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:
Junteng Pang Zhaohui Huang Lilong Huang

Supervisor:
Qingyao Wu

Student ID：201720145174
201720145112 2017201449620

Grade:
Graduate

December 29, 2017

Reconmender System Based on Matrix Decomposition

Abstract—The experimental matrix decomposition model is a method of collaborative filtering, the general principle is to pass a certain number of factors to describe the preferences of each user and the properties of each item. In this paper, two matrices are obtained through the decomposition and optimization of stochastic gradient descent method, one is the user factor matrix and the other is the item factor matrix. Multiply these two matrices to get a predictive score for all movies for all users, and then verify the validity of the experiment by experiment.

I. INTRODUCTION

The recommendation system based on stochastic gradient descent method is based on content recommendation. The system designer does not need to know the content attribute of each item and each user's preference. The factor matrix obtained by matrix factorization can describe the attributes of users and objects well Of course, this comes from the user's historical behavior, the more the user's past behavior, the more accurate the recommendation.

In this experiment we use the matrix decomposition of the loss function and use the stochastic gradient descent (SGD), one of the best optimization algorithms available today, to optimize the loss function.

The main purpose of this report can be summarized as follows:

- Analyze the composition of the recommended system.
- Understand the principles of matrix decomposition.
- Familiar with stochastic gradient descent method and use.
- Build a dataset-based recommendation system.

II. METHODS AND THEORY

•Matrix Decomposition

Matrix decomposition is the decomposition of a matrix into two or more matrices. For the above user-product matrix (score matrix), it is denoted by $R_{m\times n}$. We can decompose it into the product of two or more matrices. Suppose decomposed into two matrices $P_{m\times k}$ and $Q_{k\times n}$, we want to make the product of matrices $P_{m\times k}$ and $Q_{k\times n}$ restore the original matrices $R_{m\times n}$:

$$R_{m\times n} \approx P_{m\times k} \times Q_{k\times n} = \hat{R}_{m\times n}$$

Among them, the matrix $P_{m\times k}$ represents the relationship between m users and k topics, while the matrix $Q_{k\times n}$ represents the relationship between k topics and n products.

•Loss Function

In the above matrix decomposition, the original scoring matrix $R_{m\times n}$ is decomposed into the product of two matrices $P_{m\times k}$ and $Q_{k\times n}$,Then the next question is how to solve each element of the matrix $P_{m\times k}$ and $Q_{k\times n}$, this problem can be transformed into machine learning regression problem to solve.

We can use the square of the error between the original scoring matrix $R_{m\times n}$ and the reconstructed scoring matrix $\hat{R}_{m\times n}$ as the loss function, that is:

$$e_{i,j}^2 = \left(r_{i,j} - \hat{r}_{i,j}\right)^2 = \left(r_{i,j} - \sum_{k=1}^{K} p_{i,k} q_{k,j}\right)^2$$

•Solution of loss function

For the above squared loss function, it can be solved by the gradient descent method. The core step of the gradient descent method is to solve the negative gradient of the loss function:

$$\frac{\partial}{\partial p_{i,k}} e_{i,j}^2 = -2\left(r_{i,j} - \sum_{k=1}^{K} p_{i,k} q_{k,j}\right) q_{k,j} = -2e_{i,j} q_{k,j}$$

$$\frac{\partial}{\partial q_{k,j}} e_{i,j}^2 = -2\left(r_{i,j} - \sum_{k=1}^{K} p_{i,k} q_{k,j}\right) p_{i,k} = -2e_{i,j} p_{i,k}$$

then update the variable according to the direction of the negative gradient, through iteration until the algorithm eventually converges:

$$p_{i,k}' = p_{i,k} - \alpha \frac{\partial}{\partial p_{i,k}} e_{i,j}^2 = p_{i,k} + 2\alpha e_{i,j} q_{k,j}$$

$$q_{k,j}' = q_{k,j} - \alpha \frac{\partial}{\partial q_{k,j}} e_{i,j}^2 = q_{k,j} + 2\alpha e_{i,j} p_{i,k}$$

•Add regular part

Usually in the process of solving, in order to be able to have a good generalization ability, a regular term is added to the loss function to constrain the parameters, adding the loss function of L2 regularity as:

$$E_{i,j}^2 = \left(r_{i,j} - \sum_{k=1}^{K} p_{i,k} q_{k,j}\right)^2 + \frac{\beta}{2} \sum_{k=1}^{K} \left(p_{i,k}^2 + q_{k,j}^2\right)$$

The core step of the gradient descent method is to solve the negative gradient of the loss function:

$$\frac{\partial}{\partial p_{i,k}} E_{i,j}^2 = -2\left(r_{i,j} - \sum_{k=1}^{K} p_{i,k} q_{k,j}\right) q_{k,j} + \beta p_{i,k} = -2e_{i,j} q_{k,j} + \beta p_{i,k}$$

$$\frac{\partial}{\partial q_{k,j}} E_{i,j}^2 = -2\left(r_{i,j} - \sum_{k=1}^{K} p_{i,k} q_{k,j}\right) p_{i,k} + \beta q_{k,j} = -2e_{i,j} p_{i,k} + \beta q_{k,j}$$

then update the variable according to the direction of the negative gradient, through iteration until the algorithm eventually converges:

$$p_{i,k}{}' = p_{i,k} - \alpha\left(\frac{\partial}{\partial p_{i,k}} e_{i,j}^2 + \beta p_{i,k}\right) = p_{i,k} + \alpha\left(2e_{i,j} q_{k,j} - \beta p_{i,k}\right)$$

$$q_{k,j}{}' = q_{k,j} - \alpha\left(\frac{\partial}{\partial q_{k,j}} e_{i,j}^2 + \beta q_{k,j}\right) = q_{k,j} + \alpha\left(2e_{i,j} p_{i,k} - \beta q_{k,j}\right)$$

Using the above process, we can get the matrices $P_{m\times k}$ and $Q_{k\times n}$, so that the product j can be scored for user i

III. Experiment

In this section, we conduct some experiments for further understanding of matrix decomposition algorithm.

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 are stored randomly as followings: In this dataset, u1.base and u1.test are train set and valida- tion set respectively, seperated from the whole dataset u.data with proportion of 80% and 20%.

B. Implementation In this experiment, we use stochastic gradient descent(SGD) algorithm to optimize the loss function. The steps of our experiments are as the following:

1) Read the data set from indicated folder. We use u1.base as trianing data and u1.test as testing data directly. For convenient, we populate the original scoring matrix R against the raw data and fill 0 for null values.

2) Initialize the user factor matrix P and the item (movie) factor matrix Q, where K is the number of potential features.

3) Determinethelossfunctionandhyperparameter learning rate η and the penalty factor λ.

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

• Select a sample from scoring matrix randomly;

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

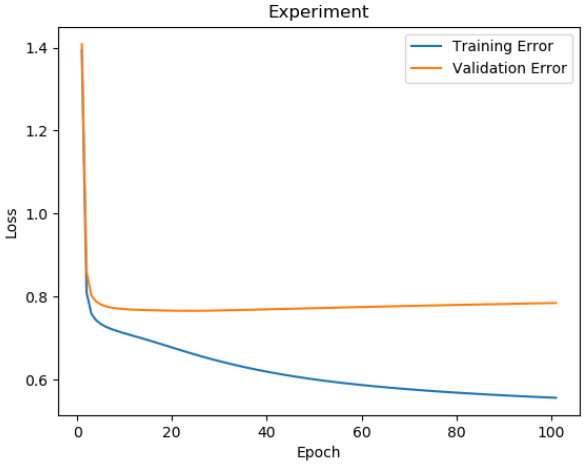
• Use SGD to update the specific row(column) of P and Q;

• Calculate the Lvalidation on the validation set, com- paring with the Lvalidation of the previous iteration to determine if it has converged.

5) Repeat step 4 several times until the validation loss has converged to get a satisfactory user factor matrix P and an item factor matrix Q.

6) The final score prediction matrix ˆ R is obtained by multiplying the user factor matrix P and the transpose of the item factor matrix Q.

In this section, we conducted several experiments to compare the performance of using different hyperparameters in this matrix factorization algorithm. In the experiment, we set the learning rate α = 0.0025, the regularization weight β = 0.025, the hyper-parameter K =10 and perform 100 iterations, using the average error to judge the performance.



Obviously, we observe that the loss of validation decreases first during the minimization of training loss. However, with the passage of time, there is almost no change in the verification loss, even with reduced training losses. This shows that the optimization algorithm has converged.

IV. Conclusion

In this report, we discuss matrix decomposition techniques that compress sparse matrices and optimize matrices that use machine learning optimization algorithms.