## horizontal line



Project: Movie Rating Prediction

Tien Dang

[horizontal line 0](#_Toc46843950)

[Introduction 2](#_Toc46843951)

[Related Works 2](#_Toc46843952)

[Data 2](#_Toc46843953)

[Dataset description 2](#_Toc46843954)

[Pre-processing 3](#_Toc46843955)

[Data split 3](#_Toc46843956)

[Methodology 3](#_Toc46843957)

[Results 5](#_Toc46843958)

[Discussion 5](#_Toc46843959)

[SVD++ 5](#_Toc46843960)

[Conclusion 6](#_Toc46843961)

[References 8](#_Toc46843962)

# 

# Introduction

Recommendation systems have gained popularity among all levels of commerce top down, this project will focus on movie recommendation systems. Particularly to find what algorithms that are used as engines will provide the highest accuracy for movies ratings of unwatched movies for each user. With additional applications to identify clusters of similar movies or users.

# Related Works

Matrix factorization techniques such as SVD (Single Value Decomposition) have gained popularity for recommendation systems. However, SVD suffers from data sparsity, where a high percentage of variable’s cells are empty or NaN due to missing data, creating imbalanced datasets, skewing prediction accuracy. The article “Matrix Factorization With Rating Completion: An Enhanced SVD Model for Collaborative Filtering Recommender Systems” by Xin Guan, Chang-Tsun Li and Yu Guan, address this issue through four variants of SVD; ESVD, MESVD, IESVD and UESVD (Guan, X., Li, C., & Guan, Y. 2017). For this project, only ESVD will be built upon. ESVD is a density-oriented approach that only allows users who have rated enough movies and movies that has been rated enough to be part of the dataset. This significantly reduces the amount of sparsity in the data, that will increase the accuracy rating of the prediction.

# Data

## Dataset description

The dataset used for movie rating predictions is sourced from [MovieLens] (http://movielens.org), it is suggested that the educational dataset should be used as it is tailored to academic purposes. With a manageable sample size of 100836 ratings, 3683 tags and 9742 movies, created by 610 users between 29th of March, 1996 and 24th of September, 2018.

Movielens academic dataset contains five files; links.csb, movies.csv, ratings.csv, README.text, and tags.csv. For this project, only ratings.csv is used, as only ratings.csv provide the necessary information needed to predict movies ratings. The data inside ratings.csv are separated into four columns; userID, movieID, rating, timestamp.

Where userID is the user's unique identification key, movieID is the movie’s unique identification key, rating is the user’s scoring the movie from 0.5 to 5 and timestamp is the time that the user rated the movie. The educational dataset uses a rating for 0 if a user has not rated a movie, resulting in large amounts of cells containing 0. As there are only 100836 ratings but a total of 9732 movies x 610 users = 5936520 cells for ratings, leaving 5835684 cells containing 0.

### Pre-processing

From the dataset, the ratings.csv file was read using the pandas read\_csv function and turned into a matrix. The data was further cleaned by removing the timestamp column.

### Data split

The data was split with a ratio of 75% training and 25% testing using the surprise train\_test\_split function returning training and testing dataset that was the original dataset. Cross validation was also used with 4 fold to determine what algorithm was more accurate.

# Methodology

Environment: Windows 10 64-bit, 32.0 GiB memory, Intel Core i7-6700k CPU @ 4.00 GHz x 4

The methodology was adopted from Xin Guan, Chang-Tsun Li and Yu Guan, article that suggests using preprocessing data to reduce data sparsity. Where only users who rated enough items and items that have been rated enough times are included in the data (Guan, X., Li, C., & Guan, Y. 2017), to reduce the number of zeros in the dataset sourced from Movielens.

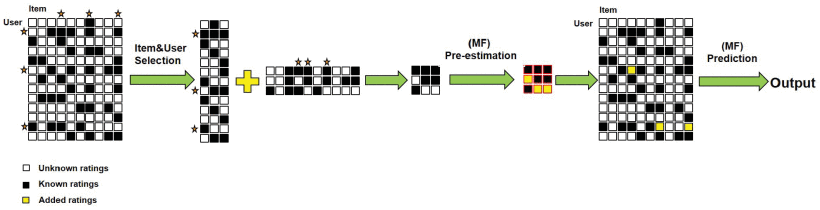
Regularized SVD shown below that was proposed by Simon Funk (Guan, X., Li, C., & Guan, Y. 2017) is then used to calculate the predicted ratings for the condensed dataset.

Regularized SVD: Let **p**u denotes the rating vector for user **u** to all movies and let and **p**i denotes the rating vector by all users to movie **i**.



The diagram below illustrates the methodology used in the article, with the preprocessing of data using the density orientated approach, then using regularized SVD to predict the ratings and then outputting them.

Figure 1: This figure shows the theoretical preprocessing of data and matrix factorisation. Taken from Guan, X., Li, C., & Guan, Y (2017).



To extend the adopted methodology, different matrix factorization based algorithms were used in conjunction with regularized SVD.

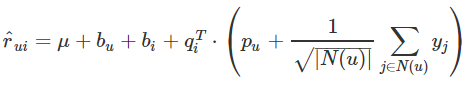
The extended method first starts the same by processing the cleaned data from ratings.cvs through a density orientated approach, limiting only users who rated at least 50 movies and movies with at least 50 ratings being included into the data.

With the data processed, three different matrix factorization based algorithms; bias SVD, regularized SVD and SVD++ are used.

Bias SVD: Let **p**u denotes the rating vector for user **u** to all movies and let and **p**i denotes the rating vector by all users to movie **i**. Let **μ,** **Bu** and **Bi** denote overall average rating of training data, user bias and movie bias respectively (Yancheng J, Changhua Z, Qinghua L, & Peng W. 2014).



SVD++: Let **p**u denotes the rating vector for user **u** to all movies and let and **p**i denotes the rating vector by all users to movie **i**. Let **μ,** **Bu** and **Bi** denote overall average rating of training data, user bias and movie bias respectively. **N(u)** is the set of implicit feedback and **yj** is the dimensions vector (Yancheng J, Changhua Z, Qinghua L, & Peng W. 2014)..



The results of the three algorithms are then cross validated using three folds and the accuracies are returned as root mean square error (RMSE).

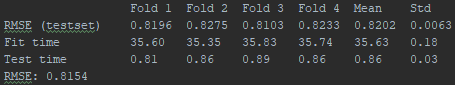
# 

# Results

After preprocessing the data to ensure data density, reducing data sparsity related errors, three algorithms: SVD, SVD++ and Baseline Only were used for cross validation using three folds and then averaged. The results indicated that SVD++ with a rmse (root mean square error) of 0.8257599 was the lowest, hence is the most accurate out of the three algorithms.

|  |  |  |  |
| --- | --- | --- | --- |
| Algorithm | test\_rmse | fit\_time | test\_time |
| SVD++ | 0.825754599 | 32.28197885 | 1.60371693 |
| Bias SVD | 0.837746726 | 1.384277185 | 0.10062782 |
| Regularised SVD | 0.843273 | 1.102183 | 0.058178 |

SVD++ was cross validated again using 4 folds, returning a similar average of the four folds. Train test split was performed with 25% used as the test set size, with SVD++ being trained with the other 75%. The classifier was then used to compare the predicted values with the test set, then calculated the rmse of the predictions to be 0.8154



# Discussion

## SVD++

SVD++ performed significantly better then regularised SVD and bias SVD, this is due to the implicit feedback being factored into what is essentially bais SVD. Although SVD++ returns the most accurate results, it does not connote that regularised SVD and bias SVD are not suitable. Rather it is that implicit data within the dataset reducing the accuracy of the algorithms.

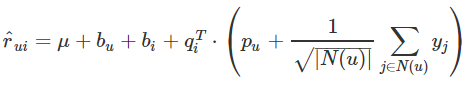
Implicit data main features can be broken down into three categories:

1. No negative feedback: As the dataset sets ratings for a movie as 0 if there is no rating, there is no way to differentiate from a user rating a movie as 0 or missing data (Narapareddy, A. 2019). Creating only tangible positive feedback, while the 0s can only be seen as missing data.
2. Inherently noisy data: The dataset for movies is inherently noisy, usually a large volume of noise would assist in creating a robust system (Narapareddy, A. 2019). However, in the case of this datasets case, the dataset is relatively small.
3. Preference vs frequency: In the cleaned ratings.cvs dataset the ratings are portrayed as users preference in numerical form. However while observing for a holistic perspective the ratings may refer to the frequency of ratings rather than the magnitude of a user's preferences (Narapareddy, A. 2019).

The issues one and two raised from the implicit data are mitigated through the preprocessing using a density orientated approach. However, issue three was not covered in the related works as regularised SVD does not consider user bias.

Bias SVD does consider bias and is shown in the formula as **Bu** and **Bi** for each user and movie, this is reflected by the lower RMSE, indicating higher accuracy. However, it does not handle the issue of distinguishing between frequency and preference.

SVD++ handles the issue distinguish between preference and frequency, by integrating implicit feedback as shown in the formula denoted as **N(u)** for the set of implicit feedback and **yj** for the dimensions vector. This is reflected by the lower RMSE, indicating higher accuracy, as SVD++ implements a global cost function while improving user to user factorized model (Koren, Y. 2010). Allowing it to predict more accurately as the global cost function can distinguish between frequency and preferences better than the heuristic movie to movie similarities.



## 

# Conclusion

The use of machine learning for the development of recommendation systems is a rapidly growing industry. The current standard in this industry is the use of collaborative filtering. Within this method of recommendation, matrix factorisation is currently the gold standard. With this in mind, a method by implementing SVD as alternative methods of factorisation was used. As we can see from the results listed in the results section. The SVD++ approach was the best method in root mean square error though the time of execution was higher it was still within reasonable boundaries. This result lends weight to the adoption of matrix factorisation as the industry standard of recommendation systems.

# References

Guan, X., Li, C., & Guan, Y. (2017). Matrix Factorization With Rating Completion: An Enhanced SVD Model for Collaborative Filtering Recommender Systems. *IEEE Access*, *5*, 27668–27678. <https://doi.org/10.1109/ACCESS.2017.2772226>

Yancheng Jia, Changhua Zhang, Qinghua Lu, & Peng Wang. (2014). Users’ brands preference based on SVD++ in recommender systems. In *2014 IEEE Workshop on Advanced Research and Technology in Industry Applications (WARTIA)* (pp. 1175–1178). IEEE. <https://doi.org/10.1109/WARTIA.2014.6976489>

Koren, Y. (2010). Factor in the neighbors: Scalable and accurate collaborative filtering. *ACM Transactions on Knowledge Discovery from Data (TKDD)*, *4*(1), 1–24. <https://doi.org/10.1145/1644873.1644874>

Narapareddy, A. (2019). Recommender system using Bayesian personalized ranking. <https://towardsdatascience.com/recommender-system-using-bayesian-personalized-ranking-d30e98bba0b9>

Xiangnan He, Lizi Liao, Hanwang Zhang, Liqiang Nie, Xia Hu and Tat-Seng Chua (2017). Neural Collaborative Filtering (pp. 3 - 4).ACM. <https://arxiv.org/pdf/1708.05031.pdf>

sklearn.model\_selection.cross\_val\_score — scikit-learn 0.21.2 documentation. (2019). Retrieved from <https://scikit-learn.org/stable/modules/generated/sklearn.model_selection.cross_val_score.html>