Quantile Matrix Factorization for Collaborative Filtering

Alexandros Karatzoglou[1] and Markus Weimer[2]

[1] Telefonica Research
Barcelona, Spain
alexk@tid.es
[2] Yahoo! Labs
Santa Clara, USA
weimer@acm.org

Abstract. Matrix Factorization-based algorithms are among the state-of-the-art in Collaborative Filtering methods. In many of these models, a least squares loss functional is implicitly or explicitly minimized and thus the resulting estimates correspond to the conditional mean of the potential rating a user might give to an item. However they do not provide any information on the uncertainty and the confidence of the Recommendation. We introduce a novel Matrix Factorization algorithm that estimates the conditional quantiles of the ratings. Experimental results demonstrate that the introduced model performs well and can potentially be a very useful tool in Recommender Engines by providing a direct measure of the quality of the prediction.

1 Introduction

Recent research on Recommender System algorithm is focused on Collaborative Filtering methods that compute the predictions for a user-item pair based on past ratings of this and other users. Factor models and more precisely Matrix Factorization approaches have been introduced with great success in this area, see e.g. [1] for an example using data from the Netflix Prize Competition.

The vast majority of the state-of-the-art Matrix Factorization techniques are based on least squares regression-like methods in order to build a model capable of predicting the rating for a given user-item pair. While many systems provide impressive performance as expressed in empirical evaluation measures such as the root mean squared error, they cannot address additional questions about the recommendation posed by both website owners and users; such as the how certain the system is about the quality of a prediction, e.g. in the form of a confidence interval.

From the perspective of the website owner, such information would be valuable in order to provide the users with the right “mix” of recommendations containing items the user will like with a high confidence as well as more adventurous recommendations. In more technical terms, recommender engines could be set to a conservative mode giving priority in the list of recommendations to items that exhibit a narrower confidence interval (i.e. we are fairly sure about our prediction) or could be set to a more adventurous mode where recommendations are given purely based on the conditional mean, median or the 1st quantile etc.

Users, on the other hand, may well be interested in using this information in order to narrow down their item search by first eliminating all the items they surely won't like. Please note that "surely" and "won't like" in the last sentence are two distinct pieces of information to be gathered from the recommender system. In fact, these two dimensions of a recommendation can be presented to the user in an intuitively understandable two dimensional field of recommendations similar to the one presented in [2].

Standard regression methods typically aim at estimating $y|x$ by finding the conditional mean. Quantile regression methods [3], [3], [4] aim at estimating the τ quantile of the conditional distribution of $y|x$ where e.g. the 5^{th} quantile corresponds to the median. Note that while squared error loss based regression techniques are sensitive to noise and outliers, the estimation of the conditional median is more robust to both outliers and noise. User rating data and collaborative data in general are known to contain noise and outliers [5].

In least squares regression the desired estimate of $y|x$ is given by a conditional mean. In certain occasions one wants to obtain a good estimate that satisfies the property that a proportion, τ, of $y|x$, will be below the estimate (for $\tau = 0.5$ this is an estimate of the median). This type of regression is known under the term quantile regression. Assume that the conditional quantile is given by the function $f(\tau|x)$ then for example for $\tau = 0.9$, $f(0.9|x)$ is the 90th percentile of the distribution of y conditional on the values of x i.e. 90% of the values of y are less than or equal to the value $f(0.9|x)$. Quantile regression has been used in many areas such as in monitoring the growth of infants given their age and gender, in ecology, in quality control and risk management where a banker might want to estimate with a high certainty a lower bound of the value of a set of financial products.

Contributions In this paper, we introduce a novel algorithm based on Matrix Factorization for computing conditional quantile estimates on Collaborative Filtering data. This algorithm provides for both a robust estimate of the conditional median of a user item combination and for any other quantile that can consequently be used to present the confidence of the provided recommendation. We present the following contributions to the field of factor models for collaborative filtering:

- A novel model for collaborative filtering based on quantile regression.
- An empirical analysis of the behavior of this system on real data.
- A simple way to integrate "external data" on the users such as demographic information and the movies such as genre into the matrix factorization model.

To the best of our knowledge, we are presenting the first matrix factorization algorithm for quantile estimation.

Organization of this Paper The remainder of this paper is organized as follows: Section 2 introduces regularized matrix factorization as well as the related work in Section 1.1. Section 3 presents the main contribution of this paper, a quantile regression model built upon the matrix factorization framework. This model is studied empirically in Section 4 before the paper ends with conclusions in Section 5.

1.1 Related Work

Factor models and more specifically matrix factorization methods have been successfully introduced to Collaborative Filtering and form the core of many successful recommender system algorithms. The basic idea is to estimate vectors $U_i \in \mathbb{R}^d$ for each user i and $M_j \in \mathbb{R}^d$ for every item j of the data set so that their inner product minimizes an explicit [6] or implicit loss function [7].

Many of the most popular matrix factorization algorithms including SVD are based upon minimizing the least squares loss function (see e.g. [8, 9]) where the sum of the squared errors between $F_{i,j}$ and $Y_{i,j}$ is used as loss function. A notable exception is the Maximum Margin Matrix Factorization approach presented in [6] which uses a multi-class hinge loss in conjunction with an L_2 regularizer to compute a model that introduces a large margin of separation, and thus improved generalization, performance.

In *matrix factorization* the observations are viewed as a sparse matrix Y where Y_{ij} indicates the rating user i gave to item j. Matrix factorization approaches then fit this matrix Y with a dense approximation F. This approximation is modeled as a matrix product between a matrix $U \in R^{n \times d}$ of user factors and a matrix $M \in R^{m \times d}$ of item factors such that $F = UM^T$.

Directly minimizing the error of F with respect to Y is prone to overfitting and thus regularization is required. Limiting the rank of the approximation by restricting d leads to a SVD of F, which is known as Latent Semantic Indexing in Information Retrieval. Note that this approach ignores the sparsity of the input data and instead models Y as a dense matrix with missing entries being assumed to be 0, thereby introducing a bias against unobserved ratings.

An alternative is proposed in [10] by penalizing the estimate only on observed values. While finding the factors directly now becomes a nonconvex problem, it is possible to use semidefinite programming to solve the arising optimization problem for hundreds, at most, thousands of terms, thereby dramatically limiting the applicability of their method. An alternative is to introduce a matrix norm, which can be decomposed into the sum of Frobenius norms [11, 6, 12]. It can be shown that the latter is a proper matrix norm on F. Together with a multiclass version of the hinge loss function that induces a margin, [6] introduced Maximum Margin Matrix Factorization (MMMF) for Collaborative Filtering. We follow their approach in this paper. Similar ideas were also suggested by [8, 13, 1] mainly in the context of the Netflix Prize.

2 Regularized Matrix Factorization

2.1 Model

In Matrix Factorization methods for Collaborative Filtering, the known data is interpreted as a sparse matrix $Y \in \mathbb{R}^{n \times m}$ where $Y_{i,j}$ contains the rating of item j by user i, if such a rating is known. The predicted rating $F_{i,j}$ of item j by user i is modeled as a linear combination of *item factors* $M_{j*} \in \mathbb{R}^d$ and *user factors* $U_{i*} \in \mathbb{R}^d$:

$$F_{ij} = \langle U_{i*}, M_{*j} \rangle \tag{1}$$

where U_{i*} is the factor vector for user i and M_{*j} the factor vector for item j.

Let $U \in \mathbb{R}^{n \times d}$ denote the matrix of all user factor vectors and $M \in \mathbb{R}^{m \times d}$ the matrix of all item factor vectors. We can then express this prediction rule as a matrix product:

$$F = UM' \tag{2}$$

When learning a matrix factorization model, the aim is to estimate U and M in such a way that the model predictions F minimize a loss L on the training set Y.

$$U, M := argmin_{U,M} L(F = UM', Y) \tag{3}$$

However, optimizing this will typically yield poor predictive performance due to overfitting. Thus, a *regularization function* $\Omega(F)$ is added for capacity control and thus overfitting prevention. This leaves us with the following objective function:

$$U, M := argmin_{U,M} L(F = UM', Y) + \lambda \Omega(F) \tag{4}$$

Here, λ is a constant that is used to control the trade-off between the regularization and the performance of the system on the known training data. Typically (e.g. in [6]), the regularizer $\Omega(F)$ is chosen to be the sum of the Frobenius (L_2) norms of the matrices U and M.

3 Quantile Matrix Factorization

In analogy to [6] we define the loss:

$$L(F, Y) := \frac{1}{\|S\|_1} \sum_{i,j} S_{ij} l(F_{ij}, Y_{ij}) \tag{5}$$

where $l : \mathbb{R} \times \mathcal{Y} \to \mathbb{R}$ is a pointwise loss function penalizing the distance between estimate and observation while by $S \in \{0; 1\}^{n \times m}$ we denote a binary matrix with nonzero entries S_{ij} indicating whenever Y_{ij} is observed.

We now seek to find matrices $M \in \mathbb{R}^{m \times d}$ and $U \in \mathbb{R}^{n \times d}$ minimizing the following objective function:

$$U, M := argmin_{U,M} \frac{1}{\|S\|_1} \sum_{i,j} S_{i,j} l(F_{ij}, Y_{ij}) + \lambda \Omega(F) \tag{6}$$

Given the factors U, M which constitute our model we have a choice of ways to ensure that the model complexity does not grow without bound. A simple option is [12] to use the penalty

$$\Omega[U, M] := \frac{1}{2} \left[\|U\|^2_{\text{Frob}} + \|M\|^2_{\text{Frob}} \right]. \tag{7}$$

Indeed, the latter is a good approximation of the penalty we will be using. The main difference being that we will scale the degree of regularization with the amount of data similar to [1]:

$$\Omega[U,M] := \frac{1}{2}\left[\sum_i n_i \|U_i\|^2 + \sum_j m_j \|M_j\|^2\right] \tag{8}$$

Here U_i and M_j denote the respective parameter vectors associated with user i and item j. Moreover, n_i and m_j are scaling factors which depend on the number of reviews by user i and for item j respectively.

Note here that the loss function is only computed on the non-zero elements of the matrix Y as zeros are treated as missing values and not ratings. In order to now build a model to estimate quantiles on collaborative data and ultimately provide more transparent and more flexible recommendations, we introduce and adapt a new loss function to matrix factorization, namely the so-called quantile loss.

3.1 Quantile Regression

Definition 1 (Quantile). *Let $y \in \mathbb{R}$ be a random variable and $\tau \in (0,1)$. The τ quantile of y μ_τ is given by the infimum over μ for which $Pr\{y \leq \mu\} = \tau$. The conditional Quantile $\mu_\tau(x)$ for a pair of random variables $(x,y) \in \mathbb{X} \times \mathbb{R}$ is defined as the function $\mu_\tau : \mathbb{X} \to \mathbb{R}$ for which μ_τ is the infimum over μ so that $Pr\{y \leq \mu|x\} = \tau$.*

The loss function for quantile regression was derived in [3] based on the observation that minimizing $l(f) = |f - y|$ would force half of the estimates of f to lie over y and half below. This error measure corresponds to the absolute error loss functions which yields a conditional median as a solution at optimality. Essentially due to the symmetry of this error measure there will be about as many data points with negative residuals as with positive.

This leads to the natural observation that by minimizing an asymmetrically weighted absolute error measure and thus giving different weights to points below and above the estimate one can retrieve the quantiles. Effectively by tilting the loss function (figure 1) in a suitable fashion, one can get estimates for any quantile. The loss function used for quantile regression is also known as the pinball loss and is given by:

$$L(F_{ij}, Y_{ij}, \tau) = \begin{cases} \tau(F_{ij} - Y_{ij}) & F_{ij} \geq Y_{ij} \\ (\tau - 1)(F_{ij} - Y_{ij}) & F_{ij} < Y_{ij} \end{cases} \tag{9}$$

where $\tau \in (0,1)$ is the quantile to be obtained. The optimization procedure given below requires the derivative of the loss with respect to F, which can be computed as:

$$\partial_F L(F_{ij}, Y_{ij}) = \begin{cases} \tau & F_{ij} \geq Y_{ij} \\ \tau - 1 & F_{ij} < Y_{ij} \end{cases} \tag{10}$$

The loss and the derivative can be calculated for each individual user-item-rating triple and thus many simple optimization methods can be used. An illustration of the loss function is given in 1.

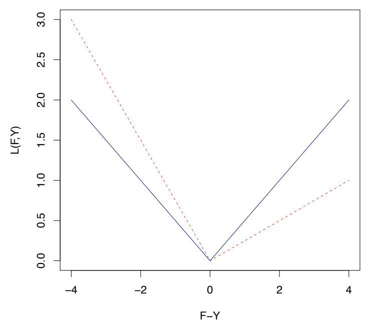


Fig. 1. Plot of the pinball loss for $\tau = 0.5$ blue line and for $\tau = 0.25$ red dotted line. Notice the higher loss incurred for negative $F - Y$ when $\tau = 0.25$

Note that when optimizing the pinball loss for Matrix Factorization the prediction function of the model $F_{ij} = \langle U_{i*}, M_{j*} \rangle$ does not return a prediction of the rating that user i might give to item j but instead returns an estimate of the conditional quantile of that prediction for the value of τ used during the optimization process. We can thus use two Quantile Matrix factorization models one for e.g. $\tau = 0.25$ and one for $\tau = 0.75$ and define a confidence interval for the ratings (in this case the interquartile range) for each user-item rating combination.

3.2 Optimization

The loss function introduced above decomposes per element of F and Y. Thus, we can employ an online optimization technique similar to the one mentioned in [8]. This method is an adaptation of the simple Stochastic Gradient Descent algorithm to the matrix factorization framework. To this end we compute the element-wise gradient of the objective function (6) with respect to F.

Notation: As the loss decomposes per element, we can define the following, more compact notation: Let f and y denote two corresponding element entries in F and Y and u and m be the corresponding rows of U and M. Then, we can formulate:

$$\partial_f E(f, y) = \partial_f L(f, y) + \frac{\lambda}{2} \partial_f \left(\|u\|_2^2 + \|m\|_2^2 \right) \tag{11}$$

The partial gradients with respect to u and m can then be written as:

$$\partial_u E = \partial_f L(f, y) m + \lambda u \tag{12}$$
$$\partial_m E = \partial_f L(f, y) u + \lambda m \tag{13}$$

This, together with a learning rate η, allows us to define the following *update rules* for an iterative optimization procedure:

$$u^{t+1} = u^t - \eta \partial_u E \tag{14}$$
$$m^{t+1} = m^t - \eta \partial_m E \tag{15}$$

This algorithm scales linearly to the number of nonzero entries in the matrix Y, and the dimensionality of d of the feature matrices that is $O(\|S\|_1 d)$ it can thus be used on very large datasets.

3.3 Feature Extensions

In Recommender Systems often additional information on the user or the items is available either in the form of demographic information on the users or of genre information for movies. In [14], the integration of features is proposed by defining a kernel between rows and columns that integrates features. Another way of introducing features is to use them as a prior for the factors, as studied in [15].

Here, we present an integration of features to the matrix factorization framework by adding several linear learning models. Assuming that the d_u user features are contained on the rows of $X^U \in \mathbb{R}^{n \times d_u}$ where n the number of users the prediction model then becomes:

$$F_{ij} = \langle U_{i*}, M_{j*} \rangle + \frac{1}{\sqrt{n_i}} \left\langle X^U_{i*}, W^U_{*j} \right\rangle \tag{16}$$

where $W^U \in \mathbb{R}^{d_u \times m}$ is a parameter matrix where m the number of movies. The same idea can be applied for the item features:

$$F_{ij} = \langle U_{i*}, M_{j*} \rangle + \frac{1}{\sqrt{n_i}} \left\langle X^U_{i*}, W^U_{*j} \right\rangle + \frac{1}{\sqrt{m_j}} \left\langle X^M_{j*}, W^M_{i*} \right\rangle \tag{17}$$

with $X^M \in \mathbb{R}^{m \times d_m}$ and $W^M \in \mathbb{R}^{n \times d_m}$ and W^U_{*j} denoting the jth column of W^U and n_i the number of items rated by user i and m_j the number of ratings for item j. We discount the influence of the external features in proportion to the square root of the number of items a user has rated. The model thus gives more weight to external features for users that have few ratings. The same principal is also applied to the items. The features can be integrated easily in the optimization procedure, i.e. in each iteration an additional update of the corresponding columns in W^U and rows in W^M is performed. The newly introduced parameter matrices are also regularized using the Frobenius L_2.

4 Experiments

Quantile Factorization models computes only quantiles and thus cannot be compared directly to standard Matrix Factorization approaches that minimize the RMSE. Moreover it is not the aim of the model to provide an exact rating prediction but instaed to provide a measure of quality for a prediction.

Section 4.1 describes the evaluation procedure, including the choice of data sets, evaluation measure and parameter tuning conducted. Section 4.2 contains the results obtained for the new model with this procedure.

4.1 Evaluation Setup

All experiments where conducted on the same data sets using the train-test splits, the evaluations measure and the number of factors d described in the following paragraphs.

Data For the experiments, we used the well known data sets EachMovie and MovieLens. Table 2 shows the descriptive statistics for these data sets. Please note that we did not perform any preprocessing of the data such as mean removal and normalization. Detailed information on proper preprocessing and main effect removal for collaborative filtering data can be found in [16] and [17].

Data set	Users	Movies	Ratings
EachMovie	61265	1623	2811717
MovieLens	6040	3900	1000209

Table 1. Data set statistics

Both data sets originate from movie recommender systems and thus contain ratings of movies by users. The ratings in the MovieLens are given on a five star scale, while EachMovie uses six rating levels. In both cases, more stars indicate a higher rating. For the EachMovie data set, we randomly extracted 10 ratings from the known ratings of each user to form the *test set* for this user. The remaining known ratings were used as the training set. For the Movielens data set, we used the 5 train-test splits provided by the GroupLens[1] research lab.

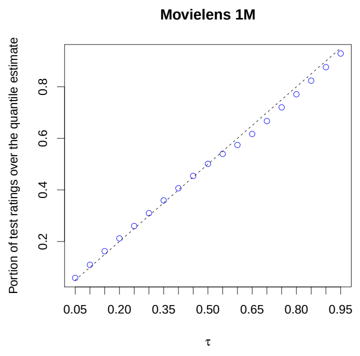
Fixed Number of Factors We tuned the dimensionality factor and regularization parameter for good performance. In all experiments the number of factors d was fixed to 20 for EachMovie and 15 for MovieLens. In our experiments we also use a single regularization parameter λ.

Model Evaluation Computing the true conditional quantile is impossible and most methods only approximate it using density estimation. Moreover, we are not aware of any other method for quantile estimation on collaborative filtering data. We thus conduct an empirical evaluation of the performance of the model by computing the portion of user-item ratings that fall over the estimated conditional quantile value F_{ij} which should be close to the quantile τ for which we are optimizing. We also explore some properties of the quantiles and the interquantile ranges.

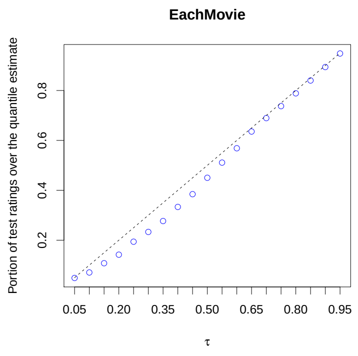
4.2 Results

Model Validation Our aim in this set of experiments is to validate the model by computing the portion of the test user ratings that fall over the computed conditional quantiles. To this end, we set the value of τ and train the model for each training set in the data. During evaluation we count the number of user-movie ratings that fall above the estimated quantile in the test set and compute the proportion these represent in the test set as mentioned this should be optimally in the order of τ. We then report the mean of these proportions over all the different train - test splits. Finally, we repeat the procedure for different τ values.

[1] `http://www.grouplens.org/`



(a)



(b)

Fig. 2. Scatterplots of the proportion of user ratings that fall over the quantile estimates in the test set for different values of the pinball loss quantile parameter τ both for the Eachmovie dataset 2(b) and the Movielens dataset 2(a). The dotted line is the $y = x$ line.

We performed model selection on one test train split for $\tau = 0.5$ for each dataset and used the computed values of λ for the procedure. Note that the standard deviation of these estimates over the different train-test sets is of the order of 0.005 and thus not reported.

Figure 4.2 contains the results of this experiment on the EachMovie and the MovieLens data. We observe very good performance of the model: the quantiles the system was trained for almost exactly carry over to the test set. We can thus deduce that the estimated conditional quantile estimates are very close to the true values and that the system exhibits a strong generalization performance.

Features We also compare the performance of the system with external features for the users and the movies provided in the Movielens dataset to the plain factorization model. The Movielens dataset contains demographic information on the users and genre information on the movies. Using 17 we include this information into the model. We repeat the experimental procedure which we used to generate figure 2 with and without features. Table 2 contains the results for different values of τ.

	Plain	Features
$\tau = 0.10$	0.1122	**0.1096**
$\tau = 0.25$	0.2609	0.2601
$\tau = 0.50$	0.5011	0.5011
$\tau = 0.75$	0.7192	0.7201
$\tau = 0.90$	0.8759	0.8761

Table 2. Results obtained form a model without and with external Features on the Movielens dataset. For $\tau = 0.10$ the model with features is statistically significantly better. For the other τ there is no statistically significant difference though the model with feature tends to perform slightly better.

The result show that including the features does only marginally improve the performance of the model. In fact this observation seems to confirm the notion that in the presence of enough rating data external features do not provide significant additional performance benefits to collaborative models [18].

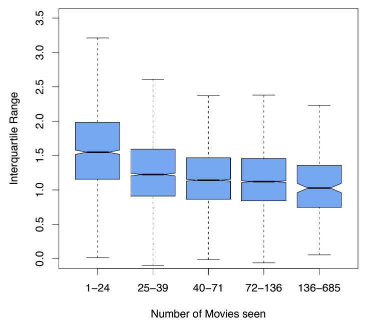


Fig. 3. Boxplot of the Interquartile Range given by groups of users grouped by numbers of movies seen on the MovieLens dataset. The upper and lower limits of the boxes represent the first and third quartiles of the interquartile ranges. The median for each group is represented by the horizontal bar in the middle of each box. If the notches of two plots do not overlap this is strong evidence that the two medians differ

Quantile Properties In this set of experiments we demonstrate the behavior of the quantiles given the amount of movies a users has seen. We calculate the 2.5 and the 7.5 conditional quantiles by training models for $\tau = 0.25$ and $\tau = 0.75$ on the MovieLens training sets. We then compute the conditional inter-quantile range that is the difference between the 2.5 and the 7.5 quantile on the test set. We split the users in five groups of users each containing about 190 users so that the first group contains users who have rated $1 - 21$ movies the second $22 - 39$ etc.

In Figure 3, we plot a notched boxplot of the interquartile ranges for different user groups grouped by the number of seen movies. We observe that the interquartile range

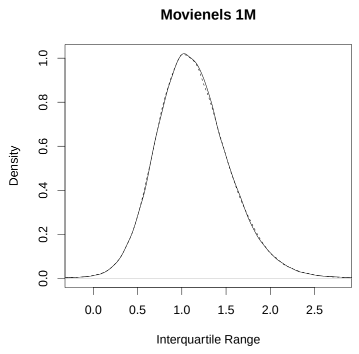


Fig. 4. Density plot of the distribution of the Interquartile range for the MovieLens dataset. The dashed line is the distribution of the interquartile range for a run without user and movie features We observe almost no significant difference in the two distributions Wilcox test p-value 0.194

is generally smaller and thus narrower for users who rated more movies. This behavior is to be expected since given more data, the model is able to identify "better" estimates, yielding somewhat narrower interquartile ranges. Moreover one would expect that users who have rated a large number of movies have developed a more consistent rating behavior.

Figure 4 shows the distribution of the width of the interquartile ranges. The interquartile range could provide for a good measure for the confidence of the rating prediction since by definition the "true" rating will be within this range with a 50% probability. We also observe that a large portion of the calculated ranges are below 1, indicating that the system is rather certain about these predictions. On the same figure we also plot the distribution of the interquartile ranges for a model with features, notice that there is no significant difference between the two.

5 Conclusions

We introduced a novel Quantile Matrix Factorization model. The model is able to estimate the confidence of the system when computing rating predictions in a Collaborative Filtering setting. To the best of our knowledge, this is the first system able to perform quantile and thus confidence prediction in Recommender Systems. Recommender Systems stand to benefit from this additional information by providing users with more information on the quality of the recommendation and giving practitioners another option in configuring their recommender engines. Experimental evaluation of the system demonstrated that the model provides an excellent estimate of the true conditional quantile and exhibits properties that warrants investigating its utility to the end-users in future research.

References

1. Bell, R., Koren, Y., Volinsky, C.: The bellkor solution to the neflix prize. Technical report, AT&T Labs (2007)
2. Ries, S.: Extending bayesian trust models regarding context-dependence and user friendly representation. In: Proc. of the 2009 ACM Symposium on Applied Computing, ACM (2009)
3. Koenker, R., Hallock, K.: Quantile regression. Journal of Economic Perspectives **15**(4) (2001) 143 – 156
4. Takeuchi, I., Le, Q.V., Sears, T.D., Smola, A.J.: Nonparametric quantile regression. Journal of Machine Learning Research **7** (2006) 1231–1264
5. Amatriain, X., Pujol, J.M., Oliver, N.: I like it... i like it not: Evaluating user rating noise in recommender systems. In: Proc. of the 7th International Conference on User Modeling, Adaptation, and Personalization. (2009)
6. Srebro, N., Rennie, J., Jaakkola, T.: Maximum-margin matrix factorization. In Saul, L.K., Weiss, Y., Bottou, L., eds.: Advances in Neural Information Processing Systems 17 NIPS, Cambridge, MA, MIT Press (2005)
7. Hoffman, T.: Latent semantic models for collaborative filtering. ACM Transactions on Information Systems (TOIS) **22**(1) (2004) 89–115
8. Takacs, G., Pilaszy, I., Nemeth, B., Tikk, D.: Scalable collaborative filtering approaches for large recommender systems. Journal of Machine Learning Research **10** (2009) 623–656
9. Koren, Y.: Collaborative filtering with temporal dynamics. In: Proceedings of the 15th ACM SIGKDD Conference on Knowledge Discovery and Data Mining (KDD), ACM, ACM Press (2009)
10. Srebro, N., Jaakkola, T.: Weighted low-rank approximations. In: Proceedings of the 20th International Conference on Machine Learning ICML, AAAI Press (2003) 720 – 727
11. Rennie, J., Srebro, N.: Fast maximum margin matrix factorization for collaborative prediction. In: Proc. of the 22nd International Conference on Machine Learning ICML. (2005)
12. Srebro, N., Shraibman, A.: Rank, trace-norm and max-norm. In Auer, P., Meir, R., eds.: Proc. Annual Conf. Computational Learning Theory. Number 3559 in Lecture Notes in Artificial Intelligence, Springer-Verlag (June 2005) 545–560
13. Salakhutdinov, R., Mnih, A.: Probabilistic matrix factorization. In: Advances in Neural Information Processing Systems 20 NIPS, Cambridge, MA, MIT Press (2008)
14. Abernethy, J., Bach, F., Evgeniou, T., Vert, J.P.: A new approach to collaborative filtering: Operator estimation with spectral regularization. Journal of Machine Learning Research **10** (2009) 803–826
15. Agarwall, D., Chen, B.C.: Regression-based latend factor models. In: Proceedings of the 15th ACM SIGKDD Conference on Knowledge Discovery and Data Mining (KDD), ACM, ACM Press (2009)
16. Bell, R., Koren, Y.: Improved neighborhood based collaborative filtering. In: The Netflix-KDD Workshop on Large-Scale Recommender Systems and the Netflix Prize Competition, ACM Press (2007)
17. Potter, G.: Putting the collaborator back into collaborative filtering. In: The 2nd-Netflix-KDD Workshop on Large-Scale Recommender Systems and the Netflix Prize Competition, ACM Press (2008)
18. Pilaszy, I., Tikk, D.: Recommending new movies: Even a few ratings are more valuable then metadata. In: Proceedings of the 3rd ACM International Conference on Recommender Systems (RecSys), ACM, ACM Press (2009)