Lecture 5

Naïve Bayes Classifier

http://www.ece.uah.edu/~dwpan/course/ee610/code/Bayes/naive_bayes_classifier.m

```
clear all;
close all;
M = 100;   % number of samples generated for each class
data = zeros(2*M, 2);
label = zeros(2*M, 1);

% Generate data entries for Class 1
m1 = [3,3];
c1 = [2,0;0,2];
rng default  % For reproducibility
r1 = mvnrnd(m1,c1,M);
data(1:M,:) = r1;
label(1:M) = 1;

% Generate data entries for Class 2
m2 = [9,9];
c2 = [2,0;0,2];
rng default  % For reproducibility
r2 = mvnrnd(m2,c2,100);

data(M+1:2*M,:) = r2;
label(M+1:2*M) = 2;

% Plot the data samples of the two classes
gscatter(data(:,1),data(:,2),label,'rb','ox')
```

$$d_j(\boldsymbol{x}) = \boldsymbol{x}^T C^{-1} m_j - \frac{1}{2} m_j^T C^{-1} m_j$$

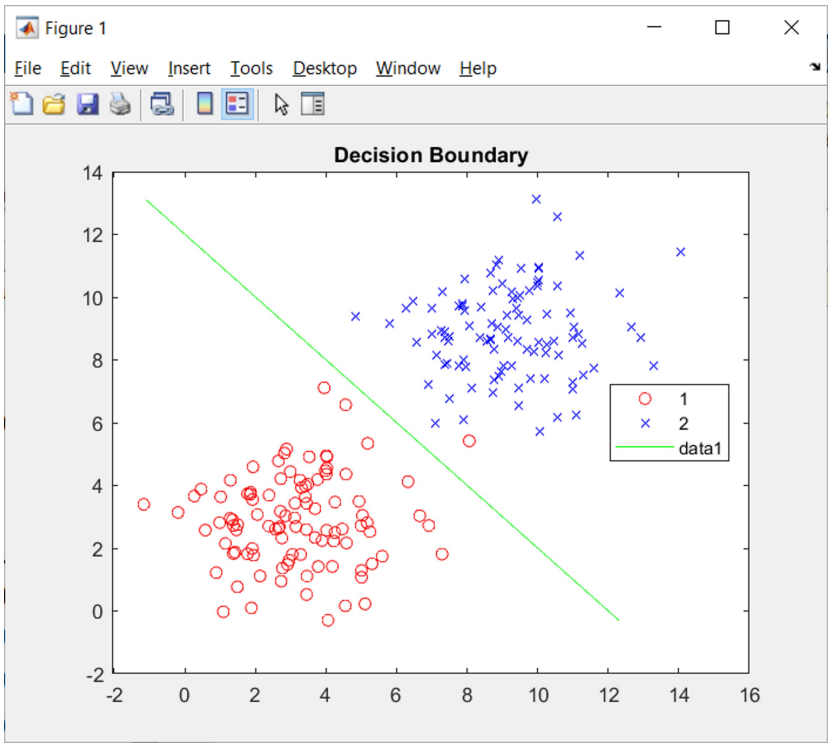
```
% means are column vector in the book
m1 = m1';
m2 = m2';

% Decision function coefficients for Class 1
U = inv(c1)*m1
const1 = -(1/2)*m1' *inv(c1) * m1

% Decision function coefficients for Class 2
V = inv(c2)*m2
const2 = -(1/2)*m2' *inv(c2) * m2

% Calculate the coefficients for the decision boundary line
% Straight line equation: ax + by + c = 0
a = V(1) - U(1)
b = V(2) - U(2)
c = const2 - const1

% Plot the decision boundary
y = min(data(:,2)): 0.01: max(data(:,2));
x = (-b/a)*y - c/a;
hold on;
plot(x, y, 'g');
title('Decision Boundary')
```



Sklearn

http://www.ece.uah.edu/~dwpan/course/ee610/code/Introduction/naive_bayes_demo.py

```
import numpy as np
from sklearn.naive_bayes import GaussianNB
clf = GaussianNB()

from sklearn.datasets import load_iris
train_samples = load_iris()
X = train_samples.data
Y = train_samples.target

clf.fit(X, Y)

clf.score(X, Y)

# Verify the classification accuracy
Y_pred = clf.predict(X)

num_correct = np.sum(Y == Y_pred)
num_sample = np.size(Y_pred)
num_correct/num_sample
```

http://www.ece.uah.edu/~dwpan/course/ee610/code/KNN/csv_import.py

```
import numpy as np
infile = r"C:\Users...mvnrnd.csv"
dataset = np.loadtxt(infile, delimiter=',')
X = dataset[:, 0:2]
y = dataset[:,2]
```

Parametric Form for $p(C_k|\mathbf{x})$

- Assume that the class-conditional densities are Gaussian.
- We consider first two classes, and assume that all classes share the same covariance matrix.
- Thus the density for class C_k is given by

$$p(\mathbf{x}|\mathcal{C}_k) = \frac{1}{(2\pi)^{D/2}} \frac{1}{|\boldsymbol{\Sigma}|^{1/2}} \exp\left\{-\frac{1}{2}(\mathbf{x}-\boldsymbol{\mu}_k)^{\mathrm{T}}\boldsymbol{\Sigma}^{-1}(\mathbf{x}-\boldsymbol{\mu}_k)\right\}$$

Decision functions (with a common covariance matrix $\boldsymbol{C}$, where $\boldsymbol{C}^{\mathrm{T}} = \boldsymbol{C}$):

Mahal Distances

$$d_1(\mathbf{x}) = \ln P(\omega_1) - \frac{1}{2}\ln|\boldsymbol{C}| - \frac{1}{2}[(\mathbf{x}-\mathbf{m_1})^{\mathrm{T}}\mathrm{C}^{-1}(\mathbf{x}-\mathbf{m_1})] \rightarrow D_M^2(\mathbf{x}, \mathbf{m}_1)$$

$$d_2(\mathbf{x}) = \ln P(\omega_2) - \frac{1}{2}\ln|\boldsymbol{C}| - \frac{1}{2}[(\mathbf{x}-\mathbf{m_2})^{\mathrm{T}}\mathrm{C}^{-1}(\mathbf{x}-\mathbf{m_2})] \rightarrow D_M^2(\mathbf{x}, \mathbf{m}_2)$$

Assuming equal class probabilities:

$$d_1(\mathbf{x}) = d_2(\mathbf{x}) \Rightarrow (\mathbf{m_1}-\mathbf{m_2})^{\mathrm{T}}\mathbf{C}^{-1}\mathbf{x} = \frac{1}{2}(\mathbf{m_1}^{\mathrm{T}}\mathbf{C}^{-1}\mathbf{m_1} - \mathbf{m_2}^{\mathrm{T}}\mathbf{C}^{-1}\mathbf{m_2})$$

- If the covariance matrix is identical. then

$$d_j(\mathbf{x}) = \ln P(\omega_j) + \mathbf{x}^T\mathbf{C}^{-1}\mathbf{m}_j - \frac{1}{2}\mathbf{m}_j^T\mathbf{C}^{-1}\mathbf{m}_j$$

- If all classes are equally likely and the covariance matrix is an identity matrix, then

$$d_j(\mathbf{x}) = \mathbf{x}^T\mathbf{m}_j - \frac{1}{2}\mathbf{m}_j^T\mathbf{m}_j \qquad j = 1, 2, \ldots, W$$

- The same decision function for a minimum-distance classifier, which is optimal in the Bayes sense if
  - The pattern classes are Gaussian.
  - All covariance matrices are equal to identity matrix.
  - All classes are equally likely.

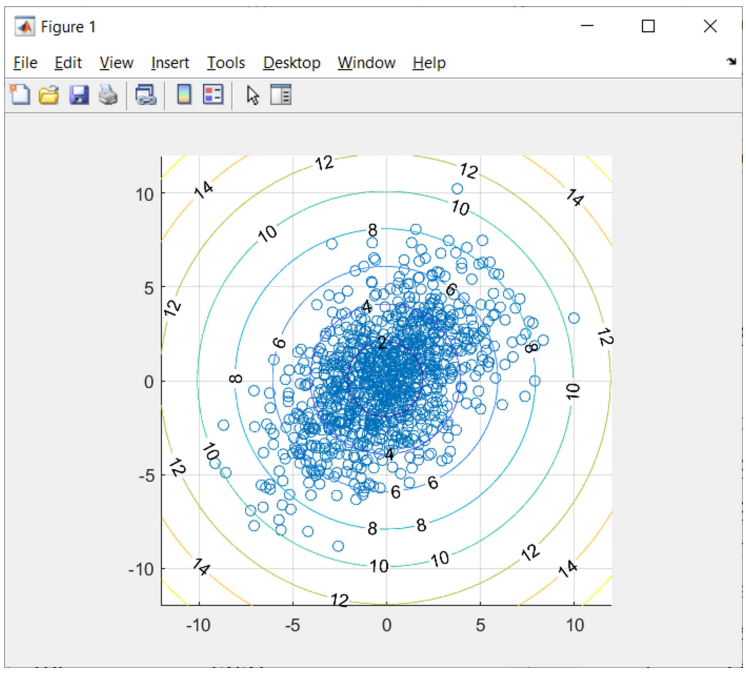
$C_1 = C_2$

Optimal Bayes Classifier $\triangleq$ Minimum-distance classifier,

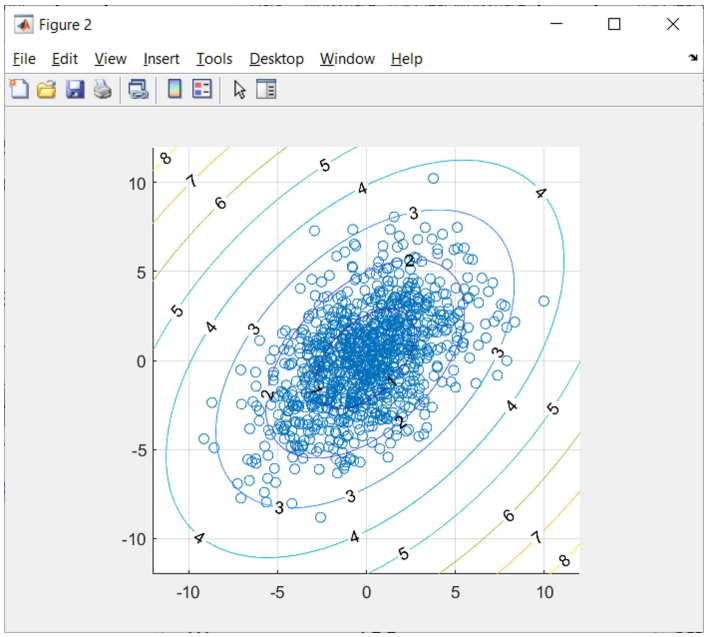
however, distance (Euclidean) $\longrightarrow$ Mahalanobis distance

http://www.ece.uah.edu/~dwpan/course/ee610/code/KNN/mahal_dist_demo.m

```
close all
m = [0, 0]
C =[8 4; 4 8]
eig(C)
Q = mvnrnd(m,C,1000);
figure;
scatter(Q(:,1),Q(:,2))
grid
axis equal
axis square
smean = mean(Q)  % Sample mean
scov = cov(Q)    % Sample covariance matrix
```




```
hold on;
x = linspace(-12,12);
y = linspace(-12,12);
[X,Y] = meshgrid(x,y);
% Euclidean Distances
D_Euc = sqrt((X - smean(1)).^2 + (Y - smean(2)).^2);
contour(X,Y,D_Euc,'ShowText', 'on')
```




```
% Mahalanobis Distance from the sample mean vector
D_Mah = zeros(100, 100);
for i = 1: 100
   for j = 1: 100
      D_Mah(i,j) = pdist2([X(i,j),Y(i,j)], smean, 'mah',
scov);
   end
end

figure
scatter(Q(:,1),Q(:,2))
grid
axis equal
axis square
hold on;

contour(X,Y,D_Mah,'ShowText', 'on')
```