Mining massive Datasets WS 2017/18

Problem Set 4

Rudolf Chrispens, Marvin, Daniela Schacherer

November 23, 2017

Exercise 05

Study the code of Albert Au Yeung for matrix factorization by GD (see slide ?References? on lecture 06).

a) Apply it to the utility matrix used in lecture 06 (slide "Recall: Utility Matrix"). Do you obtain the same matrices Q and P as shown in the lecture (slide "Latent Factor Models")? Submit as solutions your matrices Q, P and the full matrix R.

```
Utility Matrix:
[[1 0 3 0 0 5 0 0 5 0 4 0]
 [0 0 5 4 0 0 4 0 0 2 1 3]
 [2 4 0 1 2 0 3 0 4 3 5 0]
 [0 2 4 0 5 0 0 4 0 0 2 0]
 [0 0 4 3 4 2 0 0 0 0 2 5]
 [1 0 3 0 3 0 0 2 0 0 4 0]]
P_Items_x_Features:
[[ 1.9131215   1.29077939  0.69386576]
 [-0.36571992  2.18788999  1.11153101]
 [ 2.07399878  0.12752533  1.4811241 ]
 [ 0.24444465  1.66461796  1.58692477]
 [ 0.45579533  1.4474079   1.46812956]
 [ 1.73770877  1.14169258  0.76518777]]
Q.T_Users_x_Features:
[[ 0.36644633  1.45147022  0.10310276 -0.09543998  0.05403266  2.17152558
   0.49735527  0.04048629  1.23370225  0.63101074  1.724804    1.61077307]
 [-0.14797647  0.34894672  1.64542336  1.46178083  1.7640161   0.34791981
   1.29415504  0.93109973  1.51234022  0.45076068  0.18410261  0.96770206]
 [ 0.77489518  0.64644431  1.02861432  0.65986469  1.13098052  0.41409709
   1.21251738  1.40630636  0.87901893  1.10528179  0.84746994  1.64667991]]
New_Utility_Matrix:
```

```
[[ 1.04772461  3.6757977   3.03484693  2.16210581  3.16507532  4.89080778
   3.463294    2.25508736  4.92224102  2.55594999  4.12542583  5.47326927]
 [ 0.40354706  0.95116835  4.70563529  3.96658002  5.09683227  0.42732185
   3.9973268   3.58549026  3.83470134  1.98399656  0.71399017  3.35846964]
 [ 1.88885442  4.01231127  1.94717362  0.96581319  2.01214291  5.16143917
   2.99244048  2.28562155  4.05349477  3.0032584   4.85592728  5.90308521]
 [ 1.07295191  1.9615256   4.39653774  3.45713246  4.74440188  1.76711231
   4.20002342  3.79152438  4.21397752  2.65859057  2.07295065  4.61775624]
 [ 1.09048873  2.11570559  3.93873162  3.04105889  4.23830461  2.10130128
   3.87999505  3.4307745   4.04180258  2.56274318  2.29682484  4.55238191]
 [ 1.06077367  3.41527369  2.84481331  2.00797783  2.97326957  4.48755853
   3.26958928  2.20947145  4.54305735  2.45689111  3.85586924  5.16389207]]
```

b) Modify the code so that the error e is stored (or printed after each iteration), and plot the error over iterations (for matrix in a)), as well as the differences of the error in subsequent steps.

c) Bonus, 3 points: In this code the learning rate "alpha" is a constant (alpha = 0.0002). How could you change the code to speed up the convergence? Explain your idea and implement your solution.