Homework 3

15331191 廖颖泓

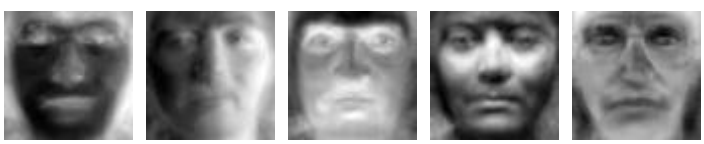
1. 主成分分析

(1) svd 函数实现 PCA 算法：

```
% 加载数据
load('yale_face.mat');
X = X';
[m, n] = size(X);
k = 5;
% 计算均值和标准差并对数据中心化
xmean = mean(X, 1);
xstd = std(X, 1);
for i = 1:m
    X(i,:) = (X(i,:) - xmean) ./ xstd;
End
% 显示均值图像
t = reshape(xmean(1,:),[64,64]);
imshow(t,[]);
imwrite(uint8(t), 'mean.bmp');
% 计算Sigma矩阵
Sigma = (1/m) * X' * X;
% svd分解
[U,S,V] = svd(Sigma);
% 显示前5个特征向量对应的图像
Ureduce = U(:,1:k);
for i = 1:k
    t = reshape(Ureduce(:,i),[64,64]);
    figure;
    imshow(t,[]);
end
```

均值图像：

前五个特征向量对应的图像：



(2) eig 函数实现 PCA 算法：

```
% 加载数据
load('yale_face.mat');
X = X';
[m, n] = size(X);
k = 5;
% 计算均值和标准差并对数据中心化
xmean = mean(X, 1);
xstd = std(X, 1);
for i = 1:m
    X(i,:) = (X(i,:) - xmean) ./ xstd;
End
% 计算Sigma矩阵
Sigma = (1/m) * X' * X;
% eig求特征值
[Q,D] = eig(Sigma);
d = zeros(1, n);
for i = 1:n
    d(1,i) = D(i,i);
end
[~,index] = sort(d, 'descend');
% 显示前5个特征向量对应的图像
z = Q(:,index(1:k));
for i = 1:k
    t = reshape(z(:,i),[64,64])
    figure;
    imshow(t,[]);
end
```

前五个特征向量对应的图像：

比较两种方法得到的特征向量:

```
isequal = isequal(U, Q);
```

显示结果：

```
isequal =

     1
```

说明两种方法得到的特征向量相同。
用以下代码比较两种方法花费的时间：

```
% svd 分解
tic
[U,S,V] = svd(Sigma);
```

```
toc
% eig 求特征值(考虑排序花费的时间)
tic
[Q,D] = eig(Sigma);
d = zeros(1, n);
for i = 1:n
    d(1,i) = D(i,i);
end
[S,index] = sort(d, 'descend');
toc
```

得到以下结果：

```
时间已过 63.448731 秒。
时间已过 121.762653 秒。
```

可以看出

用 svd 函数花费的时间是 63.448731s，用 eig 函数花费的时间是 121.762653s。(电脑 CPU 运行慢导致花费时间较长。)

(3) 代码如下：

```
% 加载数据
load('yale_face.mat');
X = X';
[m, n] = size(X);
k = 5;
% 计算均值和标准差并对数据中心化
xmean = mean(X, 1);
xstd = std(X, 1);
for i = 1:m
    X(i,:) = (X(i,:) - xmean) ./ xstd;
End
% 计算Sigma矩阵
Sigma = (1/m) * X' * X;
% svd分解
[U,S,V] = svd(Sigma);
s = zeros(1, n);
for i = 1:n
    s(1,i) = S(i,i);
end
x1 = X(1,:);
x2 = X(2,:);
x3 = X(3,:);
% 降维后的维数为100
k = 10;
% 计算方差比例
retained_var1 = sum(s(1:k))/sum(s(1:n));
```

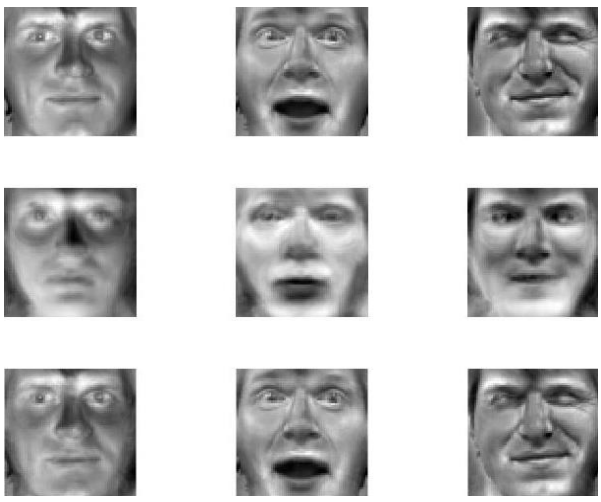
```
Ureduce = U(:, 1:k);
Y = X * Ureduce;
Y = Y * Ureduce';
y1 = Y(1,:);
y2 = Y(2,:);
y3 = Y(3,:);
% 降维后的维数为100
k = 100;
% 计算方差比例
retained_var2 = sum(s(1:k) .* s(1:k))/sum(s(1:n) .* s(1:n));
Ureduce = U(:, 1:k);
Z = X * Ureduce;
Z = Z * Ureduce';
z1 = Z(1,:);
z2 = Z(2,:);
z3 = Z(3,:);
% 显示恢复图像
figure;
subplot(3,3,1);imshow(reshape(x1,[64,64]), []);
subplot(3,3,2);imshow(reshape(x2,[64,64]), []);
subplot(3,3,3);imshow(reshape(x3,[64,64]), []);
subplot(3,3,4);imshow(reshape(y1,[64,64]), []);
subplot(3,3,5);imshow(reshape(y2,[64,64]), []);
subplot(3,3,6);imshow(reshape(y3,[64,64]), []);
subplot(3,3,7);imshow(reshape(z1,[64,64]), []);
subplot(3,3,8);imshow(reshape(z2,[64,64]), []);
subplot(3,3,9);imshow(reshape(z3,[64,64]), []);
```

计算出结果：

降维后的维数是 10，保留的方差比例是 72.15%。

降维后的维数是 100，保留的方差比例是 99.99%。

如图所示，第一行是原图，第二行是降维后的维数是 10 的恢复图像，第三行是降维后的维数是 100 的恢复图像。



2. 推荐系统

(1) 协同过滤算法代码如下

```
% 导入数据
r = [1 1 0 0 1 1 1 0;
    1 1 0 1 0 0 0 0;
    0 1 1 0 0 1 1 1;
    1 0 1 1 0 1 1 0;
    1 0 1 1 1 0 0 1;
    1 0 1 0 0 1 0 0;
    0 1 0 1 0 1 1 0;];
y = [4 4 0 0 1 1 5 0;
    5 5 0 1 0 0 0 0;
    0 4 1 0 0 1 5 4;
    5 0 2 5 0 1 2 0;
    1 0 5 4 5 0 0 1;
    1 0 5 0 0 4 0 0;
    0 1 0 5 0 5 1 0;];
% 初始化矩阵和超参数
[m,n] = size(r);
maxIter = 100000000;
;
lamda = 0.1;
alpha = 0.01;
x = abs(randn(7,4));
theta = abs(randn(8,4));
% 梯度下降
for iter = 1:maxIter
   predict = x * theta';
   loss = (predict - y) .* r;
   x_grad = loss * theta + lamda .* x;
   theata_grad = loss' * x + lamda .* theta;
   x = x - alpha .* x_grad;
   theta = theta  - alpha .* theata_grad;
end
```

运行代码，得到描述电影特征的7×4矩阵X和预测用户评级的8×4模型参数矩阵Θ分别是

$$X=\begin{pmatrix}1.8013 & 0.5574 & -0.1253 & 1.0337\\ 1.4525 & 1.2732 & -0.8411 & 1.2141\\ 1.8877 & 0.5814 & -0.1665 & 0.8782\\ -0.1072 & 1.2939 & 0.6459 & 1.9769\\ 0.5177 & 0.6341 & 2.1105 & -0.0165\\ 0.4696 & 0.9988 & 1.8471 & -0.2685\\ 0.6105 & 0.5856 & 2.3514 & 0.4485\end{pmatrix}\quad \theta=\begin{pmatrix}0.8093 & 1.2475 & -0.0833 & 1.7556\\ 1.2249 & 1.1864 & -0.3967 & 1.0593\\ 0.5066 & 1.0236 & 1.9329 & -0.2623\\ -0.2191 & 0.8914 & 1.6894 & 1.3435\\ 0.5099 & 0.6105 & 2.0153 & -0.0070\\ 0.7235 & 0.1272 & 1.8943 & -0.1406\\ 2.0143 & -0.4883 & -0.3940 & 0.9339\\ 1.5778 & 0.5033 & -0.0605 & 0.7246\end{pmatrix}$$

因此计算出预测电影评级的7×8效用矩阵即XΘ′是

$$X\theta' = \begin{pmatrix} 4.0 & 4.0 & 1.0 & 1.3 & 1.0 & 1.0 & 5.0 & 4.0 \\ 5.0 & 5.0 & 0.1 & 1.0 & -0.2 & -0.6 & 5.0 & 3.9 \\ 3.8 & 4.0 & 1.0 & 1.0 & 1.0 & 1.0 & 5.0 & 4.0 \\ 5.0 & 3.2 & 2.0 & 5.0 & 2.0 & 1.0 & 2.0 & 1.9 \\ 1.0 & 0.5 & 5.0 & 4.0 & 5.0 & 4.5 & 0.5 & 1.0 \\ 1.0 & 0.7 & 4.9 & 3.5 & 4.6 & 4.0 & 0.5 & 0.9 \\ 1.8 & 1.0 & 5.3 & 5.0 & 5.4 & 5.0 & 1.0 & 1.4 \end{pmatrix}$$

(2) 用下列代码计算平方误差

```
loss = sum(sum(((x * theta' - y) .* r) .^ 2));
```

计算出预测的电影评级与真实评级的平方误差是0.0650。

根据计算出的电影评级效用矩阵，我们可以计算不同电影之间的相似程度，在这里我们用SSIM(结构相似度，图片处理常用指标)来计算其余电影与HP1的相似程度，分别为0.88, 0.99, 0.25, -0.81, -0.81, -0.82。因此我们可以得出HP3和HP2两部电影与HP1最类似。

类似地，对于SW1，我们可以求出-0.81, -0.73, -0.78, -0.25, 0.99, 0.97。因此我们可以得出SW2和SW3两部电影与SW1最类似。

(3) 我们用另一种方式初始化矩阵

```
c = abs(randn(1,1));

x = zeros(7,4);

theta = zeros(8,4);

x(:) = c;

theta(:) = c;
```

计算得到描述电影特征的7×4矩阵X和预测用户评级的8×4模型参数矩阵Θ分别是

$$X = \begin{pmatrix} 0.9170 & 0.9170 & 0.9170 & 0.9170 \\ 0.9165 & 0.9165 & 0.9165 & 0.9165 \\ 0.9059 & 0.9059 & 0.9059 & 0.9059 \\ 0.8795 & 0.8795 & 0.8795 & 0.8795 \\ 0.9844 & 0.9844 & 0.9844 & 0.9844 \\ 1.0448 & 1.0448 & 1.0448 & 1.0448 \\ 0.7453 & 0.7453 & 0.7453 & 0.7453 \end{pmatrix} \qquad \theta = \begin{pmatrix} 0.8082 & 0.8082 & 0.8082 & 0.8082 \\ 1.0236 & 1.0236 & 1.0236 & 1.0236 \\ 0.8703 & 0.8703 & 0.8703 & 0.8703 \\ 1.0258 & 1.0258 & 1.0258 & 1.0258 \\ 0.7955 & 0.7955 & 0.7955 & 0.7955 \\ 0.6457 & 0.6457 & 0.6457 & 0.6457 \\ 0.9633 & 0.9633 & 0.9633 & 0.9633 \\ 0.6348 & 0.6348 & 0.6348 & 0.6348 \end{pmatrix}$$

因此计算出预测电影评级的7×8效用矩阵即XΘ′是

$$\begin{pmatrix} 3.0 & 3.8 & 3.2 & 3.8 & 3.0 & 2.4 & 3.5 & 2.3 \\ 3.0 & 3.8 & 3.2 & 3.8 & 3.0 & 2.4 & 3.5 & 2.3 \\ 2.9 & 3.7 & 3.2 & 3.7 & 2.9 & 2.3 & 3.5 & 2.3 \\ 2.8 & 3.6 & 3.1 & 3.6 & 2.8 & 2.3 & 3.4 & 2.2 \\ 3.2 & 4.0 & 3.4 & 4.0 & 3.1 & 2.5 & 3.8 & 2.5 \\ 3.4 & 4.3 & 3.6 & 4.3 & 3.3 & 2.7 & 4.0 & 2.7 \\ 2.4 & 3.1 & 2.6 & 3.1 & 2.4 & 1.9 & 2.9 & 1.9 \end{pmatrix}$$

计算出预测的电影评级与真实评级的平方误差是 88.2260。

该初始化方法经过相同迭代次数的梯度下降，矩阵中的参数收敛速度较慢，计算出来平方误差较(1)中方法得到的大，导致得到的效用矩阵不能准确地预测出电影评级。

3. 关联规则

(1) 数据集(e)的频繁项集数目最多，因为(e)中的频繁项集长度最长，子集数目比较多，所以频繁项集数也多；数据集(d)的频繁项集数目最少，因为(d)中没有达到最小支持度为 10%的频繁项集。

(2) 数据集(e)的频繁项集长度最长。

(3) 数据集(b)的频繁项集数有最高的最大支持度，因为 100-200 那里有一小块范围的商品在很多交易(数据库中的 transaction 应该翻译成事务)中都包含。
(4) 数据集(e)，商品的最大支持度的范围变化较多。