Project Report

Z534 Search

Instructor: Xiaozhong Liu

Collaborative Filtering and Sentiment Analysis on Yelp Dataset

Group members: Sanket Pandilwar, Akshay Kowshik, Camilla Tan

Spring 2020

Github: https://github.iu.edu/spandilw/Collaborative-Filtering-and-Restaurants-Classification-based-on-Sentiment-Analysis-on-Yelp-dataset

Introduction

Yelp is one of the largest and commonly used apps to allow people to review any type of business in the world. People use Yelp in a lot of ways, such as seeing whether the business is good or bad based on other people's reviews from their previous experiences, finding out whether the business is suitable for the occasion, or discovering which restaurant will be their next favorite. Thanks to a large number of users, businesses, and reviews on Yelp, we are able to experiment with some interesting topics with our project.

It is essential for Yelp to find a better way to help users find restaurants they are interested in or to recommend restaurants they are more likely to go to, therefore, to predict the rating of a restaurant accurately is the key. Only by looking at the average ratings given by users is not sufficient to predict the restaurant's rating star, our hypothesis is that by integrating the sentiment analysis on the users' reviews for the restaurant, we can generate a better rating score for the restaurant.

In our project, we are interested in improving the recommender system with sentiment analysis, to predict restaurants' ratings based on users' reviews and to recommend restaurants for different users or based on other similar restaurants. We divided our project into three parts. Firstly, we experiment with different approaches in Collaborative Filtering (CF) to find out what is a better way to improve the recommender system. Secondly, we use sentiment analysis and different algorithms to train our models to do the prediction and analyze the better outcomes. Lastly, we tried to combine sentiment analysis with collaborative filtering with some experiments and literature reviews, to discover different approaches to improve CF with sentiment analysis.

Our Teamwork

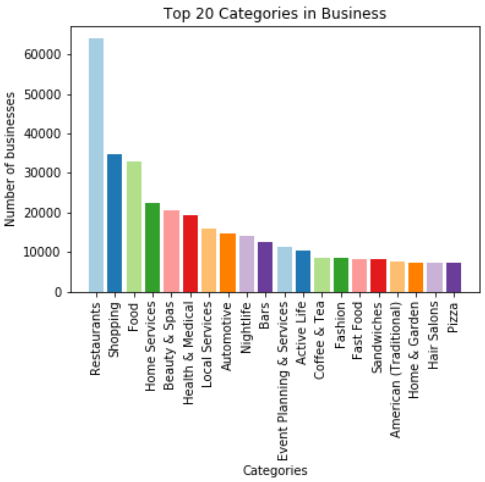
We worked together on the project with great teamwork on Zoom and Slack. We would have face-to-face discussions when we have problems and share opinions and look up all kinds of resources and share them with the group. To work more efficiently with the limited amount of time for the project, we divided our works into small parts:

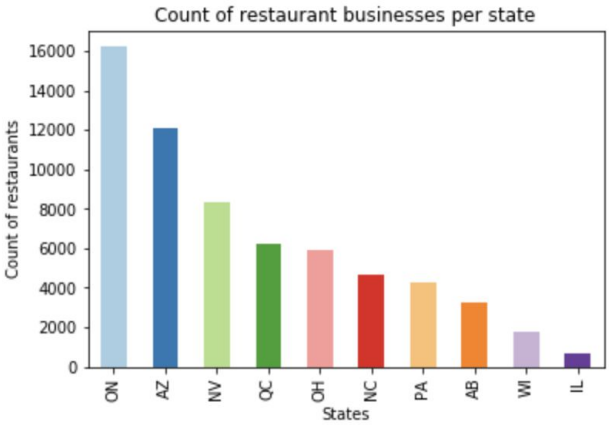
- Sanket: Prediction model, Sentiment Analysis and Data Visualization
- Akshay: Part of collaborative filtering and Part of classification of restaurants based on sentiment analysis
- Camilla: EDA, Collaborative Filtering in item-based, user-based and model-based, presentation and final documentation

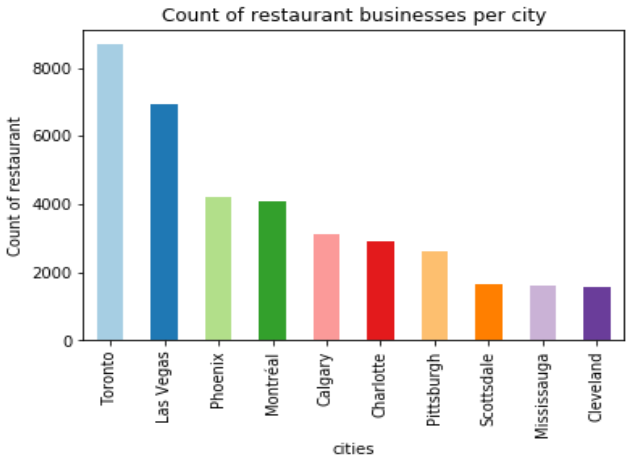
Data

The data we used are business.json, reviews.json, checkin.json, user.json, and tips.json from the Yelp Challenge open datasets. There are around 5.1M Reviews in Review.json, 210K, Businesses in Business.json, and 1.97M Users in Users.json. First of all, we want to take a look at the statistics of the Yelp data via conducting some data visualizations. Looking at the business.json, the restaurant has the biggest number in the business, and Toronto and Ontario have the greatest number of restaurants among city and state respectively. We

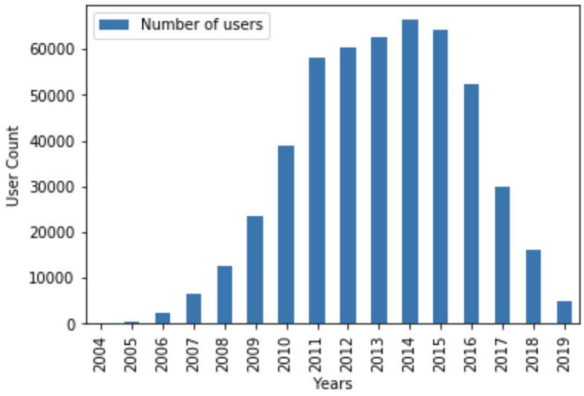
decided to use the restaurant data in Las Vegas since it has the most number of restaurants in the United States and most number of reviews overall.

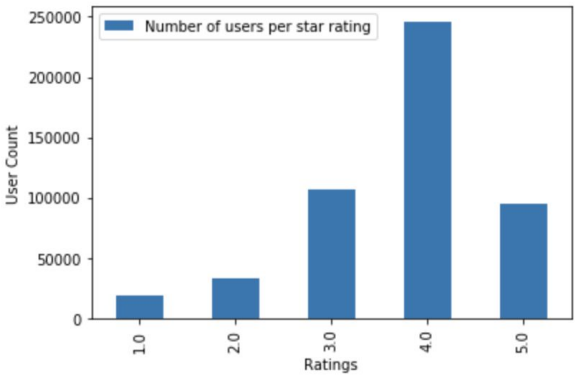




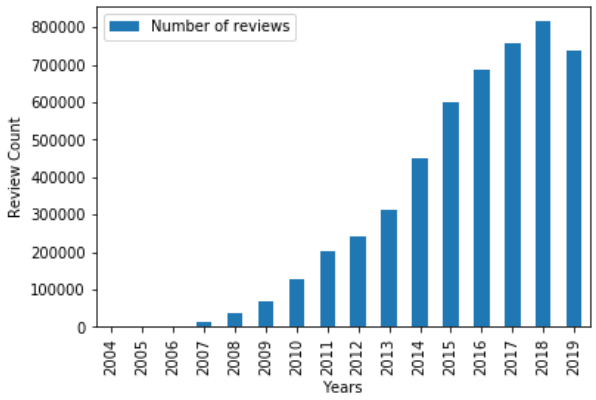


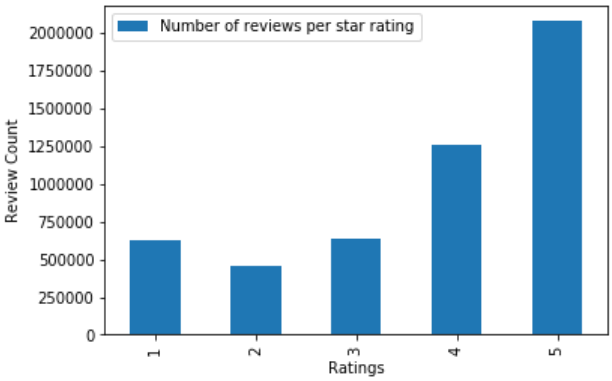
Next, we look at the Yelp user data, where we can see the number of users is increasing since it started, and significantly from 2010 to 2011; however, it decreased since 2014. The visualizations also show that from rating stars 1 to 5, rating 4 is the most popular one given by users that has the biggest portion of distribution among all levels, while star 1 and 2 have a relatively smaller count of all.





As for the overview of Yelp reviews, the number is continually increasing, and star 5 has the highest number of reviews, followed by star 4.





Data Preprocessing

- Category focus: Since we only considered Restaurant Businesses from Las Vegas City, we first extract data from all the files to get the reviews of restaurants in Las Vegas City from each user.
- Sparsity reduction: Not every user will give a lot of reviews to restaurants, so the matrix will be very sparse. In order to improve the situation, we picked users who have rated more than 100 reviews.
- Data split: For training and testing data, we divided into 80-20 split for part 1, and 70-30 split for part 2.
- Linguistic performance
    - **Lemmatization**, takes into consideration the morphological analysis of the words (Bitext, 2018). The key to this methodology is linguistics. To extract the proper lemma, it is necessary to look at the morphological analysis of each word. This requires having dictionaries for every language to provide that kind of analysis.
    - To improve the performance and accuracy of sentiment analysis, we used NLTK Python Library to conduct lemmatization to gather the words with the same root and words with the same meaning.

Methods

❖ User-based collaborative filtering

For user-based Collaborative filtering, we are trying to create a matrix where we look at the similarity of a pair of users. If user A rated an X score on restaurant A, while B also rated a similar score on the same restaurant, those two users might have a higher similarity score, and we can recommend restaurant B to user B where user A rated as a 5-star restaurant since B might also be interested.

	Restaurant A	Restaurant B	Restaurant C
User A	3	4	?
User B	?	4	5
User C	5	5	?

❖ Item-based collaborative filtering

Item-based collaborative filtering considers the similarity score between the restaurants and recommends similar restaurants to the user. For instance, if user A likes restaurant A, then we can recommend restaurant B to the user since restaurant B has a very high similarity score with restaurant A.

Restaurant X	Restaurant A	Restaurant B	Restaurant C
4	3	/	/
2	/	4	5
3	5	5	4

❖ Memory-based KNN

For both user-based and item-based CF we are implementing memory-based K nearest neighbor technique to find the similarity user/restaurant. KNN is a collaborative filtering algorithm, considering the mean ratings of each user where the nearest neighbor or the closest objects are more similar.

$$\hat{r}_{ui} = \mu_u + \frac{\sum_{v \in N_i^k(u)} \text{sim}(u, v) \cdot (r_{vi} - \mu_v)}{\sum_{v \in N_i^k(u)} \text{sim}(u, v)}$$

$$\hat{r}_{ui} = \mu_i + \frac{\sum_{j \in N_u^k(i)} \text{sim}(i, j) \cdot (r_{uj} - \mu_j)}{\sum_{j \in N_u^k(i)} \text{sim}(i, j)}$$

❖ Similarity metrics

In this part, we also used Scikit-learn to test different similarity metrics in our algorithm to find out the differences.

1. Pearson correlation coefficient similarity measures how highly correlated are two variables, where 1 indicates that the data objects are perfectly correlated,

0 means the opposite. Only common users (or items) are taken into account in the algorithm. So a rated res1, and we only take it if b also rated res1. if there are no common users or items, similarity will be 0 (and not -1). The Pearson correlation coefficient can be seen as a mean-centered cosine similarity.

$$\text{pearson\_sim}(u, v) = \frac{\sum_{i \in I_{uv}} (r_{ui} - \mu_u) \cdot (r_{vi} - \mu_v)}{\sqrt{\sum_{i \in I_{uv}} (r_{ui} - \mu_u)^2} \cdot \sqrt{\sum_{i \in I_{uv}} (r_{vi} - \mu_v)^2}}$$

$$\text{pearson\_sim}(i, j) = \frac{\sum_{u \in U_{ij}} (r_{ui} - \mu_i) \cdot (r_{uj} - \mu_j)}{\sqrt{\sum_{u \in U_{ij}} (r_{ui} - \mu_i)^2} \cdot \sqrt{\sum_{u \in U_{ij}} (r_{uj} - \mu_j)^2}}$$

2. Cosine similarity compares the space/angle between 2 vectors to find the closest two pairs, which also only considers common users (or restaurants). It is very similar to Pearson's correlation coefficient. r¯i is the vector of the user *u* and *i* rated respectively.

$$\text{cosine\_sim}(u, v) = \frac{\sum_{i \in I_{uv}} r_{ui} \cdot r_{vi}}{\sqrt{\sum_{i \in I_{uv}} r_{ui}^2} \cdot \sqrt{\sum_{i \in I_{uv}} r_{vi}^2}}$$

$$\text{cosine\_sim}(i, j) = \frac{\sum_{u \in U_{ij}} r_{ui} \cdot r_{uj}}{\sqrt{\sum_{u \in U_{ij}} r_{ui}^2} \cdot \sqrt{\sum_{u \in U_{ij}} r_{uj}^2}}$$

3. Mean Squared Difference similarity computes the Mean Squared Difference between all pairs of users (or restaurants).

$$\text{msd}(u, v) = \frac{1}{|I_{uv}|} \cdot \sum_{i \in I_{uv}} (r_{ui} - r_{vi})^2$$

$$\text{msd}(i, j) = \frac{1}{|U_{ij}|} \cdot \sum_{u \in U_{ij}} (r_{ui} - r_{uj})^2$$

❖ Model-based SVD

SVD is a matrix reduction technique that decreases the dimension of the matrix, to map each user and each restaurant respectively. Thus, it helps us better understand the relationship between users and items as they become directly comparable (Malaheb, 2016). The matrix can be very sparse since not every user does not give ratings to every restaurant, SVD can help us ignore empty cells in the matrix and only keep valuable ones. SVD can assist in finding the minimum RMSE by optimizing the space dimension of the datasets. We also used the Grid Search to find the optimized parameters by testing a number of parameters (Scikit Learn 0.22.2, 2019).

$$\min_{U,V,\Sigma} \sum_{ij \in A} \left(A_{ij} - [U\Sigma V^{\mathrm{T}}]_{ij}\right)^2$$

❖ Rating prediction based on sentiment analysis

In the second part, we are trying to predict the rating of a business based on sentiment analysis on users' reviews for the restaurants. Sentiment analysis, meaning that we need to find out the emotions of a user towards a restaurant by looking at the reviews. For example, "good" and "delicious" meaning positive, while "awful" and "bad" meaning negative. To calculate the sentiment score, we used polarity to determine the sentiment score. If the sentiment score was greater than zero then an initial polarity of 1 was given to that particular review or tip, which means it's a positive sentiment. If the sentiment score was less than zero, then an initial polarity of 0 was given to that particular review, which means it's a negative sentiment.

Polarity of Sentence = Sum of polarity of all the words in a sentence/the total number of words in the sentence

As we know different businesses have different numbers of reviews, so it would be better to obtain an overall sentiment score.

Net Positive Sentiment Score = Sum (Sentiment score for that business) / Total number of reviews

Now the overall idea is to use this score which indicates the sentiment as one of the key features in our prediction model. Since we need to predict how good the restaurant is, we are turning into a multi-level classification problem. Thus, we need to create a score metric and use them as labels for our classification models.

Average review stars	Sentiment class

>= 3.5	Positive sentiment, i.e restaurant is amazing
2.0 < r < 3.5	Neutral Sentiment, i.e restaurant is average
<= 2.0	Negative Sentiment, i.e restaurant is not so good

To test the performance of our models, we used the following algorithms to train our model: Viz., SVM, Decision Tree, Logistic Regression, Naive Bayes, KNN, Random Forest, Ada Boost, and Multi-layer Perceptron.

- **Collaborative Filtering with sentiment analysis**

There are a number of approaches we have been researching on. Leung et al (2006) used a rating inference approach to integrating collaborative filtering with sentiment analysis from users' reviews. The authors incorporate with Part-of-Speech (POS) tagging which is a helpful technique to extract more useful words such as adjectives and verbs from users' opinion, Negation Tagging which can identify the meaning of words that might be affected by words like "Not", and Feature Generalization to collect existing and valuable features from the reviews. Next, they create an opinion dictionary by putting different weights on the importance of the word a, this dictionary reflects the strength of opinion words with sentiment class c (positive/negative), with f as the frequency of the words occurs in that specific sentiment class. As a result, whole opinions are more important to decide on the strength of the review, and double weighting features and negation tagging are proven to increase accuracy.

$$OS(a_n, c) = \frac{F(a_n, c)}{\sum_{c_i \in C} F(a_n, c_i)}$$

Pappas & Popescu-Belis (2013) used MPQA polarity lexicon based on two labels (positive and negative) to classify users' comments. They utilized a sentiment-aware neighborhood model that integrates NN models with the results of sentiment analysis from comments mapping with rating values that the combination can be understood by the model. In the formula, D (u, i) means the neighborhood of the k similar items that the user already rated/commented, and r (u, j) is the mapping function that incorporates score from rating and comments. They compared three mapping functions: "RandSANN" 3-way rating (0/1/-1), rule-based (RB) sentiment classifier, and "polSANN" which generates polarity score (from 1 to -1) and add up the score as a sentence/comment as a whole.

$$\hat{r}_{ui} = b_{ui} + \sum_{j \in D_c^k(u;i)} d_{ij}(r'_{uj} - b_{uj})$$

Results & Evaluation

For the first part, we evaluated user-based CF by RMSE, MAE and MSE.

	Cosine similarity	Pearson similarity	Msd similarity
RMSE	1.4969	1.4309	1.4657
MSE	2.2407	2.0475	2.1482
MAE	1.1106	0.9740	1.0462

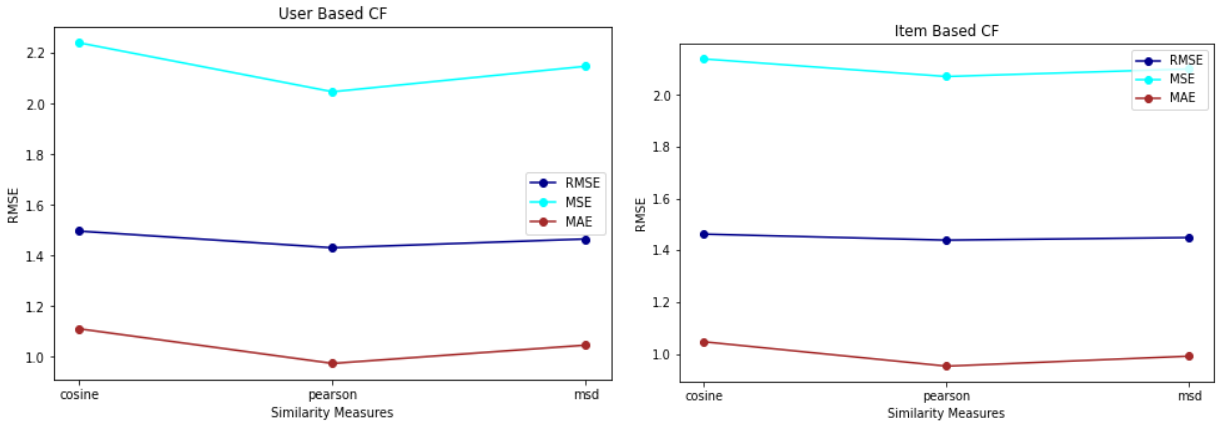
Through Grid Search on the optimized parameters, the best rmse score is 1.0008027509399238.

As for item-based CF, we are able to generate a list of similar restaurants based on the similarity score with the queried restaurant.

Queried restaurant	Loftti Cafe
Recommended restaurants	1: To See Roll 2: Brew Tea Bar 3: Cafe Summer 4: Zen Curry Express 5: Snow White Cafe 6: Is Sweet 7: Cafe 86 8: Dirt Dog - Las Vegas 9: Tea Space 10: Eis Cream Cafe

As for the comparison between user-based and item-based CF, there are some advantages and disadvantages for both of them. For user-based CF, firstly, it assumes the user will keep the same favorite tastes in the future, but it is likely that the user changes his/her favorite restaurants in reality. Secondly, it has a very sparse matrix since not every user will rate that many restaurants. It can also have a higher cost when finding K nearest neighbors when we have a large number of users. Lastly, it has a cold start problem for new users as they don't have any rating on file. For item-based CF, it can generate results recommendations for users even for new users, but it cannot capture users' interests changing overtime (Pinela, (2017).

We also compared the performance of different similarity metrics on user/item-based CF. We can see that Pearson correlation coefficient similarity shows a relatively better performance with the lowest error rate, and item-based CF has a slightly better score on evaluation.



At last, we compared the performance on the item-based CF between the item-based CF using rating and the CF using texts. Using only rating shows a lower accuracy on the results than using texts. For example, if we want to generate similar restaurants of Delhi Indian Cuisine, which is an Indian restaurant, we can see that using rating, there are some restaurants from other cuisines such as Chinese, Japanese and Mexican, while the results from using texts, we have mostly great Indian restaurants on the top of the list.

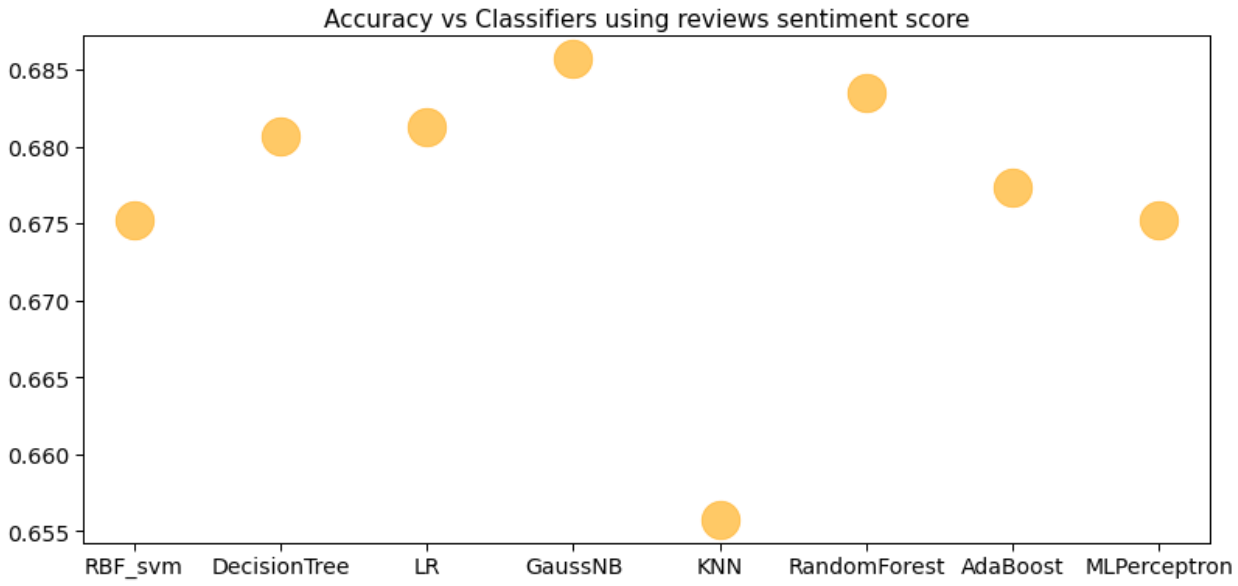
Restaurant	Delhi Indian Cuisine	
Recommended restaurants	Using rating	Using texts
	1: Big Wong Restaurant 2: Art of Flavors 3: **Mt Everest India's Cuisine** 4: **Mint Indian Bistro** 5: El Cachi Mexican Kitchen 6: Japanese Cuisine By Omae 7: SLO-Boy 8: Indian Bowl Cuisine 9: Nem Nuong Bistro	**1: Namaste Indian Cuisine** **2: Mt Everest India's Cuisine** **3: Angara Indian Spice Grill** 4: Mint Indian Bistro 5: Tamba 6: Taj Palace 7: Paradise India 8: India Oven Masala Bar & Grill 9: Mint Indian Bistro

For model-based SVD, we received the best rmse score at 0.9729 and the best parameters are 'n_epochs': 20, 'lr_all': 0.005, and 'reg_all': 0.2, where we tried parameters [5, 10, 20] for "n_epochs", [0.002, 0.005] for "lr_all" and [0.2, 0.4, 0.6] for "reg_all".

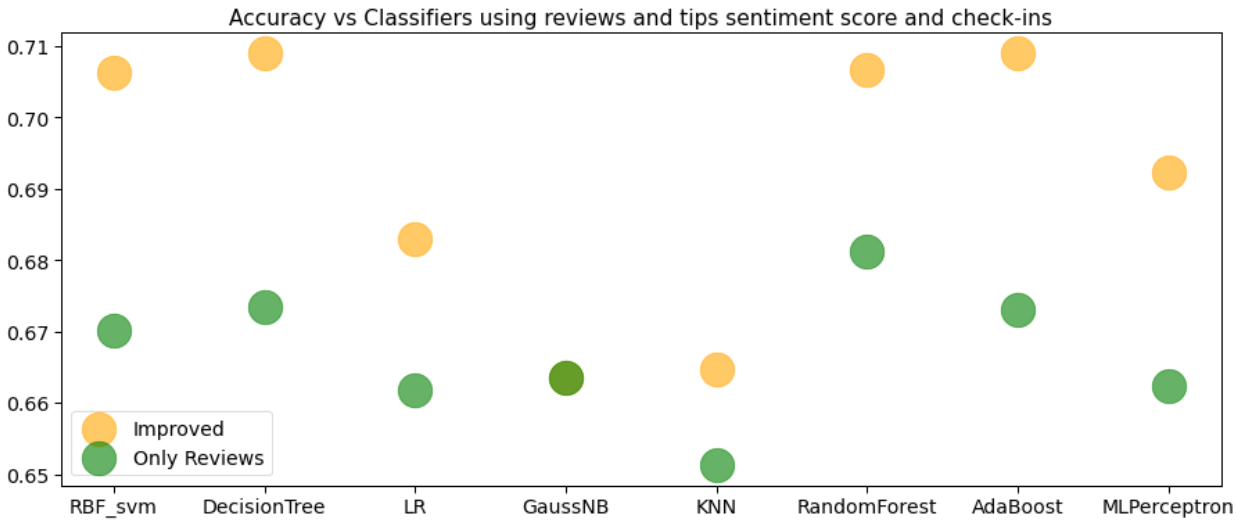
The Evaluation Metrics we used for Part 2 are Accuracy, Precision, Recall, and F1 Score.

	Precision	Recall	F1 Score	Accuracy
Training data	0.3682	0.7191	0.3462	0.6889
Testing data	0.5475	0.5690	0.5095	0.6807

Looking at the comparison of all the models, we can see GaussNB and Random Forest pose better performance among all in accuracy, while KNN has the lowest performance.



We also compared the accuracy between the data when we only used reviews as a feature and when we combined reviews, tips, and check-in. We can see most of the algorithms' accuracy performance is significantly improved except for the GaussNB because Naive Bayes considers all the features independently hence it's not of much use in this case.



Conclusion

In our project, we implemented collaborative filtering for recommender systems and rating prediction of restaurants based on user reviews and tips data with different models. We also researched different approaches to combine collaborative filtering with sentiment analysis to improve the recommender system. Based on our observation, we can see that using different datasets (ratings/reviews), data preprocessing (sparsity), classification models (random forest/decision tree), and CF techniques (item/user/model/knn) can make a huge difference in the evaluation results such as accuracy and precision/recall.

Limitation and future works

NLP being such a huge and blooming domain, it's always difficult to preprocess the text. The better the text is preprocessed and captured efficiently, the better are the results, so there are other good approaches that we might think about to implement and improve in the future. As for future works, we could look at the use of deep learning algorithms for NLP and it might lead to better accuracy. In addition, we could use more features like the availability of restaurant, location, categories, and some other features related to business to improve our prediction model. We could also try to use different NLP text preprocessing techniques and models such as stemming, semantic indexing, unigram, bigram and trigram models. We can also try to implement classification algorithms by using cross-validation techniques to find a better balance between bias and variance. As for future works of combining the collaborative filtering and sentiment analysis, we could try to use feature selection to choose better features such as location, cleanness, ambiance and customer service, to calculate the similarity score of the features and mapping the similarity score with users or restaurants.

References

Bitext. (2018). What is the difference between stemming and lemmatization? Retrieved from: https://blog.bitext.com/what-is-the-difference-between-stemming-and-lemmatization/

Karampiperis, P., Koukourikos, A., & Stoitsis, G. (2014). Collaborative filtering recommendation of educational content in social environments utilizing sentiment analysis techniques. In *Recommender Systems for Technology Enhanced Learning* (pp. 3-23). Springer, New York, NY.

Leung, C. W., Chan, S. C., & Chung, F. L. (2006, August). Integrating collaborative filtering and sentiment analysis: A rating inference approach. In *Proceedings of the ECAI 2006 workshop on recommender systems* (pp. 62-66).

Malaheb, M. (2016). Medium. Singular Value decomposition (SVD) in recommender systems for Non-math-statistics-programming wizards. Retrieved from: https://medium.com/@m_n_malaeb/singular-value-decomposition-svd-in-recommender-systems-for-non-math-statistics-programming-4a622de653e9

Mu, R., & Zeng, X. (2018). Collaborative filtering recommendation algorithm based on knowledge graph. Mathematical Problems in Engineering, 2018.

Pappas, N., & Popescu-Belis, A. (2013, July). Sentiment analysis of user comments for one-class collaborative filtering over ted talks. In *Proceedings of the 36th international ACM SIGIR conference on Research and development in information retrieval* (pp. 773-776).

Pinela, C. (2017). Recommender Systems — User-Based and Item-Based Collaborative Filtering. Retrieved from: https://medium.com/@cfpinela/recommender-systems-user-based-and-item-based-collaborative-filtering-5d5f375a127f

Scikit Learn 0.22.2. (2019). Retrieved from: https://scikit-learn.org/stable/modules/generated/sklearn.model_selection.GridSearchCV.html