Project Report

Recommendation system

item-based,user based

This Recommendation system uses item-base , user-base ,and KNN method to recommend items to customers.
In user-base, the similarity between different users is defined by
$a_{uv} = \frac{|N(u) \bigcap N(v)|}{\sqrt{N(u)*N(v)}}$
Then according to knn method, choose k neighbors and predict the pretend customer's interest in the product.
$p(u, x) = \sum(a_{uv} * r_{vx})$
Almost the same in the user-based method
$b_{ij} = \frac{|M(i) \bigcap M(j)|}{\sqrt{M(i)*M(j)}}$
$p(u, x) = \sum(b_{xi} * r_{ui})$

Actually, the rating matrix is always sparse. we try to use ALS algorithm to solve the problem.

Matrix factorization

1,Each user can be described as k features,hidden topics
2,Each item can be described by k features
3,If we multiply each feature of the user by the corrsponding feature of movies and add them together,this will be a good approximation for the model.

$r_{ui} = x_u^T * y_i = \sum(x_{uk} * y_{ki})$

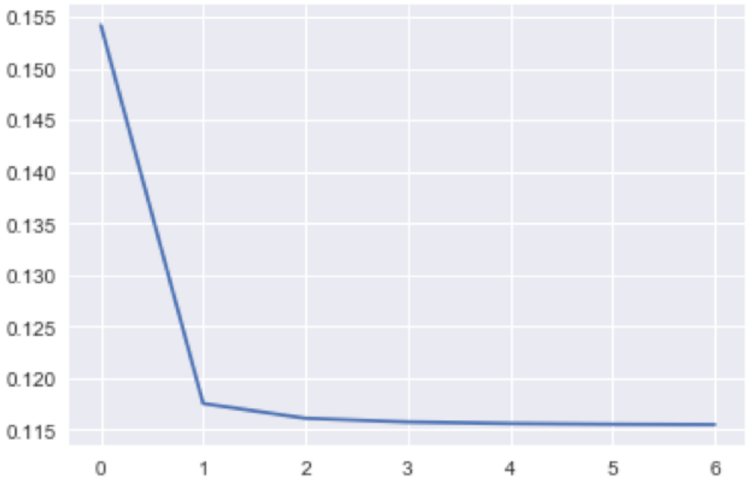
ALS

r_ui is the true rating, y,x is assumed to be colum(row) vector.The k attribite are called the latend vectors.We have to choose the best k to minimize the square of the difference between all ratings in dataset and our predictions.
Alternating least square assumed one variable is computed and use the computed value for other value
$L = \sum(r_{ui} - x_u^T * y_i)^2 + \lambda_x * \sum ||x_u||^2 + \lambda_y * \sum ||y_u||^2$
After calculation :
$x_u^T * (Y^T * Y + \lambda_x * I) = r_u * Y$
$x_u^T = r_u * Y * (Y^T * Y + \lambda_x * I)^{-1}$
$y_i^T * (x^T * x + \lambda_y * I) = r_i * x$
$y_i^T = r_i * x * (x^T * x + \lambda_y * I)^{-1}$

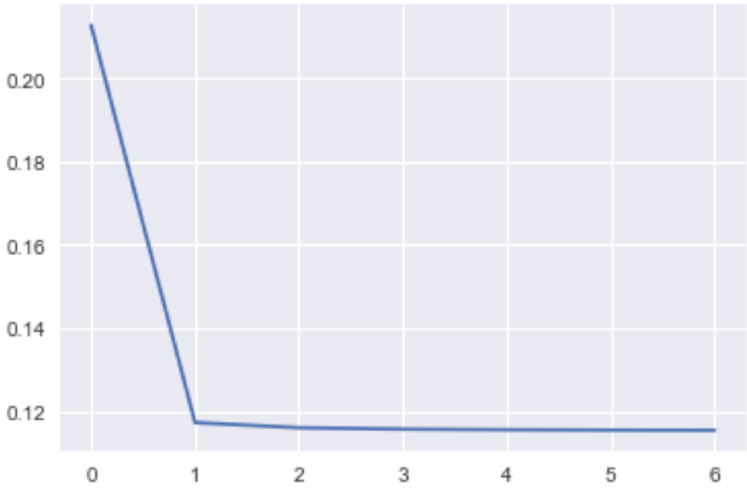
project review

In this project , I try my ALS in python, the in class ALS and the spark ALS Method. They all work well and convergent in the end. The parameter λ, the number of iterrations and the number of features will all affect the results.
MY ALS:
iteration0
Train mse0.212827914514
iteration3
Train mse0.117317858021
iteration6

```
Train mse0.116092263364
iteration9
Train mse0.115756978153
iteration12
Train mse0.11559387238
iteration15
Train mse0.115508308747
iteration18
Train mse0.115463657903
```



```
Class ALS:
Iteration: 1
Train mse: 0.154182280971
Iteration: 4
Train mse: 0.117524474032
Iteration: 7
Train mse: 0.116100241219
Iteration: 10
Train mse: 0.115737892339
Iteration: 13
Train mse: 0.11559062389
Iteration: 16
Train mse: 0.115519332251
Iteration: 19
Train mse: 0.115481796606
```



According to this data, more iterations will decrease the MSE and smaller lambda will decrease the MSE according to this data.