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:
胡永浩 袁振宇

Supervisor:
Qingyao Wu

Student ID：
201530611630 and 201530613542

Grade:
Undergraduate

December 12, 2017

Recommendation System Based on Matrix Decomposition

Abstract—Explore the matrix decomposition techniques used in recommendation systems

I. INTRODUCTION

The data source in the recommendation system is mainly the user-item matrix, and the system's job is to predict the missing values in the matrix. Using SGD to decompose the known matrix and then multiply the resulting matrix factor to get the prediction matrix is the content of this experiment

II. METHODS AND THEORY

Gradient decent：Stochastic gradient descent

1: **Require** feature matrices $\mathbf{P}$, $\mathbf{Q}$, 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:

$$E_{u,i} = r_{u,i} - \mathbf{p}_u^\top \mathbf{q}_i$$
$$\frac{\partial \mathcal{L}}{\partial \mathbf{p}_u} = E_{u,i}(-\mathbf{q}_i) + \lambda_p \mathbf{p}_u$$
$$\frac{\partial \mathcal{L}}{\partial \mathbf{q}_i} = E_{u,i}(-\mathbf{p}_u) + \lambda_q \mathbf{q}_i$$

4: **Update** the feature matrices $\mathbf{P}$ and $\mathbf{Q}$ with learning rate α and gradient:

$$\mathbf{p}_u = \mathbf{p}_u + \alpha(E_{u,i}\mathbf{q}_i - \lambda_p \mathbf{p}_u)$$
$$\mathbf{q}_i = \mathbf{q}_i + \alpha(E_{u,i}\mathbf{p}_u - \lambda_q \mathbf{q}_i)$$

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

Loss function：Root Mean Square Error（RMSE）

$$\text{RMSE} = \sqrt{\sum_{u,i \in \Omega} (\hat{r}_{u,i} - r_{u,i})^2 / |\Omega|}$$

III. EXPERIMENT

A.Dataset：

Train：Choose randomly from u1.base--u5.base

Test：Choose randomly from u1.test--u5.test

B.Implementation:

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 $Q_{n\_item,K}$, 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 $Q_{n\_item,K}$;
   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 $\hat{R}_{n\_users,n\_items}$ is obtained by multiplying the user factor matrix $P_{n\_users,K}$ and the transpose of the item factor matrix $Q_{n\_item,K}$.

ps: matrix factor P, Q using random initialization

IV. CONCLUSION

If using ALS to solve this problem,since every update of the parameters traverses all the sample data once, this is a complete gradient descent with higher accuracy, but it takes a lot of time to traverse the m-sets. When the training set is large, The method is a waste of time, so lead to a Stochastic gradient descent,for each update of parameters, do not have to traverse all training sets, using only one data, to transform a parameter. This is not as accurate as a complete gradient decline, and may take many detours, but the overall trend is toward minimum. This saves more time and the algorithm is faster.