Personalized Beer Recommendation

Qiong Wu
Georgia Institute of Technology
qwu304@gatech.edu

Yan-lin Chen
Georgia Institute of Technology
ychen3017@gatech.edu

Yang Yu
Georgia Institute of Technology
yyu391@gatech.edu

Yinuo Jiang
Georgia Institute of Technology
yjiang375@gatech.edu

Zixin Wang
Georgia Institute of Technology
zwang802@gatech.edu

Zhijian Guo
Georgia Institute of Technology
zguo321@gatech.edu

1. INTRODUCTION

Beer, as one of the oldest drinks humans have produced, dates back to 5000 BC[1]. In 2017, the beer market generated US$281 billion revenue worldwide, with total beer consumption over 144 billion liters[2]. There are over 300,000 different beers worldwide[3], and more than 150 different types are classified according to the Brewers Association[7]. A comprehensive, innovative beer recommendation system would be valuable, especially to beer lovers.

2. PROBLEM DEFINITION

To build a beer recommender characterized by users' preferences of select beers.

3. SURVEY

In general, online reviews have a decent impact on the beer industry. A study by Erik K. Clemons et all[5] suggests that a beer that is highly differentiated usually acquires market shares faster. However, these beers normally have barbell-shaped ratings, with positive quartile of reviews being extremely positive. These findings further illustrate the needs of having an effective algorithm recommending highly differentiated beers accurately in the beer industry.

There are two filtering algorithms popularly utilized in recommendation systems[8]. The first algorithm is content-based filtering, which recommends content that best match users' preferences over a domain of keywords, often created and updated automatically in response to user's desirability of items (concluded from paper[14] to [18]). The second algorithm is collaborative filtering, which recommends items that like-minded users appreciate. This algorithm "predicts" users' potential preferences in the future. However, as the algorithm recommends in real time in correspondence to all users' feedback, scalability and time complexity are major factors to be addressed (concluded from [9] to [13]).

On top of these two popular algorithms, we also notice a third algorithm: the text-mining-based algorithm, suggested by Ashwin Ravi Ittoo et al in their paper [24], aims to overcome the weaknesses of the collaborative filtering and content-based filtering algorithms in order to achieve better recommendation results. This algorithm aims to discover patterns in unstructured text in order to generate recommendations.

Our research has also revealed that there are very few recommendation systems providing innovative visualizations to make the recommendation process transparent and interactive. The Beer Recommender website [4] simply outputs a list of beer names and limited information. A project BeerViz done by Divya Anand et all [25] provides a nice visualization of different types of beers and their similar brands, however it is missing the element of recommendation. Therefore, our project aims to convert the final recommendation into visualizations for users to make their next beer choices.

4. INNOVATION

i. Multiple different algorithms – content-based filtering, collaborative filtering, attribute weighting, and text mining algorithms are explored in our recommendation engine.

ii. Two recommendation user interfaces (UI): users are able to either enter their current user ID with beeradvocate.com, or select a style and rate respectfully among the attributes of appearance, aroma, palate, and taste to quantify their preferences. The system will accordingly generate the top 10 recommendations.

iii. Results from the recommendation engine are visualized through a dynamic bubble chart. For instance, the recommendation list will generate ratings represented by 10 various sizes of circles. The size of the circle and the number in each circle reflect the likelihood of a user enjoying the beer. The opacity of the circle reflects the popularity of the beer which is determined by the number of reviews on BeerAdvocate.com for each beer.

5. DISTRIBUTION OF EFFORT

All team members have contributed a similar amount of effort.

6. PROPOSED METHOD

i. Intuition

We explored three algorithms in our recommendation system to analyze online beer reviews on BeerAdvocate.com.

The first approach applies content-based filtering and collaborative filtering algorithms to get recommendations. In the content-based filtering we group beers into clusters with similar attributes. User's inputs are passed into the model to predict the cluster, and then beers from the same cluster will be recommended. In the collaborative filtering algorithm, similar users are grouped together and their shared beers are scored by assigning similarity weights.

The second approach utilizes optimization to assign weights to the 4 attributes (appearance, aroma, palate, and taste), so that beers with similar weightings are more likely to be recommended together.

ii. Recommendation Algorithms

- Content-based filtering

Find similar beers based on the beer reviewed by the user. We explored two clustering algorithms, namely K-Means and Expectation-Maximization to group beers into clusters based on their attributes. The dimensionality reduction method was utilized to determine if it is able to improve the model performance. The resulting clusters were visualized to select the best algorithm.

When a new user inputs their preferences of 4 beer attributes, this new data point will be fed into the model to determine which cluster the user's preference belongs to. Next, the algorithm recommends the top-rated beers according to overall ratings.

- Collaborative filtering

Calculate the correlation coefficients (similarities) among users and recommend beers favored by similar users. We filtered data to reduce bias (such as filtering out beers with less than 200 reviews). Afterwards we completed the following steps:

a. *find beers rated by both users and build user-beer matrix*

b. *similarity(user, other users) = S(x, y)/sqrt(S(x, x)*S(y, y))*
   *S(x, y)=sum(x*y)-sum(x)*sum(y)/n*
   *S(x, x)=sum(x*x)-[sum(x)*sum(x)]/n*
   *S(y, y)=sum(y*y)-[sum(y)*sum(y)]/n*

c. *filter similarity > threshold and sort descending*

d. *find beers that is rated by other users but not reviewed by the user*

e. *calculate weighted scores for these beers and sort descending*
   *weighted sores = sum(beer score * similarity(user, other users)) / sum(similarity(user, other users))*

f. *return top 10 scores sorted by scores and number of reviews*

In the exploration of the content based filtering, we discovered that for new users each time re-training the model took significant time during the implementation. Considering that a timely response to user's queries is crucial to a positive user experience, in order to obtain a similarity matrix we explored singular value decomposition (SVD) to find latent features within a reduced running time. Before calculating coefficients, we utilized Truncated SVD to reduce variables. We managed to achieve our codes generating recommendations within a reasonable run time.

- Attributes weights

We mined the text reviews and compiled a dictionary of keywords for aspects of beers. For each beer, we used constrained convex optimization to calculate its weight on four attributes of one beer, with sentiment analysis on review text to further improve the rating accuracy. When new users rate their preferences from 1 to 5 on the four attributes, we recommend a group of beers sharing similar weights based on the historical review database.

iii. User interface

- Input

Two options are offered to users. With option one, if a user is a current user of BeerAdvocate.com and has provided reviews before, by entering their user ID, our application will produce a personalized recommendation list. (Figure 1)

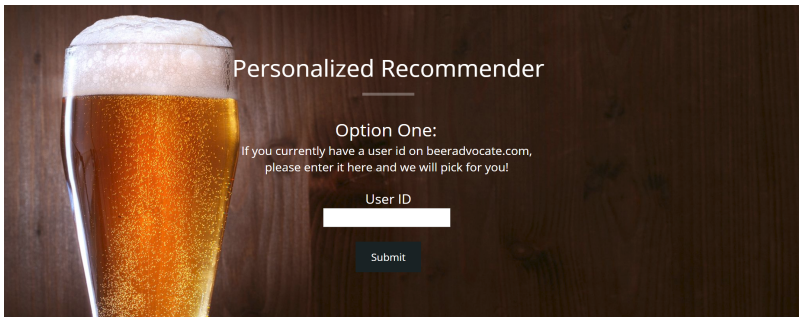


Figure 1- input with user ID

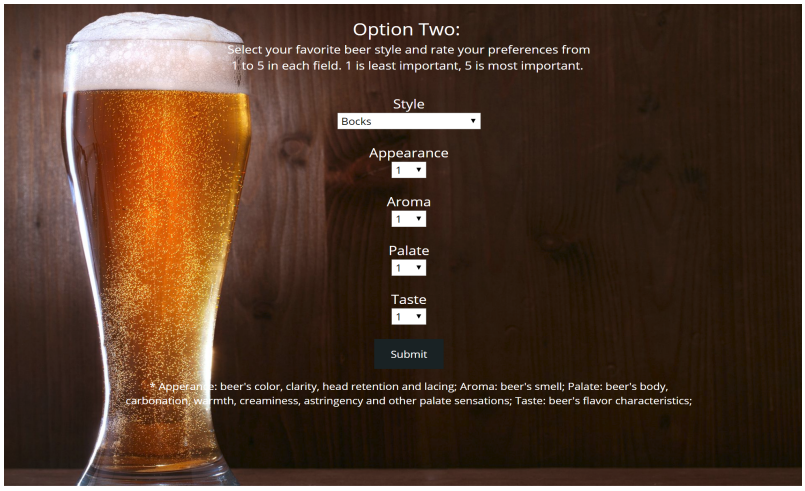


Figure 2- input with attributes preference

With option two, for non-current users of BeerAdvocate.com, they can select a beer style they prefer and rate their preferences among beer attributes including Appearance, Aroma, Palate, and Taste. (Figure 2)

- Output

Our application will initially provide 10 recommended beers based on a user's inputs. Each circle represents a recommended beer. Each color represents a style. The size of the circle and the number in each circle reflects the likelihood of a user enjoying the beer. The opacity of the circle reflects the popularity of the beer which is determined by the number of reviews on BeerAdvocate.com for each beer. According to our algorithm engine, larger circles and larger rating numbers indicate a higher chance that a certain beer will be enjoyed by this one user. (Figure 3)

Figure 3- output with 10 recommended beers

Together with the recommended beers, our recommender is also able to generate a unique word cloud, based on the reviews exclusively for the user's 10 recommended beers. This word cloud provides characteristics of the 10 recommended beers by extracting the keywords from reviews for the same beers that are contributed from other users, as shown in Figure 4.

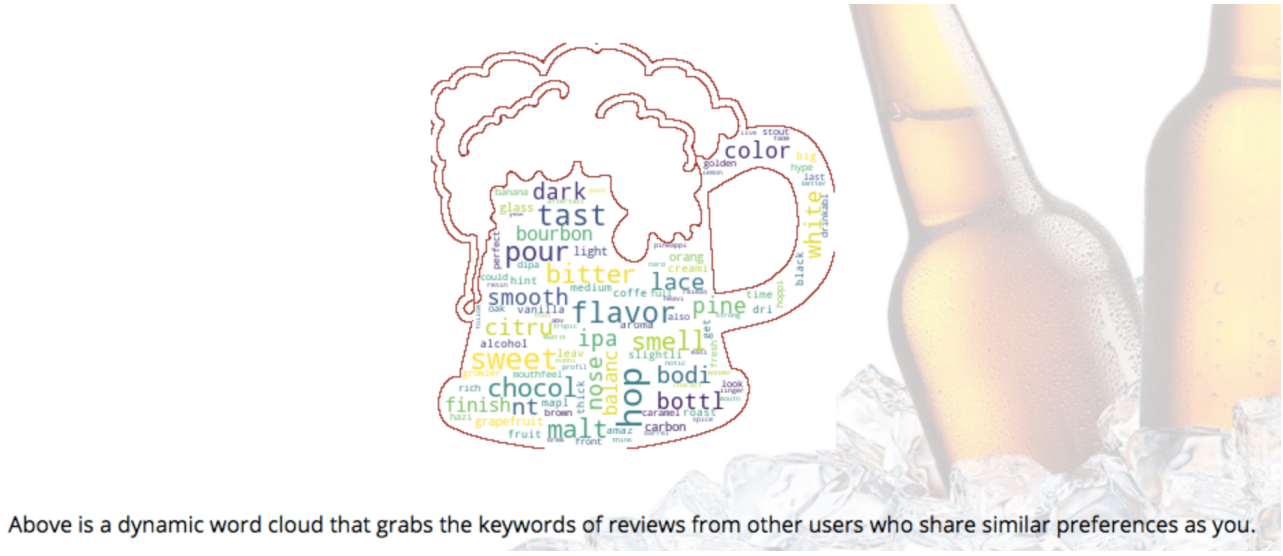


Figure 4- unique word cloud

Additionally, our user interface provides some interesting information on beers, such as providing a general guide that introduces different beer styles as shown in Figure 5:

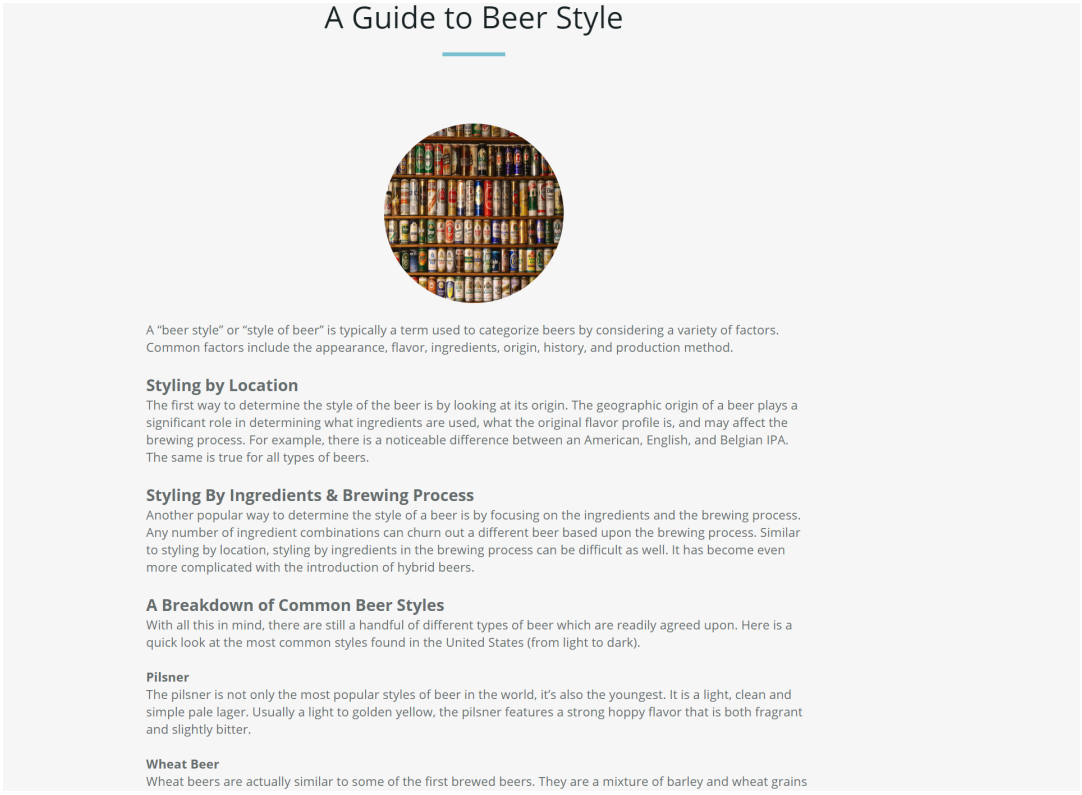


Figure 5- guide to beer styles

Furthermore, our website summarizes the popularity of beer styles among the users' reviews on BeerAdvocate.com.

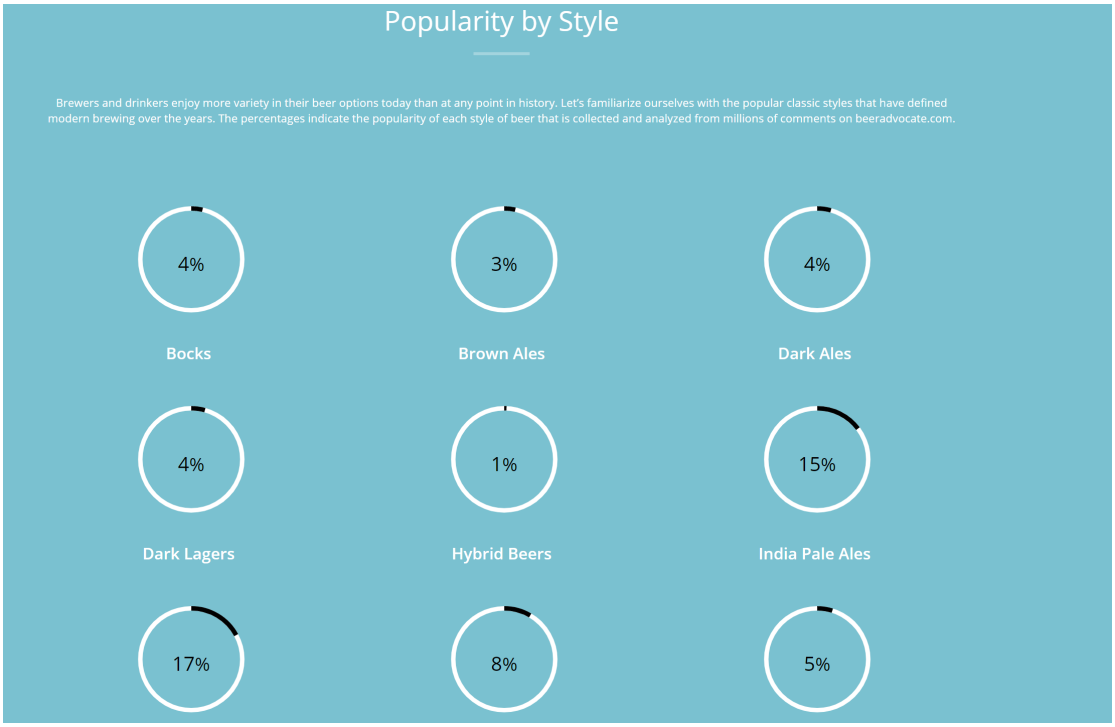


Figure 6- beer popularity by style

Lastly, our website is able to link to several noteworthy beer blogs shared on external websites.

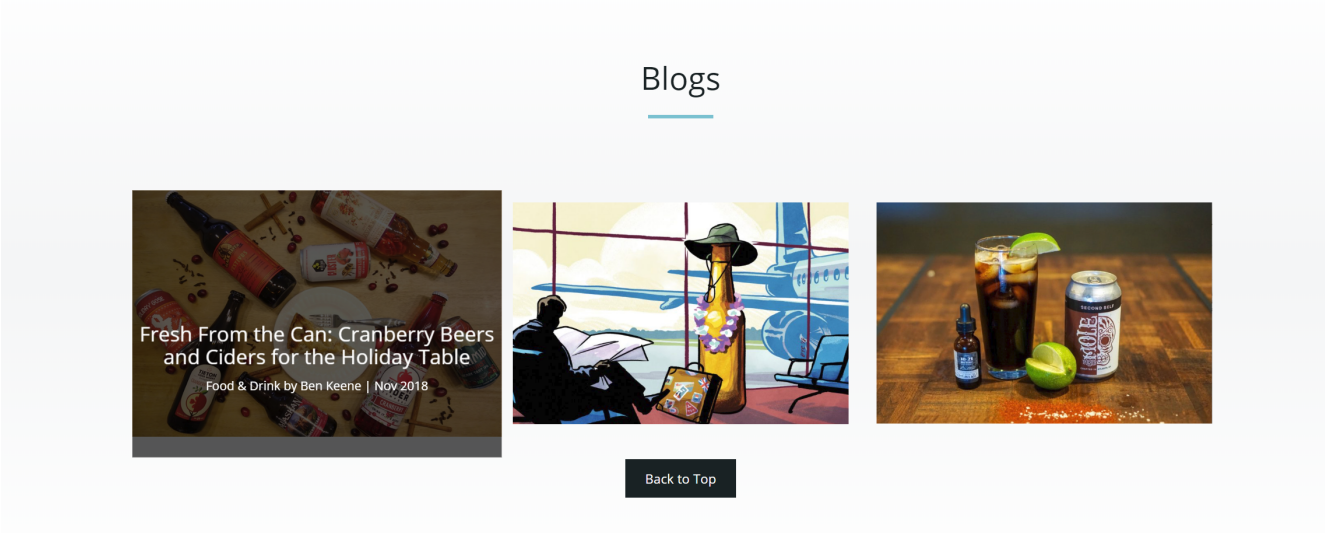


Figure 7- Links to beer blogs

7. DESIGN OF EXPERIMENTS

i. Question to answer

a) Which clustering algorithm produces the best clusters – K-Means clustering or Expectation Maximization?

b) What is the optimal number of clusters to group beers?
c) Does the dimensionality reduction method such as Principle Component Analysis improve clustering result?
d) What is the optimal number of Principle Components?
e) With the dimensionality reduced, what is the optimal number of clusters?
f) Which algorithm – optimization or clustering, produces the most accurate beer recommendations?
g) What is the optimal number of components in SVD?
h) In collaborative filtering, which method meets the expectation on query time and recommends the list?
i) Assign weights to recommend lists from different algorithms.

ii. Experiments and Observations

- Content-based algorithm experiments

The first experiment we tested is to determine the optimal number of clusters in order to group beers. To make this happen, we ran K-Means algorithm using 1 to 20 clusters, and plotted the Within-Cluster Sums of Squares (WCSS) against the number of clusters.

Using the elbow method, we concluded that using 7 clusters is most appropriate for this dataset. Next we ran K-Means clustering using 7 clusters and visualized the result using pairwise plot.

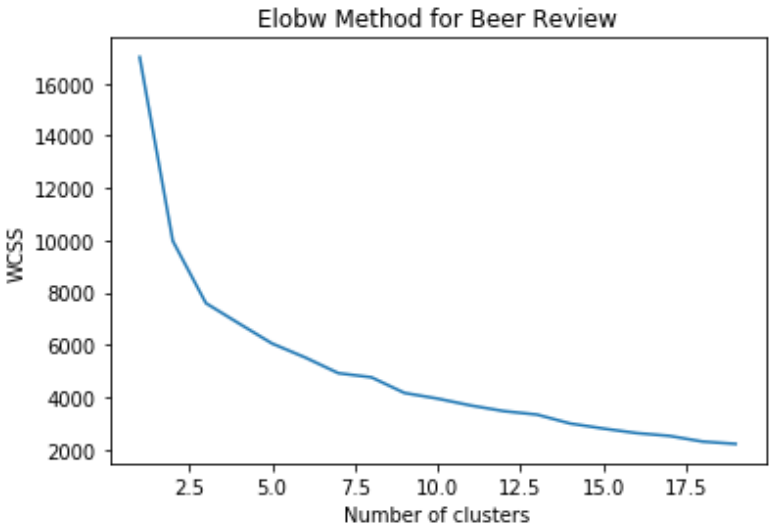


Figure 8. WCSS against number of clusters

From Figure 9, we cannot see the data points being grouped into distinctive clusters, since there are 4 dimensions. We utilized Principle Component Analysis to attempt to reduce the dimensionality.

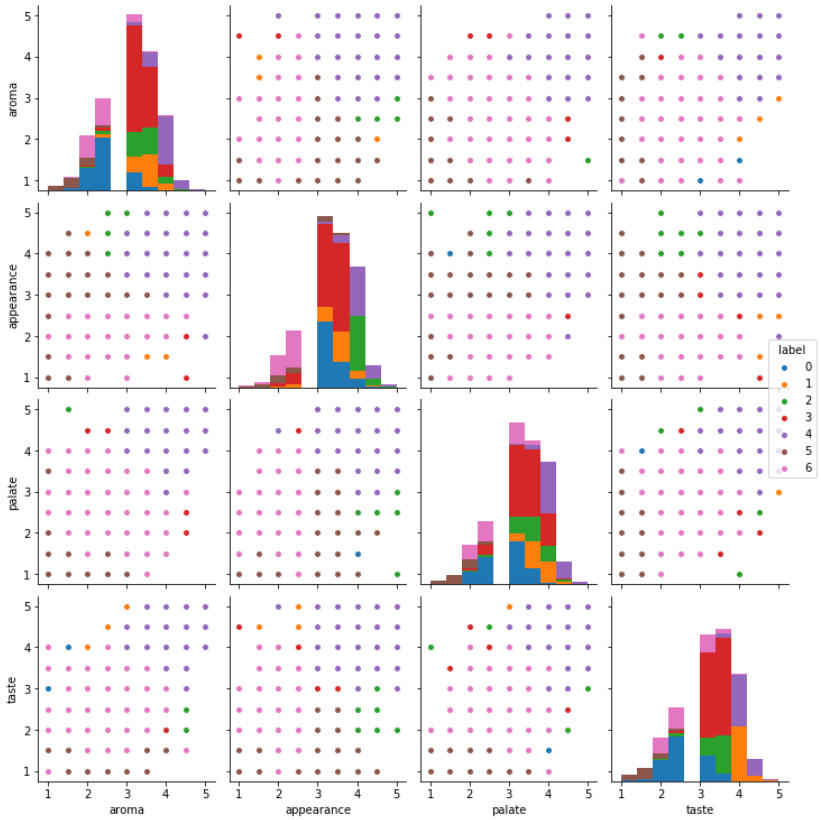


Figure 9. Pairwise plot of Beer features

It is demonstrated in Figure 10 that the first two components already explained over 80% of the variance in the data. Therefore, we decided to use only the first two components in the clustering algorithm.

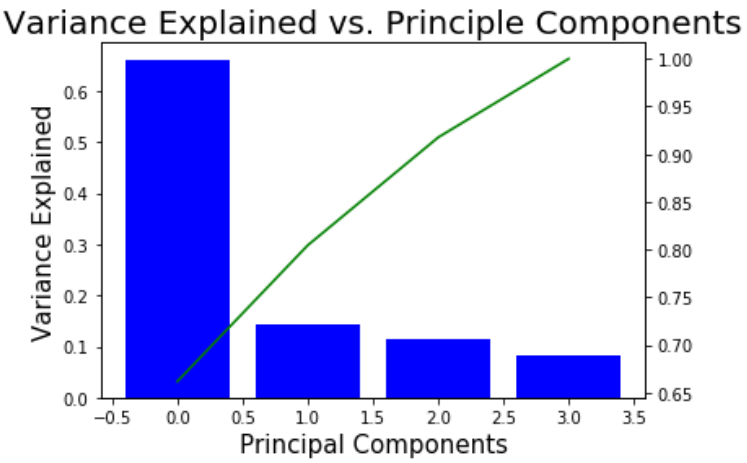


Figure 10. PCA Variance Explained vs Principle Components

Using the transformed data, we ran the same WCSS analysis across 20 clusters to determine the optimal number of clusters. Based on Figure 11, we again determined that the optimal number of clusters is 7.

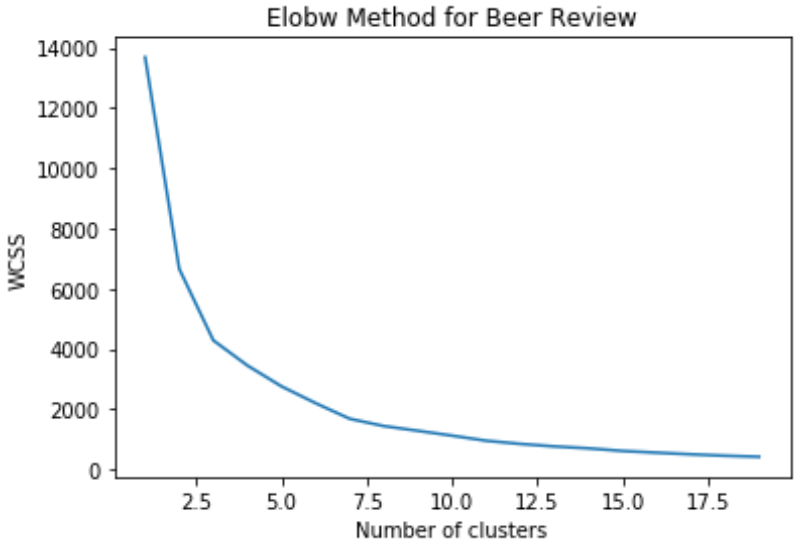


Figure 11. WCSS against number of clusters after PCA

With the dimensionality reduced data, and 7 clusters, we ran the K-Means algorithm again. The new clusters are visualized in Figure 12. The clusters now become distinctive.

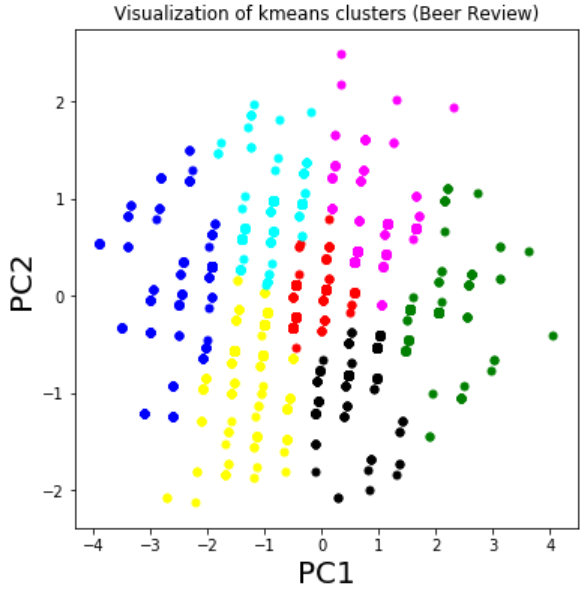


Figure 12. Clustering after PCA

We also used a different clustering algorithm – the Expectation-Maximization algorithm. Figure 13 shows the clusters obtained by the EM algorithm. Compared to Figure 12, it is clear that K-Means produces significantly better clusters.

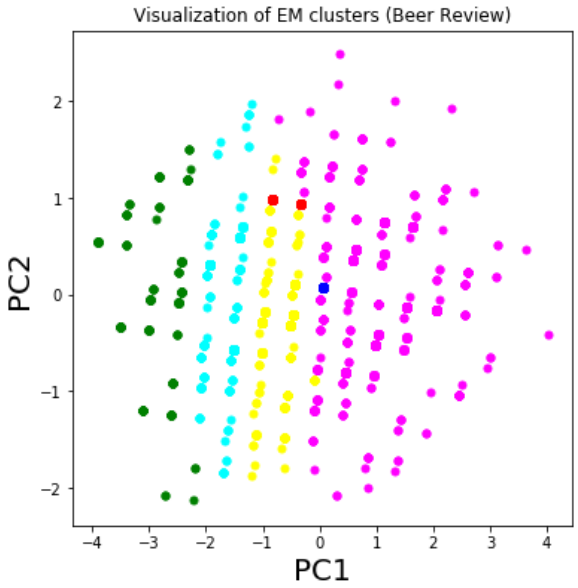


Figure 13. Clustering using Expectation-Maximization

- Collaborative Filtering experiments

In collaborative filtering, we tried three different designs. We evaluated these designs by the performance of running time.

At the beginning, we stored data in pandas.DataFrame in python. We calculated correlation coefficients using built-in functions. However, the result failed to meet our satisfaction.

Further, we tried the machine learning method, truncated SVD, intending to generate the similarity matrix for all users. Truncated SVD was able to reduce factors, then we calculated similarities between users. We preloaded this user-user matrix, then queried it when needed. However, as a result, we discovered that loading this 20 G matrix took over 1 hour. It is a sparse matrix and not very efficient in both loading and querying.

In order to improve the performance of running time, we redesigned the data structure. In general, dictionary is the fastest data structure to query. We calculated the bottleneck component and stored the result in a file (.csv or .JSON). We were able to load it first and then query. However, it still took 30 minutes to load this file.

Ultimately, we compromised to a solution with preload only {user : similar users} data. It was a trade-off of loading time and calculating time.

- Word cloud experiments

The initial word cloud obtained by our text mining algorithm contains many generic descriptive words, such as “good,” “nice,” etc. These words do not provide meaningful information to the users, therefore we removed them. After several iterations of experiments, we were able to generate much more meaningful word clouds as shown in the below figure.

Figure: word-cloud 1

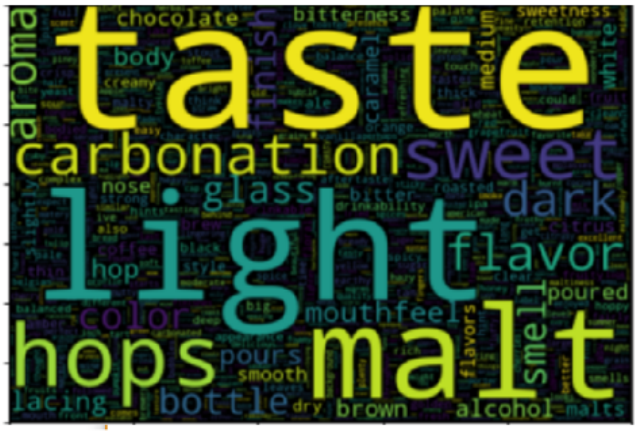
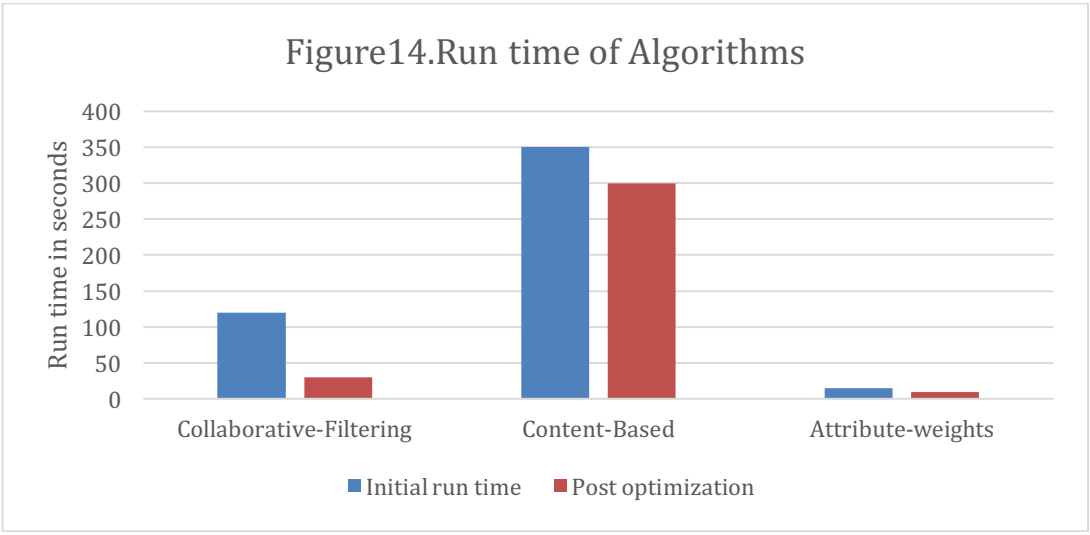


Figure: word-cloud 2

- Run time experiments

Running time of our recommender system is an important aspect to user experiments. As displayed in Figure 14, the attribute weights algorithm performed the best, and took less than 10 seconds to run. The collaborative filtering algorithm ran quite slowly in our initial setup. However, we managed to reduce the run time significantly after storing user-user relations in smaller sized files and running calculations only when required. The content based algorithm performed the worst and requires more than 5 minutes to generate the output. This is due to the need to re-train the clustering model every time a user sends an input. We tried several experiments, but were unable to reduce the run time significantly. Therefore, this algorithm was dropped in our final engine due to its negative impact to user experience.



- Input experiments

Originally, we allowed users to manually input attribute weights. For instance, users were able to weigh the attributes of beers by entering any integer or floating number. However, from the feedback from several volunteers who tested the website, it was clear that this flexibility was not

standardized and negatively affected user experience. As a result, we changed it to a dropdown menu of scores from 1 to 5, which allows users to easily select their preferences with one click.

- Recommendation visualization experiments

Initially, our recommendations appeared as a plain list and users had to thoroughly read to obtain complete recommendation information. We decided to instead use the bubble chart, which visualizes the results in a more straight forward manner. In addition, we categorized a total of 104 different beer styles into 14 major beer styles according to BeerAdvocate.com, and made the bubble colors differ based on the specific beer style. To increase data-ink ratio, we added an additional dimension, which is the opacity of the bubble. The higher the opacity means the beer was mentioned more times among user reviews on BeerAdvocate.com. This method enables users to quickly have an idea about the recommended beers with just one look, as opposed to reading the text line by line. We also tried small bubbles in a larger bubble to represent minor beer styles in a major beer style, but we failed to achieve this due to the complexity. As a result, this feature was omitted.

- User experience (UI) design experiments

We also incorporated css and jQuery techniques that enhanced the user experience design, such as full-screen responsive background image, parallax scrolling effect, and animated circles.

8. CONCLUSION

Overall, we successfully built a beer recommending system, which utilizes the three algorithms - collaborative-filtering, content-based filtering, and attribute weights, and also one visualization tool - word cloud by text mining technique. Several experiments were additionally performed that eventually improved the accuracy of the top beers recommended, as well as to ensure that the running time was acceptable. In terms of visualization, the dynamic and interactive website design was achieved by D3.js and jQuery which enables our recommendation website to be a fun exploration tool for beer lovers to seek their next favorite beers.

9. FUTURE WORK

A new functionality can be added to allow users to register and login, enabling their profiles to be saved into a database. A new page can be created to allow users to leave reviews of beers, and their reviews will be added to the database.

Users can also sign up for a subscription by email address, and the recommender system will periodically send them the latest recommendations when additional reviews are received. Further, additional methods can be tested to reduce the run time to enhance user experience.

10. REFERENCES

[1] History of beer, https://www.wikiwand.com/en/History_of_beer

[2] Alcoholic Drinks Report 2018 – Beer, https://www.statista.com/study/48816/alcoholic-drinks-report---beer/
[3] How many beers are there? https://www.beeradvocate.com/community/threads/how-many-different-beers-are-there.525578/
[4] Beer Recommender, http://www.recommend.beer/recommend/
[5] Clemons, Guodong Gao, and Hitt. "When Online Reviews Meet Hyperdifferentiation: A Study of Craft Beer Industry." *System Sciences, 2006. HICSS '06. Proceedings of the 39th Annual Hawaii International Conference on* 6 (2006): 116c.
[6] Untapped, https://untappd.com/
[7] 2018 Beer Style Guidelines, Brewers Association, 2018.
[8] Leidy Esperanza MOLINA MOLINA FERNÁNDEZ. ”Recommendation System for Netflix”. *VRIJE UNIVERSITEIT AMSTERDAM Research Paper*(2018).
[9] Breese, John S., David Heckerman, and Carl Kadie. “Empirical Analysis of Predictive Algorithms for Collaborative Filtering.” *The Fourteenth conference on Uncertainty in artificial intelligence* (1998): 43-52.
[10] Sarwar, Badrul, George Karypis, Joseph Konstan, and John Riedl. "Item-based Collaborative Filtering Recommendation Algorithms." *Proceedings of the 10th International Conference on World Wide Web* (2001): 285-95.
[11] Xiaoyuan Su, and Taghi M. Khoshgoftaar. "A Survey of Collaborative Filtering Techniques." *Advances in Artificial Intelligence* 2009 (2009): 1-19.
[12] Lyle H. Ungar and Dean P. Foster ”Clustering Methods for Collaborative Filtering”, *AAAI Technical Report* WS-98-08 (1998): 114-129.
[13] Schafer, J. Ben B., Dan Frankowski, Shilad Sen, and Jon Herlocker. "Collaborative Filtering Recommender Systems." *Lecture Notes in Computer Science (including Subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)* 4321 (2007): 291-324.
[14] Pazzani, Michael J. J., and Daniel Billsus. "Content-based Recommendation Systems." *Lecture Notes in Computer Science (including Subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)* 4321 (2007): 325-41.
[15] Melville, Prem, Raymod Mooney, and Ramadass Nagarajan. "Content-boosted Collaborative Filtering for Improved Recommendations." *Eighteenth National Conference on Artificial Intelligence* (2002): 187-92.
[16] Mooney, Raymond, and Loriene Roy. "Content-based Book Recommending Using Learning for Text Categorization." *Proceedings of the Fifth ACM Conference on Digital Libraries* (2000): 195-204.
[17] Chumki Basu, Haym Hirsh and William Cohen.”Recommendation as Classification: Using Social and Content-Based Information in Recommendation.” *AAAI-98 Proceedings* (1998).
[18] Ma, Ke. "Content-based Recommender System for Movie Website." Dissertation (2016).
[19] Li, Lu, and Xuefeng. "A Hybrid Collaborative Filtering Method for Multiple-interests and Multiple-content Recommendation in E-Commerce." *Expert Systems With Applications* 28, no. 1 (2005): 67-77.

[20] Nathaniel Good, J. Ben Schafer, Joseph A. Konstan, Al Borchers, Badrul Sarwar, Jon Herlocker, and John Riedl. ”Combining Collaborative Filtering with Personal Agents for Better Recommendations” *AAAI-99 Proceedings* (1999).
[21] Debnath, Souvik, Niloy Ganguly, and Pabitra Mitra. "Feature Weighting in Content Based Recommendation System Using Social Network Analysis." *Proceedings of the 17th International Conference on World Wide Web* (2008): 1041-042.
[22] Zhongqi Lu , Zhicheng Dou , Jianxun Lian, Xing Xie, and Qiang Yang. ”Content-Based Collaborative Filtering for News Topic Recommendation.” *Proceedings of the Twenty-Ninth AAAI Conference on Artificial Intelligence* (2015): 217-223.
[23] Burke, Robin. "Hybrid Web Recommender Systems." *Lecture Notes in Computer Science (including Subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)* 4321 (2007): 377-408.
[24] Ittoo, A.R., Y. Zhang, and J. Jiao. "A Text Mining-based Recommendation System for Customer Decision Making in Online Product Customization." *Management of Innovation and Technology, 2006 IEEE International Conference on* 1 (2006): 473-77.
[25] BeerViz, https://seekshreyas.github.io/beerviz/
[26] http://dataaspirant.com/2015/05/25/collaborative-filtering-recommendation-engine-implementation-in-python/
[27] https://towardsdatascience.com/how-did-we-build-book-recommender-systems-in-an-hour-part-2-k-nearest-neighbors-and-matrix-c04b3c2ef55c