SCHOOL OF
HUMAN SCIENCES
& TECHNOLOGY

2020

Recommendation Engines – Group Assignment

PRODUCT RECOMMENDATION | DIGITAL MUSIC

GROUP E | AMOS M., LI X., PABLO B., SARANG Z., VANIA C.

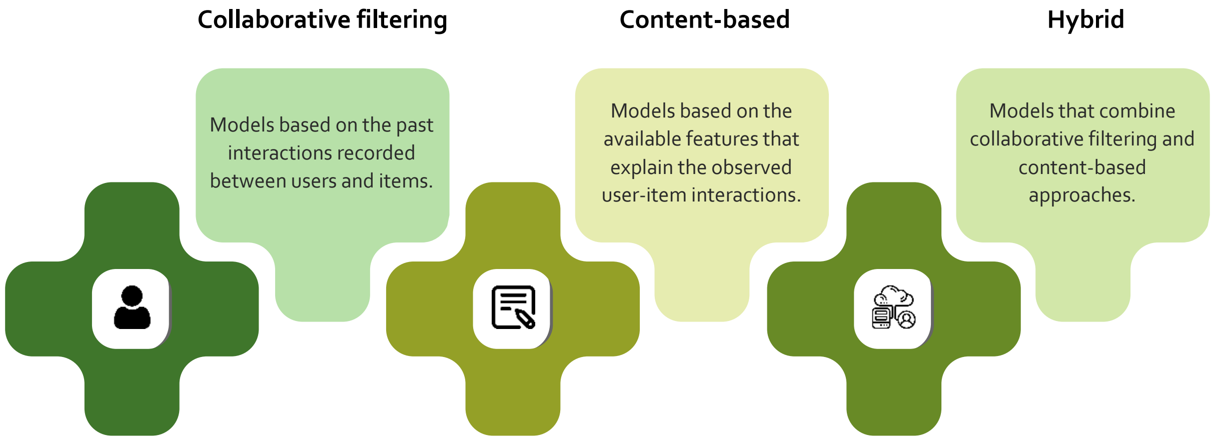
Contents

1. Introduction

Recommender systems are software tools and techniques aimed at providing relevant suggestions of products to a user. With the increase of e-business services, recommender systems become very important since they are a valuable tool to deal with the high amount of item choices presented to users, being a differentiation factor that generates an important amount of income for companies.

Using a dataset with Amazon user reviews (explicit numerical rating and contextual comments) on **Digital Music,** released by UC San Diego[1], the objective of this project is to create three types of recommender systems[2]:

a. a **collaborative filtering**, based on the user ratings;
a. a **content-based**, leveraging the textual content associated to the reviews; and
b. a **hybrid** approach, that combines collaborative filtering and content-based recommender systems.


Collaborative filtering
Models based on the past interactions recorded between users and items.
Content-based
Models based on the available features that explain the observed user-item interactions.
Hybrid
Models that combine collaborative filtering and content-based approaches.

This project was developed in Python and a notebook is delivered together with this report.

[1] Source: http://jmcauley.ucsd.edu/data/amazon/
[2] Falk, K. (2019). *Practical Recommender Systems* (1st ed.). Manning Publications; Rocca, B. (2019, June 12). *Introduction to recommender systems - Towards Data Science*. Medium. https://towardsdatascience.com/introduction-to-recommender-systems-6c66cf15ada

2. Analysis and Results

a. Analyzing the data

The Digital Music dataset has a total of 64,706 reviews made by customers on music albums that were purchased.

To further understand the dataset, a plot with the ratings distribution was created. With this, it was possible to conclude that the rating scale goes from 1 to 5 and that the distribution of the ratings is biased to high rates with 55% of reviewers giving 5 points, and 91.1% of the reviews rating the albums 3 or above.

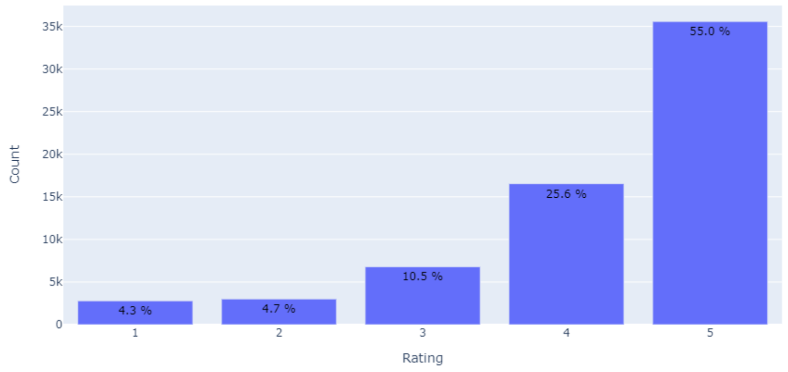


Fig1: Ratings Distribution Plot

Moreover, using a plot on the number of ratings received per alum, it was verified that most of the albums received less than 10 ratings and that each album has at least 5 overlapping customers.

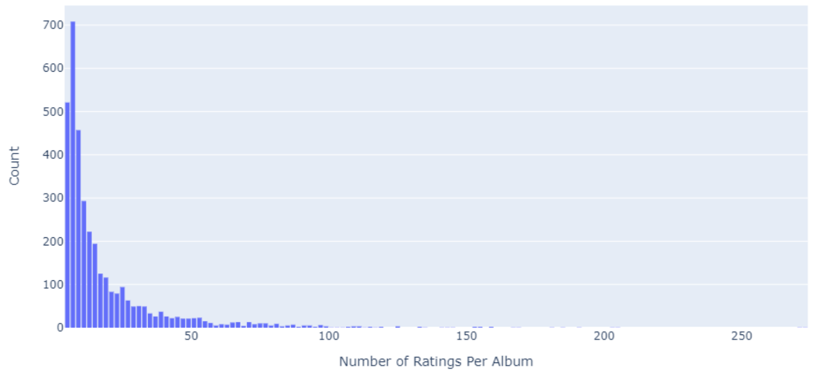


Fig2: Ratings Distribution Plot

Regarding users, most of them rated at least 7 albums and all users have rated at least 5 albums, which should satisfy the minimum number of ratings per user.

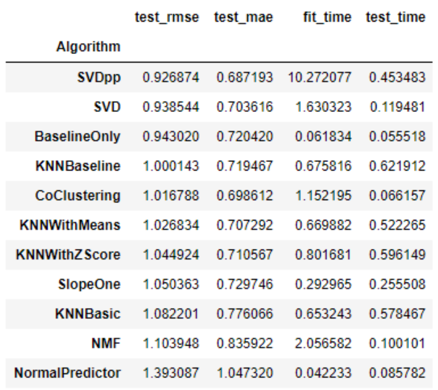
The dataset selected contains 5.541 unique users and 3.568 unique albums.

b. Collaborative Filtering (CF) Recommender System

Train / Test Split: The train/test split leverages review time data, since the goal is to predict users' most recent interests. The dataset has been sorted by review time from least recent to most recent, for each user, then the first 80% of his/her ratings were selected as the training data and the remaining 20% were used as testing data to simulate an 'offline evaluation' process. RMSE was used as the evaluation metric. RMSE was used as the evaluation metric.

System development steps:

Step 1 - Benchmark Performance of various CF algorithms: using the Surprise library, 11 algorithms were evaluated and compared to each other using the RMSE and MAE as evaluation metrics.

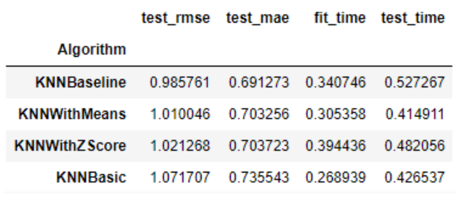


	test_rmse	test_mae	fit_time	test_time
Algorithm				
SVDpp	0.926874	0.687193	10.272077	0.453483
SVD	0.938544	0.703616	1.630323	0.119481
BaselineOnly	0.943020	0.720420	0.061834	0.055518
KNNBaseline	1.000143	0.719467	0.675816	0.621912
CoClustering	1.016788	0.698612	1.152195	0.066157
KNNWithMeans	1.026834	0.707292	0.669882	0.522265
KNNWithZScore	1.044924	0.710567	0.801681	0.596149
SlopeOne	1.050363	0.729746	0.292965	0.255508
KNNBasic	1.082201	0.776066	0.653243	0.578467
NMF	1.103948	0.835922	2.056582	0.100101
NormalPredictor	1.393087	1.047320	0.042233	0.085782

Table 1: Evaluation Results of various Algorithms

Based on the initial benchmark, matrix factorization algorithms provide the best results, and the baseline model provides very good performance too.

Step 2 – Benchmarking k-NN algorithms on similarities between Items: During the first step the similarities were user-based. During this step item similarities are considered to see if the performance of k-NN models will change.



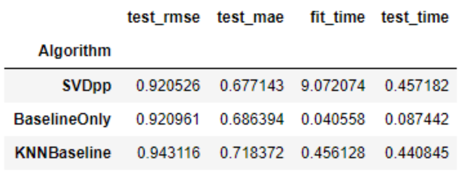
	test_rmse	test_mae	fit_time	test_time
Algorithm				
KNNBaseline	0.985761	0.691273	0.340746	0.527267
KNNWithMeans	1.010046	0.703256	0.305358	0.414911
KNNWithZScore	1.021268	0.703723	0.394436	0.482056
KNNBasic	1.071707	0.735543	0.268939	0.426537

Table 2: Evaluation Results of k-NN Inspired Algorithms

Based on the results, Item-based approach performs better. Among the KNN algorithms, KNNBaseline seems to be the best.

Step 3 – Model Hyperparameter Tuning: Based on the evaluation results in the first two steps, the BaselineOnly, KNNBaseline and SVDpp models were further tuned through a grid search.

Step 4 – Compare BaselineOnly, SVDpp, and kNN Baseline algorithms: As the best performing models, the algorithms were now compared considering the hyperparameters.



Algorithm	test_rmse	test_mae	fit_time	test_time
SVDpp	0.920526	0.677143	9.072074	0.457182
BaselineOnly	0.920961	0.686394	0.040558	0.087442
KNNBaseline	0.943116	0.718372	0.456128	0.440845

Table 3: Evaluation Results of BaselineOnly, SVDpp, and kNN Baseline Algorithms

Based on the results, the **SVDpp (RMSE = 0.921)** and **BaselineOnly (RMSE = 0.921)** algorithms provide very similar performance but the latter is considerably faster therefore it was selected as the model to be used for testing and the development of the CF based recommender system.

Step 5 – Model testing and prediction: On the final step, the BaselineOnly model was used on the test dataset to evaluate the performance of the model, which scored an **RMSE of 0.921**.

c. Content-based Recommender System

Two strategies were implemented to compare their performances:

1. A topic modelling approach given the number of words in the dataset
2. TF-IDF approach

Topic Modelling

Step 1 – Feature extraction:

1.1 Filtering data

Due to the lack of metadata the first step was to consider the positive reviews (rated at 3, 4 or 5) as the descriptions of the items. The positive reviews were grouped by the code of the album ("asin"), resulting in an input dataset having "asin" as the unique key, and all the reviews joint together in another column.

1.2 Text processing

Since the reviews' text contains a lot of irrelevant information, some text pre-processing was performed, removing undesirable marks, punctuations and stop words (additional words such as "song" or "album" were considered stop words). Words were also converted to their base form, using lemmatization (e.g.: 'walk', 'walked', 'walks' or 'walking' are converted to the base form – lemma -, 'walk').

Step 2 – Build item profile using Genism:

Genism is one of the most popular libraries to perform topic modelling. Using the LSI (Latent Semantic Indexing, similar to LDA) algorithm, it is easy to resent text as a vector and then find similar texts.

2.1 TF-IDF transformation
Suggested by Genism documentation, the corpus is transformed from a bag-of-words into a TF-IDF weighted representation.

2.2 Build LSI model and similarity Matrix
Each topic is a combination of keywords like the following:

(1, '0.356*"rap" + 0.255*"rapper" + 0.210*"beat" + 0.180*"hip" + 0.179*"hop" + 0.148*"rhyme" + 0.147*"game" + 0.122*"cube" + 0.114*"guest" + -0.108*"blue"'),

(2, '0.431*"funk" + 0.431*"jazz" + 0.166*"soul" + 0.138*"funky" + -0.138*"elton" + -0.131*"country" + 0.129*"groove" + 0.112*"blue" + 0.105*"stevie" + 0.101*"amp"'),

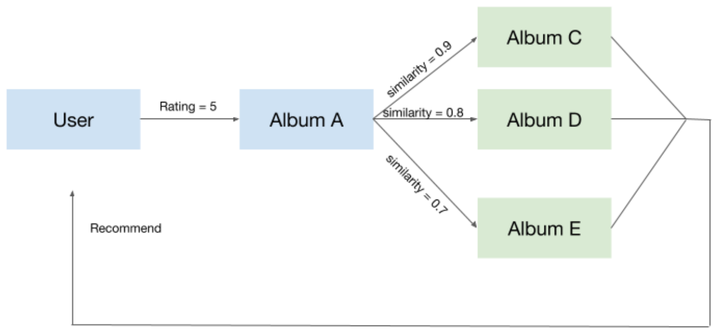
Next step is to project new documents onto the LSI model and find the most similar ones.

Step 3 – Filter recommendations:

3.1 Item-item approach

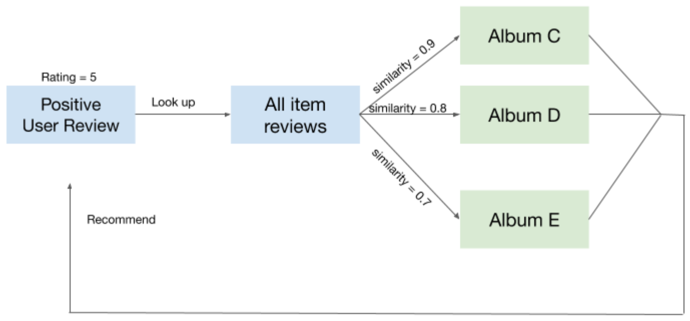
Following the approach in Practical Recommender Systems, Chapter 10, section 10.9.4, User profiles can simply be the items that user liked, in this case, albums with rating = 5. For example, Album A review is used to search in the similarity matrix for the most relevant albums C, D and E.

These albums will be recommended to the users. The whole flow is shown below:



3.2 User-item approach

The user profile in this approach is built by aggregating the review texts of albums each user rated 5. The aggregated text is then used to search in the similarity matrix which albums have the most similar reviews.



Step 4 – Evaluation:

The rating column was utilized to measure the RMSE. Rating of 5 was assigned to the top 5 items based on cosine scoring and then it was compared to the ratings provided by all users who had the item that the cosine score was derived from.

E.g. square root (Σ(Original rating - 5)**2)/n)

Item-item approach: RMSE = 1.64

User-item approach: RMSE = 2.71

Both approaches do not perform well with this dataset, however, item-item outperforms the user-item method.

<u>TF-IDF</u>

The same text preprocessing steps as above are implemented.

Using TfidfVectorizer, the review texts are transformed into a weighted scores representation, with one row per album and as many columns as there are different words in the dataset was created. Cosine similarities was used to iterate over the albums in the dataset and find the most similar ones. In this way, it was possible to obtain, for each album, the most related ones, and their similarity score.

The RMSE following this approach is 2.088.

d. Hybrid Recommender System

Mixed hybrid recommender system is chosen to be the solution to suggest albums to users. The business rules guiding recommendation are:

1. Primary recommendation list is generated from Collaborative Filtering method, I.e. 'Users with similar interest also listened to'. This list should be displayed at the most prominent position, because as we have seen it has better performance over Content-based method. Collaborative Filtering also has an advantage in serendipity, which helps users expand their interests.
2. Secondary recommendation list is generated from the Content Based method, I.e. 'Albums similar to your taste'. The main advantage is leveraging the rich review text to target specific user interest.
3. Non-personalized recommendation list: 'Trending Albums'. This list is generated by ranking the highest rated albums in the most recent 30 days. Having this list can help promote new albums, as well as expand user interest.

3. Conclusions

After the results presented in the previous section, the following conclusions can be drafted:

- KNN algorithm is item-item based because there are more unique items than users.
- Baseline algorithm and SVD++ algorithm have very strong performance in the offline test environment.
- Baseline algorithm is much faster to train, tune and test compared to SVD++. When the performance is similar, it could be a better choice to put into production.
- The quality of a content-based algorithm very largely depends on whether meaningful features can be extracted. In this project, review texts and review summary are subjective, therefore it is difficult to say the items are correctly represented. To improve the performance of Content-based recommender system, we need to enrich our data with metadata fields such as genres.

References

1. Falk, K., 2019. *Practical Recommender Systems*. Shelter Island, NY: Manning.
2. Radimrehurek.com. 2020. *Gensim: Topic Modelling For Humans*. [online] Available at: <https://radimrehurek.com/gensim/auto_examples/index.html#documentation> [Accessed 23 November 2020].