Bayesian Networks and Probabilistic Matrix Factorization for Movie Recommendation on MovieLens dataset

1. Objective

- The recommender system aims to reduce errors arising in Collaborative Filtering (CF) due to high sparsity of ratings matrix
- The recommendations thus produced enhance the user engagement in the rating process by improving the quality of recommendations
- Further, our intention is to build a novel recommendation system that can differentiate us from the competition. We intend to achieve this using Bayesian Networks (BN) for content based recommendations.
- We also explore the use of Probabilistic Matrix Factorization (PMF) recommendation system for improving recommendation accuracy on sparse dataset

2. Dataset

This dataset (ml-latest-small) describes 5-star rating and free-text tagging activity from MovieLens, a movie recommendation service. It contains 100004 ratings and 1296 tag applications across 9125 movies. These data were created by 671 users between January 09, 1995 and October 16, 2016. This dataset was generated on October 17, 2016.

Users were selected at random for inclusion. All selected users had rated at least 20 movies. No demographic information is included. Each user is represented by an id, and no other information is provided.

The data are contained in the files links.csv, movies.csv, ratings.csv and tags.csv.

<u>UserIds:</u> MovieLens users were selected at random for inclusion. Their ids have been anonymized. User ids are consistent between ratings.csv and tags.csv (i.e., the same id refers to the same user across the two files).

<u>MovieIds:</u> Only movies with at least one rating or tag are included in the dataset. These movie ids are consistent with those used on the MovieLens web site (e.g., id 1 corresponds to the URL https://movielens.org/movies/1). Movie ids are consistent between ratings.csv, tags.csv, movies.csv, and links.csv (i.e., the same id refers to the same movie across these four data files).

<u>Ratings Dataset</u>:
All ratings are contained in the file ratings.csv. Each line of this file after the header row represents one rating of one movie by one user, and has the following format:
userId, movieId, rating, timestamp
The lines within this file are ordered first by userId, then, within user, by movieId.
Ratings are made on a 5-star scale, with half-star increments (0.5 stars - 5.0 stars).

Bayesian Networks and Probabilistic Matrix Factorization for Movie Recommendation on MovieLens dataset

Tags Dataset:
All tags are contained in the file tags.csv. Each line of this file after the header row represents one tag applied to one movie by one user, and has the following format:
userId, movieId, rating, timestamp
The lines within this file are ordered first by userId, then, within user, by movieId.
Tags are user-generated metadata about movies. Each tag is typically a single word or short phrase. The meaning, value, and purpose of a particular tag is determined by each user.
Timestamps represent seconds since midnight Coordinated Universal Time (UTC) of January 1, 1970.

Movies Dataset:
Movie information is contained in the file movies.csv. Each line of this file after the header row represents one movie, and has the following format:
movieId, title, genres
Movie titles are entered manually or imported from https://www.themoviedb.org/, and include the year of release in parentheses. Errors and inconsistencies may exist in these titles.

Links Dataset:
Identifiers that can be used to link to other sources of movie data are contained in the file links.csv. Each line of this file after the header row represents one movie, and has the following format:
movieId, imdbId, tmdbId

3. Steps followed:

- Scrape data from Imdb and Tmdb databases
- Create training and testing item user matrices
- Build preliminary Bayesian model using Libpgm for initial data exploration
- Use the item feature matrix for building the Bayesian Network recommendation system
- Implement Probabilistic Matrix Factorization (PMF) recommendation system for improving the runtime and accuracy

Bayesian Networks and Probabilistic Matrix Factorization for Movie Recommendation on MovieLens dataset

4. Description:

Web Scraping:

While we finalized on Bayesian networks model to deal with the problem of hybrid recommendation by combining content-based and collaborative features, we need item-feature matrix as input for the model. The MovieLens database considered has very limited data on item features i.e. for every movie the database has only movie_title and genres. For a robust model, more features are required to describe characteristics of the movie. To address this problem, we have extracted more features from IMDb and The Movie Database (TMDb).

Description of the sources:

IMDb, formerly known as Internet Movie Database, is an online database of information related to films, television programs and video games, including cast, production crew, fictional characters, biographies, plot summaries, trivia and reviews, operated by IMDb.com, Inc., a subsidiary of Amazon. As of December 2017, IMDb has approximately 4.7 million titles (including episodes), 8.3 million personalities in its database, as well as 83 million registered users.

The Movie Database (TMDb) is a collaborative database about movies. The database contained 234,000 films and 583,000 personalities at the beginning of 2015.

Using these two databases, we have pulled all the features as given in Table 1.

Feature Extracted	Description	Source
TMDB_SCORE	User rating score	TMDb
IMDB_SCORE	User rating score	IMDb
TOTAL_RATINGS	Number of users provided score	IMDb
FEATURED CREW	Top crew who worked for the movie (Eg: Director, Screenplay, Writer etc.)	TMDb and IMDb
TOP BILLED CAST	Top paid cast (generally actors)	TMDb and IMDb
ORIGINAL LANGUAGE	Movie’s original shot language	TMDb
RUN TIME	Total movie run time	TMDb
GENRE	All the genres movie belongs to	TMDb
BUDGET	Budget to make the entire movie (USD)	TMDb and IMDb
REVENUE	Global / USA revenue collected (USD)	TMDb and IMDb

Table 1: Movie features extracted from IMDb and TMDb and their description

Firstly, the main objective was to use only open source data i.e. TMDb only but as the extracted data is sparse, we have scraped data from IMDb as well and merged together to form a dense dataset. Just using TMDb, BUDGET and REVENUE columns have only 40% density for overall items. Whereas using data from both the sources has improved overall density of BUDGET and REVENUE to 64% and 65% respectively.

Bayesian Networks and Probabilistic Matrix Factorization for Movie Recommendation on MovieLens dataset

In GitHub, refer to below given process for working on above job:

- TMDB_scraping_export_csv.ipynb: This notebook scrapes data from TMDb for all the features listed in Table 1
- IMDB_scraping_export_csv-100k links.ipynb: This notebook scrapes data from IMDb for all the features listed in Table 1
- Merge IMDb and TMDb.ipynb: This notebook merges the content scraped from both IMDb and TMDb
- web_scraping_IMDb_TMDb.csv: This is the final output file after running above three notebooks.

Density of the output file is given in Table 2.

Feature	**Density**
TMDB_SCORE	99.5
IMDB_SCORE	100.0
TOTAL_RATINGS	100.0
FEATURED CREW	99.8
TOP BILLED CAST	99.0
ORIGINAL LANGUAGE	100.0
RUN TIME	100.0
GENRE	99.6
BUDGET	64.7
REVENUE	78.4

Table 2: Density of each feature extracted from IMDb and TMDb

Bayesian Networks and Probabilistic Matrix Factorization for Movie Recommendation on MovieLens dataset

Creation of Item-Feature Matrix:
As the name describes, this matrix has items (movies) as rows and features (movie characteristics we have scraped from web) as columns. Cells corresponding to a particular item-feature gets populated as 1 if the selected item can be described using that feature, else we populate them with 0s. This matrix has all the cells filled i.e. 100% density. In the case of lack of data, we have input 0 for all the respective features. As instance of item feature matrix is given below in table3.

movieId	budget1	budget2	revenue1	revenue2	Crime	Romance	pop_class1
1	0	0	0	0	0	0	0
2	0	0	0	0	0	0	0
3	0	0	0	0	0	1	0
4	0	0	0	0	0	1	0
5	0	0	0	0	0	0	0
6	0	0	0	0	1	0	0
7	0	0	0	0	0	1	0
8	0	0	0	0	0	0	0
9	0	0	0	0	0	0	0
10	0	0	0	0	0	0	0
11	0	0	0	0	0	1	0
12	0	0	0	0	0	0	0
13	0	0	0	0	0	0	0
14	0	0	0	0	0	0	0
15	0	0	0	0	0	0	0
16	0	0	0	0	1	0	0
17	0	0	0	0	0	1	0
18	0	1	0	1	1	0	0
19	0	0	0	0	1	0	0
20	0	0	0	0	1	0	0
21	0	0	0	0	1	0	0
22	0	0	0	0	0	0	0

Table 3: Item-feature matrix populated with 1s and 0s. Features with 1s describe an item.

Data Cleaning and Feature Matrix Including Crew.ipynb and Data Cleaning and Feature Matrix - ex Crew.ipynb are the notebooks used for building item-feature matrix. Those items are removed which do not have any features to describe.

From the data extracted we have built 9083*95 item-feature matrix. The 95 features exclude crew and cast inclusion. Whereas, when we include crew and cast, the item-feature matrix is of size 9083*26705. These two files go directly as main input into BN model.

Bayesian Networks and Probabilistic Matrix Factorization for Movie Recommendation on MovieLens dataset

Description of Features:

Feature-Classification	Description
budget1, budget2, … budget5	Budget has been divided into 5 quintiles. ‘budget5’ being the top quintile
revenue1, revenue2, … revenue5	Budget has been divided into 5 quintiles. ‘revenue5’ being the top quintile
Crime, Comedy, Romance,.. etc.	There are 20 genre classifications. Each movie can have multiple genres as well. In that case, we populate with 1s for all those columns
pop_class1, pop_class2, … pop_class10	Based on the number of users who rated in IMDb and TMDb, we have classified items into 10 quintiles. ‘pop_class10’ being the basket where the item got rated by highest set of users
rating_class1, …, rating class10	Average of IMDb and TMDb score (or ratings) has been taken and then divided into 10 quintiles. ‘rating_class10’ bucket has all the movies with top ratings
running_time1, … running_time3	Based on the total running time, each movie has been allotted to one of the 3 buckets with ‘running_time3’ being the lengthiest movie
director_high, director_low	Depending on the number of movies directed by each movie director, this classification has been done. As the name describes, director_high bucket has top directed movies.
1245, 6745 etc	These are the director, crew and cast ids given to them by TMDb. Each cell gets 1 if that director, crew or cast works in the production of respective film

User-Item Matrix:
Another input that goes into BN model is user-item matrix with corresponding cells populated with rating given by user for that item. We have used MovieLens database for building this matrix. In the master dataset and 100k dataset, we have filtered out those items for which we do not have any input on features. With this new dataset, we have built user-item matrix.

Train And Test Dataset creation.ipynb (in GitHub) has been used to build the train and test dataset from the master dataset. 80% - 20% classification has been considered for train and test matrices respectively. The output files of this notebook are available in Git as well.

These datasets i.e. train dataset, test dataset and item-feature matrix goes as input into BN model.

Exploring data using preliminary Bayesian model(using package)

We built a preliminary model to check the feasibility of BN in recommending movies. For that, we have used ‘libpgm’ library of python. This library mainly focusses on implementing Probabilistic Graphical Models in Python.

As input to this model, we have sourced 100k dataset from archives of MovieLens. These files have user occupation, age, gender, movie title, user-item rating, genre and movie release date information.

Given user and item description are used as nodes in the graph and relation between nodes-nodes (for example: occupation-age, rating-genre etc) are given as edges in the libpgm package. Using PMGLearner toolkit, a preliminary model has been built.
Preliminary analysis of the model gave the following results

Bayesian Networks and Probabilistic Matrix Factorization for Movie Recommendation on MovieLens dataset

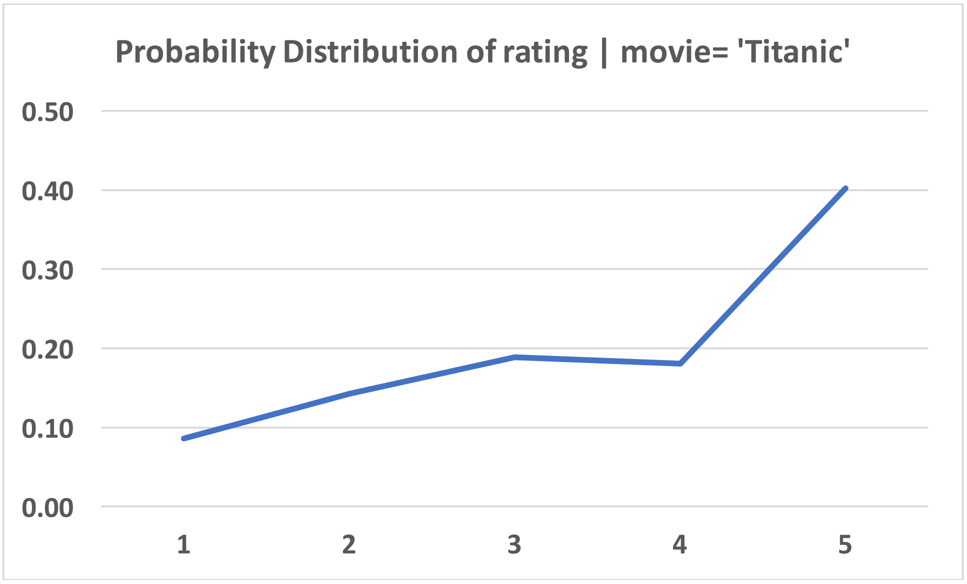


Fig 1: Probability distribution of rating given movie = Titanic

This lies in line with the fact that Titanic being popular movie and there is high probability for it being rated 5.

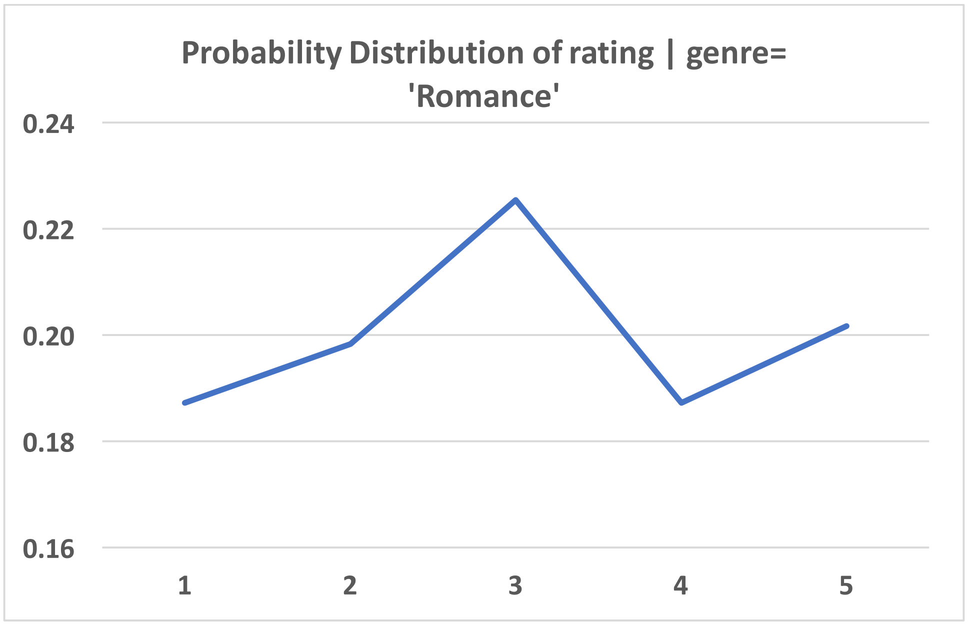


Fig 2: Probability distribution of rating given genre = Romance
Our analysis showed, romance genre has more probability of having a rating 3.

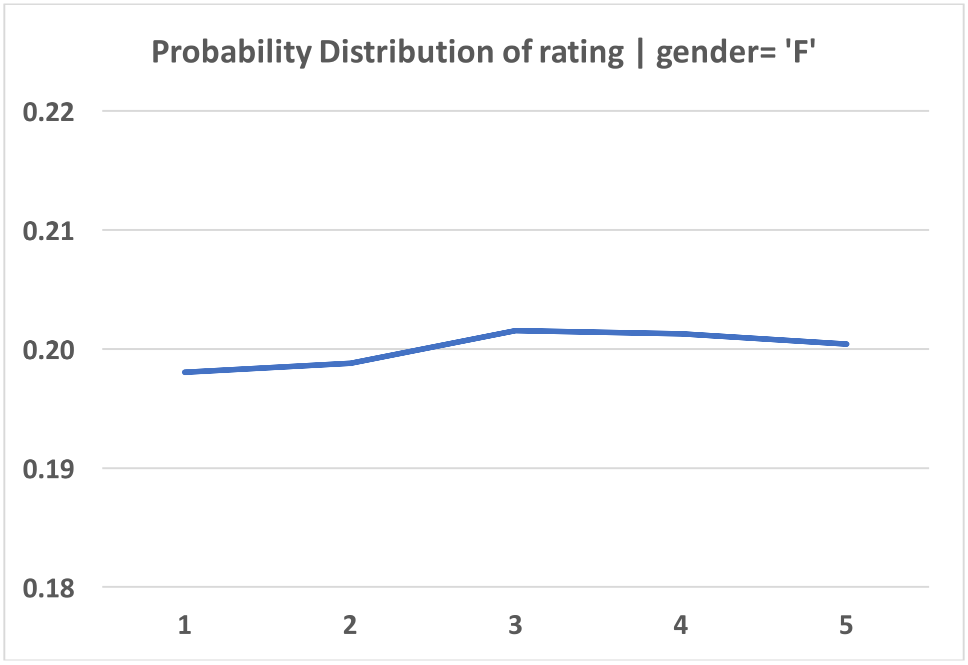


Fig 3: Probability distribution of rating given gender = Female

An average female probability distribution of ratings is almost flat; in-line with expectations.

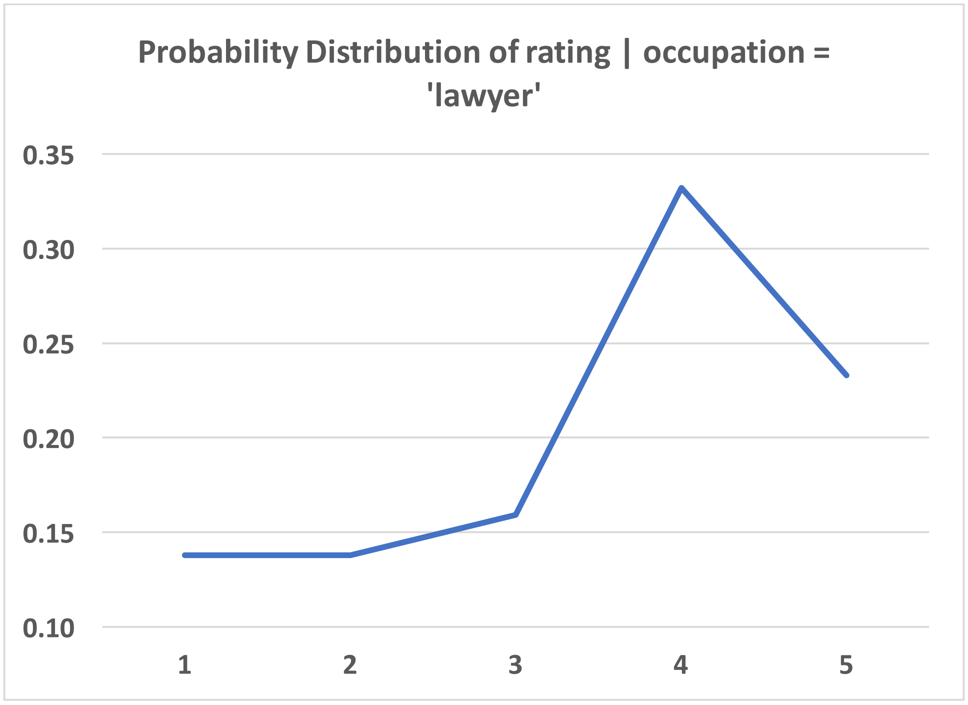


Fig 4: Probability distribution of rating given occupation = lawyer
Given a lawyer, his probability distribution is slightly skewed towards rating 4.

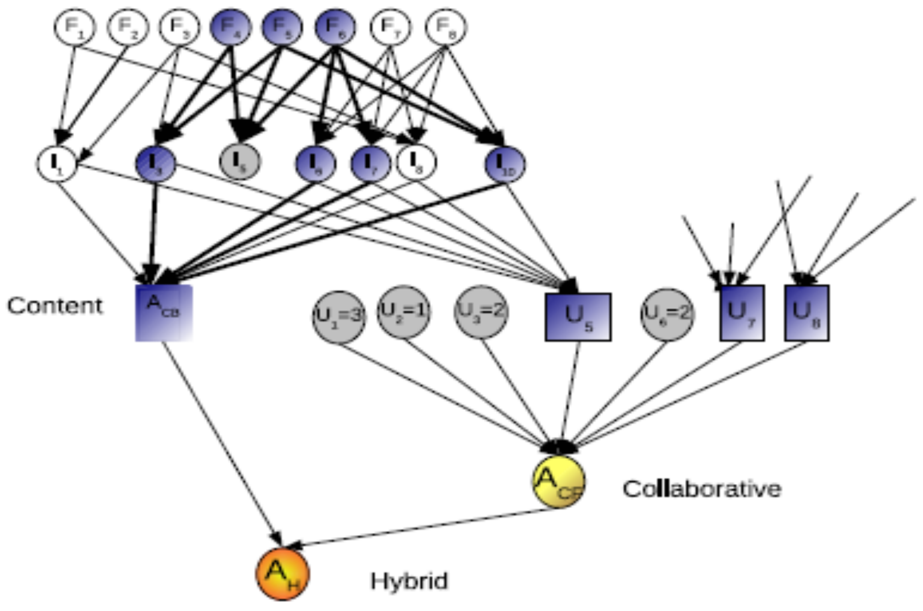
Bayesian Networks and Probabilistic Matrix Factorization for Movie Recommendation on MovieLens dataset

Predicting User Ratings Using a hybrid approach – Collaborative Filtering and Content Based Recommendations Using Bayesian Networks (Built from scratch; No ML packages used)

Recommender Systems are broadly classified into two categories:

1. *Collaborative filtering systems* that attempt to identify groups of people with similar tastes to those of the user and recommend items that they have liked. Such systems have the drawback that they suffer from the item cold-start problems which occur when recommendations must be made on the basis of few recorded ratings. These problems arise because the similarity analysis is not accurate enough.
2. *Content-based recommender systems* which use content information in order to recommend items similar to those previously preferred by the user. Such systems have the drawback that they suffer from the item cold-start problems which occur when recommendations must be made on the basis of few recorded ratings. These problems arise because the similarity analysis is not accurate enough.

In our dataset the **sparsity of the ratings matrix is 98.3%.** The predictions from using only collaborative filtering may suffer due to lack of similar users who rated the same movie as the active user. We therefore decided to combine both the above approaches to build a Hybrid recommender system. We used MovieLens Ratings data for collaborative filtering component and Web-Scraped movie feature data for content based recommendation.
We explored the use of Bayesian network to represent the relationships among users U, items I and features F, the elements involved in the recommendation processes. By using Bayesian networks, we combined a qualitative representation of how users and items are related (explicitly representing dependence and independence relationships in a Graphical structure) as well as a quantitative representation by means of a set of probability distributions, measuring the strength of these relationships.



Bayesian Networks and Probabilistic Matrix Factorization for Movie Recommendation on MovieLens dataset

Calculation of rating predictions using Bayesian Networks:
Content Based Recommendation $(\boldsymbol{A_{cb}})$:
We considered that an item's relevance will depend on the relevance values of the features that define it. Therefore, there will be an arc from each feature node, F_i, to the nodes representing those items, I_j, which have been described with this feature. By directing the links in this way, we allow two items with a common subset of features to be dependent.
We first initialize the probabilities of all the features with **a prior probability** as below:.

$$\Pr(f_{k,1}) = \frac{n_k + 0.5}{m+1}$$

n_k is the number of times that feature $f_k$has been used to describe an item. m is the total number of items. When predicting the rating for an item I_j, the features that describe it are assigned a prior probability of 1 ($that\ is\ we\ set\ f_{j,1}$)
With respect to the item nodes, I_j, as these represent a binary variable, the only weights to be defined are those needed to compute $Pr(I_{j,1}|Pa(I_j))$ Where $Pa(I_j)$ are the parents of Item I_j.
In order to assess these values we considered the following idea: Assume that F1 and F2 are two features describing an item I_j, with F1 being a common feature (in the sense that it has been used to describe many items) and F2 a rare feature (it appears in few items). It is natural to think that when both features are relevant (F1 = $f_{1,1}$,1 and F2 = $f_{2,1}$) the contribution of F2 on the I_j's relevance degree will be greater than the contribution of F1. The concept of inverted document frequency (IDF) is used to measure the term's importance. Therefore, using an IDF-based approach we use the expression:
log ((m /n_k) + 1)
to measure the importance of a feature in the entire database. When a feature is not relevant its weight is set to 0. Therefore, the weights of the edges from features to the items were computed as:
$w(f_{k,1}, i_{j,1}) = \frac{1}{M(I_j)} \log\left(\frac{m}{n_k} + 1\right)$ and $w(f_{k,0}, i_{j,1}) = 0$
with $M(I_j)$ being a normalizing factor computed as:

$$M(I_j) = \sum_{F_k \in Pa(I_j)} \log\left(\frac{m}{n_k} + 1\right)$$

Taking into account the number of users involved in the predictions, we developed a canonical model to represent the conditional probability distributions: the canonical weighted sum (CWS). When this model was assumed, we could factorize the conditional probabilities into smaller pieces (the weights describing the mechanisms) and use an additive criterion to combine these values . This is because the assumption of ''independence of the causal influences" holds . The probability of relevance for a given item I_j was then computed as a canonical weighted sum as follows:
$\Pr(I_{j,1}) = \sum_{F_k \in Pa(I_j)} w(f_{k,1}, i_{j,1}) \Pr(f_{k,1})$
Using the above calculated Probabilities $\Pr(I_{j,1})$ we calculate the **posterior probabilities** of the relevance of features with respect to an item using the formula below:

$$\Pr'(f_{k,1}|i_{j,1}) = \begin{cases} \Pr(f_{k,1}) \;\; if\ F_k\ does\ not\ belong\ to\ Pa(I_j) \\ \Pr(f_{k,1}) + \dfrac{w(f_{k,1}, i_{j,1}) \Pr(f_{k,1}) (1 - \Pr(f_{k,1}))}{\Pr(I_{j,1})} \;\; if\ \ F_k \in\ Pa(I_j) \end{cases}$$

Using the update $\Pr'(f_{k,1}|i_{j,1})$ values the posterior probabilities of items $\Pr'(i_{k,1})$are calculated as before.
For the active user A (for whom we are predicting) make content based prediction we used the following two ideas:
1. For a given user $U_{cb} \in \{A_{cb}\} \cup\ U_I^-$, whenever he or she rated an item I_k with the value s, then all the probability mass should be assigned to the same rating 's' at the user level. $A_{cb}$is the user variable of the active user for Content based recommendation. U_I^- is the list of users excluding user A who did not rate the given item 'k' .
2. We will assume that all the items are equally important for predicting the active user's rating.
Using the above assumptions, we compute the weights from items to the users as below:

$$w(i_{k,1}, u_{cb,s}) = \frac{1}{|I(U_{cb)}|}$$

$$w(i_{k,1}, u_{cb,t}) = 0, if\ t \neq s, 0 \leq t \leq \#\ r$$

$$w(i_{k,0}, u_{cb,0}) = \frac{1}{|I(U_{cb)}|}\ ,$$

$$w(i_{k,0}, u_{cb,t}) = 0, if\ 1 \leq t \leq \#\ r$$

The probability for a user giving a rating 's' is then the sum of these weights:

$$\Pr\ (\boldsymbol{A_{cf}} = \boldsymbol{s}) = \sum_{i_{k,1} \in I(A)} w(i_{k,1}, u_{cb,s}) \Pr'(i_{k,1})$$

Collaborative-based predictions ($\boldsymbol{A_{cf}}$) :
The topology of the BN consists of a variable A, representing the active user, having as its parents those user variables, $U_i\ \in U$, with most similar tastes. These parents are learned from the database. As a similarity measure between the active user A and any other user U ($\mathrm{sim(A, U)}$) a combination of two different but complementary criteria has been used:

1. on the one hand, we use rating correlation (Pearson correlation coefficient, PC) between common items to measure the similarity between the ratings patterns.
2. The second criterion attempts to penalize those highly correlated neighbours which are based on very few co-rated items, which have proved to be bad predictors.

In Collaborative-based predictions, we determined the weights reflecting the contribution of each similar user U_i in the prediction of the rating for the active user A. The particular weights are: $w(u_{i,t} a_{cf,s}) = RSim(U_i, A) \times Pr^*(A = s|U_i = t) \quad if\ 1 \leq t, s \leq \#\ r,$

$$w(u_{i,t} a_{cf,0}) = 0 \quad if\ 1 \leq t \leq \#\ r,$$

$$w(u_{i,0} a_{cf,0}) = RSim(U_i, A),$$

$$w(u_{i,0} a_{cf,s}) = 0 \quad if\ 1 \leq s \leq \#\ r$$

Where the similarities are calculated using the below formula:

Bayesian Networks and Probabilistic Matrix Factorization for Movie Recommendation on MovieLens dataset

$$Sim(A,U) = abs\big(PC(A,U)\big) \times \frac{|I(A) \cap I(U)|}{|I(A)|}$$

where I is the set of items rated by the user in the data set. In our approach, by using the absolute value of PC, abs(PC), **we considered that both positively (those with similar ratings) and negatively correlated users (those with opposite tastes) might help to predict the final rating for an active user.**

$$Pr^*(A = s | U_i = t) = \frac{N(u_{i,t}, a_s) + 1/\#r}{N(u_{i,t}) + 1}$$

where N($u_{i,t}$, a_s) is the number of items from I(U_i) ∩ I(A) which have been rated t by U_i and also s by the active user A. In addition, N($u_{i,t}$) is the number of items in I(U_i) ∩ I(A) rated with t by U_i. The final probability is

$$\boxed{\Pr(\boldsymbol{A_{cf}} = \boldsymbol{s}) = \sum_{i \in U-\{A\}} \sum_{0.5 \le t \le \#r} w\big(u_{i,t}, a_{cf,s}\big) \Pr(U_i = t)}$$

Hybrid model predictions ($\boldsymbol{A_H}$):

As $\boldsymbol{A_H}$ node has two parents, $\boldsymbol{A_{cb}}$ and $\boldsymbol{A_{cf}}$, representing the content-based and collaborative predictions, we must assess the conditional probability values $\boldsymbol{Pr}(\boldsymbol{A_H} | \boldsymbol{A_{cb}}, \boldsymbol{A_{cf}})$. These probabilities represent how to combine both types of information when predicting the active user's rating for the item I_j . It is well known that the performance of the collaborative system improves as the information used for making recommendations increases. Inversely, the prediction for new or rare items (rated by a low number of similar users) becomes more difficult. Taking this fact into account we introduced a parameter $\boldsymbol{\alpha_j}$, $\boldsymbol{0 \le \alpha_j \le 1}$, to control the contributions of each component.

The probabilities were then computed as follows: $\mathbf{Pr}(\boldsymbol{A_H} = \mathrm{s} | \boldsymbol{A_{cb}}, \boldsymbol{A_{cf}}) = \boldsymbol{\alpha_j Pr(A_{cb})} + \boldsymbol{(1 - \alpha_j) Pr(A_{cf})}$

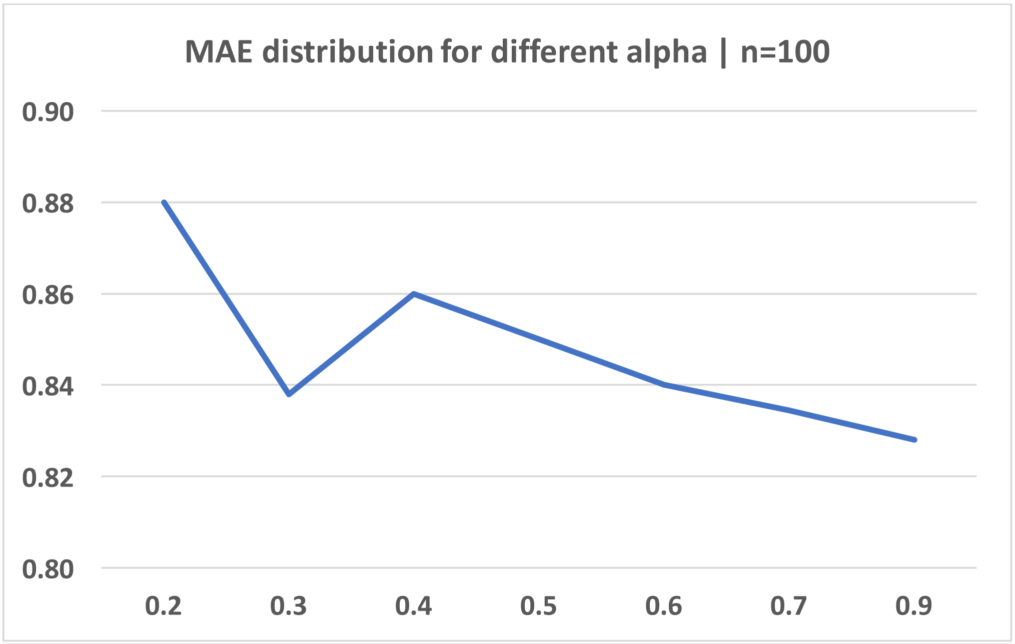
Base line model benchmark:

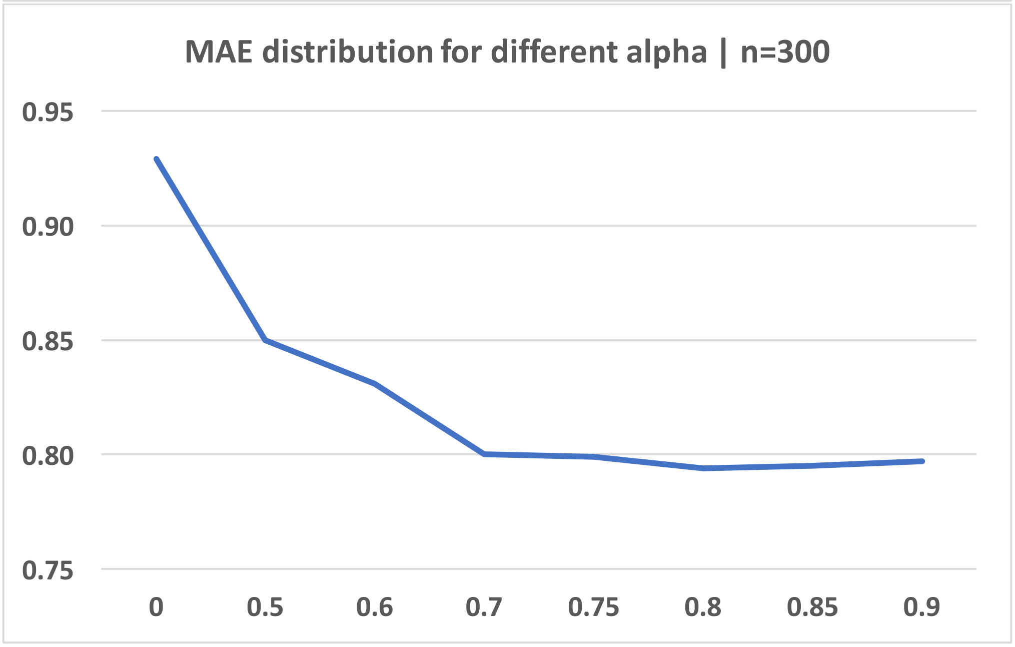
We run the model for just the **collaborative recommendation** by setting $\boldsymbol{\alpha_j = 0}$. The MAE was approximately 0.93, This MAE value was improved by adding the Content-based component to the model by increasing the value of $\boldsymbol{\alpha_j}$

Exploring the model:

- The model was run for different user sizes: n = 100, n= 300 and n = 671 (all users) and the value of the Mean Absolute Error was plotted:

Bayesian Networks and Probabilistic Matrix Factorization for Movie Recommendation on MovieLens dataset



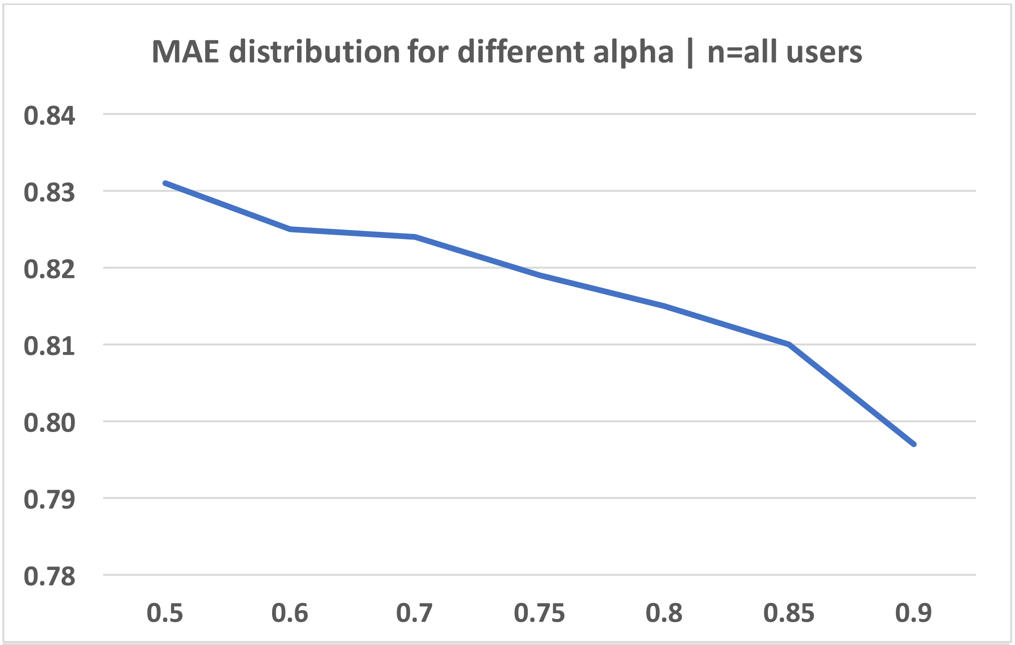


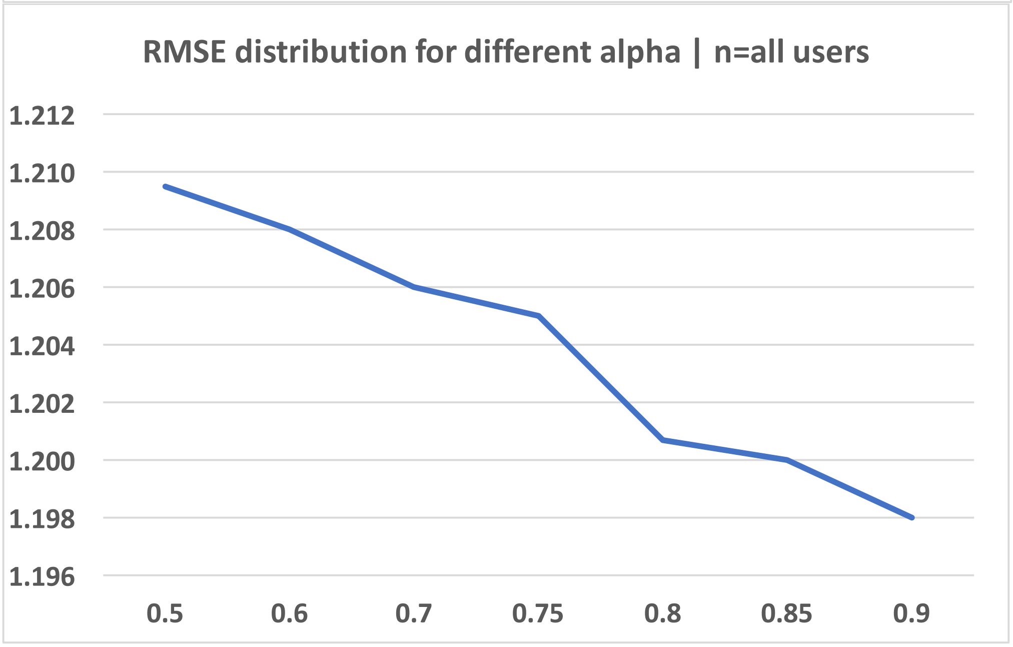
Based on the above plots, we observed that the performance of the model was best for

$$\boldsymbol{\alpha_j} \approx 0.8$$

- Below are the graphs showing the performance of the model at different values of α_j for full user base

Bayesian Networks and Probabilistic Matrix Factorization for Movie Recommendation on MovieLens dataset





This is in-line with our expectations that as we increase the component of content-based recommendation the MAE as well as the RMSE reduces. This is because, due to high sparsity of the ratings matrix, the collaborative filtering component is inaccurate. The inaccuracy is compensated by the content-based recommendation component of the recommendation.
The best RMSE observed for the model was **1.198 at** $\boldsymbol{\alpha_j \approx 0.9}$ when run for all 671 users.

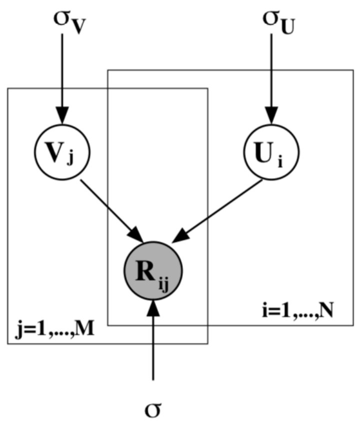
Bayesian Networks and Probabilistic Matrix Factorization for Movie Recommendation on MovieLens dataset

Probabilistic Matrix Factorization:

Many existing approaches to collaborative filtering can neither handle very large datasets nor easily deal with users who have very few ratings. The Probabilistic Matrix Factorization (PMF) model scales linearly with the number of observations and, more importantly, performs well on the large, sparse, and very imbalanced datasets.

As we know, the popular approaches to collaborative filtering are based on low-dimensional factor models. The idea behind such models is that the preferences of a user are determined by a small number of latent factors. In a linear factor model, a user's preferences are modelled by linearly combining item factor vectors using user-specific coefficients. For example, for N users and M movies, the N × M preference matrix R is given by the product of an N × D user coefficient matrix U^T and a D × M factor matrix V. Training such a model amounts to finding the best rank-D approximation to the observed N × M target matrix R under the given loss function.

The below figure shows the graphic representation for the probabilistic factorization model (PMF):



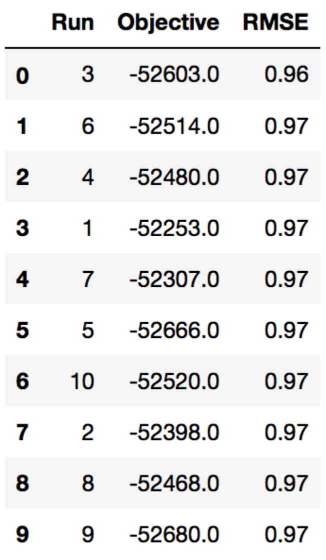
We have a matrix of N_1 users who have rated N_2 objects (movies). Let us call this matrix *M*, where M_{ij} contains the rating for user *i* of object *j*. PMF is a generative model in which we assume gaussian distribution for user and object locations and learn the gaussian parameters from the data (matrix). To obtain the parameters of the distribution we need to maximize the log-liklihood:

$$p(M_0, U, V) = \left[\prod_{(i,j)\in\Omega} p(M_{ij}|u_i, v_j) \right] x \left[\prod_{j=1}^{N_2} p(v_j) \right] x \left[\prod_{i=1}^{N_1} p(u_i) \right]$$

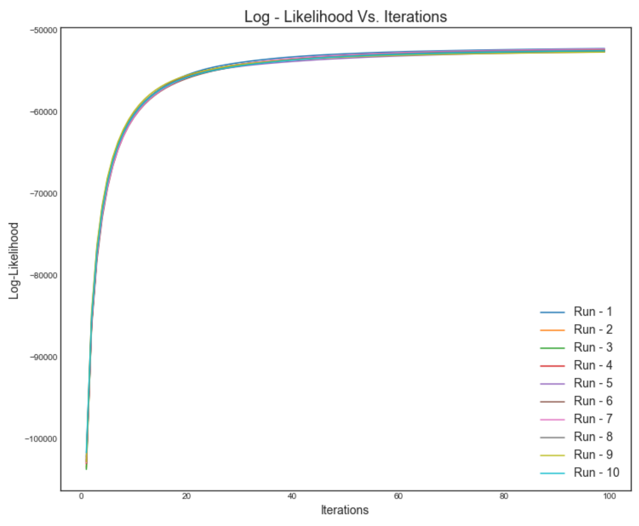
Bayesian Networks and Probabilistic Matrix Factorization for Movie Recommendation on MovieLens dataset

Results:

Accuracy measure and convergence of log likelihood:
The model yields the best RMSE of 0.96 for run 3 (Fig 1). And the log likelihood starts converging for 25 – 30 iterations (Fig 2).



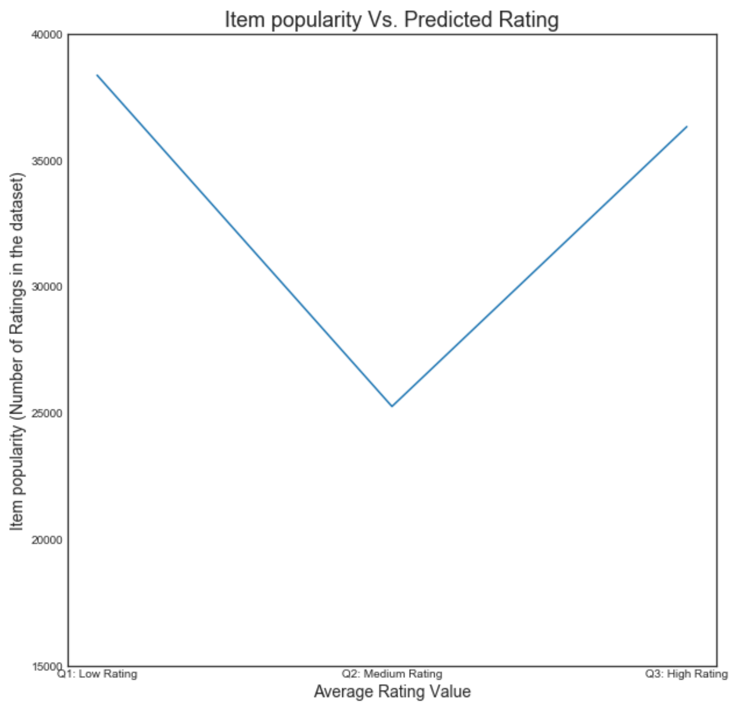
	Run	Objective	RMSE
0	3	-52603.0	0.96
1	6	-52514.0	0.97
2	4	-52480.0	0.97
3	1	-52253.0	0.97
4	7	-52307.0	0.97
5	5	-52666.0	0.97
6	10	-52520.0	0.97
7	2	-52398.0	0.97
8	8	-52468.0	0.97
9	9	-52680.0	0.97



Bayesian Networks and Probabilistic Matrix Factorization for Movie Recommendation on MovieLens dataset

Movie Popularity vs Predicted Rating:
Here we look at how the popularity of a movie and its mean predicted rating are related after using the PMF model. Popularity and mean rating are defined as below:

Popularity: Number of times the movie was rated in the given dataset
Mean Rating: Mean rating of the movie after prediction
We follow the below mentioned approach:
Step 1:
Create quantiles of the mean predicted rating
Step 2:
Find total popularity of the movies in the given quantiles



As we can see above, the model does not necessarily give a higher rating to a more popular movie. Thus, the model might be able to recommend movies that are not very popular and can give serendipitous recommendations.

How personalized are our recommendations?

We pick a user at random, find the most similar user and a very dissimilar user and see how the different are our recommendations for them.

Bayesian Networks and Probabilistic Matrix Factorization for Movie Recommendation on MovieLens dataset

We follow the below given approach for the same:
Step 1:
Pick a user at random, calculate the distances between all users and this user. Hence find the most similar and dissimilar user to the chosen user
Step 2:
Find the best recommended movie to all the three users
Step 3:
Find the distances between the recommended movies

Our **premise is that, the distance between recommended movies of similar users will be less compared** to the distance between dissimilar users.

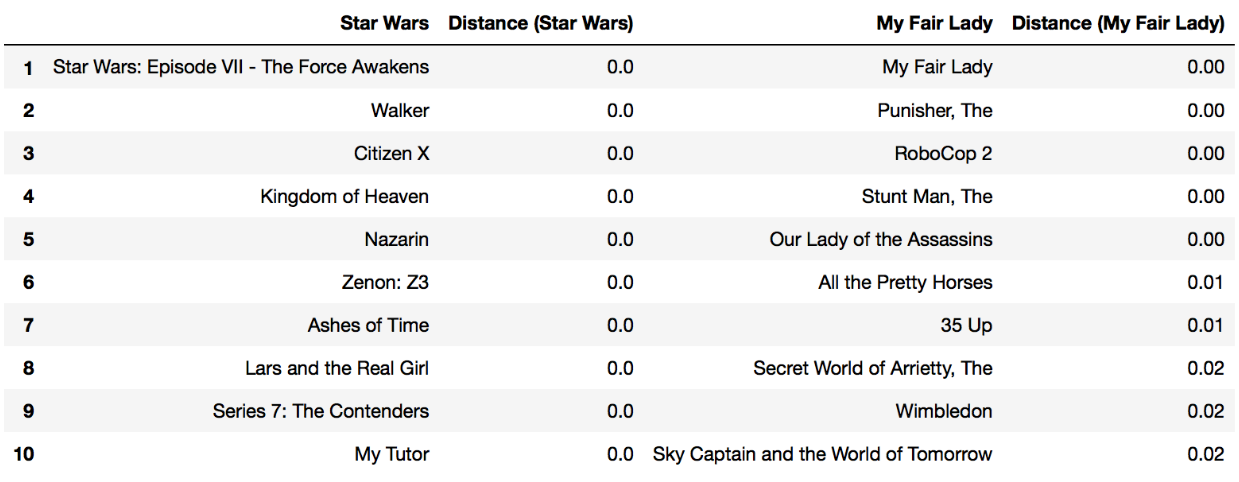
According to our findings:
Distance between movie features of top movie recommendation of two similar users = 0.47
Distance between movie features of top movie recommendation of two dissimilar users = 0.60

As discussed in our premise we can see that the distances between movie recommendations of two dissimilar users is greater than that between two similar users.

Another point to note here is that, the distance of 0.47 between top recommendation of two closest users suggests that the recommendations are personalized.

Looking at the item similarity matrix:

Let us now look at the movies which are closest to "Star Wars" and "My Fair Lady" based on the ratings matrix using the PMF model:



	Star Wars	Distance (Star Wars)	My Fair Lady	Distance (My Fair Lady)
1	Star Wars: Episode VII - The Force Awakens	0.0	My Fair Lady	0.00
2	Walker	0.0	Punisher, The	0.00
3	Citizen X	0.0	RoboCop 2	0.00
4	Kingdom of Heaven	0.0	Stunt Man, The	0.00
5	Nazarin	0.0	Our Lady of the Assassins	0.00
6	Zenon: Z3	0.0	All the Pretty Horses	0.01
7	Ashes of Time	0.0	35 Up	0.01
8	Lars and the Real Girl	0.0	Secret World of Arrietty, The	0.02
9	Series 7: The Contenders	0.0	Wimbledon	0.02
10	My Tutor	0.0	Sky Captain and the World of Tomorrow	0.02