CSE 158 Assignment2

Bruce Zhou, Qiyun Li, Yifei Wang
Nov 29, 2022

Abstract

This article mainly focuses on the brief description, analysis, and findings on a dataset related to the RateBeer's multi-aspect reviews. After performing predictive task, training Jaccard similarity and bag of words models, and comparing them with baseline model. We explore contemporary documents that also imports this dataset and try to draw analogies between their conclusions and ours.

1 Dataset

In this assignment, we are going to use the RateBeer's multi-aspect reviews. It contains reviews from Apr 2000 to Nov 2011. Each review contains the following attributes:

Attribute	**Description**
beer/name	Name of the reviewed beer.
beer/beerID	Unique ID of the reviewed beer.
beer/brewerID	Unique ID of the brewer of the reviewed beer.
beer/ABV	Alcohol by volume of the reviewed beer.
beer/style	Style/category of the reviewed beer.
review/appearance	The rating to the reviewed beer's appearance.
review/aroma	The rating to the reviewed beer's aroma/smell.
review/palate	The rating to the reviewed beer's palate.
review/taste	The rating to the reviewed beer's taste.
review/overall	The overall rating that adds each of the ratings above together.
review/time	The time stamp when the review was made.
review/profileName	The name of the user reviewing the beer.
review/text	The review texts to the reviewed beer by the user.

The database has in total 2924164 entries, which is far more than enough. Therefore, we will only take the first 100000 entries for our purpose.We will also shuffle it and split it to train/test sets later.

After exploratory analysis, we found out that there were 7551 users, 4349 beers, and 332 brewers in the first 100000 entries. Therefore,

each user made 13 reviews on average, each beer was reviewed by 23 users on average, and each brewer was reviewed by 301 users on average. We thus assumed that this amount of data was enough for us to build a decent model.

One interesting fact we found was that the most "loyal" user made 549 reviews in total. Their username was "fonefan". We also went to the RateBeer's official website, and found out this person had already rated 71304 reviews! Moreover, the most popular beer in the first 100000 reviews was Bières de Chimay which had received 3056 reviews. We could not recognize this beer though.

2 Predictive Task

In this assignment, we are going to predict the certain user/beer pairs' ratings as accurately as possible. We will use the 80000th - 90000th entries as our validation set for tuning the model, and the last 10000 entries as our test set for final evaluation. We plan to use MSE as our loss function as defined below:

$$MSE = \frac{1}{n}\sum_{i=1}^{n}(R_p - R_t)^2$$

n – Number of all reviews evaluated

R_p – Rating predicted by our model

R_t – True rating in the dataset

To be clear, the rating here we use will be the rating out of 20 instead of the fraction value that will make the MSE pretty low. So a 10/20 rating will be a 10 instead of a 0.5.

The baseline we are going to use is simple average prediction. We will calculate the average rating of all reviews in the training set, and predict the outcome every time.

We will try different models to achieve this task, and each model will use different attributes to form different features.

The first model we will try is rating prediction using Jaccard Similarity. Here, we need user name, beer name, and rating in each entry in the training dataset; we need user name, beer name to predict an unknown rating. This does not involve any regression, and the predictions are made based on the function that will be introduced in the Model part.

For processing the data, we will create a dictionary that stores the reviews each item receives in the training set. Then we will traverse the training set, add each beerID as key and append the user name to the corresponding value. This is the Jaccard Similarity based on items I. We will also try Jaccard Similarity based on users U, where the idea is the same.

The second model we will try is bag-of-words. Here, we need user review and rating in each entry in the training dataset; we need user review along to predict an unknown rating.

For processing the data, we will first count all words in all review texts and choose the first 1000 words as our dictionary. We will then yield a feature vector (2000 word counts + 1 bias term) for each review, which is composed of the dictionary word counts in the review text. More optimizations will be discussed in the Model section. After having the feature vectors, we will fit it to the ratings in the training set, and then use the fitted model to predict the ratings in the validation/test set.

3 Model

3.1 Baseline Model

The baseline model we used was the global average model. It calculated the global average rating in the training set, and then predicted the outcome for all ratings in the test set. The results were as shown in the table below:

Global Average	13.45

MSE(Valid)	8.55
MSE(Test)	8.79

This is the lower bound for our models, and we aim to at least outperform this in the following models.

3.2 Jaccard Similarity Model

First, we predicted the rating using rating prediction based on Jaccard Similarity between items. The function was:

$$r(u,\ i)\ =\ \frac{1}{z}\sum_{j\in I_u\setminus\{i\}} R_{uj} * JaccardSim(i,j)$$

$$z\ =\ \sum_{j\in I_u\setminus\{i\}} JaccardSim(i,j)$$

$$J(i,\ j)\ =\ |U_i \cap U_j| \ /\ |U_i \cup U_j|$$

U_x − The users who reviewed item x.

If z = 0, we would simply predict the global average.

The validation error for this model was 6.28, which is lower than 8.55, the validation error for the baseline model.

Then, we predicted the rating using rating prediction based on Jaccard Similarity between users, the function is similar to the last one:

$$r(u,\ i)\ =\ \frac{1}{z}\sum_{v\in U_i\setminus\{u\}} R_{vi} * JaccardSim(u,v)$$

$$z\ =\ \sum_{v\in U_i\setminus\{u\}} JaccardSim(u,v)$$

$$J(u,\ v)\ =\ |I_u \cap I_v| \ /\ I_u \cup I_j|$$

I_x − The items reviewed by user x.

If z = 0, we would simply predict the global average.

The validation error for this model was 5.27, which was lower than the last one, so we decided to continue optimizing based on this model.

The first optimization we tried was to slightly change the function to the following:

$$r(u,\ i)\ =\ \overline{R_u}\ *\ m$$

$$m\ =\ \frac{1}{z}\sum_{v\in U_i\setminus\{u\}} (R_{vi} - \overline{R_v}) JaccardSim(u,v)$$

$\overline{R_x}$ − $Average\ rating\ given\ by\ x$

All the others were the same.

The validation error for this model was 5.41, which was even slightly higher than 5.27 of the last one, so we dropped this optimization.

The second optimization we tried is to focus on the edge case. If z = 0, we chose to predict the average rating received by the item instead of the global average rating if the item had received any reviews. The validation error for this model was 5.07, which was better than the previous one with 5.27. Therefore, we decided to use this model as our final model in this subsection.

The final model's performance in this section was as the following:

MSE(Valid)	5.07
MSE(Test)	4.86

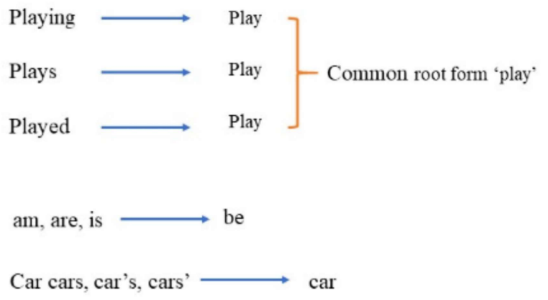
3.3 Bag of Words Model

We started from the most simple BoW model. We first counted all word frequencies in the training set's review texts and sorted the result. Then, we used the first 2000 most frequent words as our dictionary. For each review text, we converted it to a feature vector of length 2001, each entry in the vector represented the frequency of the corresponding dictionary word

in the text, and one bias term in the end. Then, we created a sklearn linear regression model and fitted the feature vectors in the training set to the ratings in the training set. At last, we used the model to predict the ratings in the validation set. The validation error for this model was 5.49.

The first optimization we did was to clean the review text by removing punctuation and capitalization. These could be easily done by using the python's built in packages. The validation error went down as expected to 5.03.

The second optimization we did was still related to cleaning the review text. However, we used a third party package 'nltk' for Natural Language Toolkit. We first stemmed all words in the review text using nltk.PorterStemmer. The examples of stemming words were as following:



Then, we removed all stopwords in the review texts, such as 'a', 'the', 'for'. The rule of removing was that we removed the word if it appeared in the nltk stopword list. The validation error went down to 4.99, where the reduction is lower than we expected. Therefore, we made several experiments, and the results were as following:

Features Used	Validation Error(MSE)
Stemmer, Stopwords	4.99
Stemmer	4.98
Stopwords	5.06
None	5.03

It seemed that stemmer was helping us reduce the error, but stop words were not. However, we all agreed that removing stop words would benefit us in the following tf-idf optimization, so we decided to keep the feature.

The last optimization we did was changing the feature vector entries. For each feature vector entry, we used "tf-idf" value instead of simply using the word counts. The formulas were shown below:

$$tf - idf(w,d) = tf(w,d) * idf(w)$$

$$idf(w) = \log_{10}(N/N_w)$$

$$tf(w,\ d) = \frac{\#\ of\ w\ in\ d}{\#\ of\ most\ frequent\ words\ in\ d}$$

$$w - word$$

$$d - document$$

$$N - \#\ of\ all\ documents$$

$$N_w - \#\ of\ all\ documents\ where\ w\ appears$$

The validation error went down to 4.86, which was better than all before. Therefore, we decided to use this model as our final model in this subsection.

The final model's performance in this section was as the following:

MSE(Valid)	4.86
MSE(Test)	4.97

4 Literature

We selected the dataset from the Standford papers that elaborates the research methods that distinguishes different parts of reviews and match them with different aspects of the product and discusses the effect of repetitive exposure to the product on their preference and the following

changes on the recommendation system to adapt this phenomenon.

4.1 Learning Attitudes and Attributes from Multi-Aspect Reviews

The first paper discusses the mathematical methodologies to divide a review into sections, each of which represent one aspect of the product.

The research is similar to the bag of words lectured in class. BOW characterize a review using every word in it. That is to say, a review is represented as the bag of its words whereas in this paper, the author determines which part of the bag characterizes an aspect of the product.

In this paper, there are a total of five different datasets to study, and the dataset we are using is one of them. They were studied in the similar fashion where they used both unsupervised and fully supervised learning to train the model by maximizing the log likelihood of corpus probability added by a regularizer using gradient descent, which is the most common way of learning nowadays. After conducting experiments, the research team performs rating predictions on aspect ratings instead of overall ratings, which generates an acceptable performance as that of the real world.

While the research topic chosen by this research team also involves the process of reviews and used content of the reviews ot predict the ratings, the team classifies the words in the reviews and uses these bags to predict ratings of the product based on their aspects instead of the overall ratings. Although our BOW performance exceeds that of baseline, we cannot draw analogies between our conclusion and theirs.

4.2 From Amateurs to Connoisseurs: Modeling the Evolution of User Expertise through Online Reviews

The second paper tries to elaborate that the consumption of the product will affect the user preference in the long run. That is to say, as consumer becomes more experienced in beer, their preference differs from what they will choose when they are less familiar with beers of different aspects.

This is the only dataset they used for the entire research as it is already large and representative enough. In this research, the study member trained this dataset by minimizing the mean square error. They optimize the parameters using L-BFGS, a quasi-Newton method for non-linear optimization, a method that is widely used as we discusses the Newton's method in class.

They finally confirming their assumptions that Users' tastes and preferences change and evolve over time, and our research does not involve the how the change of timestamp will affect the ratings. However, we may draw analogies to the fact that when training the dataset, the research team tries to minimize the mean squared error, which coincides with our attempt to our goal in predictive task.

5 Results

Both the Jaccard similarity model and bag of words model outperform baseline. None of these models failed. However, some of the models may sightly outperform the others by some tiny optimizations.

5.1 Jaccard Similarity Model

With Jaccard similarity model, the baseline model obtains MSE of 8.55 with validation set. However, for all of the three possible models here, their MSE floats around 5, which hugely outperforms the baseline model that uses global

average to predict as edge cases when the user product pair is not in the set. Therefore, we have reached an effective method to estimate the rating given by a user. We study the rating of a beer given by a user using Jaccard similarity.

Without optimization, our MSE is 5.27. After subtracting the rating with to the average rating of that user as the first optimization method, the MSE on validation set rises to 5.41. However, because of the close proximity of these two data, we cannot say the this optimization slows down the performance or not. Nevertheless, we can say that the second optimization, replacing the global average rating to the average rating of that specific user as the edge case, the MSE for validation set drops to 5.07, which is a huge improvement. The reason why this optimization is particularly successful is obvious: each user tend give similar ratings on the beers with similar aspects. Replacing global average rating with users' average rating is more sensible.

5.2 Bag of Words Model

With bag of words model, the baseline model obtains MSE of 8.55 with validation set. The final validation set MSE falls below 5, which hugely outperforms the baseline model that uses global average to predict as edge cases when the user product pair is not in the set. Therefore, we have reached an effective method to estimate the rating given by a user. We study the rating of a beer given by a user using the number of distinct of words and their occurrences in all the reviews as a huge feature vector.

Based on the table where we generated by applying stemmer or stopwords, we can see that the validation MSE is 5, meaning that using stemmer or stopwords optimization does not affect the overall MSE. This is probably because that both of these text processing techniques have limited influence on the text itself. It may be that there are very few stopwords in the original text, or that the active user who post huge number of reviews also take the lead both with and without these words.

Finally, we applied a successful optimization by using term frequency and inverse document frequency. After this optimization, we lowered the validation MSE to 4.86, which outperforms all other possible models.

Therefore, although all of the models outperforms the baseline model, the bag of words model with tf-idf optimization is the one that obtains the lowest validation MSE.

6 Reference

Learning attitudes and attributes from multi-aspect reviews
Julian McAuley, Jure Leskovec, Dan Jurafsky
International Conference on Data Mining (ICDM), 2012

From amateurs to connoisseurs: modeling the evolution of user expertise through online reviews
Julian McAuley, Jure Leskovec
WWW, 2013