华南理工大学

South China University of Technology

The Experiment Report of *Machine Learning*

SCHOOL: SCHOOL OF SOFTWARE ENGINEERING

SUBJECT: SOFTWARE ENGINEERING

Author:
Junwen Mo, Huantong Li and Zhiwei Lin

Supervisor:
Mingkui Tan

Student ID:
201530612545, 201530611944 and
201530612316

Grade:
Undergraduate

January 6, 2018

Recommender System Based on Matrix Decomposition

***Abstract*—This report is about the Recommender System Based on Matrix Decomposition. By introducing the basic idea of Recommender System and the method of Matrix Decomposition, we try to build up a model that can fit the data.**

I. Introduction

THIS experiment mainly needs us to solve the problem about the Recommender System. Recommender System applies statistical and knowledge discovery techniques to the problem of making product recommendations. It help a lot in our daily life. The basic algorithm is Collaborative Filtering(CF). The motivations of this experiment are as below: 1)Explore the construction of recommended system; 2)Understand the principle of matrix decomposition; 3)Be familiar to the use of gradient descent; 4)Construct a recommendation system under small-scale dataset, cultivate engineering ability.

II. Methods and Theory

In this experiment, we try using the stochastic gradient descent(SGD) and alternate least squares optimization(ALS) to implement the matrix decomposition and then compare their performance. The goal function is as equation (1). And we use equation (2) and (3) ti update the matrix P and Q in the stochastic gradient descent.

$$e_{ij}^2 = (r_{ij} - \sum_{k=1}^{K} p_{ik} qkj)^2 + \frac{\beta}{2} \sum_{k=1}^{K} (||P||^2 + ||Q||^2) (1)$$

$$p_{ik}^{'} = p_{ik} + \alpha \frac{\partial}{\partial p_{ik}} e_{ij}^2 = p_{ik} + \alpha (2 e_{ij} q_{kj} - \beta p_{ik}) (2)$$

$$q_{kj}^{'} = q_{kj} + \alpha \frac{\partial}{\partial q_{kj}} e_{ij}^2 = q_{kj} + \alpha (2 e_{ij} p_{ik} - \beta q_{kj}) (3)$$

In the ALS, we can make the $\frac{\partial e^2}{\partial p_u} = 0$ and get the equation(4). At last we can get the equation (5) to update every row of P. In the same way, we can use $\frac{\partial e^2}{\partial q_i} = 0$ to get the equation (6) and (7),which is used in the update of every column of Q.

$$\sum_i (-r_{ui} Q_i^T + P_u Q_i Q_i^T) + \lambda P_u = 0 (4)$$

$$P_u = \sum_i (-r_{ui} Q_i^T)(\sum_i Q_i Q_i^T + \lambda E)^{-1} (5)$$

$$\sum_u (-r_{ui} P_u^T + P_u^T P_u Q_i) + \lambda Q_i = 0 (6)$$

$$Q_i = (\sum_u P_u^T P_u + \lambda E)^{-1} \sum_u (-r_{ui} P_u^T) (7)$$

III. Experiments

A. Dataset

1. Utilizing MovieLens-100k dataset.

2. u.data – Consisting 10,000 comments from 943 users out of 1682 movies. At least, each user comment 20 videos. Users and movies are numbered consecutively from number 1 respectively. The data is sorted randomly

3. u1.base / u1.test are train set and validation set respectively, seperated from dataset u.data with proportion of 80% and 20%. It also make sense to train set and validation set from u1.base / u1.test to u5.base / u5.test.

TABLE I
Data Samples

user id	item id	rating	timestamp
196	242	3	881250949
186	302	3	891717742
22	377	1	878887116
244	51	2	880606923
166	346	1	886397596

4. You can also construct train set and validation set according to your own evaluation method.

B. Implementation

1) Experiment Steps:

Using stochastic gradient descent method(SGD):

1. Read the data set and divide it (or use u1.base / u1.test to u5.base / u5.test directly). Populate the original scoring matrix R_{n_users,n_items} against the raw data, and fill 0 for null values.

2. Initialize the user factor matrix $P_{n\_users,K}$ and the item (movie) factor matrix $Q_{n\_items,K}$, where K is the number of potential features.

3. Determine the loss function and hyperparameter learning rate η and the penalty factor λ.

4. Use the stochastic gradient descent method to decompose the sparse user score matrix, get the user factor matrix and item (movie) factor matrix:

4.1 Select a sample from scoring matrix randomly;

4.2 Calculate this sample's loss gradient of specific row(column) of user factor matrix and item factor matrix;

4.3 Use SGD to update the specific row(column) of $P_{n\_users,K}$ and $Q_{n\_items,K}$;

4.4 Calculate the $L_{validation}$ on the validation set, comparing with the $L_{validation}$ of the previous iteration to determine if it has converged.

5. Repeat step 4. several times, get a satisfactory user factor matrix P and an item factor matrix Q, Draw a $L_{validation}$ curve with varying iterations.

6. The final score prediction matrix $\widehat{R}_{n\_users,n\_items}$ is obtained by multiplying the user factor matrix $P_{n\_users,K}$ and the transpose of the item factor matrix $Q_{n\_items,K}$.

2) Experiment Results and Analysis:

In our experiment, the hyper-parameter K we used is 2. In the SGD, the learning rate is 0.03 and the regularization parameter is 0.005. The loss curve of ALS is as Fig. 1 and the loss curve of SGD is as Fig. 2. We can see that the convergence speed in ALS is fast, while the convergence speed in SGD is slow. But ALS use the whole dataset in every epoch while SGD only use a sample. So ALS consume a lot of memory but it can make good use of the advantage of the matrix operation.

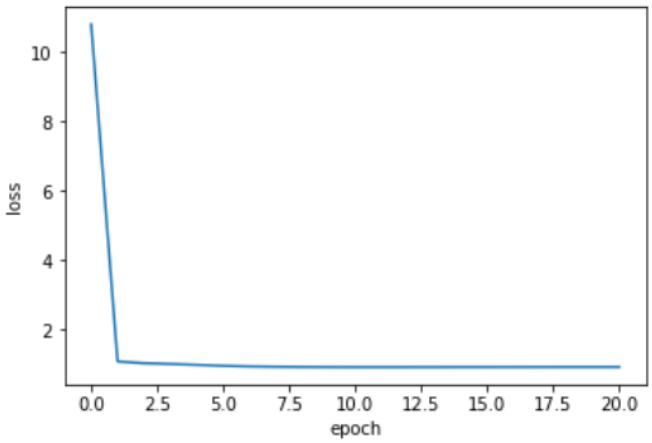

Fig. 1. The loss value of ALS with the increment of the epoch.

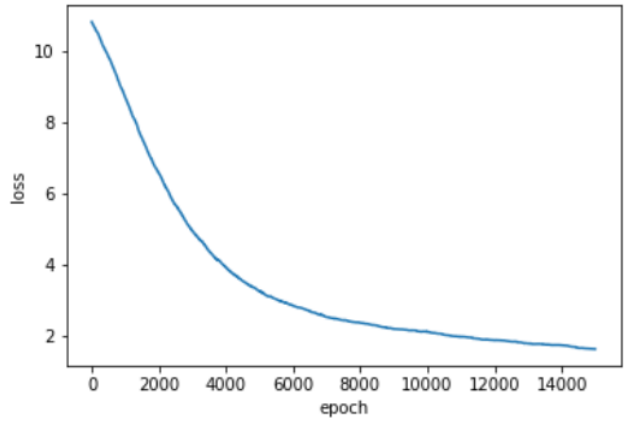

Fig. 2. The loss value of stochastic gradient descent with the increment of the epoch.

IV. Conclusion

In this experiment, we use matrix decomposition to finish the rating prediction. Our team try completing it with 2 methods, ALS and stochastic gradient descent. Convergence of ALS is much faster than that of stochastic gradient descent. But the calculation speed of ALS will be slower when the dataset become so large that more memory is required.