Predicting usefulness of online reviews using stochastic gradient boosting and randomized trees

Madhav Kumar[1] **Shreyes Upadhyay**[2]

[1] Fractal Analytics,
New Jersey, USA
Email: madhavkumar2005@gmail.com

[2] Diamond Management and Technology Consultants
Mumbai, India
Email: shreyes.upadhyay@gmail.com

Abstract

This paper presents our analysis of online user reviews from different business categories posted on the internet rating and review services website Yelp. We use business, reviewer, and review level data to generate predictive features for estimating the number of useful votes an online review is expected to receive. Unstructured text data are mined using natural language processing techniques and combined with structured features to train two different machine learning algorithms - Stochastic Gradient Boosted Regression Trees and Extremely Randomized Trees. The results from both of these algorithms are ensembled to generate better performing predictions. The approach described in this paper mirrors the one used by one of the authors in a Kaggle competition hosted by Yelp. Out of 352 participants, the author stood 3^{rd} on the final leaderboard.

Keywords: gbm, helpfulness, online user review, opinion mining, randomized trees, text mining

1 Introduction

Virtual social infrastructures have created a market place for opinions where insights are exchanged to facilitate informed consumer decision making. From an economic perspective, these opinion markets tend to be both efficient and effective with potential safeguards against information asymmetry for the consumers and a low cost marketing channel for the producers. Between the consumers and the producers are e-commerce websites which are either directly (e.g., Amazon[1], ebay[2], App store[3]) or indirectly (e.g., Yelp[4], FourSquare[5], IMDB[6]) involved with the sales of the product. Understanding this tripartite structure and the dynamics behind it is crucial for both producers, since these opinions tend to have a direct impact on product sales (Duan et al. 2008, Ghose and Ipeirotis 2011), and for e-commerce retailers, given that the cost of consumer migration from one retailer to the other is negligible.

The principal constituent of these opinion markets are online reviews posted by consumers based on their experience and/or knowledge of the product. Online reviews provide a wealth of information about the characteristics of the product and its quality. These pieces of information help potential consumers in making an informed choice before purchasing the product. Hence, it is important that this information be available to the consumer in a easily digestible and succinct manner. However, on popular websites like Amazon and IMDB, this information can be spread out in multiple pages with thousands of free text reviews, not all of which are equally relevant. Digging through this text overload to get to relevant pieces of information can be inefficient and needless to say, extremely difficult.

How does one then decide which information is relevant and which is not? To put this into perspective of online buyers and e-commerce retailers, how does one decide which reviews are useful and which are not? Amazon answered this question and hit a $2.7B jackpot (Spool 2009) when it provided users with the option of voting for a review that they found helpful. Most websites that host review services include this feature now. This partially solves the problem of filtering out useful reviews from the others. However, among the ones that did not get helpful/useful votes, which ones were unhelpful and which were simply not read? This information is not easily available and by not accounting for it, the system as a whole becomes inefficient by losing out on potentially useful data. Given the competitive nature of such websites, it is imperative that the user experience be as smooth as possible with the most relevant information be available with minimum cognitive effort.

In this paper we have attempted to solve the generic problem faced by e-commerce retailers and review service websites – of minimizing text overload by prioritizing information based on usefulness. Specifically, we try to identify how many viewers will find a particular review to be useful. For empirical analysis, we used online reviews posted on Yelp. We then mined the text using natural language processing techniques and trained machine learning algorithms to estimate the number of useful votes a review is expected to receive. The approach described in this paper is largely what one of the authors followed while participating on a online data mining competition

[1] www.amazon.com
[2] www.ebay.com
[3] https://itunes.apple.com/us/genre/ios/id36?mt=8
[4] www.yelp.com
[5] www.foursquare.com
[6] www.imdb.com

conducted by Yelp on Kaggle[7].

Kaggle is an online portal that hosts data mining competitions for large corporations, start-ups, governments, and universities. Work on Kaggle is largely competitive where the data are twisted and turned to squeeze out tiny pieces of information for marginal improvements. Hence, not all methodologies applied there are applicable in a research driven setting. Keeping this in mind we present a pruned version of the approach that will provide a business and statistical view of the problem along with a thorough explanation of our solution.

This paper is divided into 8 sections. Section 2 describes the data mining problem followed by the literature review in section 3. Sections 4 and 5 present a detailed description of the data and our pre-processing work respectively. Section 6 describes the modeling process along with the algorithms used. Section 7 presents the results and section 8 concludes the paper.

2 Problem definition

The data on websites like Yelp are built on three verticals - 1) the viewers, 2) the reviewers, and 3) the businesses. Since this is a dynamic portal, the role of viewers and reviewers is interchangeable. At any given point of time, viewers are essentially the current consumers of Yelp and the potential consumers of a business. They view what different businesses have to offer and make their consumption decision based on the reviews posted by reviewers. To facilitate this decision making process, Yelp provides viewers with three pieces of information – 1) a quantitative field which denotes the average star rating of the business till date, 2) a second quantitative field (multiple instances) for the star rating given to the business by a particular reviewer, and 3) actual text written by reviewers describing their opinion about the business. Many reviews provide a detailed description of the reviewer's personal experience and a justification of the star rating. However, not all of these opinions are useful to viewers. While some are very well written providing enough details about the business and the experience, some are curt statements with a few words of praise or critique, and some are just spam.

Reviews for popular businesses can go up in hundreds with a certain proportion falling in each bucket as described above. Such volume of data leads to information overload for the viewer. On top of it, the presence of unhelpful matter and spam makes the problem worse by wasting the viewers' time as they dig through useless text. To overcome this problem Yelp provides viewers the option of voting for reviews that they find helpful. The votes can be assigned to one or more of the following three attributes:

1. Was this review useful?
2. Was this review funny?
3. Was this review cool?

Aggregation of votes over time allows easy distillation of reviews based on their usefulness to viewers. When a new viewer then visits the page for the business, the most useful reviews can be shown on top for fast and effective decision making.

Using the votes as a indicator for usefulness is a fast and easily scalable method. However, judging usefulness based on this sole criterion would not be statistically robust or economically efficient even though it might be directionally correct. This methodology raises at least three concerns that we present below:

Age of review (in days)	Mean of useful votes
[0, 30]	0.741
(30, 180]	0.814
(180, 540]	1.037
(540, 1000]	1.199
(1000, 2875]	2.228

Table 1: Mean useful votes with increasing age bracket

Inflated useful votes – reviews that get more useful votes tend to get more attention from other viewers which leads to further increase in the number of useful votes. Such a cycle inflates the true usefulness of the review as compared to the other reviews only because they receive more visibility which eventually leads to over-shadowing of the usefulness of less read reviews.

Age bias – reviews tend to accumulate votes over time. Any review that has been recently posted might not get noticed, especially if the business already has a good number of reviews. Such situations reinforce the point above for reviews that do not get highlighted due to over-crowding of older reviews that have been voted useful by many viewers. Table 1 presents this scenario with the mean of the number of useful votes received split by age brackets. The age bracket shows range of the difference in days between the review draft date and the data snap shot date.

Not useful vs. not read – Yelp does provide viewers the opportunity to vote for reviews that they find useful. However, there is no way to highlight whether the other reviews were not useful or were simply not read. Unlike Amazon or IMDB, Yelp does not have a ballot to vote for reviews that were not useful. For example, a viewer might read the first 2–3 reviews and vote them to be useful. This does not imply that the reviews below were not useful; they might not have received the attention of the viewer. This again follows in line with the first point that reviews that initially receive a few useful votes tend to develop on those votes to receive more and the cycle continues.

We believe that the search for solutions to these problems motivated Yelp to host an open data mining competition. The goal of the competition was to identify reviews that viewers would find most useful. Statistically, the objective was to predict the number of useful votes a review is expected to receive based on the business, reviewer, and review level information.

3 Literature review

There has been considerable research under the umbrella of opinion mining each with a slightly different

[7] www.kaggle.com

flavor of definition, data, or algorithm but widely different presentation of results. Many previous studies have taken data from Amazon (Forman et al. 2008, Kim et al. 2006, Ghose and Ipeirotis 2011, Liu et al. 2007, Mudambi and Schuff 2010) as the base for their analysis. In addition, data from CNETD (Cao et al. 2010), IMDB (Liu et al. 2008), and Epinions (Turney 2002) have also been used for analyzing opinions. This is understandable considering that these websites are the forerunners in their respective categories and have a large user base.

While there has been consensus on the sources of data, the definition of the problem has witnessed remarkable variation. Kim et al. and Ghose et al. estimate the helpfulness of a review by defining it as the ratio total number of helpful votes divided by the total number of votes (Kim et al. 2006, Ghose and Ipeirotis 2011). Defining the problem in terms of the helpfulness ratio creates an imbalance since only reviews that have received any votes can be considered. For example, Liu et al. only consider reviews that have received at least 10 votes (Liu et al. 2008). A different approach is taken by Ghose & Iperiotis who convert helpfulness to a binary variable by mapping the helpfulness ratio to 0–1 based on a threshold value (Ghose and Ipeirotis 2011). Cao et al. convert the problem to one of ordinal logits and model on the number of helpful votes with an upper bound at 7 (Cao et al. 2010). While this approach circumvents the helpfulness ratio imbalance, it places an arbitrary artificial cap on the number helpful votes a review is expected to receive. Liu et al. go a step further by manually categorizing reviews according to their helpfulness as either "best", "good", "fair", or "bad" (Liu et al. 2007). They define the helpfulness criteria based on the specification quality of the review. Though they have invested considerable effort in understanding the helpfulness of reviews, their working sample is small (4,909 reviews) and limited in scope to digital cameras only.

In terms of analyzed features, most variables from previous papers can be categorized as per the of classification Kim et al. into – a) structural (e.g., punctuation, review length, number of words), b) lexical (e.g., n-grams), c) syntactic (e.g., part of speech tagging), d) semantic (e.g., information about the product), and e) meta-data (e.g., user and product level information) (Kim et al. 2006). For example, Cao et al. describe their features as basic, stylistic, and semantic (Cao et al. 2010). Their basic features include the age of the review in days, the extremeness level of the review, and whether the reviewer wrote in different sections of the review. Their stylistic features are similar to the structural ones mentioned above, and the syntactic features are a mix of the semantic and lexical from Kim et al. Ghose et al. use the reviewer data as well in their analysis. Specifically, they create features from the reviewer's profile and reputation (Ghose and Ipeirotis 2011). In addition, they also include information on readability and subjectivity of the review.

Much of the relevant literature is dominated by the use Support Vector Machines (SVM) (Burges 1998). Using the radial basis function (RBF) to predict helpfulness of reviews seems to have provided promising results (Kim et al. 2006, Liu et al. 2007, 2008). Ghose et al. deviate from this by first doing an exploratory study using 2SLS with instrumental variables (Wooldridge 2002) to predict the logarithm of helpfulness (Ghose and Ipeirotis 2011) and then building a binary classification model using Random Forests (Breiman 2001). Similar exploratory approach is used by Forman et al. except that they model directly on helpfulness without transforming it (Forman et al. 2008). Cao et al. add more variety to the choice of algorithms by using ordinal logits for their predictive model (Cao et al. 2010).

To summarize, considerable effort has gone into developing the intellectual capital surrounding opinion mining with each new piece of research building upon the previous one and increasing the community knowledge base in this area. In our paper, we extend the scope of these previous research efforts by – 1) considering reviews that are not limited to a single category and thereby eliminating any industry or product specific bias; 2) removing the helpfulness ratio imbalance by analyzing all reviews irrespective of whether they have or have not received any useful/helpful votes; 3) applying two different machine learning algorithms and ensembling their outputs in an effort to improve the accuracy of the predictions and reduce variance of the system.

4 Data

Data were provided by Yelp for 252,863 reviews out of which useful votes for 222,907 reviews were available for training. These reviews were subset from the universe on Yelp in two ways – 1) spatially by considering businesses only in the state of Arizona and 2) temporally by considering all reviews up till Jan 19, 2013 for training and from Jan 20, 2013 till Mar 12, 2013 for testing.

The data were stored in 4 file sets - review, user, business, and checkin. Each file set had two files, one for training and the other for testing. A description of these file sets is provided below.

4.1 Review

The review data were unique at a review id level. They contained the actual review in the form of free text along with the following structured attributes – review draft date, star rating given by the reviewer to the business, user id of the reviewer, and the business id of the business. The training portion also included the number of useful, funny, and cool votes received by the review. The values in the votes variables were the cumulative number of votes received for each attribute by the review till the data snapshot date – Jan 19, 2013. From the three votes variables, the number of useful votes was the target and much of the attention in this paper is focused on this variable.

Table 2 provides the frequency of the number of useful votes received by reviews in the training data. About 95,000 (41%) reviews had not received any useful vote and about 65,000 (28%) reviews had received only 1 useful vote.

4.2 Reviewers

There were 51,296 reviewers present in the data set. For most reviewers, Yelp stores information on the total number of funny, useful, and cool votes received along with the name, average rating given by them to different businesses, and total number of reviews posted. However, some users can choose to keep their profiles private, in which case, none of the fields mentioned above are populated. In addition, to prevent an easy mapping of the number of useful votes received by a particular review and the total number of useful votes received by all the reviews written by a particular reviewer, the votes information was not provided for reviewers solely present in the testing

Useful votes	Frequency
0	95,370
1	65,301
2	31,466
3	15,351
4	7,997
5	4,560
6	2,773
7	1,814
8	1,308
9	896
>=10	3,071

Table 2: Frequency table of number of useful votes

set. To summarize, we had three types of reviewers along with their availability of information

1. 43,873 *training reviewers* with complete data on historical statistics of votes, rating, and review count publicly available. These reviewers were present only in the training data.
2. 5,105 *testing reviewers* with data on historical statistics of rating and review count but no information on votes. These reviewers were present only in the testing data. There are two points to note about these reviewers:
   (a) Their historical data on votes is available publicly but was not provided during the competition
   (b) They only formed a subset of testing data. Reviews written by training reviewers could be part of the testing data if they were drafted after Jan 19, 2013
3. 2,318 *private reviewers* with no data for votes, rating, or review count. These reviewers chose to keep their profiles private and hence no information for them is publicly available. They were present both in the training and testing data.

4.3 Businesses

The reviews were for 12,742 businesses belonging to 523 categories. However, neither the businesses nor the categories were mutually exclusive. For example, food joints with multiple branches like KFC were counted as separated businesses in overlapping categories of fast food and restaurants. The categories for different businesses ranged from restaurants to stores to services shops to pet groomers. Figure 1 shows the number of reviews by popular business categories. About 76% of the reviews in the data set were for food, restaurants and related businesses.

For each business, information on its name, address, city, state, latitude, longitude, whether still open or closed, categories, total number of reviews, and average star rating was provided.

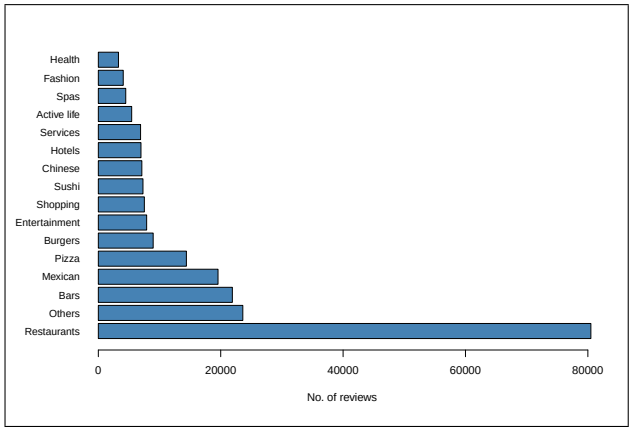

Figure 1: No. of reviews by business categories

4.4 Checkin

Checkin data were at a business id level. They were available for 9,016 businesses and included the cumulative number of checkins during each hourly interval of the day for each day of the week till the data snapshot date. Most of the data were sparse with a high number of missing values for each checkin point.

5 Pre-processing

Data were cleaned and processed to convert raw variables and unstructured text data into meaningful and algorithm readable features. It was done in two stages – feature creation and feature selection. Details for both have been provided below.

5.1 Feature creation

Four feature sets were created from review, reviewer, business, and checkin data for building models to predict the number of useful votes a review is expected to get. The feature sets included structural, lexical, meta, and interaction features.

5.1.1 Structural features

Structural features were created at a review level and contained information on the writing framework behind each review. These included the length of the review, number of sentences, number of lines, number of words, number of punctuation marks, number of numbers, number of capitalized words, and presence of a url. Parts of speech were tagged to calculate the number of adjectives, nouns, and verbs. The star rating given by the reviewer to the business was taken directly from the original data. Lastly, age of the review was calculated as the difference in days between the review draft date and the data snapshot date.

5.1.2 Lexical features

Lexical features included document-term matrices of n-grams $\{n : 1, 2, 3\}$. A document-term matrix is a two dimensional array created from a set of text documents. Each document forms the rows of the array and each unique word from all the documents shapes the columns (Feinerer et al. 2008, Feinerer and Hornik 2013, Manning et al. 2008). The cells in the array are filled with frequency count of the word occurring (defined by the column) in each corresponding document (defined by the row). Using the frequency count of the

words to fill the cells is known as the term frequency (tf) weighing scheme. Another weighing scheme popularly used is term frequency-inverse document frequency (tf-idf). Inverse document frequency of a term is given by

$$idf_t = \log(\frac{N}{df_t}) \quad (1)$$

where t is the term, N is the total number of documents, and df_t is the number of documents that contain the term t. The value in each cell is then equal to $tf \times idf$.

We created document-term matrices with both weighing schemes - term frequency and term frequency-inverse document frequency (tf-idf). N-grams were created after removing common English stop words and stemming using the Porter Stemmer (Manning et al. 2008). A sparsity threshold of 99.9%–99.99%[8] was chosen to constrain the data set to a manageable size. Finally, a total of 3,840 1-grams, 6,116 2-grams, and 4,281 3-grams were created with both weighing schemes respectively.

5.1.3 Meta features

Meta features were created at reviewer and business levels. For reviewers, summary statistics for length of reviews, age of reviews, distinct number of categories reviewed, and distinct number of businesses reviewed were calculated. The total number of reviews and the total number of useful votes were taken directly from the reviewer files. For testing and private reviewers, the total number of useful votes were imputed using the mean value from the training reviewers.

Similar summary statistics were extracted for businesses as well except for the number of useful votes. In addition, binary variables from categories were created to indicate if the business belonged to a particular category or not. Checkin data were also mined, however, due to their high level of sparsity only three meaningful aggregate variables were extracted - mean, max, and sum of all checkins till the data snapshot date.

5.1.4 Interaction features

Interaction features included products/ratios between structural and meta features, and unsupervised clusters using reviewer meta data. For example, the product of length of review and age of review, the product of length of review and the number of distinct business categories reviewed, and a set of ten clusters based on the number of reviews by a reviewer and the average number of useful votes received by the reviewer. The unsupervised clustering was performed using k-means clustering. Most of these features had little intuition behind them and were mainly derived for competitive gain on leaderboard.

5.2 Feature selection

Feature selection largely revolved around the lexical feature set. The idea was to select the best subset of features that – 1) would not demand exceptionally high computation time and power, and 2) give reasonably comparable accuracy as the larger set. We used two methods for feature selection – regularized linear regression and principal component analysis.

[8] Sparsity threshold of 99.9% implies that the terms should be present in atleast 0.1% of all reviews

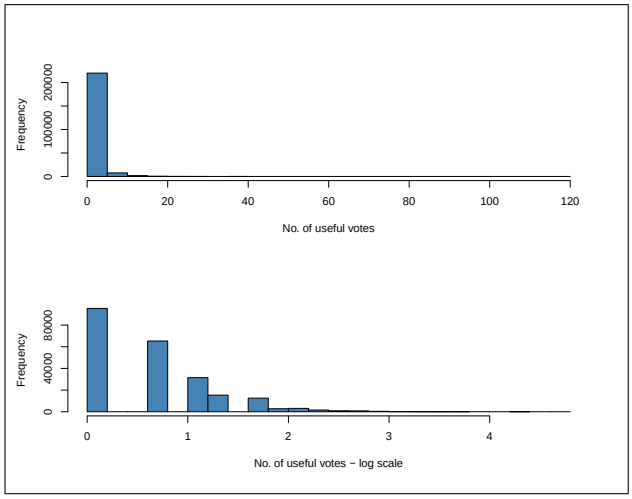

Figure 2: Distribution of the number of useful votes

5.2.1 Regularized linear regression

Regularized linear regression, using glmnet (Friedman et al. 2010), was run for each document-term matrix separately and the best features were selected through a two-level 25 (5x5) fold cross validation procedure. The first level involved cross-validating through the training data and the second level involved cross-validating within each fold from the first level to select the best parameter value (lambda) for glmnet. For the entire analysis the alpha parameter was fixed at 1 as required for a LASSO penalty (Friedman et al. 2010, Tibshirani 1996).

5.2.2 Principal component analysis

Randomized PCA using singular value decomposition (Pedregosa et al. 2011) was performed on each document-term matrix with the number of components fixed at 50.

6 Modeling

The objective of the competition required to build a model to predict the number of useful votes a review is expected to receive. The model's accuracy were to be judged by comparing the predictions to the ground truth values from the testing data using the root mean square log error (rmsle) as given in equation 2

$$rmsle = \sqrt{\frac{\sum_{i=1}^{n}(log(y_i+1) - log(\hat{y}_i+1))^2}{n}} \quad (2)$$

where n is the total number of reviews, y_i is the actual number of useful votes a review received, and $\hat{y}_i$ is the predicted number of useful votes a review is expected to receive. Most regression algorithms operate by minimizing the root mean square error instead of the root mean square log error. Keeping this in mind, we modified our target variable by taking its natural log, i.e., $\tilde{y} = \ln(y+1)$, where y is the number of useful votes and the 1 is added to check for reviews with zero votes. Figure 2 shows the distribution of the original dependent variable and the log transformed dependent variable.

Many previous winning methods on Kaggle[9] used tree based methods like Random Forests and Boosted

[9] www.kaggle.com

Regression Trees. We used both these methods and combined their results to generate the final predictions. The models were trained using 85% of the original training sample, keeping the remaining 15% as a hold-out for parameter selection and model calibration. A brief description of these methods and our selected hyper-parameters is provided below.

6.1 Stochastic Gradient Boosted Regression Trees (GBM)

Gradient Boosting is an algorithm that builds stage-wise additive models in a greedy fashion (Friedman 2001). Stochastic Gradient Boosting allows each additive model to be built on a random sub-space of training observations by randomly sampling with replacement before each iteration (Friedman 2002). The training model within the algorithm is called a base learner. For our purpose, we used the GBM implementation in R (Ridgeway 2013) with regression trees as base learners. We built multiple GBM models using different combinations of structural, meta, and interaction features along with different parameter settings and validated each of them independently on the hold-out sample. The two main hyper-parameters, shrinkage and number of trees, were chosen keeping the accuracy–computation time trade off in mind. The parameter settings for the best model were as follows – shrinkage = 0.04, number of trees = 4000, interaction depth = 7, minimum observations in a node = 70, and percentage of training data to be randomly sampled for building each tree = 50%.

6.2 Extremely Randomized Trees (ERT)

Extremely randomized trees is an algorithm similar to Breiman's Random Forests (Breiman 2001). Random Forests involves creating multiple bagged (bootstrap aggregated) trees each over a random subspace of features. The predictions are then averaged over the entire forest. In Random Forests, each parent node is split into further nodes by choosing the best split among the random subset of features chosen for the particular tree (Breiman 2001, Liaw and Weiner 2002). ERT is different from this in two ways – 1) it builds each tree on the same sub-sample of training observations, 2) it first generates contender split thresholds randomly and then chooses the best split from these thresholds (Geurts et al. 2006). By adding randomness in splits, ERT reduces the variance of the system more than Random Forests do but at the expense of an increased bias (Pedregosa et al. 2011).

We used the scikit-learn implementation in python (Pedregosa et al. 2011) to build Extremely Randomized Trees. We built 150 trees by allowing them to grow fully using the combination of *max_depth = None* and *min_samples_split = 1* as parameters. All four feature sets were used for building the model. Only one document-term matrix was used at a time from the lexical feature sets. A total of six models were built, each time with a different document-term matrix. Out of the six contender feature matrices, bi-grams of term frequencies had the lowest error on the holdout sample.

6.3 Ensemble

Ensembling is a technique that involves combining multiple models to improve the accuracy of the overall system. It is the same concept that is used internally while building a Random Forest (Breiman 2001) or training a Gradient Boosting Machine (Friedman 2001) and can be generalized to combining models trained using different algorithms. Ensembling different models was popularized during the Netflix Prize [10] when the milestone and final winners published their methodology (Koren 2009, Piotte 2009, Toscher and Jahrer 2009). We combined two models–a GBM and an ERT using a simple average with equal weights to create an ensemble of their predictions.

7 Results

7.1 Important factors for useful reviews

Prior research has shown varied results as to which variables are most important for predicting usefulness of reviews. Cao et al. found that semantic characteristics (substance of the review) provide most important pieces of information towards usefulness (Cao et al. 2010) . Using Random Forests and converting helpfulness to a classification problem Ghose et al. concluded that readability, subjectivity, and reviewer characteristics to be of equal predictive power (Ghose and Ipeirotis 2011). Using non-linear SVMs, Liu et al. found reviewer expertise, writing style, and timeliness to be important predictors for review helpfulness (Liu et al. 2008).

Our results show that reviewer characteristics are most influential in predicting the number of useful votes. The single most important variable in our model was the reviewer's cumulative number of useful votes for till date. This was followed by a set of structural features like the length of the review, the age of the review, and the number of lines in a review. We found lexical features to be least predictive among all the four feature sets. Table 3 presents a list of top 12 variables in order of their importance from GBM.

Rank	Variable
1	No. of useful votes of the reviewer
2	Age of review
3	Length of review × Age of review
4	Length of review × No. of distinct categories reviewed
5	No. of lines in review
6	Reviewer cluster (categorical)
7	No. of useful votes of the reviewer × Age of review
8	Average age of all reviews by the reviewer
9	Star rating by reviewer
10	No. of reviews by reviewer
11	Length of review
12	Average age of all reviews by the reviewer

Table 3: Important variables from GBM

[10] http://www.netflixprize.com/

Model	Data sample		
	Holdout	Public LB	Private LB
GBM	0.4505	0.4506	0.4535
ERT	0.4577	0.4508	0.4514
Ensemble	**0.4469**	**0.4446**	**0.4462**
Rank 1 model	-	0.4390	**0.4404**
Rank 2 model	-	**0.4377**	0.4408

Table 4: RMSLE of models on different data samples

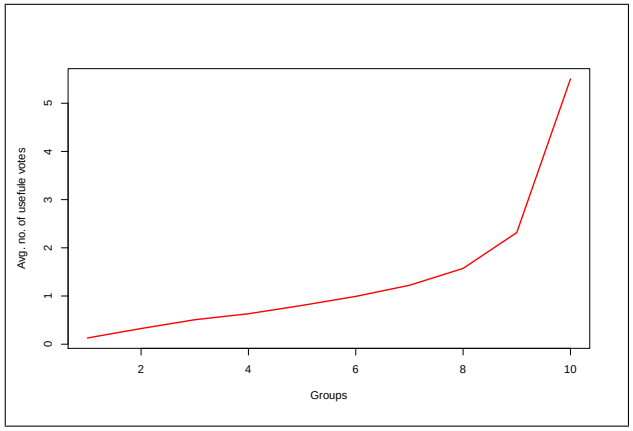

Figure 3: Lift separation based on predicted values

7.2 Model performance

Our ensemble model was the third best entry on the private leaderboard as judged by Equation 2. Table 4 presents a comparison of the relative scores of our models on the hold out sample, public leaderboard (LB), and the private leaderboard. The scores of the best two models on the leaderboards are also shown.

Figure 3 shows the lift separation provided the predictions from the model. The graph is generated by arranging the ground truth values based on the predictions, bucketing them in to 10 groups using the prediction order, and taking the mean of the ground truth values in each group. Based on the graph, the model gives a Lift Ratio [11] of 3.91 in the highest group and a normalized gini of 0.7687 on the holdout sample.

8 Conclusion and future work

Our paper builds on the previous work by adding value in three different ways – 1) we consider reviews for different and diverse business categories ranging from restaurants to pet shops which makes our results easily generalizable across different product environments; 2) we analyze all reviews irrespective of the whether they have received any useful vote till date or not; 3) we build an ensemble of ensembles by training two different machine learning algorithms and combining their results for superior accuracy.

In addition to the above three points, our results are based on a sample of more than 250,000 reviews by 50,000 reviewers across 12,000 businesses. According to our knowledge, this is considerably larger than any other sample analyzed in previous research efforts.

We base our results on two robust and powerful machine learning algorithms trained on a large and varied sample using structural, lexical, reviewer, and business characteristics. Our results suggest that reviewer information is most important in predicting usefulness of reviews. These results are similar to those of Forman et al. (Forman et al. 2008). We also find that lexical features derived from the review text are least influential in predicting the number of useful votes a review will receive. These results are contradictory to that obtained by Cao et al. (Cao et al. 2010). However, Cao et al. do not consider reviewer information in their analysis making it difficult to directly compare our results with theirs. Lastly, our model performs well on unseen new data. Competing against 351 other models, our model ranked 3^{rd} on the final leaderboard standings.

Though our work augments other research undertakings in different ways, it is constrained by many limitations. First, we do not explore the structure of the review fully. For example, Ghose et al. use multiple readability indexes to estimate the effort required by the viewer in reading the review (Ghose and Ipeirotis 2011). They also include subjectivity analysis in their model and their results suggest that reviewer characteristics, readability, and subjectivity are equally important in predicting helpfulness of reviews. Including these additional dimensions in our work will help provide a better understanding of the reviewer-usefulness complexity. Second, the hyperparameters of our training algorithms were based on previous experience, reference manuals, and results of past Kaggle competitions hinting that they might be sub-optimal for this particular problem. A better approach would be to do a grid search across the length of the training data using cross-validation to determine the optimal values. Third, the objective of our study was more predictive rather than exploratory in nature. To this goal, we used algorithms that are proven to be more accurate but partially black-box, eclipsing the true underlying relationship between the target and the predictor variables.

Our paper provides valuable extensions across multiple dimensions of opinion mining. However, given the limitations mentioned above, this paper can be extended in depth and breadth in many different ways. For instance, our results hold good for reviews that have reviewer characteristics available. But there are numerous reviews written by anonymous reviewers. Directly implementing our model on these might not yield the best results. A more suited methodology would be to deal with these reviews as a cold-start problem and analyze them separately. Another challenging problem to solve would be to build a system for judging review usefulness that develops in a self-correcting manner through feedback loops by factoring in new reviews at regular intervals (Liu et al. 2008). Such a system would prove useful for review websites like Yelp which are devoted to service oriented businesses.

[11] Lift Ratio = $\frac{\mathbb{E}(Target|Segment)}{\mathbb{E}(Target|Population)}$

References

Koren, Y., (2009), The BellKor Solution to the Netflix Grand Prize, Available from: http://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.162.2118

Breiman, L. (2001), 'Random forests', *Machine Learning*, Vol. 45.1, pp. 5–32.

Burges C.J.C.,(1998), 'A tutorial on support vector machines for pattern recognition', *Data Mining and Knowledge Discovery*, Vol. 2, Issue 2, pp. 121–167.

Cao, Q., Duan, W., & Gan, Q., (2011), 'Exploring Determinants of Voting for the "Helpfulness" of Online User Reviews: A Text Mining Approach', *Decision Support Systems*, Vol. 50, Issue 2, pp. 511–521.

Duan, W., Gu, B., & Whinston, A.B., (2008), 'The dynamics of online word-of-mouth and product sales an empirical investigation of the movie industry', *Journal of Retailing*, Vol. 84, Issue 2, pp. 233–242.

Feinerer, I., Hornik, K., & Meyer, D., (2008), 'Text mining infrastructure in R', *Journal of Statistical Software*, Vol. 25, Issue. 5, pp. 1–54

Feinerer, I., & Hornik, K., (2013), 'tm: Text mining package. R package version 0.5-8.3.', Available from: http://CRAN.R-project.org/package=tm

Forman C., Ghose, A., & Wiesenfield, B., (2008), 'Examining the relationship between reviews and sales: the role of reviewer identity disclosure in electronic markets', *Information Systems Research*, Vol. 19, Issue 3, pp. 291–313.

Friedman, J. (2001), 'Stochastic gradient boosting', *Computational Statistics & Data Analysis*, Vol. 38, pp. 367–378.

Friedman, J. (2002), 'Greedy function approximation: A gradient boosting machine', *The Annals of Statistics*, Vol. 29, pp. 1189–1232.

Friedman, J., Trevor, H., & Tibshirani, R., (2010), 'Regularization paths for generalized linear models via coordinate descent', *Journal of statistical software*, Vol. 33, Issue 1, pp. 1–22.

Geurts, P., Ernst, D., & Wehenkel, L., (2006), 'Extremely randomized trees', *Machine Learning*, Vol. 63, Issue 1, pp. 3–42.

Ghose, A., Ipeirotis, P.G. (1993), 'Estimating the helpfulness and economic impact of product reviews: Mining text and reviewer characteristics', *'IEEE Transactions on Knowledge and Data Engineering'*, Vol. 23, Issue 10, pp. 1498–1512.

Liaw, A. & Weiner, M. (2002), 'Classification and regression by randomForest', *R News*, Vol. 2/3, pp. 18–22.

Liu, J., Cao, Y., Lin, C.Y., Huang, Y., & Zhou, M., (2007), 'Low quality product review detection in opinion summarization', *Joint conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning*, pp. 334–342.

Liu, Y., Huang, X., An, A., & Yu, X., (2008), 'HelpMeter: A nonlinear model for predicting the helpfulness of online reviews', *IEEE/WIC/ACM International Conference on Web Intelligence and Intelligent Agent Technology*, Vol. 1, pp. 793–796.

Kim, S.M., Pantel, P., Chklovski, T., & Pennacchiotti, M., (2006), 'Automatically assessing review helpfulness', *Conference on Empirical Methods in Natural Language Processing*, pp. 423–430.

Manning, C., Raghavan, P., & Schutze, H., (2008), 'Introduction to Information Retrieval', *Cambridge University Press*

Mudambi, S., & Schuff, D., (2010), 'What makes a helpful online review? A study of customer reviews on amazon.com', *MIS Quarterly*, Vol 34, Issue 1, pp. 185–200

Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., Blondel, M., Prettenhofer, P., Weiss, R., Dubourg, V., Vanderplas, J., Passos, A., Cournapeau, D., Brucher, M., Perrot, M., & Duchesnay, E., (2011), 'Scikit-learn: Machine learning in python', *Journal of Machine Learning Research*, Vol. 12, pp. 2825–2830.

Piotte, M., & Chabbert, M., (2009), The Pragmatic Theory Solution to the Netflix Grand Prize, Available from: http://www.netflixprize.com/assets/GrandPrize2009_BPC_PragmaticTheory.pdf

Ridgeway, G. (2013), 'gbm: Generalized Boosted Regression Models', Url. http://CRAN.R-project.org/package=gbm

Spool, J., (2009), The magic behind Amazon's 2.7 billion dollar question, Available from: http://www.uie.com/articles/magicbehindamazon/

Tibshirani, R., (1996), 'Regression shrinkage and selection via the Lasso', *Journal of the Royal Statistical Society*, Vol. 58, Issue. 1, pp. 267–288.

Toscher, A., & Jahrer, M., (2009), The BigChaos Solution to the Netflix Grand Prize,

Turney, P., (2002), 'Thumbs up or thumbs down?: semantic orientation applied to unsupervised classification of reviews', *Proceedings of the 40^{th} Annual Meeting of the Association for Computational Linguistics*, pp. 417–424

Wooldridge, (2002), 'Econometric analysis of cross-sectional and panel data', *MIT Press*, Cambridge, MA.