Predicting the "Usefulness" of Customer Reviews

Anonymous
Affiliation
Address
email

Abstract

This paper compares the performance of various machine learning regression techniques while predicting the usefulness of customer reviews. The dataset is obtained from Yelp, an Internet based business review service, as part of it's recently announced dataset challenge on Kaggle.com. The paper suggests that the usefulness of a review depends on the contents of the review and the language it uses. During the experiment, it is discovered that feature selection and preparing and massaging the data for this task are very tedious processes. For this reason, the techniques used for feature selection and data massaging have also been described in detail in this paper. It is found that for the given dataset, a non-linear model best describes the model, and support vector machines using radial basis functions provide a good way to regress the number of useful votes for a review. It is also recommended to experiment with other non-linear regression techniques such as regression with Gaussian processes and neural networks.

1 Introduction

Online communities rely heavily on trust within the community. Building trust has always been a difficult task, and doing this in an online community is especially difficult. A commonly used technique that is often used to support such a community is to enable community powered ratings and reviews. This is commonly seen in Internet based crowd-sourced reviewing services such as Yelp and collaborative consumption based services such as Airbnb and GetAround.

However, the growing popularity of a community often results in a large amount of variance in the quality of customer reviews. This can have an adverse effect on the experience of the user, and it might impact the trust within the community. Therefore it becomes essential to control which reviews are displayed to users, and what order they are ranked in. Traditionally, community managers have done this manually. However, as the service scales, it becomes essential to find an automated way to rank these reviews. For this reason, Yelp gathers feedback on each user review from other reviewers. Yelp tries to measure each review with three community-powered metrics: useful, funny and cool. These metrics are then used to decide how the reviews should be ranked.

However, the problem is that gathering votes for how useful, funny or cool a review can be a time consuming process. It can often take weeks if not months. And the usefulness of a review also deteriorates with age, thus rendering reviews that would otherwise have been valuable useless. For this reason, it is very important to have an automated way to predict how useful a review will be before the community finishes voting on the review. This will ensure that high quality reviews are ranked high regardless of the number of votes awarded by the community.

To be able to rank these reviews according to usefulness, it is important to understand what makes a review useful. This is where the concept of feature selection comes into play. Feature selection is a machine learning technique that helps create a set of relevant features for use in model construction. Since the reviews on Yelp are text-based, intuition would suggest that most of the relevant features would lie in the text of the reviews themselves. While one can also think of other non-textual features that would be relevant for such a task (such as number of past reviews by the same user), this paper exclusively focuses on the textual contents of the reviews.

Yelp has recently released a dataset of reviews from businesses in Phoenix, Arizona, and has hosted a competition on the data science competition website Kaggle.com. The goal of the competition is to predict the number of "useful" votes a review will get over its lifetime. The dataset consists of 220,000 training reviews. This paper aims to find the best technique that should be used to predict the number of "useful" votes for a review.

2 Related Work

As mentioned above, in this paper we are trying to predict the number of votes a review will get. This means that we are trying to predict how important the review is, and how high up it should be ranked when retrieved amongst other reviews for a specific business. Since the task at hand is to train a model for a ranking task, the problem can be classified as a learning to rank problem.

Since review ranking is a problem faced by several online services, it is very likely that many such services use their own machine learning techniques to rank their reviews.

Several researchers have previously explored the problem of learning to rank. A Microsoft paper from 2005 introduced RankNet- an implementation of gradient descent methods applied to learning ranking functions using a neural network to model the underlying function [1].

Learning to rank can be employed in a number of applications such as document search, definition search, information retrieval, key phrase extraction, collaborative filtering, document summarization and machine translation [2].

While we have found some prior work related to ranking methods using neural networks, we have not found any prior work where an item is ranked based on the regression of its textual contents using natural language processing. We believe that this might be an effective method of predicting the rank of a review, given labeled training data where reviews have already a qualitative indicator.

3 Background

The Yelp dataset consists of four different kinds of data- data about businesses such as their names, neighborhoods, addresses and categories, data about users such as their name and the number of times they have reviewed or rated businesses, review data such as the text of the review along with the useful, cool and funny votes assigned, and check-in data about how many users checked into a business in a specific period of time. The data that is most interesting for our task is the review data itself. The review data is what will define the feature set for our task. The figure below describes the contents of the review data.

The review data provided by Yelp is all text based. To use this data effectively, we must use some natural language processing techniques. Natural language processing is a field of artificial intelligence concerned with the interactions between computers and natural human languages. Several open source tools have been built to help process natural language text. One such tool called nltk (Natural Language Toolkit) was used for processing and massaging the data.

4 Feature Selection Method

While creating a feature set, we decided to use only the words within the textual contents of the review as features. We did not choose contextual information or meta-data such as the type of business, geographic location of the business, and reviewer information. This is because we do not think that the contents of reviews are geographically dependent, and we believe that the quality of the review does not change with type of business. While it can be argued that restaurant reviews would be viewed by more users than say jewelry store reviews, the usefulness of reviews does not vary with the type of business. Since each review is to be ranked before the number of votes for the review starts to plateau, the test reviews are assumed to be fresh. For this reason, review freshness is not used as a feature.

Yelp's reviews are user generated. Each user can write reviews from Yelp's mobile and web apps after visiting a business. Yelp employees do not moderate reviews before they are posted online. For this reason, there is lots of noise in the data. Words are often misspelt, and sentences are often badly constructed. Several reviews don't make grammatical sense. Variance in these reviews is also high- some reviews comprise of multiple paragraphs, whereas some are only a few words long. Some reviews describe the quality of services or products offered, whereas some only talk about prices or wait times.

Intuition suggests that reviews with certain descriptive words or phrases will be more useful than others. Words such as "awesome", "clean" and "fresh" are clear indicators that the review is positive in nature, and phrases such as "try the Calamari" are high in information gain about what is good (or bad) at a business.

We started by trying to create a feature set of each unique word in all the reviews. This resulted in a feature set of over 120,000 words. We discovered that This feature set, while exhaustive was too large. Fitting the training data and predicting results with such a large feature set would take hours, if not days. For this reason, it became essential that we filtered according to parts of speech. To do this, we used nltk.

Adjectives, adverbs and nouns are the most descriptive and provide the highest amount of information gain for our task. The table below lists the parts of speech that were used for this project. Other parts of speech such as conjunctions, prepositions, verbs and modals were not included in this list and were filtered out so that they would not be included in the feature set.

Table 1: Parts of speech used to create feature set

FW	Foreign Word	Tandoori
JJ	Adjective	Big
JJR	Adjective, comparative	Bigger
JJS	Adjective, superlative	Biggest
NN	Noun, singular or mass	Door
NNP	Noun, plural	Doors
NNS	Proper noun, singular	John
NNPS	Proper noun, plural	Vikings
RB	Adverb	Good
RBR	Adverb, comparative	Better
RBS	Adverb, superlative	Best

Another important part of feature selection was the use of punctuation and emoticons in user generated content. Since reviews use informal English, it is not uncommon to see users' use of punctuation and emoticons to express tone in a review. For example, a smiley face would indicate that the user was pleased with the service/product, where as an exclamation would indicate an extremely positive or negative experience. For this reason, we included certain emoticons and punctuations were part of the feature list. A table in the appendix details the list of emoticons and punctuations that were included.

We encountered another problem while preparing a feature set that was filtered by parts of speech. The process of creating the feature set involved iterating through each of the 220,000 reviews, tokenizing them and separating them according to the various parts of speech, filtering by part of speech, and checking whether the words of the review are already in the feature list. This was a tedious process, and when repeated over 220,000 times, it became difficult even for powerful machines. For this reason, we decided to cut down the number of reviews that we would create the feature set out of. We cut them down from 220,000 reviews to just 5,000 reviews, to create a feature set of 11,000 words. Figure 1 below shows a snippet of code that uses parts of speech tagging and emoticon/punctuation based filtering to create a feature set.

```
for review in reviews:
    review_index += 1
    if (review_index >= feature_set_size):
        break
    current_review = review.replace(".", " ")
    for each_pair in nltk.pos_tag(nltk.word_tokenize(current_review)):
        if each_pair[1] in ['FW', 'JJ', 'JJR', 'JJS', 'NN', 'NNS', 'NNP', 'NNPS', 'POS', 'RB', 'RBR', 'RBS',] \
            or each_pair[0] in [':)', ':D', ':P', '=)', 'XD', '=D', ':o', '=]', 'D:', ';D', '!']:
            if not apriori.has_key(each_pair[0]):
                apriori[each_pair[0]] = index
                index += 1
```

Figure 1: Code snippet to demonstrate feature selection using parts of speech tagging

5 Experiments

We conducted several experiments to determine the best model for this dataset. We experimented with linear as well as non-linear regression.

We used the scikit-learn module to experiment with each method. This greatly reduced implementation and prototyping time, as the algorithms we used were already implemented. Most of the work now lay in massaging the data and cross-validating the parameters.

We used the metric of mean squared errors to measure the success of the model. Below is a summary of the results of each technique each we experimented with.

5.1 Ordinary Least Squares

We started with a simple linear regression model using ordinary least squares. While this method was easy to implement, it also resulted in the highest mean squared error. This technique was discarded early on due to the poor initial results, and we moved from linear to non-linear regression after this.

5.2 Support Vector Machines

Next we decided to experiment with non-linear regression using Support Vector Machines, as they are known to be able to efficiently perform non-linear regression tasks.

While it is not possible to plot the result of support vector regression with a feature set of over 11,000 features, Figure 2 below shows an example of what the results of using support vector regression can look like with data from a cosine function.

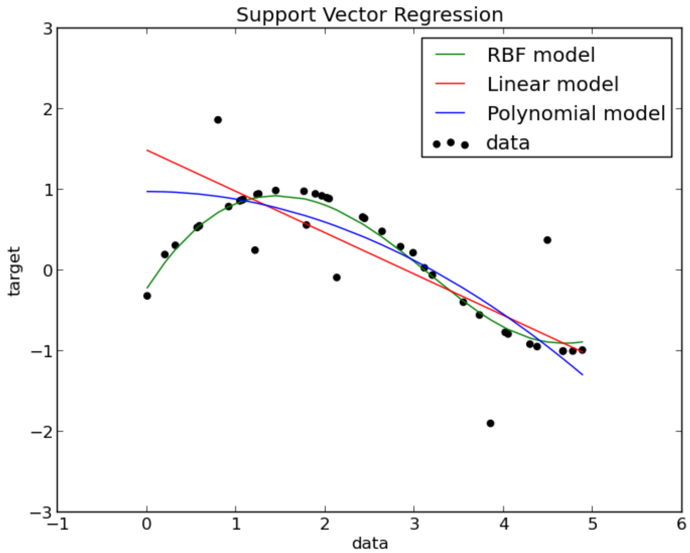

Figure 2: Example of using regression with Support Vector Machines

5.2.1 SVR with polynomial kernels

We started experimenting with Support Vector Regression by using polynomial kernels of degree two and higher. Regressing with these models gave much better results than regressing with a linear model. This gave us an early indicator that linear models in fact are unsuitable for this data, and that further experimentation with non-linear models is a better approach.

We cross-validated with different polynomials as well as other parameters such as the penalty of the error term. Figure 3 below shows the results of this experimentation. It can be seen that while the results do not vary much, on average, polynomials with degree three (blue) and two (red) performed slightly better than polynomials with higher degrees (green). As mentioned previously, mean squared error has been chosen as the metric for measuring the success of the regression.

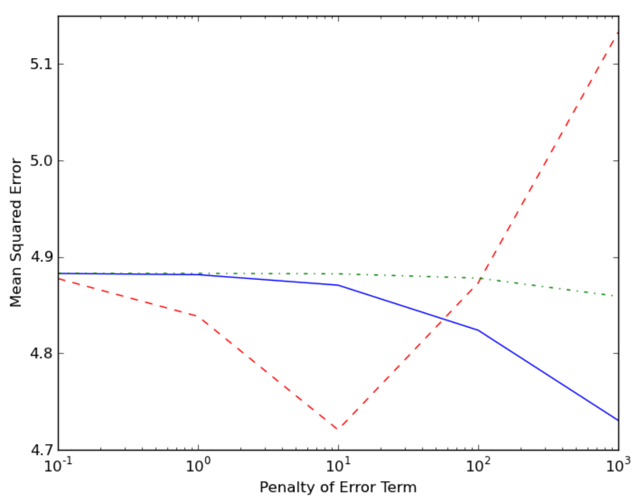

Figure 3: Variation in mean square error with penalty of error term for polynomials with different degrees. Here, the red line is for second-degree polynomials, the blue line is for third degree polynomials, and the green line is for fourth degree polynomials.

5.2.2 SVR with radial basis function kernels

After experimenting with polynomial kernels, we decided to try using support vector machines with radial basis function kernels next.

We cross-validated with different values for penalty of error terms and kernel coefficients. As it can be seen in Table 2 below, the results show that using radial basis functions as kernels for support vector regression provides us with much better results than polynomial kernels. The lowest mean squared error we found through cross validation was 4.197 for penalty of error term 10, and kernel coefficient 0.01.

Table 2: Results of Support Vector Regression using radial basis functions

Penalty of Error term	Kernel coefficient	Mean squared error
1000	0.1	4.573830217
100	0.1	4.495839661
10	0.1	4.47455495
1	0.1	4.750245158
0.1	0.1	4.869590974
1000	0.01	4.536311987
100	0.01	4.364271694
10	**0.01**	**4.196941828**
1	0.01	4.431219925
0.1	0.01	4.574464125
1000	0	4.998818106
100	0	4.418512673
10	0	4.423488496
1	0	4.507002748
0.1	0	4.68913493

6 Conclusions and Recommendations

In this paper, we explored linear and non-linear regression techniques to predict the usefulness of user generated reviews. While we have determined that for this dataset non-linear regression with Support Vector Machines (SVMs) with radial basis functions as kernels are better than linear regression with original least squares, and SVMs with polynomial kernels, it is essential that other techniques are also investigated. Given the short time frame of the project, we were unable to experiment with other promising techniques such as regression with Gaussian processes and neural nets. As suggested by Burges [1], RankNet could also be used. In fact, deep nets could be used to learn a ranking function without the need for having user feedback for training the model.

Through cross-validation we found the best hyper parameters to tune the model. While this is a widely used and reliable technique, a better way to do tune the model would have been to use Bayesian optimization techniques.

On submitting our predictions of the test set to Kaggle.com, we found that our regression technique with support vector machines pushed us into the top 50 ranks on the leaderboard.

With further cross validation and experimentation, we are confident that this technique will push us higher up onto the leaderboard.

While we have only used a set of words as features, a more exhaustive feature set might provide better results. Instead of individual words, bigrams and collocations would be much more powerful as indicators of useful reviews. Examples include phrases such as “seasoned perfectly”, “money’s worth” and “*no complaints*”, which would be much more powerful indicators than the individual words that they consist of. Other meta-data and contextual information such as the reviewing experience of the reviewer are also likely to serve as useful features.

References

[1] Burges, C., Shaked, T., Renshaw, E., Lazier, A., Deeds, M., Hamilton, N., et al. (2005) Learning to Rank using Gradient Descent. In *Synthesis Lectures on Human Language Technologies*.

[2] Li, H. (2011) A Short Introduction to Learning to Rank. In *IEICE TRANS INF. & SYST., E94-D(10),*1854-2862.