STAT 503 Final Project

Jim Curro, Eric Hare, Alex Shum

Apr. 26, 2013

Introduction

The dataset that we have includes reviews of the different businesses written by yelp users in the Phoenix, Arizona metropolitan area in the time frame of January 13 - March 12, 2013. Our analysis is using the reviews given to us and identify which reviews were classified as useful based on only the number of stars the user gave the business, the total number of reviews by the user, the average star given by the user, and the text data of the review. Using this data, we fit an SVM in order to predict the usefulness of the test set of reviews.

Data

We began by taking the unparsed json files from Yelp and producing three main datasets: business.json, user.json, and reviews.json.

business.json has information for every business in the Phoenix area that yelp has recorded and includes the following variables (name of business, business id, address, type of business, # of reviews, # of checkins, # of stars). Because of the aim of our study the business dataset will be used as an initial data analysis to examine the differences between businesses and try to identify any potential concerns in the data. Concerns we will be aware of include an employee from the business giving the business many false positive reviews of a business. Also we can use this data to identify differences among businesses and analyze businesses that have a high percentage of reviews per check in.

user.json has all of the different information for each user that yelp keeps track of. Within this dataset we have for each user the variables (name of user, user id, # of funny reviews, # of useful reviews, # of cool reviews, # average stars, total # reviews). Different from

the business.json data there are some important variables in this dataset that is given in the test set that we can use to better our analysis of useful reviews. By looking at the total number of reviews the user has and their average star count possibly we can identify a trend with these two variables and see how it pertains to predicting whether the users reviews are classified as useful or not. There are also other variables given in this dataset that we can use to further explore our data that is given, however those will not be useful in our final analysis of predicting useful reviews.

reviews.json is our primary dataset for predicting whether a review is useful or not because it has every review recorded and all of the different information recorded by yelp for each review. These variables include (review id, user id, business id, # votes funny, # votes useful, # votes cool, # stars given, date of review, text). Based on the goals of our study we will only be concerned with whether the review was useful, the stars given by the user, and the text data to identify the unique aspects of a review that others find to be useful.

Initial Data Analysis

Figure 1 displays the number of useful votes against the number of cool votes in blue, and the number of useful votes against the number of funny votes in red. It can be seen that while both cool and funny votes are great predictors of useful votes, the slope is larger for funny. In other words, we would expect that the reviews with a set number of funny votes would tend to have more useful votes than those with the same set number of cool votes. Note, however, that we ultimately chose not to use funny and cool as predictors because the Yelp rules disallowed this.

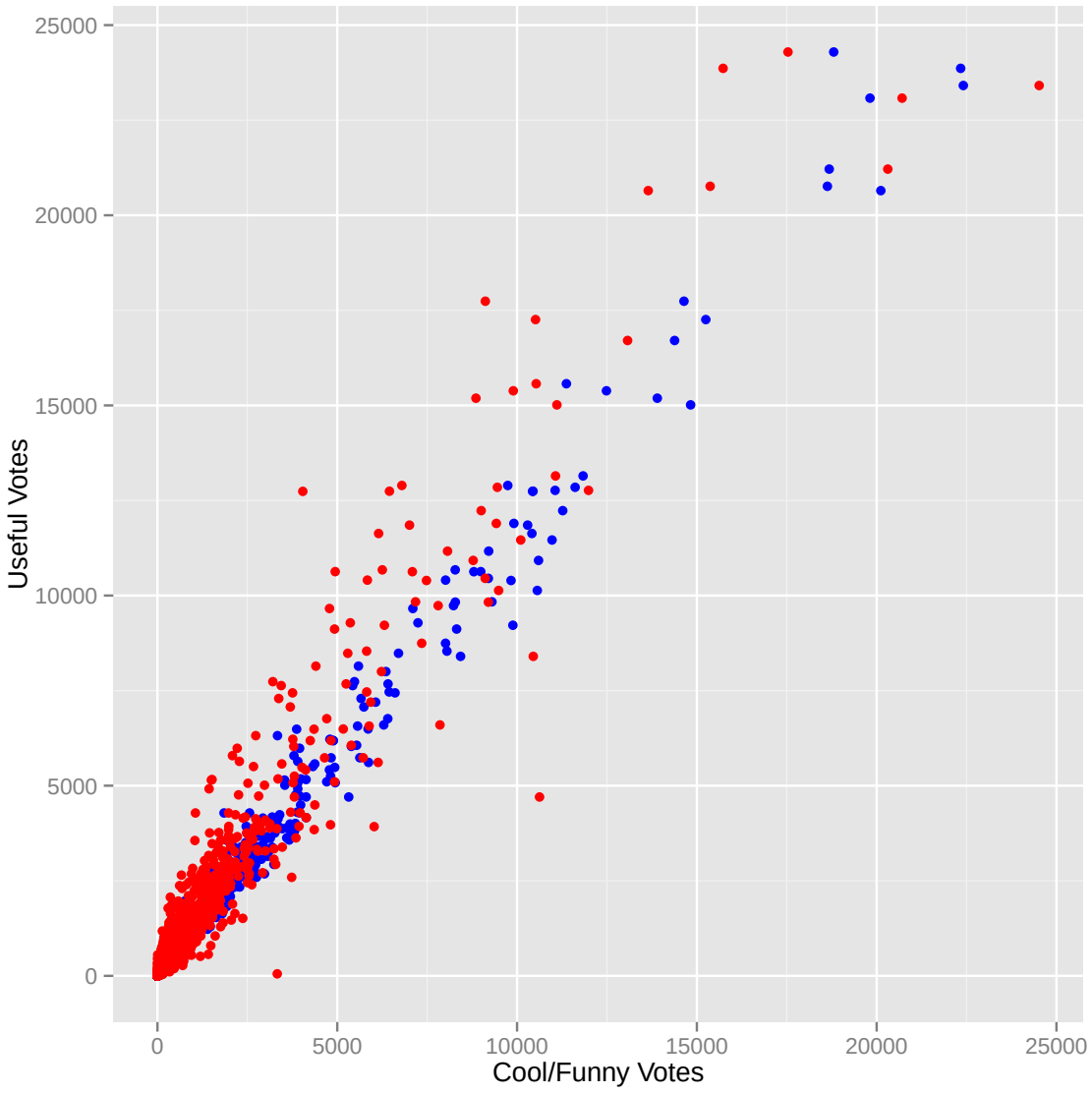

Figure 1: Useful votes vs cool (blue) and funny(red) votes. This plot indicates that a review which is voted as being funny and/or cool is also likely to be voted as being useful.

Table 1 shows the top 10 cities by number of checkins in the database. It also shows the total number of reviews for all businesses in that city, as well as the percentage of reviews over checkins. It can be seen that for the biggest cities in the database, there is a very uniform number of about .27 reviews per checkin.

city	reviews	checkins	percentage
Phoenix	91453.00	310640.00	0.29
Scottsdale	49042.00	202946.00	0.24
Tempe	27021.00	93446.00	0.29
Chandler	13944.00	48067.00	0.29
Mesa	9786.00	34328.00	0.29
Glendale	7017.00	26306.00	0.27
Gilbert	5803.00	22723.00	0.26
Peoria	2599.00	11361.00	0.23
Surprise	1274.00	5109.00	0.25
Avondale	1243.00	5053.00	0.25

Table 1: Top ten cities by the number of checkins in that city

Figure 2 shows boxplots of the number of checkins to each business by the average star rating of that business. It can be observed that businesses with about a four star rating tend to have the most checkins, but five star rating businesses actually have fewer. This may be either due to the fact that businesses with such a high rating have fewer total reviews, or they are more expensive and less likely to have a high number of customers.

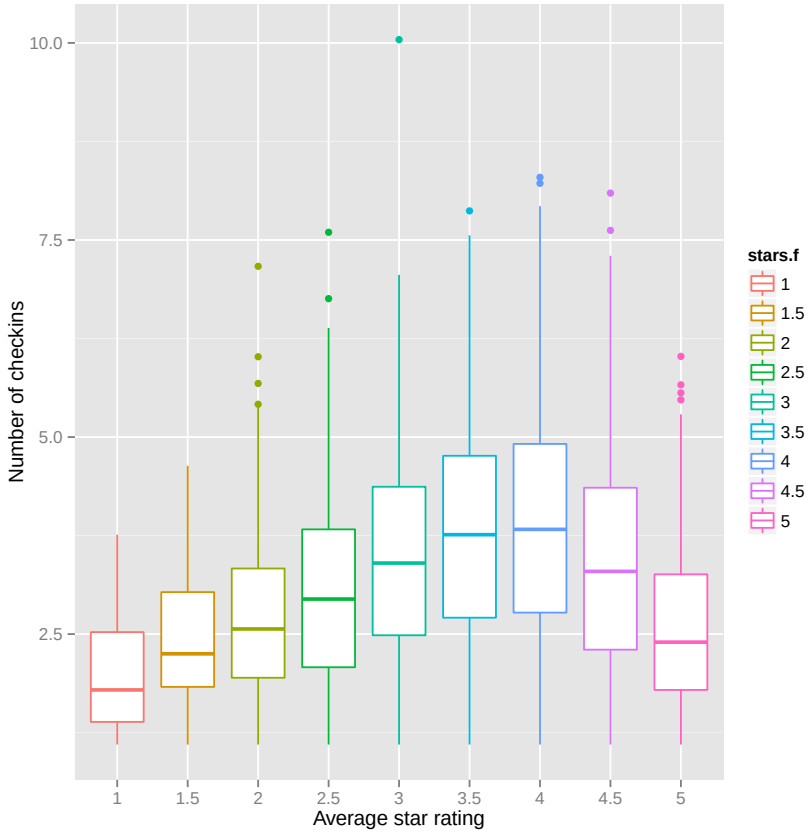

Figure 2: Number of checkins to each business by average star rating of that business

Exploratory Data Analysis

Table 2 displays the top ten users by the total number of useful votes divided by the total number of reviews for that user. We have selected only users who have made at least 100 reviews. We felt this would provide us with valuable information about what characteristics the "best" reviewers possess.

funny	useful	cool	name	average_stars	review_count	good
20707.00	23080.00	19815.00	Hazel	4.06	814.00	28.35
13074.00	16707.00	14381.00	Katie	4.19	770.00	21.70
11070.00	13146.00	11836.00	Chris	3.77	624.00	21.07
9488.00	10134.00	10563.00	Robin	3.79	530.00	19.12
7481.00	10396.00	9832.00	Raider	4.24	546.00	19.04
1993.00	2586.00	2600.00	Clyde	3.90	143.00	18.08
5391.00	6063.00	5539.00	Marlon	3.92	346.00	17.52
2544.00	2631.00	2638.00	William	3.97	153.00	17.20
10451.00	8401.00	8429.00	Thomas	3.67	517.00	16.25
9003.00	12233.00	11270.00	Jennifer	4.08	764.00	16.01

Table 2: Top ten users in the Yelp data by total number of useful votes per review (Minimum 100 reviews).

Figure 3 shows the number of useful reviews by the average number of stars for each user, colored by the user's total number of reviews.

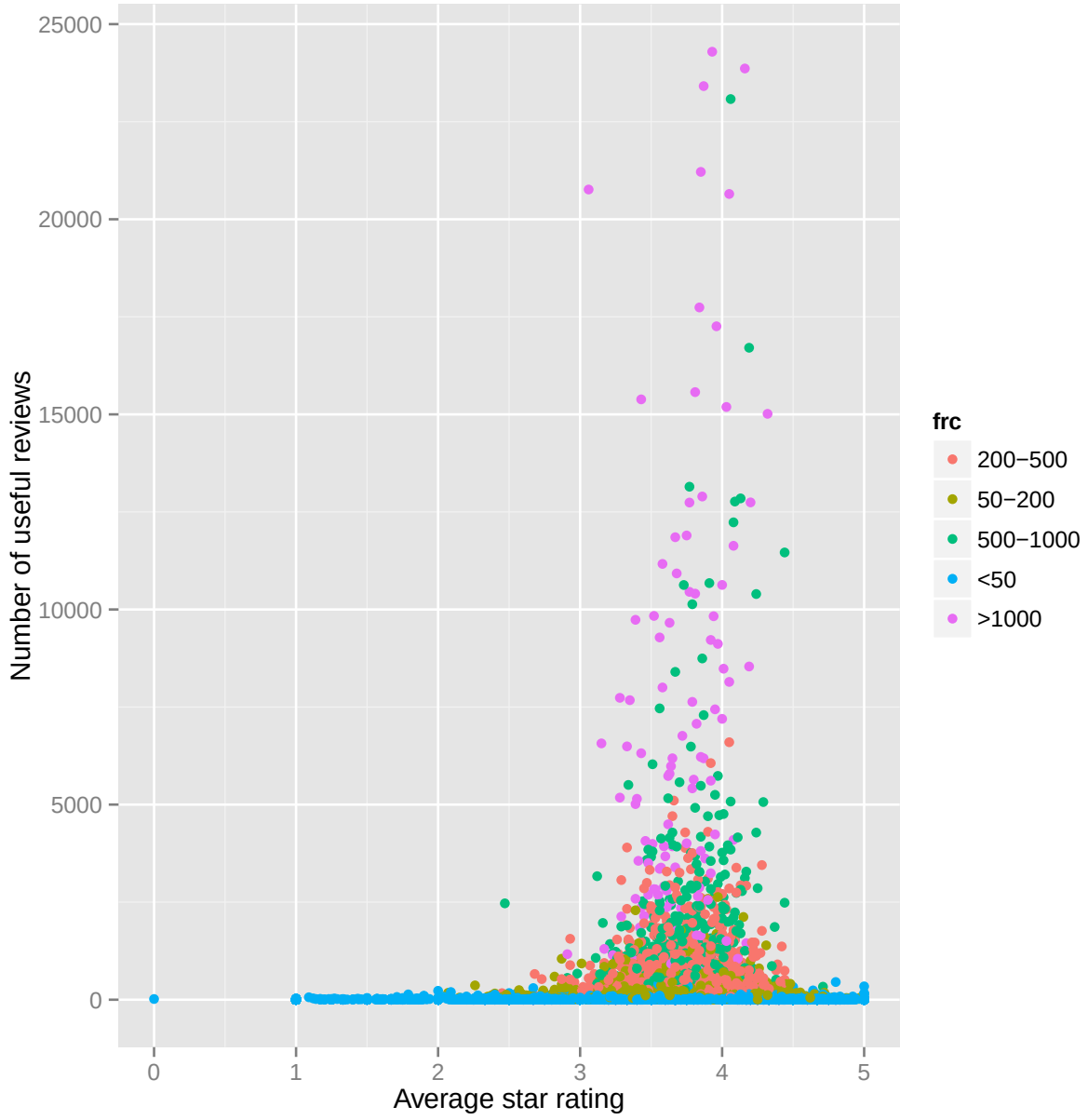

Figure 3: Displays for each user their number of useful reviews by the average stars that each user gives any review. The users are colored by the total number of reviews showing a clear trend in number of reviews and number of useful reviews

The results of this plot are very encouraging that the average stars a user gives will be a strong predictor of useful reviews. The plot demonstrates that users with average stars given between 3 and 4.5 are most likely to have reviews that are the most useful. It also shows that users with more total reviews are highly likely to have more useful reviews.

Preprocessing and Classification

The original collection of reviews was provided by Yelp with minimal preprocessing. Starting with the text reviews written by yelp users (our corpus), we removed whitespace, put words into lowercase, removed all punctuation, removed stop words, and stemmed words to put them into the same form. A standard text model is the bag of word model in which a corpus is represented as a term-document matrix; each word that appears in the corpus is a column and individual reviews (documents) are rows. From our original corpus we only included words that appear in at least 10 reviews. The resulting term document matrix is a 229,907 by 27,876 sparse matrix.

Our response variable is the number of useful votes given to a particular review. In order to turn this into a classification problem we categorized reviews as useful if at least one person voted it useful. Roughly half the data has zero useful votes and the rest of the reviews have more than one useful vote. ¡add in description of response variable¿

R Memory Issues

Building a term document matrix, reducing dimension using singular value decomposition and classifying based on the rows or columns of the matrix is a fairly standard method in text mining. Unfortunately we ran into a number of memory issues in R.

The tm package in R by default outputs sparse matrices. Many of the classification algorithms as implemented in R do not support sparse matrices. It was also not possible to convert sparse matrices into regular matrix objects in R due to the size and dimensionality of our dataset.

An alternative approach was to further reduce the number of dimensions. After removing stop words and running the corpus through a stemming algorithm we removed overly sparse words. Words that appeared in fewer than 10 documents were removed. We were able to reduce the number of columns from 27,826 to around 5,000. However the resulting matrix was still too large to fit in memory as a normal matrix object. We were able to reduce the dimensions further by sampling 20,000 documents from the original 229,907. The resulting matrix could fit into memory but this subset of our data was still too large to train an SVM or random forest.

We considered sampling from an even smaller subset of the dataset but we were already reluctant to use a dataset of this size due to the dimensionality and sparsity. If memory constraints were not an issue we would need to construct a term document matrix not only with the training data but also the test data. Unfortunately it seems weve reached the limits of base R.

Classification

Since the text mining approach is not feasible in R we classified based on some summary statistics of the text data. We computed summary statistics on an 18,000 review subset of the original unprocessed text of the reviews and then tabled the results by whether or not the reviews were voted as useful (seen in Table 3).

	useful_bin	numChar	numCap	numPunc	numPar	hasDont	hasTime
1	FALSE	536.65	0.03	0.04	0.00	0.14	0.20
2	TRUE	830.09	0.03	0.04	0.00	0.21	0.26

Table 3: Summary statistics for all reviews by whether the review was voted as useful or not. The variables include the number of characters, percentage of letters capitalized, percentage of the review containing punctuation, the percentage of reviews that contain the word dont, and the percentage of reviews that contain the word time

The variables we considered were number of stars a review gave, number of characters in a review, number of capital letters used, number of punctuation marks used, number of paragraphs and whether or not the review contains the words "time" and "dont". We reasoned that longer reviews contained more useful information and useful reviews are more likely to contain proper punctuation, proper capitalization and more likely to be structured in paragraphs. We also calculated the word frequencies for useful reviews versus the word frequencies for unuseful reviews. Based on word frequencies we calculated the probabilities of words that are likely to appear in useful versus unuseful reviews. We found that the words dont and time are much more likely to appear in a useful review than an unuseful review. Additionally, we reasoned that a review with a higher star rating would be more likely to have a useful explanation to justify the star rating.

We initially fit all of the variables into a random forest classifier and used mean decrease in accuracy and mean decrease gini from the random forest algorithm to find the most important variables. The results are displayed in Table 4.

	FALSE	TRUE	MeanDecreaseAccuracy	MeanDecreaseGini
stars	0.01	0.00	0.00	299.63
numChar	0.05	0.03	0.04	1502.66
numCap	-0.00	0.00	0.00	1188.36
numPunc	0.00	0.01	0.01	1216.28
numPar	0.04	0.00	0.02	994.32
hasDont	0.01	-0.00	0.00	85.99
hasTime	0.01	-0.00	0.00	97.42

Table 4: A list of the variables and their importance as determined by the randomForest algorithm. We ultimately selected three of these variables, numChar, numCap, and numPunc for use in our SVM.

From the previous importance results, we fit a model on an 18,000 document subset of our original dataset to predict usefulness based on the number of characters, capitalization and number of paragraphs. We tested our model on a separate randomly selected 2,000 document subset of our original dataset. Our overall accuracy is 0.3728%. The full classification results can be seen in the confusion matrix displayed in Table 5.

	FALSE	TRUE
FALSE	467	371
TRUE	352	810

Table 5: Truth Table for the results of our model