The Experiment Report of Machine Learning

SCHOOL: SCHOOL OF SOFTWARE ENGINEERING

SUBJECT: SOFTWARE ENGINEERING

Author:
Xiaoxuan Peng

Supervisor:
Mingkui Tan

Student ID：
201730683321

Grade:
Undergraduate

November 4, 2019

Recommender System Based on Matrix Decomposition

Abstract—This experiment intends to use ALS or SGD or other methods to implement a recommender system based on matrix decomposition.

I. INTRODUCTION

Recommender system is a common technique to recommend something such as movies, goods for us based on our favourites. In the experiment, I use SGD to implement a small recommender system based on matrix decomposition.

II. METHODS AND THEORY

A. Collaborative Filtering

Make automatic predictions (filtering) about the interests of a user by collecting preferences or taste information from many other users (collaboration).

There are some types of collaborative filtering.

a) Memory-based CF: utilize the entire user-item database to generate a prediction.

b) Model-based CF: build a model from the rating data (Matrix factorization, etc.) and use this model to predict missing ratings.

In this experiment, I use model-based CF.

B. Matrix Factorization

• Give a rating matrix $\mathbf{R} \in \Re^{m\text{x}\,n}$, with sparse ratings from m users to n items.

• Assume rating matrix $\mathbf{R}$ can be factorized into the multiplication of two low-rank feature matrices $\mathbf{P} \in \Re^{\text{mx}\,k}$ And $\mathbf{Q} \in \Re^{k\text{x}\,n}$

C. Stochastic Gradient Descent(SGD) in Matrix Factorization

• SGD is to minimize the following objective function:

$$\mathcal{L} = \sum_{u,i \in \Omega} (r_{u,i} - \mathbf{p}_u^T \mathbf{q}_i)^2 + \lambda_p ||\mathbf{p}_u||^2 + \lambda_q ||\mathbf{q}_i||^2$$

Notes:

$$\mathbf{P} = [\mathbf{p}_1, \mathbf{p}_2, \cdots, \mathbf{p}_m]^T \in \Re^{m\text{x}\,k}$$

$$\mathbf{Q} = [\mathbf{q}_1, \mathbf{q}_2, \cdots, \mathbf{q}_n] \in \Re^{k\text{x}\,n}$$

$r_{u,i}$ denotes the actual rating of user u for item i

Ω denotes the set of observed samples from rating matrix $\mathbf{R}$

λ_p , λ_q are regularization parameters to avoid overfitting

Algorithm 3 General Steps of SGD

1: **Require** feature matrices $\mathbf{P}$, $\mathbf{Q}$, observed set Ω, regularization parameters λ_p, λ_q and learning rate α.
2: **Randomly** select an observed sample $r_{u,i}$ from observed set Ω.
3: Calculate the **gradient** w.r.t to the objective function.
4: **Update** the feature matrices $\mathbf{P}$ and $\mathbf{Q}$ with learning rate α and gradient.
5: **Repeat** the above processes until **convergence**.

• Calculate the gradient and update the feature matrices

Objective function:

$$\mathcal{L} = (r_{u,i} - \mathbf{p}_u^T \mathbf{q}_i)^2 + \lambda_p ||\mathbf{p}_u||^2 + \lambda_q ||\mathbf{q}_i||^2$$

We randomly select an observed sample $r_{u,i}$

And then we calculate the prediction error:

$$E_{u,i} = r_{u,i} - \mathbf{p}_u^T \mathbf{q}_i$$

Also, we calculate the gradient:

$$\frac{\partial \mathcal{L}}{\partial \mathbf{p}_u} = E_{u,i}(-\mathbf{q}_i) + \lambda_p \mathbf{p}_u$$

$$\frac{\partial \mathcal{L}}{\partial \mathbf{q}_i} = E_{u,i}(-\mathbf{p}_u) + \lambda_q \mathbf{q}_i$$

Finally, we update the feature matrices **P** and **Q** with **learning rate** α:

$$\mathbf{p}_u = \mathbf{p}_u + \alpha(E_{u,i}\mathbf{q}_i - \lambda_p \mathbf{p}_u)$$

$$\mathbf{q}_i = \mathbf{q}_i + \alpha(E_{u,i}\mathbf{p}_u - \lambda_q \mathbf{q}_i)$$

Algorithm 4 SGD Algorithm

1: **Require** feature matrices $\mathbf{P}$, $\mathbf{Q}$, observed set Ω, regularization parameters λ_p, λ_q and learning rate α.
2: **Randomly** select an observed sample $r_{u,i}$ from observed set Ω.
3: Calculate the **gradient** w.r.t to the objective function:

$$E_{u,i} = r_{u,i} - \mathbf{p}_u^T \mathbf{q}_i$$
$$\frac{\partial \mathcal{L}}{\partial \mathbf{p}_u} = E_{u,i}(-\mathbf{q}_i) + \lambda_p \mathbf{p}_u$$
$$\frac{\partial \mathcal{L}}{\partial \mathbf{q}_i} = E_{u,i}(-\mathbf{p}_u) + \lambda_q \mathbf{q}_i$$

4: **Update** the feature matrices $\mathbf{P}$ and $\mathbf{Q}$ with learning rate α and gradient:

$$\mathbf{p}_u = \mathbf{p}_u + \alpha(E_{u,i}\mathbf{q}_i - \lambda_p \mathbf{p}_u)$$
$$\mathbf{q}_i = \mathbf{q}_i + \alpha(E_{u,i}\mathbf{p}_u - \lambda_q \mathbf{q}_i)$$

5: **Repeat** the above processes until **convergence**.

III. EXPERIMENT

A. Dataset

This dataset is MovieLens-100k dataset. u.data in the dataset contains 10000 ratings on 1682 movies from 943 users. Each user has rated at least 20 movies. user_id and item_id are marked with the beginning of number 1. Also, the data is distributed randomly.

B. Implementation

(1) Initialization

In this experiment, I initialize the initial rating matrix with all zero and I initialize users matrix P and items matrix Q by random distribution method.

(2) Parameters

Table 1 shown below shows all the parameters in the experiment.

Table 1-SGD on matrix decomposition

Parameters	Values
Learning rate	0.0002
Number of iterations	50
Penalty factor	100
Number of potential features	150

(3) Results

Figure 1 shown below shows the result using the parameters listed above.

Figure 1-Loss value result

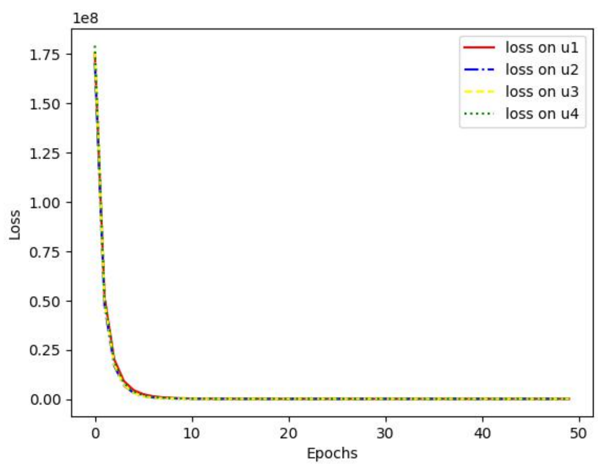


Figure 2 shown below shows the result using a smaller potential features. Here, potential features is 50. Initial loss value is small.

Figure 2-Loss value result(Smaller potential features)

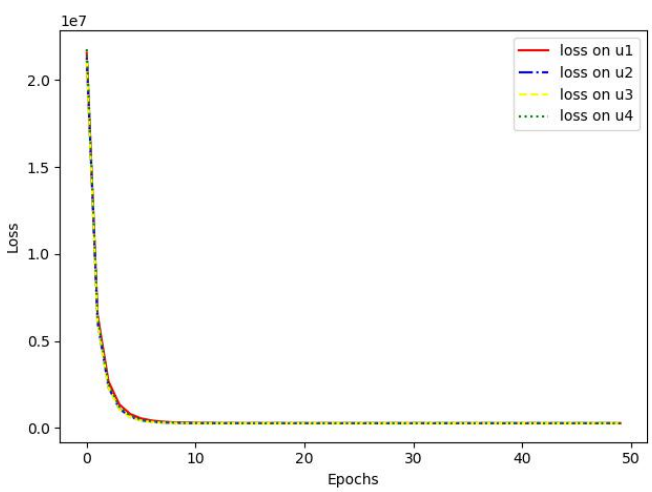


Figure 3 shown below shows the result using a larger potential features. Here, potential features is 350. Initial loss value is large.

Figure 3-Loss value result(Larger potential features)

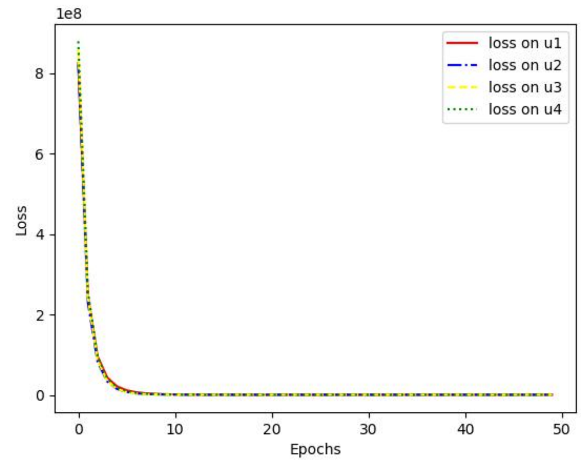


Figure 4 shown below shows the result using a larger penalty factor. Here, penalty factor is 800. Initial loss value is small and it is very fast to converge.

Figure 4-Loss value result (Larger penalty factor)

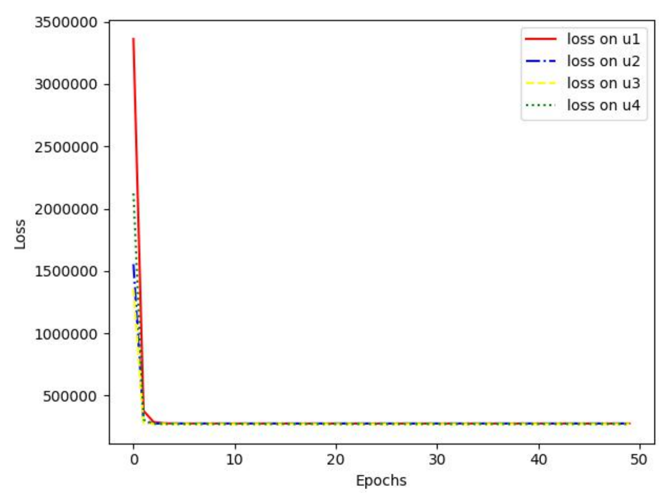


Figure 5 shown below shows the result using a smaller penalty factor. Here, penalty factor is 0.5. Initial loss value is large and it is slow to converge.

Figure 5-Loss value result(Smaller penalty factor)

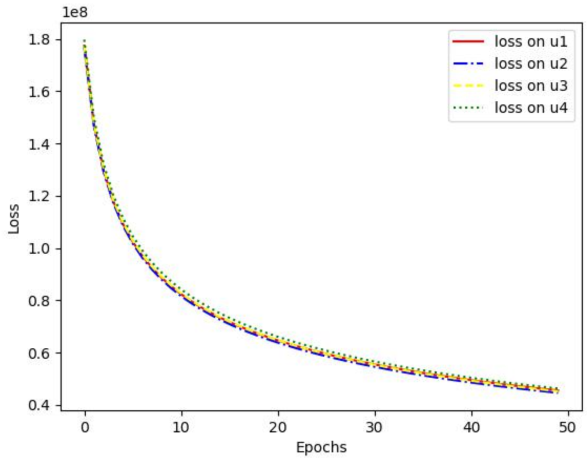


Figure 6 shown below shows the result using a smaller learning rate(LR). Here, learning rate is 1e-7. Initial loss value is very large and it is very slow to converge.

Figure 6-Loss value result(Smaller learning rate)

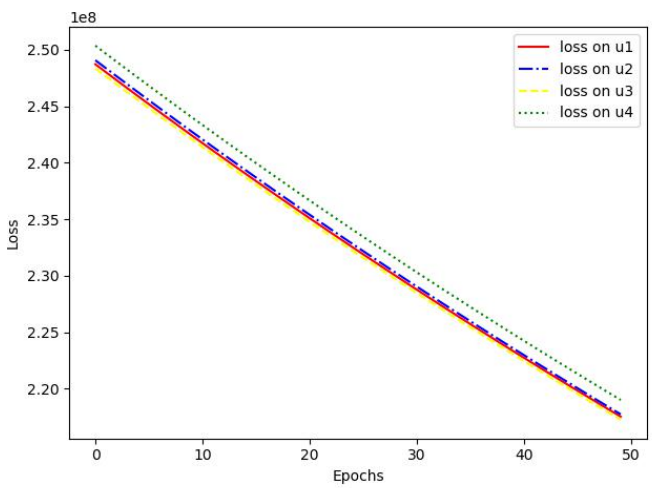
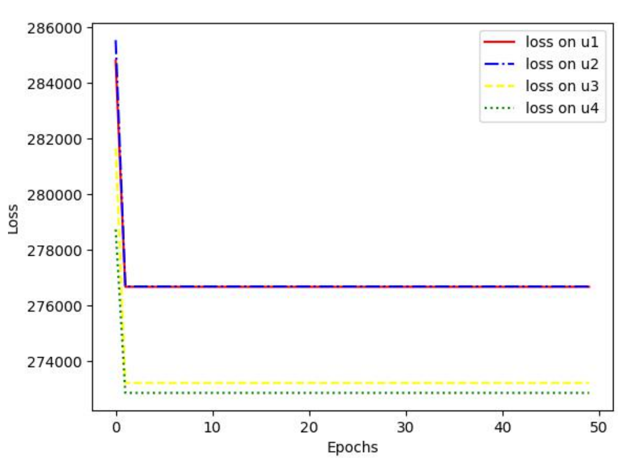


Figure 7 shown below shows the result using a larger learning rate(LR). Here, learning rate is 0.01. Initial loss value is small and it is very fast to converge.

Figure 7-Loss value result(Larger learning rate)



IV. CONCLUSION

(1) Recommender system is a common technique to recommend something useful for users. There are many methods and models to implement a recommender system such as ALS, SGD, SVD.

(2) Parameters will play an important role in the performance of the models. A large learning rate will make the loss function faster to converge or it won't be converged. A small learning rate will make the loss function slower to converge. A large potential features will make the initial loss value larger. A small potential features will make the initial loss value smaller. Also, a large penalty factor will make the initial loss value smaller and it will be faster to converge in this experiment. A small penalty factor will make the initial loss value larger and it will be much slower to converge.