South China University of Technology

The Experiment Report of *Machine Learning*

SCHOOL: SCHOOL OF SOFTWARE ENGINEERING

SUBJECT: SOFTWARE ENGINEERING

Author:
Jining He and Hui Han

Supervisor:
Qingyao Wu

Student ID:
201721045565 and 201721045572

Grade:
Graduate

December 29, 2017

Recommender System Based on Matrix Factorization

***Abstract*—In this experiment, we build a recommender system based on matrix factorization. We train the model on MovieLens-100k dataset. The result shows that Matrix Factorization is a useful model-based collaborative filtering algorithm.**

I. Introduction

A recommender system is a subclass of information filtering system that seeks to predict the "rating" or "preference" that a user would give to an item. Recommender systems have become increasingly popular in recent years, and are utilized in a variety of areas including movies, music, news, books, research articles, search queries, social tags, and products in general. In this experiment, we use matrix factorization algorithm to build a recommender system.

II. Methods and Theory

Matrix Factorization is the most widely used model-based collaborative filtering algorithm.

Given a rating matrix R of size $m \times n$, with sparse ratings from m users to n items. We assume matrix R can be factorized into the multiplication of two low-rank feature matrices P of size $m \times K$ and Q of size $K \times n$.

To solve this, we define following objective function:

$$L = (r_{u,i} - p_u^T q_i)^2 + \lambda_p \|p_u\|^2 + \lambda_q \|q_i\|^2 \qquad (1)$$

We use SGD to optimize this object function. We also need to calculate prediction error:

$$E_{u,i} = r_{u,i} - p_u^T q_i \qquad (2)$$

And gradient:

$$\begin{aligned} p_u &= p_u + \alpha(E_{u,i} q_i - \lambda_p p_u) \\ q_i &= q_i + \alpha(E_{u,i} p_u - \lambda_q q_i) \end{aligned} \qquad (3)$$

The whole SGD algorithm is following:

Algorithm 1 Matrix Factorization SGD Algorithm

1: **Require** Feature matrices **P**, **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:

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

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

$$p_u = p_u + \alpha(E_{u,i} q_i - \lambda_p p_u)$$

$$q_i = q_i + \alpha(E_{u,i} p_u - \lambda_q q_i)$$

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

III. Experiments

A. Dataset

We use MovieLens-100k dataset which consists 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. In this dataset, u1.base / u1.test are train set and validation set respectively, seperated from the dataset with proportion of 80% and 20%

B. Implementation

There are many algorithms to solve this problem. We use stochastic gradient descent(SGD). The steps is following:

1. Read the dataset. We use u1.base / u1.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 α 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:
   (1) Select a sample from scoring matrix randomly;
   (2) Calculate this sample's loss gradient of specific row(column) of user factor matrix and item factor matrix;

(3) Use SGD to update the specific row(column) of $P_{n\_users,K}$ and $Q_{n\_items,K}$;
(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 $\hat{R}_{n\_users,n\_items}$ is obtained by multiplying the user factor matrix $P_{n\_users,K}$ and the transpose of the item factor matrix $Q_{n\_items,K}$.

C. Result

We did a lot of experiments and found that the model would overfit when the regularization term was small. We found parameters that would fit the model, as Table. I

TABLE I
PARAMETERS

learning rate	0.1
λ_p	10
λ_q	10

We plot the loss on validation set as Fig. 1. We can see that as the number of iterations increases, loss decreases. This tell us SGD is a useful optimize algorithm for matrix factorization

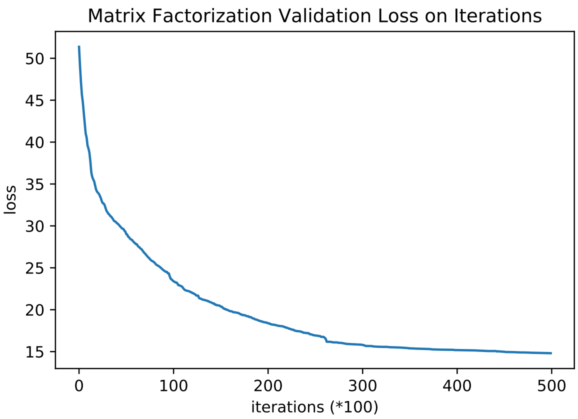

Fig. 1. SGD algorithm loss on validation set.

IV. CONCLUSION

In this experiment, we implemented a recommendation system based on matrix factorization. This gives is a deeper understanding of the principles of recommendation systems. At the same time, we also consolidated the knowledge of matrix factorization and stochastic gradient descent algorithm.