**INFORMATION RETRIEVAL**

**CS F469**

**UB-CF BASED RECOMMENDERSYSTEM**



**Project submitted by:**

Devesh Darshan 2018A5PS0978P

Harshita Gupta 2018A2PS0147P

Hemant Bhartiya 2018A1PS0006P

Ishita Singhal 2018A8PS0349P

Ishanuj Hazarika 2018A2PS0690P

**TABLE OF CONTENTS**

[**ACKNOWLEDGEMENT**](#_o3tunnqdg1cb) **3**

[SURPRISE- The library](#_s2eurzuhxrp5) 4

[Resnick Prediction Formula :](#_p8ics6tdpsre) 4

[Significance of multiplying the value of rv,m by the similarity of user u to user v, in resnick prediction formula :](#_q6l39p9dbtnv) 5

[Evaluation and Testing:](#_k7v27451vtf) 5

[**RESULTS**](#_5n9t69gp5sz6) **5**

[**IMPROVEMENTS**](#_8j0dgftmzs0u) **7**

[**REFERENCES**](#_7skz82eegn09) **12**

# **ACKNOWLEDGEMENT**

*We would like to express our special thanks of gratitude to Prof. Vinti Agarwal Ma’am who provided us with this opportunity to work in this project and get us started with some practical experience in this field. This helped us in doing a lot of research and how to put our knowledge in AI to use in practical use. We are really thankful to them. Secondly we would also like to thank our friends who helped us in this project.*

**METHODOLOGY**

## **SURPRISE- The library**

Surprise is a python library used for building and analyzing recommender systems. This library is the backbone of our model. It provides various built-in prediction algorithms and similarity measures (like Pearson, cosine). It makes the implementation of new algorithm ideas much easier and also provides tools to evaluate, analyse and compare the algorithm’s performance.

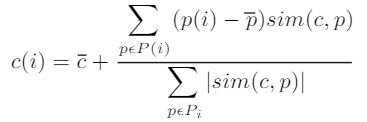
In this library, the utility matrix is not explicitly created.

For building our own prediction model using surprise, we -

* constructed a class derived from ‘[AlgoBase](https://surprise.readthedocs.io/en/stable/algobase.html#surprise.prediction_algorithms.algo_base.AlgoBase)’ that has an ‘estimate’ method. We defined our Resnick prediction formula using this class.
* constructed a function for k-fold cross validation using which we obtained MAE scores for different folds.
* selected a similarity measure (Pearson) and read the input file with the help of the ‘Reader’ object. This object helps us to define the range of our ratings.

## **Resnick Prediction Formula :**

The Resnick prediction formula predicts the ratings in a way that more similar neighbours have greater impact on the final rating.



Here, c(i) is the rating to be predicted for item i for the user c and p(i) is the rating of item i by the neighbour p. And, and are the average ratings for the users c and p respectively. Here, sim(c,p) is the similarity measure between users c and p and is obtained using Pearson's correlation coefficient.

## **Significance of multiplying the value of rv,m by the similarity of user u to user v, in resnick prediction formula :**

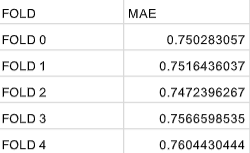
By multiplying the neighbour rating with the similarity coefficient, we ensure that the neighbour which is more closer to the target user, that is , having more similarity coefficient will be weighted more or will have greater impact on the final rating.

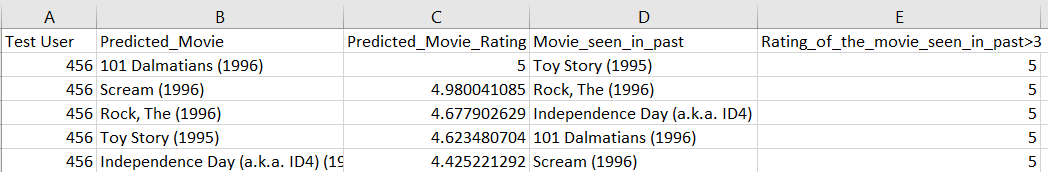
## **Evaluation and Testing:**

For evaluation and testing purposes, we saved the original ratings file from the MovieLens dataset as ori\_ratings.csv, then created test\_user.txt file with 10 randomly selected user ids and removed this user ids from the original ratings file and then saved this new file as ratings.csv which was used for evaluation purposes.

# **RESULTS**

We implemented user based collaborative filtering using the above mentioned algorithm on the input file “ratings.csv”. After fitting the algorithm, for evaluation of the model, we used k-fold (k=5) cross validation scheme and obtained the following mean absolute errors -



Along with this, we also predicted top-5 movies for 10 randomly chosen users which were not the part of the training phase. The obtained results are saved in ‘output.csv’ . **For example**, for user **ID = 456,** this is the obtained result.

**LIMITATIONS**

User based collaborative filtering is easy to implement and have greater diversity but it faces the following challenges -

* **Sparsity**, as the very less percentage of people rate the items
* **Cold-start**, as for making predictions for a new user ,we have almost no information to find out its neighbours for prediction
* More the number of users in our data, more the **computational cost** for finding nearest neighbours.
* High maintenance costs, as it requires more **frequent computation** of user neighbours with the addition of new users.
* More **space** requirements when compared to item based approaches, as generally there are more number of users then the number of items.

# 

# 

# 

# 

# 

# **IMPROVEMENTS**

1. *Removing movies that have been rated by less than 3 users.*

* The probability of a movie being recommended which has been only rated by less than 3 users is very less as it is very less viewed compared to other movies.
* This method significantly reduces the sparsity, resulting in better results as well as less time.

Problem with the approach:

* This method although reduces the sparsity in the data, but by a very crude way of eliminating data points, hence this method does not use the whole information provided to us.

1. *Using cosine similarity on the above obtained dataset*

* One advantage of cosine similarity is its low-complexity, especially for **sparse vectors**: only the non-zero dimensions need to be considered.
* Smaller cosine angle implies greater similarity between the two users.
* In cosine similarity, a subset of a larger data set clusters in the same way it did in the large set, because the data is not centered and normalized again (which is what Pearson would do).

Problem with the approach:

Since it considers only the common ratings between the users, it will not take into account the rating given by a user if it’s unrated by the other and count it as zero.

1. *Using KNN-algo with pearson similarity and item based collaborative filtering*

* In item based collaborative filtering the ratings are predicted using the user’s own rating on neighbouring items.
* Since the ratings are predicted using the rating of users itself, the predicted rating tends to be much more consistent with other ratings of this user.
* Item’s neighbourhood changes much slower in comparison to user’s neighbourhood rating
* Item based methods are more robust to shilling attacks in the recommender system.

Problem with the approach:

* The KNN algorithm for the selection of the similar users takes an additional amount of time while not giving a very satisfactory improvement then the original algorithm.

1. *Using Support Vector Decomposition (SVD) as the predicting Algorithm*

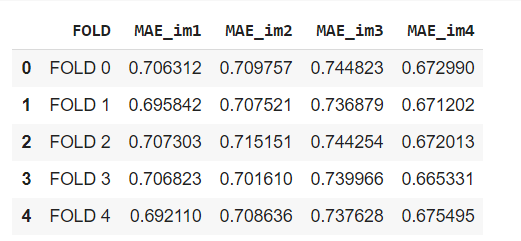
* SVD is a matrix factorisation technique, which reduces the number of features of a dataset by reducing the space dimension from N-dimension to K-dimension (where K<N).
* The SVD decreases the dimension of the utility matrix *A* by extracting its latent factors.

Problem with the approach:

* Any matrix factorisation method doesn’t take similarity of the items or users into account. The results can be greatly improved by taking similarities into account, which can be done using the Graph Neural Network approach.

**MAE Results:**

The MAE results of above mentioned improvements can be seen below:



**Further Improvements:**

Some more improvements can be :

* Matrix Factorization as shown by SVD
* Optimization of user-based collaborative filtering
* Neural Graph based collaborative filtering.

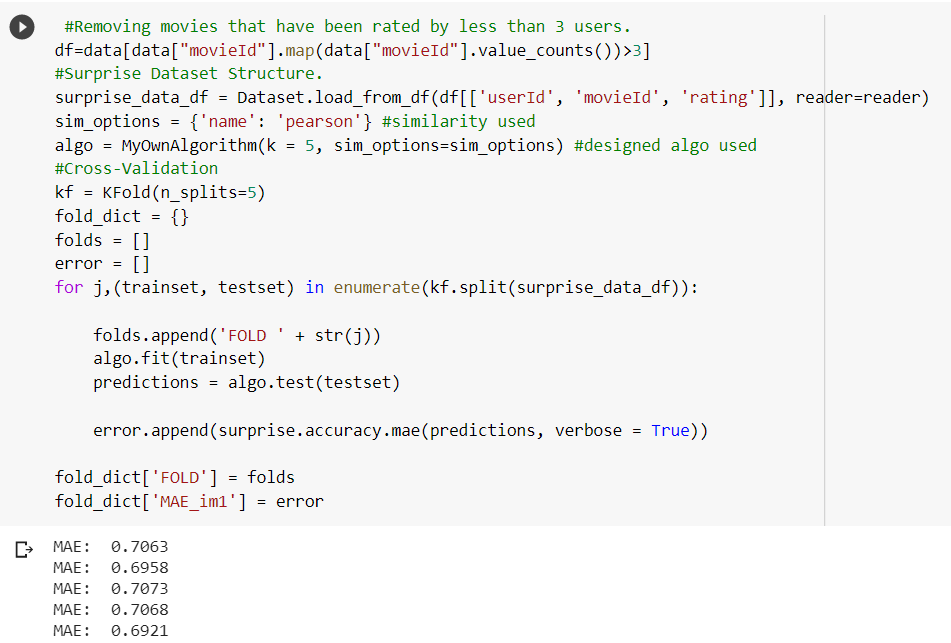
**Content Based Recommender System**

The additional possible information from data to be exploited to improve the existing system developed in part A can be done by a Content Based Recommender system to recommend other items similar to what the user likes, based on their previous actions or explicit feedback.

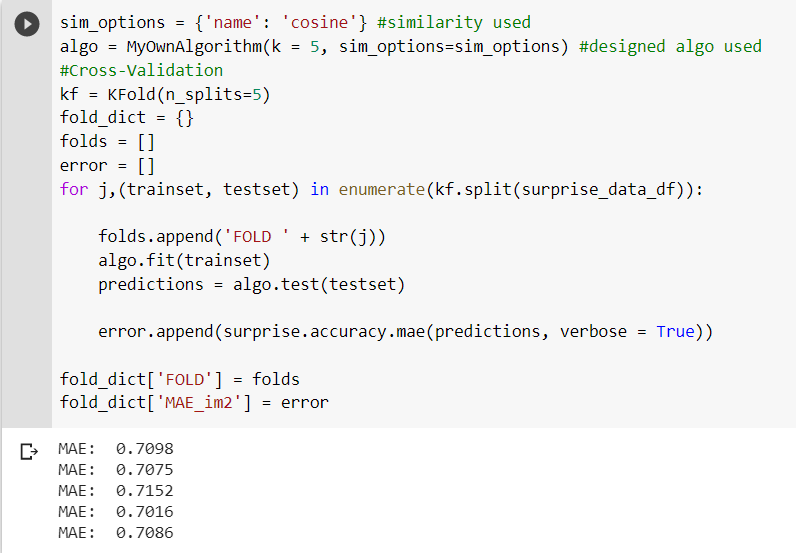
**APPENDIX:**

* **Implementation of above mentioned improvements**

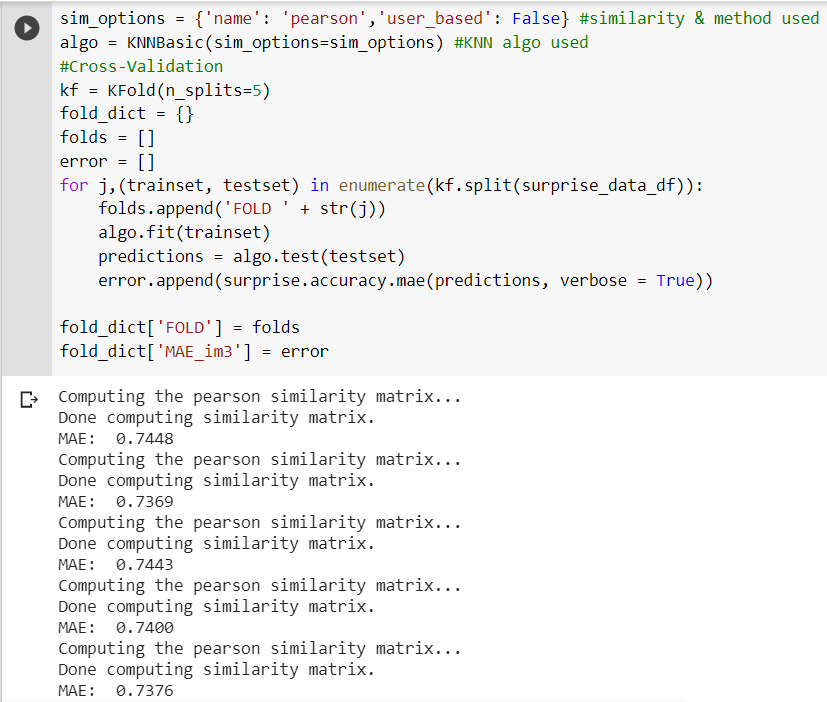
1. *Removing movies that have been rated by less than 3 users.*



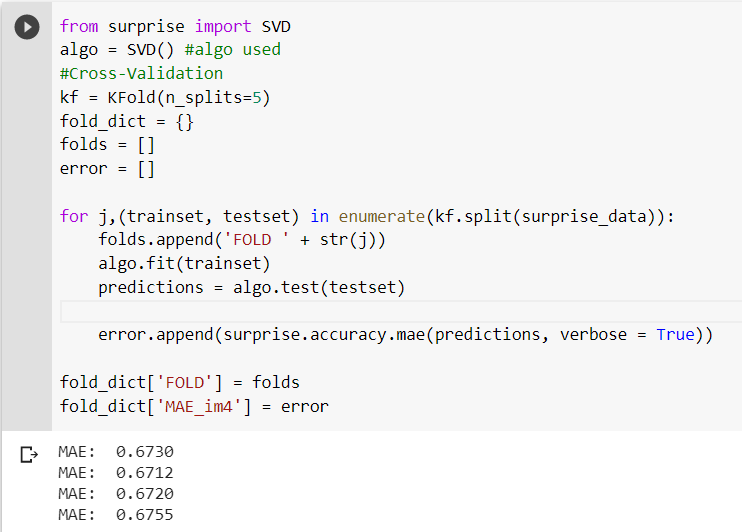
1. *Using cosine similarity on the above obtained dataset*



1. *Using KNN-algo with pearson similarity and item based collaborative filtering*



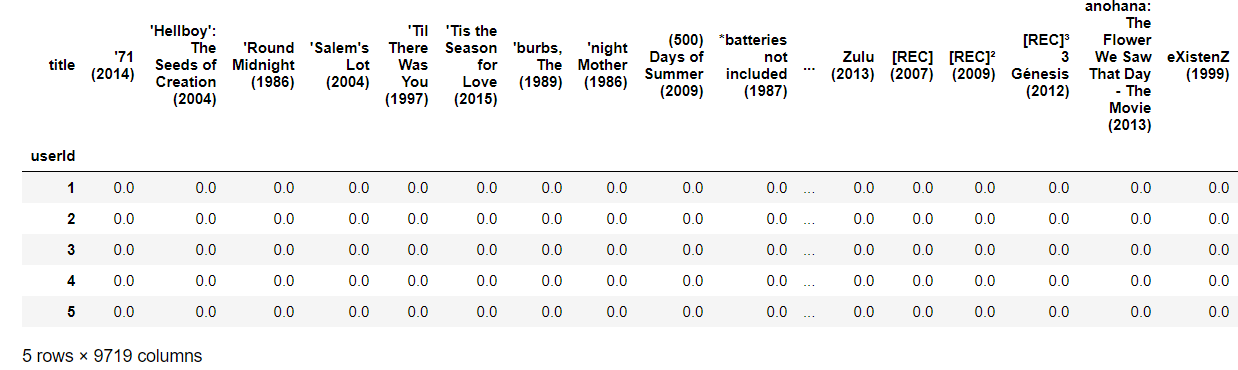
1. *Using Support Vector Decomposition (SVD) as the predicting Algorithm*



* **Utility Matrix:**

Though in our model which uses the Surprise library, utility matrix was not explicitly required for predicting ratings, but for a better visualisation and to have an idea of sparsity in our input data, we obtained the following utility matrix. In this matrix, each row corresponds to different users and each column to different movies. All the movies that a particular user didn’t rate has been filled with a rating of 0.

The utility matrix of the MovieLens data is given below:



# **REFERENCES**

1. Surprise Documentation : <https://surprise.readthedocs.io/en/stable/index.html>
2. Billsus, Daniel, and Michael J. Pazzani. "Learning collaborative information filters." *Icml*. Vol. 98. 1998.
3. Sarwar, Badrul, George Karypis, Joseph Konstan, and John Riedl. "Item-based collaborative filtering recommendation algorithms." In *Proceedings of the 10th international conference on World Wide Web*, pp. 285-295. 2001.