Task 1: Yelp data based Restaurant Recommendation system

Chhavi Sharma[1]*,Prahasan Gadugu [2]*, Pranamya Vadlamani[3]*

Abstract
Recommendation systems attempt to predict the preference or rating that a user would give to an item. Knowledge discovery techniques can be applied to the problem of making personalized recommendations about items or information during a user's visit to a website. Collaborative Filtering algorithms give recommendations to a user based on the ratings of other users in the system. Traditional collaborative filtering algorithms face issues such as scalability, sparsity and cold start. In the proposed framework, prediction using item based collaborative filtering is combined with prediction using SVD models and a Content based recommendation system. The proposed solution will be scalable while addressing user cold start.

Keywords
Recommendation Systems — Collaborative Filtering — Memory based Collaborative Filtering —Singular Value Decomposition — SVD — TF-IDF — Cosine Similarity—Content Based Filtering — N-Grams — KNN — K-Fold — Cross Validation — Pipeline — Yelp — Lemmatization —

[1] *Masters in Data Science, School of Informatics, Computing, and Engineering, Indiana University, Bloomington, IN, USA*
[2] *Masters in Data Science, School of Informatics, Computing, and Engineering, Indiana University, Bloomington, IN, USA*
[3] *Masters in Computer Science, School of Informatics, Computing, and Engineering, Indiana University, Bloomington, IN, USA*
***Course Instructor**: Xiaozhong Liu
***Associate Instructor**: Zheng Gao

Contents

Introduction

Nowadays every company and individual can use a recommender system – not just customers buying things on Amazon, watching movies on Netflix, or looking for food nearby on Yelp. In fact, one fundamental driver of data science's skyrocketing popularity is the overwhelming amount of information available for anyone trying to make a good decision. The goal of this work is to construct a personalized recommender system for the Yelp dataset that can accurately predict a user's preference for a business. Motivated by the popularity among real-world recommender systems of collaborative filtering, and matrix factorization methods in particular, this work explores these methods to build the recommender system.

1. Data

The dataset used for this project is taken from Yelp. We randomly chose Charlotte city as our primary place to recommend the restaurants. At the outset, we calculated the number of reviews given by each user to the business by grouping the data based on user id and the review id and selected the users with more than 20 restaurant ratings. An overall of 20K samples were taken out of them, which are further divided into training and testing sets to build the models and evaluate our models.

1.1 Data Cleaning

Before modelling, we performed the following pre-processing steps on the data:

1. Lemmatization- Firstly, we grouped together all inflexions of a word in order to analyse them as a single item during class prediction.

2. Changed the entire corpus to lowercase in order to make our predictions case-insensitive by not differentiating between the same word in different cases.

3. Removed all stop words like prepositions, articles, conjunctions etc. since they do not add much value to our predictions and are fairly common to all reviews

4. Used Regex formatting to remove punctuation marks like full-stop, comma etc. since they are useless towards predicting the class label. However, we retained the exclamation mark since it adds value to the sentiment of the user.

5. Breaking the grammatical structure of the data in order to get a corpus of words that can be used as our feature set.

6. Removed null values encountered due to data cleaning(removal of stop words).

1.2 Splitting the data

For further model building and evaluation of the information, we divided our dataset into training and testing set in the ratio 80:20.

2. Evaluation Metrics

We used the following metrics to evaluate our model,

1. Root Mean Square Error (RMSE) calculated as follows,

$$RMSE = \sqrt{\frac{1}{N}\sum(x_i - \hat{x_i})^2}$$

where x_i and $\hat{x_i}$ are the true and predicted values respectively, N being the number of samples

2. Mean Absolute Error(MAE) given as:

$$\text{MAE} = \frac{\sum_{i=1}^{n} |y_i - x_i|}{n}$$

where y_i, x_i are the predicted and the true values respectively. n being the number of samples.

3. Mean absolute percentage error (MAPE)

$$\text{M} = \frac{1}{n}\sum_{t=1}^{n}\left|\frac{A_t - F_t}{A_t}\right|$$

where A_t, F_t are the actual and the estimated values respectively, n being the number of samples.

3. Collaborative Filtering

3.1 Problem Statement

We wanted to take advantage of the user base and the restaurant rating history that we have from the Yelp Data set to provide better recommendation models. Hence, we started with the Collaborative Filtering approach and implemented Memory-based and Model-Based approaches. Considering the former, Memory-based Collaborative filtering as the baseline, we moved onto improving our results.

3.2 Data Pre-processing

The first step in the Data pre-processing is to generate a user-item matrix with data points as ratings given by each user to the business. For this purpose, we integer encoded both the business_ ids and user _ ids and formed the following matrix:

	user_id_key	business_id_key	review_stars
0	0	0	5.0
1	1	0	4.0

Figure 1. user–item integer encoded

User-item ratings matrix is created in the following fashion,

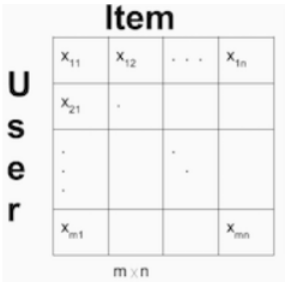

Figure 2. user–item ratings matrix

Using the above matrix, we implemented item-item and user-item similarities as mentioned in the following methods.

3.3 Memory based Collaborative Filtering

The memory-based approach uses user rating data to compute the similarity between users or items. Typically there are two varieties known as item-item collaborative filtering(users who liked this restaurant also like following..) and user-item collaborative filtering(users who rated similar to you also like following...)

3.3.1 Experiment Design

Using this user-item matrix, we can calculate the similarities between two items, and between two users.

Between two items, we are looking to find how their ratings were similar. For user-user similarity, we look at all the items that were commonly rated by the two users and how much their ratings were similar.

By doing so, we obtain similarity metrics for item-item and user-user, leading to item-item collaborative filtering and user-item collaborative filtering, respectively.

Item – Item Collaborative Filtering: Item-Item collaborative filtering will measure similarity by observing all users who have rated both restaurants.

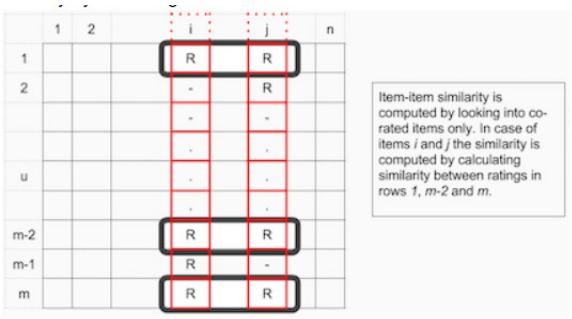

Figure 3. Item – Item Collaborative Filtering

User-Item collaborative filtering: User-Item collaborative filtering will measure similarity by observing all items that are rated by both users.

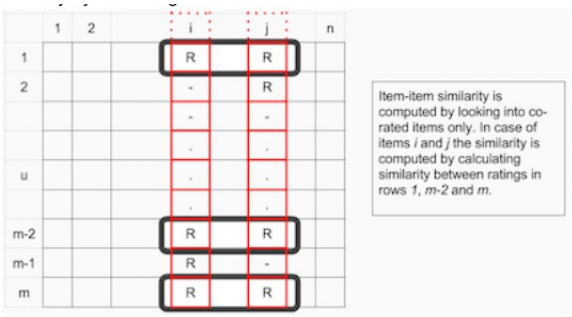

Figure 4. User-Item collaborative filtering

Cosine Similarity: We used cosine similarity to calculate the similarity. We consider the ratings as vectors in n- dimensional space and calculate the angle between these vectors to determine the similarity. Cosine similarity for users a and m are calculated using the following formula:

$$s_u^{cos}(u_k, u_a) = \frac{u_k \cdot u_a}{\|u_k\| \, \|u_a\|} = \frac{\sum x_{k,m} x_{a,m}}{\sqrt{\sum x_{k,m}^2 \sum x_{a,m}^2}}$$

To calculate similarity between items m and b we used formula:

$$s_u^{cos}(i_m, i_b) = \frac{i_m \cdot i_b}{\|i_m\| \, \|i_b\|} = \frac{\sum x_{a,m} x_{a,b}}{\sqrt{\sum x_{a,m}^2 \sum x_{a,b}^2}}$$

The output of cosine distance will range between 0 and 1, where a value closer to 1 means a higher similarity.

3.3.2 Results

As shown above, the results still have room for improvement. After implementing the memory based model, the disadvantages we noted are explained in the next section.

Method \ Metric	RMSE	MAE	MAPE
user-item based collaborative filtering	3.73	0.044	96.52
item-item based collaborative filtering	3.79	0.039	98.23

Figure 5. Memory Based model results

3.3.3 Disadvantages with Memory Based Method

We observed that our dataframe is very sparse (98.9 %), as the overlap between users and restaurants is not huge in Charlotte. Also, there is no defined process to reduce the sparsity and increase the scalability of the system. Memory-based system is not an efficient way to deal with Cold Start problem since given a user without sufficient history of rating, it cannot be taken into the matrix. Hence, we tried to implement the model-based approach.

3.4 Model based Collaborative Filtering

In model-based system, we handle high sparsity through dimensionality reduction and latent variable decomposition, thereby making it a more scalable approach. Moreover, we felt that underlying the user's ratings, there are latent features which need to be learnt by the model in order to recommend better.

3.4.1 Experiment Design

We used matrix factorization, which provides the latent vectors and allows filling in the sparse, original matrix: it predicts unknown ratings by taking the dot product of the latent features of users or items. For implementing matrix factorization, we used singular value decomposition (SVD). SVD is solved as follows:

$$X = U * S * V^{(T)}$$

where X is the prediction matrix, U represents the feature vectors corresponding to users in the hidden feature space, V represents the feature vectors corresponding to items in the hidden feature space, The matrix X will be factorized into U,S and V. Finally, we make our predictions by taking the dot product as shown in the formula.

The S matrix: The diagnol matrix S (K diagnol elements) has the latent vectors(optimal K value) which needs to be identified. To know the number of latent vectors needed, we have to find an approach that minimizes the error.

Determining K value: Since it is unknown how many latent (singular) vectors we should use for the model, we decided to used 5 fold cross validation to choose the optimal K. We plotted the RMSE, MAPE and MAE scores train and test matrices for different values of K. The plots are as follows,

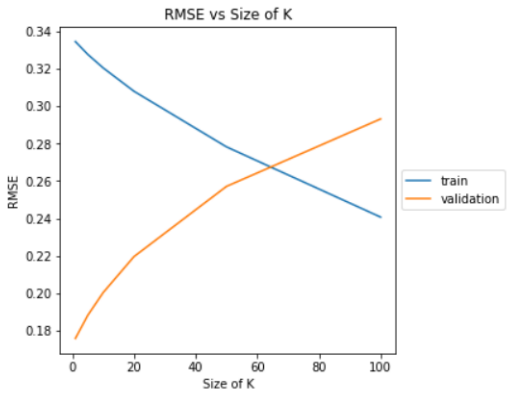

Figure 6. RMSE vs Size of K

As per the plots we can infer that the validation error increases with an increase in K value whereas the training error decreases with an increase in K value. This may seem that the model is overfitting the data. Hence an optimal K value is somewhere between 40 to 60 since the trend sharply changes(as per the RMSE plot).

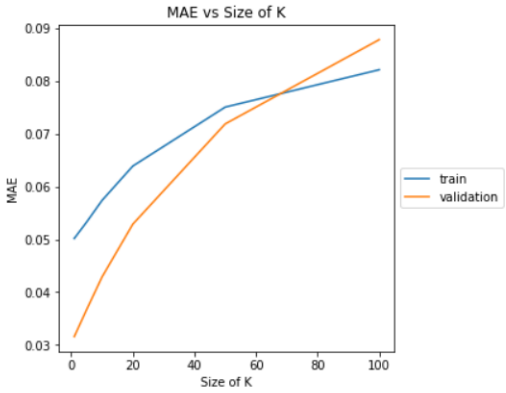

Figure 7. MAE vs Size of K

MAE increases with an increase in K for train as well as test values.

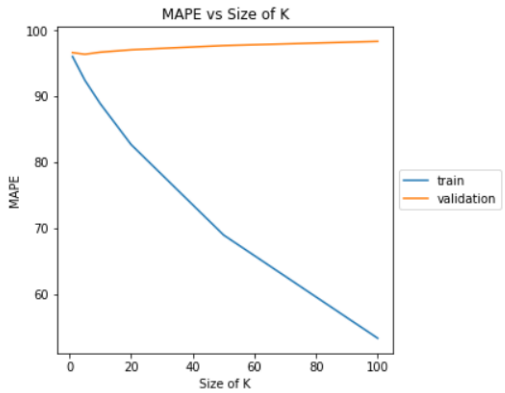

Figure 8. MAPE vs Size of K

MAPE score is large due to huge variations in the train and test values.

Hence, RMSE is a valid metric to analyse the performance based on which we choose the K value.

Predictions vs K value: We need to accommodate most of the predictions, hence when we plotted a graph for the K value and the predictions, we got the following curve.

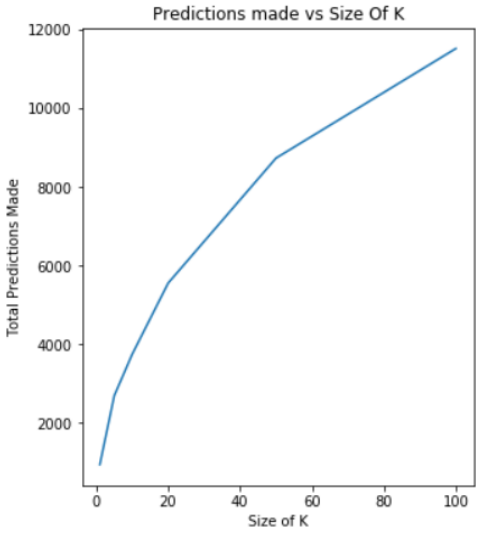

Figure 9. MAPE vs Size of K

Comparing this plot with the previous one, we realized that actual reason behind the erro ebing larger with an increased k is that the model is able to make a greater number of predictions with a larger k. Thus, the overall error is larger. Furthermore, the difference in RMSE is only about 0.15, so even at the ”worst” RMSE at k=10, the model-based CF is performing much better than the memory-based CF and the baseline model. We chose K = 80.

3.4.2 Results

The SVD Model-Based Collaborative filtering results are as follows,

1. RMSE = 0.29
2. MAE = 0.09
3. MAPE = 97.32

4. Content Based Filtering

4.1 Problem Statement

In addition to the user rating history, if we can use the review data combined separately for both users and businesses, it would be useful and will provide an additional edge to counter the cold start problem. So, if there is no user profile, we can just take the text information related to his/her liking and recommend the restaurants.

4.2 Data Preprocessing

We grouped the already cleaned reviews based on user_ id named as user_ text _ charlotte and also grouped based on the business_ id and named it as biz_ text _ charlotte (Figure 10).

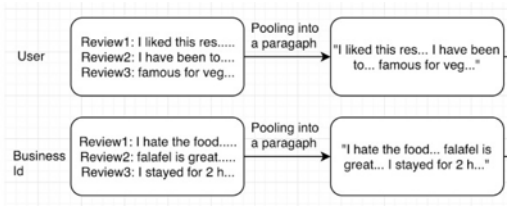

Figure 10. user and restaurant text pooling

4.2.1 Feature Extraction using TFIDF

Tf-Idf (Term Frequency Inverse Document Frequency) It is a text mining technique used to categorize documents. It emphasizes on words that occur frequently in a particular review, while at the same time de-emphasizes words that occur frequently in every review. This method will help us concentrate on portions which might be less frequent but have more significance for the overall polarity of the review. We have extracted top 1000 TF-IDF features for the user and restaurant pool system as shown below,

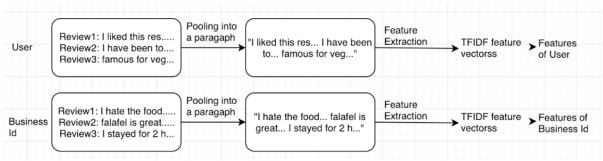

Figure 11. Feature extraction

4.3 Experiment Design and Results

Now that user and business matrices have been constructed, we also build a matrix, say M, of ratings given by each user to each business, with rows as user_ id and columns as business_ id.

Next, this matrix M is factorized into two matrices P and Q (feature vector matrix of user_ id and business_ id respectively).

The objective function that we are attempting to minimize here is

$$\min_{P\in\mathcal{R}^{m\times k},\in\mathcal{R}^{k\times n}} \sum_{(i,j)\in K}(M_{i,j} - (PQ)_{i,j})^2 + \lambda(\|P\|^2 + \|Q\|^2)$$

Figure 12. Objective Function

We minimize this function over certain number of iterations to make sure that the user and business vectors that we created relates to the rating matrix.

4.3.1 Building the recommendations

Once the objective function is minimized, we get the updated P and Q matrices with more relativity to the M matrix, take the text information from the user and recommend. For example, user gives the text, "I am interested in falafel", he/she gets the following recommendations,

business_id	Rating
2V-3ZFt2tLzU_0V6ZI183w	0.378872
qW-JrBmarQFijWWiN0xoTg	0.271636
wWKdUnwXDS91H09D_zC1SQ	0.268209
KLluzHXsTV0zCw48EJTEEg	0.250344
wrs4Zbz17q_G49AIRxRPVw	0.242592

Figure 13. Recommendations

The recommendations are built first by transforming the given text into TF-IDF vectors and then getting the dot product of the same with the updated Q matrix. Hence, it shows a rating as well.

4.3.2 Results

On evaluating Content Based Filtering approach, we achieved the following scores:

1. RMSE = 0.34
2. MAE = 0.28
3. MAPE = 42.52

5. Summary: Comparison of different filtering approaches

Last but not the least, we decided to compare the results obtained by all filtering approaching in order to decide which approach works best for our recommendation system. (Figure 14) summarizes our results:

Metric / Method	RMSE	MAE	MAPE
user-item based collaborative filtering	3.73	0.044	96.52
item-item based collaborative filtering	3.79	0.039	98.23
Model Based Collaborative Filtering	0.29	0.09	97.32
Content Based Filtering	0.34	0.28	42.52

Figure 14. Comparison of Filtering Approaches

On comparing the RMSE values, we can conclude that Model-based collaborative filtering gives us the best recommendation results. In order to resolve the Cold Start problem, we could use Content based filtering which also gives us a decent RMSE of 0.34.

6. Conclusion and Future Scope

Between all of the models tested, Model -based collaborative filtering (SVD) had the best score when using the RMSE metric (Content: 0.34; memory CF(User-item): 3.73; memory CF(item-item): 3.79; model-based CF (SVD): 0.29). The drawback to SVD is that it only makes predictions for 10% of users, meaning that an ensemble method is required to make predictions using an SVD model. Another drawback to our SVD model is that it just makes a prediction; there are no coefficients, which makes it difficult to interpret the features.

If given more time, we would revisit the cosine similarity functions. It seems that these functions failed because there are such a large amount of users. If we were able to filter to a subset of the most important users for any given business, we would be able to reduce the size of this matrix and make a more accurate similarity model as a result. However, this approach would be limited to only those users who are active in Yelp.

Lastly, another avenue for improvement would be to incorporate the regularized user bias and restaurant bias into our collaborative filtering models so that it increases the interpretability and the number of predictable users. Such hybrid approach would also address the cold-start problem that is currently being handled by Content filtering alone.

Contributions

Chhavi Sharma: Memory-based Collaborative filtering

Prahasan Gadugu: Model-based Collaborative filtering

Pranamya Vadlamani: Content-based filtering

References

1. https://towardsdatascience.com/evaluation-metrics-for-recommender-systems-df56c6611093
2. https://www.analyticsvidhya.com/datahack-summit-2019/schedule/hack-session-content-based-recommender-system-using-transfer-learning/
3. https://towardsdatascience.com/various-implementations-of-collaborative-filtering-100385c6dfe0
4. https://towardsdatascience.com/intro-to-recommender-system-collaborative-filtering-64a238194a26
5. https://medium.com/@sumith.gannarapu/restaurant-recommendation-system-b52911d1ed0b