Bayesian matrix factorization

CHAN Yuk Kit

Abstract

Matrix factorization is a widely used technique in collaborative filtering for recommendation systems. It aims to uncover latent factors that explain the observed user-item interactions, enabling accurate predictions of user preferences. In this paper, I will compare the performance of three popular matrix factorization models: Probabilistic Matrix Factorization (PMF), Constrained Probabilistic Matrix Factorization (CPMF), and Bayesian Probabilistic Matrix Factorization (BPMF).

1 Introduction

In the era of information overload, recommendation systems have become indispensable tools for helping users navigate through vast amounts of content and discover relevant and personalized recommendations. These systems employ various techniques, one of which is collaborative filtering, to provide accurate and effective recommendations based on the collective wisdom of a user community.

Suppose we have a large database of user ratings for different movies. Collaborative filtering can analyze the historical ratings and identify patterns among users with similar movie preferences.For instance, if User A and User B have consistently rated action movies highly, and User A has recently watched and enjoyed a new action movie, the system can recommend this movie to User B based on their shared preferences.

In this paper, we focus on tackling the collaborative filtering problem through three mathematical and statistical solutions(PMF, CPMF, BPMF). We briefly explain the differences in their implementation and compare their performance through real-life experiments. By evaluating their predictive accuracy, scalability, and ability to handle sparse data, we aim to provide insights into the strengths of each solution. Our findings will assist researchers and practitioners in selecting the most suitable matrix factorization model for recommendation systems.

2 Matrix factorization

Low Rank approximation is one of the solution to tackle this problem. Let's say we would like to build an collaborative filtering for a Movie Platform. By using the rating given by different User, we could generate the matrix $R \in \mathbb{R}^{N \times M}$ storing the rating given by Users. N stands for the n different users, while M stands for the m different movies.

Of course, this generated matrix is sparse. Our goals would be predicting appropriate data by collaborative filtering explained above. Instead of directly conducing mathematical or statistical operations on the given matrix, We could generate two matrix $U \in \mathbb{R}^{D \times N}$ and $V \in \mathbb{R}^{D \times M}$. Here the each columns vectors for U and V will be able to store the feature information that the user or the movie have. For each dimension, it may present a specific features that the user may have, like gender, age or etc. It is also worth to note that the dimension of the matrix U and V can be smaller than R through $D \lll \min(N, M)$ while the performance would not be affected much.

Afterwards, We could form a new matrix $\hat{R} = U^T \times V$ and use it to compare the difference on sparse matrix R on known rating. In the experiment part, I will showcase how Stochastic Gradient Descent

(SGD) and proper loss function can help predict the unseen rating. That would be the basic idea of low rank approximation [1] and the matrix factorization.

Singular Value Decomposition (SVD) is one of the method for getting the two matrix U and V. However, related research points out that SVD implementation could not tackle the non-convex optimization problem in some cases [1]. Therefore, I would introduce some related Matrix Factorization supported to tackle this problem.

2.1 Probabilistic Matrix Factorization

Research by Mnih suggested the following likelihood [2]:

$$p(R|U,V,\sigma^2) = \prod_{i=i}^{N}\prod_{j=1}^{M}[\mathcal{N}(R_{ij}|g(U_i^T V_j),\sigma^2)]^{I_{ij}}$$

where σ would be the standard derivation and hyper-parameter that needed to be assigned manually. Also, I_{ij} would be the indicator variable for the observed rating [2]. The g function here would be a sigmoid function limiting the range of $U_i^T V_j$ into [0,1]. Meanwhile, it is also suggested that there should be some prior distribution for matrix

$$p(U|\sigma_U^2) = \prod_{i=i}^{N}\mathcal{N}(U_i|0,\sigma_U^2 I) \qquad p(V|\sigma_V^2) = \prod_{i=i}^{M}\mathcal{N}(V_i|0,\sigma_V^2 I)$$

where the σ_U^2 and σ_V^2 are also variance and manually assigned hyper-parameter

Therefore, by bayes theroem, we could derive the following result:

$$p(U,V|R,\sigma^2,\sigma_U^2,\sigma_V^2) = \frac{p(R|U,V,\sigma^2,\sigma_U^2,\sigma_V^2)\times p(U,V,\sigma^2,\sigma_U^2,\sigma_V^2)}{p(R,\sigma^2,\sigma_U^2,\sigma_V^2)}$$

$$p(U,V|R,\sigma^2,\sigma_U^2,\sigma_V^2) \propto p(R|U,V,\sigma^2)\times p(U,V|\sigma_U^2,\sigma_V^2)$$

$$p(U,V|R,\sigma^2,\sigma_U^2,\sigma_V^2) \propto \prod_{i=i}^{N}\prod_{j=1}^{M}[\mathcal{N}(R_{ij}|g(U_i^T V_j),\sigma^2)]^{I_{ij}}\times p(U|\sigma_U^2)\times p(V|\sigma_V^2)$$

This would be the posterior distribution for U and V after having the observation and the matrix factorization would be a maximum posterior problem.

By applying logarithm and removing unnecessary terms, we could have following loss function:

$$L = \frac{1}{2}\sum_{i=i}^{N}\sum_{j=1}^{M} I_{ij}(R_{ij}-g(U_i^T V_j))^2 + \frac{\lambda_U}{2}\sum_{i=1}^{N}\|U_i\|_{Fro}^2 + \frac{\lambda_V}{2}\sum_{i=1}^{M}\|V_i\|_{Fro}^2$$

As aforementioned, we would like to apply gradient descent to find the optimized solution. Here would be the derivative:

$$\frac{\partial L}{\partial U_i} = -\sum_{j=1}^{N}[I_{ij}(R_{ij}-g(U_i^T V_j)) * g^{'}(U_i^T V_j)V_j] + \lambda_U U_i$$

$$\frac{\partial L}{\partial V_j} = -\sum_{i=1}^{M}[I_{ij}(R_{ij}-g(U_i^T V_j)) * g^{'}(U_i^T V_j)V_i] + \lambda_U U_j$$

One of the problem for directly using PMF is the low performance for users with low rating experience. For users who usually don't rate the movie, their posterior distribution would be basically the same. Therefore, Here comes with Constrained Probabilistic Matrix Factorization.

Moreover, it may also noted that another limitation of this training method is the requirement for manual management of complexity, which is crucial for ensuring the model's ability to generalize effectively, especially when dealing with sparse and imbalanced datasets. One approach to controlling model complexity is to search for appropriate values of the regularization parameters λ_U and λ_V mentioned earlier.

[1] More information for the rank of these matrix: rank($\hat{R} \leq \min(rank(U), rank(V)) \leq D$)

[2] 1: Observed rating, 0: Unobserved rating

2.2 Constrained Probabilistic Matrix Factorization

As aforementioned, the Probabilistic Matrix Factorization could not perform well in cases of User with low frequency on rating. Therefore, constrained Probabilistic Matrix Factorization uses another users' feature vector and introduces the latent similarity constraint matrix $W \in \mathbb{R}^{D\times N}$. The new users' feature vector would then be [1]

$$W_i = Y_i + \frac{\sum_{k=1}^{M} I_{ik} W_k}{\sum_{k=1}^{M} I_{ik}}$$

This new feature vector try to capture the information that whether this user rate a lot or rate a few movies. The W_k matrix here will represent the effect that rating artist k has on the user's feature vector [4]. Generally, we could reach the following result: higher the similarity between features vector from A and feature B, more they would like the same movie. It is noted that matrix Y and W would also follow the zero-mean spherical Gaussian prior. Therefore, we could reach the following result:

$$L = \frac{1}{2}\sum_{i=i}^{N}\sum_{j=1}^{M} I_{ij}(R_{ij} - g(W_i))^2 + \frac{\lambda_Y}{2}\sum_{i=1}^{N}\|Y_i\|_{Fro}^2 + \frac{\lambda_U}{2}\sum_{i=1}^{N}\|U_i\|_{Fro}^2 + \frac{\lambda_V}{2}\sum_{i=1}^{M}\|V_i\|_{Fro}^2$$

where $\lambda_Y = \frac{\sigma^2}{\sigma_Y^2}$, $\lambda_U = \frac{\sigma^2}{\sigma_U^2}$ and $\lambda_V = \frac{\sigma^2}{\sigma_V^2}$

It is same as PMF, I would also perform stochastic gradient descent to determine the performance of this model.

2.3 Bayesian Probabilistic Matrix Factorization

As aforementioned in PMF part, We have to manually adjust the hyper-parameter by validation sets. Here, in Bayesian Probabilistic Matrix Factorization, we hope to build a model that can choose the hyper-parameter by itself and no manual adjustment. It will be conducted with Markov Chain Monte Carlo (MCMC).

The likelihood of Bayesian Probabilistic Matrix Factorization would be same as what PMF introduce. However, the distribution for feature matrix U and V would no longer the zero mean. Here are the prior distribution for two matrix [3].

$$p(U|\mu_u, \Lambda_u) = \prod_{i=1}^{N}\mathcal{N}(U_i|\mu_u, \Lambda_u^{-1}) \qquad p(V|\mu_v, \Lambda_v) = \prod_{i=1}^{N}\mathcal{N}(V_i|\mu_v, \Lambda_v^{-1})$$

Where the Λ_u and Λ_v would be the inverse of covariance matrix, i.e. the precision matrix. Also, as aforementioned, We hope that the hyper-parameter do not need to be adjusted manually. Here the hyper-parameter, the mean and precision matrix will follow Gaussian-Wishart priors [3].

$$p(\Theta_U|\Theta_0) = p(\mu_U|\Lambda_U)p(\Lambda_U) = \mathcal{N}(\mu_U|\mu_0, (\beta_0\Lambda_U)^{-1})\mathcal{W}(\Lambda_U|W_0, \nu_0)$$
$$p(\Theta_V|\Theta_0) = p(\mu_V|\Lambda_V)p(\Lambda_V) = \mathcal{N}(\mu_V|\mu_0, (\beta_0\Lambda_V)^{-1})\mathcal{W}(\Lambda_V|W_0, \nu_0)$$

The prediction probability for certain user i on certain movies J would be:

$$p(R_{ij}^*|R, \Theta) = \int\int p(R_{ij}^*|U_i, V_j)p(U, V|R, \Theta_U, \Theta_V)p(\Theta_U, \Theta_V|\Theta_0)$$

that would then be converted into

$$p(R_{ij}^*|R, \Theta) = \mathbb{E}_{U,V\sim f(U_i,V_j|\Theta_U,\Theta_V,\Theta_0)}[p(R_{ij}^*|U_i, V_j)]$$
$$p(R_{ij}^*|R, \Theta) \approx \frac{1}{n}\sum_{k}^{n} p(R_{ij}^*|U_i^k, V_j^k)$$

These samples could be obtained from Gibbs Sampling in Markov Chain Monte Carlo (MCMC).

Algorithm 1 Gibbs Sampling on matrix U

Inputs:
  T
Initialize:
  $U^0 \leftarrow [U_1^0 U_2^0 \dots U_n^0], \quad U_i^0 = \vec{0}, \quad i = 1, \dots, n$
for t = 1 to T **do**
  for j = 1 to n **do**
    $U_j^t \leftarrow$ sample from $P(U_j^t | R, U^t, V, \theta_U, \sigma)$
  end for
end for

2.3.1 Gibbs Sampling in MCMC

MCMC here would actually try to do a sampling on the matrix U and V. Also, due to the high dimensional of matrix and simple calculation on the conditional probability of U and V, here we will use Gibbs Sampling which is the special case of Metropolis-Hasting Algorithm.

The basic ideas of the implementation Gibbs Sampling would be presented in the pseudo-code (Algorithm 1). Here,with the use of Bayes theorem, the $P(U_j^t | R, U^t, V, \theta_U, \alpha)$ could be expressed to

$$\prod_{j=1}^{M} [\mathcal{N}(R_{ij} | U_i^T V_j, \sigma^2)]^{I_{ij}} * \mathcal{N}(U_i | \mu_u, \Lambda_u)$$

However, we also note that it should be normalized by an integrating factor.

$$(P(U_j^t | R, U^t, V, \theta_U, \alpha) = f = \frac{\prod_{j=1}^{M} [\mathcal{N}(R_{ij} | U_i^T V_j, \sigma^2)]^{I_{ij}} * \mathcal{N}(U_i | \mu_u, \Lambda_u)}{Z} = \frac{g}{Z}$$

We could use the idea from Metropolis-Hasting Algorithm. Through introducing a proposal distribution, like some normal distribution, we have the following algorithm

Algorithm 2 Metropolis-Hasting Algorithm

Inputs:
  Conditional Proposal distribution q
$z^{t+1} \leftarrow$sample from Conditional Proposal distribution $q(z|z^t)$
$a \leftarrow \mathrm{U}[0, 1]$
$b \leftarrow \min(\frac{f(z^{t+1}) * q(z^{t+1}|z^t)}{f(z^t) * q(z^t|z^{t+1})}) = \min(\frac{g(z^{t+1}) * q(z^{t+1}|z^t)}{g(z^t) * q(z^t|z^{t+1})})$
if $a \leq b$ **then**
  $z^t \leftarrow z^{t+1}$
end if

Therefore, we could easily sample each feature vector by a multivariate normal distribution and erase the integrating factor.

Moreover, due to the Bayesian structure, the hyper-parameter μ_u, Λ_u is also the random variable, following the Gaussian-Wishart distribution. Here is the complete algorithm on sampling.

After completing the sampling (algorithm 2), we will obtain a array of matrix {(U^1, V^1),(U^2, V^2) ... (U^T, V^T)}. We then could use these samples to predict the unknown rating through the aforementioned formula,

$$p(R_{ij}^* | R, \Theta) \approx \frac{1}{n} \sum_{k}^{n} p(R_{ij}^* | U_i^k, V_j^k)$$

It is also worth noting that we can select {(U^2, V^2),(U^4, V^4) ... (U^{2k}, V^{2k})} as the real sample to ensure the independent relationship.

Algorithm 3 Complete sampling on matrix U& V

Inputs:
T
Initialize:
$U^0 \leftarrow [U_1^0 U_2^0 \dots U_n^0], \;\; U_i^0 = \vec{0}, \;\; i = 1, \dots, n$
Initialize:
$V^0 \leftarrow [V_1^0 V_2^0 \dots V_n^0], \;\; V_i^0 = \vec{0}, \;\; i = 1, \dots, n$
for t = 1 to T **do**
  $\theta_U \leftarrow$ sample from $\mathcal{N}(\mu_U|\mu_0, (\beta_0\Lambda_U)^{-1})\mathcal{W}(\Lambda_U|W_0, \nu_0)$
  $\theta_V \leftarrow$ sample from $\mathcal{N}(\mu_V|\mu_0, (\beta_0\Lambda_V)^{-1})\mathcal{W}(\Lambda_V|W_0, \nu_0)$
  for j = 1 to n **do**
    $U_j^t \leftarrow$ sample from $P(U_j^t|R, U^t, V, \theta_U, \sigma)$ ▷ Using algo 1
  end for
  for j = 1 to m **do**
    $V_j^t \leftarrow$ sample from $P(V_j^t|R, V^t, U, \theta_V, \sigma)$ ▷ Using algo 1
  end for
end for

3 Experiments

3.1 Description

In this experiment, we test different matrix factorization, from standard PMF to Bayesian PMF as mentioned in this paper. Then we compare how different matrix factorization impact a recommendation system.

3.2 Architecture and Data

The architecture of the network would follow the suggestions made in those papers. The parameters would be updated either through stochastic gradient descent or MCMC. [3] Additionally, due to computational constraints, I will conduct the experiment using a small dataset instead of a large one.

3.2.1 PMF & CPMF

There are some chosen hyper-parameter for developing the model in PMF & CPMF.

Regarding the mean of distribution and D value of PMF and CPMF, we will directly follow what the author suggested, setting $\mu = 0$ and $D = 30$ [4] under standard and constrained PMF. Yet, for the variance, we would have the following assignment:

$$\lambda_U^{pmf} = 0.001 \quad and \quad \lambda_V^{pmf} = 0.0001$$
$$\lambda_Y^{cpmf} = \lambda_U^{cpmf} = \lambda_V^{cpmf} = 2$$

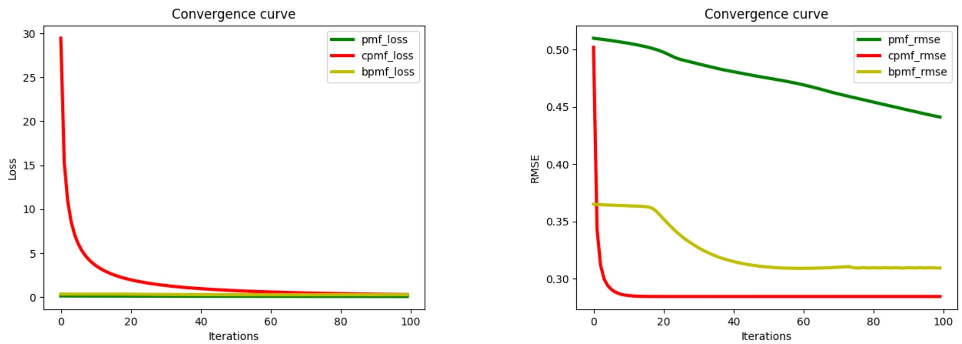
These are some suggested result through several cross validation. Those model tend to have good performance under this setting.

3.2.2 Bayesian PMF

Regarding the D value of BPMF, we will also select 30 so as to have a fair competition among these three models. For more detail and hyper-parameters suggested in [3], we have the following assignments.

$$W_0^U = W_0^V = I \quad and \quad \mu_0^U = \mu_0^V = \vec{0}$$
$$\frac{1}{\sigma^2} = \alpha = 2 \quad and \quad \beta_0^U = \beta_0^V = 2$$

[3]The code will be shown on the files.
[4]It is much smaller than the $\min{(N, M)} = \min{(610, 9742)}$



(a) Loss curve

(b) RMSE curve from testing phase

Figure 1: Experiments on different PMF model

3.2.3 General Detail

During the training and testing, the movie rating will be converted to scale of 0 to 1. By using the formula suggested by author, $\hat{r} = \frac{r-0.5}{4.5}$ will be used for determining the degree of the rating. Also, noted that the root mean square error will also base on this scale. Therefore, the RMSE here will be in range of $[0, 1]$.

3.3 Result

Here will be the discussion on the result of these three different models.

3.3.1 PMF vs CPMF

As aforementioned, it is clearly shown that PMF and constrained PMF share a similar architecture and algorithm, which is stochastic gradient descent. The only difference is the additional consideration of the effects from rating, represented by the matrix $W \in \mathbb{R}^{D \times N}$. . However, we can clearly observe that the performance of CPMF is much better than PMF. In Figure 1, PMF shows weak performance in terms of RMSE (0.44) during the testing phase. It does not demonstrate strong predictability and tends to suffer from overfitting. In comparison, Constrained PMF performs much better (0.22) and reaches the local minimum at a faster rate. This demonstrates a different result as shown by the author [2]. Under the passages about the comparison between PMF and constrained PMF. This result support that films which have been watched but don't have known ratings are a valuable source of information, especially for users who appear multiple times in the test set and have only a few ratings in the training set. The constrained PMF model can effectively incorporate this information. The matrix $Win\mathbb{R}^{D \times N}$ does help the model give more accurate result.

3.3.2 PMF vs BPMF

On the other hand, Bayesian PMF also perform a better result than standard PMF. We also obtain a similar result as shown in [3]. We can clearly see that Bayesian PMF could achieve a better result (0.3) than standard PMF (0.44), with 30% improvement in prediction.

These could be suggested that setting the hyper-parameters, i.e. the mean and variance, as random variable could not only reduce the timing in selection on hyper-parameter for standard PMF, but also provide a significantly higher predictive accuracy. Although it is hard for us to determine whether the MCMC is reaching the desired distribution or not, we still can confirm that BPMF would be a better choice than PMF in term of prediction.

4 Conclusion

In conclusion, this project experimented the performance of different PMF, including the standard Probabilistic Matrix Factorization (PMF), Constrained Probabilistic Matrix Factorization (CPMF), and Bayesian Probabilistic Matrix Factorizationb(BOMF). The results conducted under this project show that both CPMF and BPMF perform better than the standard PMF model in terms of predictive accuracy. CPMF effectively use the information from films with unknown ratings, while BPMF, by adopting a Bayesian approach, provides higher predictive accuracy. Thus, for improved prediction performance, using either CPMF or BPMF is recommended over the standard PMF model.

References

[1] Nathan Srebro and Tommi Jaakkola. Weighted low-rank approximations. In Tom Fawcett and Nina Mishra, editors, Machine Learning, Proceedings of the Twentieth International Conference (ICML 2003), August 21-24, 2003, Washington, DC, USA, pages 720–727. AAAI Press, 2003.

[2] A. Mnih and R. R. Salakhutdinov, 'Probabilistic Matrix Factorization', in Advances in Neural Information Processing Systems, 2007, vol. 20.

[3] R. Salakhutdinov and A. Mnih, 'Bayesian probabilistic matrix factorization using Markov chain Monte Carlo', Helsinki, Finland, 2008, pp. 880–887.