Predicting the 2012 Reprublican Primary Results using Twitter Sentiment Analysis

Baskin, Sean
xpq514@mocs.utc.edu

Reinsmidt, Eric
bgw326@mocs.utc.edu

Nunn, Jonathan
xbb829@mocs.utc.edu

Department of Engineering and Computer Science,
University of Tennessee at Chattanooga

April 23, 2012

Abstract

Public sentiment analysis represents a critical component to individuals and organizations engaged in the United States political process, and the source of this data has become ever more prominent. Data mining of the social-media micro-blogging service Twitter is used to produce a substantial dataset for public sentiment analysis of Republican presidential primary candidates to investigate the relationship between sentiment scores and the number of delegates won. As a result of the large data employeed in this study, Apache Hadoop and MapReduce are utilized in aggregrating sentiment scores for different candidates and primaries. The results of our study indicate that simple linear regression is promising but does not fit every data set it is applied, possibly requiring additional explanatory variables.

Contents

1 Introduction

1.1 Overview and Rational

The microblogging website Twitter has blossomed into a social media power house with over ninety-two million users utilizing its service per month in the United States [2]. The authors of this study wanted to investigate whether Twitter was being utilized as a forum for discussion about politics, and if so what influence did the content of user's tweets have on the outcome of the 2012 Republican presidential primaries. Specifically, we propose a novel approach to predicting the outcome of the Republican presidential primary that postulates the sentiment of each tweet directly influences the number of delegates won - represented as a percentage - by each candidate in each primary election.

Our approach deviates from previous methodologies [9],[5] in that our model produces a net sentiment score for a duration of time for each caucus. The time frame is defined as the time period from the previous primary caucus to the primary caucus who's score is being calculated. For days which multiple primary caucuses fall, an average winning delegate percentage is calculated and assigned to the resultant net sentiment score. The sentiment scoring algorithm is based in part on the technique presented by Liu[6]; each tweet is tokenized into a collection of words, each word is compared to a dictionary of positive and negative words and assigned a score, and the overall tweet sentiment score is the difference between the sum of positive scores and the sum of negative scores.

1.2 Linear Model

The study uses an Ordinary Least Squares (OLS) regression model of the form

$$\hat{Y}_i = \hat{\beta}_0 + \hat{\beta}_1 X_{1i} + \ldots + \hat{\beta}_k X_{ki} \tag{1}$$

specifically a simple linear regression model expressed as by the following form

$$Y_i = \alpha + \beta X_i + \epsilon_i \tag{2}$$

where Y_i is the value of the dependent random variable, X_i is the value of the ith level of the fixed variable X, and ϵ is the random error or disturbance [3]. The model parameters are selected to minimize the sum of squared residuals to estimate the unknown parameter β by OLS. The estimate of β can be expressed in the following form by letting

$$S(a,b) = \sum_{i=1}^{n} (Y_i - a - bX_i)^2 \tag{3}$$

be the total sum of squared deviations from the regression line to the n points. The values of a and b that minimize S can be obtained by differentiating equation 3 with respect to a and b

$$
\begin{aligned}
\frac{\partial S}{\partial a} &= -2\sum_{i=1}^{n}(Y_i - a - bX_i) \\
\frac{\partial S}{\partial b} &= -2\sum_{i=1}^{n}(Y_i - a - bX_i)X_i
\end{aligned} \tag{4}
$$

Setting each derivative equal to zero and simplifying yields

$$
\begin{aligned}
\sum_{i=1}^{n}(Y_i - a - bX_i) &= 0 \\
\sum_{i=1}^{n}X_i(Y_i - a - bX_i) &= 0
\end{aligned} \tag{5}
$$

which leads to

$$\sum_{i=1}^{n}Y_i - na - b\sum_{i=1}^{n}X_i = 0 \tag{6}$$

$$\sum_{i=1}^{n}X_iY_i - a\sum_{i=1}^{n}X_i - b\sum_{i=1}^{n}X_i^2 = 0 \tag{7}$$

Dividing by n and simplifying again yields the solution for a

$$a = \frac{\sum_{i=1}^{n}Y_i}{n} - b\frac{\sum_{i=1}^{n}X_i}{n} \tag{8}$$

and by applying the identities $\bar{X} = \frac{\sum_{i=1}^{n}X_i}{n}$ and $\bar{Y} = \frac{\sum_{i=1}^{n}Y_i}{n}$ yields

$$a = \bar{Y} - b\bar{X} \tag{9}$$

To solve for b we substitute equation 9 into equation 7 and solve:

$$\sum_{i=1}^{n}X_iY_i - \left(\frac{\sum_{i=1}^{n}Y_i}{n} - b\frac{\sum_{i=1}^{n}X_i}{n}\right)\sum_{i=1}^{n}X_i - b\sum_{i=1}^{n}X_i^2 = 0 \tag{10}$$

multiplying by n and solving for b yields:

$$b = \frac{n\sum_{i=1}^{n}X_iY_i - \sum_{i=1}^{n}X_i\sum_{i=1}^{n}Y_i}{n\sum_{i=1}^{n}X_i^2 - \left(\sum_{i=1}^{n}X_i\right)^2} \tag{11}$$

The simple linear regression model also makes the following assumptions which must be satisfied with OLS estimation:

Linearity

$E\{Y_i|X\} = x_i\beta$, where x_i is the ith row of X;

Homoscedasticity
  $Var\{Y_i|X\} = \sigma^2$ for $i = 1, 2, \ldots, n$;

Uncorrelatedness
  $Cov\{Y_i, Y_j|X\} = 0$ where $i \neq j$; and

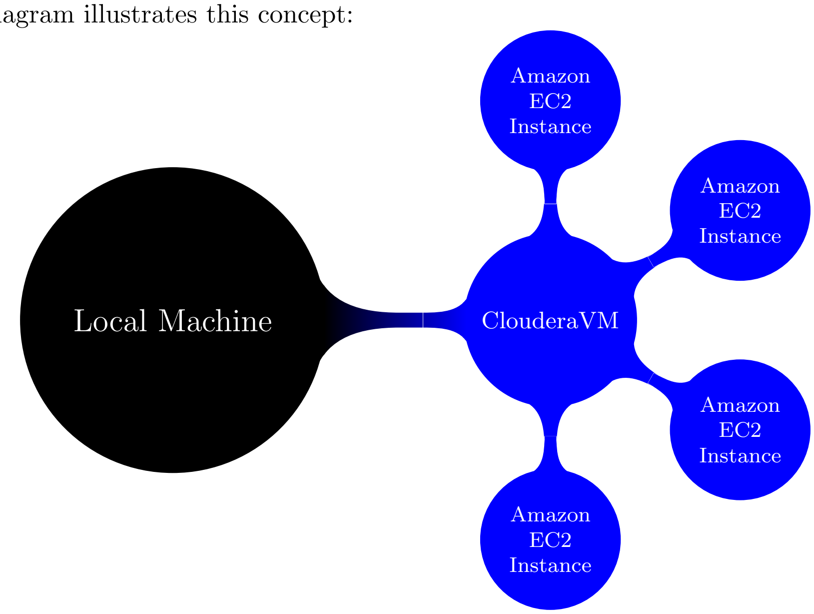
Normality
  $Y_i|X$ for $i = 1, 2, \ldots, n$ have normal distributions.

The model will be applied to each candidate for the same time periods and caucuses, where the response variable $\hat{Y}_i$ is the percentage of winning delegates at each Republican primary caucus and the predictor variables X is the net tweet sentiment score.

2 Methodology

2.1 Experiment Environment

The base for the experiment environment is a virtual machine supplied by Cloudera. The benefit of this system is that with some minor additions, it can be used to spin up Amazon Web Services Elastic Complete Cloud instances as needed to increase processing power for map reducing large data sets. The following diagram illustrates this concept:



2.1.1 Operating System

The operating system used in the virtual machine is a freely available community version of red hat server, CentOS 5.7 By default the virtual machine image is preset to use 1GB of memory and a single processor which proves to be insufficient even for small jobs. The virtual machine is easily improved for local jobs by increasing the memory size to 4GB and doubling the processors used.

2.1.2 Programming Languages and Software

PHP is the scripting language used for data mining. Topsys Otter API provides access to the social data that is indexed on Topsy.com Additionally, PHP is used to pre-process the mined data to remove noise from the data such as spam. The pre-processing is also used to remove unwanted formatting so that analysis in RStudio is streamlined.

The Cloudera virtual machine is preinstalled with CDH3, which is in essence hadoop 0.2.0 with improvements added by Cloudera to increase speed in Map Reducing data amongst other things.

While local file system Map Reducing is ok for smaller data sets, sometimes larger data sets will require more processing power and memory than the virtual machine can supply. To alleviate this restriction, Whirr 0.7.1 was installed on the virtual machine. Whirr is an Apache product, which allows a local machine to spin up Amazon Elastic Cloud Compute instances as desired. Whirr is easily configurable and passes data from the local file system to instances where Map Reducing is completed and the reduced data is passed back to the local machine whereupon the instances are terminated. Whirr also includes the ability to bid on spot instances, which can reduce cost of processing large data sets.

R is "a language and environment for statistical computing and graphics ... [and is] an open source solution to data analysis that's supported by a large and active worldwide research community" [4]. The company Revolution Analytics has produced R packages that interface with Hadoop's MapReduce, HDFS, and HBase [10] components with `rmr`, `rhdfs`, and `rhbase`, respectively.

2.2 MapReduce Algorithms and Implementation

The input formatter algorithm reads through each line of a document and emits a key-value pair where the key is `NULL` and the value is a vector containing <`candidate, primary, tweet`> attributes. Below is the implementation in R:

```
tweet.csvtextinputformat = make.input.format( format = function(line) {

  values = unlist( strsplit(line, "\\,") )

  names(values) = c('Candidate', 'Primary', 'Tweet')

  return( keyval(NULL, values) )
} )
```

The mapper algorithm takes the key-value pair emitted by the input formatter function as follows

function MAPPER(NULL, <candidate c, primary p, tweet t>)
    for all tweet $t \in documentD$ **do**

function SentimentScore(c, p, t)
    Tokenize tweets into words
    score each word ws against opinion lexicon
    sentiment score $s = \sum ws$
    EMIT(c, p, s)
end function
end for
end function

where the key-pair emitted takes the form <k,v>where k is a vector containing $<candidate, primary>$ and v is a vector containing $<tweetScore>$. Below is the implementation in R,

```
mapper.candidate.primary.tweet = function(key, value) {

  # Skip header lines
  if ( !identical(as.character(val['Candidate']), 'Candidate')
       & !identical(as.character(val['Primary']), 'Primary')
       & !identical(as.character(val['Tweet']), 'Tweet') ) {

    # Calculate the tweet sentiment score
    tweet.score <- score.sentiment(c(val['Tweet']), pos.words, neg.words)

    # The key consists of the candidate, primary
    output.key = c(val['Candidate'], val['Primary'])

    # The value consists of the sentiment score
    output.val = tweet.score

    return( keyval(output.key, output.val) )
  }
}
```

and the sentiment score function implementation:

```
score.sentiment = function(tweet, pos.words, neg.words, .progress='text')
{
  require(plyr)
  require(stringr)

  scores = (tweet, function(tweet, pos.words, neg.words) {

    # Clean tweet
    # Remove punctuation characters
    tweet = gsub('[[:punct:]]', '', tweet)
    # Remove control characters
    tweet = gsub('[[:cntrl:]]', '', tweet)
    # Remove digits
    tweet = gsub('\\d+', '', tweet)
    # Convert to lower-case
    tweet = tolower(tweet)

    # Tokenize into words
    word.list = str_split(tweet, '\\s+')
    # sometimes a list() is one level of hierarchy too much
    words = unlist(word.list)
```

```
    # Compare to the positive and negative lexicons
    pos.matches = match(words, pos.words)
    neg.matches = match(words, neg.words)

    # match() returns the position of the matched term or NA
    # we just want a TRUE/FALSE:
    pos.matches = !is.na(pos.matches)
    neg.matches = !is.na(neg.matches)

    # TRUE/FALSE will be treated as 1/0 by sum():
    score = sum(pos.matches) - sum(neg.matches)

    return(score)
  }, pos.words, neg.words, .progress = .progress )

  scores.df = data.frame(score=scores)
  return(scores.df)
}
```

The reducer function takes as input the key-value pair emitted from the mapper function and performs a reduce function that emits a key-value pair of the form $<(c, p), s>$

1: **function** REDUCER(candidate c, primary p, score s)
2: Sum the values corresponding to each key
3: EMIT(c, s)
4: **end function**

The final key-pair values are saved as a data frame in R and are consumed as input to the regression analysis portion of the study. The function implementation is below:

```
reducer.candidate.primary.sentscore = function(key, val.list) {

  # Change from a row vector to a column vector
  val.df = ldply(val.list, as.numeric)
  colnames(val.df) = c('net.tweet.score')

  output.key = key
  output.val = val.df

  return( keyval(output.key, output.val) )
}
```

Running the MapReduce job is very straight forward, requiring a minimal configuration of Hadoop itself. See below for a sample job:

```
library(rmr)

source('R/mapreduce.R')

# Set the input and output paths for the HDFS
hdfs.input.path = 'Analysis/data'
hdfs.output.root = 'Analysis/out'


mr.candidate.primary.sentscore = function (input, output) {
  mapreduce(input = input,
```

```
            output = output,
            input.format = tweet.csvtextinputformat,
            map = mapper.candidate.primary.tweet,
            reduce = reducer.candidate.primary.tweet,
            backend.parameters = list(
              hadoop = list(D = "mapred.reduce.tasks=10")
              ),
            verbose=T)
}

hdfs.output.path = file.path(hdfs.output.root, 'results')
results = mr.candidate.primary.sentscore(hdfs.input.path, hdfs.output.path)

results.df = from.dfs(results, to.data.frame=T)
colnames(results.df) = c('candidate', 'primary', 'net_sent_score')

save(results.df, file="out/candidate.RData")
```

3 Data Collection and Preprocessing

The core of any data analysis is the corpus of data itself. In respect to the Republican Primary prediction, a decision was made to use a popular source of social interaction to retrieve or data from, namely data from twitter. As with any data source that is pulled from the wild of the Internet, cleansing of data must be performed. This preprocessing of data is not only important to putting the data in a format that can be easily parsed, but to remove various pieces of unwanted noise from the data, e.g. spam.

3.1 Historical Twitter Data

While twitter itself is a good source for obtaining individual social comments, or tweets, the time constraint of the twitter search within its APIs stated limits is a week [1]. However the common consensus is that more often than not, data is not available if it is older than four days from the current date. Regardless if the time length availability of twitter data is four days or one week, neither fit the purpose of this study, which requires data from months prior to the current date. There are social media indexers that collect and make available historical twitter, and other social media, data. Topsy is the largest of these social media indexers. An easy to use API named Otter is provided for access to Topsys data records.

3.2 Data Mining with Otter

There are numerous ways to search using Otter, but of primary interest for this study is a query involving a beginning and ending time. In addition, for any particular query, a user of the API can retrieve at most 10 pages of data with 100 tweets per page. Because of this, if there are greater than 1,000 tweets in the time frame specified not all tweets will be available. Another available option for searching is the inclusion of a time window parameter that is passed to the API in seconds. An innovative solution is implemented that uses both starting and ending times while including a time window. With a sliding start time it is possible to tailor the time window to a small enough value so that they are less than 1,000 are always retrieved and hence all tweets can be mined. When the time window plus the start time exceed the end time, all of the tweets for an overall time frame will have been acquired.

```
// pseudo code for tweetminer.php

start time = 1/1/2012
end time = 4/1/2012
time window = 3600 seconds
query = Otter.API.request
corpus =

begin mining

if (query > 1000)
{
        time window = time window / 2
    restart mining
}

else
{
        tweets = query
        corpus = corpus + tweets

        if (start time >= end time)
        {
                end mining
        }

        else
        {
                start time = start time + time window
                restart mining
        }
}
```

It is of interest to note that for each tweet returned from an Otter query, corresponding timestamp data is added by the above pseudo code.

3.3 Data Preprocessing

Once data has been collected from Topsy using the Otter API, the data must be cleaned for multiple reasons. As previously mentioned, social media inherently will have spam, as it is a likely target. In addition, many tweets contain erroneous data such as external http/s links that are not germane to sentiment analysis of the publics opinion of Republican candidates. Additionally, after removing http/s links from some tweets, the size of the remaining data is insufficient to process. These tweets were removed the data set. As well, a small but significant number of tweets fall outside of the 140-character limit of twitter, indicating that the tweet was created outside of the bounds of the terms of service of twitter usually involving an exploit of some kind. Exploiting a security hole in this manner indicates that an individual is not representative of the general populace, and because of this, tweets outside the limit of acceptable use are removed.

```
// pseudo code for processor.php
// desired date is in format:
        timestamp,tweet content

corpus = data from tweetminr.php
semi processed corpus =
final corpus =

for corpus
{

        // remove erroneous line breaks
if (line break is not followed by a timestamp)
{
                line break = space
                semi processed corpus = corpus
}

}

for each line in semi processed corpus
{

// remove tweets longer than 140 characters
if (this line is > 140 characters)
{
        this line is deleted
}

if (this line is < 1 character)
{
        this line is deleted
}

if (this line contains > 1 comma)
{
        remove all commas beyond first comma
}

semi processed corpus = semi processed corpus + this line
}

// remove empty lines from the for each
for semi processed corpus
{

        if (line break is not followed by a timestamp)
{
                line break = space
                final corpus = semi processed corpus
```

}

}

The final data that is output from processing is in a format readable, i.e. a unix timestamp followed by a comma which is in turn followed by the processed content of the tweet. It is essential that comma is used as a delimiter for further processing in RStudio, which only accepts a short list of delimiters.[8]

4 Results and Discussion

The use of `R` introduces the ability to significantly decrease the amount of time invested in manual regression diagnostics through the use of the `gvlma` package [8]. The package utilizes the methodology described by Peña [7] for "globally testing the four assumptions of the linear model". For the sake of brevity we have included the summarized results in Appendix A and the graphical results in Appendix B. Overall the results demonstrated that our model satisified all of the assumptions required in all cases with the exception of Newt Gingrich.

While the results were promising for the other candidates, *we must reject that the proposed model is suitable as a predictor of the percentage of delegates won*. The authors speculate the reason the model failed may be directly related to Newt Gingrich's poor performance in many of the caucuses, in many cases obtaining zero delegates. The heavily number of zero percent outcomes negatively affected the distribution of percentage of winning delegates.

The authors suggest that in future work additional explanatory variables be included in the model such as the volume of tweets for a particular candidate over a given time period. The number of explanatory variables could be many, spanning various mediums of communication; however, the focus is to keep the study within the domain of the tweets themselves to investigate if this contemporary medium has its own inferential attribute.

5 Appendix

5.1 Appendix A - `gvlma` Results

This appendix contains the results from using the `gvlma` package using our linear model for each candidate.

Mitt Romney

```
Call:
lm(formula = Romney.Winning.Percentage ~ Romney.Score, data = master.agg.df)

Residuals:
     Min       1Q   Median       3Q      Max
```

```
-0.42127 -0.19390  0.03099  0.12533  0.47890

Coefficients:
               Estimate Std. Error t value Pr(>|t|)
(Intercept)   5.325e-01  1.215e-01   4.383  0.00074 ***
Romney.Score -1.870e-05  4.357e-05  -0.429  0.67477
---
Signif. codes:  0 *** 0.001 ** 0.01 * 0.05 . 0.1   1

Residual standard error: 0.2654 on 13 degrees of freedom
Multiple R-squared: 0.01398,Adjusted R-squared: -0.06187
F-statistic: 0.1843 on 1 and 13 DF,  p-value: 0.6748


ASSESSMENT OF THE LINEAR MODEL ASSUMPTIONS
USING THE GLOBAL TEST ON 4 DEGREES-OF-FREEDOM:
Level of Significance =  0.05

Call:
 gvlma(x = romney.lm)

                      Value p-value                Decision
Global Stat        0.660702  0.9561 Assumptions acceptable.
Skewness           0.003626  0.9520 Assumptions acceptable.
Kurtosis           0.288080  0.5915 Assumptions acceptable.
Link Function      0.029412  0.8638 Assumptions acceptable.
Heteroscedasticity 0.339585  0.5601 Assumptions acceptable.
```

Rick Santorum

```
Call:
lm(formula = Santorum.Winning.Percentage ~ Santorum.Score, data = master.agg.df)

Residuals:
     Min       1Q   Median       3Q      Max
-0.14673 -0.08233  0.01068  0.06866  0.16914

Coefficients:
                Estimate Std. Error t value Pr(>|t|)
(Intercept)    1.411e-01  2.817e-02   5.010 0.000239 ***
Santorum.Score 3.291e-06  9.782e-06   0.336 0.741960
---
Signif. codes:  0 *** 0.001 ** 0.01 * 0.05 . 0.1   1

Residual standard error: 0.1075 on 13 degrees of freedom
```

```
Multiple R-squared: 0.008628,Adjusted R-squared: -0.06763
F-statistic: 0.1131 on 1 and 13 DF,  p-value: 0.742


ASSESSMENT OF THE LINEAR MODEL ASSUMPTIONS
USING THE GLOBAL TEST ON 4 DEGREES-OF-FREEDOM:
Level of Significance =  0.05

Call:
 gvlma(x = santorum.lm)

                     Value p-value                Decision
Global Stat        2.38174  0.6659 Assumptions acceptable.
Skewness           0.04637  0.8295 Assumptions acceptable.
Kurtosis           0.68435  0.4081 Assumptions acceptable.
Link Function      0.03877  0.8439 Assumptions acceptable.
Heteroscedasticity 1.61225  0.2042 Assumptions acceptable.
```

Ron Paul

```
Call:
lm(formula = Paul.Winning.Percentage ~ Paul.Score, data = master.agg.df)

Residuals:
     Min       1Q   Median       3Q      Max
-0.11690 -0.08334 -0.02535  0.08688  0.20095

Coefficients:
              Estimate Std. Error t value Pr(>|t|)
(Intercept)  1.242e-01  3.993e-02   3.111  0.00827 **
Paul.Score  -1.373e-05  1.270e-05  -1.081  0.29948
---
Signif. codes:  0 *** 0.001 ** 0.01 * 0.05 . 0.1   1

Residual standard error: 0.107 on 13 degrees of freedom
Multiple R-squared: 0.08243,Adjusted R-squared: 0.01185
F-statistic: 1.168 on 1 and 13 DF,  p-value: 0.2995


ASSESSMENT OF THE LINEAR MODEL ASSUMPTIONS
USING THE GLOBAL TEST ON 4 DEGREES-OF-FREEDOM:
Level of Significance =  0.05

Call:
 gvlma(x = paul.lm)
```

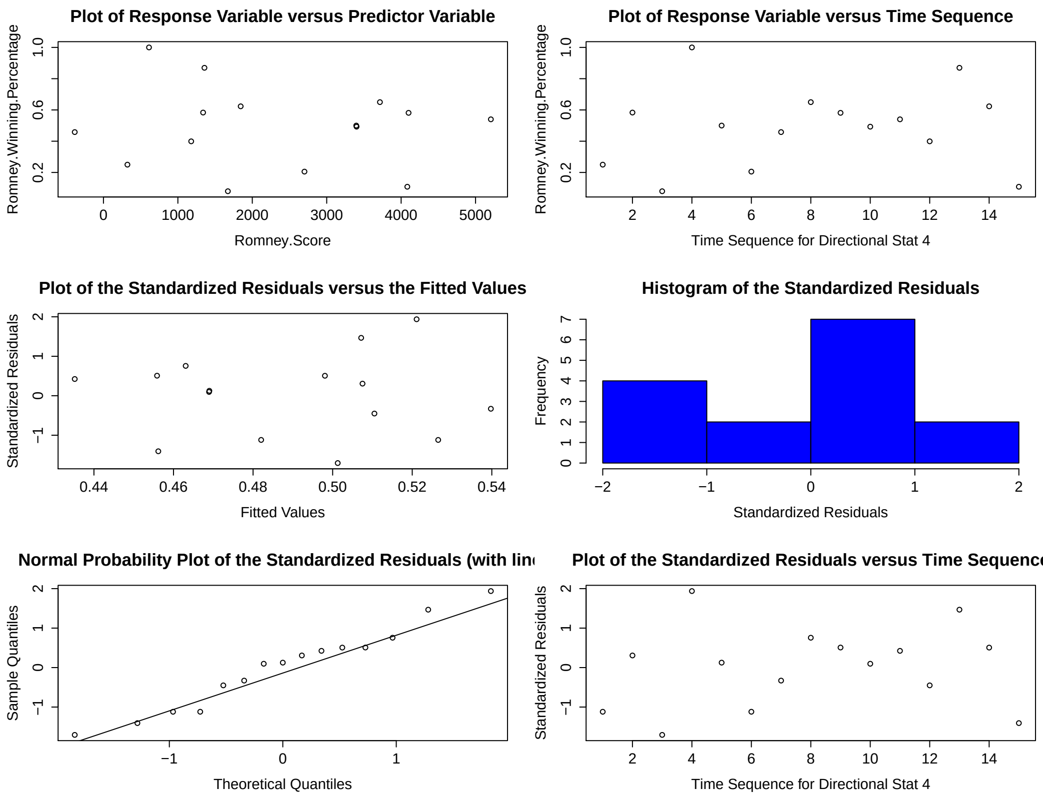
```
                      Value p-value                Decision
Global Stat        1.9386717  0.7470 Assumptions acceptable.
Skewness           0.8950199  0.3441 Assumptions acceptable.
Kurtosis           0.6079938  0.4355 Assumptions acceptable.
Link Function      0.0007755  0.9778 Assumptions acceptable.
Heteroscedasticity 0.4348825  0.5096 Assumptions acceptable.
```

Newt Gingrich

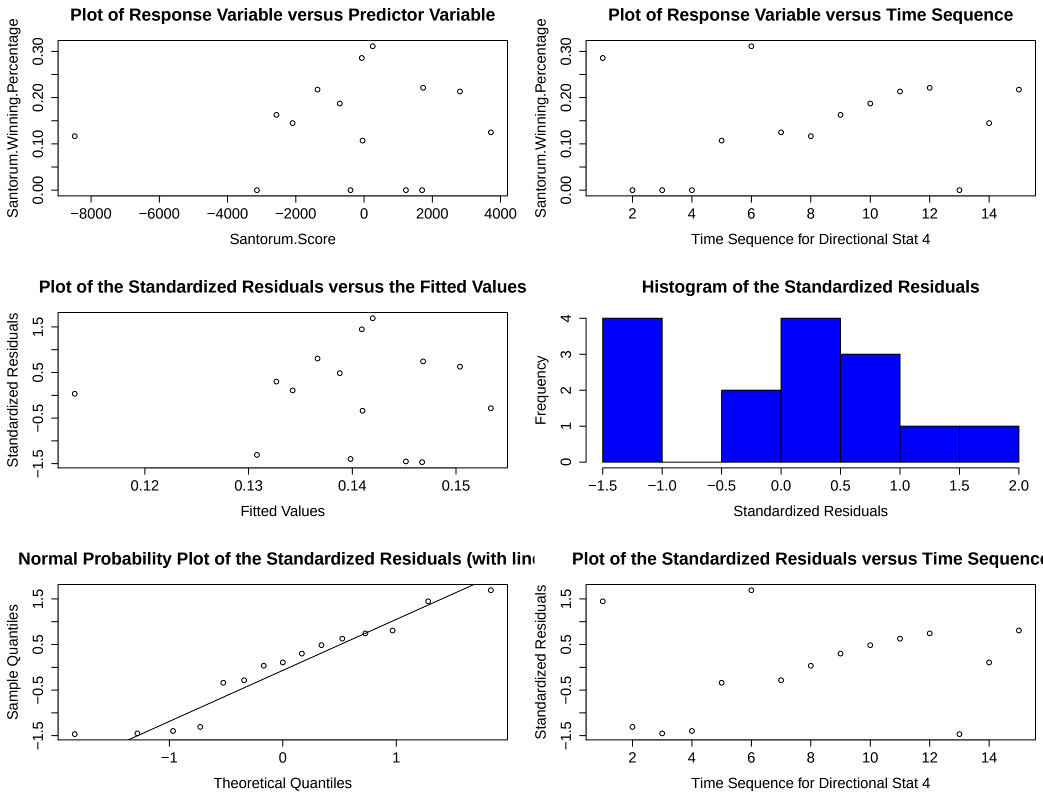
```
Call:
lm(formula = Gingrich.Winning.Percentage ~ Gingrich.Score, data = master.agg.df)

Residuals:
     Min       1Q   Median       3Q      Max
-0.18132 -0.12610 -0.06656  0.04781  0.74147

Coefficients:
                 Estimate Std. Error t value Pr(>|t|)
(Intercept)     1.393e-01  6.857e-02   2.032   0.0631 .
Gingrich.Score -1.117e-04  9.271e-05  -1.205   0.2499
---
Signif. codes:  0 *** 0.001 ** 0.01 * 0.05 . 0.1   1

Residual standard error: 0.2322 on 13 degrees of freedom
Multiple R-squared: 0.1004,Adjusted R-squared: 0.0312
F-statistic: 1.451 on 1 and 13 DF,  p-value: 0.2499


ASSESSMENT OF THE LINEAR MODEL ASSUMPTIONS
USING THE GLOBAL TEST ON 4 DEGREES-OF-FREEDOM:
Level of Significance =  0.05

Call:
 gvlma(x = gingrich.lm)

                     Value   p-value                   Decision
Global Stat        47.9594 9.624e-10 Assumptions NOT satisfied!
Skewness           16.5084 4.843e-05 Assumptions NOT satisfied!
Kurtosis           24.9861 5.774e-07 Assumptions NOT satisfied!
Link Function       0.5937 4.410e-01    Assumptions acceptable.
Heteroscedasticity  5.8712 1.539e-02 Assumptions NOT satisfied!
```

5.2 Appendix B - `gvlma` Graphical Results

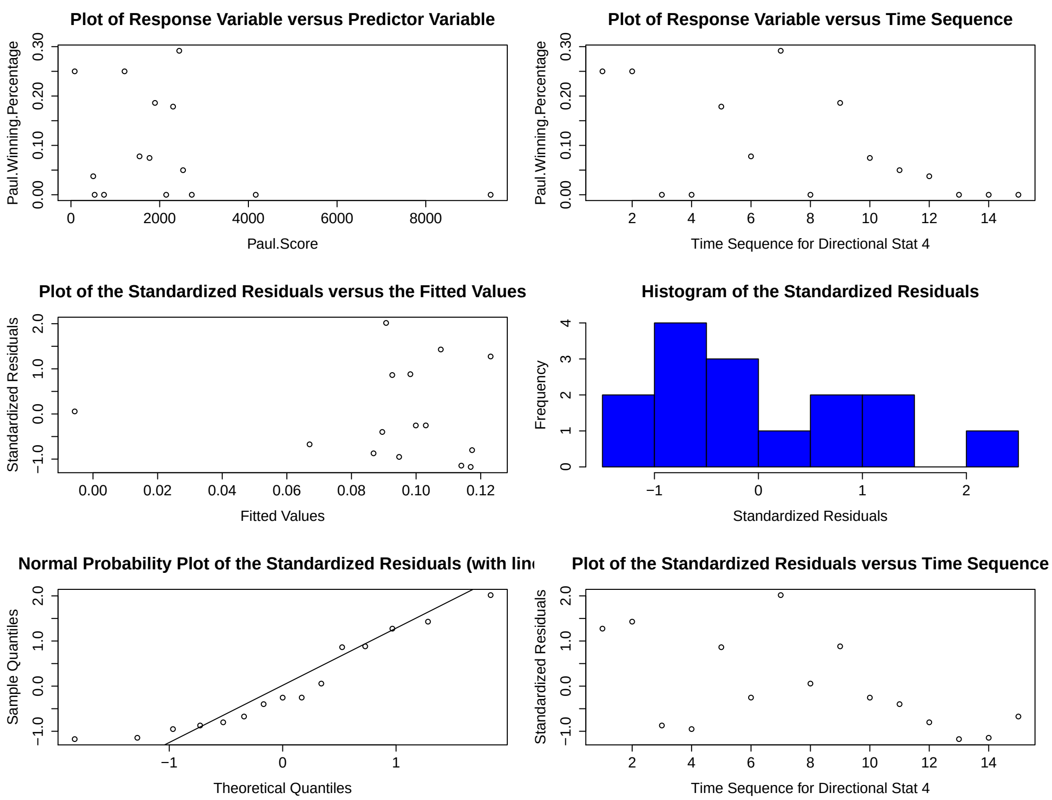
Mitt Romney



Rick Santorum

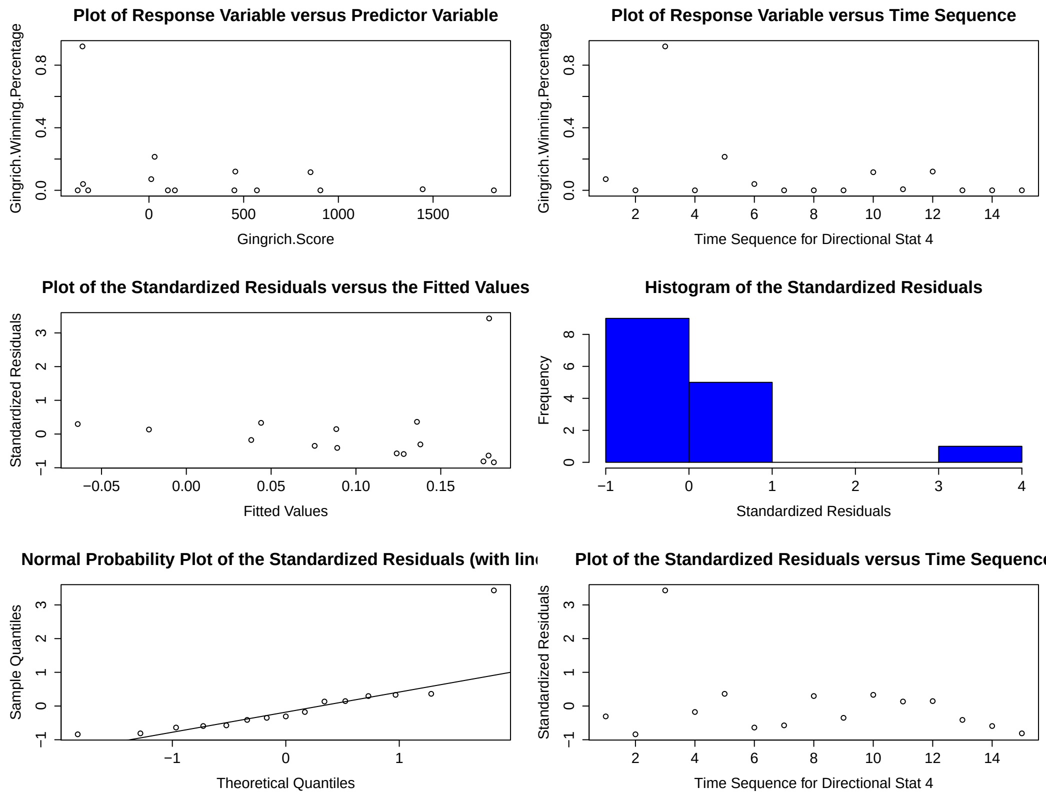


Ron Paul



Plot of Response Variable versus Predictor Variable
Paul.Winning.Percentage
Paul.Score
Plot of Response Variable versus Time Sequence
Paul.Winning.Percentage
Time Sequence for Directional Stat 4
Plot of the Standardized Residuals versus the Fitted Values
Standardized Residuals
Fitted Values
Histogram of the Standardized Residuals
Frequency
Standardized Residuals
Normal Probability Plot of the Standardized Residuals (with lin
Sample Quantiles
Theoretical Quantiles
Plot of the Standardized Residuals versus Time Sequence
Standardized Residuals
Time Sequence for Directional Stat 4

Newt Gingrich



References

[1] Get search — twitter developers.

[2] Twitter.com traffic and demographic statistics by quantcast, April 2012.

[3] J.E. Burt, G.M. Barber, and D.L. Rigby. *Elementary Statistics for Geographers*. Guilford Press, 2009.

[4] R. Kabacoff. *R in Action*. Manning Pubs Co Series. Manning Publications, 2011.

[5] Topsy Labs. Social media political analysis. Technical report, Topsy Labs, 2012.

[6] Bing Liu. Sentiment analysis and subjectivity. In Nitin Indurkhya and Fred J. Damerau, editors, *Handbook of Natural Language Processing, Second Edition*. CRC Press, Taylor and Francis Group, Boca Raton, FL, 2010. ISBN 978-1420085921.

[7] Edsel A. Pena and Elizabeth H. Slate. Global validation of linear model assumptions. *Journal of the American Statistical Association*, 101:341–354, March 2006.

[8] Edsel A. Pena and Elizabeth H. Slate. *gvlma: Global Validation of Linear Models Assumptions*, 2010. R package version 1.0.0.1.

[9] Andranik Tumasjan, Timm Oliver Sprenger, Philipp G. Sandner, and Isabell M. Welpe. Predicting elections with twitter: What 140 characters reveal about political sentiment. In William W. Cohen and Samuel Gosling, editors, *ICWSM*. The AAAI Press, 2010.

[10] Tom White. *Hadoop: The Definitive Guide*. O'Reilly, first edition edition, june 2009.