

Data Mining, Spring 2018

Problem Set #3: PCA, Recommender System, and Association Analysis

(Due on June 24, Sunday)

Submission Instructions

These questions require thought but do not require long answers. Please be as concise as possible. We do not do reverse engineering, so please DO NOT provide MATLAB (or other programming language) codes WITHOUT **method description**. You should also declare in the assignment that **the MATLAB (or other programming language) code was written by you, not by others either partially or fully**.

You should submit your answers as a write-up in PDF format to DataMining_2018@126.com. The email title is formatted as “hwk3_学号_姓名”.

If you are in doubt, talk to me (majh8@mail.sysu.edu.cn) or our teaching assistants to understand more.

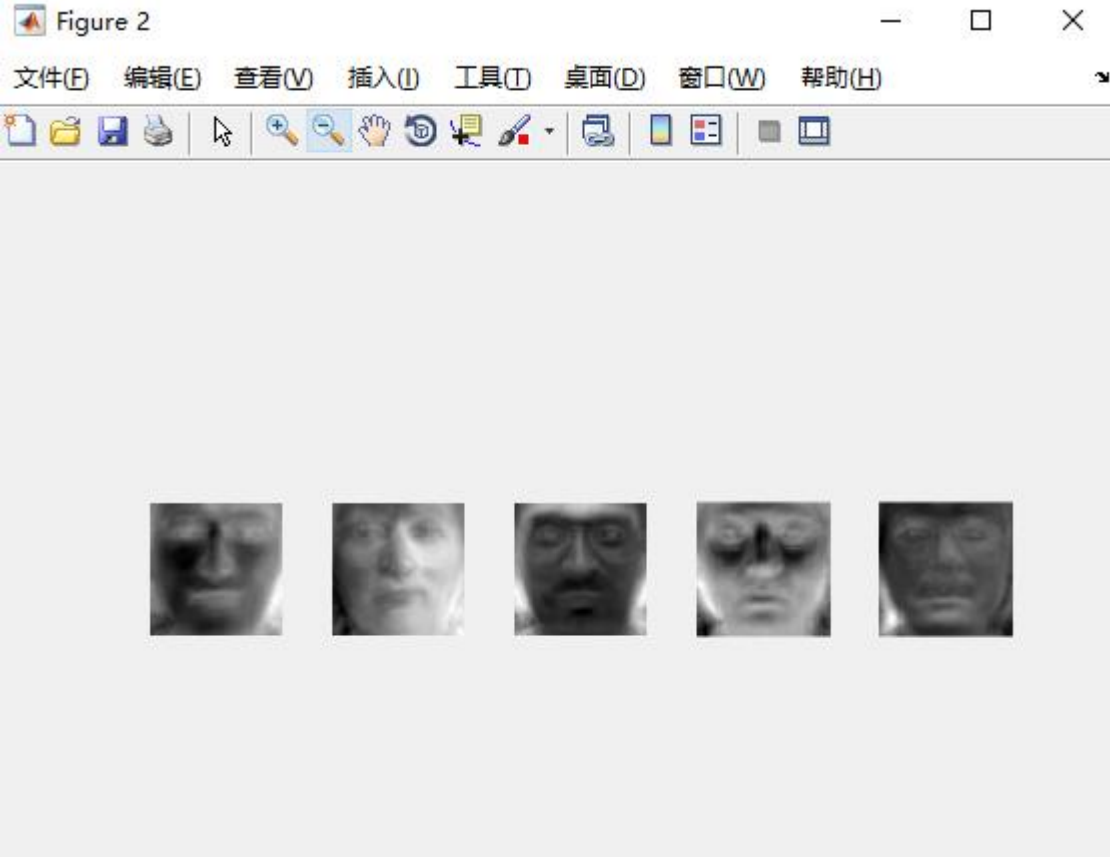
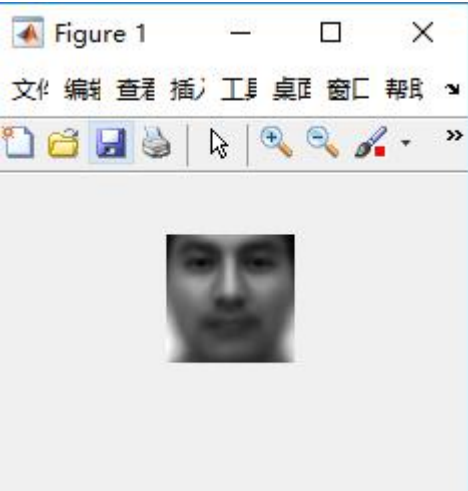
Questions

1. 主成分分析 (Principal Component Analysis, PCA)

请从课程网站或[此链接](#)下载 Yale 人脸数据集进行降维。通过 MATLAB 命令 `load('yale_face.mat')` 读取数据，包含一个 4096×165 矩阵。（请注意，这里的矩阵 X 是课件第 21 页定义的数据矩阵 X 的转置。）此矩阵的每一列是由一张 64×64 的灰度人脸图像所转成的 4096 维向量，即每一 4096 维列向量是一个训练样本，特征的维数是 $n = 4096$ ，样本的个数是 $m = 165$ 。例如，可以使用 `imshow(reshape(X(:,1),[64 64]),[])` 命令显示第一张人脸图像（第一个训练样本）。

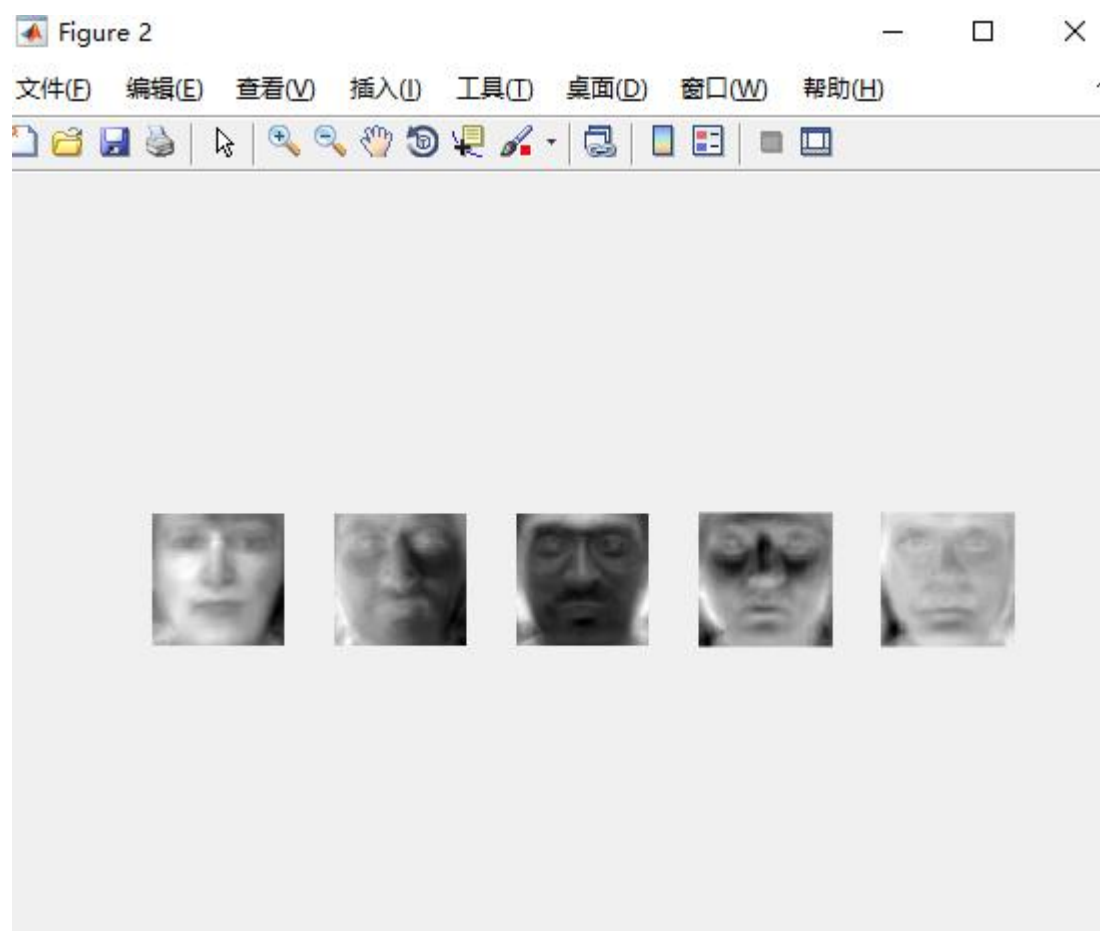
- （1）试使用 MATLAB 中的 `svd` 函数实现 PCA 算法，即输入数据矩阵 X 和降维后的维数 k ，对每一个样本进行去中心化，然后对进行去中心化后的数据矩阵 X_c 用 `svd` 函数 `[U,S,V] = svd(Xc)`，输出降维的投影矩阵 `Ureduce`（即 U 的前 k 列），降维后的坐标表示 $Z = Ureduce' * X_c$ ，训练样本均值 μ 。并令 $k=5$ ，显示样本均值 μ 的图像和 `Ureduce` 的五个列向量（即协方差矩阵的前五个特征向量）所对应的图像；

Problem Set #3



```
function PCA
-   load('yale_face.mat');
-   %imshow(reshape(X(:,1),[64 64]),[]);
-   %均值矩阵
-   mean_yale_face = mean(X,2);
-   figure
-   imshow(reshape(mean_yale_face,[64 64]),[]);
-   %去中心化矩阵
-   for i=1:165
-       Xc(:,i) = X(:,i) - mean_yale_face;
-   end
-   tic;
-   [U,S,V] = svd(Xc);
-   toc;
-   %投影矩阵Ureduce
-   Ureduce = U(:,1:5);
-   figure
-   for i=1:5
-       subplot(1,5,i);
-       imshow(reshape(Ureduce(:,i),[64 64]),[]);
-   end
-   z = Ureduce' * Xc;
-   end
```

- (2) 试对协方差矩阵使用 MATLAB 中的 `eig` 函数计算特征值和特征向量,即 $[U,D]=\text{eig}(Xc \cdot Xc' / m)$, 显示前五个最大的特征向量所对应的图像, 并比较对数据矩阵使用 `svd` 函数的所得出的特征向量的图像与运算时间;



svd: 时间已过 0.438822 秒。

eig: 时间已过 20.749727 秒。

明显可以看出 svd 函数运行速度比 eig 快很多。

```

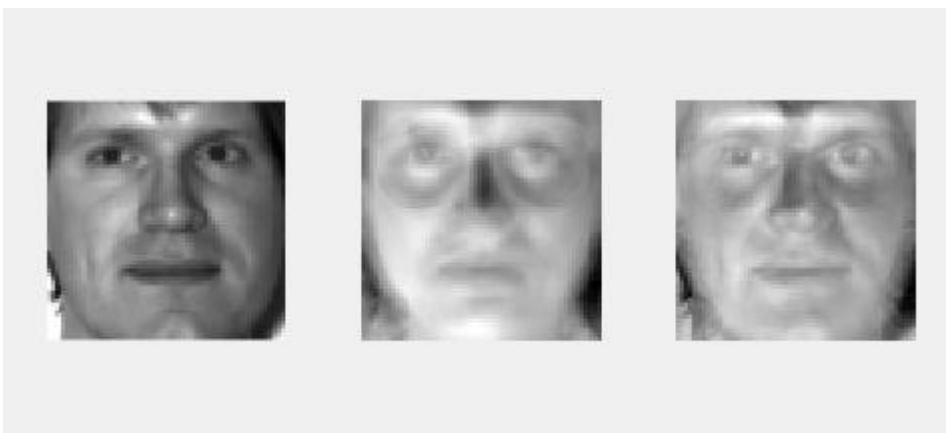
function PCA2
load('yale_face.mat');
%imshow(reshape(X(:,1),[64 64]),[]);
%均值矩阵
mean_yale_face = mean(X,2);
%去中心化矩阵
for i=1:165
    Xc(:,i) = X(:,i) - mean_yale_face;
end
sigma = cov(Xc');
tic;
[U2,D] = eig(sigma);
toc;
U2 = (rot90(U2))';
Ureduce2 = U2(:,1:5);
figure
for i=1:5
    subplot(1,5,i);
    imshow(reshape(Ureduce2(:,i),[64 64]),[]);
end
z = Ureduce2' * Xc;
end

```

(3) 试计算当降维后的维数分别是 10 和 100 时，保留的方差的比例（即 $1 - \frac{\sum_{i=1}^m \|x^{(i)} - x_{approx}^{(i)}\|^2}{\sum_{i=1}^m \|x^{(i)}\|^2}$

或使用（1）中的 S 计算 $\sum_{i=1}^k S_{ii}^2 / \sum_{i=1}^m S_{ii}^2$ 或使用（2）中的 D 计算 $\sum_{i=1}^k D_{ii} / \sum_{i=1}^m D_{ii}$ ），并分别利用 10 维和 100 维坐标恢复原高维空间中的人脸图像，对前三张人脸图像，对比原图和两张恢复的图像。

第一张图：（从左到右分别是原图，10 维恢复图，100 维恢复图，下同）





```
load('yale_face.mat');
mean_yale_face = mean(X, 2);
for i=1:165
    Xc(:, i) = X(:, i) - mean_yale_face;
end
[U, S, V] = svd(Xc);
U1 = U(:, 1:10);
U2 = U(:, 1:100);
Z1 = U1' * Xc;
Z2 = U2' * Xc;
Z_1 = U1 * Z1;
Z_2 = U2 * Z2;
figure()
subplot(1, 3, 1), imshow(reshape(X(:, 1), [64 64]), []);
subplot(1, 3, 2), imshow(reshape(Z_1(:, 1), [64 64]), []);
subplot(1, 3, 3), imshow(reshape(Z_2(:, 1), [64 64]), []);
figure()
subplot(1, 3, 1), imshow(reshape(X(:, 2), [64 64]), []);
subplot(1, 3, 2), imshow(reshape(Z_1(:, 2), [64 64]), []);
subplot(1, 3, 3), imshow(reshape(Z_2(:, 2), [64 64]), []);
figure()
```

2. 推荐系统（Recommender System）

考虑以下的 8 个用户（A-H）对 7 部电影评级（1 到 5 级）的一个效用矩阵：

	A	B	C	D	E	F	G	H
HP1	4	4			1	1	5	
HP2	5	5		1				
HP3		4	1			1	5	4
TW	5		2	5		1	2	
SW1	1		5	4	5			1
SW2	1		5			4		
SW3		1		5		5	1	

电影的名字 HP1、HP2、HP3 分别代表《哈利波特》（Harry Potter）I、II、III，TW 代表《暮光之城》（Twilight），SW1、SW2 和 SW3 分别代表《星球大战》（Star Wars）I、II、III。

- （1）试实现协同过滤算法（Collaborative filtering algorithm，课件第 19 页，不需要进行去均值操作），令正则化参数 $\lambda = 0.1$ ，特征向量维数 $n = 4$ ，学习率 $\alpha = 0.01$ ，分别计算描述电影特征的 7×4 矩阵 X 和预测用户评级的 8×4 模型参数矩阵 Θ （定义见课件第 23 页，所得结果保留小数点后 4 位），并计算预测电影评级的 7×8 效用矩阵即 $X\Theta'$ （保留小数点后 1 位）；

X:

7x4 double				
	1	2	3	4
1	1.1652	1.4270	-0.0023	0.8046
2	1.7683	1.3463	-0.5217	0.8514
3	1.0177	1.5773	-0.0933	0.7364
4	1.6999	-0.2236	1.0659	1.1068
5	-0.3751	0.5314	1.8763	0.6688
6	-0.4155	0.6768	1.8055	0.6541
7	-0.2323	0.7210	2.1843	0.8892

Θ :

8x4 double

	1	2	3	4
1	1.8888	0.5714	0.3778	1.0895
2	1.3100	1.2698	-0.1526	0.7774
3	-0.4545	0.7050	2.0080	0.7209
4	0.8179	-0.1843	1.9411	1.0665
5	-0.3877	0.6012	2.0375	0.7399
6	-0.5515	0.8261	1.5630	0.5323
7	1.1749	1.8964	-0.3468	0.7737
8	0.9177	1.4912	0.0300	0.7222

X_0 :

7x8 double

	1	2	3	4	5	6	7	8
1	0.0117	0.0013	0	0	1.1355e-05	0.0015	0.0910	0
2	0.0257	0.0540	0	0.0088	0	0	0	0
3	0	0.0060	4.8179e-05	0	0	1.4500e-04	0.0445	0.0342
4	0.0951	0	6.3540e-05	0.1018	0	0.0177	0.0036	0
5	0.0011	0	0.0420	0.0024	0.0472	0	0	1.5347e-04
6	9.7409e-06	0	0.0561	0	0	0.0017	0	0
7	0	9.5471e-04	0	0.0182	0	0.1513	6.2073e-04	0


```

for i = 1:num_movies
    x = X(i, :);
    sum = 0;
    for j = 1:num_user
        if Y(i, j) > 0
            theta = Theta(j, :);
            sum = sum + (theta * x' - Y(i, j))*theta + lamda * x;
        end
    end
    X(i, :) = x - alpha * sum;
end
for j = 1:num_user
    theta = Theta(j, :);
    sum = 0;
    for i = 1:num_movies
        if Y(i, j) > 0
            x = X(i, :);
            sum = sum + (theta * x' - Y(i, j))*x + lamda * theta;
        end
    end
    Theta(j, :) = theta - alpha * sum;
end
end

```

(2) 试计算(1)中预测的电影评级与真实评级的平方误差, 即 $\sum_{(i,j): r(i,j)=1} ((\theta^{(j)})^T x^{(i)} - y^{(i,j)})^2$,

其中 $r(i,j), y^{(i,j)}, \theta^{(j)}, x^{(i)}$ 的定义见课件第 7 页；讨论哪两部电影和 HP1 最相似，哪两部电影和 SW1 最相似；

计算预测的电影评级与真实评级的平方误差  0.8188。

计算其它电影与 HP1 的平方误差如下：

1x7 double								
	1	2	3	4	5	6	7	
1	0	7.4551	0.2720	24.5402	89.7129	84.6359	98.7597	

看出 HP2, HP3 与 HP1 最相似；

计算其他电影与 SW1 的平方误差如下：

	1	2	3	4	5	6	7	
1	89.7129	146.5354	91.8076	44.9567	0	0.1543	4.2628	

看出 SW2, SW3 与 SW1 最相似。

```

for i = 1:7
    for j = 1:8
        if Y(i, j) > 0
            square_error(i, j) = (Xt(i, j) - Y(i, j))^2;
            Square_error = Square_error + square_error(i, j);
        end
    end
end
end

```

- (3) 试使用一个非零常数对协同过滤算法中的变量 $x^{(1)}, \dots, x^{(n_m)}, \theta^{(1)}, \dots, \theta^{(n_u)}$ 进行初始化（即更改课件第 19 页 Collaborative filtering algorithm 的第一步为 $x^{(1)} = \dots = x^{(n_m)} = \theta^{(1)} = \dots = \theta^{(n_u)} = c\mathbf{1}$ ，其中 c 为非零实数， $\mathbf{1}$ 为所有元素都是 1 的 $n = 4$ 维列向量；使用 (1) 中相同的参数，分别计算描述电影特征的 7×4 矩阵 X 和预测用户评级的 8×4 模型参数矩阵 Θ （所得结果保留小数点后 4 位），并计算预测电影评级的 7×8 效用矩阵即 $X\Theta'$ （保留小数点后 1 位）和预测的电影评级与真实评级的平方误差，即 $\sum_{(i,j): r(i,j)=1} \left((\theta^{(j)})^T x^{(i)} - y^{(i,j)} \right)^2$ ；与 (1) 和 (2) 中的结果比较，讨论此初始化方法的问题。

答：将 X 和 Θ 都初始化为全 1 矩阵，计算得到的结果矩阵如下：

X :

7x4 double

	1	2	3	4
1	0.8797	0.8797	0.8797	0.8797
2	0.8929	0.8929	0.8929	0.8929
3	0.8701	0.8701	0.8701	0.8701
4	0.8470	0.8470	0.8470	0.8470
5	0.9419	0.9419	0.9419	0.9419
6	1.0010	1.0010	1.0010	1.0010
7	0.7241	0.7241	0.7241	0.7241

Θ :


8x4 double

	1	2	3	4
1	0.8228	0.8228	0.8228	0.8228
2	1.0320	1.0320	1.0320	1.0320
3	0.8862	0.8862	0.8862	0.8862
4	1.0340	1.0340	1.0340	1.0340
5	0.8166	0.8166	0.8166	0.8166
6	0.6553	0.6553	0.6553	0.6553
7	0.9718	0.9718	0.9718	0.9718
8	0.6525	0.6525	0.6525	0.6525

x_0 :

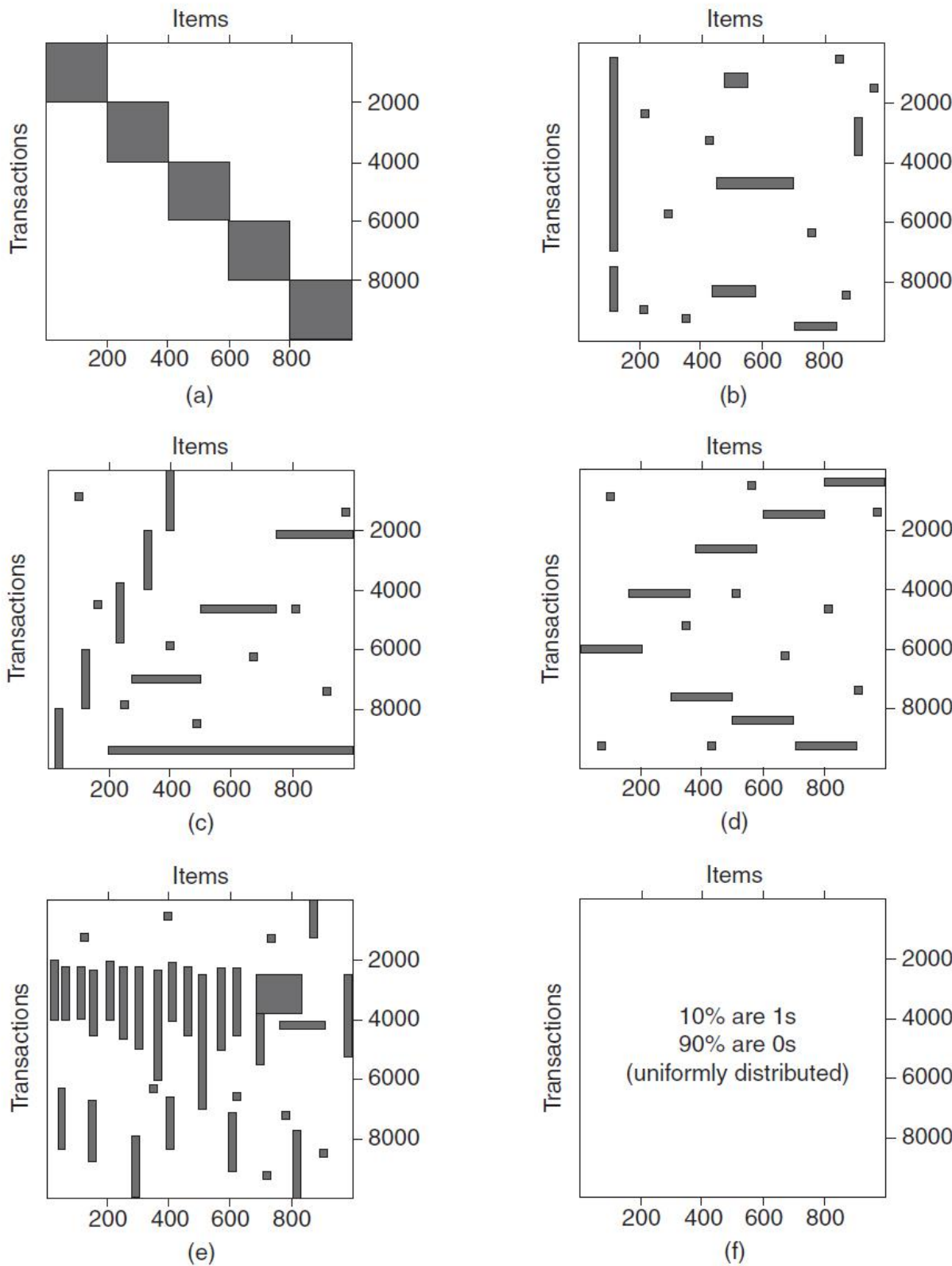
7x8 double

	1	2	3	4	5	6	7	8
1	2.8952	3.6312	3.1184	3.6383	2.8736	2.3057	3.4196	2.2961
2	2.9387	3.6858	3.1652	3.6929	2.9167	2.3403	3.4709	2.3306
3	2.8637	3.5917	3.0845	3.5987	2.8423	2.2806	3.3824	2.2711
4	2.7878	3.4965	3.0027	3.5032	2.7669	2.2201	3.2927	2.2109
5	3.1000	3.8881	3.3390	3.8956	3.0768	2.4688	3.6614	2.4585
6	3.2945	4.1320	3.5485	4.1401	3.2699	2.6237	3.8912	2.6128
7	2.3832	2.9890	2.5669	2.9948	2.3654	1.8979	2.8148	1.8900

预测的电影评级与真实评级的平方误差为  88.5191 。

与（1）（2）中的结果相比，发现将参数初始化为同一个值会导致很严重的问题，算法的效果变差。这是因为参数都相同在算法学习中难以达到梯度下降的优化效果，而随机初始化通过打破对称性来确保 x 和 θ 在算法执行中保持不相同。

3. 关联规则



上面六个图 (a) 到 (f) 中的每一个图包含 1000 个商品和 10000 个交易的记录。灰色位置表示存在商品交易，而白色表示不存在商品交易。我们使用 Apriori 算法提取频繁项集，并设定频繁项集的最小支持度为 10%，即 $\text{minsup}=10\%$ （即频繁项集包含在至少 1000 个交易中）。

根据上图回答以下问题：

(1) 哪一个或几个数据集的频繁项集数目最多？哪一个或几个数据集的频繁项集数目最少？

a 数据集的频繁项集数目最多；d 数据集的频繁项集数目最少。

(2) 哪一个或几个数据集的频繁项集长度最长（即包含最多商品）？

a 数据集的频繁项集长度最长。

(3) 哪一个或几个数据集的频繁项集有最高的最大支持度（highest maximum support）？

b 数据集的频繁项集有最高的最大支持度。

(4) 哪一个或几个数据集的频繁项集有最大的支持度范围（例如频繁项集的支持度范围可以从小于 20%变化到大于 70%）？

b 数据集的频繁项集有最大的支持度范围。