Brewing up Recommendations: Collaborative Filtering for Beer Recommendations

Zachary Labkovski

Understanding Markets with Data Science

Table of Contents

Introduction:

Throughout the course of the last 10-15 years, technology has become increasingly involved in our day to day life. With technology at the forefront, convenience has not only become valued, but expected, ranging from streaming services that put entertainment at the forefront to online shopping and delivery services that allow consumers to make purchases from the comfort of their home. Since the start of the COVID-19 pandemic, these trends have been further exacerbated, with companies like DoorDash taking over the food delivery industry, and Drizly becoming the industry leader in alcohol delivery as "your online liquor store". Unlike DoorDash, Drizly is a true online storefront that focuses on products, not providers. As a result, it's imperative they have an optimal recommendation system in place to maximize purchases. Because Drizly does not publicly release user ratings, we used a dataset from BeerAdvocates, the world's largest collection of beer reviews, to simulate the recommendation model development process.

For this project, I used data provided by BeerAdvocate, a forum for beer reviews and promotion that has been around since 1996. The dataset consists of over 1.5 million beer reviews in a csv file that can be downloaded from Kaggle at this link. For each review, the dataset contains a brewery ID number, the brewery's name, the time of the review, the overall review, the review of the aroma, the review of the beer's appearance, the review of the palate, the review of the taste, the profile name associated with the review, the beer style, the beer name, the ABV (alcohol by volume) of the beer, and the beer ID. All review values are on a 1-5 scale, with 1 being the worst and 5 the best, allowing for increments of 0.5.

Given the size of the dataset and number of features, it may be wise to implement multiple techniques to construct a recommendation system that is actually effective in recommending compatible beers to users. The overall methodology I used was collaborative filtering; however, different similarity metrics are necessary to explore which one produces the most valuable results. Moreover, there are intrinsic properties in the data that may make it more suitable for user-based or item-based collaborative filtering. As a result, I tried a variety of similarity measures, starting with Jaccard similarity on binary review data (whether a user reviewed a specific beer). I then used overall reviews to create ternary data to be able to implement a cosine similarity-based system, with a user's review taking on a value of 1 if it is at or above a 3, taking on a value of -1 if it is below a 3, and a value of 0 if a user did not review a specific beer. Finally, I attempted to implement a model based on Pearson Correlation using overall reviews (scaled 1-5). For all of these similarity measures, I implemented models using both user-based and item-based collaborative filtering and tested all models using both train-test split and cross-validation.

Literature Review:

There has been limited academic research on recommendation systems developed for beer drinkers; however, I was able to come across a few projects other graduate-level students have developed on different datasets.

The first came from Tim Childers, then a student at the University of Colorado, who developed an app called "BeerBro" containing a recommendation system using ratings from ratebeer.com. Tim's analysis was built on a database of over 9 million beer reviews on over 250,000 unique beers. Each review contained scores for look, smell, taste, feel, and overall score, similar to those in my dataset. The reviews also included notes containing text descriptions of the beer, along with objective beer characteristics, such as style, ABV (alcohol by volume), availability, original brewery, and geographic origin (state and/or country). Rather than a comparative approach like I used, Childers had a fairly set methodology, using NLP to featurize text reviews before implementing an SVD based collaborative filtering model.

Another project that examined beer reviews to implement a recommendation system came from a group of PhDs enrolled in the NYC Data Science Academy in early 2017, a system they called "NINKASI", named after the Sumerian goddess of beer. They later developed their recommendation system into a web app using Flask. Rather than relying on a downloaded dataset, the group scraped over 280,000 reviews from ratebeer.com, limiting their scraping to only the top 20-25 beers from all US states plus Washington, D.C. The group used two different modeling methods, the first being a content-based method based on Natural Language Processing and Latent Semantic Analysis, followed by an application of cosine similarity to form recommendations. Their second approach was a collaborative filtering based method, using both Singular Value Decomposition and a Restricted Boltzmann Machine. In the end, the project combined the two collaborative filtering approaches in a linear ensemble to output the final recommendations.

Data Collection and Preparation:

For my analysis, I downloaded a dataset collected from BeerAdvocate that can easily be found on Kaggle. The dataset contained over 1.5 million reviews, with each review containing brewery name and ID number, time of review, overall review, reviews of individual characteristics (aroma, appearance, palate, and taste), user profile name, and beer information including name, style, ABV, and ID. All ratings were on a 1-5 scale that allowed for ratings in 0.5 increments. In order to implement my desired collaborative filtering models, I first needed to alter the data to a more usable form.

Because of the nature of collaborative filtering, in which single values are compared across multiple users/items, it was necessary to cut ratings down to a single value; as a result, my first step in data preparation was reducing the data to three columns: Profile Name (for user identification), Beer ID (for item identification), and Overall Review (to compute similarity). After that, I removed all NA rows in the dataset and any duplicate ratings (i.e., ratings of the same beer by the same user).

One final concern about the usability of the data was the size of the dataset; with over 1.6 million reviews, computational power and efficiency were in question. As this was mostly a test in viability of various modeling techniques, the exercise was mainly to provide proof of concept with an assumption of scalability. Therefore, I sought to greatly reduce the size of the input data to roughly 500,000 reviews. After using histograms to visualize aggregate data on both number of reviews per user and per beer, I adjusted minimum requirements for inclusion until I reached my target threshold. In the end, the numbers I ended up implementing were a minimum of 500 reviews for a beer to be included and 100 reviews for a user to be included.

Finally, for the data to be compatible with R's recommenderlab package for collaborative filtering, I had to convert the data to type "realRatingMatrix".

Analysis Process:

As aforementioned, my primary goal in this process was to compare different collaborative filtering models to determine which methods would be most effective in producing beer recommendations. This required evaluating different combinations of model types–in both user- and item-based collaborative filtering–in addition to different similarity measures.

In terms of model types, there are two options to select when working with collaborative filtering: user-based and item-based. For the sake of this implementation, user-based collaborative filtering looks at similarities between users' preferences (i.e., for a given user, which other users tried similar subsets of beers) and then recommends beers based on this similarities. On the other hand, item-based collaborative filtering looks at similarities between beers (i.e., for a subset of users that frequently tried a specific beer, which other beers they most often also had) to recommend beers for a user.

For similarity measures for my implementation, I looked at three main options: Jaccard similarity, cosine similarity, and Pearson correlation. Jaccard similarity uses binary data, with a user assigned value 1 if that user rated a beer, and value 0 if that user did not rate that beer. Similarity is computed using the below formula, the size of the intersection of two users' sets of beers rated divided by the size of the union of their sets of beers rated (or in the case of item-based collaborative filtering, the size of the intersection of two beers' sets of users divided by the union).

The next similarity measure I used was cosine similarity, which, unlike Jaccard similarity, uses

$$J(A, B) = \frac{|A \cap B|}{|A \cup B|}$$

ternary data, with 3.5 as the "good rating" threshold. That is, if a user gave a beer a rating of at least 3.5, they would be assigned value 1; if they gave that beer rating below a 3.5, they would be assigned value -1; if they did not rate a given beer, they would be assigned value 0. Cosine similarity is calculated using the below formula, which represents the dot product of vectors divided by the product of their magnitudes.

$$\cos \theta = \frac{\vec{a} \cdot \vec{b}}{\|\vec{a}\| \cdot \|\vec{b}\|}$$

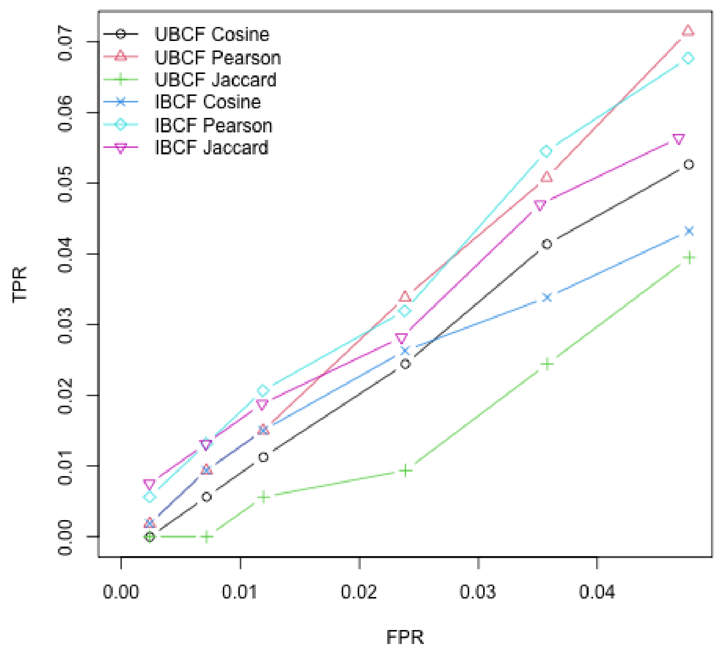
The final similarity measure is Pearson correlation, which simply uses the overall rating from each review on the 1-5 scale. Pearson correlation is calculated using the below formula, which is based around rating deviations from the mean.

$$r = \frac{\sum(x-\overline{x})(y-\overline{y})}{\sqrt{\sum(x-\overline{x})^2 \sum(y-\overline{y})^2}}$$

Between the two model types and three similarity measures, there were six total models to compare. Additionally, I decided for evaluation to use both a simple train-test split and cross-validation, resulting in a total of twelve separate implementations. After implementing the models using the "evaluationScheme" and "evaluate" methods in recommenderlab, I used true positive rate and false positive rate for evaluation.

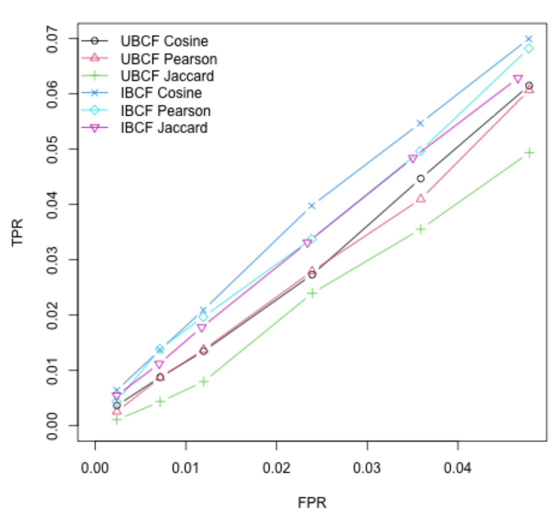
Findings:

The following graph shows false positive rate (on the x-axis) and true positive rate (on the y-axis) for all six implemented models using an 80/20 train-test split.



With the train-test split used, Pearson correlation for both item-based and user-based collaborative filtering were most effective, with Jaccard similarity the worst for user-based collaborative filtering and cosine similarity performing worst for item-based collaborative filtering.

The following graph shows false positive rate (on the x-axis) and true positive rate (on the y-axis) for all six implemented models using cross-validation, which tends to be a more robust evaluation and likely is more indicative of what results would look like on larger-scale data.



Here, all similarity measures performed better in item-based collaborative filtering than in user-based collaborative filtering. Additionally, in both item-based and user-based collaborative filtering, cosine similarity was the most effective, slightly edging out Pearson coefficient; however, Jaccard similarity performed far worse than both of the other similarity measures in both model types.

Conclusion:

Based on the results from both train-test split and cross-validation, the safest bet for an effective beer recommendation system seems to be item-based collaborative filtering, with Pearson coefficient having the best combination of results across the two implementations. As aforementioned, scalability is assumed in this analysis; however, the modeling approach's scalability would need to be tested for future implementations of larger datasets.

Appendix/Code:

Code and data are provided in the zip file submitted on Canvas.

References for literature review:

1. NYC Data Science Academy. (n.d.). Ninkasi Beer Recommender System. https://nycdatascience.com/blog/student-works/ninkasi-beer-recommender-system/
2. Childers, T. (2018). BeerBro. GitHub. https://github.com/timchilders/BeerBro