

Nolan McQueen  
Kunhuan Liu  
Anna Flynn  
CSC-272  
25 April 2016

## **Sentiment Analysis of GOP Debate Tweets: Nominal Versus Text Mining**

### Introduction

The heated political atmosphere of the 2016 presidential race in the United States draws many eager potential voters to watch live television debates of presidential candidates. These enthralling debates captivate audiences around America and lead many to express their opinion on social media outlets like Twitter, Facebook, and Tumblr. Like never before there is a vast amount of data that is available to everyone that expresses voter sentiment towards political candidates and ideas. There is only one problem, this data is only accessible as long strings and words that are very messy. To gain the wealth that lies buried in the text you must find a way to filter through the noisy data and learn an appropriate sentiment model. Once you have a model that can give you the sentiment of a potential voter through their tweets, political campaigns and political strategists can better learn from debates and help make their candidates even stronger for the next one.

This report overviews the differences between the two ways of modeling the sentiment of tweets for political debates through Text Mining and Nominal modeling. Text Mining would just be analyzing the text of the tweet while Nominal learning would classify based on the associative information of each tweet (e.g. location). The training and test data for the modeling came from a dataset that we obtained from CrowdFlower. The file that was used contained Twitter data from a GOP debate that took place last August in Ohio. This data includes the tweets during the debate as well as the sentiment (positive, negative, and neutral) of each tweet among other attributes like retweet counts and locations. This dataset contained over 10,000 tweets. The original dataset was very messy and required much preprocessing in order to run our modeling and testing. The data analytics software package Weka (Waikato Environment for Knowledge Analysis) was used to analyze the data through its various classifiers like Naive Bayes, J48, PART, IBk (a nearest neighbor algorithm), 1R, and 0R. Using Weka's string to word vector filters, the text of the Tweets became a multinomial set of word frequencies that Weka could train on. For learning and testing, a 20% learning and 80% testing split was used because of the vast number of tweets that the dataset contained. In addition, we used the training set to run each algorithm. We found that a Text Mining based model was more accurate in general of predicting the sentiment of a debate tweet than just using the nominal associating data of each tweet.

### Dataset Description

[http://cdn2.hubspot.net/hubfs/346378/DFE\\_CSVs/GOP\\_REL\\_ONLY.csv?t=1458080228879](http://cdn2.hubspot.net/hubfs/346378/DFE_CSVs/GOP_REL_ONLY.csv?t=1458080228879).

Our dataset was obtained from CrowdFlower. CrowdFlower is located in San Francisco, California. It is the "essential data enrichment platform for data science teams" as well as a data mining and crowdsourcing company (LinkedIn). This file contained data from a GOP debate

that took place last August in Ohio. The data included the tweets during the debate as well as the sentiment of each tweet among other attributes. The file included 19 attributes but we only used 11 when running our various algorithms (many were not relevant such as 'name' (Twitter handle) because we were trying to determine the sentiment of the tweet, not who wrote it). These are the ones that we used: candidate, candidate: confidence, sentiment (of tweet), subject\_matter, subject\_matter: confidence, retweet\_count, tweet\_coord, tweet\_created, tweet\_location, user\_timezone, and text (Tweet).

<u>Attribute</u>	<u>Description</u>	<u>Type (Value)</u>
candidate	The GOP candidate mentioned in the tweet.	nominal {'No candidate mentioned','Scott Walker','Donald Trump','Ted Cruz','Ben Carson','Mike Huckabee','Jeb Bush','Chris Christie','Marco Rubio','Rand Paul','John Kasich'}
candidate: Confidence	Confidence value that it is actually the candidate mentioned in the tweet	numeric
<b>Sentiment (class attribute)</b>	<b>The sentiment/mood of the tweet</b>	<b>nominal {Neutral,Positive,Negative}</b>
subject_matter:confidence	Confidence value that it is actually the subject matter mentioned in the tweet	numeric
retweet_count	The number of times the tweet was retweeted.	numeric
subject_matter	The subject of the tweet	nominal {'None of the above','FOX News or Moderators','Foreign Policy','Women's Issues (not abortion though)','LGBT issues','Abortion','Racial issues','Jobs and Economy','Religion','Healthcare (including Medicare)','Immigration','Gun Control'}
text	The words mentioned by the user (the Tweet)	varies (string)

tweet_coord	coordination	numeric
tweet_created	The time the tweet was created	nominal
tweet_location	The location the tweet was posted at	nominal
user_timezone	Timezone of the location of the tweet	nominal

## Data Preparation

In order to prepare our data for data mining in Weka, we had to complete numerous preprocessing steps. First of all, the dataset (original GOP dataset) was in .csv format, so we were able to open it in Excel. However, Excel was not able to separate values correctly--some attribute values were wrongly parsed (Figure 1).

1	In_Related_News	215	RT @pattonoswalt: ##### 6.30E+17 San Diego Pacific Time (US & Canada)
0.6778	MDiner92	0	Hey @ChrisChristi: ##### 6.30E+17 Central Time (US & Canada)
0.3923	gina_catch22gg	45	RT @CarolCNN: #Do: ##### 6.30E+17
1	ERNESTZorro	7	RT @johncardillo: ##### 6.30E+17 Mountain Time (US & Canada)
0.686	LordWhatAMess	0	reason comment is ##### 6.30E+17 Eastern Time (US & Canada)
1	oak523	93	RT @PamelaGeller: ##### 6.30E+17 NJ Eastern Time (US & Canada)
1	rickymeghee	6	RT @ChuckNellis: (##### 6.30E+17 fripp island, sc/ southeast ga
0.4492	TeaTraitors	2	RT @mchar: ##### 6.30E+17 Conspirac Pacific Time (US & Canada)
0.3484	hoopsfanatic28f	404	RT @erinmallory10: ##### 6.30E+17 Lynnwood Pacific Time (US & Canada)
0.6701	ottcanada	415	RT @thekevinryder: ##### 6.30E+17 Vancouver Eastern Time (US & Canada)
1	brattymad	93	RT @MrPooni: Fox ! ##### 6.30E+17
1	America4Aliens	0	#GOPDebate ranking: ##### 6.30E+17 Sydney, New South Wales
1	mch7576	19	RT @TheBaxterBean: ##### 6.30E+17 USA
0.2279	lex_ruu	156	RT @feministabulo: ##### 6.30E+17 Eastern Time (US & Canada)
1	OldOne4Sure	2	RT @mch7576: RT 票 ##### 6.30E+17 Cleveland Central Time (US & Canada)
0.405	outlawjoZ	4965	RT @HillaryClinton: ##### 6.30E+17 Everywhere Eastern Time (US & Canada)
0.6702	BMcCluregolf	0	@fbhw they're goi: ##### 6.30E+17 Green Bay, WI
0.4171	Ruppy_puppy	14	RT @LisaVikingsta: ##### 6.30E+17 Atlantic Time (Canada)

In order to clean up our dataset, we utilized three programs: 1) Notepad++, a text editor to help us manually analyze the problematic lines in the file; 2) CSVed, an open-source software that helps us identify problematic lines automatically; 3) a Java program written on our own, to perform specific operations to the dataset file.

We first evaluated how many lines had less or more columns than the first line (the line with column names) by CSVed. Among 16640 lines there were 5760 lines that had either more or less columns. We investigated these lines which were written out in the evaluation report, and noticed that there were quite a few lines starting with either a url or signs like “@” and “#.” It was considered as a systematic error, and therefore deleting these lines haphazardly was not a good solution (given 13,870 instances). Instead, most fragmental lines were able to be joined based on the observation that some lines were for some reason separated into fragments (Figure 2).



```

1 Found: 3453 errors in: D:\Users\Runhuan\Desktop\English\大二下\CSC272\final project\CSVed\third try\version1.csv
2 (They had less/more columns than the first row which has 20 columns)
3
4 line 13: Mike Huckabee,1,yes,1,Positive,1,Foreign Policy,0.652,,KLWorster,188,,,"RT @WayneDupreeShow: Just woke up to tweet this out #GOPDebate ,,,,,,,,,,
5 line 15: Best line of the night via @GovMikeHuckabee http://t.co/60V5hxH1cy ,8/7/15 9:54,6.29697E+17,"Fargo, ND",Central Time (US & Canada),,,,,,,,,,
6 line 40: ,0.2277,yes,0.6495,Negative,0.3402,None of the above,0.4218,,RotoStocks,2,,,"RT @stockwizards3: Trump/Carson ticket would win in a landslide... [with:
7 line 41: @realDonaldTrump @seanhannity @AnnCoulter #Republican #potus #Trump2016 善",8/7/15 9:54,6.29697E+17,, [with: 1 columns]
8 line 50: No candidate mentioned,1,yes,1,Negative,0.6591,Racial issues,1,,briasslide,4416,,,"RT @LeKarmaSucre: How the #GOPDebate handled #BlackLivesMatter [with
9 line 51: http://t.co/8ZGdYv61ECT ,8/7/15 9:54,6.29697E+17,Central Time (US & Canada) [with: 1 columns]
10 line 64: No candidate mentioned,1,yes,1,Negative,0.6667,Abortion,1,,debby_lowery,,230,,,"RT @mydaughtersarmy: Dear GOP, [with: 15 columns]
11 line 65: If you want to stop abortions... [with: 1 columns]
12 line 66: #GOPDebate [with: 1 columns]
13 line 67: http://t.co/RbLlt4iPx6 ,8/7/15 9:54,6.29697E+17,NW Florida,Central Time (US & Canada) [with: 1 columns]
14 line 68: Ted Cruz,0.4664,yes,0.6829,Positive,0.3659,Foreign Policy,0.2499,,MelanieEli2015,,123,,,"RT @KarrattiPaul: Join ISIS and sign your death warrant, [with:
15 line 69: Signed @tedcruz next President of the US 榧榧 [with: 1 columns]
16 line 70: #GOPDebate #WakeUpAmerica #CruzCrew 榧榧",8/7/15 9:54,6.29697E+17,, [with: 1 columns]
17 line 72: No candidate mentioned,1,yes,1,Negative,1,FOX News or Moderators,1,,search4rr,,154,,,"RT @AmyMek: Status _榧?Single [with: 15 columns]
18 line 73: I broke up with @FoxNews last night! Goodbye @megynkelly, @BretBaier & Chris Wallace. [with: 2 columns]
19 line 74: #GOPDebate http://t.榧 ,8/7/15 9:54,6.29697E+17,"Florida,, usually",Eastern Time (US & Canada) [with: 3 columns]
20 line 75: Donald Trump,1,yes,1,Positive,0.6693,None of the above,0.6815,,toot2014,,18,,,"RT @FrankLuntz: Before the #GOPDebate, 14 focus groupers said they had fa
21 line 76: After, only 3 saw him positively. http://榧 ,8/7/15 9:54,6.29697E+17, NYC,Eastern Time (US & Canada) [with: 2 columns]
22 line 77: No candidate mentioned,1,yes,1,Positive,0.7065,None of the above,1,,DavidWarren25,,23,,,"RT @ Holly_Renee: Just made my first donation to @CarlyFlorina.
23 line 78: So impressed by her #GOPDebate performance. [with: 1 columns]
24 line 79: #Fiorina http://t.co/JLfhFvZ7ST ,8/7/15 9:54,6.29697E+17,San Diego,Pacific Time (US & Canada) [with: 1 columns]
25 line 84: Donald Trump,1,yes,1,Positive,0.3508,None of the above,1,,ancient_echoes,,484,,,"RT @MsPackyetti: Donald Trump's campaign reveals 1 important thing: Twi
26 line 85: And they vote. [with: 1 columns]
27 line 86: That should scare eve榧",8/7/15 9:54,6.29697E+17,"Oxnrd, California",Arizona [with: 2 columns]
28 line 97: No candidate mentioned,0.6691,yes,1,Neutral,0.6691,None of the above,1,,MBryant_73,,18,,,"RT @GovMikeHuckabee: Luntz: ""As the lines go up, they almost
29 line 98: 榧榧",8/7/15 9:54,6.29697E+17,Searcy,Central Time (US & Canada) [with: 1 columns]
30 line 105: No candidate mentioned,1,yes,1,Negative,0.6889,Abortion,0.6889,,JoshDorner,,0,,,"They榧榧 all Todd Akin now: How the Planned Parenthood sting backfire
31 line 106: http://t.co/ekXPFKtHQm ,8/7/15 9:54,6.29697E+17,The District. ,Quito [with: 1 columns]
32 line 128: No candidate mentioned,1,yes,1,Neutral,0.6557,None of the above,1,,KristenLuciano1,,156,,,"RT @CarlyFlorina: At #GOPDebate, I won over our largest audi
33 line 129: 榧榧",8/7/15 9:54,6.29697E+17,Central Time (US & Canada) [with: 1 columns]

```

Figure 2. Automatically generated report on problematic lines by CSVed. The circled lines could be joined together to form a complete, normal instance.

A java program was written to join the lines together (Figure 3-a,3-b).

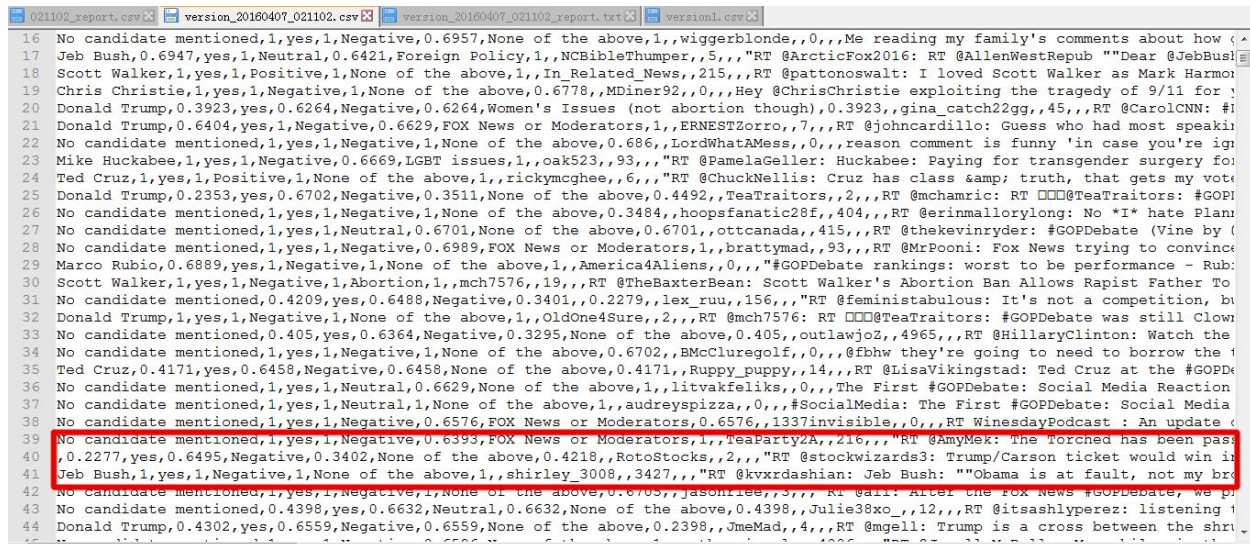
```

22 No candidate mentioned,1,yes,1,Negative,1,None of the above,0.686,,LordWhatAMess,,0,,,"reason comment is funny 'in case you're ig
23 Mike Huckabee,1,yes,1,Negative,0.6669,LGBT issues,1,,oak523,,93,,,"RT @PamelaGeller: Huckabee: Paying for transgender surgery fo
24 Ted Cruz,1,yes,1,Positive,1,None of the above,1,,rickymcgee,,6,,,"RT @ChuckNeillis: Cruz has class & truth, that gets my vot
25 Donald Trump,0.2353,yes,0.6702,Negative,0.3511,None of the above,0.4492,,TeaTraitors,2,,,"RT @mchamric: RT 榧榧TeaTraitors: #GOPI
26 No candidate mentioned,1,yes,1,Negative,1,None of the above,0.3484,,hoopsfanatic28f,,404,,,"RT @erinmallorylong: No *I* hate Plan
27 No candidate mentioned,1,yes,1,Neutral,0.6701,None of the above,0.6701,,ottcanada,,415,,,"RT @thekevinryder: #GOPDebate (Vine by
28 No candidate mentioned,1,yes,1,Negative,0.6989,FOX News or Moderators,1,,brattymad,,93,,,"RT @MrPooni: Fox News trying to convince
29 Marco Rubio,0.6889,yes,1,Negative,1,None of the above,1,,America4Aliens,,0,,,"#GOPDebate rankings: worst to be performance - Rub
30 Scott Walker,1,yes,1,Negative,1,Abortion,1,,mch7576,,19,,,"RT @TheBaxterBean: Scott Walker's Abortion Ban Allows Rapist Father To
31 No candidate mentioned,0.4209,yes,0.6488,Negative,0.3401,,0.2279,,lex_ruu,,156,,,"RT @feministabulous: It's not a competition, bu
32 Donald Trump,1,yes,1,Negative,1,None of the above,1,,oldOne4Sure,,2,,,"RT @mch7576: RT 榧榧TeaTraitors: #GOPDebate was still Clow
33 No candidate mentioned,0.405,yes,0.6364,Negative,0.3295,None of the above,0.405,,outlawjoz,,4965,,,"RT @HillaryClinton: Watch the
34 No candidate mentioned,1,yes,1,Negative,1,None of the above,0.6702,,BMcCluregolf,,0,,,"@fbhw they're going to need to borrow the
35 Ted Cruz,0.4171,yes,0.6458,Negative,0.6458,None of the above,0.4171,,Ruppy_puppy,,14,,,"RT @LisaVikingstad: Ted Cruz at the #GOPDe
36 No candidate mentioned,1,yes,1,Neutral,0.6629,None of the above,1,,litvakfeliks,,0,,,"The First #GOPDebate: Social Media Reaction
37 No candidate mentioned,1,yes,1,Neutral,1,None of the above,1,,audreyspiza,,0,,,"#SocialMedia: The First #GOPDebate: Social Media
38 No candidate mentioned,1,yes,1,Negative,0.6576,FOX News or Moderators,0.6576,,1337invisible,,0,,,"RT WinesdayPodcast : An update
39 No candidate mentioned,1,yes,1,Negative,0.6393,FOX News or Moderators,1,,TeaParty2A,,216,,,"RT @AmyMek: The Torched has been pas
40 0.2277,yes,0.6495,Negative,0.3402,None of the above,0.4218,,RotoStocks,,2,,,"RT @stockwizards3: Trump/Carson ticket would win i
41 @realDonaldTrump @seanhannity @AnnCoulter #Republican #potus #Trump2016 善",8/7/15 9:54,6.29697E+17,,
42 Jeb Bush,1,yes,1,Negative,1,None of the above,1,,shirley_3008,,3427,,,"RT @kvxrdashian: Jeb Bush: ""Obama is at fault, not my br
43 No candidate mentioned,1,yes,1,Negative,1,None of the above,0.6705,,jasonrlee,,3,,,"RT @ali: After the Fox News #GOPDebate, we p
44 No candidate mentioned,0.4398,yes,0.6632,Neutral,0.6632,None of the above,0.4398,,Julie38xo,,12,,,"RT @itsashlyperez: listening!
45 Donald Trump,0.4302,yes,0.6559,Negative,0.6559,None of the above,0.2398,,JmeMad,,4,,,"RT @mgell: Trump is a cross between the shu
46 No candidate mentioned,1,yes,1,Negative,0.6596,None of the above,1,,nathansiegel,,4006,,,"RT @JanelleMyBelle: Meanwhile, in the
47 No candidate mentioned,0.3974,yes,0.6304,Neutral,0.3261,None of the above,0.3974,,Chase_Leal0,,50,,,"RT @RowdyGentleman: If you榧
48 No candidate mentioned,1,yes,1,Negative,0.6869,Racial issues,0.6869,,Blessed_Shaday,,450,,,"RT @brownblaze: PLEASE RT. #KKKorGOP
49 Donald Trump,1,yes,1,Negative,1,Women's Issues (not abortion though),0.6713,,SarahMLevin,,0,,,"Trump thinks criticism of his mis
50 No candidate mentioned,1,yes,1,Negative,0.6591,Racial issues,1,,briasslide,4416,,,"RT @LeKarmaSucre: How the #GOPDebate handled i

```

Figure 3-a. The dataset before the process of joining. Line 41 was the fragmentation of the tweet from line 40.





```

16 No candidate mentioned,1,yes,1,Negative,0.6957,None of the above,1,,wiggerblonde,,0,,Me reading my family's comments about how < ^
17 Jeb Bush,0.6947,yes,1,Neutral,0.6421,Foreign Policy,1,,NCBibleThumper,,5,,,"RT @ArcticFox2016: RT @AllenWestRepub ""Dear @JebBush!
18 Scott Walker,1,yes,1,Positive,1,None of the above,1,,In Related News,,215,,,"RT @pattonoswalt: I loved Scott Walker as Mark Harmoi
19 Chris Christie,1,yes,1,Negative,1,None of the above,0.6778,,MDiner92,,0,,,"Hey @ChrisChristie exploiting the tragedy of 9/11 for
20 Donald Trump,0.3923,yes,0.6264,Negative,0.6264,Women's Issues (not abortion though),0.3923,,gina_catch22gg,,45,,,"RT @CarolCNN: #I
21 Donald Trump,0.6404,yes,1,Negative,0.6629,FOX News or Moderators,1,,ERNestZorro,,7,,,"RT @johncardillo: Guess who had most speaki
22 No candidate mentioned,1,yes,1,Negative,1,None of the above,0.686,,LordWhatAMess,,0,,,"reason comment is funny 'in case you're ign
23 Mike Huckabee,1,yes,1,Negative,0.6669,LCBT issues,1,,oak523,,93,,,"RT @PamelaGeller: Huckabee: Paying for transgender surgery for
24 Ted Cruz,1,yes,1,Positive,1,None of the above,1,,rickymcgee,,6,,,"RT @ChuckNellis: Cruz has class & truth, that gets my vote
25 Donald Trump,0.2353,yes,0.6702,Negative,0.3511,None of the above,0.4492,,TeaTraitors,,2,,,"RT @mchamric: RT @TeaTraitors: #GOPDebate
26 No candidate mentioned,1,yes,1,Negative,1,None of the above,0.3484,,hoopsfanatic28f,,404,,,"RT @serinmallorylong: No *I* hate Plan
27 No candidate mentioned,1,yes,1,Neutral,0.6701,None of the above,0.6701,,ottcanada,,415,,,"RT @thekevinryder: #GOPDebate (Vine by
28 No candidate mentioned,1,yes,1,Negative,0.6989,FOX News or Moderators,1,,brattymad,,93,,,"RT @MrPooni: Fox News trying to convince
29 Marco Rubio,0.6889,yes,1,Negative,1,None of the above,1,,America4Aliens,,0,,,"#GOPDebate rankings: worst to be performance - Rub
30 Scott Walker,1,yes,1,Negative,1,Abortion,1,,mch7576,,19,,,"RT @TheBaxterBean: Scott Walker's Abortion Ban Allows Rapist Father To
31 No candidate mentioned,0.4209,yes,0.6488,Negative,0.3401,,0.2279,,lex_ruu,,156,,,"RT @feministabulous: It's not a competition, bu
32 Donald Trump,1,yes,1,Negative,1,None of the above,1,,OldOne4Sure,,2,,,"RT @mch7576: RT @TeaTraitors: #GOPDebate was still Clow
33 No candidate mentioned,0.405,yes,0.6364,Negative,0.3295,None of the above,0.405,,outlawjoz,,4965,,,"RT @HillaryClinton: Watch the
34 No candidate mentioned,1,yes,1,Negative,1,None of the above,0.6702,,EMcCluregolF,,0,,,"@fbhw they're going to need to borrow the
35 Ted Cruz,0.4171,yes,0.6458,Negative,0.6458,None of the above,0.4171,,Ruppy_puppy,,14,,,"RT @LisaVikingstad: Ted Cruz at the #GOPDe
36 No candidate mentioned,1,yes,1,Neutral,0.6629,None of the above,1,,litvakfeliks,,0,,,"The First #GOPDebate: Social Media Reaction
37 No candidate mentioned,1,yes,1,Neutral,1,None of the above,1,,audreyspizaa,,0,,,"#SocialMedia: The First #GOPDebate: Social Media
38 No candidate mentioned,1,yes,1,Negative,0.6576,FOX News or Moderators,0.6576,,1337invisible,,0,,,"RT WinesdayPodcast : An update
39 No candidate mentioned,1,yes,1,Negative,0.6393,FOX News or Moderators,1,,TeaParty2A,,216,,,"RT @AmyMek: The Torch has been pass
40 0.2277,yes,0.6495,Negative,0.3402,None of the above,0.4218,,RotoStocks,,2,,,"RT @stockwizards3: Trump/Carson ticket would win in
41 Jeb Bush,1,yes,1,Negative,1,None of the above,1,,shirley3008,,3427,,,"RT @kvxrddashian: Jeb Bush: ""Obama is at fault, not my br
42 No candidate mentioned,1,yes,1,Negative,1,None of the above,0.6703,,jasonhies,,3,,,"RT @all: After the Fox News #GOPDebate, we p
43 No candidate mentioned,0.4398,yes,0.6632,Neutral,0.6632,None of the above,0.4398,,Julie38xo,,12,,,"RT @itsashlyperez: listening
44 Donald Trump,0.4302,yes,0.6559,Negative,0.6559,None of the above,0.2398,,JmeMad,,4,,,"RT @mgell: Trump is a cross between the shr

```

Figure 3-b. The dataset after joining fragment lines. The line 41 in figure 3-b is the line 42 in figure 3-a.

We then put our fragment-joined dataset into CSVed, looking for problematic lines. Surprisingly, CSVed parsed the dataset correctly. Although we still saw flaws in CSVed, the text editor nor Excel correctly parsed the dataset like it did.

We decided to let Weka judge our dataset. As there were potential quotes and commas (‘, ’) in the “text” value of an instance, we performed an “escape” on these special characters using our own java program. The dataset was successfully loaded into weka as a .csv file (Figure 4).

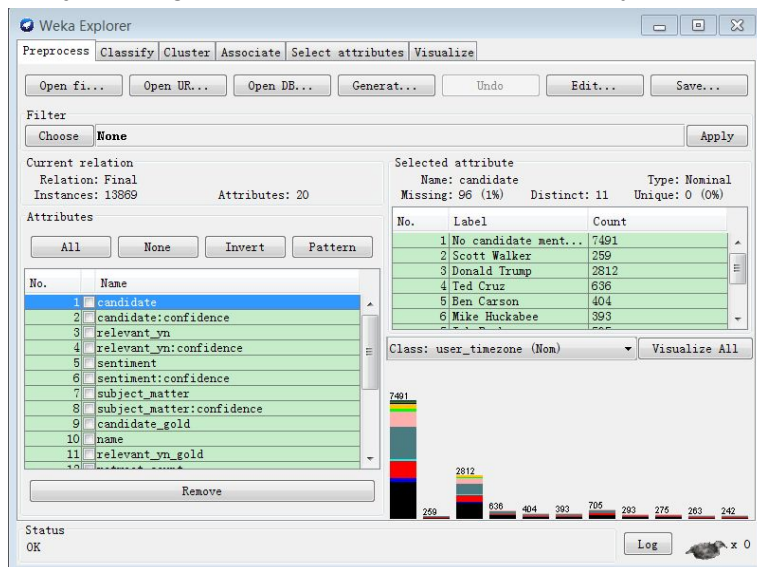


Figure 4. The preprocess of the dataset was successful. The dataset loaded into Weka.

Loaded from a .csv file, all attributes were assigned to be nominal. We used a filter to transform automatically assigned type (by Weka) into the desirable type. For example, the type of “candidate:confidence” was transformed into numeric by the NominalToNumeric filter.

## Data Analysis

### Classification Models

For the data analysis we used a standard set of classifiers that are available on Weka. We implemented a variety of classifiers but in order to obtain the highest accuracy we ran Naive Bayes, Naive Bayes Multinomial, PART, J48, the nearest-neighbor algorithm, IBk, 1R, and 0R. These classifiers were implemented on both the Text Mining and on the Nominal attribute Twitter data. *NaiveBayes* implements the Naive Bayes classifier which calculates the probabilities of each class value. Naive Bayes uses a normal distribution of probabilities for the classification which is good for the messy data that comes from Twitter data (need to say why we picked this?) The *Naive Bayes Multinomial* classifier is especially good when it comes to text mining with the word frequencies that are discussed later. In WEKA, It is incarnated in the [weka.classifiers.bayes.NaiveBayes](#) class. This algorithm is a multinomial form of Naive Bayes that takes into account word frequencies, this helps in determining the class of a document (or Tweet in our case). The *rule learner PART*, which induces a list of rules by learning partial decision trees, is a symbolic algorithm that produces rules which can be very valuable as they are easy to understand. PART is also a “separate and conquer” algorithm. It produces decision lists, each containing a list of rules. Each item in the dataset is “assigned the classification of the first matching rule” (WordPress) and at each iteration, a partial decision tree is built and the “best” leaf in each tree becomes a rule. This algorithm is implemented by the [weka.classifiers.rules.PART](#) class. Of course, the *J48* algorithm was chosen because of its visualization capabilities. J48 implements the C4.5 algorithm in Weka. It is a decision tree classifier in which a pruned decision tree is built from training data in order to classify new, unseen items. The lazy learner *k-Nearest Neighbors (kNN)* occasionally gives excellent results in text classification problems. The WEKA class that implements this algorithm is [weka.classifiers.lazy.IBk](#). This algorithm is called IBk in Weka and it compares each new instance in the testing data to a certain number of instances (neighbors) in your training data. The instance is then classified according to its nearest-neighbor (K) or instance that is most similar to it. *0R* is an algorithm that chooses the majority class value. *0R* is not the best algorithm to use because it does not take into account any input attributes. We chose *1R* because it is the most simple rule-based algorithm. It learns a set of rules that are only based on one input attribute.

### StringToWordVector

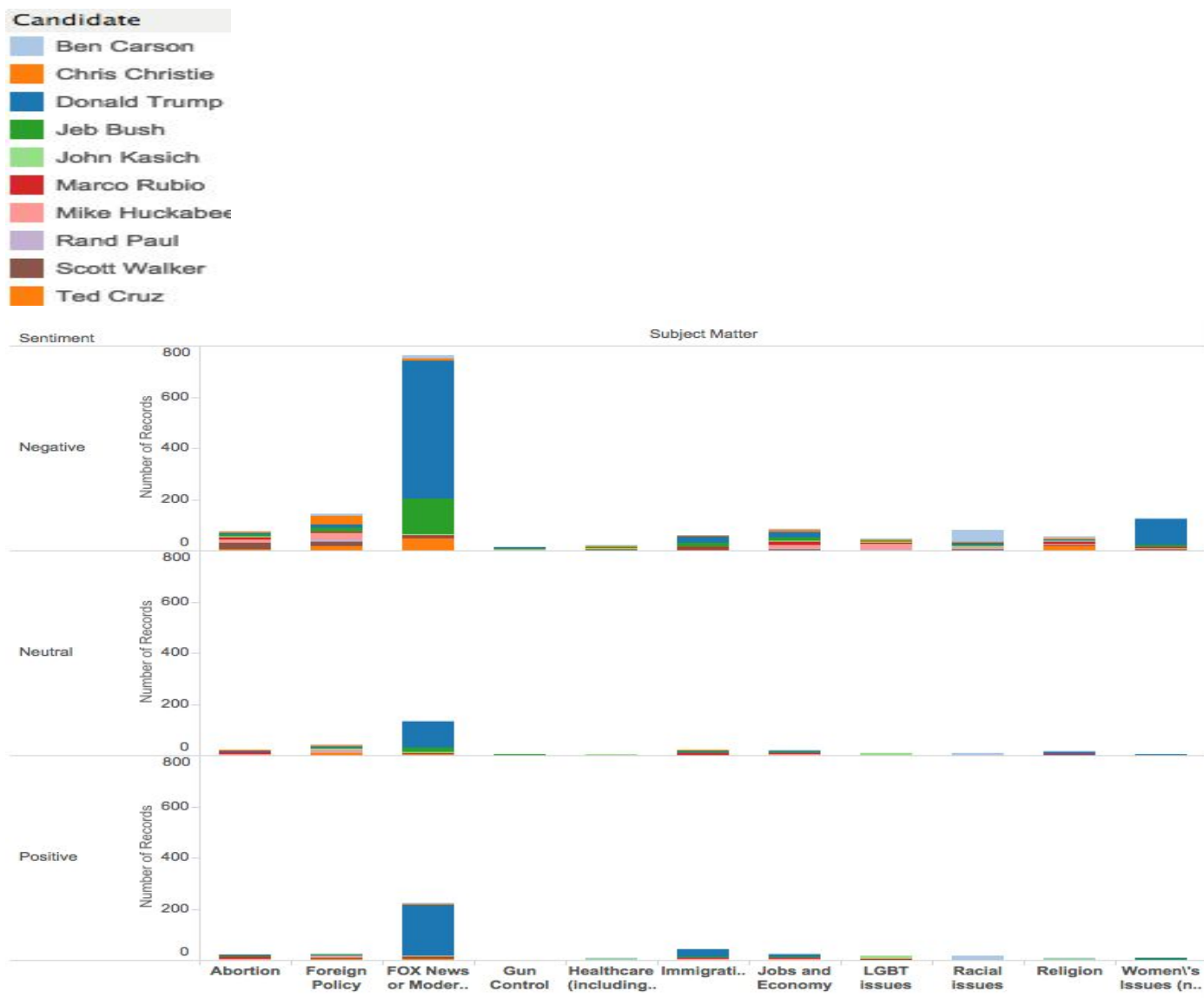
For the text mining portion of this analysis it was needed to be able to break apart the strings of text that came from our dataset of tweets. Weka offers a feature that makes it easy to do so, the *StringToWordVector*. *StringToWordVector* is an unsupervised filter that can be found in the preprocess tab. This filter takes the tweet text string attribute and converts it to a set of new attributes that are words found in the tweet. By using the *OutputWordCounts* feature that allows you to output words based on the frequency of their occurrence in the tweet data. If a word occurs over a certain minimum frequency it will be added as a new attribute to the dataset. By

default, *StringToWordVector* just outputs whether a certain word attribute occurs or not so you must implement *OutputWordCounts*. Once we implemented the *StringToWordVector* we were able to run all of our classifications on the text mining dataset. Because of the Multinomial nature of the output of the *StringToWordVector* we chose *Naive Bayes MultiNomial* as one of our classifiers. The use of decision trees in J48 made it a good candidate for high accuracy with text mining as well. By splitting the strings of text into its individual words was very useful in creating an appropriate text mining model for our sentiment analysis.

### **Stratification**

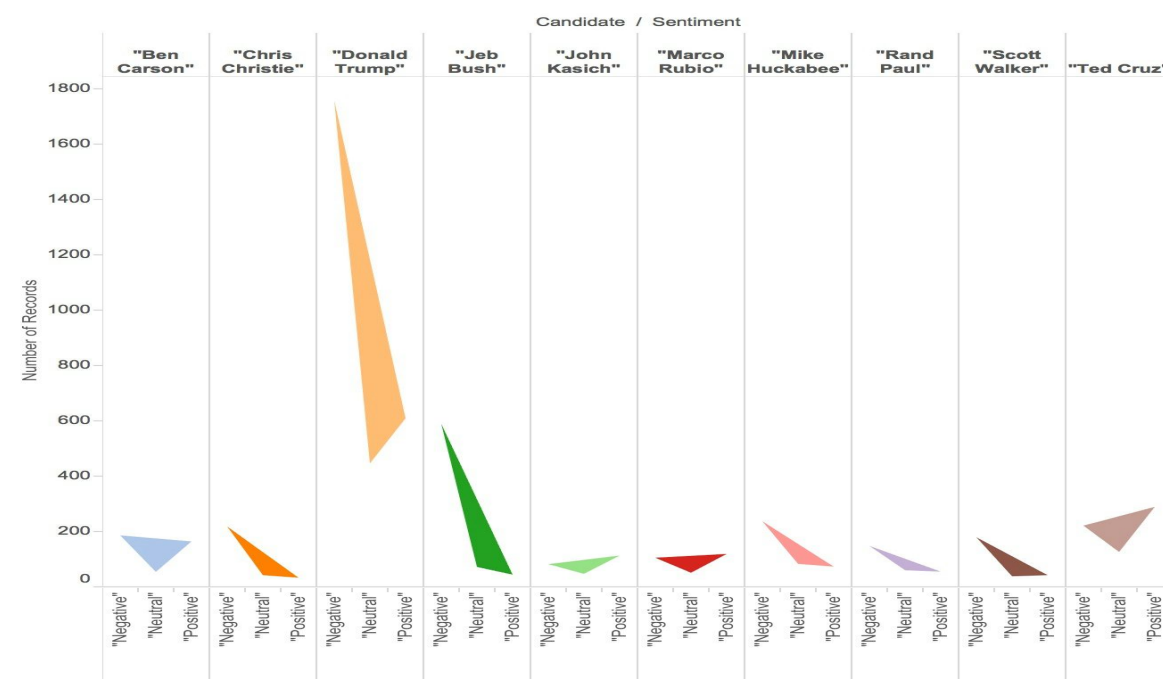
After our successful preprocessing, we had 13,870 instances in our dataset. There are two reasons we performed a stratification that kept class value in the same proportion as the original dataset but split the data into two files. First, it is too large for Weka to generate machine learning models--20 percent of the dataset means a 2774-instance sample, which is enough for our learning purpose. Second, we don't want to generate models based on the whole dataset, but instead we want to build our own test dataset. The stratification gave us a dataset with 2774 instances (later for training purpose) and one with 11096 instances (for testing purposes).

## Visualizations



This bar graph shows the relationship between the subject matter (of the tweet), candidate, and sentiment. The y-axis displays the sentiment values and the x-axis shows the subject matter values. Each candidate is represented by a different color in the bar graph (as seen in the key above). In the third column from the left, the most popular subject matter of the tweets is 'FOX News or Moderators.' The majority of those tweets have a negative sentiment. Even though this shows Trump being mentioned in the highest amount of negative tweets, it is important to remember that the tweet is classified based on the sentiment of the tweet, not the sentiment of the candidate. Perhaps this shows how much influence the media has over people's responses and that more people tweet if they are feeling negatively about something.





This polygon bar chart reflects the mentions of a candidate in a tweet and its respective sentiment classification. The reason why this visualization of our sentiment is especially interesting is because of the way the polygon bar chart can help aid in discovering trends among candidates. On the x-axis there is sentiment values for each candidate in the order of negative, neutral and positive. This creates a sort of sentiment triangle for each candidate. As you see for a few of the candidate their triangles seem to point in a certain direction of sentiment. For example, Donald Trump's sentiment triangle seems to point strongly towards the negative while others like Ted Cruz seem to point towards a more positive sentiment direction. This is interesting because now 8 months later Ted Cruz has gained much more positive sentiment and popularity. This may not yield any specific results but is certainly good for hypothesis creation and for discovering new trends among political candidates.

## Results

### Nominal Accuracy

We were able to perform 0R, 1R, Naive Bayes, PART, IBk and J48 using percentage split (66%) on 20% of our training data and using 80% as testing data. Both the 0R algorithm and the J48 algorithm gave the highest accuracies (62.4602%) when performing the 66% percentage split. 0R predicted the class value as 'negative' because it is the majority class value. When performing J48, only 1 leaf was produced (negative) most likely because the majority of the tweets were negative in the training set. 1R produced hundreds of rules (ex. If location->Texas then sentiment->positive). We wanted to see which attribute was the most powerful in the classification of a particular instance. The problem is that a Twitter user can type in any location they want (including a fake location) so 1R produced hundreds of different rules. Each rule's

antecedent was a particular 'location,' thus being the most powerful attribute but was overfitting. We used PART, another rule classifier, to determine whether we could get a better percentage than J48 since it takes into account (partial) decision trees. However, J48 proved to perform slightly better. One of the rules produced was: if location->Los Angeles, CA and candidate=No candidate mentioned and subject matter=Fox News or Moderators then sentiment=negative. Naive Bayes did not perform very accurately but had a greater percentage than 1R. When performing IBk (nearest-neighbor) on our dataset, we were not surprised that the percentage accuracy increased when we increased the k value. We used a smaller k value than the test data because we were testing on far less instances.

When using 80% of our original dataset to test, the percentages varied greatly. Again, 0R and J48 performed almost equally. J48 only produced 1 leaf which was negative sentiment, like the majority class attribute that 0R predicted. When performing IBk, we used a larger neighborhood size since we were testing on 80% of our original dataset. Surprisingly, when we increased k to 150, it decreased in accuracy instead of continuing to increase. Running PART on the test data performed similarly to the percentage split and used 'location' to predict the class value. It produced 124 rules (e.g. If location=Florida then sentiment=negative). 1R performed very poorly because it chose location in which to create rules from since it had the lowest error rate but the rules produced were overfitting when creating the training model, so it did not perform very well on the test data. Naive Bayes performed less accurately on the test data than the other algorithms we used. The accuracies varied when using percentage split versus test data depending on the algorithm we used.

Classification Algorithm	Percentage Split (66%)	Test Data (80%)
0R	62.4602%	61.19%
1R	39.2365%	44.142%
Naive Bayes	58.6426%	59.2376%
PART	62.1421%	60.7606%
IBk	k=1 54.2948% k=3 54.9311% k=5 55.5673%	k=50 61.5627% k=100 62.2206% k=150 61.9863%
J48	62.4602%	61.1932%

Table 1

\*we used different neighborhood sizes for the Nominal dataset because percentage split only used 66% of the 20% training data versus using 80% on the test data which tested on many more instances, therefore needing a larger neighborhood size

## Analysis of Each Algorithm

### 0R

The rule generated by the 0R algorithm was “sentiment->negative.” The confusion matrix shows an accuracy rate of zero for “positive” and “negative” instances.

	Neutral	Positive	Negative
Neutral	0	0	206
Positive	0	0	148
Negative	0	0	589

### 1R

This algorithm uses “tweet\_location” to predict the sentiment. There were 4286 values for this attribute, leading to the creation of 4287 rules (including a rule for a missing value). Deleting this overfitting attribute results in overfitting with attribute “tweet\_created.” Overall, 1R is not very helpful. The confusion matrix shows that none of the three values have a desirable accuracy rate (Neutral: 41%, positive: 4.0%, negative: 47.3%)

	Neutral	Positive	Negative
Neutral	85	9	112
Positive	81	6	61
Negative	294	16	279

### Naive Bayes - probabilistic approach

This algorithm used 9 attributes (excluding the text and the class attribute) to build the model. However, an analysis on the model finds that the nominal attributes such as tweet\_created and tweet\_location have too many unique values, and this is the case of overfitting because for a lot of values there is only 1 instance that has the value. No doubt the model performs poorly.

	Neutral	Positive	Negative
Neutral	46	6	154
Positive	22	16	110
Negative	83	15	491

### J48 - decision tree algorithm

The J48, as an advanced algorithm that generates decision trees, chose to use OR to predict. The choice that J48 made suggests its concession that nominal attributes are not predictive.

	Neutral	Positive	Negative
Neutral	0	0	206
Positive	0	0	148
Negative	0	0	589

### PART - rule algorithm

PART is a “separate-and-conquer” algorithm that builds partial C4.5 decision trees while making sure that at least 1 attribute is involved. In other words, the algorithm uses C4.5 (the same algorithm used by J48) but won’t use OR even in the worst situation, since PART first separates and then runs the C4.5 algorithm.

The model generated by PART uses tweet\_location first, and under each leaf by tweet\_location, some additional rules are added to predict for sentiment.

For example:

tweet\_location = San Francisco, California: Positive (4.19/2.01)

tweet\_location = Texas AND

candidate = No candidate mentioned: Negative (8.14/3.73)

The matrix built shows there is an effort to attain a better accuracy rate for each value. However, it failed.

	Neutral	Positive	Negative
Neutral	2	2	202
Positive	1	5	142
Negative	8	2	579



### IBk - nearest neighbour

The nearest neighbour algorithm helps us to make a final conclusion that nominal attributes are not predictive. There is not much similarity among neutral instances, neither is there among positive instances. By increasing the k value we did get a best overall accuracy rate, but when we look at the confusion matrix, we noticed that the accuracy for neutral sentiment and the accuracy for positive sentiment both decrease. The overall accuracy rate went up simply because there is a better prediction on the majority--when only looking at 1 nearest neighbour around a negative instance, there is a larger chance to have a neutral or positive instance.

<b>k=1</b>	<b>Neutral</b>	<b>Positive</b>	<b>Negative</b>
<b>Neutral</b>	67	25	114
<b>Positive</b>	32	43	73
<b>Negative</b>	126	61	402

<b>k=50</b>	<b>Neutral</b>	<b>Positive</b>	<b>Negative</b>
<b>Neutral</b>	41	9	156
<b>Positive</b>	9	21	118
<b>Negative</b>	52	9	528

### Text Mining Accuracy

We ran the four classification algorithms Naive Bayes Multinomial, PART, IBk, and J48. We also ran Zero-R and One-R to compare our results. We cannot run a test on test data (80% of the whole dataset) due to an OutOfMemory error (Figure 5). The highest accuracies came from Naive Bayes MultiNomial and from PART with 68 and 66 percent accuracies each. All of the algorithms fell in between 60 and 70 percent accuracy. By using the large number of words from the tweets these algorithms were able to stay on par and even fare better in some cases with regards to nominal classification. Because we could not run on test data due to memory issues we only ran the text mining classifiers on 20% of the data using a 66% split. This gave us almost 3000 instances of twitter text to be able to perform sentiment analysis. As you can see in the J48 tree that was produced for Text Mining (Figure 6) the large amount of data leads to models that are very complex. However, algorithms that came up with simple models were just as predictive. For example, 1R in our Text Mining results came up with “\_thanks” as the word that was most predictive of sentiment in the set of data. In viewing the results of such classifiers as J48 more positive words and names seem to be the biggest predictors of sentiment. The lack of positive words seems to be large predictor in Text Mining classification trees. The introduction of n-grams may help classification accuracy in the future instead of using only 1-gram word frequencies as we did in this report.

```

Weka - Log
---Registering Weka Editors---
Trying to add database driver (JDBC): jdbc.idbDriver - Error, not in CLASSPATH?
Warning : data contains more attributes than can be displayed as attribute bars.
Exception in thread "Thread-9" java.lang.OutOfMemoryError: GC overhead limit exceeded
    java.util.Arrays.copyOf(Arrays.java:3332)
    java.lang.AbstractStringBuilder.expandCapacity(AbstractStringBuilder.java:137)
    java.lang.AbstractStringBuilder.ensureCapacityInternal(AbstractStringBuilder.java:121)
    java.lang.AbstractStringBuilder.append(AbstractStringBuilder.java:421)
    java.lang.StringBuilder.append(StringBuilder.java:136)
    java.util.ResourceBundle.throwMissingResourceException(ResourceBundle.java:1564)
    java.util.ResourceBundle.getBundleImpl(ResourceBundle.java:1387)
    java.util.ResourceBundle.getBundle(ResourceBundle.java:773)
    weka.gui.visualize.Messages.getString(Messages.java:60)
    weka.gui.visualize.VisualizePanel.setUpComboBoxes(VisualizePanel.java:2393)
    weka.gui.visualize.VisualizePanel.addPlot(VisualizePanel.java:2487)
    weka.gui.explorer.ClassifierPanel$16.run(ClassifierPanel.java:1812)

```

Figure 5. The OutOfMemoryError happened when Weka tried to generate report.



Figure 6. J48 Decision tree

Classification Algorithm	Percentage Split (66%)
0R	63.62%
1R	65.64%
Naive Bayes Multinomial	68.0806%
PART	66.4899%
IBk	60.4454%
J48	65.3234%

Table 2

## Confusion Matrices (Percentage Split-66 %)

Text Mining**Naive Bayes Multinomial**

	<b>Neutral</b>	<b>Positive</b>	<b>Negative</b>
<b>Neutral</b>	31	21	132
<b>Positive</b>	7	78	74
<b>Negative</b>	30	37	533

Table 9

Naive Bayes multinomial was the most successful algorithm at predicting sentiment. The probabilistic approach of the Naive Bayes MultiNomial classifier is most likely why it was able to correctly predict a majority of the attributes.

**IBk k=1**

	<b>Neutral</b>	<b>Positive</b>	<b>Negative</b>
<b>Neutral</b>	94	21	69
<b>Positive</b>	39	76	44
<b>Negative</b>	167	33	400

Table 10

The nearest neighbor algorithm IBk came up with the least accuracy of the Text Mining classifiers and this may have to do with the fact that there are too many attributes and finding the distance of each attribute may be negatively affected by this.

**J48**

	<b>Neutral</b>	<b>Positive</b>	<b>Negative</b>
<b>Neutral</b>	36	33	115
<b>Positive</b>	8	86	65
<b>Negative</b>	39	67	494

Table 11

J48 was very good at accurately predicting the sentiment. The tree based approach is very good at learning which words to split on for each instance.

## Conclusion

Overall, most debate tweets are negative. This could just be the nature of the reactions to debates, but it could also be the sentiment of normal users on Twitter. In our data analysis, Text mining analysis was proved to be much better than using other nominal attributes to determine the sentiment of a tweet. Text mining had higher overall accuracies across the board, although it was only marginally better than the baseline and the nominal classification.

The rule based algorithms for nominal-based data mining were generally not effective. They either produced rules that are overfitting or used OR and predicted the class as negative. Nearest-neighbor gave a more predictive model based on a fairly large k value (50-100).

In text mining, a positive word suggests a higher chance that the tweet will be positive and the lack of a positive tweet suggest a higher chance of a negative sentiment. In the case of 1R in Text Mining, the simple positive word “\_thanks” was the most predictive word in determining sentiment. While our results did not yield especially significant accuracies for determining the sentiment of debate tweets, they did yield some interesting trends in mining Twitter for sentiment. In future studies the implementation of n-grams or associative learning algorithms may yield models that are better at Twitter sentiment analysis.