Novel approach to predict usefulness of online user reviews

Speech and Natural Language processing term project
Department of Computer Science and Engineering
Indian Institute of Technology, Kharagpur

Khadpe Pranav Ajay (15EE30025)
Joshua Peter Ebenezer (15EC10023)
Atul (15EC10067)

Project Mentor:
Soumya Sarkar

Course co-ordinator: Professor Niloy Ganguly

November 30, 2017

Abstract

In this project report, an approach to predict the usefulness of online reviews of restaurants is presented. The data is collected from round 10 of Yelp's dataset challenge. yelp.com is a website where users can rate and give reviews for commercial establishments. Readers of reviews can vote reviews as useful, and this is used as a measure of usefulness. The goal of our method is to be able to predict how many useful votes a given review would receive, immediately after the review is written and uploaded. This would enable Yelp to float to the top reviews that are predicted to be useful, which might otherwise be lost in the huge number of reviews, and may never be seen though they are useful. We used LIWC features and a gaussian naive bayes for binary classification, and softmax for multi-class classification. The model achieved an accuracy of 72% on binary classification as 'useful' and 'not useful' and performed very well on multi class classification to predict if a review might be 'not useful', 'somewhat useful', 'useful', or 'very useful'.

Contents

1 Introduction

The website yelp.com allows users to write reviews for restaurants and other commercial establishments. This helps other customers to make an informed decision on whether to visit an establishment or not. However, due to the popularity of the website, a massive number of reviews are available for almost every eatery. As a result, many reviews are actually never seen, as they get lost in the others. Yelp also allows users to vote a review as 'useful', 'funny' and 'cool'. Over a period of time, these votes accumulate and stabilize, as new trends emerge, restaurants change and other reviews are read more. Yelp! released a dataset of approximately 5 million reviews [1] in round 10 of it's dataset challenge. The challenge we took up was to identify reviews that have a good chance of accumulating a good number of 'useful' votes over time, so that such reviews could be floated up so that other users can read them, while the ones that are not useful are kept lower down in the web-page so that the 'useful' reviews are read first.

We decided to look for language features that would identify a review as useful, and train a model to use the features to classify the review. Previous work in this regard has been based on metadata and text features, but this is the first time LIWC features are being used for this problem. Zhang et al. [2] used VADER features and metadata to get an accuracy of 68% on binary classification for 0 and non-zero useful votes. Liu et al. [3] used Glove vectors to develop an RNN based model that took in the words of the review and made predictions based on them. Shen et al. [4] used BOW and metadata as well, to train a batch mode weighted localized regression model.

This report presents a novel approach that uses only text features to build the classifier. This actually enables our approach to be used not only for yelp, but even for other sites such as Amazon and Google Play Store to develop better review display systems so that customers can read the most helpful reviews to make informed decisions.

2 The data

Yelp holds a public challenge, wherein it releases a raw dataset of reviews, users, and businesses to the public, and asks the community to submit their ideas on how the datset can be used to improve the user experience at yelp, using any technique or method. Round 10 of the dataset challenge began on September 10th, 2017. The dataset of reviews is provided in a json file, and has 5 million reviews, which are much more than required to develop a decent classifier, besides the fact that 5 million reviews is too large a set to be processed with available computational resources. Besides the reviews themselves, descriptions of each user and business are available as well in separate json files. Each review contains fields, the business id of the business that the review is written for, the user id of the writer of the review, the text of the review, the number of stars the reviewer has given the establishment, the date the review was written, and the number of 'useful', 'funny' and 'cool' votes the review has received.

2.0.1 Sparsity of reviews voted as useful

The frequency distribution of the number of useful votes received by the reviews is shown in Figure 2.1 for about 2 million of the reviews.

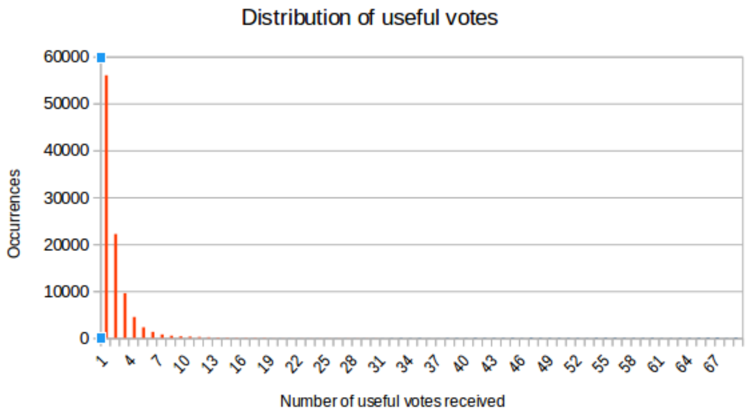

Figure 2.1: Frequency distribution of useful votes

The distribution shows quite clearly that the vast majority of reviews have received 0 useful votes. This may because they were genuinely not useful, or because they were never seen. The second possibility is more likely, considering the vast number of reviews available on yelp. This poses a challenge in terms of data sparsity as well as data representation. The reviews that are voted as useful are few in number and the distribution quickly tapers off after 4 useful votes, which would cause a bias in the model if sampled directly. Besides this, the reviews having 0 useful votes cannot be immediately classified as ’not useful’ unless it is established that they were seen by the other users. Hence features extracted from this set directly cannot be used to represent the ’not useful’ class. Quantitatively, 45.2% reviews have 0 upvotes and 98.6% have less than 20.

2.0.2 Distribution of number of words and characters

The graph of the distribution of the number of words is shown in Figure 2.2

Most reviews had a length of about 30 words. Almost all reviews were less than 500 words.

3 Method

In this section, the methodology we used to pre-process data as well as train the model is discussed.

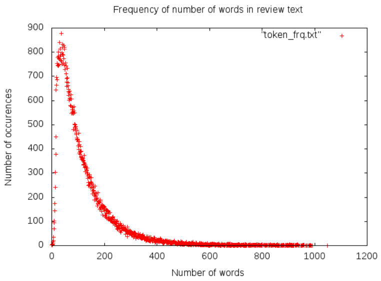

Figure 2.2: Distribution of number of words

3.1 Pre-processing

3.1.1 Removing reviews that were surely never seen

In order to be able to build an unbiased classifier and obtain features that actually represent their class, we only kept reviews that had atleast 1 useful, funny or cool vote.

This would ensure that every single review had been seen by atleast 1 person. Yet since useful, funny and cool are independent metrics, we still had the majority of the reviews having 0 useful votes.

3.1.2 Removing old and very recent reviews

If a review is written very recently (with respect to the date of release of the database), chances are high that it was seen by very few people, and may have got very few 'useful' votes because of this. Figure 3.1 the distribution of epoch times (the time in seconds since January 1st, 1970).

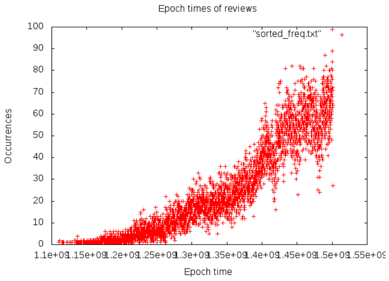

Figure 3.1: Distribution of epoch times

Reviews that are very old may not accurately reflect recent trends, because in a rapidly changing world standards keep changing very often.

To completely remove the effect of time, we removed reviews that were written after January 1st, 2017, and reviews that were written before January 1st, 2015.

3.1.3 Removing non-English reviews

Some of the reviews were written in European languages (to review European restaurants). Since this is a natural language approach focusing on language level features (and not on metadata), we removed all non-English reviews by using the langid classifier.

3.1.4 Removing very short and very long reviews

We did not expect very short reviews (less than 10 words) to give us much information. We also expected that the LIWC features would work best when the texts are all of moderate length, and hence we removed reviews larger than 500 words.

3.1.5 Clustering

We used k means clustering to obtain 4 bins. The first one had 0 votes, the second had 1 or 2 votes, the third had 3 or 4 votes, and the fourth bin had all reviews with more than 4 votes.

3.1.6 Under-sampling to remove biases

The class with more than 4 useful votes was the least populated. In order to counter this bias, we first randomly sampled 100,000 data points and then performed all the above steps to end up with around 50,000 reviews. We saw that class 4 (most useful) had the least number of instances (1100). Hence, we sampled only 1100 features from each category, and discarded the rest. Hence, our dataset had 4400 reviews.

3.2 LIWC features

LIWC stands for Linguistic Inquiry and Word count [5]. It is a proprietary software that provides 72 features for any given text. The features provided are standard linguistic dimensions (verbs, nouns etc), psychological processes that cover social, cognitive, affective, perceptual, biological processes and relativity. Personal concerns and spoken categories are included too. The value of each feature category is the percentage of the text that belongs to the feature label (for eg:, if 20% of the text can be classified as having a postive sentiment, then the value at the postive sentiment feature field would be 1. Categories may overlap but every category has it's significance. Using the extreme tree algorithm, we selected the top 25 features and used them for training.

3.3 Training models used

We used the following training models (shown in table 3.1)

Table 3.1: Classifiers used for multi-class

SVM one-vs-one
SVM one-vs-all
Logistic regression
Random forest
K nearest neighbours

4 Experimental results

4.1 Preliminary results

We first attempted to generate tf-idf vectors for each review, and then use these to predict the exact number of votes that each review would receive (without any preprocessing of the data). We used the softmax classifier and treated it as a multi-class classification with the number of votes being the bins. We got an accuracy of 54% but this was because almost all reviews were being assigned to the 0 useful votes class. This highlighted the need for undersampling and proper handling of biases and sparsity.

4.2 Multi-class classifier

We then used k-means clustering to divide the reviews into 4 bins ('not useful', 'somewhat useful', 'useful', and 'very useful'). This was done using k-means clustering. The 4 bins generated had the number of useful votes being: 0 useful votes for the 1st bin, 1 and 2 useful votes for the 2nd bin, 3 and 4 useful votes for the third bin, and all reviews having more than 4 votes were placed in the 4th bins by the k means clustering. The results for the multi-class classifier were not very good on first glance, as we obtained an accuracy of 40% with the Random forest model. However, when we looked at the binwise distribution, we found that that most misclassified instances differed from the actual bin only by 1. The results were as follows:

Table 4.1: Breakdown of results

Correctly predicted:480	Off by 1: 440
Off by 2:190	Off by 3: 74

Considering the fact that, in the real world, a misclassification that is off by one 1 bin corresponds to just one or two votes. One or two votes is certainly not a significant error. User behaviour online is not predictable to the exact number of votes, but having a model that predicts it roughly is good enough, even if the numerical accuracy is not good since there is no metric to represent the fact the misclassification by 1 bin is not very bad in this problem statement. If the model predicts that the review will get 3 votes, and the review happens to get 5 votes, it is not a big deal.

4.3 Binary classifier

We also thought of another approach: to separate the data into just two bins- 'useful' and 'not useful'. We tried out an SVM, logistic regression, and naive bayes classifier. This gave an average best classification accuracy of 72% with the gaussian naive bayes classifier, with a precision of 79% and a recall of 62% for five fold cross validation. The gives a better numerical accuracy with the meaning of the results remaining the same as the multi-class. The confidence of the classifier can be used as a measure of the usefulness as well. Our results are better than Zhang et. al's method, that achieved an accuracy of 68%, and almost as good as the results of Liu et. al, who achieved 75% accuracy with an RNN.

5 Conclusion

We have developed two approaches to classify the reviews. The numerical accuracy is not very promising, but we must understand that the classification problem in our case is distinctly different from other classification problems, where we might have classes such as 'car', 'bird', 'person' etc., where each class is clearly distinguishable. Here we have classes separated by the judgment of a few people in clicking the 'useful' up-vote. The bins that we have used are thus actually giving very good results, since classifying a review as 'somewhat useful' is acceptable even if the review happens to get slightly fewer votes than we predicted, and thus is actually in the 'useful' class. This can thus be used by yelp or other online review sites as well, to provide a better user experience by ranking reviews right after they are written.

References

[1] https://www.yelp.com/dataset/challenge

[2] D.Liu, G.Singh, "Using Recurrent Neural Network to Predict The Usefulness of Yelp Reviews", (https://web.stanford.edu/class/cs221/2017/restricted/p-final/dzliu/final.pdf)

[3] X.Liu, M.Schoemaker, N.Zhang, "Predicting Usefulness of Yelp Reviews" (https://pdfs.semanticscholar.org/3293/de14486e2f16f0eecb13976c2724172290fd.pdf)

[4] R.Shen et al., "Predicting usefulness of Yelp reviews with localized linear regression models", Software Engineering and Service Science (ICSESS), 2016 7th IEEE International Conference on

[5] Yla R. Tausczik and James W. Pennebaker, The Psychological Meaning of Words:LIWC and Computerized Text Analysis Methods, Journal of Language and Social Psychology