

**Applications of Machine Learning**

2021 MOD006567 TRI2 F01CAM

Component 3: Report

Movie Recommender System Using Matrix Factorization  
Student ID- 1940984

Table of Contents

1. Objectives-------------------------------------------------------------------------------------- 3
2. Research questions---------------------------------------------------------------- 3
3. Literature Survey and background--------------------------------------------- 3
4. Motivation-------------------------------------------------------------------------------------- 3
5. Introduction------------------------------------------------------------------------------------ 4
6. Content Based---------------------------------------------------------------------- 4
7. Collaborative filtering------------------------------------------------------------- 4
8. Hybrid--------------------------------------------------------------------------------- 4
9. Dataset------------------------------------------------------------------------------------------ 4
10. Pre-Processing Data-------------------------------------------------------------------------- 6
11. Construction of training and validation set--------------------------------------------- 10
12. Methodology----------------------------------------------------------------------------------- 12
13. Learning Loop----------------------------------------------------------------------- 12
14. Loss Function----------------------------------------------------------------------- 12
15. Estimation Process------------------------------------------------------------------ 13
16. Matrix Factorization------------------------------------------------------------------------- 12
17. Training the model----------------------------------------------------------------- 13
18. Evaluation----------------------------------------------------------------------------- 13
19. Experiment-------------------------------------------------------------------------------------- 14
20. Conclusion--------------------------------------------------------------------------------------- 16
21. References--------------------------------------------------------------------------------------- 17
22. Git Repository link----------------------------------------------------------------------------- 17

**Objectives**

The focus of the project is to create a model for Movie Recommendation using one of the widely used technique “Matrix Factorization” in building a recommender system. Main purpose of any Recommender system is to predict what will be the most relevant item to show after considering previous user item interactions. Our model will be using the user’s historical behaviour to predict what he might like in future. We collect user’s interactions with the movies and form a user preference vector, and doing this with all the users we get user-item matrix which is also called as design matrix.

**Research Questions**

* Considering the historical behaviour of the user like the past movie ratings, can we predict user’s likes and dislikes.
* To present most accurate estimates and also to calculate the user and review matrix so that we can minimize the distance between observed rating and estimates.
* Can we implement a function which will evaluate how good a given solution is.
* With the help of user-item training matrix is it possible to find various trends.
* Defining set of latent vectors associated with the user and also with the reviews grouped in matrix form.

**Literature Survey and Background**

Recommender Systems considered as a class for application which helps an individual to avoid confusion for simple problems like overloading information with the help of filtered recommendation on different information or an item. (Rajabpour, Bardsiri, Mohammadighavam, & Molaei, 2014)

There are different techniques which are used for generating recommendations for different problems as the data information are high it is tough to continue getting good accuracy with same quality. (L. Sharma & Gera, 2013)

As the personalized information is very important because of which we can understand a user’s preferences and avoid the unnecessary recommendation.

**Motivation**

1. Previously in the assignment I created a recommender system using ‘cosine similarities’ function where I created tags for the movies using the technique ‘bag of words’ but unfortunately it was not enough for the project.
2. My brother watches gaming videos on streaming platforms and when the video is about to finish, next video is ready which is a similar content video. He need not search every time for a next video to play the platform was already recommending him similar content to watch.
3. I personally like watching movies and when I want to watch something I am always confused what to watch, so with the help of this model, based on historical ratings we can predict the likes for future and recommend them accordingly.

**Introduction**

What is a recommender System?

Recommender System is nothing but a tool which helps user to find similar content and overcome the information overload. With the help of building a interest model based on the users It will predict the interest of the user.

With the help of the knowledge on a product the system will recommend the user with the help of the user’s preferences. Now a days in almost all the sectors, we can see the usage of recommender system, for example e-commerce, Spotify, YouTube, Facebook, Instagram, Netflix and many more.

If classified broadly there are 3 types of recommender systems.

1. **Content based**

Content based recommender system is nothing but the recommendation is done based on the similar content.   
Content-based recommender system is continuation and development of collaborative filtering, where we don’t need any user’s information or evaluation.

1. **Collaborative filtering**

It recommends based on the users’ interest, for example there are 2 users. A and B and based on their behaviour we can see that they have a similarity score of 0.9. And if a movie M1 is seen by user A and the user liked the movie. Then as user B is similar to that of A, the movie M1 is recommended to user B as well.

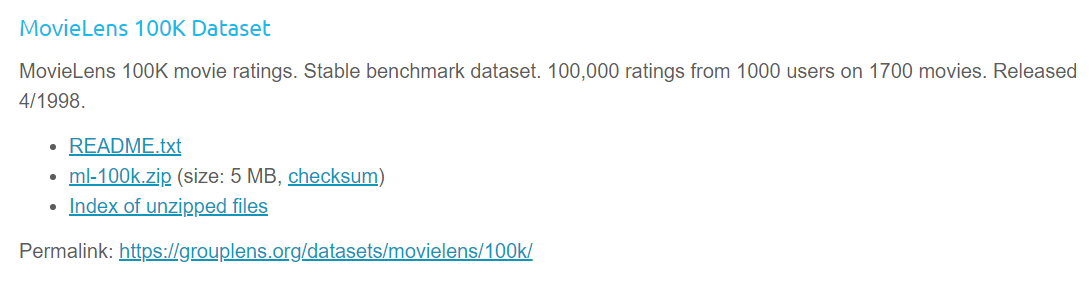
1. **Hybrid**

The combination of both Content and Collaborative based is known as Hybrid.

**Dataset**

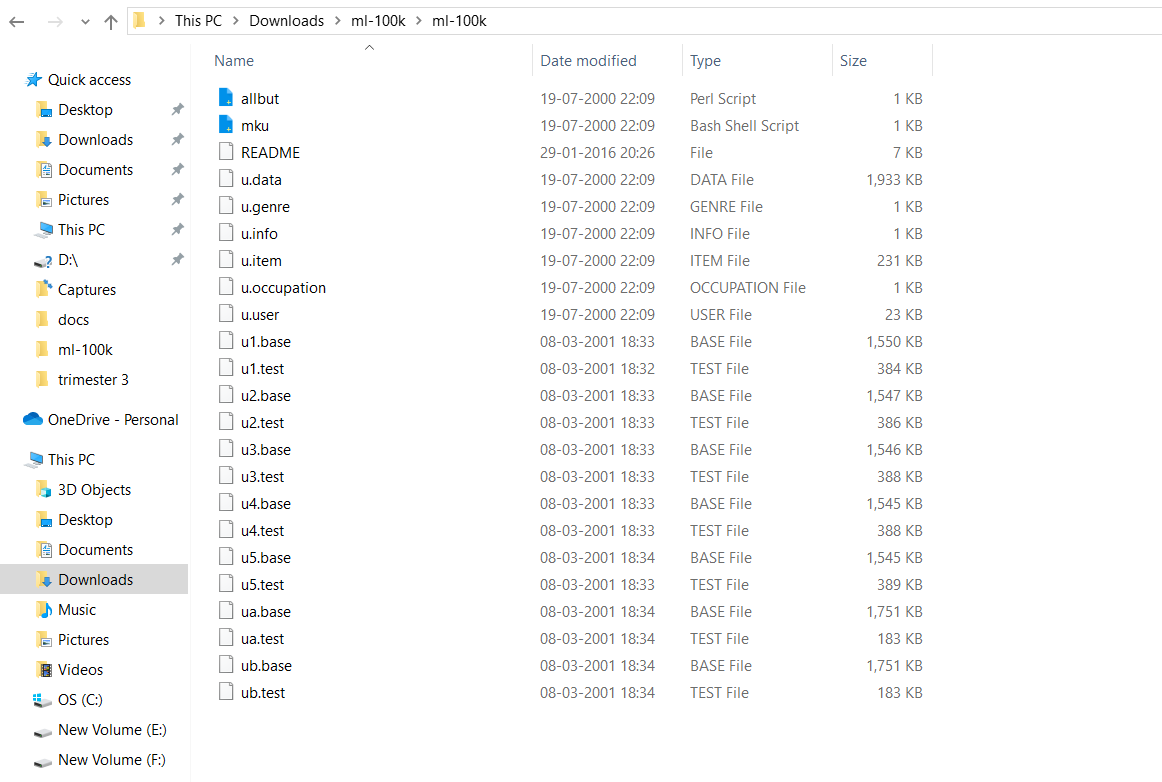
For developing a model to recommend movies based on the historical behaviour and liking of a user I felt the dataset provided by MovieLens is suitable for this particular type of problem.

So, for this project we will be using one of the “MovieLens” dataset to train and build our model. There are many datasets available, but for the type of model we require user’s historical ratings for the movies so considering the requirements of the model I have chosen ML-100k dataset which has around 100k movie ratings of around 1600 movies which were reviewed by 943 users. In addition to these we also have information about the user that performed the movie ratings and the details of the movies that were rated.



To work with this dataset, we will be using dependence library, pandas is a python library that provides fast and easy to use and powerful tool to analyse and manipulate data.

It is very important to understand the characteristics of our data and to make sure that what data we use for training can have a large effect on how good our recommendations will be on our future data points.



Let’s look at the set of items from the u.items csv file. This file contains one row per each movie where in for each movie we will be considering 3 features:

Movie\_id  
movie\_title

and corresponding genre

This dataset contains other features as well such as imdb link and the release date, and for now we are ignoring these data for now.

In ratings as already discussed we have 100k lines where each line has 4 different features. The features are as follows:

User\_id

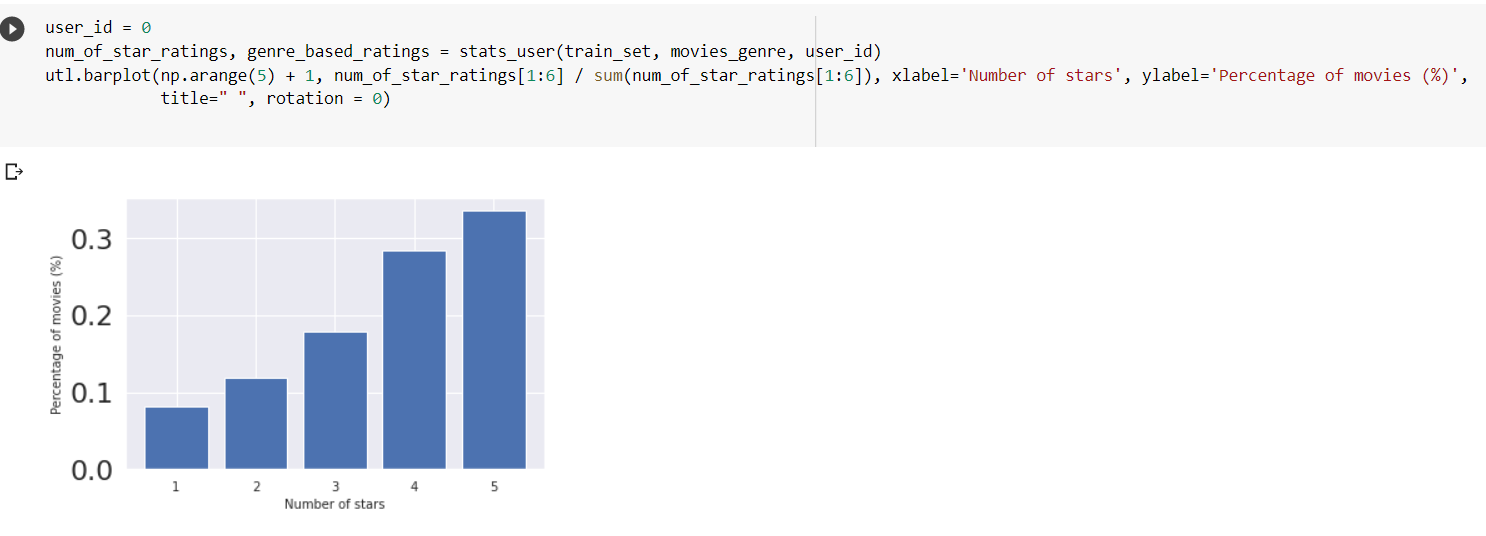
Movie\_id

Associated user rating

And timestamp for when the rating was done

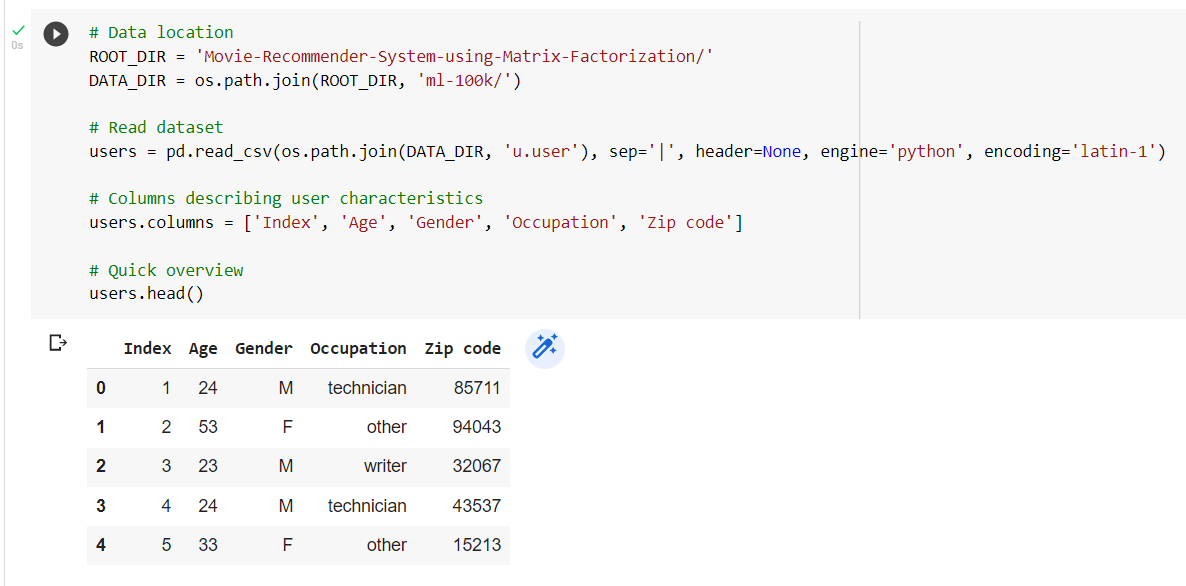
Where these 100k ratings are split into two files for training and test set in 80:20% respectively.

U1 base is the file where we can find the training dataset. And u1.test is the test dataset.

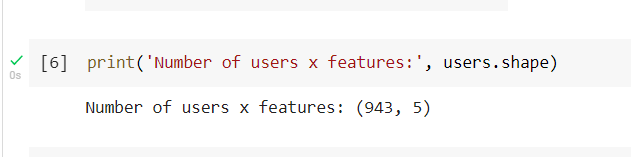


**Pre-Processing Data**

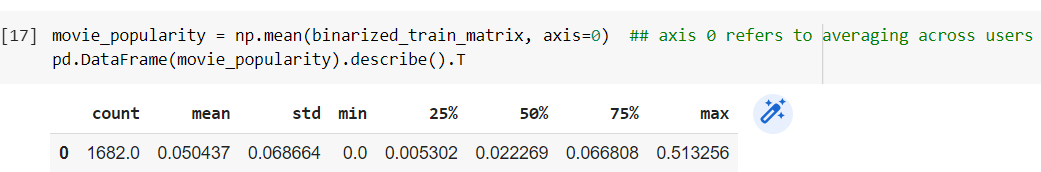
We provide the dataset location and naming it as shortform of Data Directory and to read this data we use read.csv which is a pandas function to read the data into a data frame. After getting the data frame object we can set the columns parameters of this data frame which are required for processing and training our model and then calling the data to view by using the head() function.



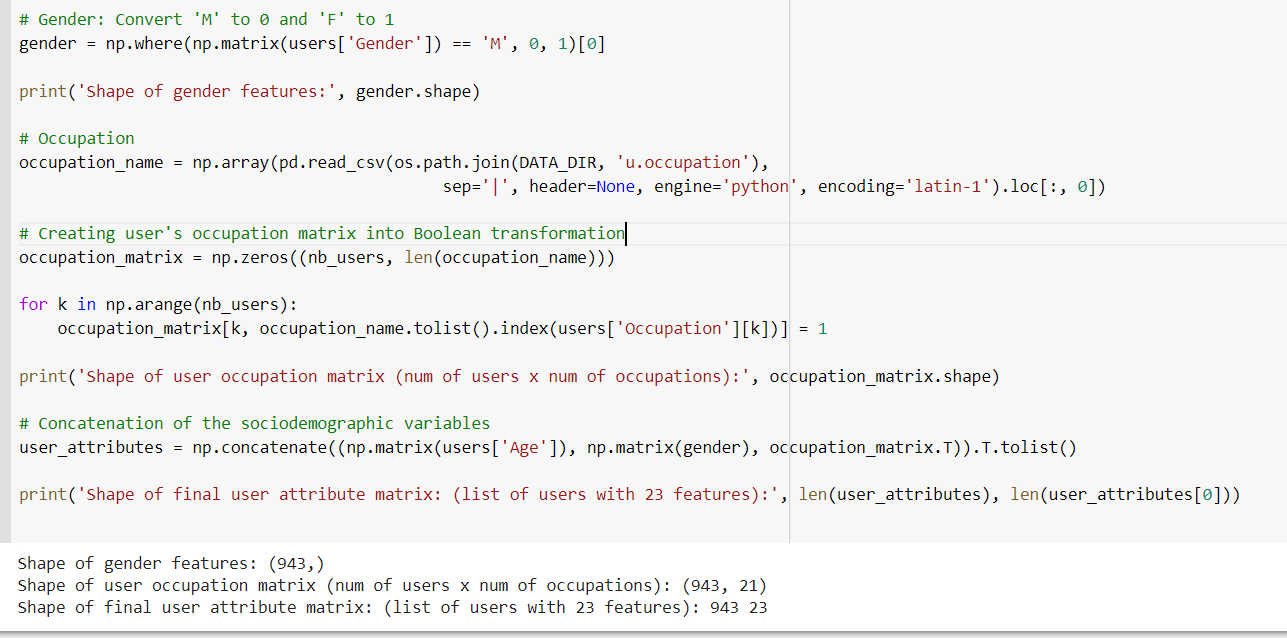
In the user dataset we can clearly see that there are 943 number of users with the help of shape function. And each user have 5 different features.



Now we start performing transformation of data frame such that the format works well with our model, for that specifically, all we want to do is transform categorical data into a binarized format and then collect it into a list of features for every individual user.



So, first look for all the values in the table, and we can see that the Gender and Occupation column in the table has string values so we will convert those into binary format. To do that we will assign the values of 0 and 1 for the gender column where 0 indicates male and 1 female.

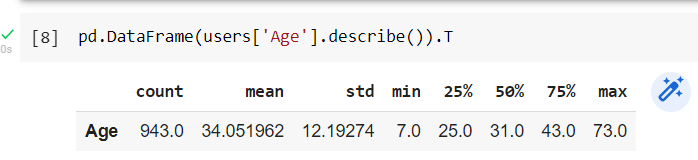


Similarly, we will convert the occupation column and store it in ‘occupation name’ numpy array which contains list of all the occupations a user can have. Now we create a matrix for the occupation where we have number of users and number of all occupations which then we will iterate through each user and assign the concerned occupation which will match with the particular user.

We then concatenate all the features related to the user and store them in ‘user\_attributes’ list which contains the user’s age, gender and the occupation matrix. And if we look at the shape of the user attributes list where we will be having 23 features in total for 943 users. It is an underlying data.

Let’s understand User a bit better-

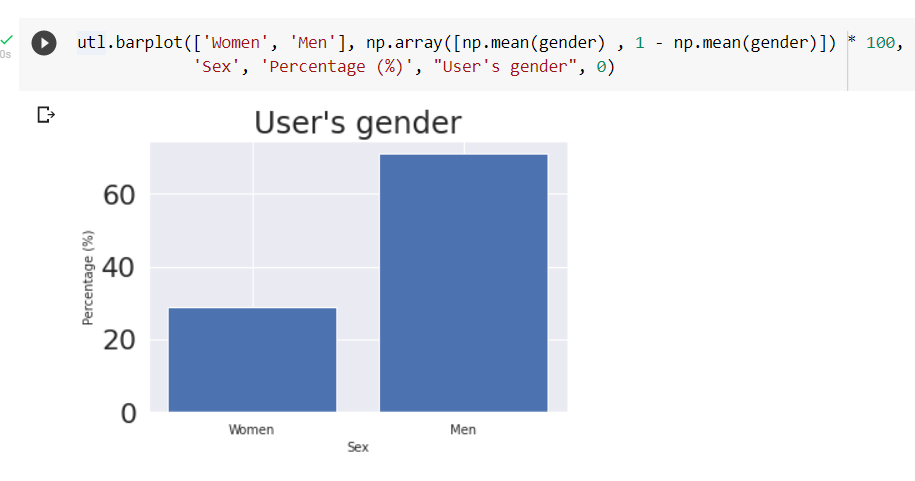
We can look for the age distribution across users using the describe function from pandas which will provide detailed information about mean, standard deviation, minimum and maximum and various percentiles.



From this we can see that the median age is 31 with the youngest viewer being only 7 and the oldest user being 73.

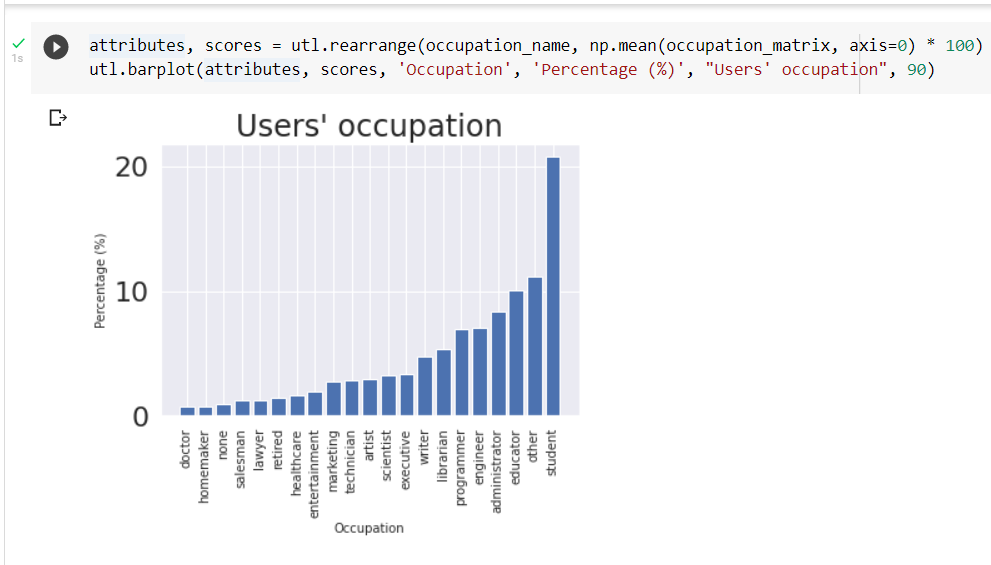
Next, we can look at the gender breakdown in the data set.

For this we can use the bar plot function from the utilities file that we imported. Viewing the barplot we can see that 70% of the users are male and only 30% are female.



Let’s do the same with the occupation feature of the user

Here in this bar plot we can see the percentage of the users that have a given occupation where we can plot the set of occupations on the x-axis and percentage of users on the y-axis.

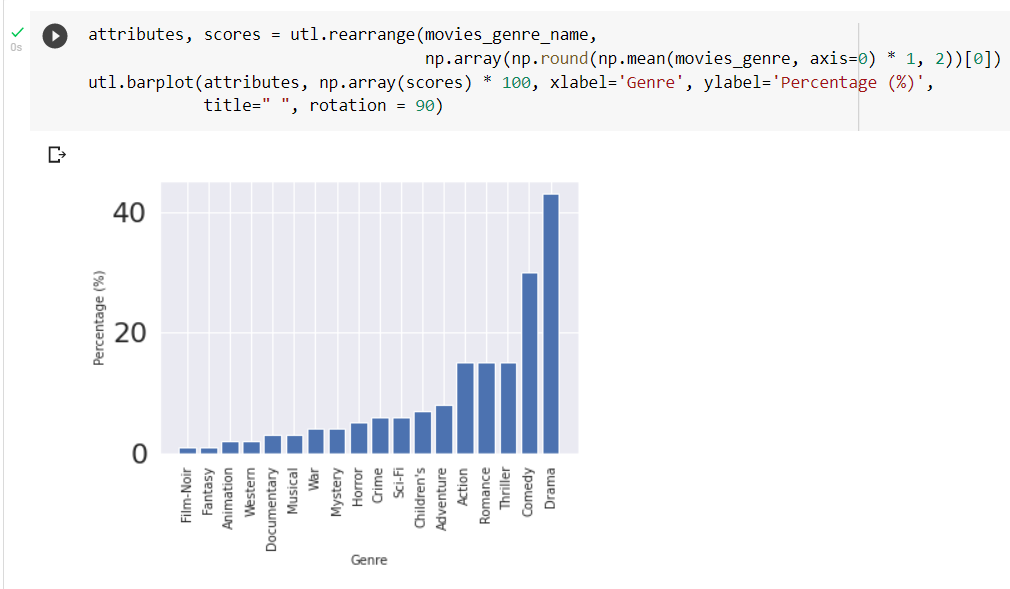


From the plot we can see that the maximum count in percentage are students.

And if we look at the list of genres columns, we indicate 0 for the genre does not hold for that movie and 1 indicates that the genre holds for that movie, we can see here that one movie can have multiple genres.

Let’s look at some statistics about the movies

Using the bar plot, we can see the percentage of movies that belong to each of the genres, where in we can see that the list of genres is plotted on the x-axis and percentage on the y-axis.

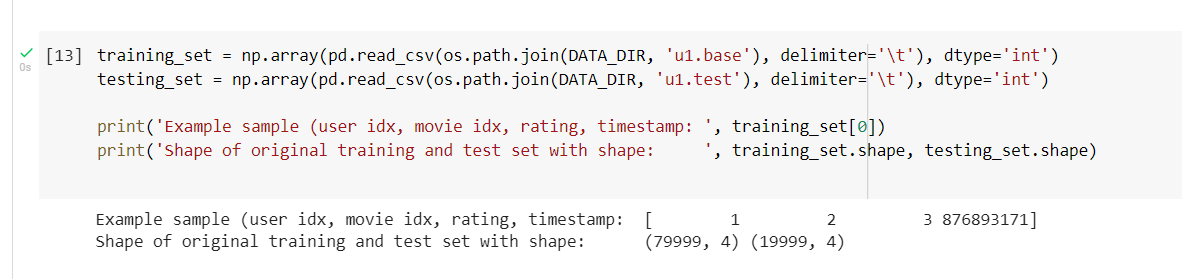


From the bar plot we can figure out that top biggest categories are comedy and drama

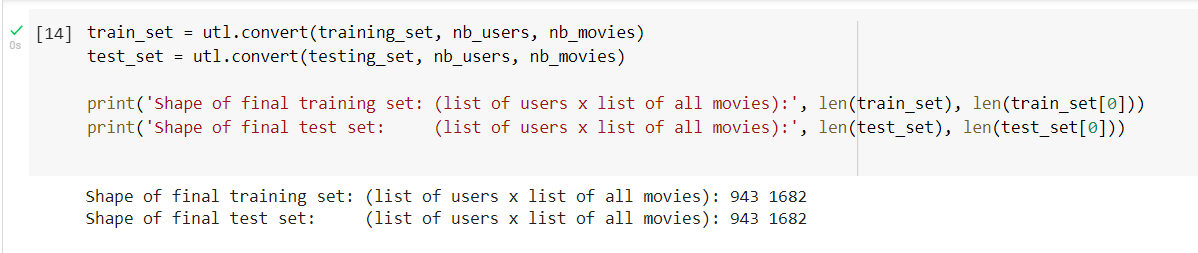
**Construction of Training and Validation Set**

In recommender system it might be to predict a set of items that the user will likely interact with, we could break this further down that the task is to predict when a given item is relevant to a user or not.

In order to build a model to perform this task we need to come up with a matrix that will measure how good our trained model is. If we look for an example of an image classification for cats and dogs’ task this matrix could be the percentage of images that were correctly classified, in short, we can just use the accuracy but for the recommendation task this matrix could be the number of retrieved items that our model was able to predict as relevant.

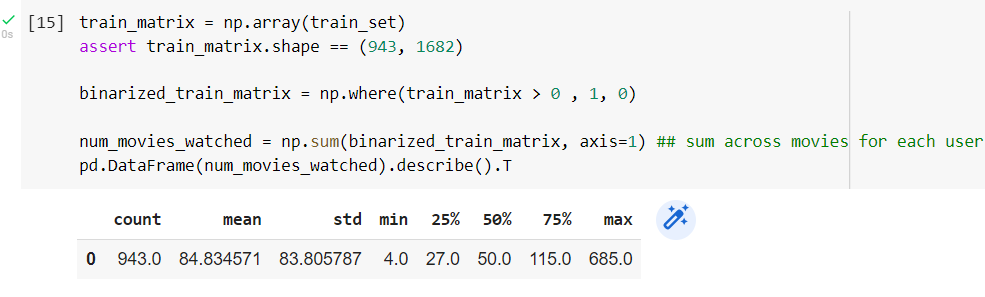


* In fact due to our choice of the recommender model here the data we will use for evaluation is not linked to a new set of users but rather it just contains user item interactions by same set of users. Now these user item interactions are not seen during training, this way we can test our model’s ability to generalize the scenario’s that it has not encountered before.
* AS a result, the data associated with the training and test sets are no longer independent as we assumed which complicate things. So here we will just natively assume that each observation is independent of each other.

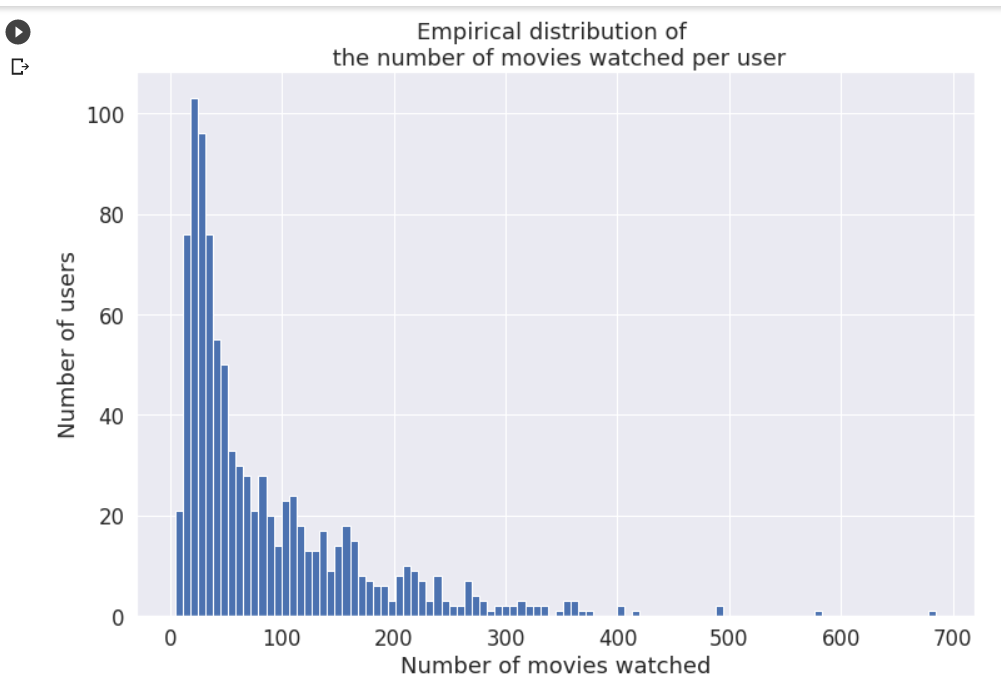


* Our dataset which is provided by MovieLens has already provided us with training and test split, however it did not give us validation set. So to get the validation set we created a split function that will sample 80% of our training set and assign the rest as the validation set.

Later when we train our model on training set we will use this validation set to ensure that our model can generalize to an independent hold out dataset.



* We split our data so that we can measure our model’s ability to generalize to new data points.



**Methodology**

There are many different methods we can use to build a recommender system and in the previous assignment I tried to build a model using ‘Cosine Similarity’ function using ‘Bag of Words’ technique where I created tags for the movies for recommendation but was not enough. So, this time I tried building a model using “**MATRIX FACTORIZATION ALGORITHM**”.

One of the more popular methods used to gauge progress in building a Recommender system is Matrix Factorization.

Matrix factorization algorithm is based on the assumption that a given user’s rating for an item can be estimated by 2 vectors, where one represents the user and another vector that represents an item. These vectors are such that when taking their scalar product, we can estimate the user’s rating for that given item.

To calculate the observed evaluation with respect to that of number of users and movies we use:

Where is the observed evaluation w.r.t number of users ‘U’ and movies ‘I’.

We can also model latent vector in the following format.

These will be our model parameters that we will want to estimate from our data. We will be referring them as latent factors or latent vectors or a latent model.

There are 3 main components in our model’s implementation:

1. Learning Loop-

First, we take a data tuple which will have our training and validation sets, it will take features which will tell us the dimension of our model parameters. We will also consider learning rate(lr), number of iterations which is epochs, weight decay which will be the multiplier on our regularization term and also considering a stopping threshold value.

Now we create 2 latent matrix P and Q for each user and latent vector for all items respectively. Initially both the matrices use random samples drawn from the uniform distribution then we will begin the optimization process where e represents the iteration, we are currently at then we will iterate through each user where we will mark which user we are currently looking at and each item where I will denote which item we are looking at.

To ensure that the model is learning We would like to calculate a loss term both on the training and validation sets. So, we will be using the loss function.

Using the loss function we initialize the following objects loss\_train and loss\_valid, for now these are two empty lists which will contain the loss value from each iteration.

In the model training phase, we do not want any role of test set instead we will calculate the test loss when the model has already been trained.

1. Loss Function-

Here we will calculate how far off our model’s predictions are from the true observations and finally for estimation here we will define the stochastic gradient descent algorithm which will help in estimating the model parameters P and Q that are associated with users and items respectively.

It has to translate various properties of model performance to a single number such that improvements in the number reflects in improvement of the model. Our type of loss function has to be motivated by the type of model output. As in this recommender system we are working with explicit data in the form of user ratings, where they vary from 1 to 5.

The mean squared error is a perfect choice. We can calculate the mean squared error as the average of the squared differences between the predicted and actual values. The results are always positive regardless of the sign of the predicted and actual values.

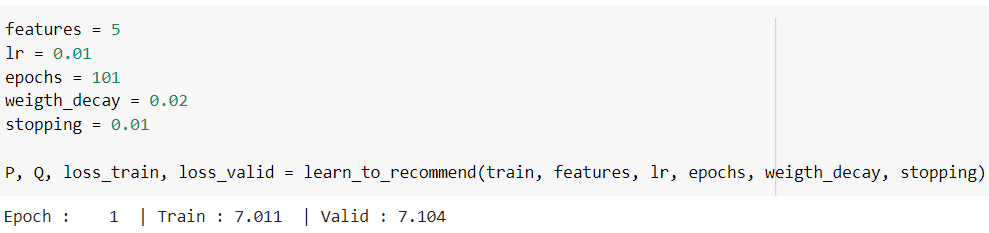
In the current scenario for us the user ratings will be the actual values and the predicted will be the estimated rating.

1. Estimation Process-

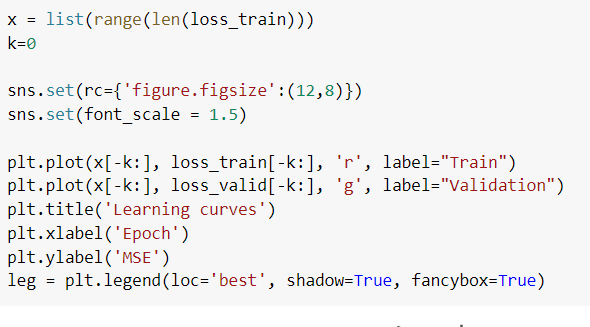
Estimate our model parameters with the help of Stochastic Gradient Descent Algorithm. Stochastic Gradient Descent is an iterative method to minimize our objective function.

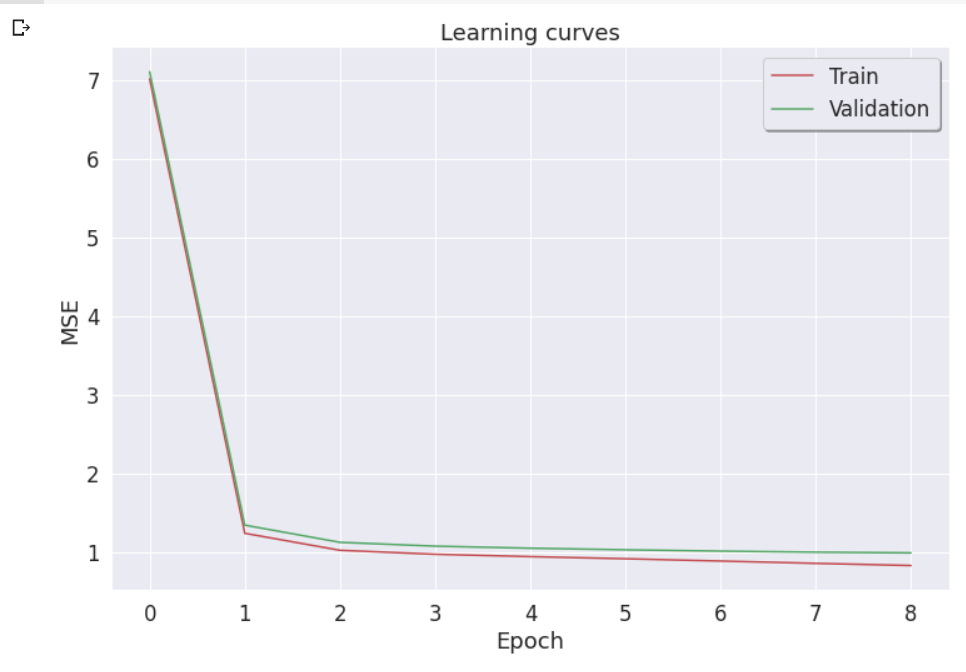
**Training the Model**

As the matrix factorization is done implementing, we can train the model using different parameters. To get the best results we can adjust the parameters.



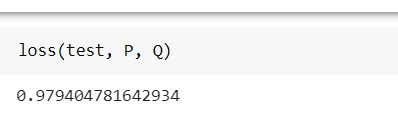
After the iterations we can plot a graph for leaning curve where we can show the training and validation curves.





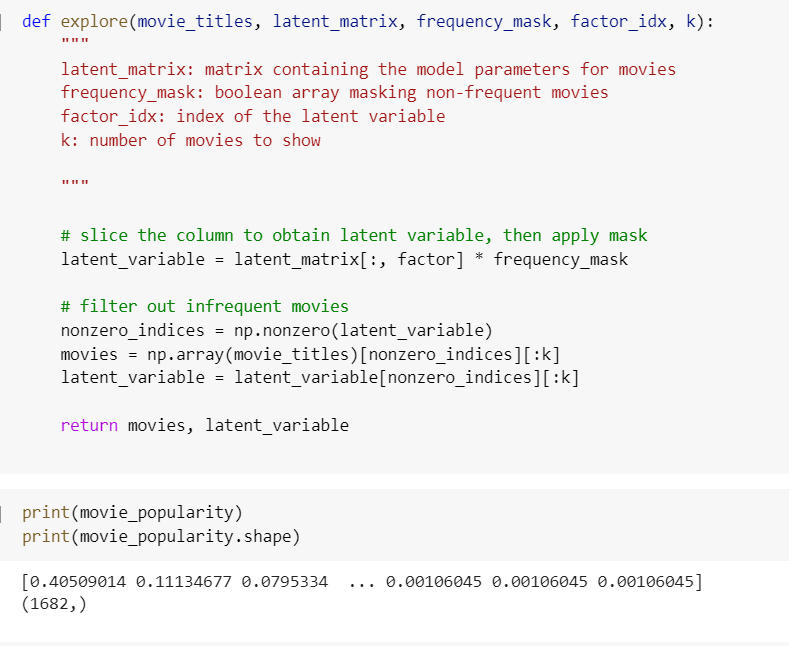
**Evaluation**

After training our model we can evaluate the final performance of the model on the test set with the help of loss function.

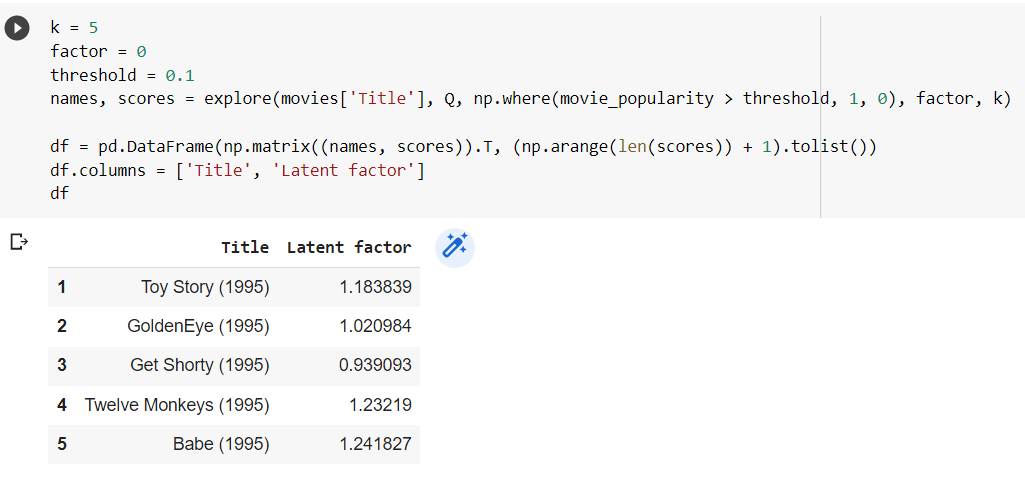


**Experiment**

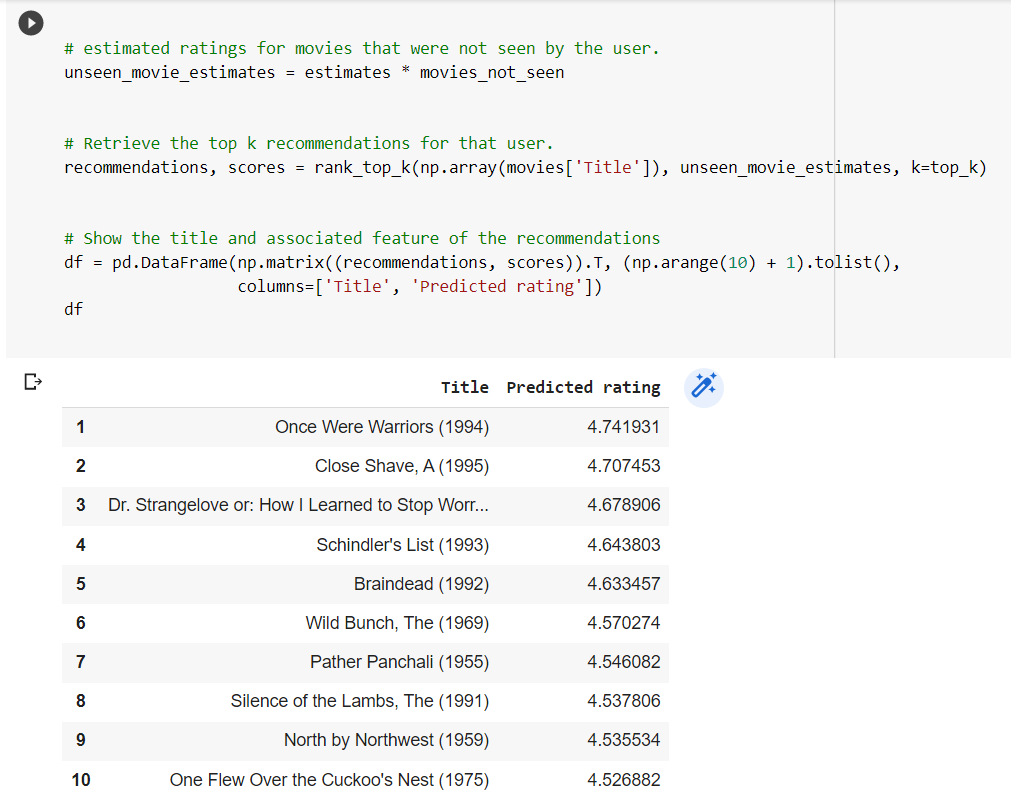
We first defined an explore function with the parameters: movie\_titles, latent\_matrix, frequency\_mask, Factor\_idx and k where k is the number of movies to show.



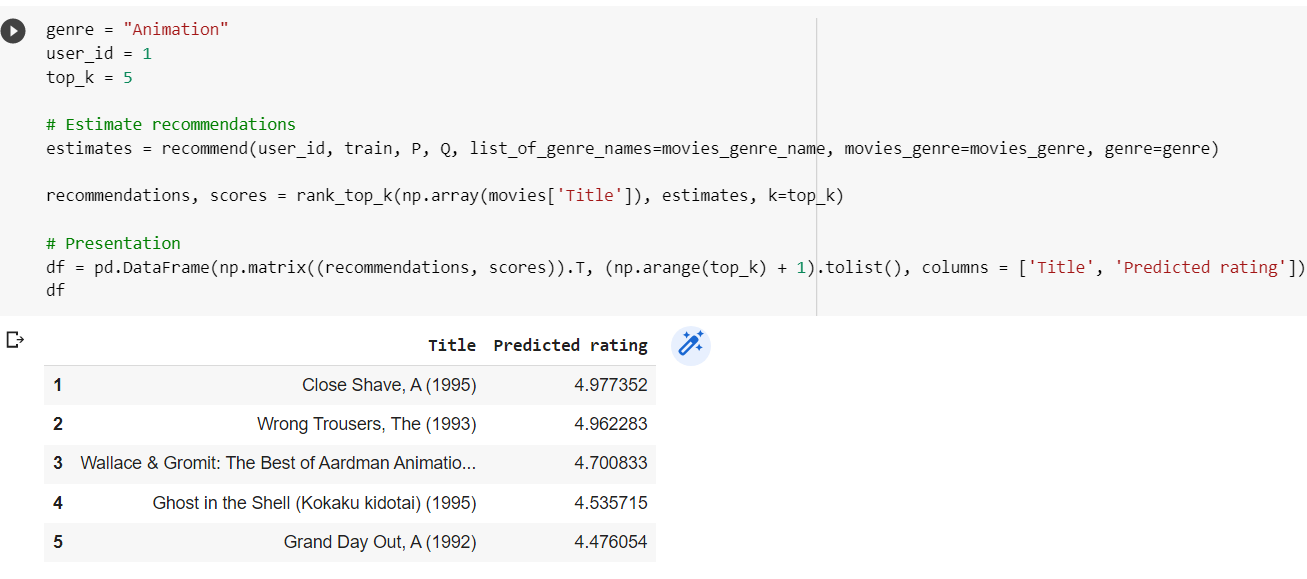
Giving the count of k to be 5 where the model recommends 5 movies with the help of latent factor.



Here with the help of estimation function we are predicting the rating for movies which were not seen by the user.



Similarly with the help of genre we are predicting top 5 movies based on the estimation function.



**CONCLUSION**

To conclude, a recommender system built by using matrix factorization performed well on the test set with the help of the loss function which was minimum and was brought down to 0.97 the model was adjusted for better results and The model helped us in predicting ratings for movies that were not seen by the user.

**References**

1. Lü, L., Medo, M., Yeung, C.H., Zhang, Y.C., Zhang, Z.K. and Zhou, T., 2012. Recommender systems. *Physics reports*, *519*(1), pp.1-49
2. Shi, Y., Larson, M. and Hanjalic, A., 2014. Collaborative filtering beyond the user-item matrix: A survey of the state of the art and future challenges. *ACM Computing Surveys (CSUR)*, *47*(1), pp.1-45.
3. Zamani, H. and Croft, W.B., 2020, January. Learning a joint search and recommendation model from user-item interactions. In *Proceedings of the 13th International Conference on Web Search and Data Mining* (pp. 717-725)
4. Inuzuka, K., Hayashi, T. and Takagi, T., 2016, October. Recommendation system based on prediction of user preference changes. In *2016 IEEE/WIC/ACM International Conference on Web Intelligence (WI)* (pp. 192-199). IEEE
5. Kassak, O., Kompan, M. and Bielikova, M., 2015. User preference modeling by global and individual weights for personalized recommendation. *Acta Polytechnica Hungarica*, *12*(8), pp.27-41
6. Koren, Y., Bell, R. and Volinsky, C., 2009. Matrix factorization techniques for recommender systems. *Computer*, *42*(8), pp.30-37
7. Takács, G., Pilászy, I., Németh, B. and Tikk, D., 2008, December. Investigation of various matrix factorization methods for large recommender systems. In *2008 IEEE International Conference on Data Mining Workshops* (pp. 553-562). IEEE
8. Lee, D. and Seung, H.S., 2000. Algorithms for non-negative matrix factorization. *Advances in neural information processing systems*, *13*
9. Hering, A.S. and Genton, M.G., 2011. Comparing spatial predictions. *Technometrics*, *53*(4), pp.414-425
10. Hochreiter, S., Younger, A.S. and Conwell, P.R., 2001, August. Learning to learn using gradient descent. In *International conference on artificial neural networks* (pp. 87-94). Springer, Berlin, Heidelberg
11. Bottou, L., 1991. Stochastic gradient learning in neural networks. *Proceedings of Neuro-Nımes*, *91*(8), p.12

GITHUB REPOSITORY: <https://github.com/ManinderSingh27/Movie-Recommender-System-using-Matrix-Factorization>