Kobe O’Neil
May 2019
LING 406

Final Project Report

I. Introduction

We, as humans, live in a society dominated by opinions of others, usually depicted in social media and from promotions from companies. The main issue lies in what we want to invest our time and money into, in which there are usually many ways of obtaining the information needed to enact on the user’s preferences. Moreover, one way that most people can easily obtain the information they need is through the Internet, whether from forms of various other entertainments or from blogs and much more. However, all of these sites or apps share one thing that overtakes the user’s decision to make their opinion: sentiments of others.

Sentiment analysis proves to be extremely important into today’s era, since everyone’s opinions can make or break a product or event’s ability to pull in revenue. For example, the *Marvel* movie, *Venom*, seemed like a great movie from the trailers given to the public a year ago. I thought the movie was going to be a hit, because it seemed like a nice step in the direction for the *Marvel* franchise since the *Avengers* series was coming to an end. However, upon asking my friends, they said that they hear from bigger-known YouTubers that they believe the movie was going to be bad. By the time the movie premiered in the box office, its mixed reviews made it lose a lot of revenue since it wasn’t able to convince those who denied at first.

Applications that can be implemented through the works of sentiment analysis are used in many apps that I use today such as Uber, Doordash / Grubhub / Postmates, ACM Theatre, etc. These apps give ratings based on consumer reviews, which also gives the implication to the user to choose restaurants or watch from theaters with high ratings and avoid theaters with low ratings.

In this assignment, I will be implementing a bag-of-words baseline system that will take in the list of movie reviews and outputs the percentage of which negative reviews were actually negative given the context and vice versa for positive reviews. For the different features, we’ll be using Naive Bayes Classifier, Logistic Regression Classifier, Decision Tree Classifier, etc.

II. Problem Definition

Sentiment analysis, in computational linguistics, is the process in which corpora or a group of words are given subjective or objective meaning. In subjective meaning, it is mostly based off of opinionated reasons, and in objective meaning, it is usually given as a comparison to two or more other items, which could in some cases be interpreted as factual. For example, taking a look back at Marvel's *Venom*, if a critic were to say: "This was a horrendous start of a possible grand movie series.", the sentiment given proves to be subjective, since it is the critic giving his / her opinion, and also gives a negative connotation to it. Now, if a different critic where to say: "Marvel's *Venom* had a shorter runtime than Marvel's *Avenger: Endgame*.", the critics notion proves to be factual, in which makes his / her connotation had to perceive as positive or negative. Thus, we give it a neutral connotation, neither for or against the movie.

However, that creates a problem at hand for consumers, in which most consumers don't want to hear a neutral review about a movie or about a restaurant, in this case for Yelp, (which we will look at later in my analysis). We, as humans, prefer subjective notions in order to make a decision to invest in something rather than an objective connotation.

III. Previous Works

In Sam Gibert's and Elsa Kim's *Detecting Sadness in 140 Characters: Sentiment Analysis and Mourning Michael Jackson on Twitter*, the two conducted experiments on tweets upon the death of Michael Jackson. They wanted to see and were curious about how people mourn over social media, in this case through the social media, Twitter. Their data set consisted of using multiple aliases that identify Michael Jackson, and filtered those tweets that contain Michael Jackson's name with association to the word "sad", which they split into three different types of categories: valence, arousal, and dominance. In this experiments, they they use an experiment that I found very similar to my approach (bag of words) in my data set, and found that majority of people did experience some form of sadness in his death via Twitter. They were able to conclude that their data set was a great tool in finding out how people truly felt about Michael Jackson after his death.

In Misailovic's and Yessenov's *Sentiment Analysis of Movie Review Comments*, the two make an analysis about messages send about a particular movie. In this experiment, or series of experiments to be more particular, they count in each message the positive correlation towards a given movie, the negative correlation towards a given movie, and the neutral connotation for a given movie. They used the exact approach I used for my analysis, but decided to use different features, such as negation and used a lexical database (WordNet) to take care of situations where there are multiple homonyms or synonyms can create ambiguity or even incorrect measurements for their testing data. They ultimately used Naive Bayes classifier, Decision Trees, and two I haven't heard of, in using K-means Clustering and Maximum-Entropy, and and found that all of these classifiers performed very well in their testings, although with WordNet, their results yielded no major significance. They were able to conclude that there bag of words approach did well and perform well with their dataset.

Finally, in Strapparava's and Mihalcea's *Learning to Identify Emotions in Text*, their methodization prove to be similar to techniques we used back in Assignment 2, in which the dataset was given as a gold standard and they used emotional labels as a means of finding emotions headlined in news articles as their dataset. Also in this dataset, they also use WordNet (this time, as their baseline model) and Naive Bayes classifier that was trained on the blog posts, which were ran against the gold standard. They also removed tags and kept the body paragraphs of the text as a means of preprocessing. Using this technique, they found that most blog posts show a sense of the word "Joy", followed by "Surprise", and "Angry" in third place. In their second run, they found in most runs for fine and coarse identifications that Naive Bayes classified and their initial baseline system found the most cases of having higher percentages of all the emotions they classified earlier.

IV. Approach

The bigger problem is that sometimes systems cannot pick up the actuality of a review and looks at the ratio between positive words and negative words, which can mess up the accuracy of the reviews.

As such, the task at hand is to create a sentiment analysis that gives only positive or negative reviews for movies or restaurants. In order to do this, we must input data (or bag of words) of positive and negative words into our machine and create a baseline system such that it can understand the good reviews from the bad.

For the process, I used the bags-of-words approach, in we first start with two sets of text data, the positive text and the negative text, which both are sited in my report in the Works Cited section. These sets of data will act as our “filter counters” for our baseline system. Since for our baseline in which we won't be using any classifier, my approach was to calculate the accuracy, recall, and precision of each groups of text (our datasets) given to us.

I did decide to pre-process the text, in which I removed punctuation, stopwords, and I changed everything to lowercase, which are also my features for this experiment. I decided to remove the punctuation by editing out the punctuation of each line to give a better sense of word choice per review. I also decided to remove stopwords since I found later, that they played a very big role in determining the accuracy as a lot lower than it should be and in which I would frustrate my results prior to receiving the results, dropping my accuracy, recall, and precision. I decided on using Naive Bayes Classifier, Logistic Regression Classifier, and Decision Tree Classifier, all using given using WEKA, though for the last two, I was not happy with how they turned out.

For the Yelp reviews, I decided to use the exact same approach, which would be using a bag of words method, consisting of having a positive list of words as text and a negative list do words as text. Like before, I pre-process the words using the same features I used for the movie reviews, if not already stated above. With that I attempted to use logistic regression and a decision tree classifier, but I was not happy with the results, perhaps due to lll implementation.

In other words, my approach is to first read the data from the bag of words, clean the review data of movie and Yelp reviews reviews respectively, train the data, and then test the data to calculate the accuracy, precision, and recall.

V. Results

In terms of my result, I was a bit disappointed in my implementation, especially in regards to my baseline model. For my baseline model, I was consistently getting different scores in between the 50 and 100% range for accuracy, and for recall and precision being in the 50%. When the data is ran, it will give a number between 0 and 1, which if you multiply by 100 will give you the percentage. I feel as though my implementation didn't necessarily show the inconsistency found in our training data, since I'm sure the data had instances where the reviewer used more positive words than negative, yet was given the negative review set. However, I do feel like in my Yelp review baseline model, that testing seem to be on a more consistent run, giving me values of around 70% for accuracy, 70% for recall, and 50% precision.

As stated before, I used WEKA for classifiers and feature separation analysis. In these analyses, I used Naive Bayes classifier, Decision Tree Classifier, and Logistic Regression, with features as punctuation, stopword, etc. I found the following results, using 5-fold.cross-validation like in Assignment 3:

Naive Bayes classifier: in terms of accuracy

- NB + all: 70.3%
- NB + punctuation: 75.2%
- NB + stopword: 68.9%

Decision Tree classifier: in terms of accuracy

- DT + all: 71.1%
- DT + punctuation: 73.4%
- DT + stopword: 70.7%

Logistic Regression classifier: in terms of accuracy

- LR + all: 67.6%
- LR + punctuation: 69.1%
- LR + stopword: 67.9%

Out of the three of these, I find it very hard to find a clear winner, considering that Decision Trees and Naive Bayes both had fairly overlapping numbers comparing their other features. Personally, I would give it to the Decision Trees classifier since it gave us the best accuracy, declaring that the best feature was with punctuation rather than with stopwords, which I find interesting.

VI. Discussion and Conclusions

In this project**,** I learned a lot about the natural language process, and using and gaining more experience with programming classifiers. I learned that out of all the features given, that removing punctuation in both the Yelp and movie reviews for the preprocessing section is the best / most important feature, when compared against removing stopwords and capitalization changes. Though with the classifiers that I used, in which all performed well against my dataset, removing punctuation helped add more significant percentages to my testing for recall, accuracy, and precision. Also, the size does matter when it comes to conducting the training test. For me, when my data set was low, my percentages were all over the place while with very large sets of data, they seem to mellow out as if it was a logistical regression model; this was for both cases when it came to the movie reviews and the Yelp reviews.

If I had to do this project again, I would have chose to add more features to my experiment and add more classifiers, as I intended to do but fell short of the mark. I feel like this could have given me a better distribution of which features matter when finding subjective matter, in the testing data that I wanted.

Going into a more detailed response for my conclusion, some linguistic representations that I feel could improve our model would be to use a lexicon model, like one of our previous works, that can have found synonyms and homonyms. I feel like if I could have done this again, that I will also include a lexicon model for use. I feel like one thing that is still left unsolved with my experiment is finding a better way to improve the accuracy despite having more negative / positive words in the opposite rated review (ex. Having more negative words than positive words in a positive review). I feel like a better way of finding those edge cases, would be also to implement a bigram / unigram model that keeps track of how many times a word shows up in comparison with other words. Though it could be difficult to implement due to the inconsistencies of word choice in our bag of words, I I feel that it could increase our accuracy, recall, and precision.

Works Cited

Carlo Strapparava , Rada Mihalcea. “Learning to identify emotions in text.” Proceedings of the 2008 ACM symposium on Applied computing, March 16-20, 2008, Fortaleza, Ceara, Brazil. http://www.cse.unt.edu/~rada/papers/strapparava.acm08.pdf

Kim, Elsa and Sam Gilbert. “Detecting Sadness in 140 Characters: Sentiment Analysis and Mourning Michael Jackson on Twitter” Web Ecology Project, August 2009. http://www.webecologyproject.org/2009/08/detecting-sadness-in-140-characters/

Yessenov, Kuat, and Sasa Misailovic. “Sentiment Analysis of Movie Review Comments” Massachusetts Institute of Technology, Spring 2009. http://people.csail.mit.edu/kuat/courses/6.863/report.pdf

Minqing Hu and Bing Liu. "Mining and Summarizing Customer Reviews." Proceedings of the ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD-2004), Aug 22-25, 2004, Seattle, Washington, USA,

Bing Liu, Minqing Hu and Junsheng Cheng. "Opinion Observer: Analyzing and Comparing Opinions on the Web." Proceedings of the 14th International World Wide Web conference (WWW-2005), May 10-14, 2005, Chiba, Japan.