ECE175 HW2 Report

Di Guan (A91041815)

Problem 1.

Consider the matrix A, where

$$A = \begin{bmatrix} 5 & b \\ b & 5 \end{bmatrix}$$

and the function f(x), where

$$f(x) = x^T Ax$$

$$x = \begin{bmatrix} x_1 \\ x_2 \end{bmatrix}$$

and

$$-100 <= x_1 <= 100, -100 <= x_2 <= 200$$

(a). make surface and contour plot of the function f(x) for the values of b, where b ∈{1,5,10}

- when b=1,

$$f(x) = [\, x_1, x_2 \,] \begin{bmatrix} 5 & 1 \\ 1 & 5 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} = 2x_1x_2 + 5x_1^2 + 5x_2^2$$

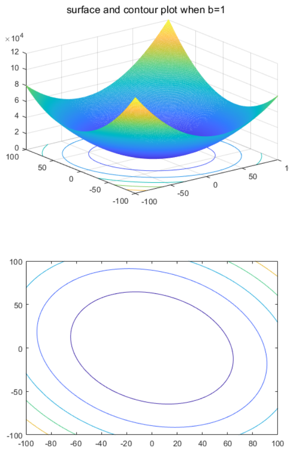
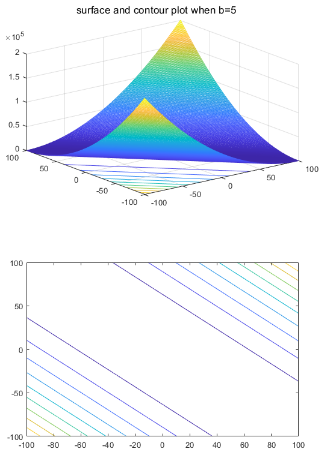
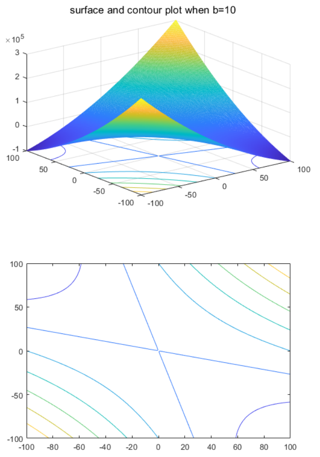
- when b=5,

$$f(x) = [\, x_1, x_2 \,] \begin{bmatrix} 5 & 5 \\ 5 & 5 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} = 10x_1x_2 + 5x_1^2 + 5x_2^2$$

- when b=10,

$$f(x) = [\, x_1, x_2 \,] \begin{bmatrix} 5 & 10 \\ 10 & 5 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} = 20x_1x_2 + 5x_1^2 + 5x_2^2$$

plots:

b=1	b=5	b=10
		

code:

```
%%(a)surface and contour plot of function f(x)=x'Ax
[x1,x2]=meshgrid(-100:1:100);
%when b=1;A=[5 1;1 5]
fx=2.*x1.*x2+5*x1.^2+5*x2.^2;
figure
subplot(2,1,1)
meshc(x1,x2,fx);
subplot(2,1,2)
contour(x1,x2,fx);
sgtitle('surface and contour plot when b=1')

%when b=5;A=[5 5;5 5]
fx=10.*x1.*x2+5*x1.^2+5*x2.^2;
figure
subplot(2,1,1)
meshc(x1,x2,fx);
subplot(2,1,2)
contour(x1,x2,fx);
sgtitle('surface and contour plot when b=5')

%when b=10; A=[5 10;10 5]
fx=20.*x1.*x2+5*x1.^2+5*x2.^2;
figure
subplot(2,1,1)
meshc(x1,x2,fx);
subplot(2,1,2)
contour(x1,x2,fx);
sgtitle('surface and contour plot when b=10')
```

(b). what are the eigenvalues of A for the three values of b?

b=1	b=5	b=10
λ = 4, 6	λ = 0, 10	λ = -5,15

code:

```
%%(b)eigenvalues of A
%when b=1, the result returns eigenvalues of 4 and 6
A=[5 1;1 5];
e1=eig(A);
%when b=5, the result returns eigenvalues of 0 and 10
A=[5 5;5 5];
e2=eig(A);
%when b=1, the result returns eigenvalues of -5 and 15
A=[5 10;10 5];
e3=eig(A);
```

(c). how would you characterize the positive-definiteness of A for each b. That is, for each b, is A positive-definite, positive semi-definite or neither?

Based on the definition of positive-definite and positive semi-definite matrix,

- Matrix A is positive definite matrix if

$$x^T Ax > 0 for x \in R^n$$

  where x is non-zero vectors ;

- Matrix A is positive semi-definite matrix if

$$x^T Ax >= 0 for x \in R^n$$

By definition, it is a little bit hard to determine whether A is positive definite matrix or not just simply by visualization of the results or plots in part (a).

However, the properties of the positive-definite and positive semi-definite matrix in terms of eigenvalues can also determine whether the matrix is positive-definite or not.

It is being said that,

- Matrix A is positive definite if and only if its eigenvalues are positive.
- Matrix A is positive semi-definite if and only if its eigenvalues are non-negative.

Thus, based on the results we obtained for eigenvalues in terms of different values of b.

- When b=1, the eigenvalues of A are all positive (λ = 4, 6), so A is positive definite matrix
- When b=5, the eigenvalues of A are all non-negative (λ = 0, 10), so A is positive semi-definite matrix
- When b=10, the eigenvalues of A have negative values (λ = -5,15), so A is neither positive definite nor positive semi-definite matrix. In fact, A is indefinite matrix since it has both positive and negative eigenvalues.

(d). if you were given the three words "bowl", "saddle", and "one-dimensional" to describe the function f(x), how would you associate these descriptions to the three possible matrix classes.

From the results from part (a), we can say that

- when b=1, A is positive definite matrix. Its surface of the quadratic form is bowl shape and it has a minimum point. This is probably because the function has positive curvature along two eigenvector directions.
- when b=5, A is positive semi-definite matrix. Its surface of quadratic form is one-dimensional shape. This is because one of eigenvalues of the matrix is 0 and it will not have curve along the associated eigenvectors.
- when b=10, it is a saddle shape. The reason of that is probably because there is a negative eigenvalues and the it will have a curve along the negative direction of associated eigenvector so that it form a saddle-shape.

(e). in which of the three cases is f(x) a norm? Why?

By definition, which is covered on our lecture, any norm must obey the following properties:

- positive-definiteness: ||x||>=0 and ||x||=0 when x =0
- homogeneity: ||ax||=|a|*||x|| where a is scalar
- triangle Inequality: ||x+y||<= ||x||+||y||

In this question, we have to check those three properties to determine f(x) is a norm or not

- when b=1, from the result of part(a), we can see that f(x) is non-negative for all x. Also, f(x) = 0 when x=0, which satisfies the first properties. Furthermore, f(x) also satisfies the second and third properties when b=1. Therefore, we can say that f(x) is a norm when b=1.
- when b=5, from the results of part(a), we can see that f(x) is non-negative for all x, but f(x) = 0 for some x that is not zero vector, which contradicts the first properties, positive-definiteness. Thus f(x) is not a norm when b=5.
- when b=10, from the results of part(a), we can see that f(x) is negative for some x, which contradict the first statement. Thus, it is not a norm when b=10.

Problem 2.

(a). generate 500 points from a 2D Gaussian of zero mean and identity covariance.

Here, the Mahalabonis distance is

$$distance = x^T {\Sigma_0}^{-1} x$$

defined by the covariance matrix

$$\Sigma_0 = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}$$

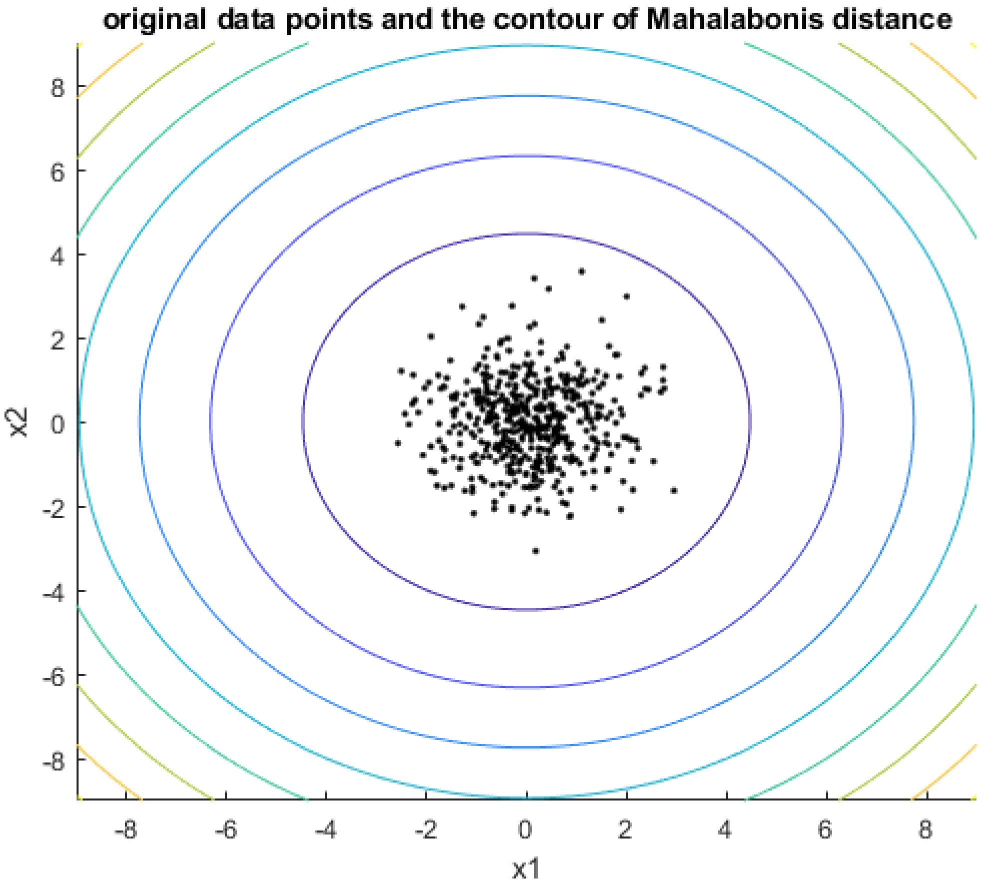
where

$$x = \begin{bmatrix} x_1 \\ x_2 \end{bmatrix}$$

since we use randn command, those 500 data points are generated randomly with zero mean and unit variance. To check the covariance matrix is correct, we obtain the covariance matrix of generated data.

$$\Sigma = \begin{bmatrix} 1.0578 & -0.0195 \\ -0.0195 & 0.9508 \end{bmatrix}$$

which is close to identity covariance matrix



code:

```
%%(a)generate 500 points of 2-D Gaussian distribution with zero mean and
%%unit covariance
%number of data points
n=500;
%variable x1
x1=randn(n,1);
%variable x2
x2=randn(n,1);

%%%%check the covariance

%%1. check the mean of the variable x1 and x2
x1_bar=mean(x1);
%return the value of -0.0897, which is close to 0
x2_bar=mean(x2);
%return the value of -0.0466, which is close to 0

%%2. check the variance of the variable x1 and x2
var_x1=sum((x1 - x1_bar).*(x1 - x1_bar))/(length(x1) - 1);
%return the value of 1.0578, which is close to 1
var_x2=sum((x2 - x2_bar).*(x2 - x2_bar))/(length(x2) - 1);
%return the value of 0.9508, which is close to 1

%%3. check the covariance between x1 and x2, cov(x1,x2)
```

```
cov_x1_x2=sum((x1 - x1_bar).*(x2 - x2_bar))/(n - 1);
%return the value of -0.0195, which is close to 0,implying that the varaible x1
and x2 is independent

%%4. construct the covariance matrix
cov_matrix=cov(x1,x2);
%return the covariance matrix
% covariance =
%
%     1.0578    -0.0195
%     -0.0195    0.9508
%it is close to an identity covariance matrix, which is
%     1     0
%     0     1

%%%%%plot the generated points
figure
scatter(x1,x2,'.black')
hold on
%plot the contour of mahalabonis distance
[fx1,fx2]=meshgrid(-9:0.1:9);
z=fx1.^2+fx2.^2;
contour(fx1,fx2,z)
title('original data points and the contour of Mahalabonis distance');
xlabel('x1');
ylabel('x2');
```

(b). apply the transformation T to the points

- what is the new covariance matrix?

  After the transformation

  $$T=\begin{bmatrix} cos\frac{\pi}{4} & -sin\frac{\pi}{4} \\ sin\frac{\pi}{4} & cos\frac{\pi}{4} \end{bmatrix}\begin{bmatrix} 3 & 0 \\ 0 & 1 \end{bmatrix}$$

  the data points becomes

  $$y=Tx$$

  the new covariance matrix becomes

  $$\Sigma_1=T\Sigma_0T^T=TT^T=\begin{bmatrix} 5 & 4 \\ 4 & 5 \end{bmatrix}$$

  while the real covariance matrix here is

  $$\Sigma'=T\Sigma T^T=\begin{bmatrix} 5.0323 & 4.0737 \\ 4.0737 & 5.0766 \end{bmatrix}$$

  which is close to the new covariance matrix

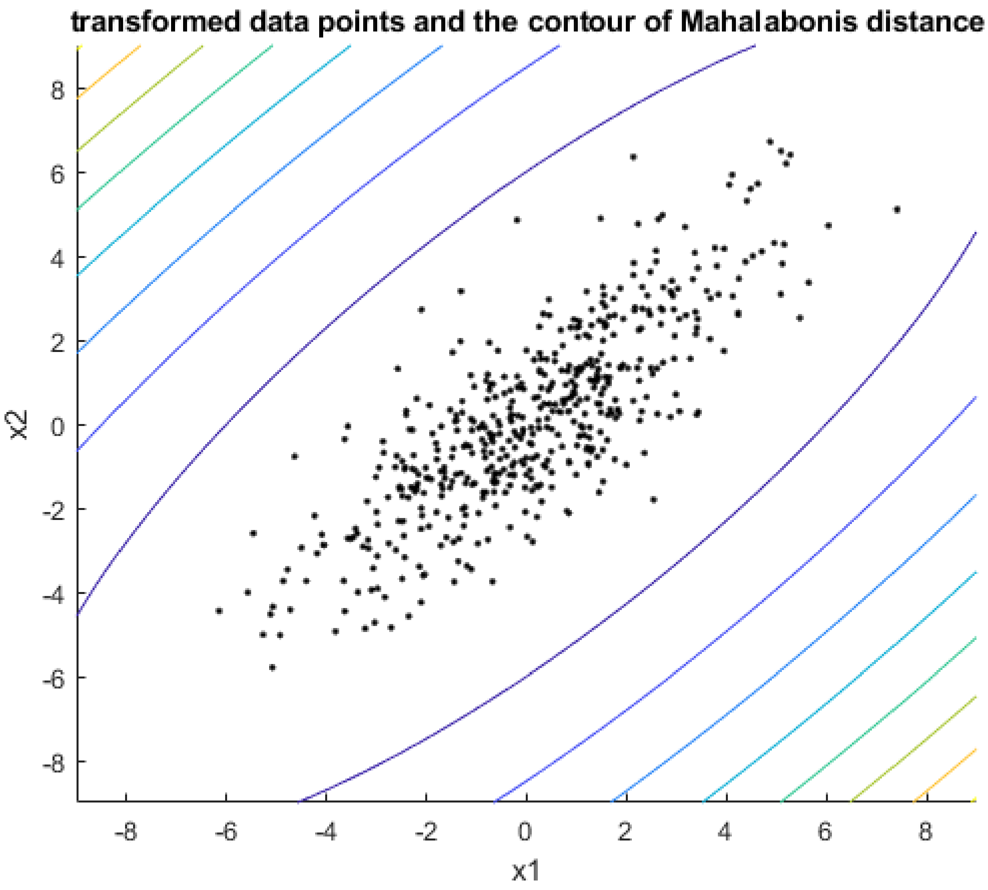
- how does the associated Mahalanobis distance differ from that of (a)?

  Here, the Mahalabonis distance is

  $$distance=x^T{\Sigma_1}^{-1}x$$

  defined by the covariance matrix

$$\Sigma_1 = \begin{bmatrix} 5 & 4 \\ 4 & 5 \end{bmatrix}$$



code:

```
%%%(b)apply the transformation to the points
%obtain the new covariance matrix
T=[cos(pi/4) -sin(pi/4);sin(pi/4) cos(pi/4)]*[3 0;0 1];
new_cov_y=T*cov_matrix*T';
new_cov_y_truth=T*T';
inv_new_cov=inv(new_cov_y_truth);
X=[x1';x2'];
new_X=T*X;
figure
scatter(new_X(1,:),new_X(2,:),'.black')
hold on
[fx1,fx2]=meshgrid(-9:0.1:9);
z=inv_new_cov(1,1).*fx1.^2+inv_new_cov(2,2).*fx2.^2+
(inv_new_cov(1,2)+inv_new_cov(2,1)).*fx1.*fx2;
contour(fx1,fx2,z)
title('transformed data points and the contour of Mahalabonis distance');
xlabel('x1');
ylabel('x2');
```

(c). suppose we want to transform the data of (a) so that the new Mahalanobis distance is

$$x^T Ax$$

where

$$A = \begin{bmatrix} 5 & 0 \\ 0 & 2 \end{bmatrix}$$

- What transformation should we apply to the data?

  now Mahalabonis distance becomes

  $$distance = x^T A x = x^T {\Sigma_1}^{-1} x$$

  which means

  $$A = {\Sigma_1}^{-1} = \begin{bmatrix} 5 & 0 \\ 0 & 2 \end{bmatrix}$$

  thus,

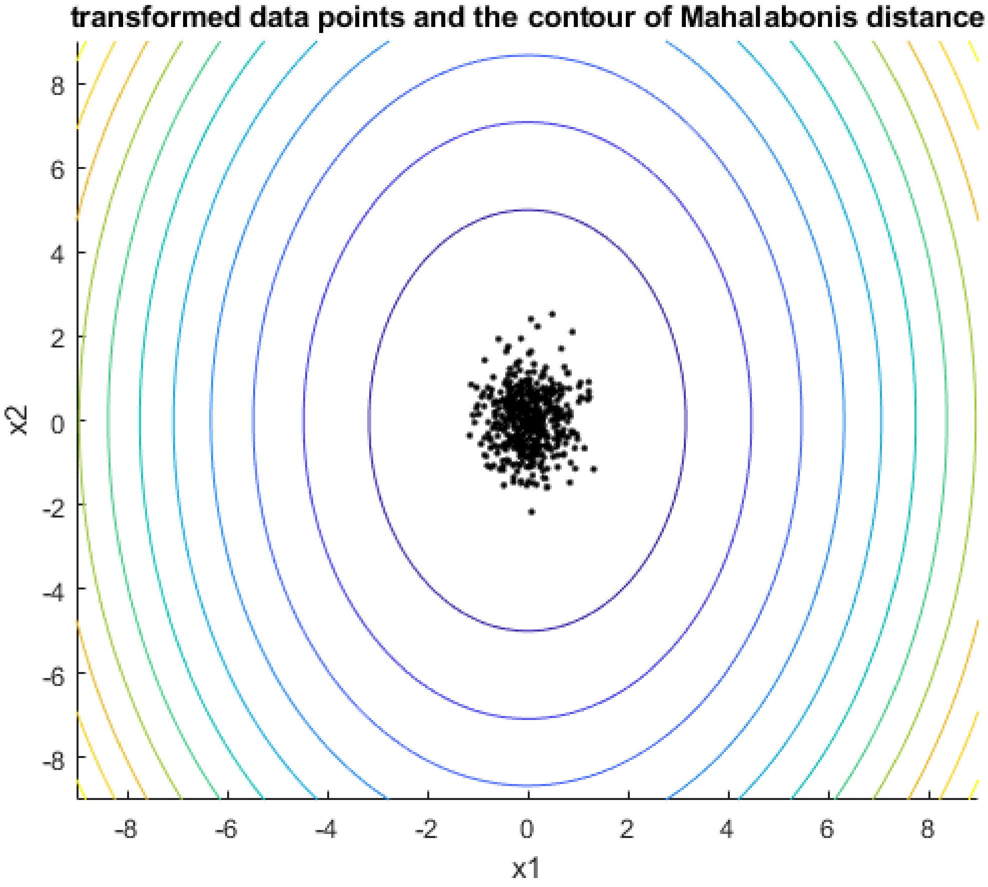
  $$\Sigma_1 = \begin{bmatrix} \frac{1}{5} & 0 \\ 0 & \frac{1}{2} \end{bmatrix}$$

  also,

  $$\Sigma_1 = T \Sigma_0 T^T = T T^T = T^2$$

  hence, the transformation is

  $$T = \begin{bmatrix} 0.4472 & 0 \\ 0 & 0.7071 \end{bmatrix} = \begin{bmatrix} \frac{1}{\sqrt{5}} & 0 \\ 0 & \frac{1}{\sqrt{2}} \end{bmatrix}$$



code:

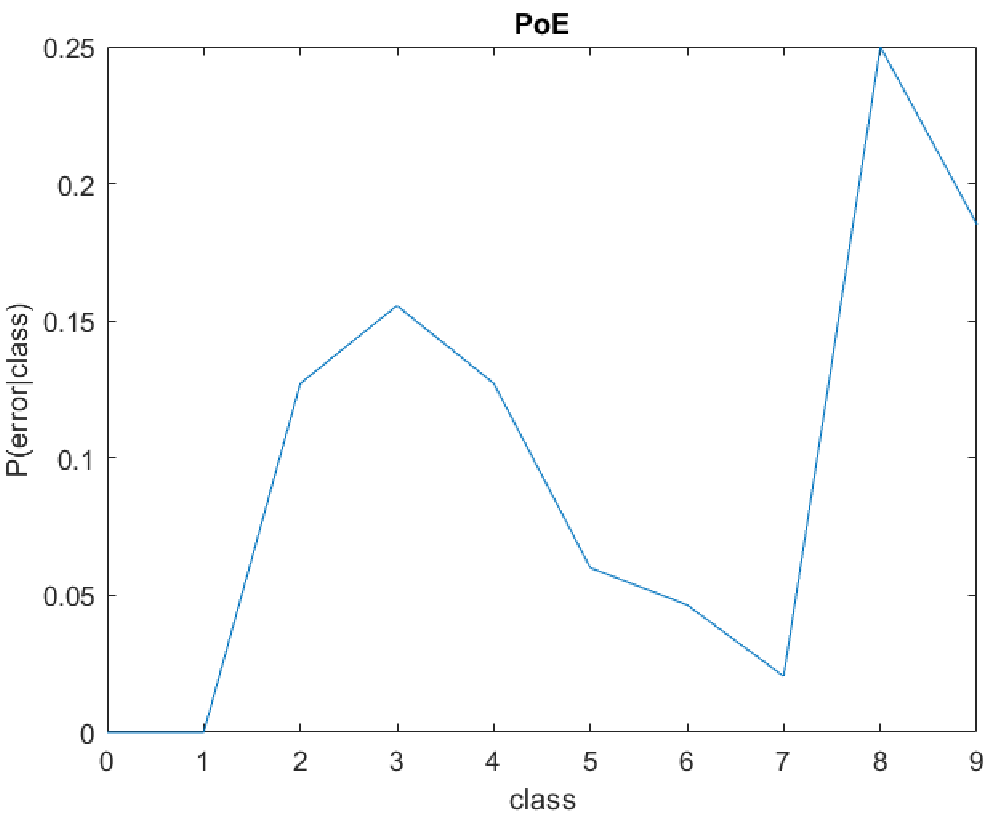
```
%%%(c)
A=[5 0;0 2];
new_cov_z_truth=inv(A);
T=new_cov_z_truth^(1/2);
new_X=T*X;
figure
scatter(new_X(1,:),new_X(2,:),'.black')
hold on
[fx1,fx2]=meshgrid(-9:0.1:9);
z=A(1,1).*fx1.^2+A(2,2).*fx2.^2+(A(1,2)+A(2,1)).*fx1.*fx2;
contour(fx1,fx2,z)
title('transformed data points and the contour of Mahalabonis distance');
xlabel('x1');
ylabel('x2');
```

Problem 3.

- **Background**: We will working with the dataset of hand-written numerals in a supervised fashion. Thus the classes for our classification problem are C={0,1,2,...9}.
- **Goal**: classify the test images into these classes such that the test error rates are minimum.
- **Methods**: use the nearest neighbor classification for the current assignment. The dataset consists of a set of 5000 training images and 500 test images of hand-written numeral, where each image is 28*28 pixels with intensity values ranging from 0-255. To simplify the experiment, we shall answer the question of *which feature to use*? Assume the feature space to be 28*28 dimensional, i.e. the observation or feature X are all the pixel intensity values of the image.

1. Use the training image and the ground truth information to train a Nearest Neighbor Classifier using Euclidean distance metric and test it on the test database.

Class	PoE	# error images	# images
0	0	0	42
1	0	0	67
2	0.1273	7	55
3	0.1556	7	45
4	0.1273	7	55
5	0.06	3	50
6	0.0465	2	43
7	0.0204	1	49
8	0.25	10	40
9	0.1852	10	54



Code:

```
%%%%part(a)
%load data and assign variable
data=load('E:\UC Social Dead\2020Winter\ECE175\Homework\hw2\HW2_Data\data.mat');
label=load('E:\UC Social Dead\2020Winter\ECE175\Homework\hw2\HW2_Data\label.mat');
imageTest=data.imageTest;
imageTrain=data.imageTrain;
labelTest=label.labelTest;
labelTrain=label.labelTrain;
%initialize the error i=list
error=zeros(10,1);
misclassified=[];
%use tabulate to conduct stats on label test data
stats_test=tabulate(labelTest);
for i =1:500
    [min_euclidean_dis,index]=euclidean_dis(imageTest(:,:,i),imageTrain);
    if labelTest(i) ~= labelTrain(index)
        error(labelTest(i)+1)=error(labelTest(i)+1)+1;
        misclassified(1,end+1)=i;
        misclassified(2,end)=index;
    end
end

%obtain the error rate for each class
error_rate=error./stats_test(:,2);

%plot the PoE for each class
figure
plot(0:9,error_rate)
xlabel('class')
ylabel('P(error|class)')
```

```
title('PoE')
```

function euclidean_dis:

```
function [min_euclidean_dis,index]=euclidean_dis(imageTest,imageTrain)
%input:
%1.imageTrain:there are 5000 images for imageTrain with the size of
%28*28*5000
%2.imageTest:there is only one image for imageTest with the size of 28*28
%output:
%after calculate the euclidean distance for the size of 28*28*5000, we
%obtain the variable 'euclidean_dis' with the size of 1*1*5000
%output the minimum value of euclidean_dis and its index

%obtain the size of imageTrain
[height,weight,depth]=size(imageTrain);
%reshape the imageTest to the same size of imageTest
imageTest=imageTest+zeros(height,weight,depth);
%calculate the euclidean distance
euclidean_dis=sqrt(sum(sum((imageTest-imageTrain).^2)));
[min_euclidean_dis,index]=min(euclidean_dis);
end
```

2. calculate the total error rate

$$TotalErrorRate = \frac{\#error}{500} = \frac{47}{500} = 0.094$$

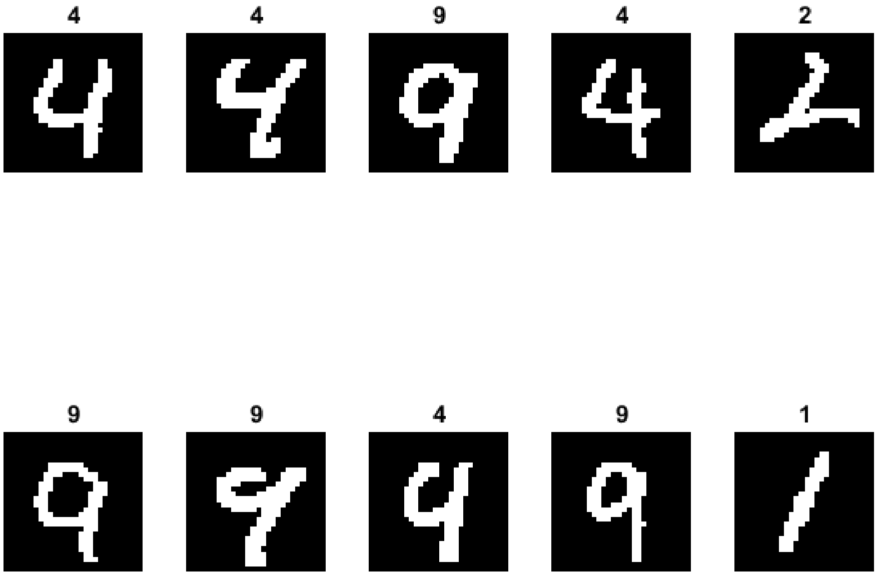
Code:

```
%obtain the total error rate
total_error_rate=sum(error)/500;
```

3. For 5 test image that was misclassified by the NN Classifier, display the test image and the image from the training database which is closest to the test image.

The first row of the plot is represented as the first 5 misclassified test images and its corresponding ground truth label. The second row of the plot is represented as the corresponding closest image in the training database to the misclassified test images.

For those misclassified test images, they are misclassified due to their unclear visualization, they are really close to the corresponding images in training database. Thus, it is not surprising to mismatch those images.



Code:

```
%%%%part(c)
figure
for i=1:5
    subplot(2,5,i)
    imshow(imageTest(:,:,misclassified(1,i)));
    title(labelTest(misclassified(1,i)))
    subplot(2,5,i+5)
    imshow(imageTrain(:,:,misclassified(2,i)));
    title(labelTrain(misclassified(2,i)))
end
```