South China University of Technology

The Experiment Report of Machine Learning

School: School of Software Engineering

Subject: Software Engineering

Author:
Yongliang Wen and Liangyu Xiao and Kexin Li

Supervisor:
Qingyao Wu

Student ID:
201530613023 and 201530613146 and 201530611999

Grade:
Undergraduate

December 28, 2017

Use ALS and SGD to Build Recommender System Based on Matrix Decomposition

Abstract—This paper will show us two methods ALS and SGD to slove the problem of building a Recommender System based on Matirx Decomposition. We present the algorithm and the effect of the two methods.

I. Introduction

RECOMMENDER System is widely used in industry. So it is important for us to complete the code of the general methods–ALS and SGD.The motivation we have four: Explore the construction of recommended system. Understand the principle of matrix decomposition. Be familiar to the use of gradient descent. Construct a recommendation system under small-scale dataset, cultivate engineering ability.

II. Methods and Theory

ALS: The alternating least squares algorithm is used through out the industry of recommmender systems. The key insight is that you can turn the non-convex optimization problem in the minimize problem into an easy quadratic problem if you fix either one variable or the other variable. ALS fixes each one of those alternatively. When one is fixed, the other one is computed, and vice versa.

SGD: In an SGD (Stochastic Gradient descent) approach, for each example in the dataset you compute the error and then you update the parameters by a factor in the opposite direction of the gradient.

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 0.8 and 0.2. It also make sense to train set and validation set from u1.base / u1.test to u5.base / u5.test.

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

TABLE I
Example data of dataset

$user_id$	$item_id$	$rating$	$timestamp$
196	242	3	881250949
186	302	3	891717742
22	377	1	878887116
244	51	2	880606923
166	346	1	886397596

B. Implementation

1) Using Alternate Least Squares Optimization Method(ALS): Give a rating matrix $\mathbf{R} \in \mathbb{R}^{m\times n}$,with sparse ratings from m users to n items, We can assume that $\mathbf{R}$ can be factorized into the multiplication of two low-rank feature matrices $\mathrm{P} \in \mathbb{R}^{m\times k}$ and $\mathrm{Q} \in \mathbb{R}^{k\times n}$ We can get a new matrix $\mathbf{R}_{new}$ by multiplying P and Q and minimize the loss between $\mathbf{R}_{new}$ and $\mathbf{R}$

ALS is to minimize the following objective function:

$$\mathrm{L} = \sum_{u,i}(r_{u,i}-\mathrm{p}_u^T\mathrm{q}_i)^2+\lambda(\sum_u n_{\mathrm{p}_u}\|\mathrm{p}_u\|^2+\sum_i n_{\mathrm{q}_i}\|\mathrm{q}_i\|^2) \quad (1)$$

$\mathrm{P}=[\mathrm{p}_1, \mathrm{p}_2, ..., \mathrm{p}_m]^T \in \mathbb{R}_{m\times k}$

$\mathrm{Q}=[\mathrm{q}_1, \mathrm{q}_2, ..., \mathrm{q}_n] \in \mathbb{R}_{k\times m}$

$\mathrm{r}_{u,i}$ denotes the actual rating of user u for item i.

λ is regularization parameter to avoid over fitting.

n_{p_u} and n_{q_i} denote the number of total ratings on user u and item i, respectively

```
Algorithm 1 ALS
1: Require rating matrix R, feature matrices P, Q and regularization parameter  .
2: Optimize P while fixing Q.
3: Optimize Q while fixing P.
4: Repeat the above processes until convergence
```

$$\frac{\partial \mathrm{L}}{\partial \mathrm{p}_u} = 0 \quad (2)$$

$$\mathrm{p}_u = (\sum \mathrm{q}_i\mathrm{q}_i^T + \lambda n_{\mathrm{p}_u}\mathrm{I})^{-1} \cdot \mathrm{Q}^T \cdot \mathrm{R}_{u*}^T \quad (3)$$

$$\mathrm{q}_i = (\sum \mathrm{p}_u\mathrm{p}_u^T + \lambda n_{\mathrm{q}_i}\mathrm{I})^{-1} \cdot \mathrm{P}^T \cdot \mathrm{R}_{*i} \quad (4)$$

In (1), we compute the gradient of objective function on p_u. Then make it equals to zeros to get p_u's expression by fixing Q. q_i can be obtained by the same method.

from Fig. 2, which is the result of experiment with λ=0.9,k=10,losses drop down in a high speed.

2) Using Stochastic Gradient Descent method(SGD): Give a rating matrix $\mathbf{R} \in \mathbb{R}^{m\times n}$,with sparse ratings from m users to n items, We can assume that $\mathbf{R}$ can be factorized into the multiplication of two low-rank feature matrices $\mathrm{P} \in \mathbb{R}^{m\times k}$ and $\mathrm{Q} \in \mathbb{R}^{k\times n}$ We can get a new matrix $\mathbf{R}_{new}$ by multiplying P and Q and minimize the loss between $\mathbf{R}_{new}$ and $\mathbf{R}$

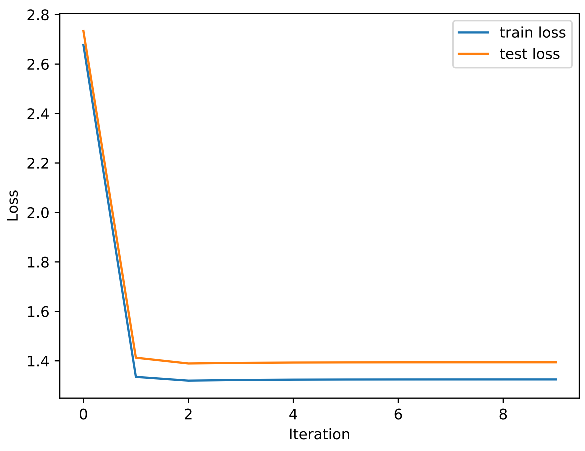

Fig. 1. train loss and test loss by ALS.

SGD is to minimize the following objective function:

$$L = \sum_{u,i\in\Omega} (r_{u,i} - \mathrm{p}_u^T \mathrm{q}_i)^2 + \beta_\mathrm{p}\|\mathrm{p}_u\|^2 + \beta_\mathrm{q}\|\mathrm{q}_i\|^2 \tag{5}$$

P=$[\mathrm{p}_1, \mathrm{p}_2, ..., \mathrm{p}_m]^T \in \mathbb{R}_{m\times k}$
Q=$[\mathrm{q}_1, \mathrm{q}_2, ..., \mathrm{q}_n] \in \mathbb{R}_{k\times m}$
Ω denotes the set of observed samples from rating matrix R.
$\mathrm{r}_{u,i}$ denotes the actual rating of user u for item i.
β_p, β_q is regularization parameter to avoid over fitting.

```
Algorithm 2 SGD
1: Require feature matrices P, Q, observed set $\Omega$, regularization parameters $\beta_\mathrm{p}$, $\beta_\mathrm{q}$ and learning rate $\alpha$.
2: Randomly select 100 observed sample $\mathrm{r}_{u,i}$ from observed set $\Omega$.
3: Calculate the gradient to the objective function.
4 Update the feature matrices P and Q with learning rate $\alpha$ and gradient.
5: Repeat the above processes until convergence.
```

Calculate the prediction error:

$$\mathrm{E}_{u,i} = r_{u,i} - \mathrm{p}_u^T \mathrm{q}_i \tag{6}$$

Calculate the gradient:

$$\frac{\partial \mathrm{L}}{\partial \mathrm{p}_u} = \mathrm{E}_{u,i}(-\mathrm{q}_i) + \beta_p \mathrm{p}_u \tag{7}$$

$$\frac{\partial \mathrm{L}}{\partial \mathrm{q}_i} = \mathrm{E}_{u,i}(-\mathrm{p}_u) + \beta_q \mathrm{q}_i \tag{8}$$

Update the feature matrices P and Q with learning rate α:

$$\mathrm{p}_u = \mathrm{p}_u - \alpha\frac{\partial \mathrm{L}}{\partial \mathrm{p}_u} \tag{9}$$

$$\mathrm{q}_i = \mathrm{q}_i - \alpha\frac{\partial \mathrm{L}}{\partial \mathrm{q}_i} \tag{10}$$

from Fig. 2, which is the result of experiment with α=0.,k=2,β_q=0.1,β_q=0.1.We only use ml-100k/u1.base and ml-100k/u1.test in this experiment.

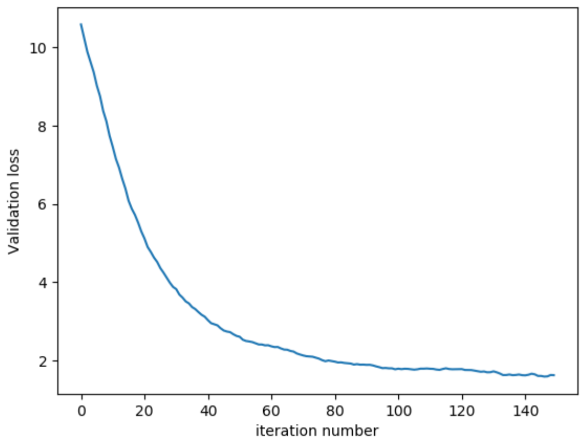

Fig. 2. validation loss by SGD.

IV. Conclusion

We can see easily SGD is faster than ALS, but SGD has a little disturbance in the update processes.And ALS need less iteration than SGD to get the convergence.This experiment make me learn more and it is interesting.