6240 Text Mining Midterm Report

Jiatuan Luo
jluo324@gatech.edu
Georgia Institute of Technology
Atlanta, Georgia

Shuyu Ding
sding64@gatech.edu
Georgia Institute of Technology
Atlanta, Georgia

Qihang Zhang
qzhang333@gatech.edu
Georgia Institute of Technology
Atlanta, Georgia

KEYWORDS

Text mining, sentiment analysis, natural language processing, feature extraction

ACM Reference Format:
Jiatuan Luo, Shuyu Ding, and Qihang Zhang. 2020. 6240 Text Mining Midterm Report. In *Proceedings of ACM Conference (Conference'17).* ACM, New York, NY, USA, 4 pages. https://doi.org/10.1145/nnnnnnn.nnnnnnn

1 ABSTRACT

Customer reviews not only help buyers to make better purchase decision but also enable companies to improve their products. In this project, we plan to train a LSTM network on the *Amazon Reviews: Unlocked Mobile Phone from Kaggle* data set to classify the correct customer sentiment from given product reviews. Besides, we plan to perform characteristic extraction on the given reviews and summarize the customers' opinions on those characteristics and give business insights to the companies. The data set we used in this project contains 413,840 data entries and six features including product name, rating, brand name, price, reviews and review votes.

Through our analysis, we found out that customer satisfaction toward the product is positively related to the product price and negatively related to the length of the review. To be more specific, more expensive phones tend to get higher ratings, and more satisfied customers tend to leave shorter comments.

We trained three models as our baselines, which are Random Forest [1], K-Nearest Neighbors (KNN), and Naive Bayes[3]. Both the Random Forest and KNN models produced good results, which obtained weighted-F1 scores of 0.822 and 0.847 respectively. Naive Bayes performed worse than the previous two models which obtain an weighted-F1 score of 0.610.

2 DATA DESCRIPTION

2.1 Data Preparation

The dataset we used for this project is named *Amazon Reviews: Unlocked Mobile Phone from Kaggle* (https://www.kaggle.com/PromptCloudHQ/amazon-reviews-unlocked-mobile-phones). The raw dataset contains more than 400,000 instances with six features, which are **Product Name**, **Brand Name**, **Price**, **Rating**, **Review**, and **Review Votes (How many people found this review useful).**

The whole dataset contains 413,840 data entries. We first decided to remove all entries that did not leave reviews, and the total number dropped to 413,778. Since only about one-third of the data in **Review Votes** is non-zero (135,397), we determined this column is not useful for our project and dropped this column. 5,933 entries of the whole data have missing values in **Price**, and we imputed the missing values with the mean of the whole data. In the data preparation stage, in order to make the original data suitable for sentiment analysis, we first added a new column named **Label** that contains the true attitude of each review. The **Label** column is transformed from the **Rating** column, where a rating above 3-star is considered as a positive attitude, a 3-star rating is considered as neutral, and a rating below 3-star is a negative attitude.

After we have obtained the ground truth, we then cleaned up the reviews to make them suitable for sentiment analysis. To clean up the reviews, we performed HTML tag removal, non-letter characters, and stop-word removal. Furthermore, we also manually set up a list of synonyms and standardized those words (For example, *picture* and *photo* are synonyms and *picture* is replaced by *photo*). After that, we used NLTK tokenizer to split each review into sentences, and trained a word2vec model to get the word embeddings. Last but not least, we clustered the word embeddings into 10 groups using K-Means, and used the result to design our matrix for sentiment classification.

On the other hand, we performed product name standardization as well for characteristic extractions. First, we ran the same steps as for the reviews to clean up and tokenize the product name. We also manually created a bag of common words to remove from the original text. The group of words includes major color terms and other common words such as "unlocked". Then, we calculated the Inverse Document Frequency score of each term of each product name and selected the top 5 terms with the highest scores. Finally, we performed clustering to group those product names, and we are still testing on the best number of clusters to be used for this project.

The objective of this project is to train a good model that is able to classify the true sentiment of a customer regarding mobile phone products and get deeper business insight by looking further into customers' preference on the most significant phone characteristics and help the company prepare for their product iterations. This data set not only contains a sufficient number of product reviews (400k) for us to not only train the model but also split a part for validation. We also have enough reviews that can support us to do characteristic extractions and provide business insights.

2.2 Raw Data Statistics

After completing data preprocessing, we explored raw data statistics. Here is a brief summary of the important properties of our dataset.

Conference'17, July 2017, Washington, DC, USA

ACM ISBN 978-x-xxxx-xxxx-x/YY/MM…$15.00
https://doi.org/10.1145/nnnnnnn.nnnnnnn

- The total number of reviews is 413,778 and the vocabulary size is 60,689 after data cleaning.
- The average number of sentences per text is around 2.958.
- The average number of tokens per sentence is 13.92.
- After using fuzzy-wuzzy to solve partial data entry inconsistent issues, there are 292 different cell phone brand names.
- 72,337 (17.5%) are 1-star ratings, 24,724 (6.0%) are 2-star ratings, 31,763 (7.7%) are 3-star ratings, 61,374 (14.8%) are 4-star ratings, and 223,580 (54%) are 5-star ratings.

Table 1 shows the summary statistics of the price, the number of sentence per review, and the number of tokens per sentence.

	Price	Sentence Per Review	Token Per Sentence
Mean	226.867	2.958	13.921
Std Dev	273.019	4.257	14.446
Min	1.730	1.000	0.000
Median	144.710	2.000	10.000
Max	2598.000	222.000	742.000

Table 1: Summary statistics of the price, the number of sentence per review, and the number of tokens per sentence.

2.3 Data Analysis

First, we are interested in whether the proportion of positive ratings is different among some major brands. In terms of major brands, we selected the top 10 most frequently appeared brands in our dataset. From the figure below, we can see that among the ten brands, nine out of ten brands have above sixty percent of positive ratings and show clear differences between the proportion of positive attitude and negative attitude, but **CNPGD**. We would like to know how the company manages to operate as a popular phone brand even though the company has more dissatisfied customers. It turns out that the reason why the company is popular is because the price range of **CNPGD**'s phone ($29.99 - $99.99) is much lower than the mean and median price of the dataset. This observation leads us to our second data analysis approach, which is whether the phone price is correlated with the ratings. Figure 1 shows the proportion of positive/neutral/negative reviews of the top 10 most popular phone brands.

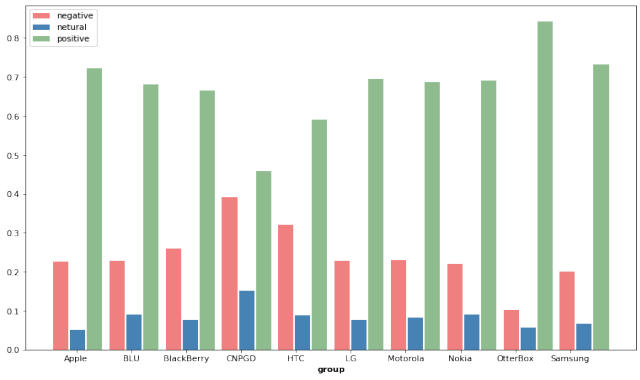

Figure 1: Proportion of positive/neutral/negative reviews of the top 10 most popular phone brands

Second, we looked into whether there is a correlation between the price of the phone and the sentiment of the customer feedback. The correlation coefficient between the price and class label is around 0.096. From the score we cannot conclude that there is a correlation between phone price and customer's opinion. Furthermore, we conducted ANOVA to see whether there is a true mean difference in phone price among the three classes. The F-statistics is 939.36 and the p-value is 0.0. From the below table, we see that the difference among the mean price of three opinion groups is statistically significant, where the mean phone price of positive reviews is bigger than that of negative reviews by $55.13, and bigger than that of neutral reviews by $41.92. On the other hand, the mean price of neutral review is lower than that of negative reviews by $13.21. Figure 2 shows the pairwise Tukey's HSD results.

```
Multiple Comparison of Means - Tukey HSD, FWER=0.05
===================================================
group1 group2 meandiff p-adj  lower    upper  reject
---------------------------------------------------
    -1      0  -13.212 0.001 -17.9499 -8.4741   True
    -1      1  41.9171 0.001   39.171 44.6633   True
     0      1  55.1291 0.001  50.8044 59.4539   True
---------------------------------------------------
```

Figure 2: The Tukey's HSD results regarding the mean price difference among the three label classes

Third, we were also interested to see whether a customer's opinion is correlated with the length of the review. Same as the previous analysis, we looked at the correlation coefficient and performed ANOVA. The score of the correlation coefficient is around -0.28, the F-statistics is 6890.85 and the p-value is 0.0. From the table below, we see that the mean length of positive reviews is statistically significantly shorter than that of negative and neutral reviews by 5.76 sentences and 5.86 sentences, while the difference between mean lengths of negative and neutral reviews are not statistically significant. Figure 3 shows the pairwise Tukey's HSD results.

```
Multiple Comparison of Means - Tukey HSD, FWER=0.05
===================================================
group1 group2 meandiff p-adj   lower   upper  reject
---------------------------------------------------
    -1      0  -0.0966 0.6174 -0.3427  0.1495  False
    -1      1  -5.8605  0.001 -6.0031 -5.7178   True
     0      1  -5.7639  0.001 -5.9886 -5.5392   True
---------------------------------------------------
```

Figure 3: The Tukey's HSD results regarding the mean sentense per review difference among the three label classes

Fourth, we explored into the pros and cons of relabeling the ratings into three classes. After relabeling, 284,954 (68.8%) reviews are positive, 31,763 (7.7%) reviews are neutral, and 97,061 (23.5%) reviews are negative. One main reason we did the transform is because we wanted to expand the class size of the non-positive class. However, after the transformation, we have obtained a larger negative class size but, unfortunately, a larger positive class size as

well. We will see how this form of labeling works in the classification steps and might apply specific techniques to solve the imbalance issue. Figure 4 shows the proportion of each class before and after the transformation.

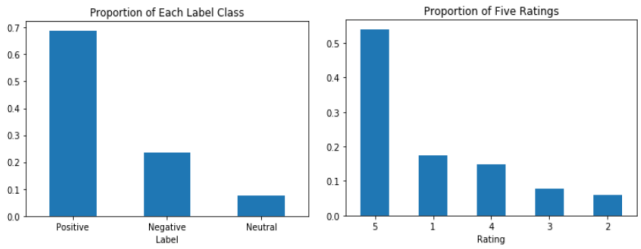

Figure 4: The proportion of each class before and after the transformation

Finally, we looked at the top 10 most frequent words in positive and negative reviews. We filtered all the reviews by setting a minimum length of three words per review. Surprisingly, we observed many overlapped common words in both groups. By further looking into the reviews, we realized that many reviewers either used sarcasm or negation words (such as "not good") when they made negative comments but "not" is considered as a stop word and has been removed in the data cleaning step. In the classification step, we will see how the results turn out and might consider not removing negation words for negative comments if it improves the classification results. Table 2 shows the top 10 most frequent words from both positive and negative reviews.

Word	Positive Review	Negative Sentence
phone	32833	99236
this	18079	52620
with	13138	38260
that	11011	32011
great	8579	25267
good	8407	24732
have	8131	23580
very	7455	22422
like	4608	13543
screen	4606	13764

Table 2: Top 10 most frequent words from both positive and negative reviews

3 EXPERIMENTAL DESIGN

We randomly sampled eighty percent of the data set as the training set and the rest twenty percent left as the validation set. Grid-search cross validation method was performed with k = 3 (number of folds). To evaluate the classification results, we decided to use accuracy score, precision, recall, and F1 score. All experiments were run on the CPU.

4 BASELINE RESULTS AND DISCUSSIONS

4.1 Baseline Description

To obtain a baseline for our experiment, we adopted word2vec to vectorize our text corpus to create the design matrix [2]. Then we selected three machine learning models that are inherently suitable for multi-class classification, which are Random Forest [1], K-Nearest Neighbors (KNN), and Naive Bayes classifier [3]. In particular, the Naive Bayes approach is inspired by [3], and for all three models we performed hyperparameter tuning and cross validation. The reason we used word2vec to generate our design matrix over other models such as bag-of-word, or term frequency inverse document frequency models is because it consistently produces the best result in terms of accuracy according to our tests and experiments. And the machine learning models we selected are among the most commonly used models applied to multi-class classification problems, which we believe should be sufficient to serve as our baseline.

The helper code and packages we relied on to produce our results include Python's NLTK, gensim, and scikit-learn libraries.

The hyperparameters we optimized include K, the number of neighbors for KNN, and max_depth, min_samples_split, min_samples_leaf for Random Forest. We were only able to look at a limited number of hyperparameters due to constraints of in time and computing power. However, we will perform a more extensive search for optimum hyperparameters in the next step. We did not try to optimize the parameters for the Naive Bayes model.

4.2 Baseline Result

We applied Grid Search CV method to perform cross validation and hyperparameter tuning, the best parameters returned are:

- K = 5 (number of nearest neighbors)
- min_samples_leaf = 1
- max_depth = 15
- min_samples_split = 2.

The four metrics we used to measure model quality - accuracy, precision, recall, and the F1 score (all weighted) as well as the confusion matrices are shown below in Table 3 and Figure 5, 6, 7.

Model	Accuracy	Precision	Recall	F1
Random Forest	0.815	0.838	0.815	0.822
KNN	0.862	0.846	0.852	0.847
Naive Bayes	0.674	0.628	0.674	0.610

Table 3: Accuracy, precision, recall, and F1 score (all weighted) of the three models

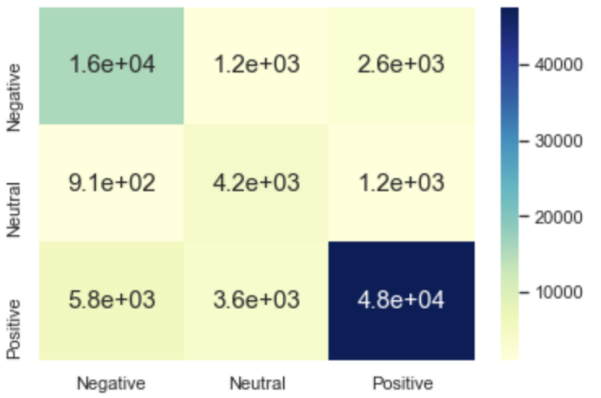

Figure 5: Confusion matrix of the random forest model. Row is the ground truth, and column is the prediction.

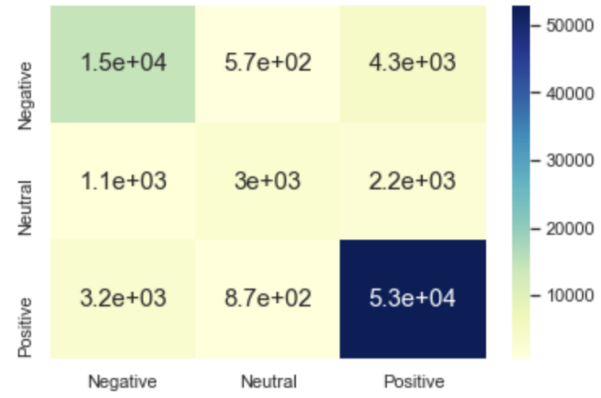

Figure 6: Confusion matrix of the KNN model. Row is the ground truth, and column is the prediction.

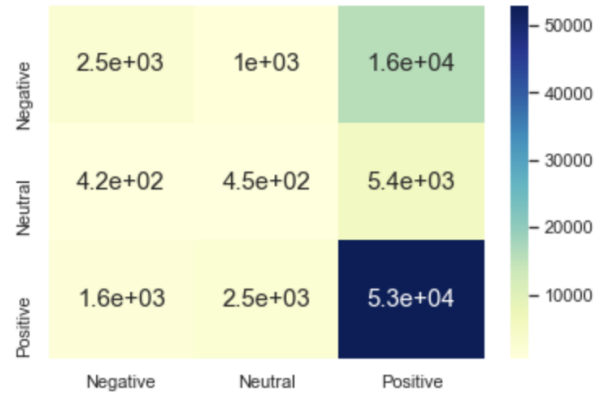

Figure 7: Confusion matrix of the Naive Bayes model. Row is the ground truth, and column is the prediction.

4.3 Result Discussion

According to our result, KNN yielded the best model across all three metrics, and Naive Bayes produced the worst model. We also took a dive into the confusion matrices and discovered that the models were particularly good at identifying positive comments, but did not so well at identifying neutral comments. This finding is consistent across all three models. We suspected that this is very likely due to the imbalance of the dataset, since we have significantly more positive entries than the other two classes. We also found that neutral comments were difficult to classify for all three models. Interestingly, a larger number of neutral comments were misclassified as positive than negative. From this we extrapolated that comments that gave three star ratings tend to either lean toward positive sentiment or negative sentiment, with a larger number leaning toward the positive. In other words, more comments with three star ratings are more similar to positive comments than negative comments. We will dive more deeply into these findings and account for such problems in the following steps of our project.

5 NEXT STEPS

Next, we will perform more extensive hyperparameter tuning and employ other sampling techniques to account for the imbalanced classes to see if we can further improve our classification results, and we will run the classification process with a LSTM network to see if applying a recurrent neural network will improve the performance. On the other hand, we will extract characteristics from the reviews and determine customers' opinions on them. Finally, we will summarize the result by each phone brand and give business insights to the companies. In terms of feature extraction, two major steps will be conducted. First, we will perform POS tagging on all reviews to find the nouns (NN) and noun-phrases (NNP) and filter out the major occurred characteristics. Second, we will manually create a group of features to consider and a group of nouns that have high occurrences in reviews but are not features to ensure the quality of our feature extraction. Finally, we will calculate the sentiment score of each characteristic and summarize a business report based on the result.

6 CONTRIBUTION

All the team members contributed equally to this project.

REFERENCES

[1] Leo Breiman. 2001. Random forests. *Machine learning* 45, 1 (2001), 5–32.
[2] Andrew L Maas, Raymond E Daly, Peter T Pham, Dan Huang, Andrew Y Ng, and Christopher Potts. 2011. Learning word vectors for sentiment analysis. In *Proceedings of the 49th annual meeting of the association for computational linguistics: Human language technologies-volume 1*. Association for Computational Linguistics, 142–150.
[3] Lin Zhang, Kun Hua, Honggang Wang, Guanqun Qian, and Li Zhang. 2014. Sentiment analysis on reviews of mobile users. *Procedia Computer Science* 34 (2014), 458–465.