1 Introduction

Sentiment analysis is the computational approach to evaluating text, which includes determining both sentiment (positive/negative) and intensity. There has always been an interest in the idea of sentiment analysis or objectively evaluating text for polarity. But, the field has seen a burst of activity in recent years due to a few key factors - the availability of text through the world wide web, the advance of big data, and the improvement in machine learning techniques.

The question of sentiment analysis's real effectiveness is often brought up. The general consensus is that a ceiling does exist for sentiment analysis, much like a ceiling exists for human analysis. There are statements where humans disagree on the sentiment. The ceiling is statements that have near unanimous agreements. If most humans can agree on a statements, it's not a stretch to assume that an algorithm could come to a similar conclusion.

This paper aims to use sentiment analysis to "poll" twitter users. The model built will make the trade-off specific targeted questions for a large sample size. To make use of collected data, it's important that the sentiment analysis algorithm used is effective and fast; after all, it has to evaluate three million tweets.

2 Lexicons

The sentiment analysis this paper uses and examines are all rule-based model/algorithm. The algorithm recognizes positive and negative words, and then finds a method to average these words and determine an overall sentiment and intensity for the statement. But how does it know which words are positive and which words are negative? It uses a lexicon.

Lexicons are dictionaries. Large dictionaries which contain a list of positive and negative words, each with an intensity expressed. They allow algorithms to quickly ascertain the meaning of a word without having to re-define the word each time. A well-constructed lexicon allowed the model itself to remain simple. The model can be as simple as an average of all the positive and negative terms in the statement.

Lexicons can be broken into two types - semantic orientation or polarity based, and sentiment intensity or valence based. These general classifications cast a wide net and almost all lexicons fit in one or the other.

A well-known polarity-based lexicon is the Linguistic Inquiry and Word Count or LIWC (pronounced "Luke"). LIWC organized words into seventy

six categories and has about 900 words that are organized into two categories that are associated with emotion. It's the results of efforts by experts in psychology, sociology and linguistics. It's seen as arguably the best lexicon available, but does not recognize acronyms, abbreviations, or slang. These are all large parts of analyzing social text, and LIWC falls short.

The original valence-based lexicon is The Affective Norms for English Words or ANEW. It created ratings for over one thousand words of the english dictionary on a scale from one to nine, with a neutral midpoint at five. Words with a value greater than five will have a positive and pleasant meaning and negative and unpleasant otherwise.

3 VADER

The algorithm used to evaluate the data set is known as the Valence Aware Dictionary for sEntiment Analysis or VADER. It was developed by a pair of researchers CJ Hutto and Eric Gilbert at the Georgia Institute of Technology for the express use of evaluating valence in social media.

Social media and microblogging in particular poses a challenge to to sentiment analysis algorithms. This is the result of several problems. Earlier lexicons and algorithms were developed before social media's usage rates grew exponentially. They simply cannot handle the rate and volume that social media content is produced. Next, microblogging content is short. Twitter is limited to just two hundred and eighty characters. The average tweet is no more than a sentence or two. This limits context clues for traditional algorithms to pick up. Lastly, social media is famous for its abbreviations. Phrases like "LOL", "IRL", and "TBH" are commonplace on Twitter, but not commonplace on lexicons developed for traditional text. These are problems that VADER solves.

3.1 Construction

Hutto and Gilbert had four goals when they constructed the lexicon and the algorithm:

1º The algorithm performs effectively on social media text, but can still be generalized to all media sources.

2º The algorithm should have no training required from the user, but should still be based on a human created and cured lexicon.

3º The algorithm should be fast enough to use online.

4º There's no need to make a speed-performance tradeoff.

A basic overview of the VADER construction process is as follows. Hutto and Gilbert took the LIWC lexicon and made every single term in the lexicon a candidate for a new lexicon. Then, they proceeded to add social media specific terms which include abbreviations, phrases, and emojis as new candidates. These terms were trimmed and weighted in several stages until the final lexicon was produced.

The first stage of trimming used Wisdom-of-the-Crowd. WotC uses a large number of people to evaluate words in the lexicon. The large numbers combats the inexperience of the human raters themselves. Terms with a non-zero valence score were eliminated. Terms with a standard deviation greater than 2.5 which indicates that crowd couldn't reach a reasonable consensus were also eliminated. The terms were left had both a sentiment and an intensity. Terms such as "good" and "okay" while both positive, had a difference in score distinguishing their different meanings.

Their next stage used professional raters to identify parts of texts that can modify intensity despite not necessarily having an intensity themselves. These heuristics includes punctuation, verbal modifiers, and capitalization. Other heuristics identified were terms like "but" that signals a shift in sentiment and negated sentiment.

The last stage used professional raters to distinguish specific intensity modification of heuristics. This was done by taking a tweet, modifying either the syntax or the punctuation, and then randomly inserting these modifications into the dataset for the raters to evaluate. That way the algorithm could put a score on the change in intensity.

3.2 Effectiveness

The algorithm's effectiveness can only be evaluated by comparing it to the ground truth. The research team obtained ground truth statements for four domains - social media, movie reviews, product reviews, and editorials.

VADER's lexicon's competence was compared to several other well known lexicons such as LIWC, Bing Liu, Harvard Inquirer's, and more. This comparison made sure to use the same rule-based model for all of the lexicons. The three factors to evaluate the lexicons were:

- The correlation of computed intensity to the ground truth intensity.
- Precision - the number of items correctly classified divided by the number of total items.

- Recall - the number of items correctly classified divided by the number of items in the particular class
- F1 Score - the harmonic mean of the precision and the recall.

VADER outperformed some of the best lexicons in the industry currently. The results show that VADER's correlation was consistently higher, and it had the best F1 score for tweets. Hutto and Gilbert concluded that the reason was incorporating human heuristics into their model, essentially balancing a quantitative and qualitative approach.

3.3 Using VADER

VADER was written in Python, and is available to be implemented in Python. Below is how to use VADER to evaluate a single tweet.

```
from vaderSentiment.vaderSentiment import SentimentIntensityAnalyzer

analyzer = SentimentIntensityAnalyzer()
analyzer.polarity_scores("I'm so happy!")
```

This piece of code will return the following output:

```
{'neg': 0.0, 'neu': 0.318, 'pos': 0.682, 'compound': 0.648}
```

The package returns a series of scores, breaking down how the various components of the model's final score. Positive tweets are tweets that score greater than zero and negative tweets score less than. The intensity is determined the absolute value of the score.

A few interesting observations to note:

- Sarcasm is impossible for the algorithm to understand. To be fair, many humans also struggle with recognizing sarcasm in text.
- Any tweet that's an observation tends to be neutral, even if that observation has a positive or negative slant.

4 Building a Sentiment Profile

Each candidate in a given race has tweets about them. Using VADER, we can build a sentiment profile for them. The sentiment profile contains five statistics:

- Total Number of tweets
- Number of Neutral Tweets (Compound = 0)
- Number of Positive Tweets (Compound > 0)
- Number of Negative Tweets (Compound < 0)
- Average Compound Score

The sentiment profile will be the base of the features we'll use to model. Below is a snippet of code that calculates the profile for a given user.

```
sentiment = analyzer.polarity_scores(tweet)
if (sentiment['compound'] == 0):
neu_tweets += 1
if (sentiment['compound'] > 0):
pos_tweets += 1
if (sentiment['compound'] < 0):
        neg_tweets += 1
ave_score += sentiment['compound']
count += 1
```

This snippet exists in a for loop that iterates through all the tweets found about a candidate. To view the snippet in context, find build_sentime or sentimentFor the full script, refer to the Appendix.

Here's an example of a sentiment profile:

Candidate	Race	Count	Pos Tweets	Neg Tweets	Neu Tweets	Ave Compound
Ted Cruz	Texas Senate	111459	39642	33701	38116	0.0147602033
Beto O'Rourke	Texas Senate	34926	11445	3830	19651	0.1143955134

These profiles exist for every single candidate that participated in the election. You can find the sentiment profiles for house and senate candidates in Appendix C. The profiles are the data that will feed into the models that we'll build.