Q1 test

clc;

clear all;

load('PCA\_comp1.mat');

X = PCA\_comp1(:,1:4)';

[W, pc] = princomp('PCA\_comp1.mat'); pc=pc'; W=W';

mu = mean(PCA\_comp1',2); % compute the mean(s)

n = size(X,2);

A = X - mu\*ones(1,n);

%[coeff,score] = pca(A);

%plot(

%[U\_norm,S\_norm,V\_norm] = svd(X);

covar = X \* X'; % compute the covariance matrix

[V\_temp,D\_temp] = eig(covar); % compute an eigenvalue decomposition

[D\_vec,indx] = sort(diag(D\_temp),'descend'); %sort eigenvalues in descending order

sel\_ind = find(D\_vec>1e-5); % select non-zero eigenvalues

D = diag(D\_vec(sel\_ind)); % sorted and selected eigenvalue matrix

V = V\_temp(:,indx(sel\_ind)); % sorted and selected eigenvector matrix

X\_pca = V' \* X;

VV = V\_temp(:,indx);

X\_ppca = VV' \* X;

figure(1)

subplot(1,2,1);

scatter3(X(1,:),X(2,:),X(3,:)) % scatter de-biased, scaled data

title('De-biased, scaled data points')

xlabel('feature 1'); ylabel('feature 2'); zlabel('feature 3');

subplot(1,2,2);

scatter3(X\_ppca(3,:),X\_ppca(2,:),X\_ppca(1,:))

title('PCA results for de-biased, scaled data points in 3D')

xlabel('feature 1'); ylabel('feature 2'); zlabel('feature 3');

figure(2)

scatter(X\_pca(1,:),X\_pca(2,:))

title('PCA results for de-biased, scaled data points in 2D')

xlabel('feature 3'); ylabel('feature 2');

s = std(X);

q1 comp

clc; clear all;close all;

load('PCA\_comp1.mat');

[U\_orig,S\_orig,V\_orig] = svd(PCA\_comp1);

% Note that one of three singular values of A is approximately

% zero(3.91e-15). It's reasonable to expect that PCA will project

% 3-dimentional data to 2-dimentional.

%% (b) Compute the PCA

[m,n] = size(PCA\_comp1);

mu = mean(PCA\_comp1',2); % compute the mean(s)

X = (PCA\_comp1 - repmat(mu',1,n)); % de-biased, scaled data

covar = X \* X'; % compute the covariance matrix including the scaling 1/(n-1)

[V\_temp,D\_temp] = eig(covar); % compute an eigenvalue decomposition

[D\_vec,indx] = sort(diag(D\_temp),'descend'); %sort eigenvalues in descending order

sel\_ind = find(D\_vec>1e-5); % select non-zero eigenvalues

D = diag(D\_vec(sel\_ind)); % sorted and selected eigenvalue matrix

V = V\_temp(:,indx(sel\_ind)); % sorted and selected eigenvector matrix

X\_pca = V' \* X;

VV = V\_temp(:,indx);

X\_ppca = VV' \* X;

% According to "indx", which represents the index of eigenvalues in

% descending order, and "D\_vec" (eigenvalues), the third and the second features are the principal components

figure(1)

subplot(1,2,1);

scatter3(X(1,:),X(2,:),X(3,:)) % scatter de-biased, scaled data

title('De-biased, scaled data points')

xlabel('feature 1'); ylabel('feature 2'); zlabel('feature 3');

subplot(1,2,2);

scatter3(X\_ppca(3,:),X\_ppca(2,:),X\_ppca(1,:))

title('PCA results for de-biased, scaled data points in 3D')

xlabel('feature 1'); ylabel('feature 2'); zlabel('feature 3');

figure(2)

scatter(X\_pca(1,:),X\_pca(2,:))

title('PCA results for de-biased, scaled data points in 2D')

xlabel('feature 3'); ylabel('feature 2');

%% (c) Direct SVD of the de-biased, scaled data

[U\_norm,S\_norm,V\_norm] = svd(X);

S\_vec = diag(S\_norm);

sel\_ind\_svd = find(S\_vec>1e-5);

S = diag(S\_vec(sel\_ind\_svd));

U = U\_norm(:,sel\_ind\_svd);

X\_svd = U'\* X ;

X\_ssvd = U\_norm' \* X;

figure(3)

subplot(1,2,1);

scatter3(X(1,:),X(2,:),X(3,:)) % scatter de-biased, scaled data

title('De-biased, scaled data points')

xlabel('feature 1'); ylabel('feature 2'); zlabel('feature 3');

subplot(1,2,2);

scatter3(X\_ssvd(1,:),X\_ssvd(2,:),X\_ssvd(3,:))

title('PCA\\_svd results for de-biased, scaled data points in 3D')

xlabel('feature 1'); ylabel('feature 2'); zlabel('feature 3');

figure(4)

scatter(X\_ssvd(1,:),X\_ssvd(2,:))

title('PCA\\_svd results for de-biased, scaled data points in 2D')

xlabel('feature 1'); ylabel('feature ');

% We got different signs of the first component (by comparing U (by PCA) and V (by SVD), or

% X\_pca and X\_svd). Basically, the sign of the components is arbitrary and

% does not matter. Please see

% https://stats.stackexchange.com/questions/88880/does-the-sign-of-scores-or-of-loadings-in-pca-or-fa-have-a-meaning-may-i-revers

% for detailed discuss

Q1 t2

clc; clear all;

vbls = {'1','2','3','4'};

biplot(U(:,1:2),'scores',PCA\_comp1(:,1:2),'varlabels',vbls);

vbls = {'1','2','3','4'};

biplot(S(:,1:2),'scores',U(:,1:2),'varlabels',vbls);

[coefs,score] = pca(Y);

biplot(coefs(:,1:2),'scores',score(:,1:2),'varlabels',vbls);

eig(Y)

vbls = {'1','2','3','4'};

F = U\*S;

biplot(F(:,1:2),'scores',PCA\_comp1(:,1:2),'varlabels',vbls);

Q = pca(X);

biplot(pc(:,1:2),'Scores',score(:,1:2),'VarLabels',...

{'X1' 'X2' 'X3' 'X4'})

Undefined function or variable 'pc'.

biplot(COEFF(:,1:2),'Scores',SCORE(:,1:2),'VarLabels',...

{'X1' 'X2' 'X3' 'X4'})

biplot(COEFF(:,1:3),'Scores',SCORE(:,1:3),'VarLabels',...

{'X1' 'X2' 'X3' 'X4'})

Scaled = bsxfun(@rdivide, A, std(A))

Undefined function or variable 'A'.

Scaled = bsxfun(@rdivide, X, std(X))

[COEFF, SCORE, latent] = pca(Scaled)

biplot(COEFF(:,1:3),'Scores',SCORE(:,1:3),'VarLabels',...

{'X1' 'X2' 'X3' 'X4'})

biplot(COEFF(:,1:2),'Scores',SCORE(:,1:2),'VarLabels',...

{'X1' 'X2' 'X3' 'X4'})

Q1 t3

clc;

clear all;

load('PCA\_comp1.mat');

X = PCA\_comp1(:,1:4)';

%[W, pc] = princomp('PCA\_comp1.mat'); pc=pc'; W=W';

mu = mean(PCA\_comp1',2); % compute the mean(s)

n = size(X,2);

A = X - mu\*ones(1,n);

[U\_norm,S\_norm,V\_norm] = svd(X);

S\_vec = diag(S\_norm);

sel\_ind\_svd = find(S\_vec>1e-5);

S = diag(S\_vec(sel\_ind\_svd));

U = U\_norm(:,sel\_ind\_svd);

X\_svd = U'\* X ;

X\_ssvd = U\_norm' \* X;

figure(3)

subplot(1,2,1);

scatter3(X(1,:),X(2,:),X(3,:)) % scatter de-biased, scaled data

title('De-biased, scaled data points')

xlabel('feature 1'); ylabel('feature 2'); zlabel('feature 3');

subplot(1,2,2);

scatter3(X\_ssvd(1,:),X\_ssvd(2,:),X\_ssvd(3,:))

title('PCA\\_svd results for de-biased, scaled data points in 3D')

xlabel('feature 1'); ylabel('feature 2'); zlabel('feature 3');

figure(4)

scatter(X\_ssvd(1,:),X\_ssvd(2,:))

title('PCA\\_svd results for de-biased, scaled data points in 2D')

xlabel('feature 1'); ylabel('feature ');

mnrfit

3.

Case 1 Linear Regression:

plotDataLR(x,y);

hold on;

plot(X,[ones(size(x,1),1)X]\*beta,’-‘)

legend(‘Training data’,’Linear regression’)

hold off

function plotDataLR(x,y)

figure;

load(‘Comp1\_IE529.mat’);

X = lift\_kg’;

Y = putt\_m;

M = length(y);

Plot(x, y, ‘rx’, ‘MarkerSize’, 10);

End

Case 2: Polynomial Regression

%Load data

load(Comp1)

Q1.

First question: Pseudo-code:

1. Preprocessing: Subtract the mean and divide by sigma
2. Compute covariance matrix SIGMA
3. Compute ‘eigenvectors’ of matrix SIGMA
4. Select k appropriate principal components

Third question

1. There are 2 obvious clusters.
2. There are 2 separate species.