# STAT 605: Project Proposal

**Kou Wang** kwang432
**Shushu Zhang** szhang695
**Xinyue Wang** xwang2438
**Yiqun Xiao** yxiao85

**Sixu Li**
sli739

## 1 Introduction

We are working on Amazon review data, with including over 35 million reviews up to March 2013. The data set contains different categories with 10 variables, including product and user information, ratings, and a plain text review. We are interested in *mining which words have strong relationships to the five star rating and which ones highly correlate to the negative reviews.* Specifically, we separate the data into different categories such as *movie* and *electronic*, each of them contains 10 variables. Therefore, among the 10 variables in different categories, we mainly explore the relationship between user text reviews and ratings. Since the reviews contain different information for different categories of the products, we break up the entire data set into 28 trunks by categories. Representatives for different sizes of the 28 trunks are characterized in Table 1. For each chunk, we first use some natural language processing (NLP) techniques, such as Document-Term Matrix (DTM) representation, TF-IDF to clean the data. Then, we use regularized lasso logistic regression to fit the model with cross-validation. These chunks are designed to be ran on HPC. We implement the code in R and run the code on Slurm with the sizes of the chunks scaling up. As a result, we are able to provide useful suggestions to all the business owners listed based on the significance of positive and negative words, except for Movies & TV, which will be left for future research.

Table 1: Chunks of the Entire Data set

| Chunks (Categories) | Size |
|---|---|
| Cellphones | 70MB |
| Kindle Store | 191MB |
| Sports & Outdoors | 368MB |
| Home & Kitchen | 758MB |
| Electronics | 1.1GB |
| Movies & TV | 8.8GB |

# 2 Statistical Analysis

## 2.1 Data Description & Data Cleaning

The web data is provided by Jure Leskovec, an associate professor from Stanford University. It's a data set containing about 35 million reviews from Amazon. The corresponding time spans from June 1995 to March 2013, a period of 18 years. The size of whole data set is 11 gigabtyes and it is available on website Amazon reviews. We mainly focus on user review texts and review scores (on a 1-5 scale).

In order to extract useful information from the reviews, we need to preprocess the text data using NLP techniques. We (1) transform the target reviews to corpus; (2) change to lower case; (3) remove all the numbers; (4) remove all the stop words; (5) strip whitespace; (6) remove punctuation; (7) change to Document-Term Matrix (DTM) representation (i.e., reviews as rows, and words as columns, frequencies of the occurance of a word for each text review as values of the matrix); (8) remove sparse terms from a Term-Document Matrix with 0.99 threshold (i.e., remove all the words that occur less than 1% of the number of the words) ; (9) use TF-IDF (Term Frequency-Inverse Document Frequency) to transform the matrix (with details filled below); (10) obtain binary response variable by replace all the review scores of 4,5 to 1, and 1,2,3 to 0; (11) use bigram frequencies; (12) split training and test data.

Here are some details about TF-IDF. For DTM, if a word occurs more often than others in general, it is more likely to occur more frequently in a review, such as "movie" in our case, which is unfair to other words. Therefore, we use TF-IDF (Term Frequency-Inverse Document Frequency) to offset these effects. TF-IDF can be mathematically characterized by

$$\begin{aligned} TF(\omega, t) &= \frac{\#\omega \text{ in t}}{\#\text{words in t}} \\ IDF(\omega, t) &= log(\frac{\#\text{words in t}}{\#\text{reviews that contains } \omega}) \\ TFIDF(\omega, t) &= TF(\omega, t) * IDF(\omega, t) \end{aligned} \tag{1}$$

where $\omega$ is a word and $t$ represents a review. Figure 1 is a sample of our word counts

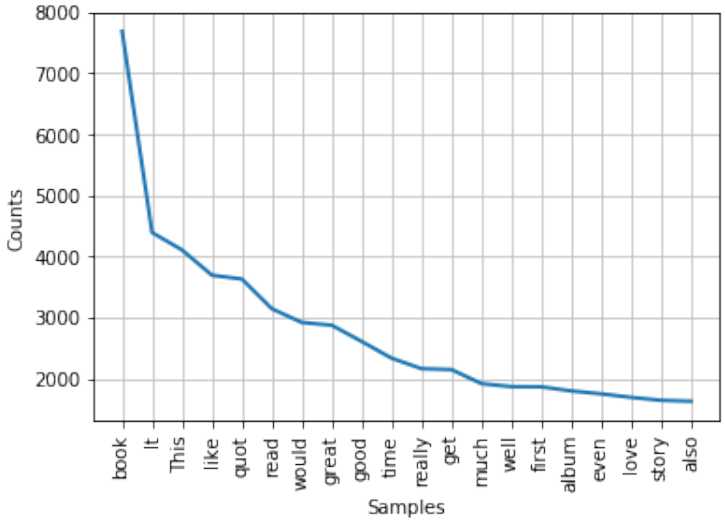
## 2.2 Statistical Models

Lasso logistic regression is selected as the statistical model to apply on our document matrix. Since lasso procedure encourages simple and sparse models, we have reduced dimensions in the previous data preprocessing procedure.

### 2.2.1 Logistic Regression

Logistic Regression is one of the few algorithms that is used for the task of Classification of data. Since we are interested in mining which words have strong relationships to the five star rating and which ones highly correlate to the negative reviews, and in regression analysis, the logistic regression model is a form of binary regression and fits well in our purpose. In

Figure 1: word count samples




this project, logistic regression is estimating the parameters of a logistic model. Consider a generalized linear model function parameterized by $\theta$,

$$h_\theta(X) = \frac{1}{1 + e^{-\theta^T X}} = Pr(Y = 1|X; \theta)$$

We are assuming that all the observations in the samples are independently Bernoulli distributed.

### 2.2.2 Lasso

Although we reduced our data dimension, it is still not enough to perform regression analysis. Lasso regression is a type of linear regression that uses shrinkage, which stands for least absolute shrinkage and selection operators. We decide to use the Lasso linear model function parameterization for our logistic regression. Lasso regression performs L1 regularization, which adds a penalty equal to the absolute value of the magnitude of coefficients. This kind of penalty fits well in our circumstances. Because this type of regularization can result in sparse model and the coefficients can be eliminated from the model. Larger penalties like L2 regularization (e.g. Ridge regression) can not result in elimination of coefficients or sparse models. By Lasso regression, we substantially reduced the number of variables.

$$\begin{gathered} \hat{\beta}^\lambda = argmin_{\beta \in \mathbb{R}^p} ||y - X\beta||_2^2 + \lambda ||\beta||_1 \\ \hat{S}^\lambda = \{k : \hat{\beta}_k^\lambda \neq 0\} \end{gathered} \tag{2}$$

where $X \in \mathbb{R}^{n*p}$ is the design matrix, $y \in \mathbb{R}^n$ is a vector of outputs, and $\lambda$ is a regularization parameter that controls the size of the selection set.

By implemented methods above with our training dataset provided by previous step, we fit our models and build our findings.

## 2.3 Computation

To implement this project, we ran our codes on HPC requesting different sizes of memory accordingly. In order to implement our code safely on HPC, we designed a scale-up method and increase the dataset gradually to protect computing power. We randomly separate our dataset into size of 70MB, 191MB, 368MB, 758MB and 1111MB for testing.We want to make sure our codes can be safely running for small dataset so it would be safe running on HPC with large requesting memory. Figure 2 and 3 below illustrating the linear relationship between different sizes of dataset and the memory usage, and also the running time.

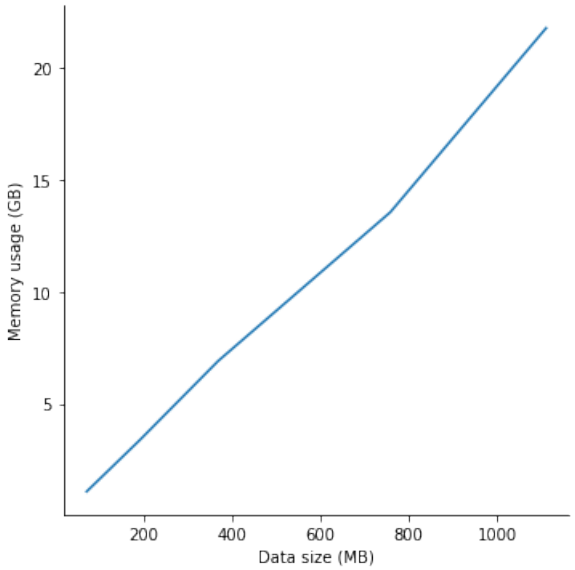



Figure 2: Memory Usage

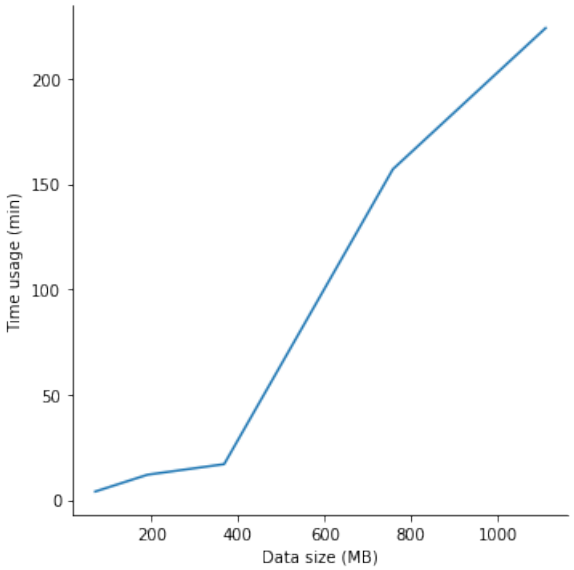



Figure 3: Time Usage

We can see that both running time and the memory usage increase with the size of the dataset.However, we failed running one of our biggest datasets, which has a size of 8.8 GB. A detailed research has been done on this case and we believe there are two reasons that cause this result.

The first reason is that our dataset is a sparse matrix. This is a common problem in NLP. A lot of words only appear few times in a few reviews. And it is even common that there are words only appear once. To solve this problem, we remove the variables which have more than 99% sparsity. This method works well in our small testing dataset. However, it cannot handle 8.8GB data. We assume this is because, first, the sparsity removing method require huge memory in large dataset, and second, the percentage to decide the removing of the variable, which is 99%, is too big.

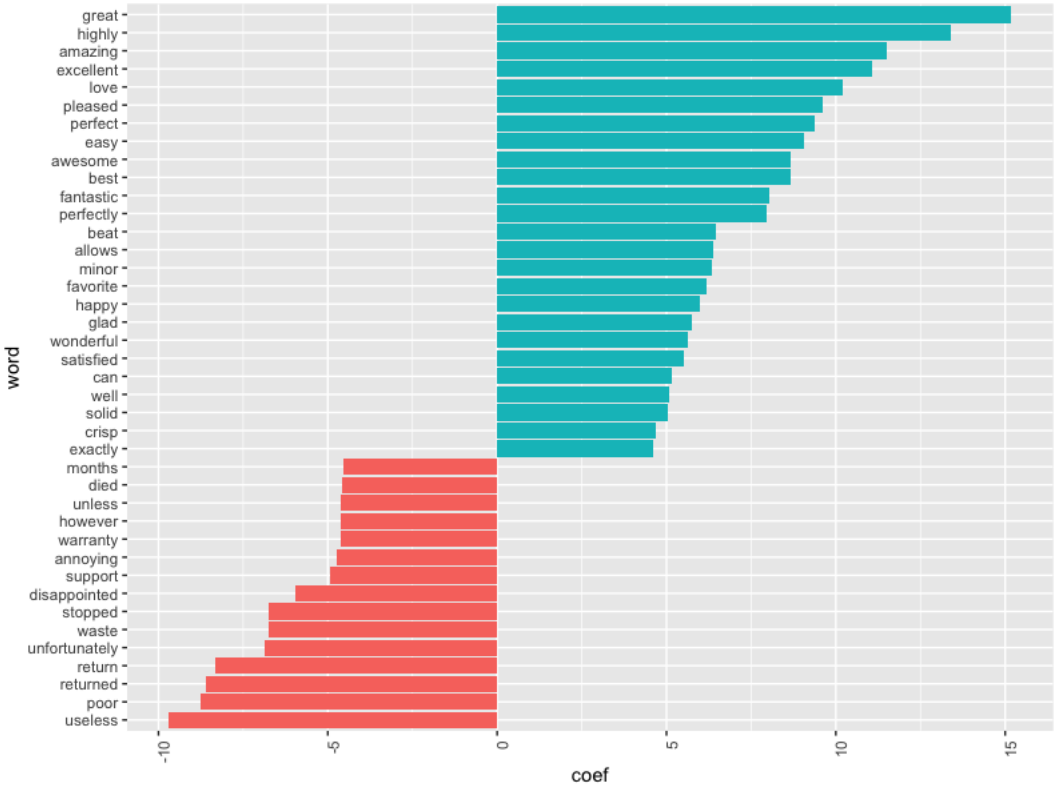
The second possible reason is that this chunk is too big and it exceeds the suggesting size 4GB. We have used two methods to reduce the computation load. First we unleash the unnecessary memory promptly, and second we convert the word matrix into a sparse matrix. In our 70MB dataset, these two methods reduced the memory usage from 4GB to 1.08GB.

# 3 Findings

We finally found several word collocations associated with positive/negative review. Figure 4 illustrates our findings.

- great, amazing, excellent, love, pleased, perfect, awesome and other positive words associated with positive reviews, as we expected. Surprisingly, complaint, far, crisp, minor are also associated with positive reviews.
- Typicla negative words associated with negative reveiws are diffcult, slow, error, work, annoying, disappointed, waste, poor and so on. One interesting finding is that words like warranty and support are also associated with negative reviews.

Figure 4: Top 100 positive and negative words




We found that some words are seemed positive by ourselves but they are associated with negative reviews, which is interesting. And we also found some collocations do not make any sense.Other interesting findings:

- The number of reviews could be an index for measuring the popularity of an commodity
- The length of reviews cannot be the index for measuring the popularity of an commodity

Figure 5: Word cloud


# 4 Conclusion

To draw our conclusion, in this project, a textual dataset with about 35 million reviews from Amazon with a time span from June 1995 to March 2013 is analyzed. By cleaning the data and reducing the dimension, performing the DTM procedure, we obtained our word matrix as our explanatory matrix and then perform the lasso logistic regression, finally build our result: the words collection associated with positive/negative reviews.The HPC is used as our computational tools and we found that the size of data set has linear relationship with the memory usage.