#Sentiment Analysis of Hotel Reviews

Ping-Jung Liu
November 4th 2017

1. Introduction to Natural Language Processing

Natural language processing has become a prominent application field of data analysis and machine learning in recent years, mainly because of the huge advance in data processing efficiency and hardware capabilities. With the combination of cultural knowledges and technical approaches, people are able to perform immediate translation, part of speech extraction, or even estimate the trends of stock market based on recent economic reports.

Before the 1980s, natural language processing systems relied mostly on intricate hand-written rules by experts with a great number of linguistic knowledges. Without the required computational power to quickly organize and calculate data set, no matter how tremendous the set of rules was, the analysis tended to be inaccurate and time consuming. Few years later came the application of decision trees, but still the algorithm was half based on loads of if-statements because of the nature of decision trees. It was after experts used hidden markov models to extract part of speech information from documents that people started to appreciate the strength of statistical modeling in natural language processing. In fact, many of the most widely used translators like Google and Microsoft implemented statistical machine learning, including bayesian analysis, to provide translation, of course, with the help of millions of training data. Just like the field of artificial intelligence went from semantics to machine learning, natural language processing also shifted to more statistical systems in recent years.

In this project, I performed sentiment analysis on ten thousand hotel reviews from Yelp in hopes of figuring out underlying relationships between review attitudes and actual star rankings. The results will be presented with both explanations and graphs, and because the R codes I wrote to gather the data could potentially help understand the concepts of my analysis, I will include some codes in this report to elaborate on the program.

2. Sentiment Analysis

Sentiment analysis uses numerous techniques like natural language processing and text mining to systematically figure out and quantify the attitude of speakers or writers. Whether a business report is subjective or objective, whether the review written for that restaurant is positive or negative, or whether the population is satisfied with a policy announcement; these are all possible applications of sentiment analysis.

But unfortunately, current sentiment analysis still has trouble dealing with irony, which is the underlying tone of a sentence. Human language has extremely complex structures with great amount of grammatical differences. It is a miracle that we human beings have the ability to immediately understand almost 99% of conversations in real life. A simple example to show the gap between human and machine could be: “I just dropped my breakfast bagel on the ground, what a brilliant way to start a nice day.” Almost everyone could tell this person had a sad morning, while sentiment analysis would likely categorize this statement as a positive and happy one, because the sentence has “brilliant” and “nice” in it. But other than irony and subtle nuances, current sentiment analysis is able to calculate the overall attitude behind a whole document.

The fundamental task of sentiment analysis is polarity identification. Although a hotel review star point ranges from 1-star to 5-stars, I decided to only treat the reviews as positive (3 – 5 stars) or negative (1 – 2 stars). The statistical reasons behind this decision will be elaborated in later sections, but the main idea is that although there should be strong correlation between review sentiments and star point, the writing style of which people leave reviews varies too greatly that a 1 – 5 stars classification would likely be impossible in this project. Again, I will show the quantitative proof of this statement later.

For the purpose of this project, I will be using two lists of positive and negative vocabularies to identify the sentiment of each hotel review. Essentially, for each review, the program will find the number of matching words in the two vocabulary lists. For example, if a review reads: “I hate the pretty entrance” (I know this does not make any sense), the program would realize the number of positive vocabulary is one and that of negative vocabulary is one as well. Unquestionably, it would have been better to consider negations and N-grams as well, like identifying “I do not hate the dinner” as positive, but that would far exceed the scope of this project

3. Data Collection and Processing

As discussed in the previous section, three files are needed for the project: hotel reviews, positive vocabulary list, and negative vocabulary list. I will be using 13828 hotel reviews in Arizona recorded by Yelp on the Kaggle Competition. The data set was stored in “AZ hotels-ver2.csv”; the file contains columns about hotel names, reviews, dates, and star points. The two vocabulary lists were obtained from the paper “Mining and Summarizing Customer Reviews by Minqing Hu and Bing Liu” in the page http://www.cs.uic.edu/~liub/FBS/sentiment-analysis.html. The positive list contains 2006 positive words including some intentionally misspelled ones, while the negative list contains 4782 negative words.

The main reason I chose R was its NLP and tm (text mining) libraries, which allowed people to easily analyze documents by breaking them into arrays of words and counting their occurrences. The first step was to import the needed libraries and data set into R, using the following command:

```
library(openNLP)
library(openNLPdata)
library(NLP)
library(tm)
df = read.csv("AZ_hotels_bin.csv", header=T, stringsAsFactors = F)
doc = Corpus(VectorSource(df$txt))
pos_bank = readLines("positive-words.txt")
pos_bank = pos_bank[-(1:35)]
neg_bank = readLines("negative-words.txt")
neg_bank = neg_bank[-(1:35)]
neg_bank = neg_bank[-3008]
```

Corpus(VectorSource(txt)) transforms a String array into a data type that could be analyzed by the tm (text mining) library. Positive words were saved into pos-bank and negative ones saved into neg-bank. The other 1:35, -3008 commands cleaned the unwanted elements from the two vocabulary arrays.

Next step was to count the occurrences of each word in each of the review using the following command:

```
pos_dtm = DocumentTermMatrix(doc, control = list(
   dictionary = pos_bank,
   removeNumbers=T,
   removePunctuation=T
))

neg_dtm = DocumentTermMatrix(doc, control = list(
   dictionary = neg_bank,
   removeNumbers=T,
   removePunctuation=T
))
pos_dtm_mat = as.matrix(pos_dtm)
neg_dtm_mat = as.matrix(neg_dtm)
```

The method DocumentTermMatrix can count the number of occurrences of all the words included in the assigned dictionary, pos-bank and neg-bank in this case. They were then turned to matrices and saved in pos_dtm_mat and neg_dtm_mat. An example row in pos_dtm_mat: “review 1, admire:0 abound:0 affluent:1 amicable: 1……”.

Now that we have the occurrences of all these vocabularies, we can finally count the number of positive and negative words in each review with the following command:

```
for (i in 1 : 13827){
    print(i)
    for (j in 1 : 2006){
        if (pos_dtm_mat[i,][j] > 0){
            pos_count[i] = pos_count[i] + pos_dtm_mat[i,][j]
        }
    }
    for (j in 1 : 4782){
        if (neg_dtm_mat[i,][j] > 0){
            neg_count[i] = neg_count[i] + neg_dtm_mat[i,][j]
        }
    }
}
```

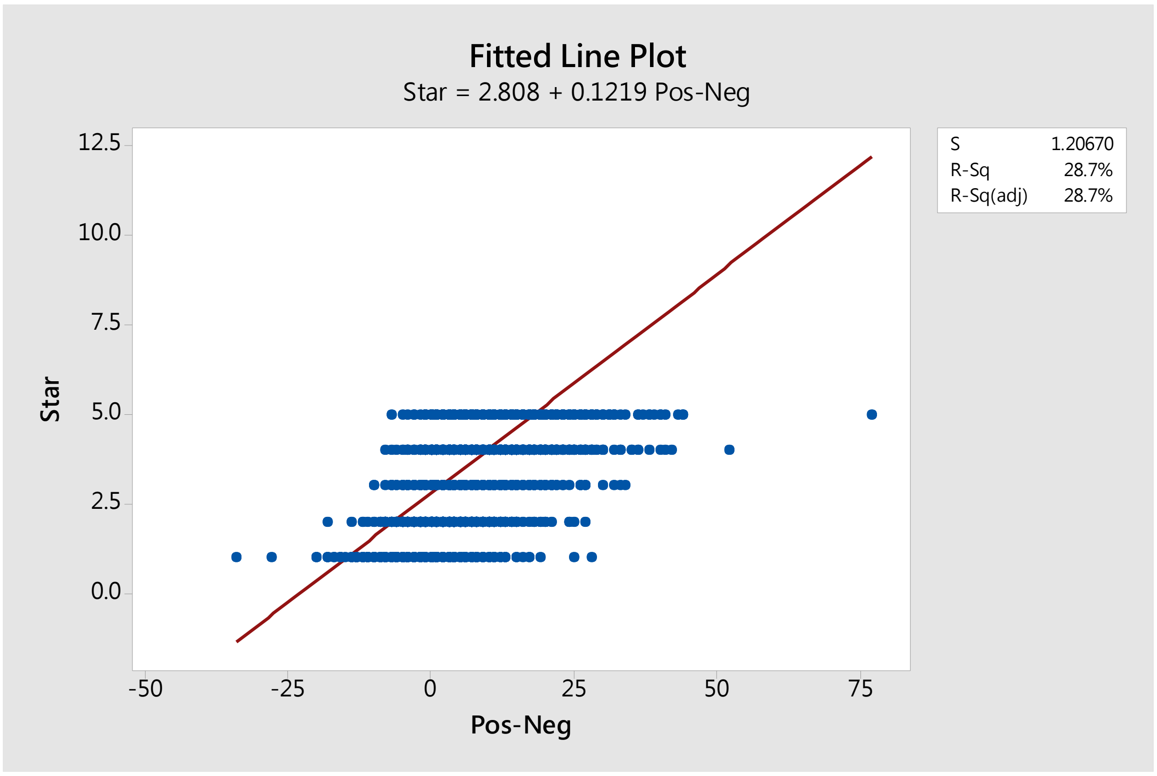
The data preparation was completed. The two arrays pos_count[] and neg_count[] now store the number of positive and negative vocabularies of each review. For example, pos_count[5] indicates positive words number in the fifth hotel review.

This is the messiest section because of the code demonstration; I just hope people can grasp the concepts better by examining the codes, which should not be hard to follow. The results from this section, pos_count and neg_count, were imported into Excel and Minitab for further calculations.

4. Verification of Sentiment Analysis

After the previous sections explaining the concepts of sentiment analysis and my data preparation process, people might still have doubt about the capabilities of sentiment analysis. In this section I will demonstrate its strength by performing a basic polarity analysis to predict the star points based on the pos_count[] and neg_count[] I obtained in section 3.

Though there are several other methods to perform this kind of classification process like logistic regression and naïve bayes, I chose polarity analysis because of its implementation simplicity and easy to understand procedure. To verify that the correlation between review sentiments and star points is infinitesimal, I performed line regression with star points being the response and “pos_word_count – neg_word_count”, which is a decent predictor for binary cases, being the predictor. The result is as follow. First, we can see R-square is 28.7%, which indicates the fit is indeed terrible. Second, even though line regression is not a decent model for this dataset, we can still observe that as star increases, the Xs, which are positivity, slightly shifts right as well. Third, it is obvious that only the extremes, i.e. Pos-Neg = 75, directly relates to the actual star point. Note that even Pos-Neg = 50 results in a 4-stars review. One way to interpret this phenomenon is that the extent of positivity and negativity of each vocabulary varies as well. “I love this place” has a higher chance of getting 5-stars than “We are mildly satisfied with this place. It was pretty ok.” This experiment is the main reason I proceeded with binary classification.



Though the correlation between actual star points and sentiments are low based on the fitted line plot, we could still observe the general shifts from negative to positive. The next section will further explore the advantages of binary classifications in the case of hotel reviews.

5. Dataset Analysis

Now that we have understood the basics of sentiment analysis, data preparations, and its potentials, we can start analyzing the data step by step. Below are the descriptive statistics star points generated by Minitab; we can observe that the mean is in fact slightly over 3, the median is 4, and Q3 is astoundingly a 5 when the highest point possible is also 5, which means more than half of the sample population left somewhat positive reviews for hotels, but the amount of words in the negative vocabulary bank is twice as large as that in the positive word bank: 4782 to 2006. We can conclude that although there are a wider variety of word choices for negative comments, people still tend to leave positive reviews more often.

I constructed a standard hypothesis test to check if the mean of star point is indeed greater than 3.

H0: u = 3, H1: u != 3

Z(0.025) = 1.96

- ⇨ Reject null hypothesis if u > 3.024
- ⇨ Sample mean is 3.435
- ⇨ Reject null hypothesis
- ⇨ 97.5% confidence that the mean of star point is greater than 3.

Descriptive Statistics: Star

Statistics

Variable	N	N*	Mean	SE Mean	StDev	Minimum	Q1	Median	Q3	Maximum
Star	13827	0	3.4351	0.0122	1.4292	1.0000	2.0000	4.0000	5.0000	5.0000

Next, I observed the descriptive statistics and probability plots of both positive word counts and negative word count. It is clear that the median number of positive words is more than three times that of negative words. Also note that the standard deviations of both are large compared to their respective means. One discrepancy is that the mean of star points is only 3.4. I speculated the usages of negation from positive to negative, i.e. “not good”, “not impressive”, greatly outnumbered those of the opposite direction, as we seldom used any negative to positive negations other than “not bad”. From the probability plots we can immediately realize that the numbers of positive and negative words are both not normally distributed. But this should not impede us from further analysis on the data, as later results will show that normality has little effect on the prediction results.

Descriptive Statistics: Positive

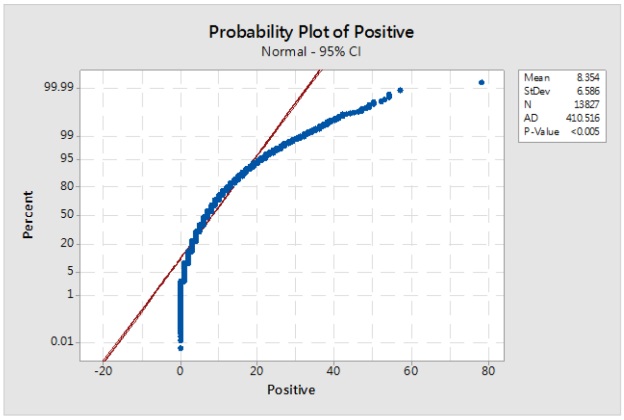
Statistics

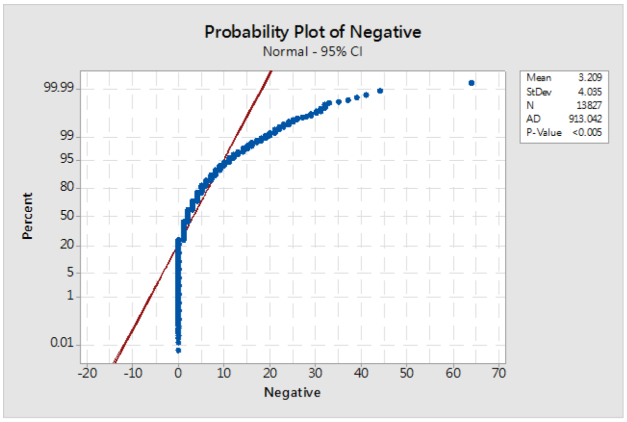
Variable	N	N*	Mean	SE Mean	StDev	Minimum	Q1	Median	Q3	Maximum
Positive	13827	0	8.3535	0.0560	6.5856	0.0000	4.0000	7.0000	11.0000	78.0000

Descriptive Statistics: Negative

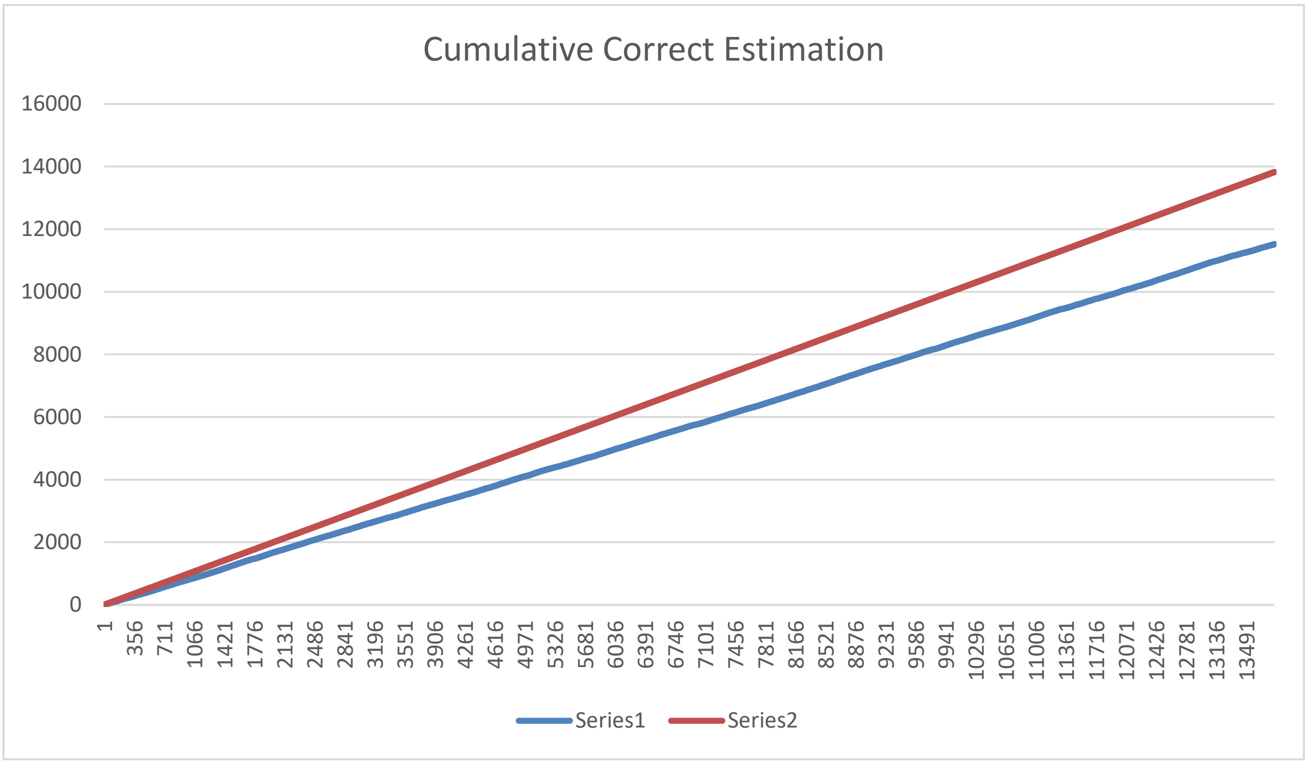
Statistics

Variable	N	N*	Mean	SE Mean	StDev	Minimum	Q1	Median	Q3	Maximum
Negative	13827	0	3.2087	0.0343	4.0348	0.0000	1.0000	2.0000	4.0000	64.0000





After studying the data thoroughly, even though the real data is not clean, and the medians of positive and negative word count varied, I could finally perform polarity analysis with the hotel reviews to observe potential relationships. Basically, for each review R, I subtracted pos-count[R], the number of positive words in review number R, by neg-count[R] to obtain its sentiment score. If the sentiment score is greater than or equal to 0, I would predict the review's star point to be 3, 4, or 5. If the score is a negative number, I would predict the star point to be 1 or 2. There is no formal proof for this classifier and it is resulted from logic and previous experience. Based on this classifying method, out of the 13827 reviews, 11521 were classified correctly, which is about an 83.5% accuracy. For such an unpolished model, I believe 83.5% accuracy is enough to show the soundness of this approach. I visualized the result with a cumulative sum line chart in Excel.



The X-axis indicates review number. Series 2 indicates the amount of reviews that the program have gone through, which is essentially the same as the X-axis. Series 1 indicates the amount of correct estimation made by the model so far. Though the slopes of the two lines obviously differed, it is also cleared that Series 1 closely followed Series 2 along the process. This strong result shows that we will be able to perform decent binary predictions even when the numbers of positive and negative word counts are not normally distributed.

6. Future Improvements

As we can all observe, the seemingly decent 83.5% accuracy was the results of numerous modification on question requirements in order to skirt problems like limited amount of reviews and not normally distributed dataset. The polarity analysis I performed was itself unsounded and merely based on logical thinking. Unquestionably, there are a number of ways to improve the process and results.

In terms of dataset, I would work hard to find a lot more reviews and extract those of similar lengths to prevent the means of positive and negative word counts from differing too much, if given more time. While normality did not affect the results profoundly, I believe resolving this issue could greatly boost the accuracy of predictions.

In terms of data preparation, I stated in previous sections that I would only be considering unigrams, that is treating each word individually. Though often times this approach serves the general purpose and many reviews can indeed be seen as a list of words, it completely neglected the existence of N-grams and phrases. But again it would be computationally impossible to consider unigram, bigrams, … , N-grams in a small scale project, so here I suggest a straightforward to include negations in our calculation. First, compile a short list of words commonly used for negations, i.e. not, no, don't, aren't. Second, for every sentence in a review, annotate every word after a negation word's appearance with a clear symbol. This would cause the original adjectives to be treated as different vocabularies. For example, given the sentence "I do not like this seemingly elegant restaurant." Would be transformed to "I do not n_like this seemingly n_elegant restaurant."

7. Conclusion

In this project, I wrote R codes to preprocessed 13827 hotel reviews gathered by Yelp, used statistical tools to observe the numerous features of the dataset, and finally performed a binary classification using polarity analysis. Though several concerns with respect to data quality and method rigorousness exist, the prediction result turned to be 83.5% correct, an acceptable accuracy.

Again, the objective of this project was not to design a superb algorithm that can 95% of the time predict the exact star points solely based on hotel reviews. The whole process demonstrates that even with limited dataset and computation power, there are always useful information within the seemingly unordered data that could explain observations.