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:
Dongming Sheng, Haokun Li and Yu Guo

Supervisor:
Mingkui Tan

Student ID:
201530612699 , 201530611883 and 201530611500

Grade:
Undergraduate

December 27, 2017

Recommender System Based on Matrix Decomposition

***Abstract*—In this experiment, we conduct a simple recommender system based on matrix decomposition with stochastic gradient descent method. We use part of MovieLens 100k dataset to train and validate our model. The result shows that by the increasing iteration times of SGD optimization the recommender system performs better.**

I. Introduction

RECOMMENDER system is a widely-used technique in daily life. This area is closely related to machine learning. In this experiment, we simplify this problem to implement a recommender system in a small-scale dataset. In order to solve it, we use matrix composition and use stochastic gradient descent to optimize the result. We choose python to implement algorithm since it has convenient libraries to use for featuring engineering and modeling.

II. Methods and Theory

A. Matrix Decomposition

In the mathematical discipline of linear algebra, Matrix Dcomposition is a factorization of a matrix into a product of matrices. Matrix Decomposition is the most widely used algorithm in Recommender System problem. Give a rating matrix $R \in R^{m\times n}$, with sparse ratings from m users to n items. Assume rating matrix R can be factorized into the multiplication of two low-rank feature matrices $P \in R^{m\times k}$ and $Q \in R^{k\times n}$. We have:

$$R \in R^{m\times n} \approx P \in R^{m\times k} \times Q \in R^{k\times n} \tag{1}$$

B. Stochastic Gradient Descent for MD

SGD is scalable to large-scale datasets when ALS is not scalable to large-scale datasets, and this experiment uses a big data set. So we use SGD to fix matrix P and Q. SGD is to minimize the following objective function:

$$L = \sum_{u,i\in\Omega} (r_{u,i} - p_u^T q_i)^2 + \lambda_p ||p_u||^2 + \lambda_q ||q_i||^2 \tag{2}$$

When $P = [p_1, p_2, \cdots, p_m]^T \in R^{m\times k}$, $Q = [q_1, q_2, \cdots, q_n] \in R^{k\times n}$. $r_{u,i}$denotas the set of observed samples from rating matrix R, Ωdenotes the set of observed samples from rating matrix R. λ_p and λ_q are regularization parameters to avoid overtting.

Then, we randomly select an observed sample $r_{u,i}$, and calculate the prediction error:

$$E_{u,i} = r_{u,i} - P_u^T q_i \tag{3}$$

Calculate the gradient:

$$\begin{aligned} \frac{\partial L}{\partial p_u} &= E_{u,i}(-q_i) + \lambda_p p_u \\ \frac{\partial L}{\partial q_i} &= E_{u,i}(-p_u) + \lambda_q q_i \end{aligned} \tag{4}$$

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

$$\begin{aligned} p_u &= p_u + \alpha(E_{u,i} q_i - \lambda_p p_u) \\ q_i &= q_i + \alpha(E_{u,i} p_u - \lambda_q q_i) \end{aligned} \tag{5}$$

Algorithm 1 SGD Algorithm

Require: feature matrices P, Q, observed set Ω, regularization parameters λ_p, λ_q and learning rate α.

Ensure: $Predict = PQ$

1: **Randomly** select an observed sample $r_{u,i}$ from observed setΩ.

2: Calculate the **gradient** to the objective function:

$$E_{u,i} = r_{u,i} - P_u^T q_i$$

$$\begin{aligned} \frac{\partial L}{\partial p_u} &= E_{u,i}(-q_i) + \lambda_p p_u \\ \frac{\partial L}{\partial q_i} &= E_{u,i}(-p_u) + \lambda_q q_i \end{aligned} \tag{6}$$

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

$$\begin{aligned} p_u &= p_u + \alpha(E_{u,i} q_i - \lambda_p p_u) \\ q_i &= q_i + \alpha(E_{u,i} p_u - \lambda_q q_i) \end{aligned} \tag{7}$$

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

III. Experiments

A. Dataset

In this experiment, we use data from *MovieLens-100k* dataset, which consists of 10,000 comments from 943 users out of 1682 movies. Each user, labeled consecutively from number 1 to 943 respectively, comments at least 20 movies. The movies are labeled in the same way. *u1.base* and *u1.test* are our training set and test set, which are randomly selected from *u.data* with proportion of 80% and 20% respectively. The structure of our dataset is shown in Table I.

B. Implementation

We implement the experiment in the following order:

TABLE I
STRUCTURE OF OUR DATASET *MovieLens-100k*

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

1) Main Structure:

- Load training set *u1.base* and test set *u1.test*. Use 0 to fill the null values in original scoring matrix R_{n_users,n_items}.
- Initialize 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.
- Determine the loss function and hyperparameter η learning rate and the penalty factor λ.
- Use stochastic gradient descent(SGD) method to decompose the sparse user score matrix, get the user-factor matrix and item (movie)-factor matrix.
- Repeat the previous step several times to get a satisfactory user-factor matrix and an item-factor matrix. Draw the L_{test} curve.
- The final score prediction matrix $\hat{R}_{n\_users,n\_items}$ 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) Procedure of Stochastic Gradient Descent Method:

- Select a sample from scoring matrix R_{n_users,n_items} randomly.
- Calculate the gradient of this sample with respect to a specific row(column) of user-factor matrix and item-factor matrix.
- Use SGD to update the specific row(column) of $P_{n\_users,k}$ and $Q_{n\_items,k}$.
- Calculate the L_{test} on the validation set, and compare it with the L_{test} in the previous iteration to determine whether it has converged.

The hyper-parameters we choose are listed in Table II. Fig 1 is the L_{test} curve with 20000 iterations. In the figure, we can observe that test loss is decreasing in a similar tendency as the test curve, which suggests a similar distribution between training set and testing set.

TABLE II
HYPER-PARAMETER SELECTED FOR MATRIX FACTORIZATION

K	10
regularization_coeffient	0.01
learning_rate	0.005
max_iteration	20000

IV. CONCLUSION

In this experiment, we implement matrix decomposition with stochastic gradient descent in python under small-scale dataset and try to build a recommender system. We explore the construction of recommended system, understand the principle of matrix decomposition and become familiar with optimizing it with gradient descent. The result shows that matrix composition performs well.

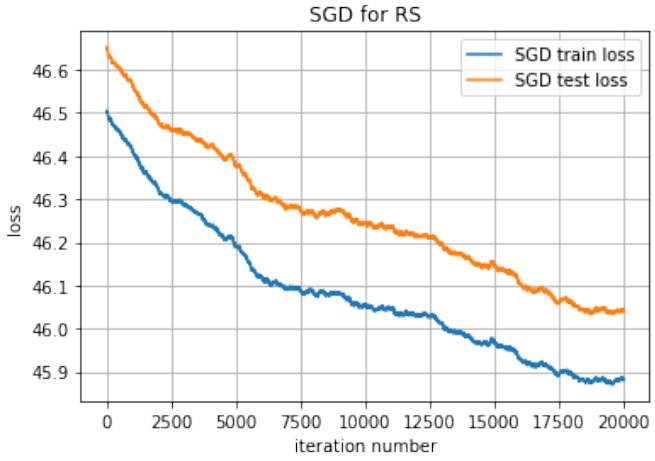

Fig. 1. The loss curves of SGD method.