The University of Texas
at Austin

CS363D Statistical Learning and Data Mining

Homework 03

Edited by LaTeX

Department of Computer Science

STUDENT

Jimmy Lin
xl5224

INSTRUCTOR

Pradeep Ravikumar

TASSISTANT

Adarsh Prasad

RELEASE DATE

March. 25 2014

DUE DATE

April. 07 2014

TIME SPENT

7 hours

April 2, 2014

Contents

List of Figures

1 MF Implementation

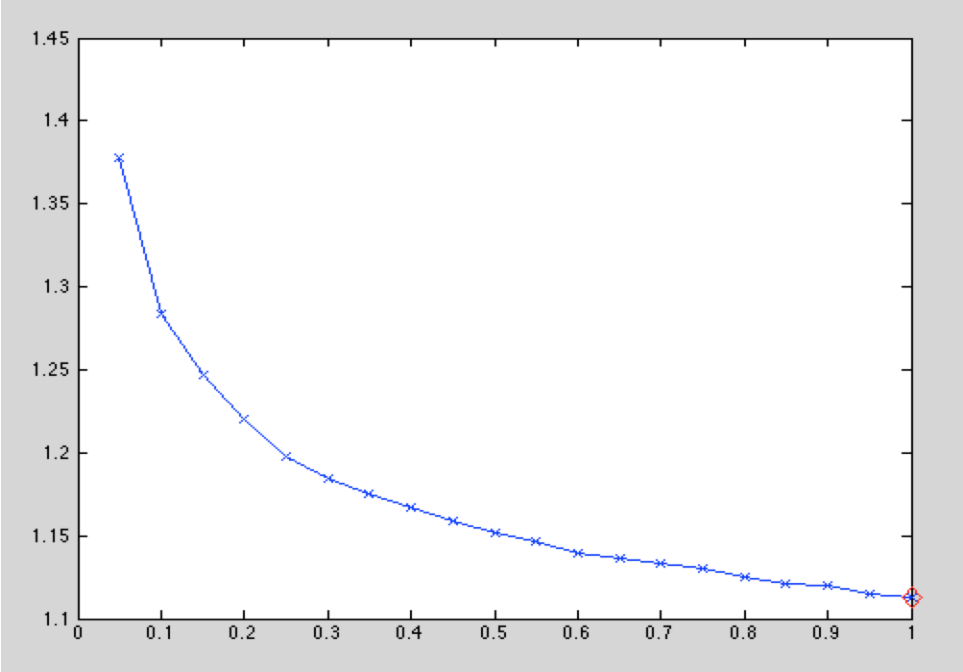

Figure 1: Errors With Regard to Parameter λ

Note that the horizental axis represents the regularization parameter, and vertical axis represents the RMSE on given regularization parameter.

For specific details, see the source code in Appendix A. For debugging, see the Execution logs in Appendix B.

2 Report optimal λ

The optimal lambda derived is $\lambda = 1$.

3 Problem when $\lambda = 0$

The problem encountered when $\lambda = 0$ is that the non-invertibility of $U^{(K_i)^T} * U^{(K_i)}$ and $M^{(K_i)^T} * M^{(K_i)}$.

4 RMSE of Test Set under optimal λ

The RMSE of test set under optimal $\lambda = 1$ is 1.0835.

A Source Code

```
% Solution for the data mining homework 03
% Author: Jimmy Lin (xl5224)

function solution()
%%% load the dataset
load('./dataset/hw3_netflix.mat');
%%% Setting about data
nCVFolds = size(cvSet, 1);
FOLDRANGE = 1:nCVFolds;
sRatings = size(Ratings);
nUsers = sRatings(1);
nMovies = sRatings(2);

%%
% PRE-SETTING
LAMBDAS = 0:0.05:1;
NITERATIONS = 30;
K = 10;
nLambdas = size(LAMBDAS, 2);

%%
% CROSS VALIDATION
avgError = zeros(1, nLambdas);
for l = 1:nLambdas,
    lambda = LAMBDAS(l);
    foldError = zeros(1,nCVFolds);
    for f = FOLDRANGE,
        %% prepare elements for training
        nItems = length(cvSet(f,:));
        cvTrainR = trR;
        cvTrainR(cvSet(f,:)) = 0;
        cvTestR = trR(cvSet(f,:));
        %% apply Alternating Minimization for training
        [U, M] = trainMF (cvTrainR, lambda, NITERATIONS, K);
        %% make prediction rating matrix
        PredictedRatings = U * M';
        %% generate prediction array for error computation
        cvPrediction = PredictedRatings(cvSet(f,:));
        %% compute root mean square error
        foldError(f) = computeRMSE (cvPrediction, cvTestR, nItems);
        fprintf('(Lambda, Fold, Error) = (%0.2f, %d, %f)\n', ...
                lambda, f, foldError(f))
    end
    fprintf('Errors when lambda=%0.2f: ', lambda)
    disp(foldError)
    %% take the mean of fold errors as error of lambda
    avgError(l) = mean(foldError);
end
    plot(LAMBDAS, avgError, 'x-')
    hold on
    %% pick up the optimal lambda
    optIdx = find(avgError <= min(avgError) + 1e-5);
    optLambda = LAMBDAS(optIdx)
    plot([optLambda], [avgError(optIdx)], 'dr', 'MarkerSize', 10)
    %% training by using optimal lambda
    [U, M] = trainMF (trR, optLambda, NITERATIONS, K);
    optPredictedRatings = U * M';
    %% compute optimal
    optRMSE = computeRMSE(optPredictedRatings(testIdx), ...
        Ratings(testIdx), length(testIdx))
end

%%% subfunction:
```

```
%% functionality: apply matrix factorization on training data
function [U, M] = trainMF (trainData, lambda, iterations, K)
nUsers = size(trainData, 1);
nMovies = size(trainData, 2);
U = randn(nUsers, K);
M = randn(nMovies, K);

for iter = 1:iterations,
    for j = 1:nMovies,
        idx = find(trainData(:, j) ~= 0);
        Uk = U(idx, :);
        M(j,:) = inv(Uk' * Uk + lambda * eye(K)) * Uk' * trainData(idx,j);
    end
    for i = 1:nUsers,
        idx = find(trainData(i, :) ~= 0);
        Mk = M(idx, :);
        U(i,:) = inv(Mk' * Mk + lambda * eye(K)) * Mk' * trainData(i,idx)';
    end
end
end

function err = computeRMSE (Prediction, GroundTruth, nItems)
err = sqrt(sum(sum((Prediction-GroundTruth).^2)) / nItems);
end
```

B Execution Logs

```
(Lambda, Fold, Error) = (0.00, 10, NaN)
Errors when lambda=0.00:    NaN   NaN   NaN   NaN   NaN   NaN   NaN   NaN   NaN   NaN

(Lambda, Fold, Error) = (0.05, 1, 1.411947)
(Lambda, Fold, Error) = (0.05, 2, 1.371217)
(Lambda, Fold, Error) = (0.05, 3, 1.360121)
(Lambda, Fold, Error) = (0.05, 4, 1.381147)
(Lambda, Fold, Error) = (0.05, 5, 1.371452)
(Lambda, Fold, Error) = (0.05, 6, 1.396735)
(Lambda, Fold, Error) = (0.05, 7, 1.387001)
(Lambda, Fold, Error) = (0.05, 8, 1.367467)
(Lambda, Fold, Error) = (0.05, 9, 1.381849)
(Lambda, Fold, Error) = (0.05, 10, 1.348981)
Errors when lambda=0.05:    1.4119    1.3712    1.3601    1.3811    1.3715    1.3967    1.3870
1.3675    1.3818    1.3490

(Lambda, Fold, Error) = (0.10, 1, 1.275159)
(Lambda, Fold, Error) = (0.10, 2, 1.291333)
(Lambda, Fold, Error) = (0.10, 3, 1.284587)
(Lambda, Fold, Error) = (0.10, 4, 1.267674)
(Lambda, Fold, Error) = (0.10, 5, 1.288017)
(Lambda, Fold, Error) = (0.10, 6, 1.313758)
(Lambda, Fold, Error) = (0.10, 7, 1.283330)
(Lambda, Fold, Error) = (0.10, 8, 1.274047)
(Lambda, Fold, Error) = (0.10, 9, 1.294935)
(Lambda, Fold, Error) = (0.10, 10, 1.263426)
Errors when lambda=0.10:    1.2752    1.2913    1.2846    1.2677    1.2880    1.3138    1.2833
1.2740    1.2949    1.2634

(Lambda, Fold, Error) = (0.15, 1, 1.241871)
(Lambda, Fold, Error) = (0.15, 2, 1.256137)
(Lambda, Fold, Error) = (0.15, 3, 1.252209)
(Lambda, Fold, Error) = (0.15, 4, 1.230881)
(Lambda, Fold, Error) = (0.15, 5, 1.243366)
(Lambda, Fold, Error) = (0.15, 6, 1.240766)
(Lambda, Fold, Error) = (0.15, 7, 1.262817)
(Lambda, Fold, Error) = (0.15, 8, 1.237307)
(Lambda, Fold, Error) = (0.15, 9, 1.254089)
```

```
(Lambda, Fold, Error) = (0.15, 10, 1.247183)
Errors when lambda=0.15:     1.2419    1.2561    1.2522    1.2309    1.2434    1.2408    1.2628
1.2373    1.2541    1.2472

(Lambda, Fold, Error) = (0.20, 1, 1.215327)
(Lambda, Fold, Error) = (0.20, 2, 1.212881)
(Lambda, Fold, Error) = (0.20, 3, 1.229502)
(Lambda, Fold, Error) = (0.20, 4, 1.205619)
(Lambda, Fold, Error) = (0.20, 5, 1.212550)
(Lambda, Fold, Error) = (0.20, 6, 1.239990)
(Lambda, Fold, Error) = (0.20, 7, 1.235272)
(Lambda, Fold, Error) = (0.20, 8, 1.211149)
(Lambda, Fold, Error) = (0.20, 9, 1.235782)
(Lambda, Fold, Error) = (0.20, 10, 1.203187)
Errors when lambda=0.20:     1.2153    1.2129    1.2295    1.2056    1.2125    1.2400    1.2353
1.2111    1.2358    1.2032

(Lambda, Fold, Error) = (0.25, 1, 1.190571)
(Lambda, Fold, Error) = (0.25, 2, 1.205245)
(Lambda, Fold, Error) = (0.25, 3, 1.205136)
(Lambda, Fold, Error) = (0.25, 4, 1.187585)
(Lambda, Fold, Error) = (0.25, 5, 1.196876)
(Lambda, Fold, Error) = (0.25, 6, 1.195276)
(Lambda, Fold, Error) = (0.25, 7, 1.204522)
(Lambda, Fold, Error) = (0.25, 8, 1.198324)
(Lambda, Fold, Error) = (0.25, 9, 1.204997)
(Lambda, Fold, Error) = (0.25, 10, 1.185293)
Errors when lambda=0.25:     1.1906    1.2052    1.2051    1.1876    1.1969    1.1953    1.2045
1.1983    1.2050    1.1853

(Lambda, Fold, Error) = (0.30, 1, 1.184220)
(Lambda, Fold, Error) = (0.30, 2, 1.190116)
(Lambda, Fold, Error) = (0.30, 3, 1.186195)
(Lambda, Fold, Error) = (0.30, 4, 1.185386)
(Lambda, Fold, Error) = (0.30, 5, 1.180466)
(Lambda, Fold, Error) = (0.30, 6, 1.180913)
(Lambda, Fold, Error) = (0.30, 7, 1.209995)
(Lambda, Fold, Error) = (0.30, 8, 1.169553)
(Lambda, Fold, Error) = (0.30, 9, 1.179413)
(Lambda, Fold, Error) = (0.30, 10, 1.176732)
Errors when lambda=0.30:     1.1842    1.1901    1.1862    1.1854    1.1805    1.1809    1.2100
1.1696    1.1794    1.1767

(Lambda, Fold, Error) = (0.35, 1, 1.164173)
(Lambda, Fold, Error) = (0.35, 2, 1.178633)
(Lambda, Fold, Error) = (0.35, 3, 1.186494)
(Lambda, Fold, Error) = (0.35, 4, 1.163389)
(Lambda, Fold, Error) = (0.35, 5, 1.179682)
(Lambda, Fold, Error) = (0.35, 6, 1.185297)
(Lambda, Fold, Error) = (0.35, 7, 1.173140)
(Lambda, Fold, Error) = (0.35, 8, 1.177644)
(Lambda, Fold, Error) = (0.35, 9, 1.183900)
(Lambda, Fold, Error) = (0.35, 10, 1.158685)
Errors when lambda=0.35:     1.1642    1.1786    1.1865    1.1634    1.1797    1.1853    1.1731
1.1776    1.1839    1.1587

(Lambda, Fold, Error) = (0.40, 1, 1.163238)
(Lambda, Fold, Error) = (0.40, 2, 1.157368)
(Lambda, Fold, Error) = (0.40, 3, 1.167919)
(Lambda, Fold, Error) = (0.40, 4, 1.173515)
(Lambda, Fold, Error) = (0.40, 5, 1.163116)
(Lambda, Fold, Error) = (0.40, 6, 1.180506)
(Lambda, Fold, Error) = (0.40, 7, 1.170660)
(Lambda, Fold, Error) = (0.40, 8, 1.151202)
(Lambda, Fold, Error) = (0.40, 9, 1.177174)
(Lambda, Fold, Error) = (0.40, 10, 1.161618)
```

```
Errors when lambda=0.40:     1.1632    1.1574    1.1679    1.1735    1.1631    1.1805    1.1707
1.1512    1.1772    1.1616

(Lambda, Fold, Error) = (0.45, 1, 1.163587)
(Lambda, Fold, Error) = (0.45, 2, 1.160354)
(Lambda, Fold, Error) = (0.45, 3, 1.157886)
(Lambda, Fold, Error) = (0.45, 4, 1.154686)
(Lambda, Fold, Error) = (0.45, 5, 1.141228)
(Lambda, Fold, Error) = (0.45, 6, 1.175810)
(Lambda, Fold, Error) = (0.45, 7, 1.157149)
(Lambda, Fold, Error) = (0.45, 8, 1.155601)
(Lambda, Fold, Error) = (0.45, 9, 1.171742)
(Lambda, Fold, Error) = (0.45, 10, 1.145750)
Errors when lambda=0.45:     1.1636    1.1604    1.1579    1.1547    1.1412    1.1758    1.1571
1.1556    1.1717    1.1458

(Lambda, Fold, Error) = (0.50, 1, 1.141676)
(Lambda, Fold, Error) = (0.50, 2, 1.154762)
(Lambda, Fold, Error) = (0.50, 3, 1.154691)
(Lambda, Fold, Error) = (0.50, 4, 1.148389)
(Lambda, Fold, Error) = (0.50, 5, 1.148762)
(Lambda, Fold, Error) = (0.50, 6, 1.160227)
(Lambda, Fold, Error) = (0.50, 7, 1.153426)
(Lambda, Fold, Error) = (0.50, 8, 1.143555)
(Lambda, Fold, Error) = (0.50, 9, 1.164293)
(Lambda, Fold, Error) = (0.50, 10, 1.144501)
Errors when lambda=0.50:     1.1417    1.1548    1.1547    1.1484    1.1488    1.1602    1.1534
1.1436    1.1643    1.1445

(Lambda, Fold, Error) = (0.55, 1, 1.131156)
(Lambda, Fold, Error) = (0.55, 2, 1.147823)
(Lambda, Fold, Error) = (0.55, 3, 1.147116)
(Lambda, Fold, Error) = (0.55, 4, 1.142009)
(Lambda, Fold, Error) = (0.55, 5, 1.152214)
(Lambda, Fold, Error) = (0.55, 6, 1.160605)
(Lambda, Fold, Error) = (0.55, 7, 1.141593)
(Lambda, Fold, Error) = (0.55, 8, 1.134633)
(Lambda, Fold, Error) = (0.55, 9, 1.147708)
(Lambda, Fold, Error) = (0.55, 10, 1.157545)
Errors when lambda=0.55:     1.1312    1.1478    1.1471    1.1420    1.1522    1.1606    1.1416
1.1346    1.1477    1.1575

(Lambda, Fold, Error) = (0.60, 1, 1.133405)
(Lambda, Fold, Error) = (0.60, 2, 1.141913)
(Lambda, Fold, Error) = (0.60, 3, 1.139707)
(Lambda, Fold, Error) = (0.60, 4, 1.132137)
(Lambda, Fold, Error) = (0.60, 5, 1.144509)
(Lambda, Fold, Error) = (0.60, 6, 1.137644)
(Lambda, Fold, Error) = (0.60, 7, 1.147042)
(Lambda, Fold, Error) = (0.60, 8, 1.132765)
(Lambda, Fold, Error) = (0.60, 9, 1.148272)
(Lambda, Fold, Error) = (0.60, 10, 1.136532)
Errors when lambda=0.60:     1.1334    1.1419    1.1397    1.1321    1.1445    1.1376    1.1470
1.1328    1.1483    1.1365

(Lambda, Fold, Error) = (0.65, 1, 1.119677)
(Lambda, Fold, Error) = (0.65, 2, 1.125584)
(Lambda, Fold, Error) = (0.65, 3, 1.137377)
(Lambda, Fold, Error) = (0.65, 4, 1.140377)
(Lambda, Fold, Error) = (0.65, 5, 1.129277)
(Lambda, Fold, Error) = (0.65, 6, 1.148645)
(Lambda, Fold, Error) = (0.65, 7, 1.140229)
(Lambda, Fold, Error) = (0.65, 8, 1.147211)
(Lambda, Fold, Error) = (0.65, 9, 1.142439)
(Lambda, Fold, Error) = (0.65, 10, 1.133074)
Errors when lambda=0.65:     1.1197    1.1256    1.1374    1.1404    1.1293    1.1486    1.1402
1.1472    1.1424    1.1331
```

```
(Lambda, Fold, Error) = (0.70, 1, 1.119570)
(Lambda, Fold, Error) = (0.70, 2, 1.140769)
(Lambda, Fold, Error) = (0.70, 3, 1.129591)
(Lambda, Fold, Error) = (0.70, 4, 1.132144)
(Lambda, Fold, Error) = (0.70, 5, 1.129148)
(Lambda, Fold, Error) = (0.70, 6, 1.142406)
(Lambda, Fold, Error) = (0.70, 7, 1.131679)
(Lambda, Fold, Error) = (0.70, 8, 1.136480)
(Lambda, Fold, Error) = (0.70, 9, 1.145003)
(Lambda, Fold, Error) = (0.70, 10, 1.121597)
Errors when lambda=0.70:     1.1196    1.1408    1.1296    1.1321    1.1291    1.1424    1.1317
1.1365    1.1450    1.1216

(Lambda, Fold, Error) = (0.75, 1, 1.121950)
(Lambda, Fold, Error) = (0.75, 2, 1.120693)
(Lambda, Fold, Error) = (0.75, 3, 1.135749)
(Lambda, Fold, Error) = (0.75, 4, 1.128547)
(Lambda, Fold, Error) = (0.75, 5, 1.125868)
(Lambda, Fold, Error) = (0.75, 6, 1.139377)
(Lambda, Fold, Error) = (0.75, 7, 1.136667)
(Lambda, Fold, Error) = (0.75, 8, 1.120701)
(Lambda, Fold, Error) = (0.75, 9, 1.145132)
(Lambda, Fold, Error) = (0.75, 10, 1.123616)
Errors when lambda=0.75:     1.1220    1.1207    1.1357    1.1285    1.1259    1.1394    1.1367
1.1207    1.1451    1.1236

(Lambda, Fold, Error) = (0.80, 1, 1.103993)
(Lambda, Fold, Error) = (0.80, 2, 1.123699)
(Lambda, Fold, Error) = (0.80, 3, 1.127682)
(Lambda, Fold, Error) = (0.80, 4, 1.128698)
(Lambda, Fold, Error) = (0.80, 5, 1.114976)
(Lambda, Fold, Error) = (0.80, 6, 1.130462)
(Lambda, Fold, Error) = (0.80, 7, 1.143441)
(Lambda, Fold, Error) = (0.80, 8, 1.128946)
(Lambda, Fold, Error) = (0.80, 9, 1.129831)
(Lambda, Fold, Error) = (0.80, 10, 1.120246)
Errors when lambda=0.80:     1.1040    1.1237    1.1277    1.1287    1.1150    1.1305    1.1434
1.1289    1.1298    1.1202

(Lambda, Fold, Error) = (0.85, 1, 1.117854)
(Lambda, Fold, Error) = (0.85, 2, 1.122193)
(Lambda, Fold, Error) = (0.85, 3, 1.124524)
(Lambda, Fold, Error) = (0.85, 4, 1.116864)
(Lambda, Fold, Error) = (0.85, 5, 1.120530)
(Lambda, Fold, Error) = (0.85, 6, 1.123193)
(Lambda, Fold, Error) = (0.85, 7, 1.124361)
(Lambda, Fold, Error) = (0.85, 8, 1.114095)
(Lambda, Fold, Error) = (0.85, 9, 1.124844)
(Lambda, Fold, Error) = (0.85, 10, 1.119965)
Errors when lambda=0.85:     1.1179    1.1222    1.1245    1.1169    1.1205    1.1232    1.1244
1.1141    1.1248    1.1200

(Lambda, Fold, Error) = (0.90, 1, 1.100922)
(Lambda, Fold, Error) = (0.90, 2, 1.126601)
(Lambda, Fold, Error) = (0.90, 3, 1.126961)
(Lambda, Fold, Error) = (0.90, 4, 1.118059)
(Lambda, Fold, Error) = (0.90, 5, 1.122241)
(Lambda, Fold, Error) = (0.90, 6, 1.126720)
(Lambda, Fold, Error) = (0.90, 7, 1.125150)
(Lambda, Fold, Error) = (0.90, 8, 1.110950)
(Lambda, Fold, Error) = (0.90, 9, 1.125494)
(Lambda, Fold, Error) = (0.90, 10, 1.112672)
Errors when lambda=0.90:     1.1009    1.1266    1.1270    1.1181    1.1222    1.1267    1.1251
1.1110    1.1255    1.1127

(Lambda, Fold, Error) = (0.95, 1, 1.098285)
```

```
(Lambda, Fold, Error) = (0.95, 2, 1.112644)
(Lambda, Fold, Error) = (0.95, 3, 1.129575)
(Lambda, Fold, Error) = (0.95, 4, 1.108022)
(Lambda, Fold, Error) = (0.95, 5, 1.113344)
(Lambda, Fold, Error) = (0.95, 6, 1.126322)
(Lambda, Fold, Error) = (0.95, 7, 1.117771)
(Lambda, Fold, Error) = (0.95, 8, 1.102943)
(Lambda, Fold, Error) = (0.95, 9, 1.130283)
(Lambda, Fold, Error) = (0.95, 10, 1.109194)
Errors when lambda=0.95:     1.0983    1.1126    1.1296    1.1080    1.1133    1.1263    1.1178
1.1029    1.1303    1.1092

(Lambda, Fold, Error) = (1.00, 1, 1.104553)
(Lambda, Fold, Error) = (1.00, 2, 1.110303)
(Lambda, Fold, Error) = (1.00, 3, 1.119303)
(Lambda, Fold, Error) = (1.00, 4, 1.110195)
(Lambda, Fold, Error) = (1.00, 5, 1.112638)
(Lambda, Fold, Error) = (1.00, 6, 1.117118)
(Lambda, Fold, Error) = (1.00, 7, 1.115305)
(Lambda, Fold, Error) = (1.00, 8, 1.105883)
(Lambda, Fold, Error) = (1.00, 9, 1.127105)
(Lambda, Fold, Error) = (1.00, 10, 1.104722)
Errors when lambda=1.00:     1.1046    1.1103    1.1193    1.1102    1.1126    1.1171    1.1153
1.1059    1.1271    1.1047


optLambda =

     1


optRMSE =

    1.0835
```