

Group 9: Recommendation Systems

First Name	Last Name
Saahil	Sofat
Shushmitha	Anthay Suthakaran

Table of Contents

1. Introduction	2
2. Data	3
3. Plans	3
4. Algorithms Implemented	3
5. Evaluations and Results	4
5.1. Results and Findings	6
6. Conclusions and Future Work	8
6.1. Conclusions	8
6.2. Limitations	9
6.3. Potential Improvements or Future Work	9
7. Appendix	9

1. Introduction

Recommendation system is an information filtering technique which provides users with information that the user may be interested in. In this day and age of technology, we have an information overload and it is very difficult to be able to decide or choose something that the user may want due to the amount of information. Recommendation systems help the user by suggesting items/services which may be closely aligned to the user's interest or aligned to the interest of similar user's.

Some of the basic functions of a recommendation system is as follows:

1. Predict how much you may like a certain product/service
2. Compose a list of **N** best items for you
3. Compose a list of **N** best users for a certain product/service
4. Explain to you why these items are recommended to you
5. Adjust the prediction and recommendation based on your feedback and other people

The engine helps in predicting how a user may rate a certain item, thereby helping the uses to avoid going through the massive inventory and suggest a product on the basis of the predicted ratings.

Types of Recommendation

1. Personalized - This system recommends items based on the user's past preference and history of purchases.
2. Non-Personalized - When the systems do not have any information about a user, but still recommends certain items based on the popularity or collective preferences in the system.

Algorithms and Techniques

- Content-Based Recommendation Algorithms:
 - The user will be recommended items similar to the ones the user preferred in the past.
 - Keywords are used to describe the items and a user profile is built to indicate the type of item this user likes.
 - Various user's items are compared with items previously rated by the user and the best-matching items are recommended.
- Collaborative Filtering Based Recommendation Algorithms:
 - The user will be recommended items that people with similar tastes and preferences liked in the past.
 - It's based on collecting and analyzing a large amount of information on users' behaviors, activities or preferences and predicting what users will like based on their similarity to other users.
- Hybrid Recommendation Algorithms:

- Combine content-based and collaborative filtering based algorithms to produce item recommendations.

2. Data

The data has been taken from the GroupLens (<http://grouplens.org/about/what-is-grouplens/>). Grouplens is a research lab in the department of Computer Science and engineering at the university of Minnesota that specialize in recommender systems.

Data set contains ratings from 700 users applied to 9,000 movies with around 100,000 ratings. The data has been divided into training and test with 80:20 ratio. We have used the training data to build the model and then used the model to predict the ratings.

3. Plans

- Our aim was to implement various recommender algorithms and techniques and be able to predict the ratings for items that have not yet been rated by users.
- To compare these algorithms and find which one has the least error RMSE (Root Mean Square Error), MAE (Mean Absolute Error).
- The data has been divided into training and test with 80:20 ratio.
- We have used the training data to build the model and then used the model to predict the ratings.

4. Algorithms Implemented

We are implementing collaborative filtering for our data

- Collaborative filtering comes from the idea that people often get the best recommendations from someone with tastes similar to themselves.
- Social filtering i.e. filters information by using the recommendations of other people.
- The recommendations of some friends who have similar interests are trusted more than recommendations from others.
- This information is used in the decision on which movie to see.
- Collaborative filtering algorithms often require
 - (1) users' active participation,
 - (2) an easy way to represent users' interests, and
 - (3) algorithms that are able to match people with similar interests.

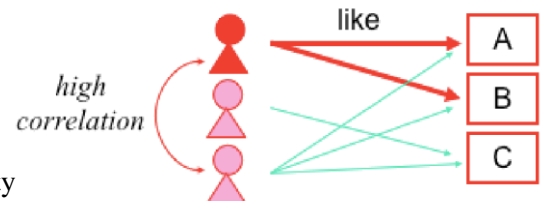
Algorithms Used –

- User Based Collaborative Filtering:
 - User-based collaborative filtering predicts a test user's interest in a test item based on rating information from similar user.
- Item Based Collaborative Filtering:
 - Item-based collaborative filtering predicts a test user's interest in a test item based on rating information from similar item.
- Item Based Collaborative Filtering with Genre:
 - We use the tf.idf (weight) of the genre in the movie and compare it with others to form a matrix.
 - We then use this matrix with similarity to predict ratings.
- Slope one:
 - It is rating-based collaborative algorithm.
 - Predicts how a user would rate a given item from other users ratings.
- Matrix Factorization:
 - It is an effective technique as they allow us to discover the latent features underlying the interactions between users and items.
 - It is simply a mathematical tool for playing around with matrices, and is therefore applicable in many scenarios where one would like to find out something hidden under the data.

5. Evaluations and Results

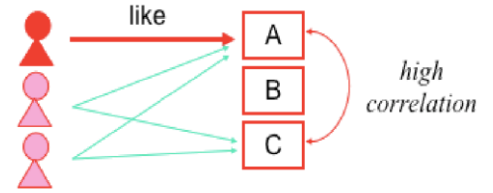
1. User Based Collaborative Filtering

- ◆ User-Based Collaborative Filtering Model
 1. Create user-item rating matrix
 2. Make user-to-user correlations
 3. Find highly correlated users
 4. Recommend items preferred by those users
- ◆ Measures used to build the model are
 - ◆ Similarity Measure Used – Cosine Similarity
 - ◆ Normalization Measure Used – Z-Score
 - ◆ K-Nearest Users considered – 5



2. Item Based Collaborative Filtering

1. Use user-item ratings matrix
2. Make item-to-item correlations
3. Find items that are highly correlated
4. Recommend items with highest correlation



we have used Cosine similarity and selected n as 10 based on cross-validation.

3. Item Based Collaborative Filtering with Genre

- ◆ We have tokenized the genre column of the ratings data in such a way that all the genre's are separated into different columns.
- ◆ Once we have the tokenized the matrix, we have featured the data. Each row will contain a csr_matrix of shape (1, num_features). Each entry in this matrix will contain the tf-idf value of the term.
- ◆ Compute the cosine similarity between two 1-d csr_matrices. Each matrix represents the tf-idf feature vector of a movie.
- ◆ Then we make predictions on the test data with the help of the similarity matrix.
- ◆ Compute the RMSE value using the predicted and original values of the test data set.

4. Slope one

- ◆ Slope One is used for ease in implementation and to improve performance. Instead of using linear regression from one items rating $f(x)=ax+b$ to another items rating. It uses a simpler form or regression with a single free parameter $f(x)=x+b$.
- ◆ The free parameter is the average of difference between the two item's ratings.
- ◆ Slope One method operates with average differences of ratings between each item and makes predictions based on their weighted value.
- ◆ Accuracy of this algorithm equals to the accuracy of more complicated and resource-intensive algorithms.
- ◆ Slope one process:
 - Split data into train and test sets
 - Ratings normalization
 - Building Slope One model
 - Making predictions for test set
 - Calculating RMSE on test set

5. Matrix factorization

- ◆ Approximate the matrix $R(m \times n)$ by the product of two matrixes of lower dimension i.e $P(k \times m)$ and $Q(k \times n)$. $R \sim P \times Q = R'$
- ◆ Matrix P represents latent factors of users. So, each k -elements column of matrix P represents each user. Each k -elements column of matrix Q represents each item.
- ◆ So, to find rating for item i by user u we simply need to compute two vectors: $P[:,u]' \times Q[:,i]$.
- ◆ Measures used to build the model are
 - Regularization cost for user factors with default value 0
 - Regularization cost for item factors with default value 0

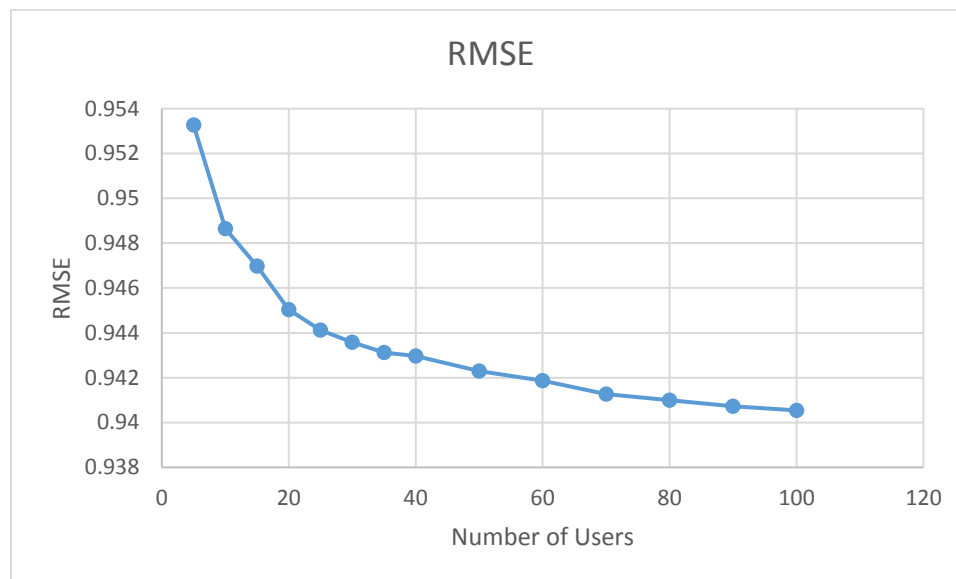
- Learning rate with default value 0.05.
- Number of folds in cross validation 10
- Number of iterations 50
- Number of latent factors – 3,4,5,6,7,8,9,10,11,14

5.1. Results and Findings

1. User Based Collaborative Filtering

Ideal value for n is 100 as we have the least RMSE value.

Running this model, we get the best Predictions for the test dataset.



```
> as(recommended.items.u1, "matrix")[,1:5]
      1      2      3      4      5
1 2.599162 2.490190 2.560329 2.545718 2.545307
2 3.587752 3.490515 3.453816 3.473164 3.441343
3 3.611455 3.544182 3.559155 3.568627 3.533992
4 4.560497 4.254019 4.263488 4.286930 4.231774
5 3.955590 3.930707      NA 3.880849 3.878823
. |
> affinity.matrix
671 x 9066 rating matrix of class 'realRatingMatrix' with 100004 ratings.
> Rec.ubcf
Recommender of type 'UBCF' for 'realRatingMatrix'
learned using 536 users.
> p.ubcf
135 x 9066 rating matrix of class 'realRatingMatrix' with 1221210 ratings.
> error.ubcf
      RMSE      MSE      MAE
0.9572412 0.9163107 0.7431505
. |
```

RMSE: 0.9572

2. Item Based Collaborative Filtering

RMSE: 1.008

3. Item Based Collaborative Filtering with Genre

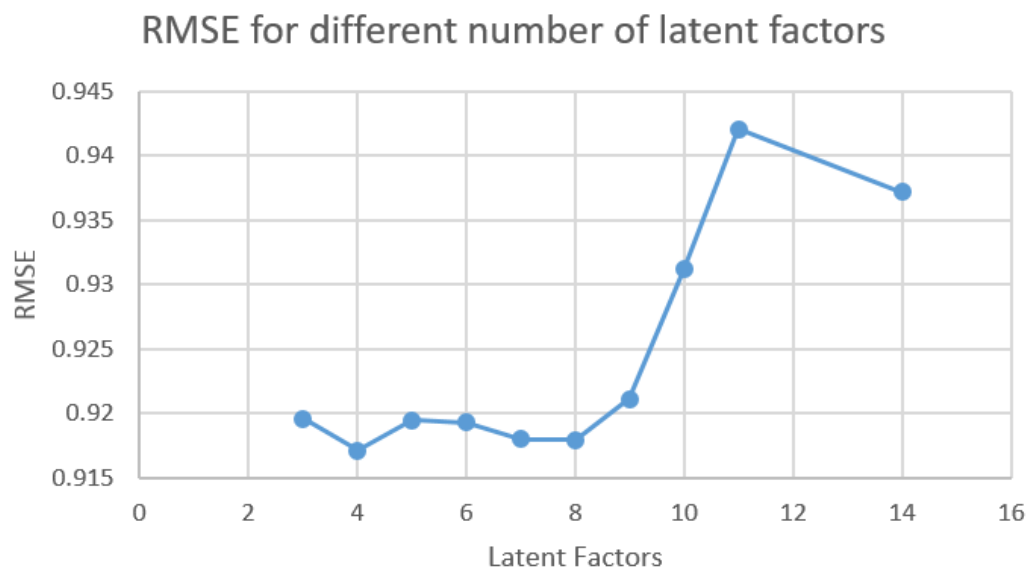
RMSE: 0.943

4. Slope one

RMSE: 1.4282

5. Matrix factorization

- ◆ We observe that the RMSE values between 3 to 8 are in the same range.
- ◆ However, the lowest RMSE value is for latent factor 4 and the value is 0.9171.



6. Conclusions and Future Work

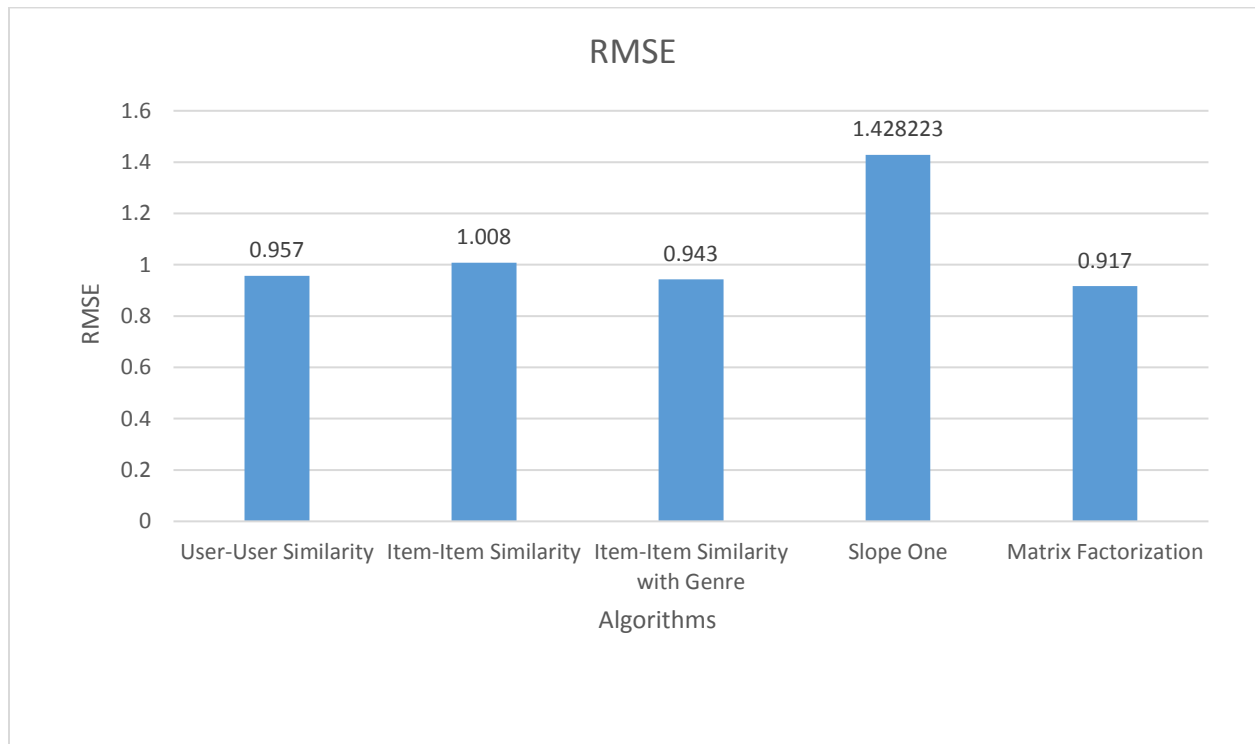
6.1. Conclusions

After implementing the algorithms that we have talked about above, we observe the findings below. We see that the RMSE (Root mean square error) using different algorithms is different. The RMSE is listed below.

According to our findings Matrix Factorization has the least RMSE value when compared to all other algorithms and that shows that it is in this case the best algorithm for recommender systems for our dataset.

After Matrix Factorization, the next best algorithm with the least RMSE value is Item-Item Similarity with

Genre. This is another version of item-item similarity wherein we are also considering genre as a part of the item's characteristics.



6.2. Limitations

Our dataset contained around 100,000 records which gave us a brief introduction of how the results look like. Since the dataset was comparatively smaller than the real world scenarios, we may not get a very accurate picture of the algorithm implementation.

Also the recommender systems can be implemented using so many more algorithms depending on the data, features and how in-depth analysis we require. We were able to produce these results by using some of the most basic and well researched algorithms in the market.

6.3. Potential Improvements or Future Work

Recommendation systems are a very vast field and can be implemented using a variety of algorithms. Since we were limited with time and resources, we were able to implement only some of the popular algorithms.

In addition to the work we have done so far, we would want to be able to implement more context aware algorithms which are listed below –

1. Factorization Machines
2. Tensor Factorization

We want to be able to use some more information about the items such as what was the age of the person watching this movie, how do they generally react to movies of different genre's or do they rate movies differently when watching or accompanied with certain people.

These context aware ratings and predictions would help us making a better and more sophisticated recommendation system.

7. Appendix

Step to implement the algorithms

1. Load the data.
2. Split the data into Train (80%) and test (20%).
3. Build the model using the train data
4. Change the parameter of the algorithm in such a way that we have the optimum model and the least error.
5. Once you have the model, you run the same on the test data set to get the prediction. The validity of the model can be checked using the error. The error can be calculated by contrasting the actual and the predicted test values.








Note: Code for all the algorithms and splitting the data are attached below.



train-final.csv



Test-final.csv

 ItemBasedWithGenre.py  user-userCF.R  IBCF.R  slopeone.R  MF1.R  trainset.txt  testset.txt