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:
Dingkun song, shaojia hong,shuai zou

Supervisor:
Qingyao Wu

Student ID：201720145006，
201720144993，201720145167

Grade:
Undergraduate

December 21, 2017

Recommendation system based on matrix decomposition

Abstract—Construct a recommendation system under a small-scale dataset using the matrix decomposition,use alternate least squares or stochastic gradient descent method to optimize the descent.

I. INTRODUCTION

A. Recommender System

Recommender System applies statistical and knowledge discovery techniques to the problem of making product recommendations.

B. Matrix decomposition

Matrix decomposition refers to the decomposition of a matrix into the product of two or more matrices.The objective of matrix decomposition is to decompose the user - project score matrix R into the form of user factor matrix and project factor matrix multiplication.like:

$$\mathrm{R}_{m\times n} = P_{m\times k} \times Q_{k\times n}$$

each row of P would represent the strength of the associations between a user and the features. Similarly, each row of Q would represent the strength of the associations between an item and the features.

C. Gradient Descent

Identify a set of hypotheses f(x;w) and define a loss function l(w),then we pick the best w^* by minimizing the loss function.This is call Gradient Decent

D. Regularization

A common extension to this basic algorithm is to introduce regularization to avoid overfitting. This is done by adding a parameter β and modify the squared error.

II. METHODS AND THEORY

Firstly, we have a set U of users, and a set D of items. Let R of size U × D be the matrix that contains all the ratings that the users have assigned to the items. we assume that we would like to discover K latent features. Our task, then, is to find two matrics matrices P (a |U| × K matrix) and Q (a |D| × K matrix) such that their product approximates R:

R=P × Q

we have to find a way to obtain P and Q.One way to approach this problem is the first intialize the two matrices with some values, calculate how different their product is to R, and then try to minimize this difference iteratively. Such a method is called gradient descent, aiming at finding a local minimum of the difference.

The difference here, usually called the error between the estimated rating and the real rating, can be calculated by the following equation for each user-item pair:

$$\mathrm{e}_{ij}^2 = (r_{ij} - \hat{r}_{ij})^2 = (r_{ij} - \sum_{k=1}^{K} p_{ik} q_{kj})^2$$

Having obtained the gradient, we can now formulate the update rules for both p_{ik} and q_{kj} :

$$\mathrm{p}'_{ik} = p_{ik} + \alpha \frac{\partial}{\partial p_{ik}} e_{ij}^2 = p_{ik} + 2\alpha e_{ij} q_{kj}$$

$$\mathrm{q}'_{kj} = q_{kj} + \alpha \frac{\partial}{\partial q_{kj}} e_{ij}^2 = q_{kj} + 2\alpha e_{ij} p_{ik}$$

Add a parameter β and modify the squared error to avoid overfitting as follows:

$$\mathrm{e}_{ij}^2 = (r_{ij} - \sum_{k=1}^{\mathrm{K}} p_{ik} q_{kj})^2 + \frac{\beta}{2} \sum_{k=1}^{K} (\| P \|^2 + \| Q \|^2)$$

p_{ik} and q_{kj} update rules are as follows:

$$\mathrm{p}'_{ik} = p_{ik} + \alpha \frac{\partial}{\partial p_{ik}} e_{ij}^2 = p_{ik} + \alpha(2 e_{ij} q_{kj} - \beta p_{ik})$$

$$\mathrm{q}'_{kj} = q_{kj} + \alpha \frac{\partial}{\partial q_{kj}} e_{ij}^2 = q_{kj} + \alpha(2 e_{ij} p_{ik} - \beta q_{kj})$$

III. EXPERIMENT

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 80% and 20%. It also make sense to train set and validation set from u1.base / u1.test to u5.base / u5.test.

B. Implementation

The Initialization value is showned as the table:

K	5
Step	500
Alpha	0.0002
Beta	0.02

Then I use the formula above to calculate loss function e^2 and update p_{ik} and q_{kj} until the iteration to the step value 500.

I use two array ,e_train and e_test to save the loss of training set and test set.

Finally,I use matplotlib to show the e_train and e_test.

IV. CONCLUSION

Using the gradient,as the number of training increases,the loss is getting smaller and smaller.Finally tends to be smooth. This is mean we find the best w.
The result is showned as follow.