MOVIE RECOMMENDATION SYSTEM

Venkat Koushik Muthyapu, Karthik Sree Kanthan, Jacob

CS 467/567 Principles and Applications of Big Data

THE UNIVERSITY OF NEW MEXICO

Abstract and Introduction

The over abundance of information that is available over the internet makes the process of information seeking a difficult process. This has especially now become more relevant with the arrival of e-commerce into the picture. Now a days more and more e-services are being created and released for public consumption.
With this increase in the large number of services, the companies now need to figure out the best way to attract and maintain the customers. The best way to do this is by coming up with a way to recommend them some new products or the products that have not been used by the consumers yet. Rather than coming up with random products as suggestions, companies need to recommend only quality products which have certain similarity to the user who purchases it or at-least by looking at the ratings provided by the customers who have already used these products. To give confidence in buying the products to the consumers, recommender systems were built.
As a part of our project we are implementing a recommender system using the famous MovieLens data set which consists of 100,000 ratings from 1000 users on 1700 movies. Thus the product being recommended to the users here are movies based on ratings of the particular user or the other users.

Recommender System (RS)

A recommender system (RS) gives personalized recommendations of products to the users by using different techniques, which helps the user to choose appropriate product from a large variety of options. These recommendations are done based on the user's interests or from the opinion of other users. The primary purpose of a recommender system is to provide personalize assistance to the customers in selecting products. There are three different ways to implement recommender systems:

- Content-based Recommender System: This system makes use of certain discrete characteristics of an item so as to recommend other products which are similar.
- Collaborative filtering Recommender System : This system makes the recommendations from a user's past behaviors and decision of other users . This system is then capable of predicting items that users might find interesting.
- Hybrid Recommender System: Combines the positives from the previous two approaches to make recommendations more accurate.

Content-based RS

To analyze description of content alone
In this technique, the system recommends any similar product that have been used or liked before by the particular user. This can be done by finding similarity between all pairs of items and then it chooses a item similar to our users rated item to generate a list of recommendations. Usually the similarity is obtained from description of item which is done by computing the TF-IDF scores. This technique used to count the concurrence of each word in a document and weight them according to the importance, and also calculate a score for the document. Building user profile and item profile based on user rated content. The term frequency (TF) and the Inverse Document Frequency (IDF) can be computed as below :

$$\mathrm{Tf}(t) = \frac{\text{Frequency occurence of term t in document}}{\text{Total number of terms in document}} \qquad \mathrm{Idf}(t) = log_{10}(\frac{\text{Total Number of documents}}{\text{Number of documents containing term t}})$$

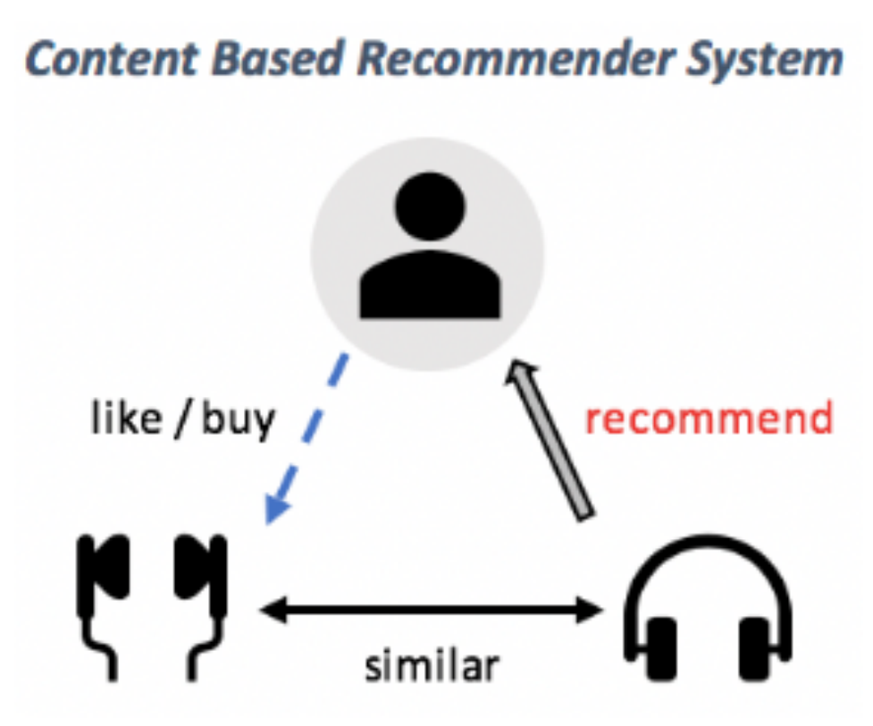
$$\textbf{TF-IDF score: } w_{ij} = TF_{ij} \times IDF_i$$

Once computed, we find the TF-IDF score by taking a product of these two for each movie vector in our dataset. This gives a score for each item in our dataset. Now once we have this scores we need to compute the similarity between each item using cosine similarity .

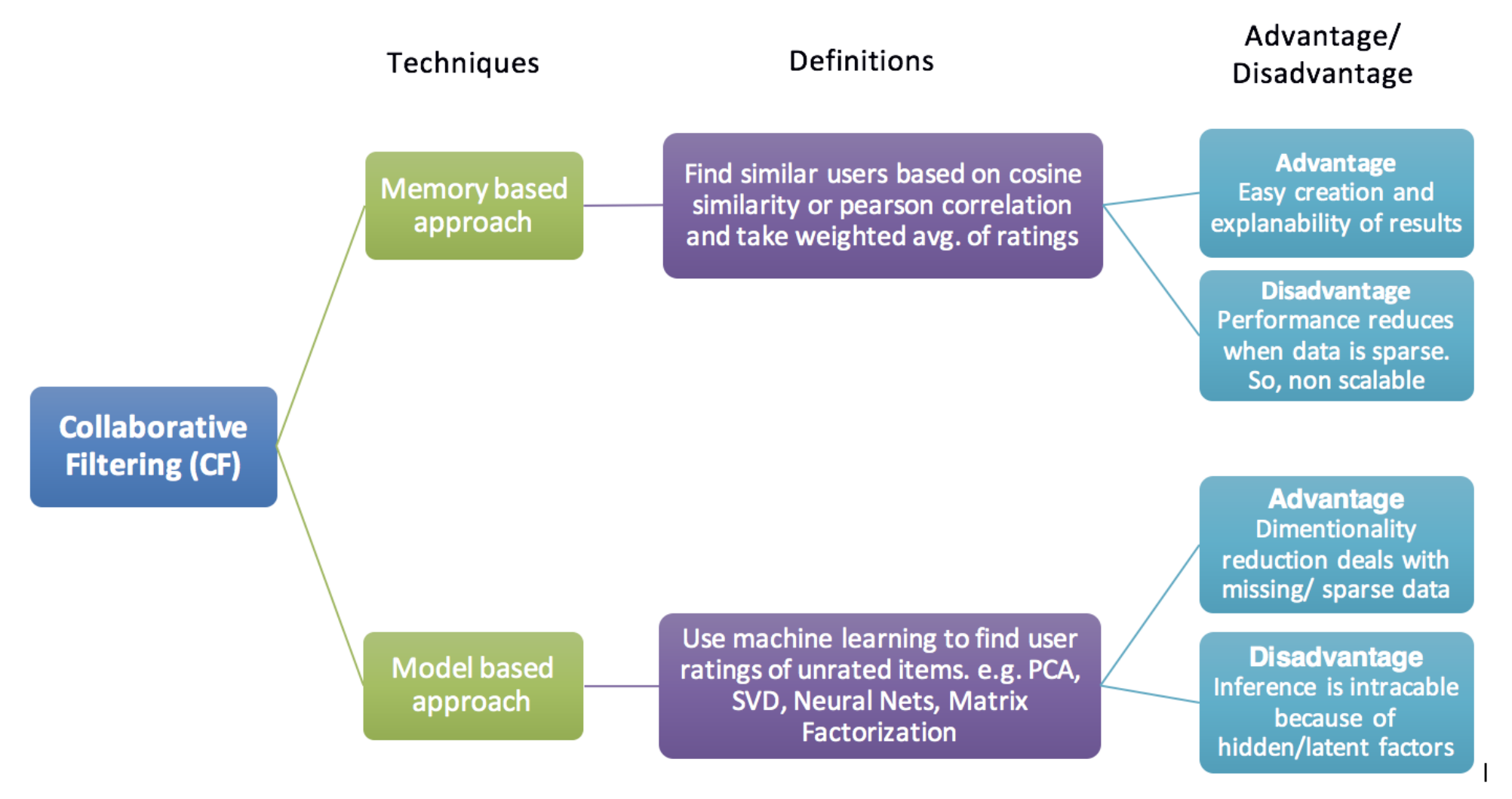
Given user profile $\boldsymbol{x}$ and item profile $\boldsymbol{i}$, estimate

$$u(\boldsymbol{x}, \boldsymbol{i}) = \cos(\boldsymbol{x}, \boldsymbol{i}) = \frac{\boldsymbol{x} \cdot \boldsymbol{i}}{||\boldsymbol{x}|| \cdot ||\boldsymbol{i}||}$$

This gives recommendations based on most similar items.



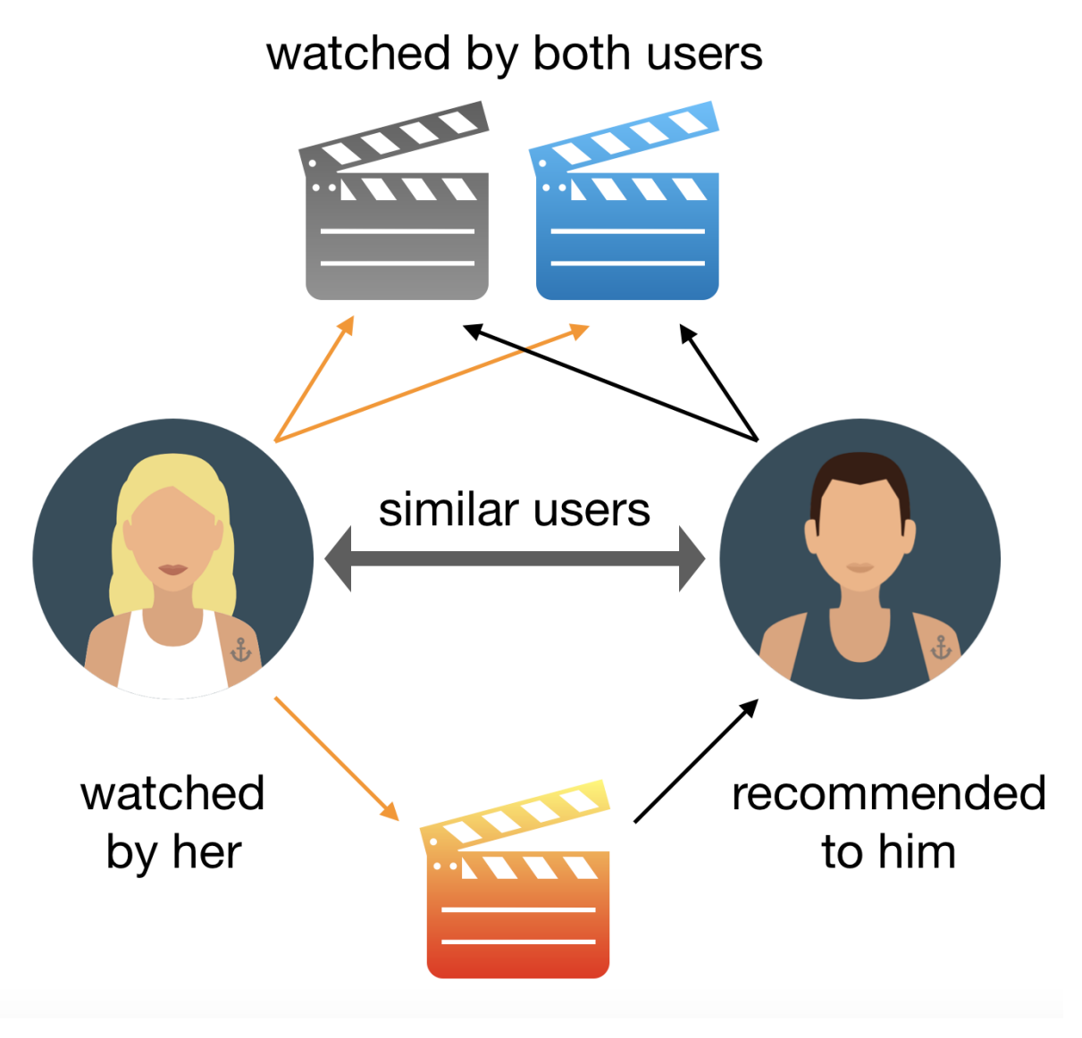
Collaborative Filtering RS (CF)


Techniques
Definitions
Advantage/ Disadvantage
Collaborative Filtering (CF)
Memory based approach
Find similar users based on cosine similarity or pearson correlation and take weighted avg. of ratings
Advantage Easy creation and explanability of results
Disadvantage Performance reduces when data is sparse. So, non scalable
Model based approach
Use machine learning to find user ratings of unrated items. e.g. PCA, SVD, Neural Nets, Matrix Factorization
Advantage Dimentionality reduction deals with missing/ sparse data
Disadvantage Inference is intracable because of hidden/latent factors

The collaborative filtering has 2 main approaches:

- **Memory Based** CF : It has two main approaches one is user-item filtering, in this approach the system finds similar users based in similarity of ratings and recommends the item that are like by both. The second approach is item-item filtering, in this approach the recommendations are done by taking an item and looking at the users who have like it and find items that are like by them or similar users.
- **Model based CF** : In this approach the system will be using machine learning algorithms in order to predict the ratings of unrated items. There are three different algorithms that can be used for this approach: clustering based algorithm, Matrix factorization based algorithms and Deep Learning.

i. Clustering based algorithm: It is almost same as memory based algorithm, where we find similarities using cosine similarity but in this system we are calculating our data as unsupervised learning model in order to improve the scalability.
ii. Matrix factorization based algorithms: In this technique the data is represented as matrix and then the matrix is broken down into two smaller matrices such that when multiplied it gives the original matrix. When the matrix is multiplied, few parameters are used to predict the unrated products.
iii. Deep Learning: In this technique neural nets are used to implement the collaborative filtering.



Our Model

After going through great amount of research on recommender systems we have came to a conclusion that matrix factorization algorithm would be a reasonable algorithm based on the type of data we have. To be precise, we are using the Alternative Least Square algorithm (ALS) because it one of the best algorithms that can deal with sparse data along with high scalability and efficient run time compared to other matrix factorization techniques.

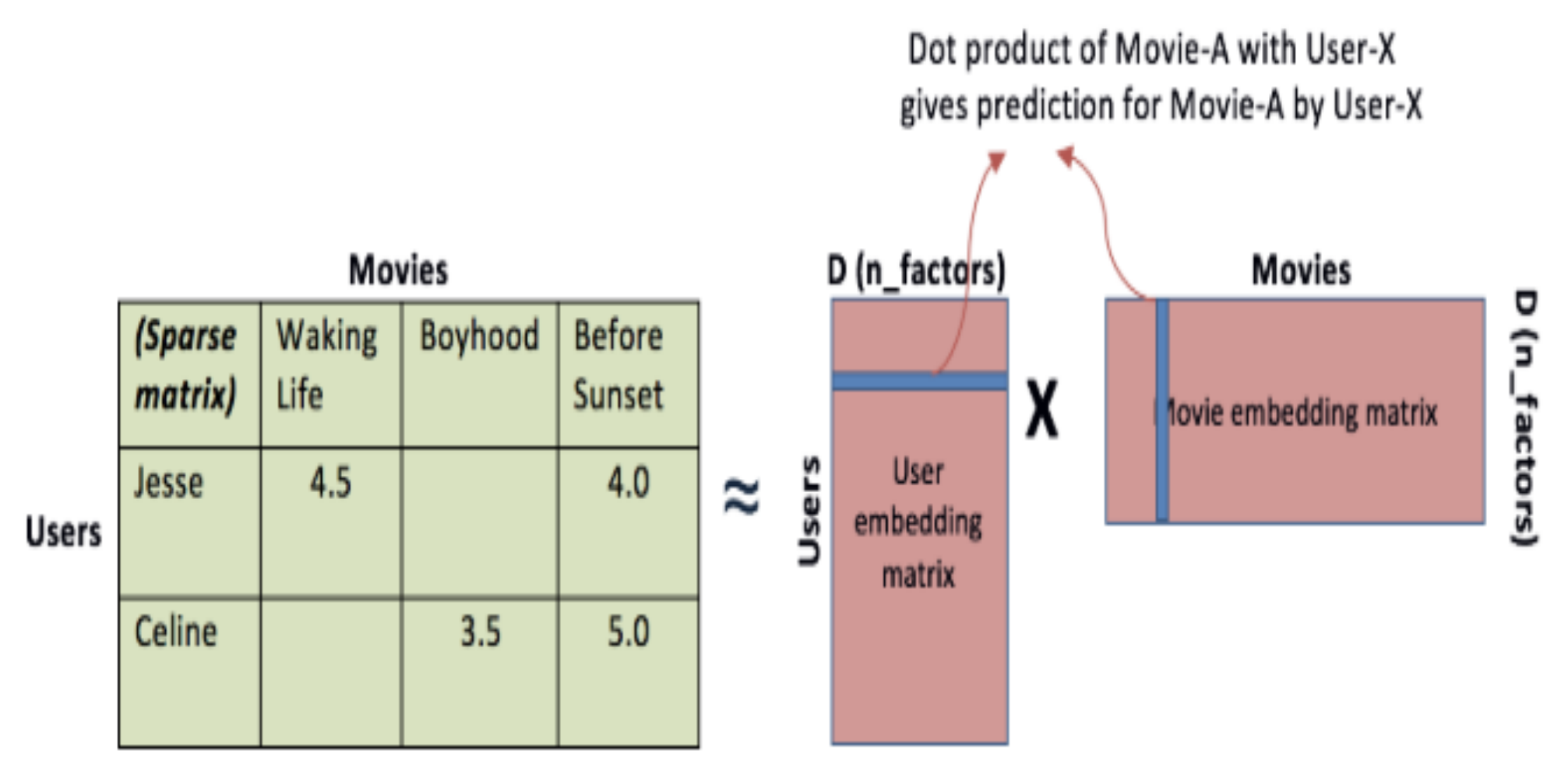


Figure 4. Visualization of matrix factorization

Our rating matrix R consists of users as rows and movies as columns. This matrix is then factorized into two smaller matrices P and Q. The P matrix consists of rows and columns where rows represent latent features and the columns represents movies where as Q consists of rows and columns where rows represent user and the columns latent features.

$$Error_{ij} = \underbrace{\sum w_{ij}}_{\text{Completion Term}} \cdot \underbrace{(R_{ij} - u_i \times p_j^{\ T})}_{\text{Cost Function}} + \underbrace{\lambda(\| \mathrm{U} \|_2 + \| \mathrm{P} \|_2)}_{\text{Regularization Term}}$$

Completion Term

$$\text{where } w_{ij} \begin{cases} 1 \; R_{ij} \text{ is known} \\ 0 \; R_{ij} \text{ is unknown} \end{cases}$$

Cost Function
Minimizes the difference between the product of our factor matrices and the original ratings matrix.

The matching solutions for u_i and p_j are:

$$u_i = (P^T \times w_i \times P + \lambda I)^{-1} \times P^T \times w_i \times r_i$$

$$p_j = (U^T \times w_j \times U + \lambda I)^{-1} \times U^T \times w_j \times r_j$$

Regularization Term
Prevents overfitting by applying a small amount to the error which requires more time/iterations to full minimize.

In the ALS model, we are trying to alternatively minimize two cost functions(error), where the first error function holds the user matrix fixed and runs gradient descent with movies matrix where as the second error function holds the movies matrix and runs gradient decent with respect to user matrix. This is more scalable than other matrix decomposition techniques such as PCA because ALS is running the gradient decent algorithm over multiple partitions of training data from a cluster of machines.
We increase the number of latent factors to improves the personalization. But, after a certain point of time the number of factors become too high and the model starts to over fit. To prevent this we introduced the regularization parameter to the above equation. Thus, the above equation on a whole minimizes the error between true rating and predicted rating.

MOVIE RECOMMENDATION SYSTEM

Venkat Koushik Muthyapu

THE UNIVERSITY OF NEW MEXICO

CS 467/567 Principles and Applications of Big Data

Methodology

Implementation:

To make our recommendation system we have implemented an ALS machine learning algorithm which makes use of matrix decomposition. The MovieLens data set consists of 4 csv files , movies.csv, ratings.csv, links.csv and tags.csv. These csv files where then loaded as tables and changed them to RDD's using pyspark. For the implementation of our recommender system we made use of movies and ratings table since we tried to predict the ratings for unrated movies based on user ratings.

The data in ratings table was then split at random into three sets; training dataset which consisted of 60 percent of data, validation dataset which consisted of 20 percent of data and finally the test data set which consisted of the remaining 20 percent of data. The ALS model has three important parameters :

- maxIter: The maximum number of iteration to run. This was set to 10 in our model.
- Rank: The number of latent features in the model. This was set to 18 in our model.
- regParam: Regularization parameter which was set to 0.05 in our model.

The above parameters are used to train our training data set. After the training we have tested our model on test data and we see that we have a RMSE (Root Mean Square Error) as follows :

```
For rank = 18 reg = 0.05  the RMSE=  0.9803222190248909
```

From the above result its evident that our model was very much close to over fitting. To avoid this from happening we have followed a evaluation method as follow.

Evaluation:

For finding a reasonable ALS model for our dataset we used the following values for the parameter.

- maxIter: The maximum number of iteration to run. This was set to 10 in our model.
- Rank: The number of latent features in the model. The model is tested with [8, 10, 12, 14, 16, 18, 20] values
- regParam: Regularization parameter which was tested with [0.001, 0.01, 0.05, 0.1, 0.2] in our model.

After this, we have iterated through these parameters and trained on the training data set and performed cross validation on validation dataset. We obtained the following results out of which our best model was as follow

```
For rank = 8 reg = 0.001  the RMSE=  1.4467448634372828
For rank = 8 reg = 0.01  the RMSE=  1.141916955112322
For rank = 8 reg = 0.05  the RMSE=  0.9856996845615988
For rank = 8 reg = 0.1  the RMSE=  0.914186924444086
For rank = 8 reg = 0.2  the RMSE=  0.8966076281249638
For rank = 10 reg = 0.001  the RMSE=  1.4560396253085752
For rank = 10 reg = 0.01  the RMSE=  1.2036612472694024
For rank = 10 reg = 0.05  the RMSE=  0.9956121741381563
For rank = 10 reg = 0.1  the RMSE=  0.9158487323974703
For rank = 10 reg = 0.2  the RMSE=  0.8964889415116521
For rank = 12 reg = 0.001  the RMSE=  1.5242743892266395
For rank = 12 reg = 0.01  the RMSE=  1.235516337012286
For rank = 12 reg = 0.05  the RMSE=  1.0054946510104066
For rank = 12 reg = 0.1  the RMSE=  0.9178090013903913
For rank = 12 reg = 0.2  the RMSE=  0.8985851115100361
For rank = 14 reg = 0.001  the RMSE=  1.5311798036079438
For rank = 14 reg = 0.01  the RMSE=  1.2574262798813998
For rank = 14 reg = 0.05  the RMSE=  1.004432265748177
For rank = 14 reg = 0.1  the RMSE=  0.9163648847902194
For rank = 14 reg = 0.2  the RMSE=  0.899009456306896
For rank = 16 reg = 0.001  the RMSE=  1.5977755203925776
For rank = 16 reg = 0.01  the RMSE=  1.2945525589067843
For rank = 16 reg = 0.05  the RMSE=  1.004848074487897
For rank = 16 reg = 0.1  the RMSE=  0.9149908997725777
For rank = 16 reg = 0.2  the RMSE=  0.8988097775919106
For rank = 18 reg = 0.001  the RMSE=  1.64173527955219
For rank = 18 reg = 0.01  the RMSE=  1.3049537155223991
For rank = 18 reg = 0.05  the RMSE=  1.008261040265912
For rank = 18 reg = 0.1  the RMSE=  0.915377249956471
For rank = 18 reg = 0.2  the RMSE=  0.8988576731676956
For rank = 20 reg = 0.001  the RMSE=  1.7042923866197208
For rank = 20 reg = 0.01  the RMSE=  1.3538113042962867
For rank = 20 reg = 0.05  the RMSE=  1.0154965096385316
For rank = 20 reg = 0.1  the RMSE=  0.9160800093109257
For rank = 20 reg = 0.2  the RMSE=  0.8999188491754373
The best model was trained with rank=  10 With reg=  0.2
```

From the above evaluations, the best model was found to be the one with rank= 10 and regParam=0.2. This model gave us a RMSE of 0.896488.

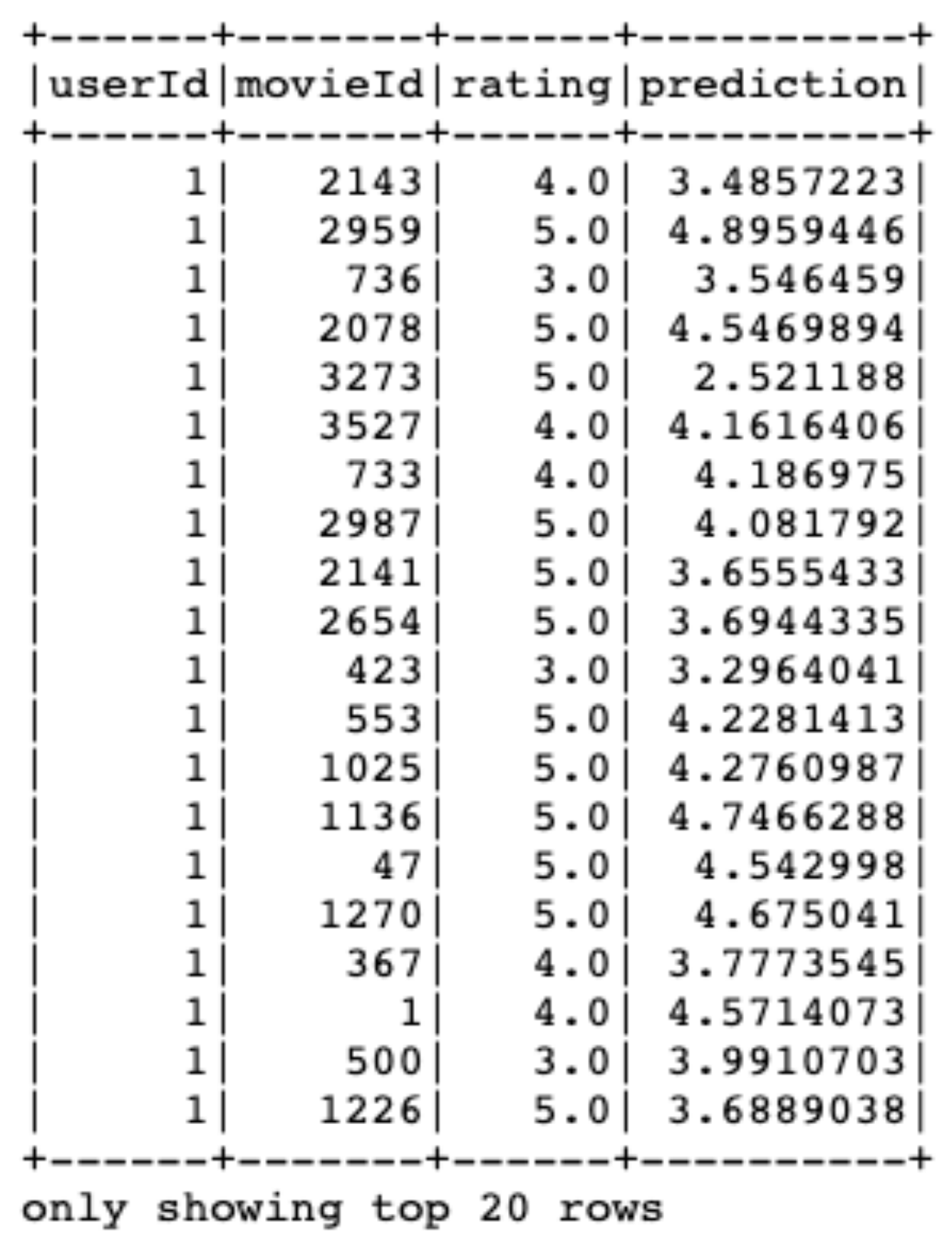
Results:

The best model that was obtained by evaluation on validation data set was now tested on the test data which was split earlier, but never user to or validate or model and the result obtained is as below.

```
test data Rmse=  0.8935807766860681
```

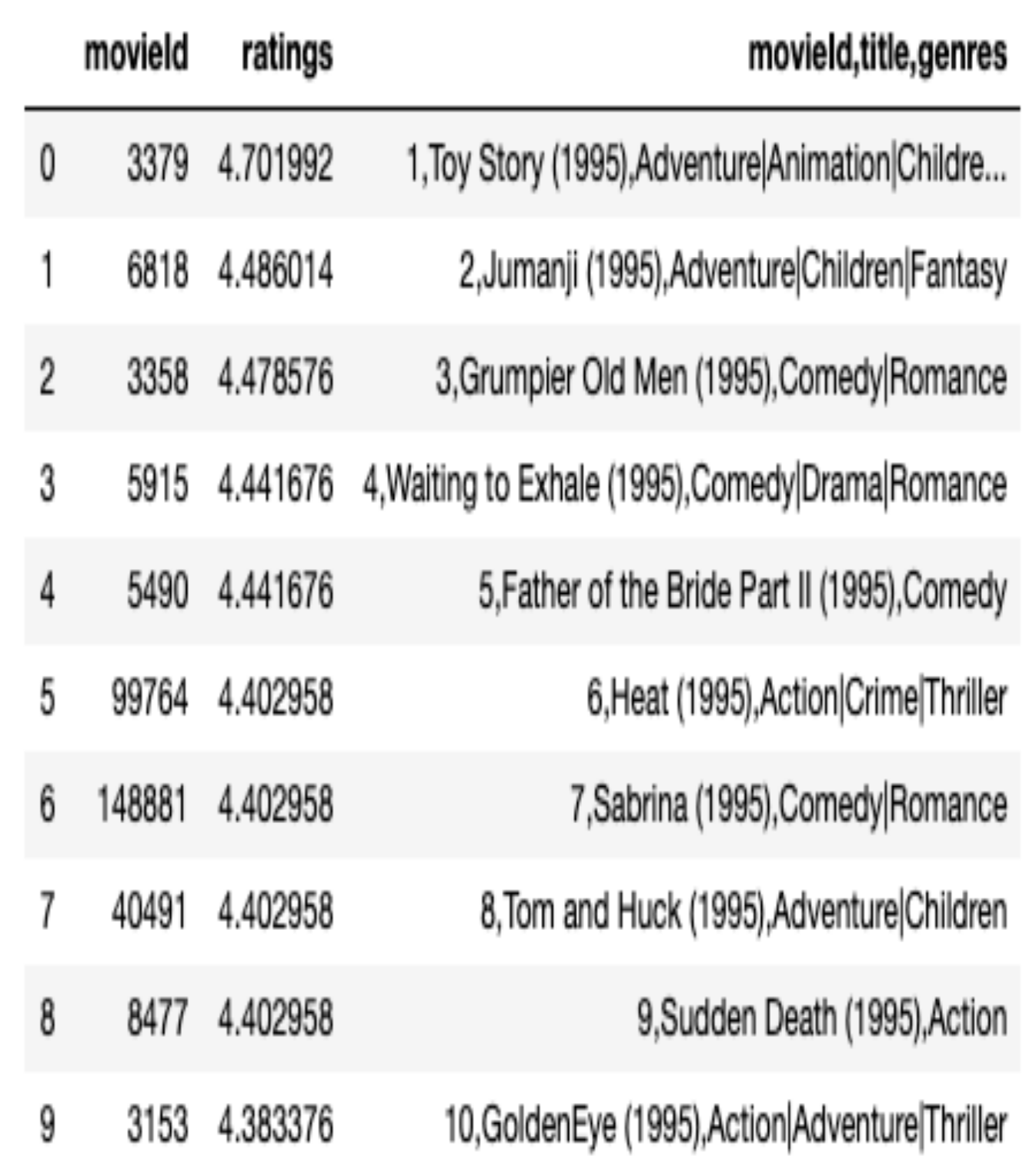
From the above obtained RMSE on test data set it is evident that our model has an RMSE lesser than that on validating data, which to an extent shows that it does not over-fit.

The below table Shows the comparisons between the user ratings and the ratings predicted by our model joined with userId and movieId, displaying the top 20 results



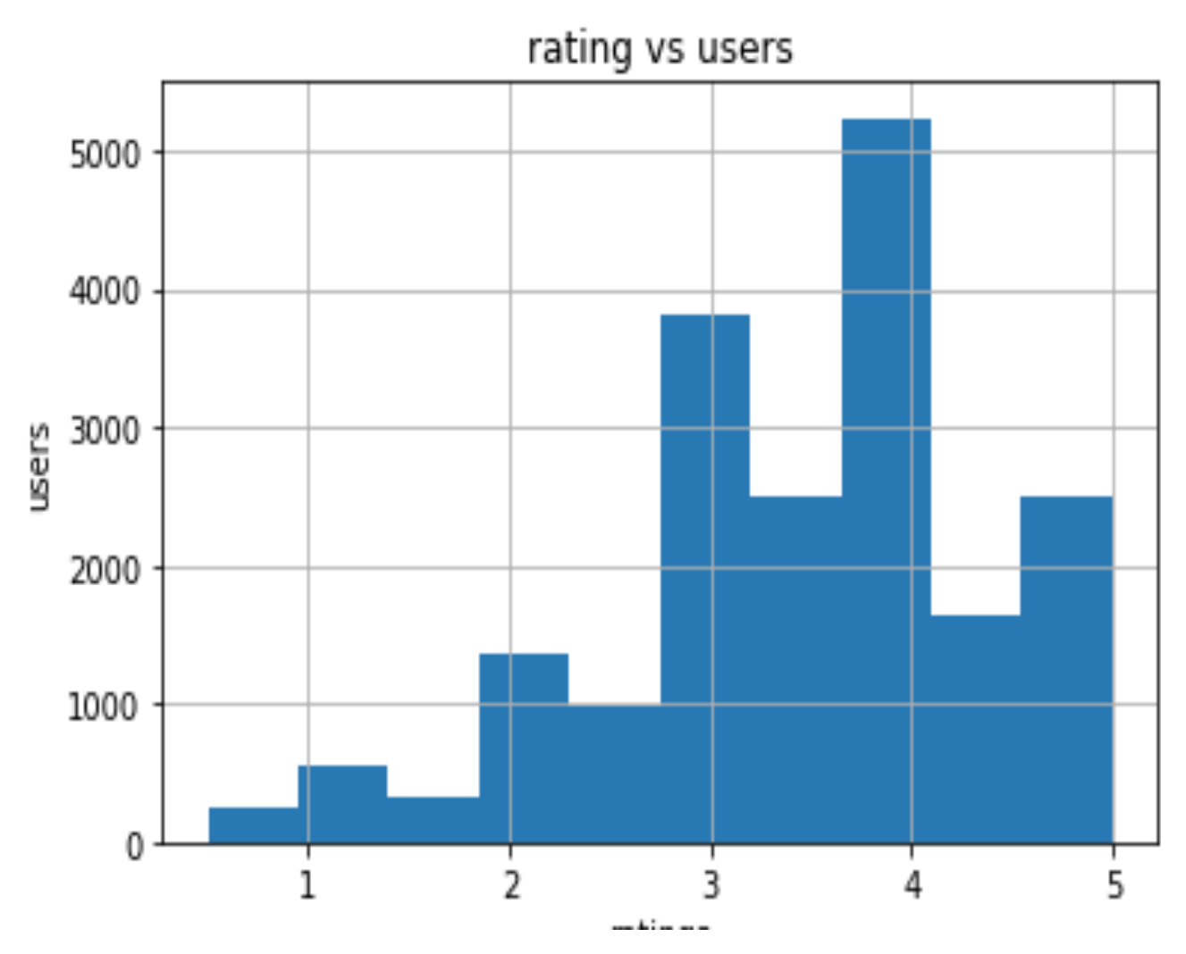
```
+------+-------+------+----------+
|userId|movieId|rating|prediction|
+------+-------+------+----------+
|     1|   2143|   4.0| 3.4857223|
|     1|   2959|   5.0| 4.8959446|
|     1|    736|   3.0|  3.546459|
|     1|   2078|   5.0| 4.5469894|
|     1|   3273|   5.0|  2.521188|
|     1|   3527|   4.0| 4.1616406|
|     1|    733|   4.0|  4.186975|
|     1|   2987|   5.0|  4.081792|
|     1|   2141|   5.0| 3.6555433|
|     1|   2654|   5.0| 3.6944335|
|     1|    423|   3.0| 3.2964041|
|     1|    553|   5.0| 4.2281413|
|     1|   1025|   5.0| 4.2760987|
|     1|   1136|   5.0| 4.7466288|
|     1|     47|   5.0|  4.542998|
|     1|   1270|   5.0|  4.675041|
|     1|    367|   4.0| 3.7773545|
|     1|      1|   4.0| 4.5714073|
|     1|    500|   3.0| 3.9910703|
|     1|   1226|   5.0| 3.6889038|
+------+-------+------+----------+
only showing top 20 rows
```

Here the values indicate that our predicted values are somewhat close to the actual ratings for those particular movies. From this we can infer that our model works good.

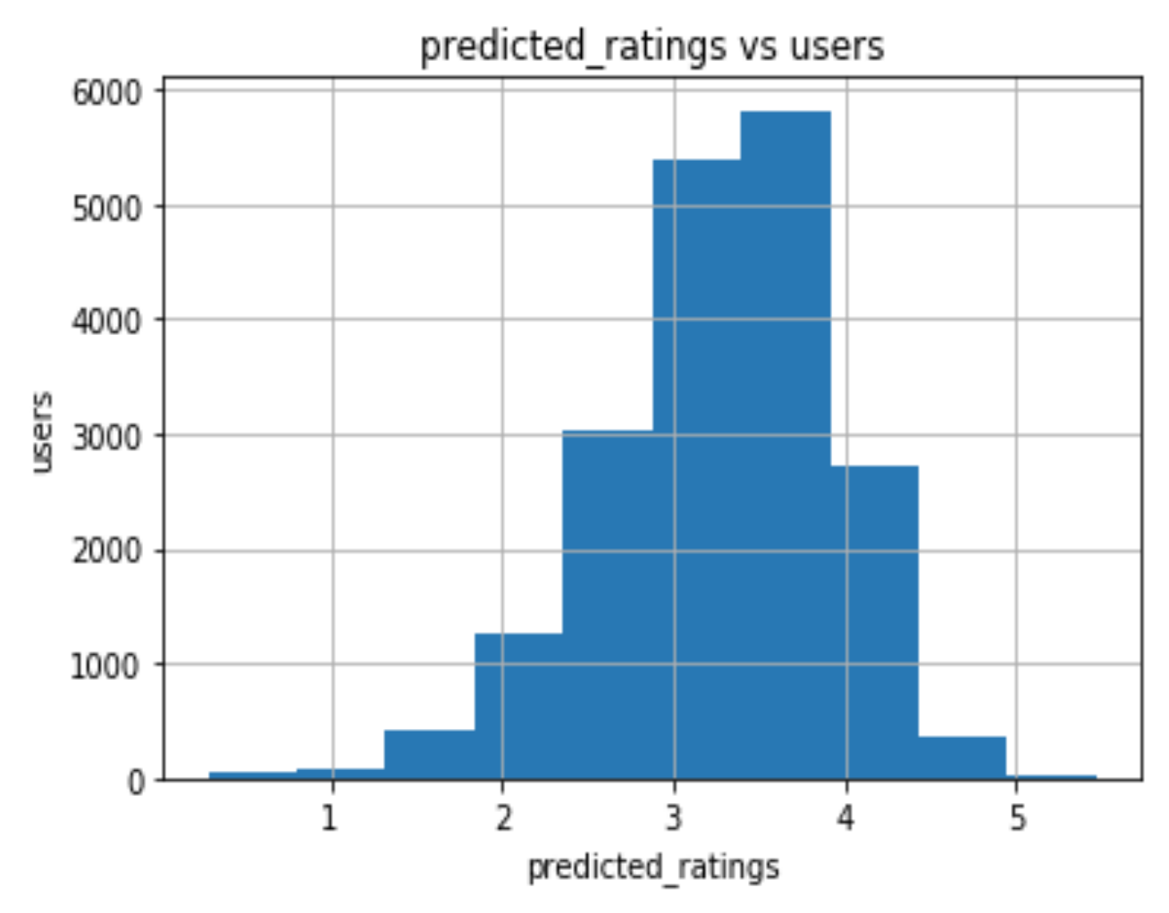


	movieId	ratings	movieId,title,genres
0	3379	4.701992	1,Toy Story (1995),Adventure|Animation|Childre...
1	6818	4.486014	2,Jumanji (1995),Adventure|Children|Fantasy
2	3358	4.478576	3,Grumpier Old Men (1995),Comedy|Romance
3	5915	4.441676	4,Waiting to Exhale (1995),Comedy|Drama|Romance
4	5490	4.441676	5,Father of the Bride Part II (1995),Comedy
5	99764	4.402958	6,Heat (1995),Action|Crime|Thriller
6	148881	4.402958	7,Sabrina (1995),Comedy|Romance
7	40491	4.402958	8,Tom and Huck (1995),Adventure|Children
8	8477	4.402958	9,Sudden Death (1995),Action
9	3153	4.383376	10,GoldenEye (1995),Action|Adventure|Thriller

The above table shows the top 10 recommendation which are recommended by our model for every user.



The above histogram shows the distribution of ratings by different users in the test dataset



The above histogram shows the distribution of predicted ratings by different users in the test dataset

Further Directions

From our research on the recommender system we came to know that is most of the cases an hybrid model can provide us more accuracy for prediction. Thus as a future work we would like to implement one more recommender system technique and combine our present model along with that to obtain a hybrid model Furthermore, to make our present model more accurate we would like to implement the same model with a bigger dataset .

Conclusion

The implementation of the collaborative filtering system was successful with the following learnings about the ALS model :

- As the number of maxIter increased the RMSE decreased.
- Regularization is required on the model to prevent the overfitting.
- When the regularization parameter is around 0.2 , we observed smaller RMSE values.

We have tried to implement the ALS learning curve but could not due to some dimensionality issues

References

[1] "Prototyping Recommender System step by step part-1" retrieved from https://towardsdatascience.com/prototyping-a-recommender-system-step-by-step-part-1-knn-item-based-collaborative-filtering-637969614ea

[2]"What is Alternating least square method in recommendation system" retrieved from https://www.quora.com/What-is-the-Alternating-Least-Squares-method-in-recommendation-systems-And-why-does-this-algorithm-work-intuition-behind-this

[3]"latent factor based method in collaborative filtering" retrieved from https://medium.com/@rabinpoudyal1995/latent-factor-based-method-in-collaborative-filtering-77756a02f675

[4]"Deploying a recommender system for movie lens dataset part-1" retrieved from https://blog.codecentric.de/en/2019/07/recommender-system-movie-lens-dataset/