Anime Recommendation System

Jingwen Yan, Jingyi Sun, Yi Zhu

April,19 2020

1 Introduction

In the era of increasing information, most popular internet products people used today are powered by recommendation systems. Netflix, Youtubethe,Amazon and ebay rely on recommendar systems to fliter millions of products to make personalized recommendations to their users. Besides those internet companies, users need recommendation system as well. Because there are too many information and too many options today, people all needs somethings to help them to make decision. In this project, in the aspect of online entertainment service, we build a recommendation system using animes data to help users select the anime they actually want to see.

2 Data Preparation

We used the Anime Recommendations Dataset from kaggle. The data contains two files: Animes.csv and Rating.csv. Here is the dataset link: `https://www.kaggle.com/CooperUnion/anime-recommendations-database`

1) Dropped observations with 'rating'=-1 because they are meaningless.
2) Combined two datasets (Animes.csv and Rating.csv)by merging them via 'anime_id'.
3) Combined two datasets (Animes.csv and Rating.csv)by merging them via 'anime_id'.
4) Created the sparse 'uer_anime_rating' matrix through the original data.
5) Transformed object type ('anime_id', 'genre', 'type', 'type' and 'episodes') into categories type.

2.1 Data description

Box plot is used to show the data. As the box plot is not affected by outliers, it can show the distribution of data more accurately and steadily. This plot summarizes the ratings of various types of animes. As we can see from the

figure1, the type of music has the lowest overall rating, whereas TV has the highest overall rating. It may indicate that compared to the

TV-type animes, the music-type animes are not very acceptable to the audience. Also, there are a lot of outliers in the type of TV, OVA and Special, which provides a reference for our subsequent data pre-processing.

As TV is the top type, we also did a bar diagram to see the top 20 genres in it. In figure 2, x-axis represents rating frequency and the y-axis represents genre. Comedy is the theme which is the most rated.

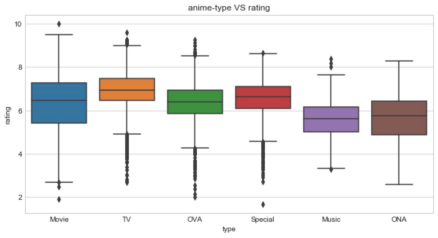


Figure 1: Box plot

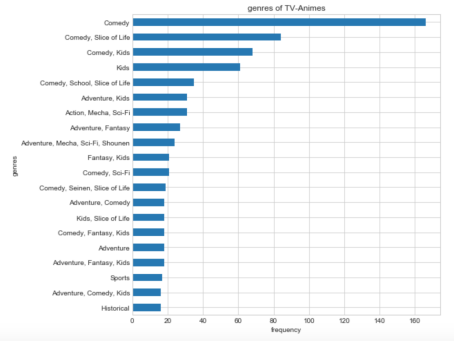


Figure 2: Bar diagram

2.2 Principal Component Analysis (PCA)

Principal component analysis (PCA) is a technique for reducing the dimensionality of such datasets, increasing interpretability but at the same time minimizing information loss. In this dataset, component is chosen as 3 to convert original variables to a new set of variables.

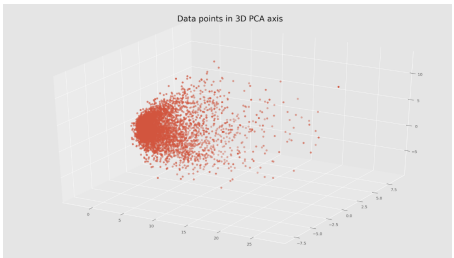


Figure 3: PCA

Figure 4: Word Cloud

2.3 Word Cloud

To better recommend anime to users, it is also important to find the popular trends among other users due to humans' conformity. Word Cloud is a data visualization technique used for representing text data which is very interesting and visual. We extracted the column "genre" and transformed it into .txt file. Users like Sci-Fi most, then is Slice of Life, Adventure, Comedy and so on.

2.4 K-means

[h] To group data, elbow method could be used to find the optimal K. From Figure 4, the best K is 3 .Thus, the target (rating_x) can be divided into three categories (dislike, medium and like) in the LightGBM.

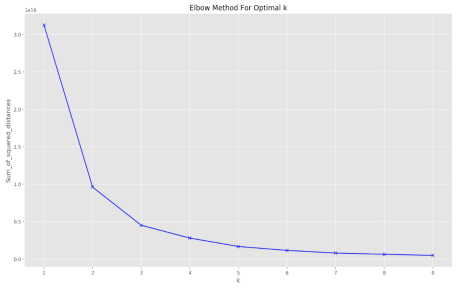


Figure 5: K-means

3 Modeling

In this part, several models are applied, including Memory-Based Collaborative Filtering, Model-Based Collaborative Filtering and LightGBM.

The advantage of Memory-based CF is that it is easy to explain the results.On the other hand,Model -based CF can deal with sparse data with dimensionality reduction. LightGBM has serveral advantages as following: (1)faster training efficiency (2)higher accuracy (3)capable of processing large data (4)low memory usage

3.1 Memory-Based Collaborative Filtering

Basically,Memory-Based CF can be divided into two main sections:User-Item Collaborative Filtering("Users who are similar to you also liked . . . ") and Item-Item Collaborative Filtering("Users who liked this item also liked. . . ").

The steps of Memory-Based CF are as follow:

1) Two user-item rating matrices are created from the original data ,one for training and another for testing.

```
train_data_matrix
```

```
array([[0., 0., 0., ..., 0., 0., 0.],
       [0., 0., 0., ..., 0., 0., 0.],
       [0., 0., 0., ..., 0., 0., 0.],
       ...,
       [0., 0., 0., ..., 0., 0., 0.],
       [0., 0., 0., ..., 0., 0., 0.],
       [0., 0., 0., ..., 0., 0., 0.]])
```

Figure 6: user-item train matrix

```
test_data_matrix
```

```
array([[0., 0., 0., ..., 0., 0., 0.],
       [0., 0., 0., ..., 0., 0., 0.],
       [0., 0., 0., ..., 0., 0., 0.],
       ...,
       [0., 0., 0., ..., 0., 0., 0.],
       [0., 0., 0., ..., 0., 0., 0.],
       [0., 0., 0., ..., 0., 0., 0.]])
```

Figure 7: user-item test matrix

2) Calculate the distance and create similarity matrices(user_similarity and item_similarity),for both training and testing data.

```
user_similarity
```

```
array([[0.        , 1.        , 0.88598347, ..., 1.        , 1.        ,
        1.        ],
       [1.        , 0.        , 0.84797796, ..., 1.        , 1.        ,
        1.        ],
       [0.88598347, 0.84797796, 0.        , ..., 1.        , 1.        ,
        1.        ],
       ...,
       [1.        , 1.        , 1.        , ..., 0.        , 1.        ,
        1.        ],
       [1.        , 1.        , 1.        , ..., 1.        , 0.        ,
        1.        ],
       [1.        , 1.        , 1.        , ..., 1.        , 1.        ,
        0.        ]])
```

Figure 8: user similarity matrix

```
user_similarity
```

```
array([[0.        , 1.        , 0.88598347, ..., 1.        , 1.        ,
        1.        ],
       [1.        , 0.        , 0.84797796, ..., 1.        , 1.        ,
        1.        ],
       [0.88598347, 0.84797796, 0.        , ..., 1.        , 1.        ,
        1.        ],
       ...,
       [1.        , 1.        , 1.        , ..., 0.        , 1.        ,
        1.        ],
       [1.        , 1.        , 1.        , ..., 1.        , 0.        ,
        1.        ],
       [1.        , 1.        , 1.        , ..., 1.        , 1.        ,
        0.        ]])
```

Figure 9: item similarity matrix

3) Make predictions based on the user-item rating matrices and similarity matrices.

```
item_prediction
```

```
array([[0.00412961, 0.00385753, 0.00426215, ..., 0.00445284, 0.00444542,
        0.00381702],
       [0.00105873, 0.00101878, 0.00109508, ..., 0.00111451, 0.00111136,
        0.00099313],
       [0.05720133, 0.05549695, 0.05883837, ..., 0.06076822, 0.06040087,
        0.05436652],
       ...,
       [0.00690281, 0.00657473, 0.007403  , ..., 0.00799993, 0.00798857,
        0.00619617],
       [0.        , 0.        , 0.        , ..., 0.        , 0.        ,
        0.        ],
       [0.00959724, 0.00938662, 0.0098244 , ..., 0.01011976, 0.01008683,
        0.00922643]])
```

Figure 10: Predictions based on item similarity

```
user_prediction
```

```
array([[ 0.43834311,  0.71546707,  0.10399271, ..., -0.04890126,
        -0.04897668,  1.14828828],
       [ 0.42910337,  0.71073575,  0.09847456, ..., -0.0524954 ,
        -0.05258731,  1.13754994],
       [ 0.48717456,  0.76120747,  0.16023045, ...,  0.00909813,
         0.00903977,  1.18005461],
       ...,
       [ 0.42987544,  0.70698924,  0.1051907 , ..., -0.04495726,
        -0.04501941,  1.11787391],
       [ 0.42557123,  0.70723532,  0.09605476, ..., -0.05385274,
        -0.05395274,  1.13405666],
       [ 0.43489945,  0.71507126,  0.10687344, ..., -0.04311202,
        -0.04319074,  1.1382911 ]])
```

Figure 11: Predictions based on user similarity

We predict the rates the users would give to the anime.If we recommend the highest rated anime to the users,We would recommend 'Yume de Aetara' (user_based)and 'Abenobashi Mahou Shoutengai' (item_based) to user1.

3.2 Model-Based Collaborative Filtering

The second category of recommendation system is model based approach, which give a reduction of sparse user characteristic matrix. Two main model of this approach are SVD and KNN, which will be used in this project.

1) SVD is one of the useful algorithm in model-based recommendation system. Using SVD to build a new matrix and calculate the similarity under the matrix, which can greatly improve the effect of the recommendation system.

```
anime_pred_svd = svd_algo.predict(uid = 325, iid = 2398)
anime_pred_svd_score = anime_pred_svd.est
print(anime_pred_svd_score)
```

```
7.707539330292379
```

Figure 12: The prediction of SVD

SVD can be used to predict a specific user's rating score for a specific anime that he/she has never watched. For example, as the result of prediction of SVD shows above, the user 325 will rate the anime 2398 (Uchuu Kaizoku Mito no Daibouken) around 8, which means the system would recommend it to the user 325.

2) Besides SVD, KNN is another method usually used on recommendation system. KNN do the recommendation by grouping the users who usually watch same types of animes, which can make full use of the users' characteristic. As there are a lot of different variables, KNN is a more simple and accurate model.

```
anime_pred_knn = knn_b_algo.predict(uid = 68, iid = 5114)
anime_pred_knn_score = anime_pred_knn.est
print(anime_pred_knn_score)
```

```
7.866090419174773
```

Figure 13: The prediction of KNN

As we can see from the result, the user 68 will rate the anime 5114 (Yondemasu yo, Azazel-san. (TV)) around 8. This movie should be recommend to the user 68 as he/she might like it.

3.3 LightGBM

Besides traditional recommendation algorithms like CF method, some GDBT algorithms may also solve this kind problem, such as XGBoost and LightGBM. Due to the lower accuracy of XGBoost, LightGBM is finally applied in this project.

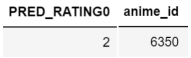
According to the result of K-means, the target(rating_x) is divided into three categories:

1) 0: Dislike (rating_x from 1 to 3)

2) 1: Medium (rating_x from 4 to 7)

3) 2: Like (rating_x from 8 to 10)

From the Figure 14, the predict rating in the first line is 2 which means the user may like the anime (anime_id 6350: Chihayafuru 2: Waga Mi Yo ni Furu Nagame Seshi Ma ni). Then our recommendation system would recommend it.



PRED_RATING0	anime_id
2	6350

Figure 14: The prediction of LightGBM

4 Evaluation

From Table 1,the RMSE of test data is slightly smaller than that of the training data,which shows that the model performs better on the test data.

Besides, the model of SVD and KNN are both overfit. the RMSE of SVD without hypertunning is 0,724(train) and 1.233(test), which is pretty overfit. SVD performs better after hypertuning. Also, the RMSE of KNN without paramtere select is 0.817(train) and 1.282(test). It's still overfit after hypertunning.

Moreover, the accuracy of LightGBM is 0.67(train)/0.66(test) which is not very high even though after hypertuning(hyperband).

Table 1: Evaluation of models

Methods	Train RMSE/Accuracy	Test RMSE/Accuracy
Memory_Based CF(user based)	7.38	7.756
Memory_Based CF(item based)	7.774	7.889
SVD	1.237	1.256
KNN	0.894	1.262
LightGBM	0.67	0.66
LightGBM(Tunning)	0.67	0.66

5 Challenges

This project has several challenges:

1) Large amount of data - around 70000 raw data (We only choose 10000 for running time reason. We believe the result will be more accurate if we use all data).

2) The data format is confusing and needs to be converted into a type that can be analysed.

3) Combine multiple analysis models and methods into one project.

6 Reference

CopperUnion, *Anime Recommendation Database.* URL:https://www.kaggle.com/CooperUnion/anime-recommendations-database.

Lastnight, *User Clustering for anime recommendation.* URL:https://www.kaggle.com/tanetboss/user-clustering-for-anime-recommendation.