Matrix Factorization Techniques in Collaborative Filtering

Nick Karlman
Mathematics and Computer Science Student
University of Wisconsin Whitewater
Whitewater, Wisconsin 53190
`KarlmanNJ13@uww.edu`

Abstract

Retailers and content providers offer a massive selection of products to the modern consumer. Rather than barraging a consumer with random recommendations, many businesses have become interested in recommender systems (RS) for their ability to produce personalized recommendations for every consumer. One strategy for forming these recommendations in RS is matrix factorization (MF) algorithms. Alternating Least Squares (ALS) and Stochastic Gradient Descent (SGD) are two famous MF algorithms that characterize each user and item, to provide accurate predictions of user-item ratings.

1 Introduction

Recommender systems have become very popular among online retailers and content providers. These systems predict the ratings a user would give an item, with no context of the relationship between that user and the item. Businesses that provide millions of products are most interested in maximizing sales. Many businesses like Netflix and Amazon have turned to RS for help in this. Customers have proven willing to indicate their level of satisfaction with particular items, so a huge volume of data is available about which items appeal to which customers. This data can be analyzed to recommend items to particular customers [1].

There are two strategies in RS: content filtering and collaborative filtering. Content filtering creates a profile for each user or item to capture its nature. A popular example of this is Pandora, which uses the Music Genome Project for its RS. In the Music Genome Project, each song was carefully analyzed by a professional music analyst and scored on many musical characteristics. Collaborative filtering relies on past behavior. By looking at how a user interacted with an item. This is usually represented by a rating, like 0-5 stars, or can be something calculated using every interaction between that user and item, like time spent watching/looking or relation to search. These ratings can be stored in a matrix, where each row is a user and each item is a column. This ratings matrix is incredibly sparse, however since it only contains past events. The goal of collaborative filtering RS is to predict the events that have not happened, that is, to predict the unknown ratings to give personal recommendations to users.

2 Matrix Factorization Modeling

Matrix factorization (MF) models characterize both users and items by mapping them to a joint latent factor space of dimensionality k. User-item interactions (ratings) are modeled as inner products in this space [1]. The vector for a user u is $p_u \in \mathbb{R}^k$, and the vector for item i is $q_i \in \mathbb{R}^k$. Then, the inner product of these is denoted by $\hat{r}_{ui} \in \mathbb{R}$, which is the predicted rating user u, would give item i.

We use MF techniques to solve the following problem:

$$r_{ui} \approx \hat{r}_{ui} \tag{1}$$

At full scale, let $P \in \mathbb{R}^{k*m}$ be the user matrix, where all $p_u \in P$, $Q \in \mathbb{R}^{k*n}$ be the item matrix, where all $q_i \in Q$, $R \in \mathbb{R}^{m*n}$ be the original ratings matrix where all known ratings $r_{ui} \in R$, and $\hat{R} \in \mathbb{R}^{m*n}$ be the predicted ratings matrix where all predicted ratings $\hat{r}_{ui} \in \hat{R}$.

$$P = \begin{pmatrix} | & & | \\ p_1 & \cdots & p_m \\ | & & | \end{pmatrix}, Q = \begin{pmatrix} | & & | \\ q_1 & \cdots & q_n \\ | & & | \end{pmatrix}, R = \begin{pmatrix} & & \\ & r_{u,i} & \\ & & \end{pmatrix}, \hat{R} = \begin{pmatrix} \hat{r}_{11} & \cdots & \cdots & \cdots \\ \vdots & \ddots & \cdots & \vdots \\ \vdots & \hat{r}_{ui} & \ddots & \vdots \\ \vdots & \cdots & \cdots & \hat{r}_{mn} \end{pmatrix}$$

To solve this problem, a cost function will need to be determined, as well as a defined $\hat{r}_{ui}$. A cost function will quantify the quality of the approximation. One useful measure is simply the square of the Euclidean distance of original and predicted ratings matrix [2][3],

$$||R - \hat{R}||^2 = \sum_{ui} (r_{ui} - \hat{r}_{ui})^2 \tag{2}$$

This is lower bounded by zero, and is only 0 if and only if $R = \hat{R}$. Soon this will be formulated into an optimization problem.

3 Approaches

Two of the several methods for performing MF in this situation are: Alternating Least Squares (ALS) and Stochastic Gradient Descent (SGD). Both of these algorithms optimize p_u and q_i in an alternating manner. First, they fix all q as constant and optimize p. SGD uses gradient information to find p iteratively, while ALS computes a closed-form solution [4]. Then, q_i is computed similarly. This process is looped and will terminate at some predetermined convergence.

3.1 Alternating Least Squares

ALS is an approach to an optimization problem. So, Equation 2 will need to be formed as one. With a L_2 regularizer, where $\lambda \in \mathbb{R}$, and $\hat{r}_u i = p_u^T q_i$, our optimization problem is:

$$\min_{p,q} \sum_{(u,i) \in known} (r_{ui} - p_u^T q_i)^2 + \lambda(\|p_u\|^2 + \|q_i\|^2) \tag{3}$$

That for every known rating $r_{ui} \in R$, use matrices P, Q to minimize the Euclidean distance between the original rating r_{ui} and predicted rating $\hat{r}_{ui}$. If this objective is convex, then the global minimum can be found.

3.1.1 ALS: Objective Function Not Convex

The convex function is as follows:

$$f(\alpha x + (1-\alpha)y) \leq \alpha f(x) + (1-\alpha) f(y) \tag{4}$$

To apply it to this situation, it will be:

$$f(\alpha p_1 + (1-\alpha)q_1, \alpha p_2 + (1-\alpha)q_2) \leq \alpha f(p_1, q_1) + (1-\alpha) f(p_2, q_2) \tag{5}$$

Separating this, the regularization term is convex [5]. So focusing on:

$$f(p,q) = (r - pq)^2$$

Choosing specific values:

$$p_1 = 1, q_1 = 0, \text{ so } p_1 q_1 = 0$$
$$p_2 = 1, q_2 = 0, \text{ so } p_2 q_2 = 0$$

$f(1,0) = r^2$ and $f(0,1) = r^2$, so:

$$f(p_1, q_1) = f(p_2, q_2) = r^2$$

In the RHS of Equation 5, and choosing $\alpha = 0.5$:

$$0.5r^2 + (1 - 0.5)r^2$$

On the LHS of Equation 5, we have:

$$f(0.5, 0.5) = (r - 0.5 * 0.5)^2 = (r - 0.25)^2$$

Bring both sides of Equation 5 together, and setting r=1:

$$(1 - 0.25)^2 = 0.5625 \leq 0.5 * 1^2 + (1 - 0.5) * 1^2 = 1 \tag{6}$$

Hence, this objective function is not convex.

3.1.2 Algorithm 1: ALS [4]

```
**Data:** Training data $D$ contains user $u \in N^+$, item $i \in N^+$ and rating $r_{u,i} \in \mathbb{R}$.
**Input:** Randomly initialize $p_u \in \mathbb{R}^k$ and $q_i \in \mathbb{R}^k$, number of users, and number of items.
**Result:** Latent vector $p_u$ for users and $q_i$ for items.
**while** *the result does not converge* **do**
    **for** $(u = 1; u \leq m; u^{++})$**do**
        $p_u = (\sum_{r_{ui} \in r_{u*}} q_i q_i^T + \lambda I_k)^{-1} \sum_{r_{ui} \in r_{u*}} r_{ui} q_i$
    **for** $(i = 1; i \leq n; i^{++})$**do**
        $q_i = (\sum_{r_{ui} \in r_{*i}} p_u p_u^T + \lambda I_k)^{-1} \sum_{r_{ui} \in r_{*i}} r_{ui} p_u$
```

3.2 Stochastic Gradient Descent

Another approach to this is SGD, and once again Equation 2 will need to be formed as an optimization problem. Similarly, with a L_2 regularizer, where $\lambda \in \mathbb{R}$. Due to issues during experimentation, the objective function used for SGD has one significant difference, which is $\hat{r}_u i = \mu + b_u - b_i + p_u^T q_i$. The bias involved in rating r_{ui} is denoted by b_{ui} and accounts for the user and item effects. The over all average is denoted by μ: the parameters b_u and b_i indicated the observed deviations of user u and item i, respectively [1]. The objective function used is:

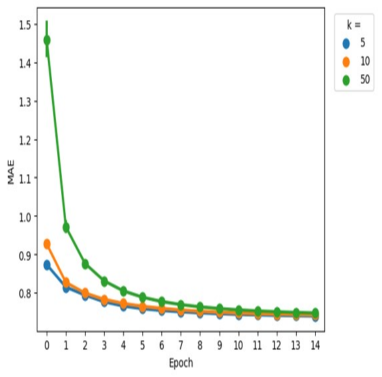
$$\min_{p,q,b} \sum_{(u,i) \in known} (r_{ui} - \mu + b_u - b_i + p_u^T q_i)^2 + \lambda(\|p_u\|^2 + \|q_i\|^2 + \|b_u\|^2 + \|b_i\|^2) \tag{7}$$

From section 3.1.1 we know that due to the $p_u^T q_i$ term, the objective function is not convex. So, we know this objective can not guarantee global minima.

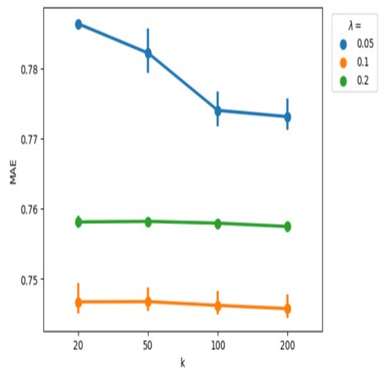
3.2.1 Algorithm 2: SGD [6]

```
**Data:** Training data $R$ contains user $u \in N^+$, item $i \in N^+$ and rating $r_{u,i} \in \mathbb{R}$.
**Input:** $k$, $\lambda, \eta$
**Result:** Latent vector $p_u$ for users and $q_i$ for items.
**Initialize:**
**for** $(u = 1; u \leq m; u^{++})$**do**
    Initialize $p_u$, $b_u$ s.t. every entry is from $\mathcal{N}(0, 0.1)$
**for** $(i = 1; i \leq n; i^{++})$**do**
    Initialize $q_i$, $b_i$ s.t. every entry is from $\mathcal{N}(0, 0.1)$
**while** *the result does not converge* **do**
    Shuffle$(R)$
    **for** every known rating $r_{ui} \in R$ **do**
        $err_{ui} = r_{ui} - p_u^T q_i$
        $b_u \leftarrow b_u + \eta(err_{ui} - \lambda b_u)$
        $b_i \leftarrow b_i + \eta(err_{ui} - \lambda b_i)$
        $p_u \leftarrow p_u + \eta(err_{ui} q_i - \lambda p_u)$
        $q_i \leftarrow q_i + \eta(err_{ui} p_u - \lambda q_i)$
```

This method - Algorithm 2 - was first informally proposed in [7] and many extensions have been proposed ever since [6].



(a) Varying dimension k of feature vectors.



(b) Comparing performance of different λ over k.

Figure 1: Finding the best ALS model by changing values of k, λ.

4 Experimentation

As mentioned before, there were some difficulties during implementation. Due to this, the code used was not our own, but found at GitHub [8] and is written by Ben Lindsay. Many experiments were performed to study the efficiency of the two approaches in terms of prediction accuracy and computation time. For a fair fight, many models of each approach were run, the best-performing model of each approach was used for final comparisons.

4.1 Experimental Setup

The dataset used in this experiment is MovieLens 100k [9]. There are 100,000 known ratings, from 943 users on 1682 movies, and each user has rated at least 20 movies. The accuracy metric used in these experimentations is Mean Absolute Error (MAE), $\frac{\sum_{(u,i)\in known} |r_{ui} - \hat{r}_{ui}|}{|R|}$.

4.2 Finding the best model

For a fair fight, many models of each approach were run, the best-performing model of each approach was used for final comparisons. The difference in these models is simple changes to values of k, λ, and η (for SGD).

4.2.1 Best Model: ALS

With MF methods like ALS, a lower k is favorable for generalizability, so $\lambda = 0.1$ and $k = 50$ is chosen for the best-performing ALS model from Figure 1.

4.2.2 Best Model: SGD

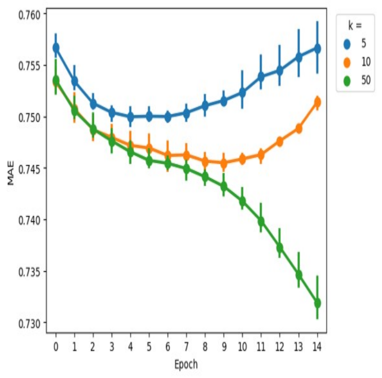
From Figure 2, the best model chosen for SGD is $k = 50, \eta = 0.01, \lambda = 0.01$.

4.3 Comparison of Approaches

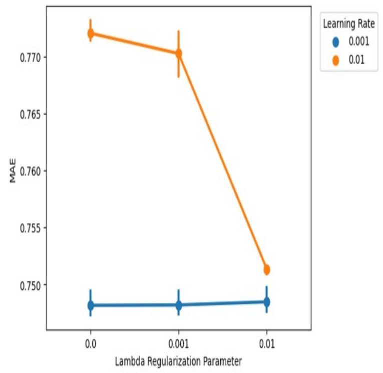
From Figure 3, in both time and MAE, ALS outperforms SGD.

4.4 Results

The results of these experiments show ALS to outperform SGD in both metrics (time and MAE). By looking at the algorithms, however, SGD a much more complicated objective function may take more time to compute, since there are more parameters.



(a) Varying dimension k of feature vectors.



(b) Comparing performance of different learning rate η over λ.

Figure 2: Finding the best SGD model by changing values of k, λ, η

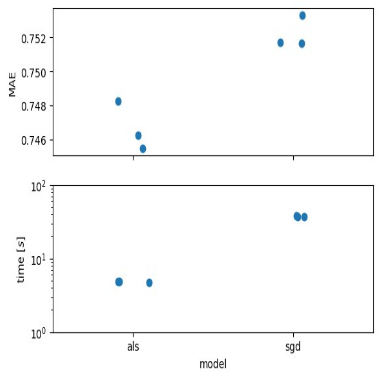


Figure 3: Time and Accuracy comparison of SGD and ALS. Each point represents the value at each of the 3 folds used for cross-validation

4.5 Summary

Now reflecting, some things can be improved from our end. More datasets should be used. One dataset, especially a small one should not be enough to determine performance. Objective functions should be the same. Since time is a performance metric, there should be no difference in $\hat{r}_{ui}$ for each approach. More models for a fairer comparison. Though this paper is mainly about MF models in collaborative filtering, other MF RS techniques should be used for a true comparison in performance. Individual values for λ. In the objective function, an individualized λ for each parameter may result in a faster or more accurate convergence.

5 Conclusion

In this paper, we introduce recommender systems as a desired system for many large content providers and retailers today. Collaborative filtering is discussed as a strategy used in RS, and two approaches in it are Alternating Least Squares and Stochastic Gradient Descent. We prove the matrix factorization problem is not convex. The experimental analysis shows that ALS outperforms SGD in both time and MAE, but we mentioned some caveats to be considered that may change the results of these experiments.

References

[1] Koren, Y. & Bell, R. & Volinsky, C. (2009). Matrix factorization techniques for recommender systems, *Computer*, **8**, pp. 30-37.

[2] Lee, D. & Seung, H. S. (2000). Algorithms for non-negative matrix factorization, *Advances in neural information processing systems,* **13**.

[3] Paatero, P. & Tapper, U (1997) Least squares formulation of robust non-negative factor analysis. *Chemometr. Intell. Lab.* **37**: 23-35.

[4]Yu, T. & Mengshoel, O. J. & Jude, A. & Feller, E. & Forgeat, J. & Radia, N. (2016) Incremental learning for matrix factorization in recommender systems, *2016 IEEE International conference on big data (Big Data)*, pp. 1056-1063. IEEE.

[5] Neumaier, A. (1998) Solving ill-conditioned and singular linear systems: A tutorial on regularization. *SIAM Review* **40**(3), pp. 636-666.

[6] Vinagre, J. & Jorge, A. M. & Gama, J. (2014) Fast incremental matrix factorization for recommendation with positive-only feedback. *UMAP-2014*:459-470.

[7] Funk, S. (2006) http://sifter.org/~simon/journal/20061211.html

[8] Lindsay, B. (2018) https://github.com/benlindsay/movielens-analysis

[9] Harper, F. M. & Konstan, J. A. (2015). The movielens datasets: History and context, *ACM Transactions on Interactive Intelligent Systems (TiiS),* **5**(4), pp. 19.