MATRIX FACTORIZATION TECHNIQUES FOR RECOMMENDER SYSTEMS

ANDREAS ARTMEIER

Abstract. Recommender systems offer recommendations for users to find items they might be interested in. Collaborative filtering based recommender systems rely on the preferences of other users by using different approaches to discover relations between items and users. This project presents a mathematical formulation of the problem and the implementation of a solver based on SVD in Matlab. In addition its accuracy and performance is analyzed.

1. Introduction

The easy access to an exponential growing amount of information on the web leads the overwhelmed user to make suboptimal decisions in deciding which product to buy, which website to visit, or which movie to watch. A recommender systems deals with this problem by offering affordable, personal and high quality recommendations. It recommends an item to a user which he might be interested in.

2. Problem

The problem of recommending items to users appear in many variations but have the following general form. Given a set of m users $U = \{u_1, u_2, \ldots, u_m\}$, a set of n items $I = \{i_1, i_2, \ldots, i_n\}$, a set S of possible values of ratings, and a set $R \subset U \times I \times S$ of ratings, we denote $I_u \subset I$ the set of items rated by user $u \in U$,

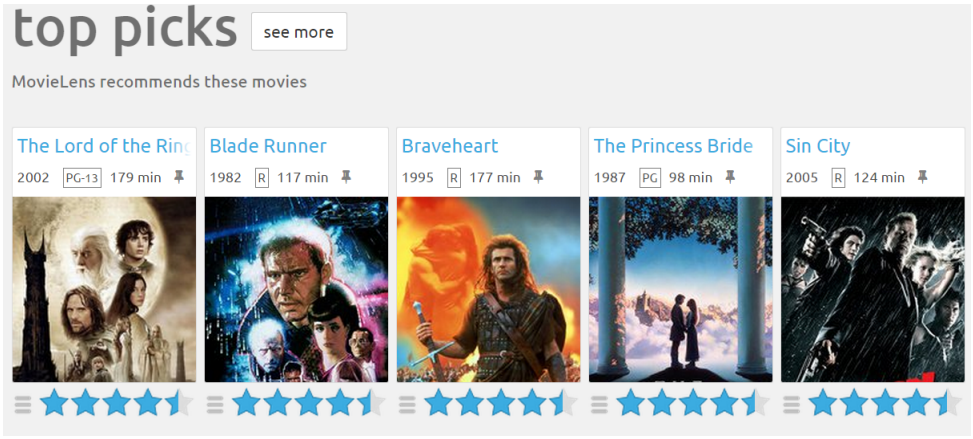

Figure 1. The movielens.org recommender recommends movies to a user

$I_{uv} = I_u \cap I_v$ the set of items rated by both users u and v, $U_i \subset U$ the set of users who rated item $i \in I$, $U_{ij} = U_i \cap U_j$ the set of users who rated both items i and j and $r_{ui} \in S$ the rating of an item $i \in I$ by user $u \in U$. Then the goal is to find a function $\hat{r} : U \times I \to S$ in order to recommend user $u \in U$ an item $i \in I \backslash I_u$ with

$$i = \arg \max_{j \in I \backslash I_u} \hat{r}(u, j)$$

Hence, the problem is a regression problem if S is continuous and a classification problem if S is discrete. If a certain rating $\hat{r}_{ui}$ should be estimated, u is called target user, i target item and $\hat{r}_{ui}$ target rating.

In the rest of the paper we use the following notation: $R_{train} \subset U \times I \times S$ is the training data, $R_{test} \subset U \times I \times S$ the test data, $\hat{r}_{ui}$ the value of the function $\hat{r}$ in (u, i), For convenience we write ui instead of (u, i), consider R sometimes as a $\{S \cup \bot\}^{m \times n}$ rating matrix with row vector r_u and column vector r_i representing the row with ratings of user u and the column of ratings of item i where missing entries are denoted by $\bot$, consider $R = R_{train}$ as the training data and some set $R_{test} \subset (U \times I \times S) \backslash R$ as test data. Then μ_r is the mean of all ratings, μ_u the mean of all ratings by user u and μ_i the mean of all ratings for item i.

The quality of a CFS is usually measured through predicted ratings $\hat{r}_{ui}$ on a test set R_{test} by the mean absolute error

$$\text{MAE} = \frac{\sum_{ui \in R_{test}} |\hat{r}_{ui} - r_{ui}|}{|R_{test}|}$$

the root mean square error

$$\text{RMSE} = \sqrt{\frac{\sum_{ui \in R_{test}} (\hat{r}_{ui} - r_{ui})^2}{|R_{test}|}}$$

and the normalized mean absolute error

$$\text{NMAE} = \frac{\text{MAE}}{r_{max} - r_{min}}$$

with $r_{\max} = \max_{ui \in R_{test}}(r_{ui})$ and $r_{\min} = \min_{ui \in R_{test}}(r_{ui})$

3. Approaches

Recommender systems can be categorized into content-based, collaborative filtering and hybrid approaches. The first approach relies on item descriptions and user profiles weighting these descriptions. The second on the assumption that users who shared same interests in the past will also have similar interests in the future. Both approaches are used together in hybrid recommender systems

The term collaborative filtering was coined in the paper [GNOT92] which deals with the problem of increasing use of electronic mail by applying a filter based on the actions of other users like replying an email. An overview of the first recommender systems is given in [RV97] where social implications like privacy concerns and possible business models are discussed. In the past Bayesian networks [BHK98], singular value decomposition with neural net classification [BP98] and induction rule learning based [BHC+98] approaches were developed. State of the art collaborative based approaches are based on neighborhood and latent factor models.

Neighborhood approaches utilize the rating matrix for each prediction and model based approaches predicting ratings based on a model. Neighborhood models can be user or item based. A user based neighborhood model has usually the following

three steps to estimate the rating $\hat{r}_{ui}$ of user u on item i. First it determines the similarity sim_{uv} between the user u and all other users v. Depending on sim_{uv} a set of similar neighbors of u is selected. The rating is estimated by a normalized weighted combination of user u's neighbors' ratings of item i. This approach was first presented in [RIS+94] through the GroupLens System where articles in newsgroups are recommended based on the ratings of other users with

$$\hat{r}_{ui} = \mu_u + \frac{\sum_{v \in U}(r_{vi} - \mu_v)w_{uv}}{\sum_{v \in U}|w_{uv}|}$$

the Pearson (product-moment) correlation coefficient as similarity measurement

$$\text{sim}_{uv} = \frac{\sum_{i \in I_{uv}}(r_{ui} - \mu_u)(r_{vi} - \mu_v)}{\sqrt{\sum_{i \in I_{uv}}(r_{ui} - \mu_u)}\sqrt{\sum_{i \in I_{uv}}(r_{vi} - \mu_v)}}$$

$w_{uv} = \text{sim}_{uv}$ and considers all users as neighbors.

Similar to user based neighborhood models, item based models predict ratings based on normalized weighted combination. But instead of relying on the preferences of like-minded users, item based approaches rely on the the active user's own ratings of similar items:

$$\hat{r}_{ui} = \frac{\sum_{j \in I_u} r_{uj}w_{ij}}{\sum_{j \in I_u}|w_{ij}|}$$

The authors in [SKKR01] analyze different item based CFSs by comparing different item-item similarity measures and prediction methods. The similarity of items i and j can be measured by the cosine similarity

$$\text{sim}_{ij} = \cos(r_i, r_j) = \frac{r_i^T r_j}{||r_i||||r_j||}$$

the (Pearson) correlation based similarity

$$\text{sim}_{ij} = \frac{\sum_{u \in U_{ij}}(r_{ui} - \mu_i)(r_{uj} - \mu_j)}{\sqrt{\sum_{u \in U_{ij}}(r_{ui} - \mu_i)}\sqrt{\sum_{u \in U_{ij}}(r_{uj} - \mu_j)}}$$

and the adjusted cosine similarity

$$\text{sim}_{ij} = \frac{\sum_{u \in U_{ij}}(r_{ui} - \mu_u)(r_{uj} - \mu_u)}{\sqrt{\sum_{u \in U_{ij}}(r_{ui} - \mu_u)}\sqrt{\sum_{u \in U_{ij}}(r_{uj} - \mu_u)}}$$

In addition a variation of the general weighted sum approach to calculate the prediction $\hat{r}_{ui}$ is given: Instead of the the values r_{uj} approximated values $\tilde{r}_{uj}$ based on a linear regression model are used. For the evaluation the MovieLens dataset containing ratings of movies is used. The experiment showed that item based models are significantly better than user based models, the adjusted cosine similarity perform significantly better than the other similarity measurements and that the regression used to generate the prediction clearly improves the performance on sparse data set compared to the basic prediction but looses it advantage if the sparsity decreases.

In contrast to neighborhood approaches model based approaches learn a model of previous ratings in order to make predictions of unknown ratings. At the beginning latent semantic analysis [Hof04] and latent Dirichlet analysis [BNJ03] were

used. Later approaches derived the latent factor model from the singular value decomposition (SVD) of the user-item matrix.

Given a $m \times n$ real matrix R, the SVD is a matrix factorization of the form $R = U\Sigma V^T =$ with U being a $m \times m$ real unitary matrix, $\Sigma = \text{diag}(\sigma_1, \sigma_2, \ldots, \sigma_m)$ a $m \times n$ non-negative real rectangular diagonal matrix with $\sigma_1 \geq \sigma_2 \geq \cdots \geq \sigma_m$ and V a $n \times n$ real unitary matrix.

A low rank approximation $\hat{R} = U_1\Sigma_1 V_1^T$ with rank r of R can be derived from SVD with $R = U\Sigma V^T = \begin{pmatrix} U_1 & U_2 \end{pmatrix} \begin{pmatrix} \Sigma_1 & 0 \\ 0 & \Sigma_2 \end{pmatrix} \begin{pmatrix} V_1 \\ V_2 \end{pmatrix}^T$ with the $m \times r$ matrix U_1, the $r \times r$ matrix Σ_1 and the $n \times r$ matrix V_1. This approach provides the best rank r approximation according to the Frobenius norm with $||R - \hat{R}||_F = \sqrt{\sigma_{r+1}^2 + \sigma_{r+2}^2 + \cdots + \sigma_m^2}$.

Setting $X = U_1\Sigma_1^{\frac{1}{2}}$ and $Y = V_1\Sigma_1^{\frac{1}{2}}$ and with x_u being the row vector of user u in X and y_i the row vector of item i in Y we can express an estimation of user u's rating of item i by a low rank matrix approximation $\hat{r}_{ui} = x_u^T y_i$.

In [SKKR00] the three main problems of neighborhood models are discussed: sparsity of the data prevent discovery of neighbors, the lack of scalability causes performance problems with growing numbers of items and users and synonyms of products names prevent neighborhood models to discover their close similarity. The MovieLens dataset of rated movies was used for the evaluation. In order to directly calculate the SVD factorization missing entries of the rating matrix had to be replaced by average ratings. Later research showed that this approach were the reasons of the observed weak performance of the applied model.

Another approach of filling the missing ratings for SVD was presented in [KY05]. The SVD is iteratively performed where in each iteration the estimation of the missing ratings is improved by the results of the estimation in the previous iteration. The drawback of this methods are the computational costs for many iterations and distortions in the data by the huge number of missing data which need to be estimated.

The collaborative filtering problem became very popular by the Netflix price [BL07]. The challenge is to develop a collaborative filtering algorithm to predict user ratings of movies based on their previous ratings. Given 100 million ratings (an integer from 1 to 5) with timestamp from October 1998 to December 2005 of 18 thousand movies by 480 thousand randomly chosen users the contestants must improve the accuracy, measured by the RMSE on a three million rating qualifying set, of Netflix own recommender Cinematch by 10% to be rewarded the grand prize of one million dollars. In addition two progress prizes for the best prediction in 2007 and 2008 were each rewarded with 50 000 dollar. This competition significantly increased the interest in collaborative filtering and advanced the SVD approach with by following two publications about recommender systems for the Netflix price.

The main problem of latent factor models was performing SVD because of the huge dimension of the matrix and its missing values. This problem was solved in [BKV07] by directly minimizing the error of the low rank matrix factorization with respect to the known set R of ratings:

$$\min_{X,Y} \sum_{(u,i) \in R} (r_{ui} - x_u^T y_i)^2$$

The latent factor matrices X an Y can be determined by alternating gradient descent where in each update one of the two matrices is alternating fixed.

The author of [Pat07] improved regularized SVD with biases b_u for user u and b_i for movie i for the average overall movie rating b resulting in

$$\hat{r}_{ui} = b + b_u + b_i + x_u^T y_i$$

where the parameters are determined by gradient descent with regularization with updates:

$$\begin{aligned} x_u &+= \alpha((r_{ui} - \hat{r}_{ui})y_i - \lambda x_u) \\ y_i &+= \alpha((r_{ui} - \hat{r}_{ui})x_u - \lambda y_i) \\ b_u &+= \alpha((r_{ui} - \hat{r}_{ui}) - \lambda_2(b_u)) \\ b_i &+= \alpha((r_{ui} - \hat{r}_{ui}) - \lambda_2(b_i)) \end{aligned}$$

with learning rate α, and regularization parameters λ, λ_2.

The SVD approach can be easily extended by social and temporal contextual information. The time an item is rated by an user is a contextual dimension of this rating. The utilization of the rating time was first presented by [ZCM01] as a time series prediction problem but only gained popularity by its use in the Netflix prize in [Kor10]. In [RGFST11] a generalized factor model based on a factorization machine [Ren10] which can incorporate arbitrary set of contextual information is presented. Let z_t be the factor vector representing the time. Then the rating of item i by user u at time t can then be estimated by

$$\hat{r}_{uit} = b + b_u + b_i + b_t + x_u^T y_i + x_u^T z_t + y_i^T z_t$$

Social-Trust based recommendation relies on the similarity of neighbors in a social network. This similarity originates from social influence from neighbors and selection when new social ties to similar users are formed. These effects are discussed in [CCH+08] and it is showed that social interactions are better predictors of future activities at Wikipedia than the similarity of tastes.

4. Experiment

For the experiments the MovieLens (http://movielens.org) data set with 100 000 ratings of the GroupLens Research project is used. MovieLens is a recommender system for movies based on the preferences of the users using collaborative filtering. The research group provides four data sets of different size where each user has rated at least 20 movies. The smallest data set was first used in [HKBR99] to analyze the accuracy of different similarity measures for user based neighborhood models. The data provides the user id, movie id, the rating value on a scale 1 to 5 and the time timestamp of each rating. Because the used ratings are within a year the timestamps were mapped to its day of the year.

Table 1. Data sets provided by the GroupLens Research project

Time period	Users	Movies	Ratings
1997-1998	943	1 682	100 000
	6 040	3 900	1 000 209
	71 567	10 681	10 000 054
1995 - 2015	138 493	27 278	20 000 263

Here 90% of the ratings are used to train the models and the remaining 10% are used to evaluate them with the RMSE.

The following models were implemented in Matlab:

Biases	$\hat{r}_{ui} = \mu + b_u + b_i$
SVD	$\hat{r}_{ui} = x_u^T y_i$
SVD with biases	$\hat{r}_{ui} = \mu + b_u + b_i + x_u^T y_i$
SVD with temporal effects	$\hat{r}_{ui} = x_u^T y_i + z_t^T y_i + x_u^T z_t$
SVD with temporal effects and bias	$\hat{r}_{ui} = \mu + b_u + b_i + b_t + x_u^T y_i + z_t^T y_i + x_u^T z_t$

with x_u the latent vector representing user u, y_i the latent vector representing item i, the biases b_u for user u and b_i for movie i and the average overall movie rating μ. The implementation of the SVD with biases models is given in the following listing. The other models were implemented analogously. The number of factors, regularization parameters, and learning rate were determined by a 10-fold cross-validation on the training data.

```
function [bu,bi,X,Y] = learnLatent(R,uNumber,mNumber,b,k,lambda,lambda2,alpha,nIterations)
%R is a matrix with the row format <userID> <movieID> <rating>

X=rand(uNumber,k)-0.5;
Y=rand(mNumber,k)-0.5;
bu=zeros(uNumber,1);
bi=zeros(mNumber,1);
n = length(R);

for p=1:nIterations
    for ui=1:n
      u =  R(ui,1);
      i =  R(ui,2);
      bias = b+bu(u)+bi(i);

      error = R(ui,3) - bias - X(u,:)*Y(i,:)';

      temp =  X(u,:);
      X(u,:) = X(u,:) + alpha*(error*Y(i,:)-lambda*X(u,:));
      Y(i,:) = Y(i,:) + alpha*(error*temp-  lambda*Y(i,:));

      bu(u) =  bu(u) + alpha*(error-lambda2*bu(u));
      bi(i) =  bi(i) + alpha*(error-lambda2*bi(i));

    end
    Re = predictLatent(RTest,b,bu,bi,X,Y );
end
end
```

The implementation avoids the construction of a sparse matrix by iterating through the training data in a sequence. The computation time and memory consumption grows asymptotically linear with the numer of ratings and iterations.

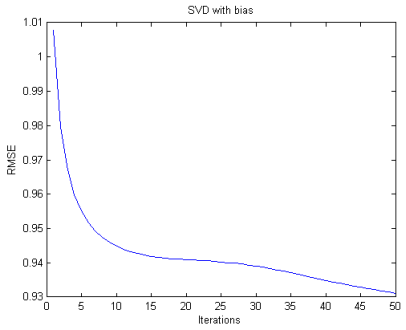

FIGURE 2. RMSE of the SVD model depending on the number of iterations for its training.

5. Evaluation

The RMSE was used to evaluate the models. Some simple statistic baseline methods were evaluated to have a comparison. In table 5 random values between 1 and 5 (rand), the average overall rating (μ), the average rating of the target user (μ_u), the average rating of the target item (μ_i) and the average between them ($\frac{\mu_u+\mu_i}{2}$) as predictors are given.

TABLE 2. Computation time of simple statistic predictors

Estimator	rand	μ	μ_u	μ_i	$\frac{\mu_u+\mu_i}{2}$
RMSE	1.8861	1.1237	1.0420	1.0233	0.9832

In Table 5 the RMSE of the models are given. The suffix T indicates that temporal effects are considered while B indicates the use of biases in the model.

TABLE 3. RMSE and computation time of SVD model with biases and temporal effects

Model	Bias	BiasT	SVD	SVDT	SVDB	SVDTB
RMSE	0.9431	0.9410	0.9319	0.9635	0.9248	0.9350
Time	0.24s	0.34s	16s	32s	22s	42s

All models show an significant lower RMSE than the statistic predictors. In addition the use of biases improve the SVD with and without temporal effects. However the use of temporal effects has a lower prediction quality than without.

Figure 2 shows the decrease of the RMSE with increasing number of iterations.

6. Conclusion

The development of the last 20 years in collaborative filtering systems showed two main developments. First, the two basic approaches, neighborhood and latent factor models were continuously improved, combined and optimized. Secondly, the basic approaches are extended by the context of ratings, items and users. The use of SVD in recommender systems offer a fast, simple and easily extendable approach for the recommender problem.

References

[BHC+98] Chumki Basu, Haym Hirsh, William Cohen, et al. Recommendation as classification: Using social and content-based information in recommendation. In *AAAI/IAAI*, pages 714–720, 1998.

[BHK98] John S Breese, David Heckerman, and Carl Kadie. Empirical analysis of predictive algorithms for collaborative filtering. In *Proceedings of the Fourteenth conference on Uncertainty in artificial intelligence*, pages 43–52. Morgan Kaufmann Publishers Inc., 1998.

[BKV07] Robert Bell, Yehuda Koren, and Chris Volinsky. Modeling relationships at multiple scales to improve accuracy of large recommender systems. In *Proceedings of the 13th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 95–104. ACM, 2007.

[BL07] James Bennett and Stan Lanning. The netflix prize. In *Proceedings of KDD cup and workshop*, volume 2007, page 35, 2007.

[BNJ03] David M Blei, Andrew Y Ng, and Michael I Jordan. Latent dirichlet allocation. *the Journal of machine Learning research*, 3:993–1022, 2003.

[BP98] Daniel Billsus and Michael J Pazzani. Learning collaborative information filters. In *ICML*, volume 98, pages 46–54, 1998.

[CCH+08] David Crandall, Dan Cosley, Daniel Huttenlocher, Jon Kleinberg, and Siddharth Suri. Feedback effects between similarity and social influence in online communities. In *Proceedings of the 14th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 160–168. ACM, 2008.

[GNOT92] David Goldberg, David Nichols, Brian M Oki, and Douglas Terry. Using collaborative filtering to weave an information tapestry. *Communications of the ACM*, 35(12):61–70, 1992.

[HKBR99] Jonathan L Herlocker, Joseph A Konstan, Al Borchers, and John Riedl. An algorithmic framework for performing collaborative filtering. In *Proceedings of the 22nd annual international ACM SIGIR conference on Research and development in information retrieval*, pages 230–237. ACM, 1999.

[Hof04] Thomas Hofmann. Latent semantic models for collaborative filtering. *ACM Transactions on Information Systems (TOIS)*, 22(1):89–115, 2004.

[Kor10] Yehuda Koren. Collaborative filtering with temporal dynamics. *Communications of the ACM*, 53(4):89–97, 2010.

[KY05] Dohyun Kim and Bong-Jin Yum. Collaborative filtering based on iterative principal component analysis. *Expert Systems with Applications*, 28(4):823–830, 2005.

[Pat07] Arkadiusz Paterek. Improving regularized singular value decomposition for collaborative filtering. In *Proceedings of KDD cup and workshop*, volume 2007, pages 5–8, 2007.

[Ren10] Steffen Rendle. Factorization machines. In *Data Mining (ICDM), 2010 IEEE 10th International Conference on*, pages 995–1000. IEEE, 2010.

[RGFST11] Steffen Rendle, Zeno Gantner, Christoph Freudenthaler, and Lars Schmidt-Thieme. Fast context-aware recommendations with factorization machines. In *Proceedings of the 34th international ACM SIGIR conference on Research and development in Information Retrieval*, pages 635–644. ACM, 2011.

[RIS+94] Paul Resnick, Neophytos Iacovou, Mitesh Suchak, Peter Bergstrom, and John Riedl. Grouplens: an open architecture for collaborative filtering of netnews. In *Proceedings of the 1994 ACM conference on Computer supported cooperative work*, pages 175–186. ACM, 1994.

[RV97] Paul Resnick and Hal R Varian. Recommender systems. *Communications of the ACM*, 40(3):56–58, 1997.

[SKKR00] Badrul Sarwar, George Karypis, Joseph Konstan, and John Riedl. Application of dimensionality reduction in recommender system-a case study. Technical report, DTIC Document, 2000.

[SKKR01] Badrul Sarwar, George Karypis, Joseph Konstan, and John Riedl. Item-based collaborative filtering recommendation algorithms. In *Proceedings of the 10th international conference on World Wide Web*, pages 285–295. ACM, 2001.

[ZCM01] Andrew Zimdars, David Maxwell Chickering, and Christopher Meek. Using temporal data for making recommendations. In *Proceedings of the Seventeenth conference on Uncertainty in artificial intelligence*, pages 580–588. Morgan Kaufmann Publishers Inc., 2001.