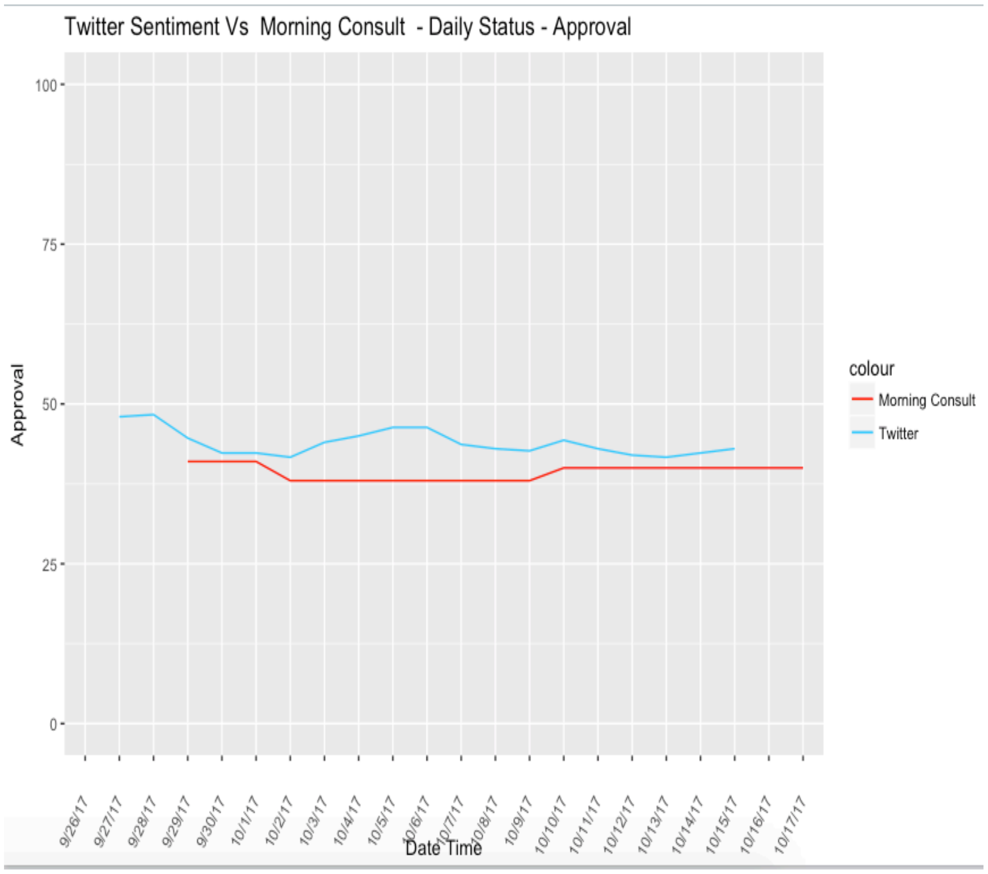
7.3.1 Analysis

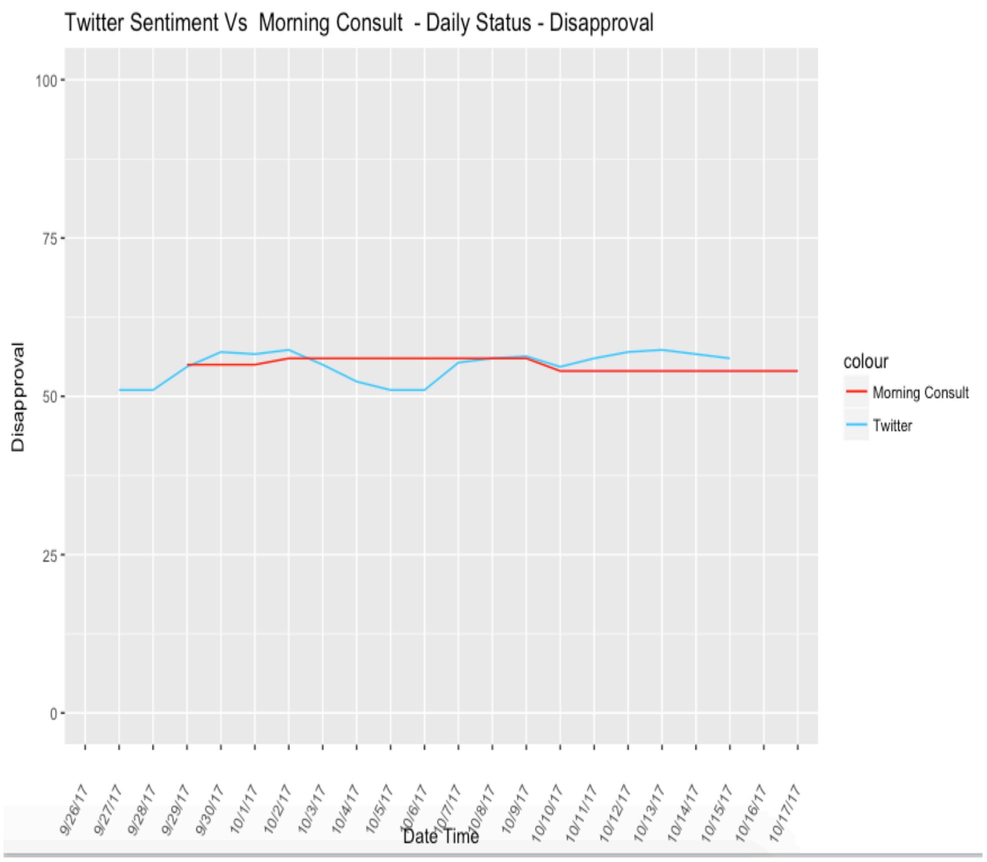
The ISPOS results do not maintain any positive correlation with Twitter sentiment results. Also, this trend does not match with Rasmussen Reports and Gallup Poll Results.

The ISPOS Poll results meets Twitter Sentiment results at several points for the Negative comments. This is another indication that there is a possibility to predict the President Job approval rating from the public sentiments in the Twitter. Also, we need to understand that there is a scope to influence President job approval ratings by making suitable statements and introducing the right policies for the welfare of the people.

7.4 Twitter Vs Morning Consult

Figure 6





Twitter Sentiment Vs Morning Consult Reports

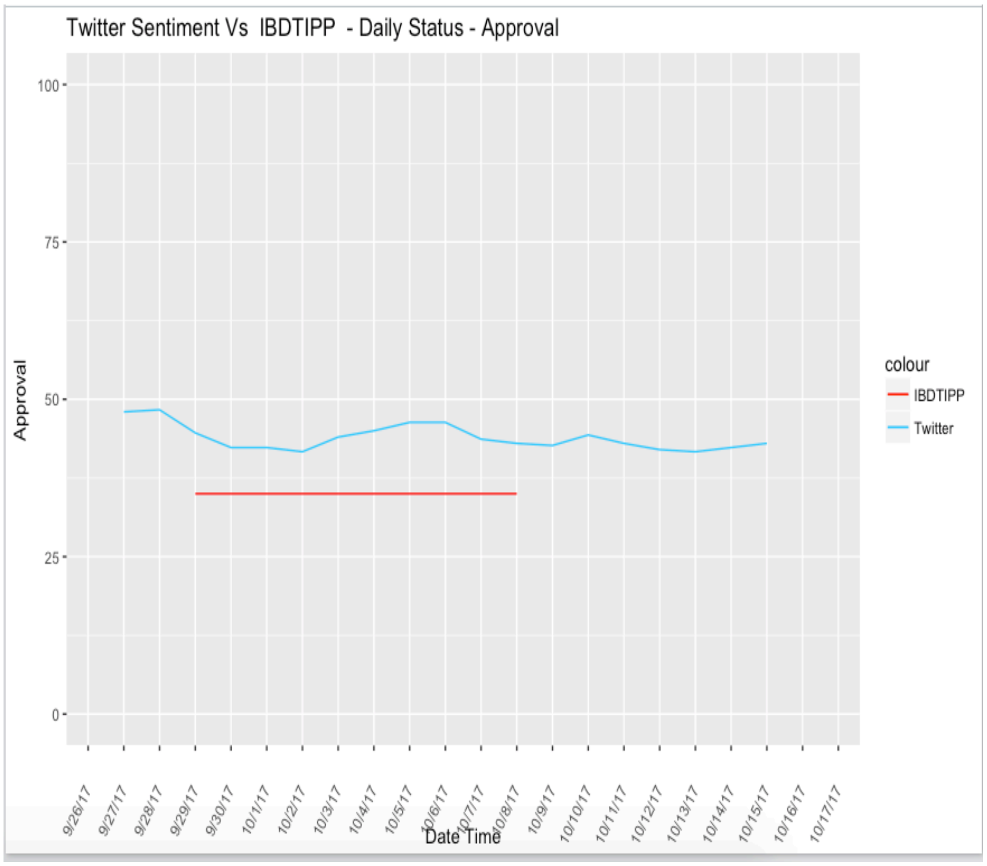
7.4.1 Analysis

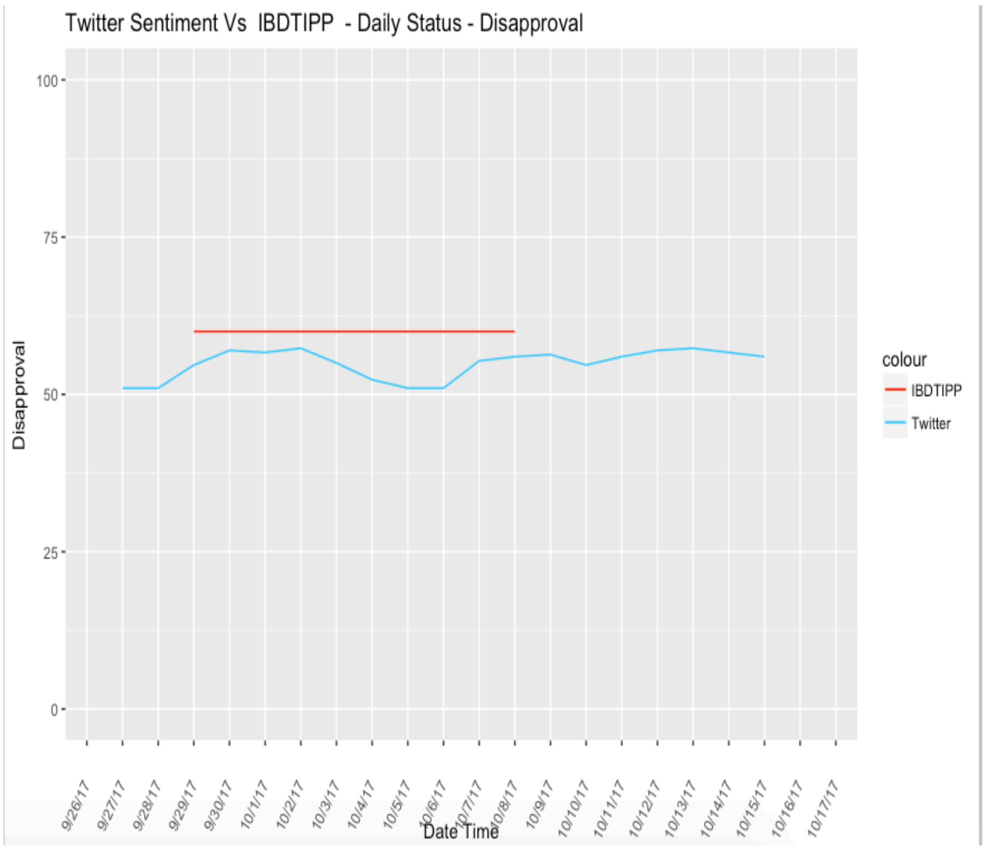
The poll results of Morning Consult meets the results of Twitter Sentiments only for 20% to 30%. There is a major difference between the Twitter Sentiment results and Corning Consult results. Also, we need to note that the 'Strongly Approve' values of ISPOS are comparatively less that other poll rating company's results.

The Negative sentiments of Twitter matches with the Disapproval rating values of Morning Consult for around 40%.

7.5 Twitter Vs IBD/TIPP

Figure 7





Twitter Sentiment Vs IBD/TIPP Reports

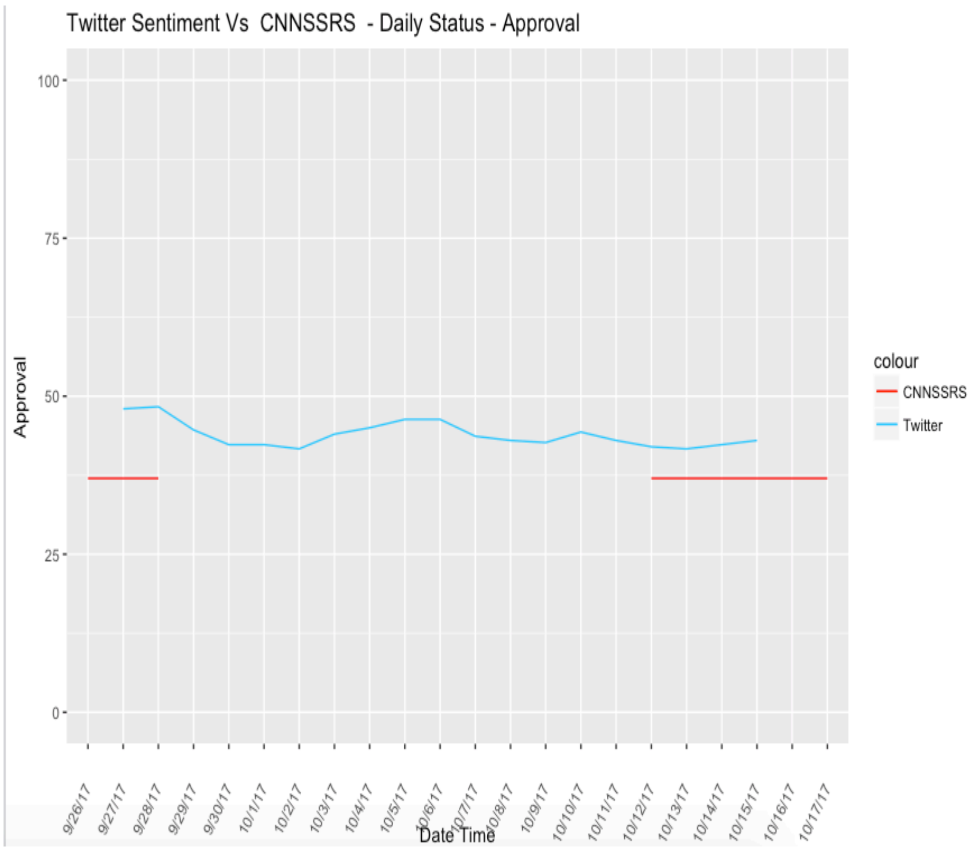
7.5.1 Analysis

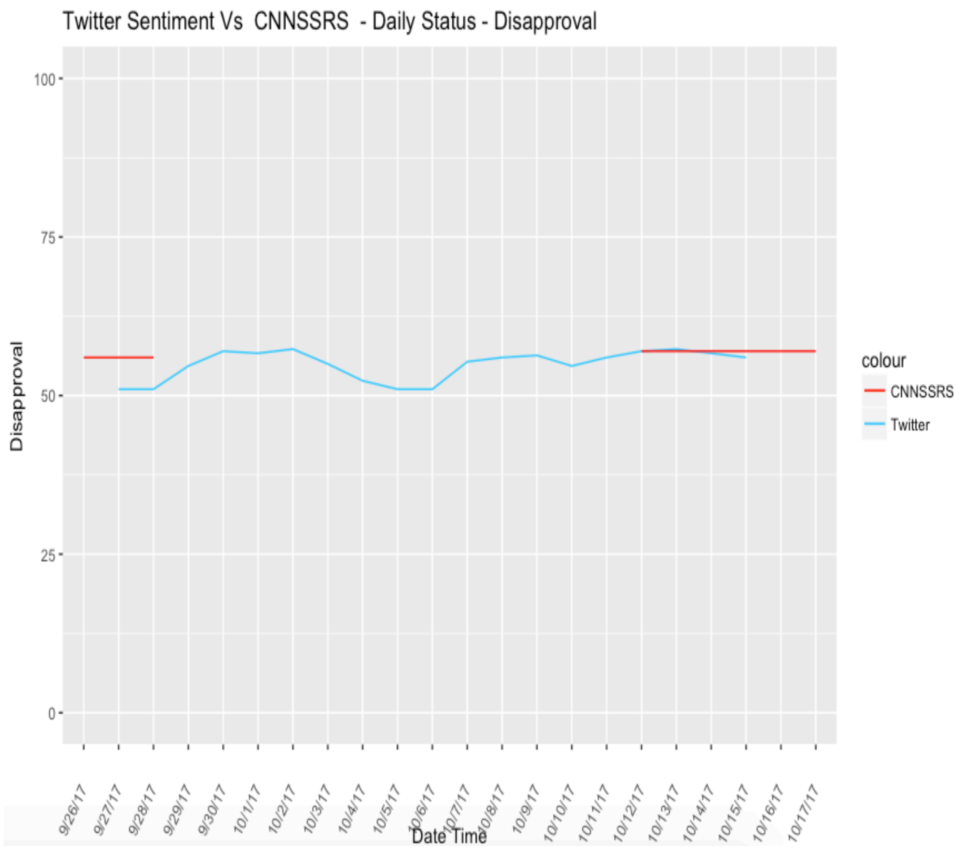
The data for IBD/TIPP was available only for few days. And also, the Approval rating results for positive side maintains a constant value from Sep-29-2017 to OCT-10-2017. Due to less number of data points from IBD/TIPP, we could not compare or contrast the results.

The data for IBD/TIPP was available only for few days. And also, the Approval rating results for positive side maintains a constant value from Sep-29-2017 to OCT-10-2017. Due to less number of data points from IBD/TIPP, we could not compare or contrast the results.

7.6 Twitter Vs CNN/SSRS

Figure 8





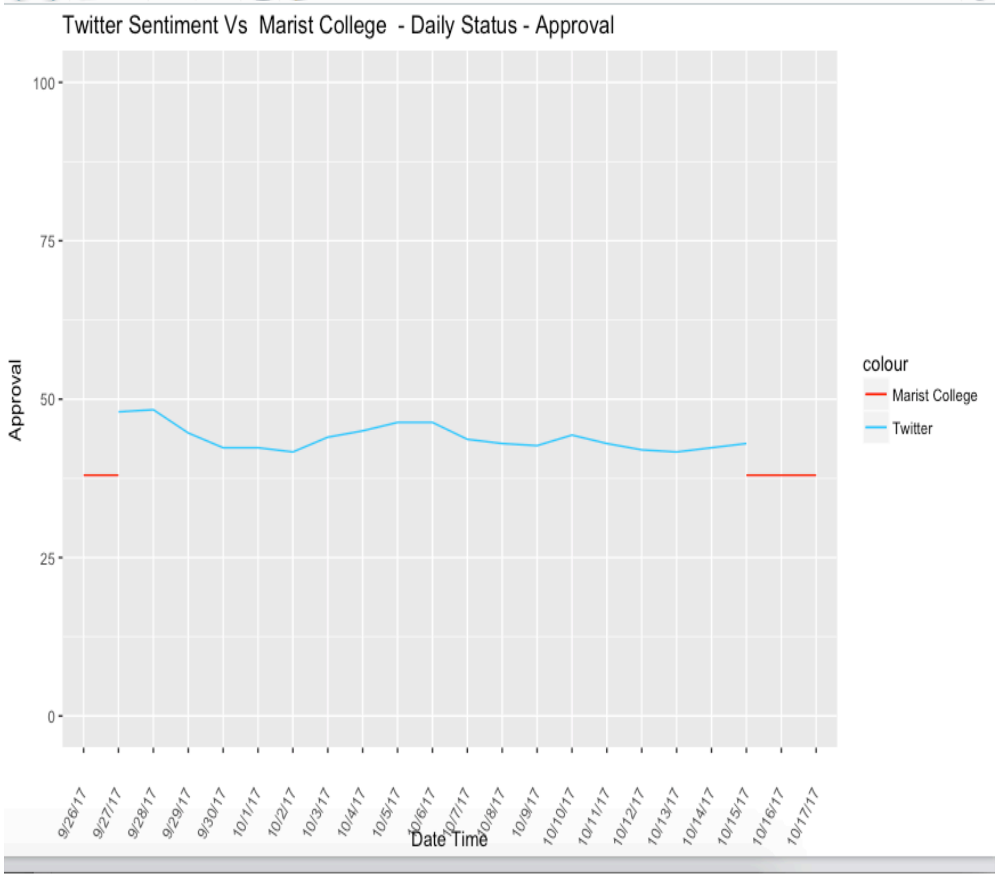
Twitter Sentiment Vs CNN/SSRS Reports

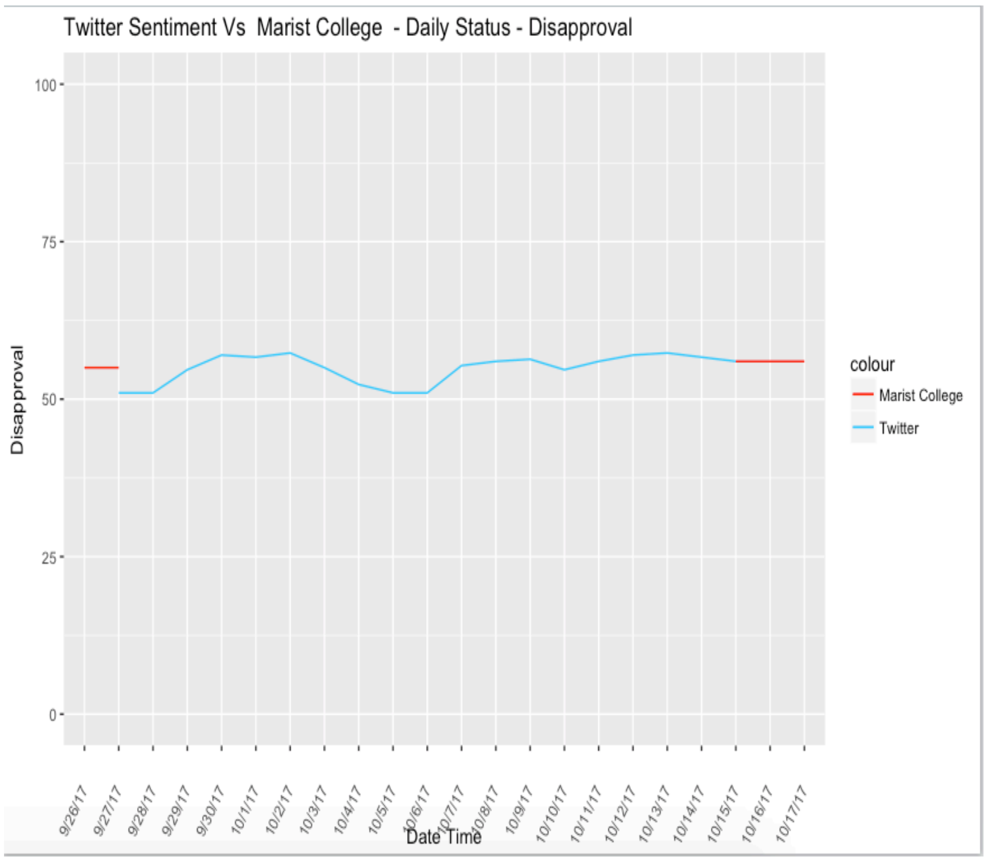
7.6.1 Analysis

The data available for CNN/SSRS poll results were less and we could not perform any analysis against CNN/SSRS poll results.

7.7 Twitter Vs Marist College

Figure 9





Twitter Sentiment Vs Marist College Reports

7.7.1 Analysis

The data available for Marist College poll results were less and we could not perform any analysis against Marist College poll results.

8. Discussion

The Naïve Bayes classifier has produced sentiment results almost equivalent to President job approval rating company's results. The trends and spikes in the graph for both approval rating point and sentiment factors are the reflection of some of the major events, criticism that happened in the last 3 weeks of time which are related to Hurricane Maria, attack on North Korea and tax reforms. The trend between approval rating points and Twitter Sentiments are almost trending in line with each other.

The Twitter Sentiment results are almost trending with some of the key Approval Rating Company's reports such as Rasmussen Reports, Gallup Reports, Morning Consult and ISPOS for Approval and Disapproval ratings This indicates that there is a scope to understand President Job approval rating in real time using Twitter Sentiments. Also, there is a possibility for the President and Republicans to influence the approval rating results by sending right signals to the public through social media.

9. References

1. http://www.gallup.com/interactives/185273/presidential-job-approval-center.aspx?g_source=WWWV7HP&g_medium=topic&g_campaign=tiles
2. http://machinelearningmastery.com/supervised-and-unsupervised-machine-learning-algorithms/
3. http://machinelearningmastery.com/supervised-and-unsupervised-machine-learning-
4. http://scikit-learn.org/stable/
5. https://www.quora.com/What-is-the-difference-between-supervised-and-unsupervised-learning-algorithms
6. https://research.googleblog.com/2016/06/wide-deep-learning-better-together-with.html
7. https://github.com/yogeshg/Twitter-Sentiment
8. www.gallup.com
9. http://fivethirtyeight.com/features/how-were-tracking-donald-trumps-approval-ratings/
10. https://insidebigdata.com/2013/09/12/beyond-volume-variety-velocity-issue-big-data-veracity/
11. https://www.coursera.org/learn/big-data-introduction/lecture/IIsZJ/characteristics-of-big-data-velocity
12. http://www.foxnews.com/official-polls/index.html

13. http://www.cnn.com/2017/04/26/politics/donald-trump-100-days-poll/index.html
14. http://www.investors.com/politics/ibdtipp-poll-presidential-approval-direction-of-country/
15. http://www.nbcnews.com/feature/data-points/
16. http://www.rasmussenreports.com/public_content/politics/trump_administration /prez_track_jun9
17. http://www.dataintensity.com/characteristics-of-big-data-part-one/
18. https://insidebigdata.com/2013/09/12/beyond-volume-variety-velocity-issue-big-data-veracity/
19. https://en.wikipedia.org/wiki/CNN
20. https://en.wikipedia.org/wiki/Opinion_Research_Corporation
21. https://en.wikipedia.org/wiki/Sentiment_analysis

10. Appendix

10.1 Scala Code for Data Collection from Twitter

FilterTweets.Scala

```
import java.util.regex.Pattern
import com.datastax.spark.connector._
import org.apache.log4j.Logger
import org.apache.log4j.Level
import org.apache.spark.sql._
import org.apache.spark.SparkContext._
import org.apache.spark.streaming._
import org.apache.spark.streaming.twitter._
import org.apache.spark.streaming.StreamingContext._
import org.apache.spark._
import scala.reflect.runtime.universe
import org.apache.spark.streaming.Seconds
import org.apache.spark.streaming.StreamingContext
import org.apache.spark.streaming.twitter._
import org.apache.spark.SparkConf



// This Spark-Scala program downloads tweets from Twitter into cassandra
data base.


object FilterTweets {
  def main(args: Array[String]) {

    val filter = Array("Trump","Donald Trump")

    import org.apache.log4j.{Level, Logger}

    val conf = new SparkConf()
    conf.set("spark.cassandra.connection.host", "127.0.0.1")
    conf.set("spark.cassandra.auth.username", "cassandra");
    conf.set("spark.cassandra.auth.password", "cassandra");
    conf.setMaster("local[*]")
    conf.setAppName("FilterTweets")

    val Flag = "TRUE"
    // Configure Twitter credentials using twitter.txt
    setupTwitter()

    val ssc = new StreamingContext("local[*]", "FilterTweets",
Seconds(2))
    setupLogging()


    val stream = TwitterUtils.createStream(ssc, None, filter)


    //  val tweets = stream.map(r => r.getText)
    val twrec = stream.map(status =>
      (   scala.util.Try(status.getId()) getOrElse "error",
        scala.util.Try(status.getCreatedAt()) getOrElse "error",
        scala.util.Try(status.getGeoLocation()) getOrElse "error",

scala.util.Try(status.getGeoLocation).map(_.getLatitude()).getOrElse(""),
"error",

scala.util.Try(status.getGeoLocation).map(_.getLongitude()).getOrElse("")
, "error",
        scala.util.Try(status.getPlace()) getOrElse "error",
        scala.util.Try(status.getText()) getOrElse "error",
        scala.util.Try(status.getUser.getName()) getOrElse "error",
        scala.util.Try(status.getLang()) getOrElse "error",
        scala.util.Try(status.getUser.getLocation()) getOrElse "error",
        scala.util.Try(status.getUser.getTimeZone()) getOrElse "error",
        scala.util.Try(status.getSource()) getOrElse "error"


    ))

    twrec.foreachRDD((rdd, time) => {
      rdd.cache()

      println("Writing " + rdd.count() + " rows to Cassandra")
      rdd.saveToCassandra("twitterdb2", "approvalrating",
SomeColumns("id", "createdat", "geolocation", "lattitude", "longitude",
"place", "userlanguage","source" ,"tweettext", "username","usertimezone"
,"userlocation"))

    })

    ssc.start()
    ssc.awaitTermination()
  }

  def setupLogging() = {

    val rootLogger = Logger.getRootLogger()
   rootLogger.setLevel(Level.WARN)


  }
  /** Configures Twitter service credentials using twiter.txt in the main
workspace directory */
  def setupTwitter() = {
    import scala.io.Source

    for (line <- Source.fromFile("credentials.txt").getLines) {
      val fields = line.split("=")
      if (fields.length == 2) {
        println("fields : " + fields(0), fields(1))
        System.setProperty("twitter4j.oauth." + fields(0), fields(1))
      }
    }
  }

  /** Retrieves a regex Pattern for parsing Apache access logs. */
  def apacheLogPattern():Pattern = {
    val ddd = "\\d{1,3}"
    val ip = s"($ddd\\.$ddd\\.$ddd\\.$ddd)?"
    val client = "(\\S+)"
    val user = "(\\S+)"
    val dateTime = "(\\[.+?\\])"
    val request = "\"(.*?)\""
    val status = "(\\d{3})"
    val bytes = "(\\S+)"
    val referer = "\"(.*?)\""
    val agent = "\"(.*?)\""
    val regex = s"$ip $client $user $dateTime $request $status $bytes
$referer $agent"
    Pattern.compile(regex)
  }

}
```

10.2 Python program for performing Sentiment Analysis using Naïve Bayes Classifier

TWClassifer.py

```
import collections
  import nltk.metrics

  import collections
  import nltk.metrics
  import nltk
  from nltk.classify import NaiveBayesClassifier
  from nltk.corpus import movie_reviews
  import nltk

  from nltk.probability import *
  import collections
  from numpy import genfromtxt
  import pandas as pd
  import itertools
  import numpy
  import pandas as pd
  import csv,random
  import numpy as np
  import pandas as pd
  import re
  from nltk.corpus import stopwords
  #import tweet_features, tweet_pca
  import pickle
  import itertools
  from nltk.collocations import BigramCollocationFinder
  from nltk.metrics import BigramAssocMeasures
  from nltk.sentiment.vader import SentimentIntensityAnalyzer as SIA
  nltk.download('stopwords')

  # Bigram finder
  def bigram_word_feats(words, score_fn=BigramAssocMeasures.chi_sq, n=200):
      bigram_finder = BigramCollocationFinder.from_words(words)
      bigrams = bigram_finder.nbest(score_fn, n)
      return dict([(ngram, True) for ngram in itertools.chain(words,
  bigrams)])


  #-------------------------------------------------------------------------
  ----
  # This function is used for preparing classifier.
  #  Hence, it addresses the following data cleansing requriements.
  #   1. Remove the URLs from the twitter text
  #   2. Remove the words that start with @. This is a kind of key word
  hence removed.
  #   3. Remove the words that start with #Tags
  #   4. Remove multiple white spaces
  #   5. Remove & quot that appear in the text
  #   6. Remove The key word 'RT :' which appears for tweet text.
  #   7. Remove any numbers from the text
  #   8. Remove non-english letters
  #   9. Remove the special characters ,'|', ',', '*', '#',
  #   10. Remove english stop words such as this, that  , and etc.
  #   11. Remove repeating characters.
  #   12. Convert tweets to lowercase
  #   13. Consider Verb, Adjectives and Adverbs as they qualify the
  statements.

  def parseTweets(TwFilename):

      tweets = []
      print(TwFilename)
      with open(TwFilename, 'rb') as csvfile:
          spamreader = csv.reader(csvfile, delimiter=',')
          #spamreader = csv.reader(open(TwFilename, 'rU'),csvfile,
  delimiter=',')
          for row in spamreader:
             # print row[0]
              tweet =  row[0]
              #print(row[1])

              # Remove URLs
              tweet = re.sub(r"http\S+", "", tweet)

              # Remove words starting with @
              #tweet = re.sub(r'(\s)@\w+', r'\1', tweet)
              tweet = re.sub(r'@([^\s]+)', r'\1', tweet)

              # Remove  #
              tweet = re.sub(r'(\s)#\w+', r'\1',tweet)

              # Remove multiple white spaces
              tweet = re.sub('[\s]+', ' ', tweet)

              # Remove  & quot;
              tweet = re.sub(r"&\S+", "", tweet)

              # Remove  & quot;
              tweet = re.sub(r"RT :\S+", "", tweet)

              tweet = tweet.replace("RT : ", '')

              # remove numbers
              tweet = re.sub("\d+", " ", tweet)

              #tweet = bigram_word_feats(tweet)

              # Remove non-English Characters
              tweet = re.sub(r'[^\x00-\x7F]+', ' ', tweet)

              # Remove | symbol
              tweet = tweet.replace('|', '')
              # Remove comma
              tweet = tweet.replace(',', '')

             # Remove !
              tweet = tweet.replace('!', '')

             # Remove *
              tweet = tweet.replace('*', '')

             # Remove #
              tweet = tweet.replace('#', '')

              # trim
              tweet = tweet.strip('\'"')

              # Remove stop words
              pattern = re.compile(r'\b(' +
  r'|'.join(stopwords.words('english')) + r')\b\s*')
              tweet = pattern.sub('', tweet)

              # remove repeating characters
              tweet = reduce(lambda x, y: x + y if x[-2:] != y * 2 else x,
  tweet, "")

              # convert to lower case
              words = tweet.lower()

              tagged_sentence = nltk.tag.pos_tag(words.split())
              splwords = []
              for word, tag in tagged_sentence:
                 if tag == 'VBP':
                     splwords.append(word)

              for word, tag in tagged_sentence:
                  if tag == 'RB':
                      splwords.append(word)

              for word, tag in tagged_sentence:
                  if tag == 'JJ':
                      splwords.append(word)

              words = ' '.join(splwords)
              words_filtered = [e.lower() for e in str(words).split() if
  len(e) >= 3]
              tweets.append((words_filtered, row[1]))

      return(tweets)

  #------------------------------------------------------------------------
  --------
  # labelTweets() function is used for laveling the tweets as positive and
  negative.
  # It classify the tweets to positive and negative using Naive Bayes
  Classifier.
  # Also, it uses  NLTK sentiment analyzer to findout the sentiment
  polarity of the tweets for
  # positive, negativem neutral and compund.

  def lableTweets(data_in,data_out):
      totaltweets = 0
      poscnt = 0
      negcnt = 0
      neucnt = 0

      fd = open(data_out, 'a')
      fd.write("Original Tweet,Transformed Tweet,Negative,Neutral,Positive,
  Compound,Sentiment Result,Geo Location,City,State")
      fd.close()

      with open(data_in, 'rb') as csvfile:
          spamreader = csv.reader(csvfile, delimiter=',')
          for row in spamreader:

              original_tweet=""
              tweet=""

              original_tweet = row[2]
              tweet = row[2]

              citystate = row[12]
              geoloc = row[13]
              # Remove comma from original tweet
              original_tweet = original_tweet.replace(',', '')

              # Remove URLs
              tweet = re.sub(r"http\S+", "", tweet)

              # Remove words starting with @
              #tweet = re.sub(r'(\s)@\w+', r'\1',tweet)
              tweet = re.sub(r'#([^\s]+)', r'\1', tweet)

              # Remove words starting with #
              tweet = re.sub(r'(\s)#\w+', r'\1',tweet)

              # Remove multiple white spaces
              tweet = re.sub('[\s]+', ' ', tweet)

              # Remove  & quot;
              tweet = re.sub(r"&\S+", "", tweet)

              # Remove  & quot;
              tweet = re.sub(r"RT :\S+", "", tweet)

              tweet = tweet.replace("RT : ", '')

              # remove numbers
              tweet = re.sub("\d+", " ", tweet)

              # Remove non-English Characters
              tweet = re.sub(r'[^\x00-\x7F]+', ' ', tweet)

              # Remove | symbol
              tweet = tweet.replace('|', '')

              # Remove comma
              tweet = tweet.replace(',', '')

              # trim
              tweet = tweet.strip('\'"')

              # Remove stop words
              pattern = re.compile(r'\b(' +
  r'|'.join(stopwords.words('english')) + r')\b\s*')
              tweet = pattern.sub('', tweet)

              # convert to lower case
              tweet = tweet.lower()

              label = ""
              label =
  (classifier.classify(extract_features(tweet.split())))

              sid = SIA()

              ss = sid.polarity_scores(original_tweet)
              result1 =""
              for k in ss:
                  result1= result1 + ('{0}, '.format(ss[k]))
              totaltweets = totaltweets + 1
              if (label == 'positive') :
                  poscnt = poscnt + 1
              elif (label == 'negative') :
                  negcnt = negcnt + 1
              else:
                  neucnt = neucnt + 1
                  result=""
              result = original_tweet +  ","  + tweet +  ","  + result1   +
  label +"," + geoloc +  ","  + citystate
              ""
              fd = open(data_out, 'a')
              fd.write(result + "\n")
              fd.close()

      fd = open(data_out, 'a')
      pos_per = (poscnt / totaltweets) * 100
      neg_per = (negcnt / totaltweets) * 100
      neu_per = (neucnt / totaltweets) * 100
      result=""
      fd.write("Summary\n")
      result = "totTweets,poscnt , negcnt, neucnt, pos_per, neg_per,
  neu_per\n"
      fd.write(result)
      result = '%s,%s,%s,%s,%s,%s,%s\n' % (totaltweets,poscnt,
  negcnt,neucnt,pos_per, neg_per,neu_per)
      fd.write(result)

      fd.close()

  def removearticles(text):
    re.sub('\s+(a|an|and|the)(\s+)', '\2', text)


  def get_words_in_tweets(tweets):
     all_words = []
     for (words, sentiment) in tweets:
         all_words.extend(words)
     return all_words


  def get_word_features(wordlist):
     wordlist = nltk.FreqDist(wordlist)
     word_features = wordlist.keys()
     return word_features
```

```
def extract_features(document):

   document_words = set(document)
   features = {}

   for word in word_features:
       features['contains(%s)' % word] = (word in document_words)
   return features


# This is the input training set used for preparing the classifer.
TweetsDataSet =
"/Users/hariharsubramanian/PycharmProjects/TrainingData/TWeetDataSetSep26
.csv"

print(len(TweetsDataSet))

tweets = parseTweets(TweetsDataSet)

word_features = get_word_features(get_words_in_tweets(tweets))

random.shuffle( tweets );

# the folloign lines are not required for second execution
#-----------------------------------------------------------
training_set = nltk.classify.apply_features(extract_features, tweets)
#classifier = nltk.NaiveBayesClassifier.train(training_set)

# The generated classifer is used saved into pickle file since generating
them for
# every iteration is really a time consuming process and slows down the
program execution.

# The following lines are used only once while creating the classifier
for the first time.
# Hence commented out.
#save_classifier =
open("/Users/hariharsubramanian/PycharmProjects/TwitterOutputs/naivebayes
.pickle","wb")
#pickle.dump(classifier, save_classifier)
#save_classifier.close()
#-----------------------------------------------------------


# Open classifier to generate sentiment results for the next set of test
data.
classifier_f =
open("/Users/hariharsubramanian/PycharmProjects/TwitterOutputs/naivebayes
.pickle", "rb")
classifier = pickle.load(classifier_f)
classifier_f.close()


#-----------------------------------------------------------

# Try with a sample tweet as part of testing to see if they are
classified correctly.
tweet = 'Larry is my friend'
label = (classifier.classify(extract_features(tweet.split())))
print (tweet)
print (label)

tweet = 'Larry is my enemy'
label = (classifier.classify(extract_features(tweet.split())))
print (tweet)
print (label)

# Once the above test is running correctly, try with the new test data
for
# finding sentiments for each tweet.
data_in =
"/Users/hariharsubramanian/PycharmProjects/TwitterInputs/Oct17.csv"
data_out =
"/Users/hariharsubramanian/PycharmProjects/TwitterOutputs/Oct17_out.csv"
print("Processing : " + data_in)
lableTweets(data_in,data_out)
print("************************** "+ data_out)
print("**************************  END *************************")
```

10.3 R Program to generate analytical reports

GenerateReports.R

# This R program generates the trend analysis charts for positive and negative sentiments # of Twitter against the Approval and Dis-approval rating results of poll companies.

```
library(ggplot2)



# Read Report Data

reportdata =
  "/Users/hariharsubramanian/Documents/MyProgs/AnalytixShinyDashboard/Presi
  dentApprovalRatingAnalysis/PollDataModified_V1.csv"

reportf = read.csv(reportdata,header=TRUE,sep=",")

reportdf <- as.data.frame((reportf))

head(reportdf)

str(reportdf)

summary(reportdf)

pollingreportname <- 'Marist College'
```

```
# Generate Positive comparative analysis Report for approval rating
  against positive Twitter sentiments


GeneratеARLineGraphPositiveDatewise(reportdf,pollingreportname)


# Generate Positive comparitive analysis Report for dis-approval rating
  against negative Twitter sentiments

GenerateARLineGraphNegativeDatewise(reportdf,pollingreportname)


# This rfunction generate the comparitive analysis for twitter positive

# sentiments and 'Approved' ratings.

GenerateARLineGraphPositiveDatewise <- function(reportdf,
  pollingreportname) {


  if (pollingreportname == 'Ramussen Reports') {

    rdf <- subset(reportdf , select = c(Date,rrpositive,  rrnegative ))

    rdf$source = "Rasmussen Reports"

  }

  else if (pollingreportname == 'Gallup') {

    rdf <- subset(reportdf , select = c(Date,gpositive,  gnegative ))

    rdf$source = "Gallup"

  }

  else if (pollingreportname == 'ISPOS') {

    rdf <- subset(reportdf , select = c(Date,ispositive,  isnegative ))

    rdf$source = "ISPOS"

  }

  else if (pollingreportname == 'Morning Consult') {

    rdf <- subset(reportdf , select = c(Date,mcpositive,  mcnegative ))

    rdf$source = "Morning Consult"

  }
```

```
else if (pollingreportname == 'IBDTIPP') {
  rdf <- subset(reportdf , select = c(Date,itpositive,  itnegative ))
  rdf$source = "IBD/TIPP"
}
else if (pollingreportname == 'CNNSSRS') {
  rdf <- subset(reportdf , select = c(Date,cspositive,  csnegative ))
  rdf$source = "Rasmussen Reports"
}
else if (pollingreportname == 'Marist College') {
  rdf <- subset(reportdf , select = c(Date,mrpositive,  mrnegative ))
  rdf$source = "CNN/SSRS"
}
tdf <- subset(reportdf , select = c(Date,twpositive,  twnegative ))

tdf$source = "Twitter"

colnames(tdf) <- c("Date","Positive", "Negative","Source")
colnames(rdf) <- c("Date","Positive", "Negative","Source")
data1 <- ""
data1 = merge(tdf, rdf, by="Date", all=F, sort=F)

#data1 <- head(data1,24)
DateTime <- data1$Date
Positive <- data1$Positive.x
RPositive <- data1$Positive.y

data1 <- data.frame(DateTime,  Positive ,RPositive)

dates <- unique(sort(data1$DateTime))
```

```
  data1$DateTime <- factor(data1$DateTime, labels = dates,  ordered = T)



  p <- ggplot(data1, aes(x=DateTime, y=Positive,group=1 )) +

    geom_line(aes(y = Positive, colour = "Twitter")) +

    geom_line(aes(y = RPositive, colour = pollingreportname)) +

    labs(x = "Date Time", y = "Positive Sentiments",

         title = paste("Twitter Sentiment Vs " , pollingreportname, " -
  Daily Status - Positive", paste=" " )) +

  scale_color_manual(values=c( "#FF0000", "#33CCFF")) +

    scale_x_discrete(limits=data1$DateTime) +

    theme(axis.text.x=element_text(angle=60,hjust=1,vjust=0.5))

  p


}
```

# This rfunction generate the comparative analysis for twitter negative sentiments and 'Disapproved' ratings.

GenerateARLineGraphNegativeDatewise <- function(reportdf, pollingreportname) {

```
 if (pollingreportname == 'Ramussen Reports') {

    rdf <- subset(reportdf , select = c(Date,rrpositive,  rrnegative ))

    rdf$source = "Rasmussen Reports"

  }

  else if (pollingreportname == 'Gallup') {

    rdf <- subset(reportdf , select = c(Date,gpositive,  gnegative ))

    rdf$source = "Gallup"

  }

  else if (pollingreportname == 'ISPOS') {

    rdf <- subset(reportdf , select = c(Date,ispositive,  isnegative ))
    rdf$source = "ISPOS"

  }

  else if (pollingreportname == 'Morning Consult') {

    rdf <- subset(reportdf , select = c(Date,mcpositive,  mcnegative ))

    rdf$source = "Morning Consult"

  }

  else if (pollingreportname == 'IBDTIPP') {

    rdf <- subset(reportdf , select = c(Date,itpositive,  itnegative ))

    rdf$source = "IBD/TIPP"

  }

  else if (pollingreportname == 'CNNSSRS') {

    rdf <- subset(reportdf , select = c(Date,cspositive,  csnegative ))

    rdf$source = "Rasmussen Reports"

  }

  else if (pollingreportname == 'Marist College') {

    rdf <- subset(reportdf , select = c(Date,mrpositive,  mrnegative ))

    rdf$source = "CNN/SSRS"

  }

  tdf <- subset(reportdf , select = c(Date,twpositive,  twnegative ))


  tdf$source = "Twitter"


  colnames(tdf) <- c("Date","Positive", "Negative","Source")

  colnames(rdf) <- c("Date","Positive", "Negative","Source")

  data1 <- ""

  data1 = merge(tdf, rdf, by="Date", all=F, sort=F)


  #data1 <- head(data1,24)

  DateTime <- data1$Date
```

```
  Negative <- data1$Negative.x

  RNegative <- data1$Negative.y


  data1 <- data.frame(DateTime,  Negative ,RNegative)


  dates <- unique(sort(data1$DateTime))

  data1$DateTime <- factor(data1$DateTime, labels = dates,  ordered = T)




  p <- ggplot(data1, aes(x=DateTime, y=Negative,group=1 )) +

    geom_line(aes(y = Negative, colour = "Twitter")) +

    geom_line(aes(y = RNegative, colour = pollingreportname)) +

    labs(x = "Date Time", y = "Negative Sentiments",

         title = paste("Twitter Sentiment Vs " , pollingreportname, " -
  Daily Status - Negative", paste=" " )) +

    scale_color_manual(values=c( "#FF0000", "#33CCFF")) +

    scale_x_discrete(limits=data1$DateTime) +

    theme(axis.text.x=element_text(angle=60,hjust=1,vjust=0.5))

  p


}
```