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:Zeshuai Deng、Yue Su、
Jingchao Huang

Supervisor:Qingyao Wu

Student ID：201530611425、
201530612774、201530611722

Grade:Undergraduate

December 9, 2017

Linear Regression, Linear Classification and Gradient Descent

Abstract—

I. Introduction

1. Explore the construction of recommended system.

2. Understand the principle of matrix decomposition.

3. Be familiar to the use of gradient descent.

4. Construct a recommendation system under small-scale dataset, cultivate engineering ability.

II. Methods and Theory

ALS： ALS is to compute the commodity characteristic matrix and user characteristic matrix alternately by fixing the user characteristic matrix and commodity characteristic matrix respectively.ALS algorithm is not like user based or item based collaborative filtering algorithm. It predicts and recommends similarity by calculating similarity, ALS uses matrix decomposition to predict user's score of movie

SGD：

Gradient Descent is a very classic technique used to find the minimum (sometimes local) value of a function.If you've heard of the back propagation for training neural networks, then BackProp is just a gradient - computing technology that is later used for gradient descent. SGD is one of the one billion varieties of gradient descen.

III. Experiment

Using alternate least squares optimization(ALS):

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 the hyperparameter learning rate η nd the penalty factor λ.

4. Use alternate least squares optimization method to decompose the sparse user score matrix, get the user factor matrix and item (movie) factor matrix:

4.1 With fixd item factor matrix, find the loss partial derivative of each row (column) of the user factor matrices, ask the partial derivative to be zero and update the user factor matrices.

4.2 With fixd user factor matrix, find the loss partial derivative of each row (column) of the item factor matrices, ask the partial derivative to be zero and update the item

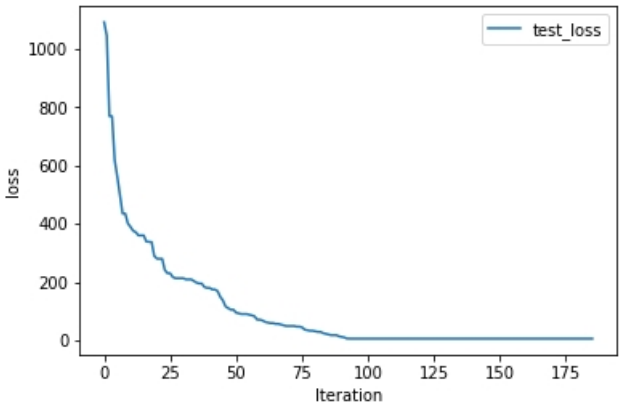
4.3 Calculate the $L_{validation}$ on the validation set, comparing with the $L_{validation}$ of the previous iteration to determine if it has converged.

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 $Q_{n\_item,K}$.

```
# parameters use to train with
params = {
    "train_file": "./ml-100k/u2.base",
    "test_file": "./ml-100k/u2.test",
    "k": 150,
    "beta": 1,
    "user_number": 943,
    "item_number": 1682,
    "learning_rate": 0.0001,
    "epoch": 2
}
main(**params)
```



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;

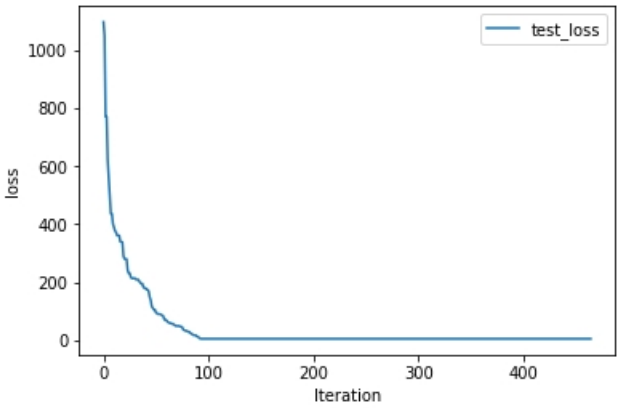
4.3 Use SGD to update the specific row(column) of $P_{n\_users,K}$ and $Q_{n\_item,K}$;

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 $Q_{n\_item,K}$.

```
# parameters use to train
params = {
    "train_file": "./ml-100k/u2.base",
    "test_file": "./ml-100k/u2.test",
    "k": 150,
    "beta": 1,
    "user_number": 943,
    "item_number": 1682,
    "learning_rate": 0.0001,
    "epoch": 5
}
main(**params)
```



IV. CONCLUSION

ALS：Use ALS to optimize the loss function. The idea of the algorithm is that we generate it randomly and fix it, then gain the solution fixedly. This is going on alternately until the

optimal solution is obtained. Because each iteration will reduce the error, and the error has lower bounded, so ALS will converge. But the problem is not convex, ALS does not guarantee that it converges to the global optimal solution. But in practical applications, ALS is not very sensitive to the initial point, and the effect of the global optimal solution is not significant.

SGD：

Recommendation system based on sSGD method is based on the content recommendation, but the system designers need not know every items of content attributes and each user preferences, the factor matrix derived by matrix decomposition can describe the attributes of users and items well.Of course, these come from the historical behavior of users, and the more historical behavior is, the more accurate it is.Because it's a collaborative filtering method, there is also a problem of cold start.For example, if a new product is not interacting with any user, it will never be recommended, and it can't be recommended to a new registered user, because there is no historical behavior.Secondly, there is a sparsity problem. When there is few interaction between the user and the object, the recommendation results are often not accurate.