EE 610, ML Fundamentals

Discriminant Analysis

Dr. W. David Pan

Dept. of ECE

UAH

Topics

- Discriminant Analysis
- Discriminant Functions
- Decision Boundaries
- Fisher’s Linear Discriminant
- Linear Discriminant Analysis
- Quadratic Discriminant Analysis
- Implementations

Discriminant Analysis

- Discriminant analysis classifies data by finding linear combinations of features.
- Discriminant analysis assumes that different classes generate data based on Gaussian distributions.
- Training a discriminant analysis model involves finding the parameters for a Gaussian distribution for each class.
- The distribution parameters are used to calculate boundaries, which can be linear or quadratic functions. These boundaries are used to determine the class of new data.
- Best used if …
  - You need a simple model that is easy to interpret.
  - Memory usage during training is a concern.
  - When you need a model that is fast to predict.

Linearly Separable Classes

- The goal in classification is to take an input vector $\boldsymbol{x}$ and to assign it to one of K discrete classes C_k where $k = 1, \ldots, K$.
- In the most common scenario, the classes are taken to be disjoint, so that each input is assigned to one and only one class.
- The input space is thereby divided into *decision regions* whose boundaries are called *decision boundaries* or *decision surfaces*.
- Here we consider linear models for classification, where the decision surfaces are linear functions of the input vector $\boldsymbol{x}$ and hence are defined by $(D - 1)$-dimensional hyperplanes within the *D*-dimensional input space.
- Data sets whose classes can be separated exactly by linear decision surfaces are said to be *linearly separable*.

Discriminant Functions

- A discriminant is a function that takes an input vector $\boldsymbol{x}$ and assigns it to one of K classes, denoted C_k.
- Here we restrict attention to *linear discriminants*, for which the decision surfaces are hyperplanes.
- we consider first the case of two classes and then investigate the extension to more than two classes.
- The simplest representation of a linear discriminant function is obtained by taking a linear function of the input vector so that $y(\boldsymbol{x}) = \boldsymbol{w}^T\boldsymbol{x} + w_0$, where $\boldsymbol{w}$ is called a weight vector, and w_0 is a bias.
- An input vector $\boldsymbol{x}$ is assigned to class C_1 if $y(\boldsymbol{x}) \geq 0$ and to class C_2 otherwise.
- The corresponding decision boundary is therefore defined by the relation $y(\boldsymbol{x}) = 0$, which corresponds to a $(D - 1)$-dimensional hyperplane within the D-dimensional input space.

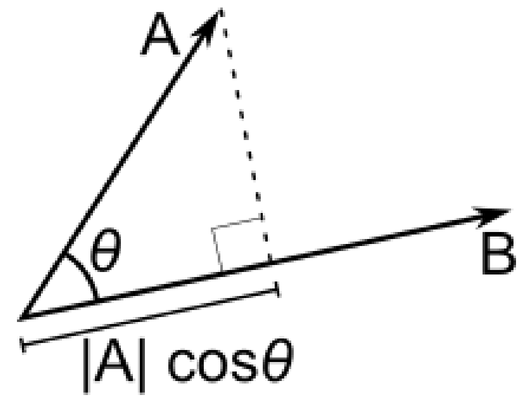
- Consider two points $\boldsymbol{x}_A$ and $\boldsymbol{x}_B$, both of which lie on the decision surface:
- Because $y(\boldsymbol{x}_A) = y(\boldsymbol{x}_B) = 0$, we have $\boldsymbol{w}^T(\boldsymbol{x}_A - \boldsymbol{x}_B) = 0$, and hence the vector $\boldsymbol{w}$ is orthogonal to every vector lying within the decision surface, and so
- $\boldsymbol{w}$ determines the orientation of the decision surface.
- Similarly, if $\boldsymbol{x}$ is a point on the decision surface, then $y(\boldsymbol{x}) = 0$, and so the normal distance from the origin to the decision surface is given below, where the bias parameter w_0 determines the location of the decision surface.

$$\frac{\boldsymbol{w}^T\boldsymbol{x}}{||\boldsymbol{w}||} = -\frac{w_0}{||\boldsymbol{w}||}$$

Inner Product and Projection

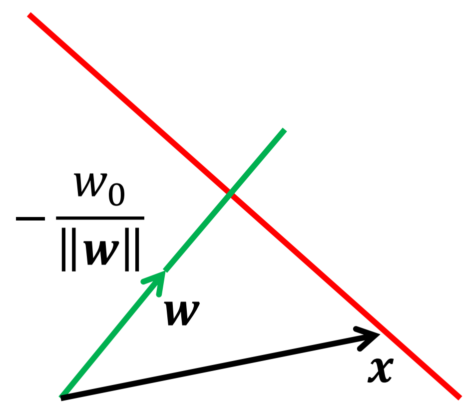
- The inner product of two same-length column vectors A and B is given by A^TB, and $A^TB = B^TA$.
- $A^TB = B^TA = \|A\|\|B\|cos\theta$
- The projection of A onto B is then:

$$\|A\|cos\theta = \frac{\|A\|(A^TB)}{\|A\|\|B\|} = \frac{A^TB}{\|B\|} = \frac{B^TA}{\|B\|}$$



if $\boldsymbol{x}$ is a point on the decision surface, then $\frac{\boldsymbol{W}^T\boldsymbol{x}}{\|\boldsymbol{w}\|}$ is the projection of the point $\boldsymbol{x}$ onto the weight vector $\boldsymbol{W}$. The projection remains the same regardless of the location of $\boldsymbol{x}$.

$$\frac{\boldsymbol{w}^T\boldsymbol{x}}{\|\boldsymbol{w}\|} = -\frac{w_0}{\|\boldsymbol{w}\|}$$



Geometry

if $\boldsymbol{x}$ is a point on the decision surface, then

$$\frac{\boldsymbol{w}^T\boldsymbol{x}}{\|\boldsymbol{w}\|} = -\frac{w_0}{\|\boldsymbol{w}\|}$$

$$y(\boldsymbol{x}) = \boldsymbol{w}^T\boldsymbol{x} + w_0$$

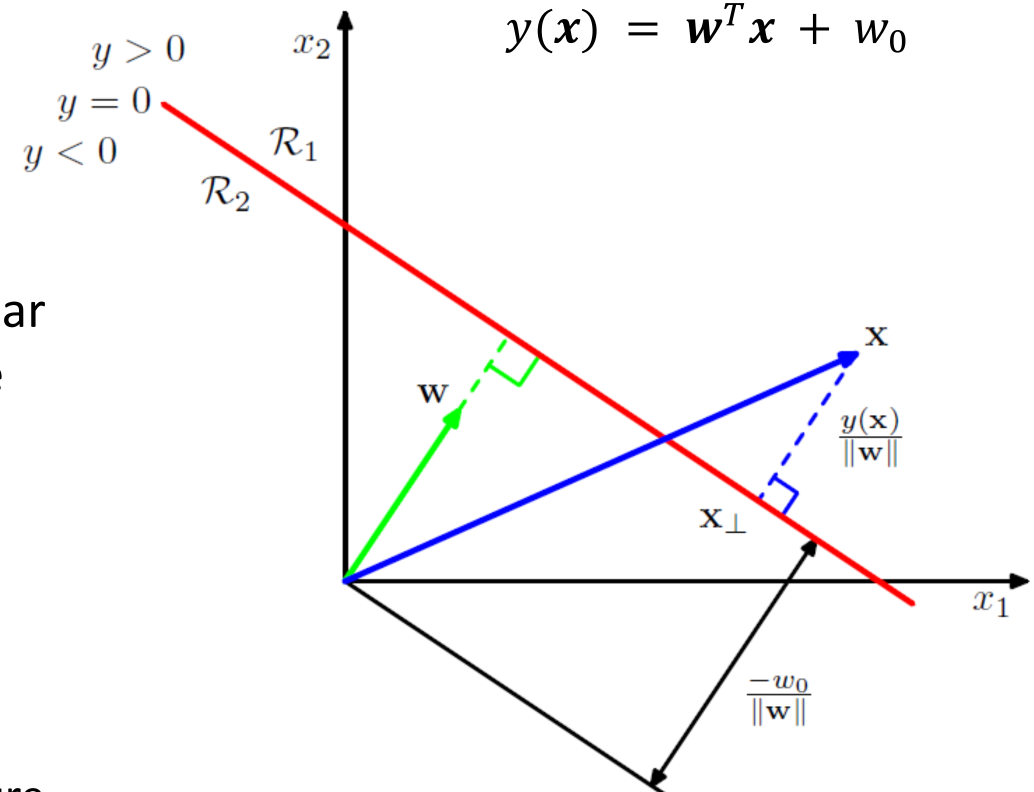


Illustration of the geometry of a linear discriminant function in 2D

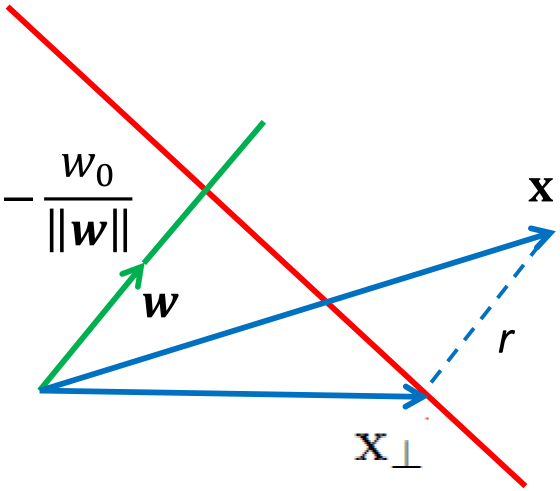
- The decision surface, is perpendicular to $\boldsymbol{w}$, and its displacement from the origin is controlled by the bias parameter w_0.
- the signed orthogonal distance of a general point $\boldsymbol{x}$ from the decision surface is given by $\frac{y(\boldsymbol{x})}{\|\boldsymbol{w}\|}$.
- The value of $y(\boldsymbol{x})$ gives a signed measure of the perpendicular distance r of the point $\boldsymbol{x}$ from the decision surface.

$\mathbf{x}$ is an arbitrary point:

$$\mathbf{x} = \mathbf{x}_\perp + r\frac{\mathbf{w}}{\|\mathbf{w}\|} \qquad \ldots (1)$$

$\mathbf{x}_\perp$ is the projection of **x** onto the decision surface

r is the perpendicular distance of the point **x** from the decision surface.



Multiplying both sides of (1) by $\mathbf{w}^T$ and adding w_0, and making use of $y(\boldsymbol{x}) = \mathbf{w}^T\mathbf{x} + w_0$, and $y(\mathbf{x}_\perp) = \mathbf{w}^{\mathrm{T}}\mathbf{x}_\perp + w_0 = 0$,

$$\mathrm{y}(\mathbf{x}) = \mathbf{w}^T\mathbf{x} + w_0 = r\frac{\mathbf{w}^T\mathbf{w}}{\|\mathbf{w}\|} = r\frac{\|\mathbf{w}\|^2}{\|\mathbf{w}\|} = r\|\mathbf{w}\|, \text{ thus}$$

$$r = \frac{y(\boldsymbol{x})}{\|\boldsymbol{w}\|}$$

Example

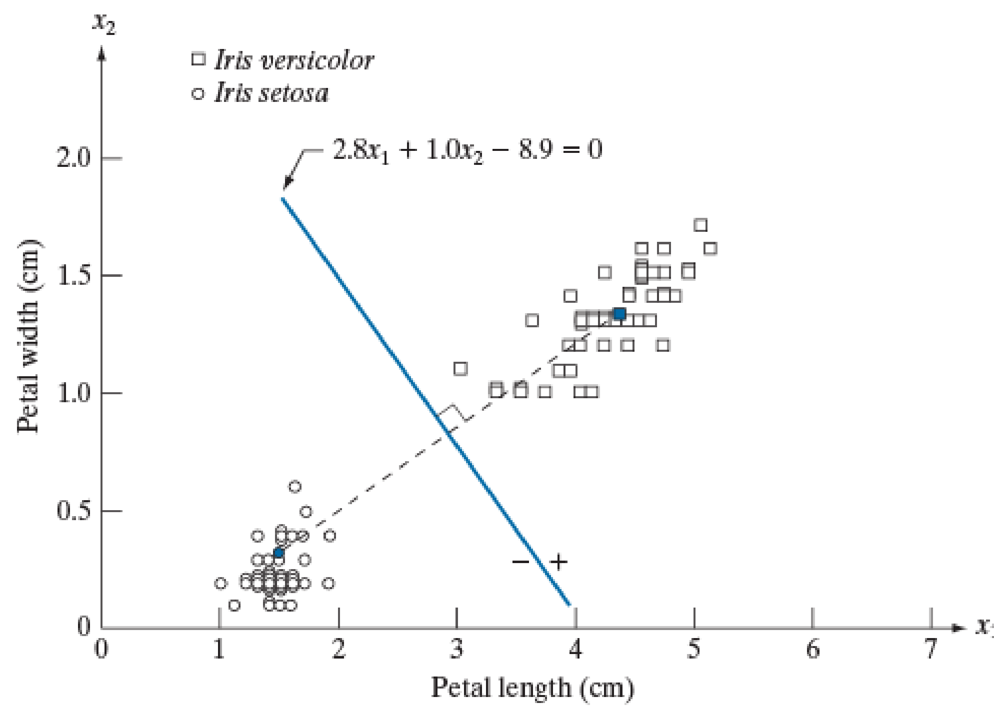
- Minimum Distance Classifier
  - Compute a distance-based measure between an unknown pattern vector and each of the class prototypes.
  - The prototype vectors are the mean vectors of the various pattern classes

$$\mathbf{m}_j = \frac{1}{N_j} \sum_{\mathbf{x} \in \omega_j} \mathbf{x}_j \qquad j = 1, 2, \ldots, W$$

  - Then assign the unknown pattern to the class of its closest prototype.

$$D_j(\mathbf{x}) = \|\mathbf{x} - \mathbf{m}_j\| \qquad j = 1, 2, \ldots, W$$

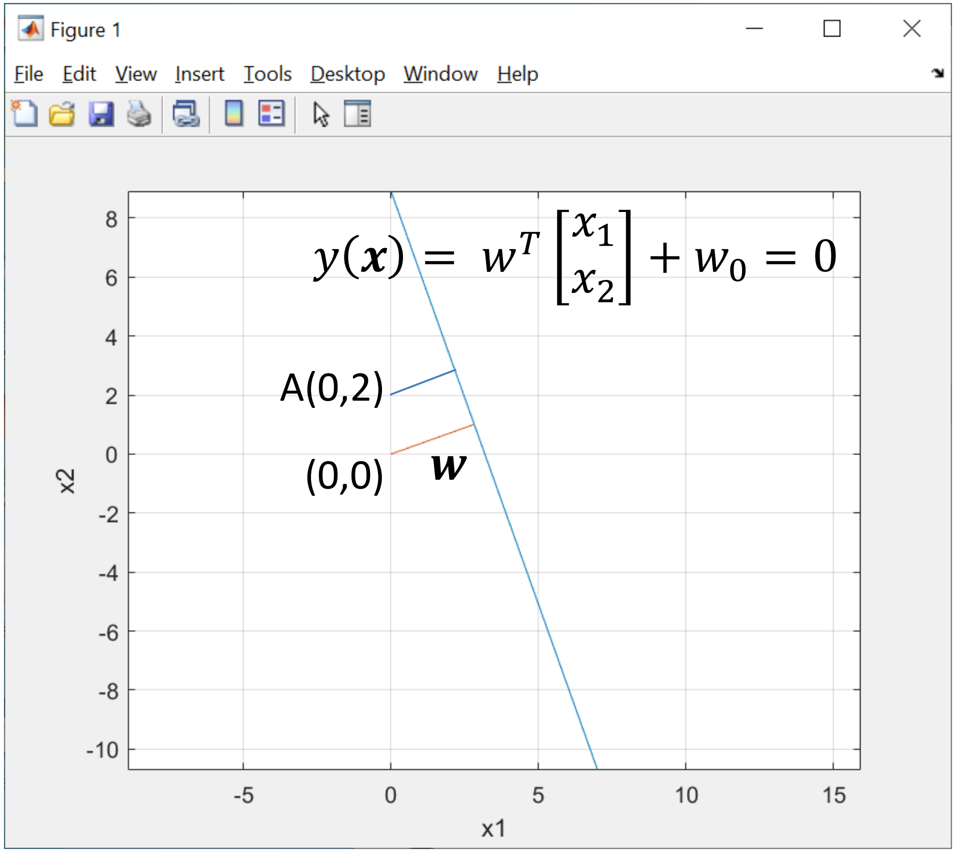
$$\|\mathbf{a}\| = (\mathbf{a}^T\mathbf{a})^{1/2}$$



Decision Boundary:

$$y(\boldsymbol{x}) = \boldsymbol{w}^T\boldsymbol{x} + w_0 = \begin{bmatrix} 2.8 \\ 1 \end{bmatrix}^T \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} - 8.9 = 0$$

```
>> w0 = -8.9;
>> x1 = 0: 0.001: 7;
>> x2 = - 2.8*x1 - w0;
>> plot(x1, x2); grid
>> xlabel('x1'); ylabel('x2')
>> w = [2.8; 1]    % The weight vector
>> hold on; plotv(w)
>> axis equal
```




```
% Shortest distance between the
origin and the decision line
>> dist = sqrt(x1.^2 + x2.^2);
>> min(dist)
ans =
   2.9934
```

```
>> -w0/norm(w)
ans =
   2.9934
```

$$-\frac{w_0}{\|\boldsymbol{w}\|}$$

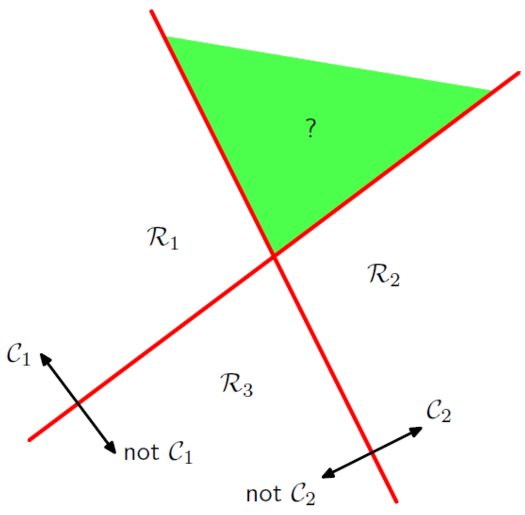
```
% Arbitrary chosen point A
> A = [0; 2];
>> distA = sqrt((x1-A(1)).^2 + (x2-A(2)).^2);
>> min(distA)
ans =
   2.3207
```

```
% Using the formula for the
% signed orthogonal distance
>> (dot(w,A) + w0)/norm(w)
ans =
  -2.3207
```

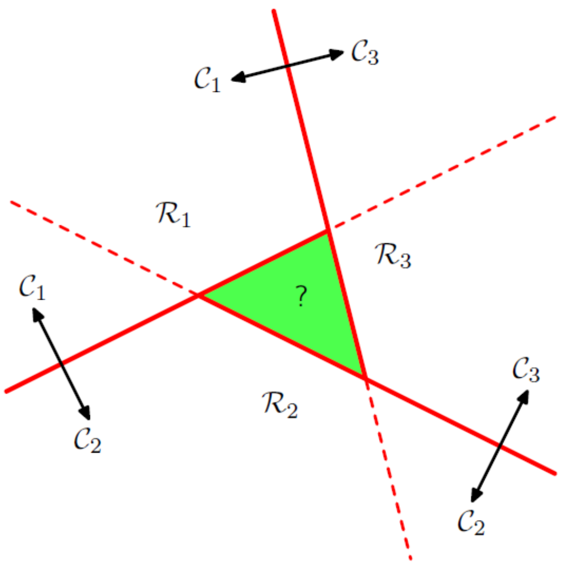
$$r = \frac{y(\boldsymbol{A})}{\|\boldsymbol{w}\|}$$

Multiple Classes

- We can extend the linear discriminants to more than two classes.
- We might be tempted be to build a K-class discriminant by combining a number of two-class discriminant functions. However, this leads to some difficulties (with ambiguous regions).



One-versus-the-rest classifier



One-versus-one classifier

Decision Boundaries

- We can avoid these difficulties by considering a single K-class discriminant comprising K linear functions of the form
$y_k(\boldsymbol{x}) = \boldsymbol{w}_k^T \boldsymbol{x} + w_{k0}$
- We then assign a point $\boldsymbol{x}$ to class C_k if $y_k(\boldsymbol{x}) > y_j(\boldsymbol{x})$ for all $j \neq k$.
- The decision boundary between class C_k and C_i is given by $y_k(\boldsymbol{x}) = y_j(\boldsymbol{x})$, which corresponds to a (D − 1)-dimensional hyperplane defined by
$(\boldsymbol{w}_k - \boldsymbol{w}_j)^T \boldsymbol{x} + (w_{k0} - w_{j0}) = 0.$
- The decision boundary has the same form as the decision boundary for the two-class case, and so analogous geometrical properties apply.

Fisher's Linear Discriminant

- Consider case of classifying two classes using a linear classification model:
  - We take the D-dimensional input vector $\mathbf{x}$ and project it down to one dimension using $y = \mathbf{w}^{\mathrm{T}}\mathbf{x}$.
  - If we place a threshold on y and classify $y \geq -w_0$ as class C_1, and otherwise class C_2.
  - This can be viewed as a dimensionality reduction method.
- In general, the projection onto one dimension leads to a considerable loss of information, and classes that are well separated in the original D-dimensional space may become strongly overlapping in one dimension.
- However, by adjusting the components of the weight vector $\mathbf{w}$, we can select a projection that maximizes the class separation, which is the idea of Fisher's Linear Discriminant method.

Two-Class Problem

- Consider a two-class problem in which there are N_1 points of class C_1 and N_2 points of class C_2, so that the mean vectors of the two classes are given by

$$\mathbf{m}_1 = \frac{1}{N_1} \sum_{n \in \mathcal{C}_1} \mathbf{x}_n, \qquad \mathbf{m}_2 = \frac{1}{N_2} \sum_{n \in \mathcal{C}_2} \mathbf{x}_n.$$

- The simplest measure of the separation of the classes, when projected onto $\mathbf{w}$, is the separation of the projected class means. Thus we might choose $\mathbf{w}$ so as to maximize

$$m_2 - m_1 = \mathbf{w}^{\mathrm{T}}(\mathbf{m}_2 - \mathbf{m}_1)$$

where m_k is the mean of the projected data from class C_k:

$$m_k = \mathbf{w}^{\mathrm{T}}\mathbf{m}_k$$

- It is possible that two classes, which are well separated in the original space, have considerable overlap when projected onto a the line joining their means.
- Fisher's idea is to maximize a function that will give a large separation between the projected class means, while also giving a small variance within each class, thereby minimizing the class overlap.
- The projection $y = \mathbf{w}^{\mathrm{T}}\mathbf{x}$ transforms the set of labeled data points in $\mathbf{x}$ into a labeled set in the one-dimensional space y. The within-class variance of the transformed data from class C_k is therefore given by

$$s_k^2 = \sum_{n \in \mathcal{C}_k} (y_n - m_k)^2$$

where $y_n = \mathbf{w}^{\mathrm{T}}\mathbf{x}_{\mathrm{n}}$

- We can define the total within-class variance for the whole data set to be s $s_1^2 + s_2^2$.

Fisher's Criterion

- The Fisher criterion is defined to be the ratio of the between-class variance to the within-class variance and is given by

$$J(\mathbf{w}) = \frac{(m_2 - m_1)^2}{s_1^2 + s_2^2}$$

- This Fisher criterion can be rewritten in matrix form as:

$$J(\mathbf{w}) = \frac{\mathbf{w}^{\mathrm{T}}\mathbf{S}_{\mathrm{B}}\mathbf{w}}{\mathbf{w}^{\mathrm{T}}\mathbf{S}_{\mathrm{W}}\mathbf{w}}$$

where $\mathbf{S}_{\mathrm{B}}$ is the *between-class* covariance matrix given by

$$\mathbf{S}_{\mathrm{B}} = (\mathbf{m}_2 - \mathbf{m}_1)(\mathbf{m}_2 - \mathbf{m}_1)^{\mathrm{T}}$$

and $\mathbf{S}_{\mathrm{w}}$ is the total *within-class* covariance matrix, given by

$$\mathbf{S}_{\mathrm{W}} = \sum_{n \in \mathcal{C}_1} (\mathbf{x}_n - \mathbf{m}_1)(\mathbf{x}_n - \mathbf{m}_1)^{\mathrm{T}} + \sum_{n \in \mathcal{C}_2} (\mathbf{x}_n - \mathbf{m}_2)(\mathbf{x}_n - \mathbf{m}_2)^{\mathrm{T}}.$$

Maximizing the Criterion

$$J(\mathbf{w}) = \frac{\mathbf{w}^{\mathrm{T}}\mathbf{S}_{\mathrm{B}}\mathbf{w}}{\mathbf{w}^{\mathrm{T}}\mathbf{S}_{\mathrm{W}}\mathbf{w}}$$

- Determine the value of $\mathbf{w}$ such that $J(\mathbf{w})$ is maximized, by differentiating $J(\mathbf{w})$ with respect to $\mathbf{w}$:

$$J'(\mathbf{w}) = \frac{\left(\mathbf{w}^{\mathrm{T}}\mathbf{S}_{\mathrm{B}}\mathbf{w}\right)'\left(\mathbf{w}^{\mathrm{T}}\mathbf{S}_{\mathrm{W}}\mathbf{w}\right) - \left(\mathbf{w}^{\mathrm{T}}\mathbf{S}_{\mathrm{B}}\mathbf{w}\right)\left(\mathbf{w}^{\mathrm{T}}\mathbf{S}_{\mathrm{W}}\mathbf{w}\right)'}{(\mathbf{w}^{\mathrm{T}}\mathbf{S}_{\mathrm{W}}\mathbf{w})^2}$$

$$= \frac{2\mathbf{S}_{\mathrm{B}}\mathbf{w}\left(\mathbf{w}^{\mathrm{T}}\mathbf{S}_{\mathrm{W}}\mathbf{w}\right) - 2\mathbf{S}_{\mathrm{W}}\mathbf{w}\left(\mathbf{w}^{\mathrm{T}}\mathbf{S}_{\mathrm{B}}\mathbf{w}\right)}{(\mathbf{w}^{\mathrm{T}}\mathbf{S}_{\mathrm{W}}\mathbf{w})^2} = 0$$

Thus

$$\mathbf{S}_{\mathrm{B}}\mathbf{w}\left(\mathbf{w}^{\mathrm{T}}\mathbf{S}_{\mathrm{W}}\mathbf{w}\right) = \mathbf{S}_{\mathrm{W}}\mathbf{w}\left(\mathbf{w}^{\mathrm{T}}\mathbf{S}_{\mathrm{B}}\mathbf{w}\right)$$

$$\frac{\mathbf{S}_{\mathbf{W}}^{-1}\mathbf{S}_{\mathrm{B}}\mathbf{w}\left(\mathbf{w}^{\mathrm{T}}\mathbf{S}_{\mathrm{W}}\mathbf{w}\right)}{(\mathbf{w}^{\mathrm{T}}\mathbf{S}_{\mathrm{B}}\mathbf{w})} = \mathbf{w}$$

$$\mathbf{w} = \frac{\mathbf{S}_{\mathrm{W}}^{-1}\mathbf{S}_{\mathrm{B}}\mathbf{w}(\mathbf{w}^{\mathrm{T}}\mathbf{S}_{\mathrm{W}}\mathbf{w})}{(\mathbf{w}^{\mathrm{T}}\mathbf{S}_{\mathrm{B}}\mathbf{w})}, \text{ where}$$

$$\begin{aligned}
\mathbf{S}_{\mathrm{B}}\mathbf{w} &= (\mathbf{m}_2 - \mathbf{m}_1)(\mathbf{m}_2 - \mathbf{m}_1)^{\mathrm{T}}\mathbf{w} \\
&= (\mathbf{m}_2 - \mathbf{m}_1)\left(\mathbf{w}^{\mathrm{T}}(\mathbf{m}_2 - \mathbf{m}_1)\right)^{\mathrm{T}} \\
&= (\mathbf{m}_2 - \mathbf{m}_1)(m_2 - m_1)
\end{aligned}$$

Since $\left(\mathbf{w}^{\mathrm{T}}\mathbf{S}_{\mathrm{W}}\mathbf{w}\right)$, $\left(\mathbf{w}^{\mathrm{T}}\mathbf{S}_{\mathrm{B}}\mathbf{w}\right)$ and $(m_2 - m_1)$ are all scalar factors, we can drop them if we care only about the direction of the weight vector $\mathbf{w}$, instead of its magnitude. Thus we can obtain

$$\mathbf{w} \propto \mathbf{S}_{\mathrm{W}}^{-1}(\mathbf{m}_2 - \mathbf{m}_1)$$

Choice of Direction for Projection

- The result: $\mathbf{w} \propto \mathbf{S}_{\mathbf{W}}^{-1}(\mathbf{m}_2 - \mathbf{m}_1)$ is known as *Fisher's linear discriminant.*
- If the within-class covariance is isotropic, so that $\boldsymbol{S}_{\mathbf{w}}$ is proportional to the unit matrix, then the optimal **w** is proportional to the difference of the class means.
- Although *Fisher's linear discriminant* is actually a specific **choice of direction for projection of the data down to one dimension**, the projected data can subsequently be used to construct a discriminant, by choosing a threshold y_0 so that we classify a new point as belonging to C_1 if $y(\boldsymbol{x}) \geq y_0$ and classify it as belonging to C_2 otherwise.
- For example, we can model the class-conditional densities $p(y|C_k)$ using Gaussian distributions. The justification for the Gaussian assumption comes from the Central Limit Theorem by noting that $y = \mathbf{w}^T\mathbf{x}$ is the sum of a set of random variables.
- Having found Gaussian approximations to the projected classes, we can determine the optimal threshold y_0, by using Bayes' rule and assigning each value y to the class having the higher posterior probability $p(C_K|y)$.

Fisher's Discriminant for Multiple Classes

- We generalize the Fisher discriminant to $K > 2$ classes, and we assume that the dimensionality D of the input space is greater than the number K of classes.
- Another generalization: instead of dimensionality reduction to 1-D, we introduce $D' > 1$ linear "features" $y_k = \mathbf{w}_k^T \mathbf{x}$, where $k = 1, \ldots, D'$. These feature values can be grouped together to form a vector $\boldsymbol{y}$.
- The weight vectors $\{\mathbf{w}_k\}$ can be considered to be the columns of a matrix $\mathbf{W}$, so that $\boldsymbol{y} = \mathbf{W}^T \mathbf{x}$.
- The generalization of the within-class covariance matrix to the case of K classes: $\mathbf{S}_{\mathrm{W}} = \sum_{k=1}^{K} \mathbf{S}_k$, where

$$\mathbf{S}_k = \sum_{n \in \mathcal{C}_k} (\mathbf{x}_n - \mathbf{m}_k)(\mathbf{x}_n - \mathbf{m}_k)^{\mathrm{T}}$$

$$\mathbf{m}_k = \frac{1}{N_k} \sum_{n \in \mathcal{C}_k} \mathbf{x}_n$$

and N_k is the number of samples in class C_k.

- In order to find a generalization of the between-class covariance matrix, we consider first the total covariance matrix

$$\mathbf{S}_{\mathrm{T}} = \sum_{n=1}^{N} (\mathbf{x}_n - \mathbf{m})(\mathbf{x}_n - \mathbf{m})^{\mathrm{T}}$$

where $\boldsymbol{m}$ is the mean of the total data set

$$\mathbf{m} = \frac{1}{N} \sum_{n=1}^{N} \mathbf{x}_n = \frac{1}{N} \sum_{k=1}^{K} N_k \mathbf{m}_k$$

and $N = \Sigma_k N_k$ is the total number of data points.

- The total covariance matrix can be decomposed into the sum of the within-class covariance matrix ($\boldsymbol{S}_{\mathrm{W}}$), plus an additional matrix $\boldsymbol{S}_{\mathrm{B}}$, which we identify as a measure of the between-class covariance:

$$\mathbf{S}_{\mathrm{T}} = \mathbf{S}_{\mathrm{W}} + \mathbf{S}_{\mathrm{B}}$$

$$\mathbf{S}_{\mathrm{B}} = \sum_{k=1}^{K} N_k (\mathbf{m}_k - \mathbf{m})(\mathbf{m}_k - \mathbf{m})^{\mathrm{T}}$$

- With covariance matrices having been defined in the original $\mathbf{x}$-space, we can now define similar matrices in the projected D'-dimensional $\mathbf{y}$-space:

$$\mathbf{s}_{\mathrm{W}} = \sum_{k=1}^{K} \sum_{n \in \mathcal{C}_k} (\mathbf{y}_n - \boldsymbol{\mu}_k)(\mathbf{y}_n - \boldsymbol{\mu}_k)^{\mathrm{T}}$$

$$\mathbf{s}_{\mathrm{B}} = \sum_{k=1}^{K} N_k (\boldsymbol{\mu}_k - \boldsymbol{\mu})(\boldsymbol{\mu}_k - \boldsymbol{\mu})^{\mathrm{T}}$$

$$\boldsymbol{\mu}_k = \frac{1}{N_k} \sum_{n \in \mathcal{C}_k} \mathbf{y}_n, \qquad \boldsymbol{\mu} = \frac{1}{N} \sum_{k=1}^{K} N_k \boldsymbol{\mu}_k.$$

```
clear all;
% 1D case (covariance become variance)
N = 100000;
X = randn(1, N);

% Split into two groups randomly
N1 = floor(N*rand);
N2 = N - N1;
X1 = X(1:N1);
X2 = X(N1+1: N);

m = mean(X);
m1 = mean(X1);
m2 = mean(X2);

ST = sum((X - m).^2);
SW = sum((X1 - m1).^2) + sum((X2 - m2).^2);
SB = N1*(m1-m)^2 + N2*(m2-m)^2;
% ST = SW + SB ?
abs(ST - (SW+SB))
```

```
% 2D case
m_model = [4, 0];
C_model = [9,4; 4,9];
Y = mvnrnd(m_model,C_model,N);

% Split Y into three groups randomly
N1 = floor(N*rand);
Diff = N - N1;
N2 = floor(Diff*rand);
N3 = N - (N1 + N2);

Y1 = Y(1:N1,:);
Y2 = Y(N1+1: N1+N2, :);
Y3 = Y(N1+N2+1: N, :);

my = mean(Y);
my1 = mean(Y1);
my2 = mean(Y2);
my3 = mean(Y3);

% Note the definition of cov() in Matlab,
need to multiply by (N-1)
STy = (N-1)*cov(Y);
SWy = (N1-1)*cov(Y1) + (N2-1)*cov(Y2) +
(N3-1)*cov(Y3);
SBy = N1*(my1-my)'*(my1-my) + N2*(my2-
my)'*(my2-my) + N3*(my3-my)'*(my3-my);

abs(STy - (SWy + SBy))
```

Choice of Projection Matrix

- We want to construct a scalar that is large when the between-class covariance is large and when the within-class covariance is small.

- One possible choice of criterion is $J(\mathbf{W}) = \mathrm{Tr}\left\{\mathrm{s}_{\mathrm{W}}^{-1}\mathrm{s}_{\mathrm{B}}\right\}$

- This criterion can then be rewritten as an explicit function of the projection matrix $\mathbf{W}$ in the form:

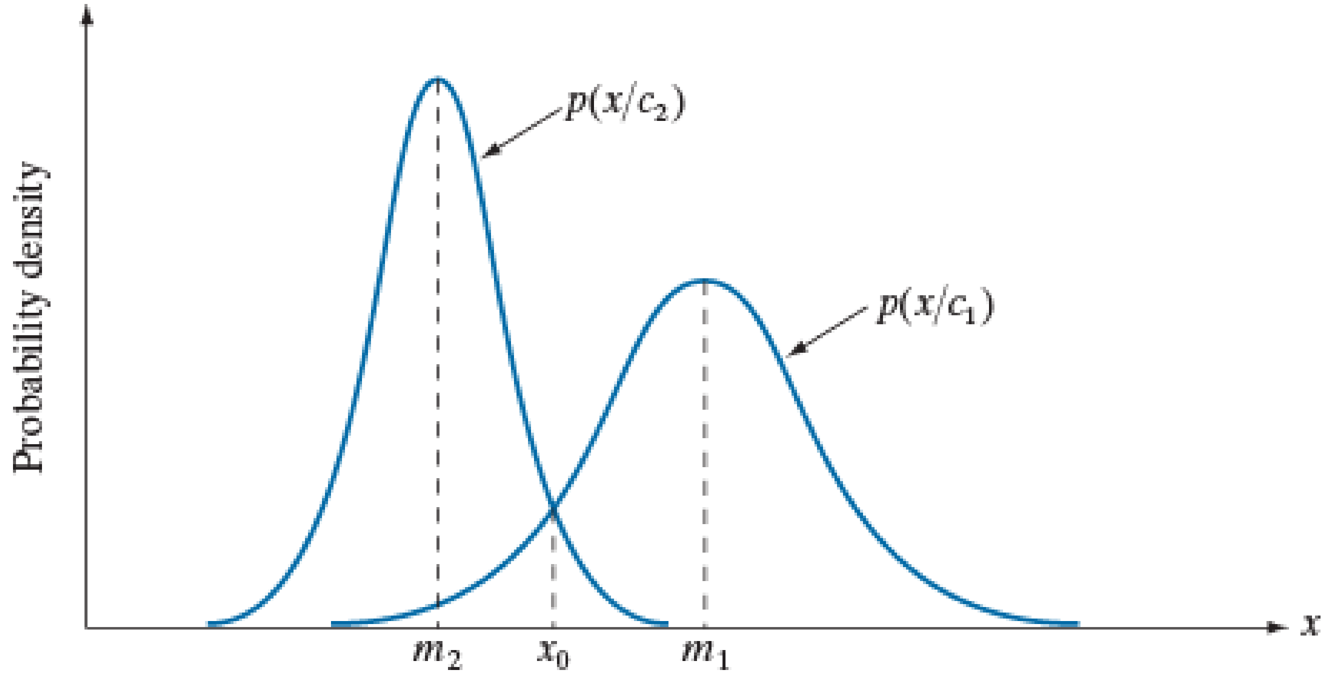
$$J(\mathbf{w}) = \mathrm{Tr}\left\{(\mathbf{W}\mathbf{S}_{\mathrm{W}}\mathbf{W}^{\mathrm{T}})^{-1}(\mathbf{W}\mathbf{S}_{\mathrm{B}}\mathbf{W}^{\mathrm{T}})\right\}$$

- It can be shown that the weight values are determined by those eigenvectors of $\mathbf{S}_{\mathrm{W}}^{-1}\mathbf{S}_{\mathrm{B}}$, which correspond to the D' largest eigenvalues.
- It can be shown $\mathbf{S}_{\mathrm{B}}$ has rank at most equal to $(K-1)$ and so there are at most $(K-1)$ nonzero eigenvalues. So we are therefore unable to find more than $(K-1)$ linear "features".

Relation to LDA

- Linear discriminant analysis (LDA), normal discriminant analysis (NDA), or discriminant function analysis is a generalization of Fisher's linear discriminant.
- LDA is to find a linear combination of features that characterizes or separates two or more classes of objects or events.
- The resulting combination may be used as a linear classifier, or, more commonly, for dimensionality reduction before subsequent classification.
- In general, discriminant analysis assumes that the class-conditional densities to have multivariate Gaussian distributions.
  - For linear discriminant analysis (LDA), the model assumes the same covariance matrix for each class -- only the means vary.
  - For quadratic discriminant analysis (QDA), the model considers varying mean vectors and covariance matrices of each class.

Gaussian Pattern Classes



Decision Function:

$$d_j(x) = p(x|c_j)P(c_j) = \frac{1}{\sqrt{2\pi}\sigma_j} e^{-\frac{(x-m_j)^2}{2\sigma_j^2}} P(c_j)$$

where $\quad j = 1, 2$

n-Dimensional Gaussian PDF

$$p(\mathbf{x}/\omega_j) = \frac{1}{(2\pi)^{n/2}|\mathbf{C}_j|^{1/2}} e^{-\frac{1}{2}(\mathbf{x}-\mathbf{m}_j)^T \mathbf{C}_j^{-1}(\mathbf{x}-\mathbf{m}_j)}$$

where the mean vector is $\mathbf{m}_j = E_j\{\mathbf{x}\}$

and the covariance matrix is

$$\mathbf{C}_j = E_j\{(\mathbf{x} - \mathbf{m}_j)(\mathbf{x} - \mathbf{m}_j)^T\}$$

We can approximate with taking the averages of sample vectors:

$$\mathbf{m}_j = \frac{1}{N_j} \sum_{\mathbf{x} \in \omega_j} \mathbf{x} \qquad \mathbf{C}_j = \frac{1}{N_j} \sum_{\mathbf{x} \in \omega_j} \mathbf{x}\mathbf{x}^T - \mathbf{m}_j \mathbf{m}_j^T$$

Logarithm of the Decision Function

$$d_j(\mathbf{x}) = \ln\left[p(\mathbf{x}/\omega_j)P(\omega_j)\right] = \ln p(\mathbf{x}/\omega_j) + \ln P(\omega_j)$$

$$p(\mathbf{x}/\omega_j) = \frac{1}{(2\pi)^{n/2}|\mathbf{C}_j|^{1/2}} e^{-\frac{1}{2}(\mathbf{x}-\mathbf{m}_j)^T\mathbf{C}_j^{-1}(\mathbf{x}-\mathbf{m}_j)}$$

$$d_j(\mathbf{x}) = \ln P(\omega_j) - \frac{n}{2}\ln 2\pi - \frac{1}{2}\ln|\mathbf{C}_j| - \frac{1}{2}\left[(\mathbf{x} - \mathbf{m}_j)^T\mathbf{C}_j^{-1}(\mathbf{x} - \mathbf{m}_j)\right]$$

$$d_j(\mathbf{x}) = \ln P(\omega_j) - \frac{1}{2}\ln|\mathbf{C}_j| - \frac{1}{2}\left[(\mathbf{x} - \mathbf{m}_j)^T\mathbf{C}_j^{-1}(\mathbf{x} - \mathbf{m}_j)\right]$$

LDA

- Two assumptions of linear discriminant analysis (LDA):
  - Multivariate normality
  - Homoscedasticity: Equal covariance for all classes
- Estimation of the covariance matrix in actual implementations:

Matlab

Predictor Covariance Treatment
• All classes have the same covariance matrix. • $\widehat{\Sigma}_\gamma = (1-\gamma)\widehat{\Sigma} + \gamma \mathrm{diag}(\widehat{\Sigma}).$ $\widehat{\Sigma}$ is the empirical, pooled covariance matrix and γ is the amount of regularization.

Sklearn

- Shrinkage is a form of regularization used to improve the estimation of covariance matrices.
- The ‘shrinkage’ parameter can be set to ‘auto’. This automatically determines the optimal shrinkage parameter in an analytic way.
- The shrinkage parameter can also be manually set between 0 and 1.
  - 0 corresponds to no shrinkage, which means the empirical covariance matrix will be used.
  - 1 corresponds to complete shrinkage, which means that the diagonal matrix of variances will be used as an estimate for the covariance matrix.

Two Classes As an Example

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

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

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

Decision Boundary (assuming equal class probabilities): $d_1(\mathbf{x}) = d_2(\mathbf{x})$

$$(\mathbf{x} - \mathbf{m_1})^{\mathrm{T}}\mathbf{C}^{-1}(\mathbf{x} - \mathbf{m_1}) = (\mathbf{x} - \mathbf{m_2})^{\mathrm{T}}\mathbf{C}^{-1}(\mathbf{x} - \mathbf{m_2})$$

$$\begin{aligned} &\mathbf{x}^{\mathrm{T}}\mathbf{C}^{-1}\mathbf{x} - \mathbf{x}^{\mathrm{T}}\mathbf{C}^{-1}\mathbf{m_1} - \mathbf{m_1^{\mathrm{T}}}\mathbf{C}^{-1}\mathbf{x} + \mathbf{m_1^{\mathrm{T}}}\mathbf{C}^{-1}\mathbf{m_1} \\ &= \mathbf{x}^{\mathrm{T}}\mathbf{C}^{-1}\mathbf{x} - \mathbf{x}^{\mathrm{T}}\mathbf{C}^{-1}\mathbf{m_2} - \mathbf{m_2^{\mathrm{T}}}\mathbf{C}^{-1}\mathbf{x} + \mathbf{m_2^{\mathrm{T}}}\mathbf{C}^{-1}\mathbf{m_2} \end{aligned}$$

Cancellation
due to the assumption of same covariance (LDA); otherwise quadratic function of $\mathbf{x}$, thus QDA results.

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

Decision Boundary is a Line

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

Let weight vector $\mathbf{w} = \mathbf{C}^{-1}(\mathbf{m_1} - \mathbf{m_2})$, then

$$(\mathbf{m_1} - \mathbf{m_2})^{\mathrm{T}}\mathbf{C}^{-1}\mathbf{x} = \mathbf{w}^{\mathrm{T}}\mathbf{x} \quad \text{and}$$

$$\mathbf{w}^{\mathrm{T}}(\mathbf{m_1} + \mathbf{m_2}) = (\mathbf{m_1} - \mathbf{m_2})^{\mathrm{T}}\mathbf{C}^{-1}(\mathbf{m_1} + \mathbf{m_2})$$

$$= \mathbf{m_1^{\mathrm{T}}}\mathbf{C}^{-1}\mathbf{m_1} + \mathbf{m_1^{\mathrm{T}}}\mathbf{C}^{-1}\mathbf{m_2} - \mathbf{m_2^{\mathrm{T}}}\mathbf{C}^{-1}\mathbf{m_1} - \mathbf{m_2^{\mathrm{T}}}\mathbf{C}^{-1}\mathbf{m_2}$$

$$= \mathbf{m_1^{\mathrm{T}}}\,\mathbf{C}^{-1}\mathbf{m_1} - \mathbf{m_2^{\mathrm{T}}}\mathbf{C}^{-1}\mathbf{m_2}$$

Thus the decision function is a function of a linear combination of the observations:

$$\mathbf{w}^{\mathbf{T}}\mathbf{x} = \frac{1}{2}\mathbf{w}^{\mathrm{T}}(\mathbf{m_1} + \mathbf{m_2}), \text{ where } \mathbf{w} = \mathbf{C}^{-1}(\mathbf{m_1} - \mathbf{m_2})$$

Consistent with Fisher's linear discriminant with projection:

$\mathbf{w} \propto \mathbf{S_W^{-1}}(\mathbf{m}_2 - \mathbf{m}_1)$, where $\mathbf{S_W} = 2\boldsymbol{C}$

Example

$$\mathbf{x} = \begin{bmatrix} x_1 \\ x_2 \end{bmatrix}, \mathbf{m_1} = \begin{bmatrix} 3 \\ 3 \end{bmatrix}, \mathbf{m_2} = \begin{bmatrix} 9 \\ 9 \end{bmatrix},$$

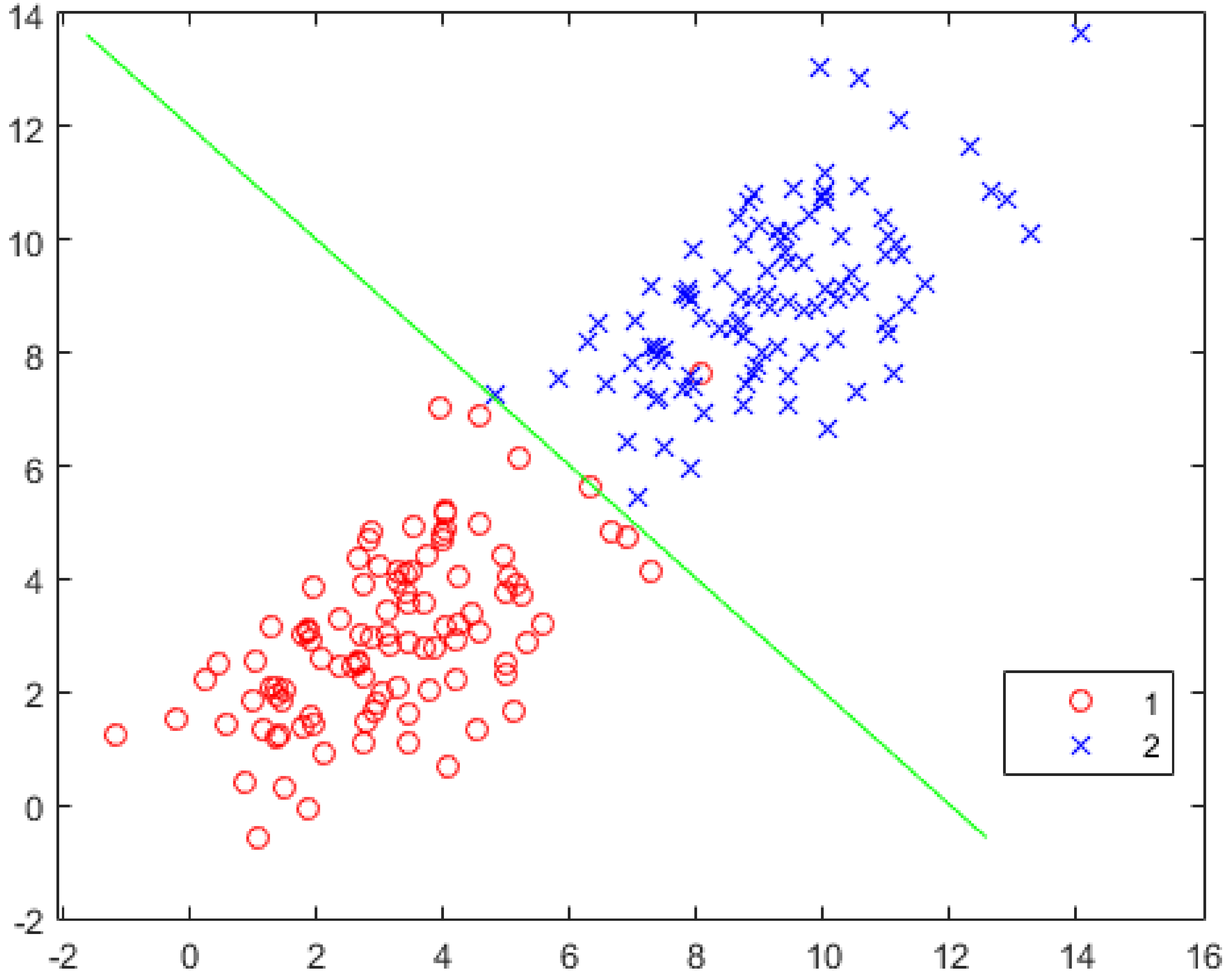
$$\boldsymbol{C}_1 = \boldsymbol{C}_2 = \boldsymbol{C} = \begin{bmatrix} 2 & 1 \\ 1 & 2 \end{bmatrix} \qquad \boldsymbol{C}^{-1} = \tfrac{1}{3}\begin{bmatrix} 2 & -1 \\ -1 & 2 \end{bmatrix}$$

$$\mathbf{w} = \mathbf{C}^{-1}(\mathbf{m_1} - \mathbf{m_2}) = \tfrac{1}{3}\begin{bmatrix} 2 & -1 \\ -1 & 2 \end{bmatrix}\begin{bmatrix} -6 \\ -6 \end{bmatrix} = \begin{bmatrix} -2 \\ -2 \end{bmatrix}$$

$$\frac{1}{2}\mathbf{w}^{\mathrm{T}}(\mathbf{m_1} + \mathbf{m_2}) = \frac{1}{2}\begin{bmatrix} -2 & -2 \end{bmatrix}\begin{bmatrix} 12 \\ 12 \end{bmatrix} = -24$$

The decision boundary is $\mathbf{w}^{\mathrm{T}}\mathbf{x} = \frac{1}{2}\mathbf{w}^{\mathrm{T}}(\mathbf{m_1} + \mathbf{m_2})$

$$x_1 + x_2 = 12$$


14
12
10
8
6
4
2
0
-2
-2
0
2
4
6
8
10
12
14
16
1
2

Varying Covariance Matrix

Decision Boundary (assuming equal class probabilities): $d_1(\mathbf{x}) = d_2(\mathbf{x})$

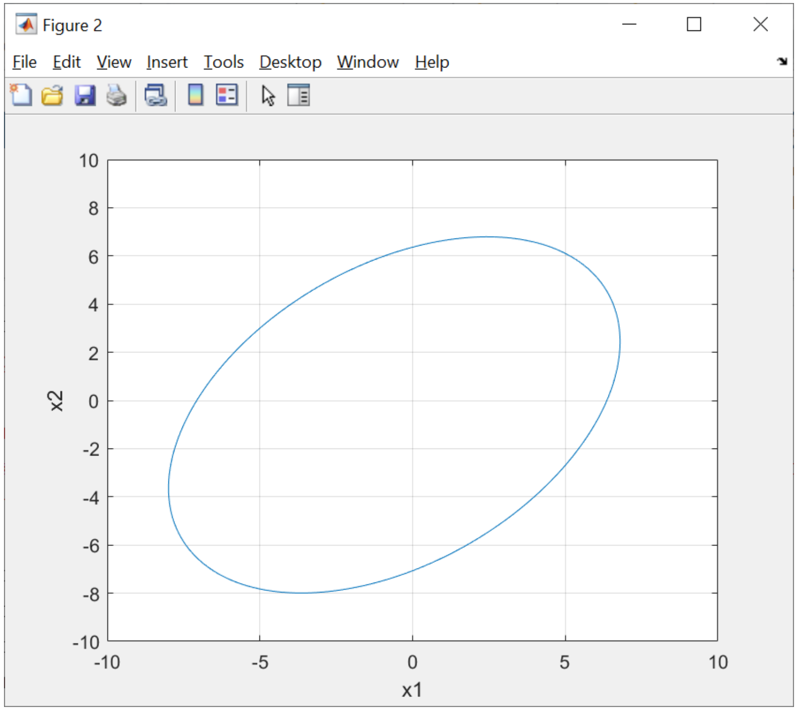
$$\ln|\mathbf{C}_1| + (\mathbf{x}-\mathbf{m_1})^{\mathrm{T}}\mathbf{C}_1^{-1}(\mathbf{x}-\mathbf{m_1}) - \ln|\mathbf{C}_2| + (\mathbf{x}-\mathbf{m_2})^{\mathrm{T}}\mathbf{C}_2^{-1}(\mathbf{x}-\mathbf{m_2}) = 0$$

Example:

$$\mathbf{x} = \begin{bmatrix} x_1 \\ x_2 \end{bmatrix}, \mathbf{m_1} = \begin{bmatrix} 3 \\ 3 \end{bmatrix}, \mathbf{m_2} = \begin{bmatrix} 9 \\ 9 \end{bmatrix}, \mathbf{C}_1 = \begin{bmatrix} 2 & 1 \\ 1 & 2 \end{bmatrix}, \mathbf{C}_2 = \begin{bmatrix} 5 & 3 \\ 3 & 5 \end{bmatrix}$$

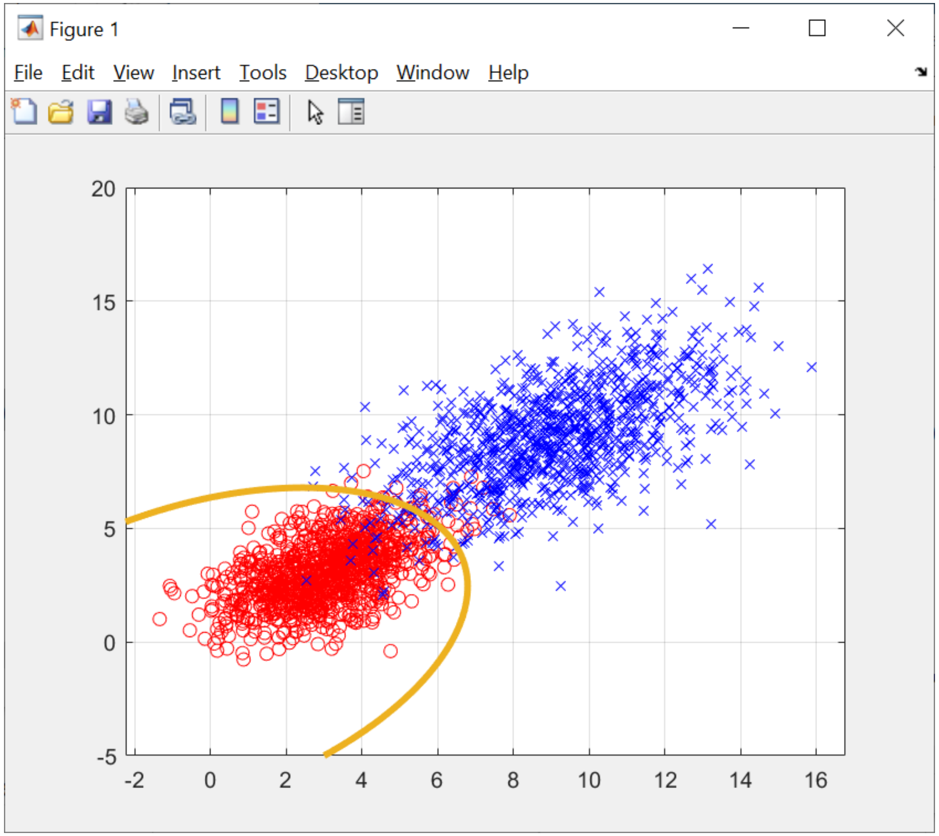
g =
(17*x1^2)/48 - (7*x1*x2)/24 + x1/4 +
(17*x2^2)/48 + x2/4 – 15.92

```
% Symbolic math
syms f x x1 x2
x = [x1; x2];
c1 = inv(cov1);
c2 = inv(cov2);
f = log(abs(det(cov1))) + (x-
m1).'*c1*(x-m1) - log(abs(det(cov2))) -
(x-m2).'*c2*(x-m2);
g = simplify(f)
fimplicit(g,[-10 10]); grid
```



QDA

```
N = 1000;

% Class 1
m1 = [3, 3]';    % Mean vector
cov1 = [2 1; 1 2]; % Covariance matrix
r1 = mvnrnd(m1,cov1,N);

data_C1 = zeros(N, 2);
data_C1 = r1;
label_C1 = ones(N, 1);

% Generate data entries for Class 2
m2 = [9, 9]';
cov2 = [5,3; 3,5];
r2 = mvnrnd(m2,cov2,N);

data_C2 = zeros(N, 2);
data_C2 = r2;
label_C2 = 2*ones(N, 1);
```

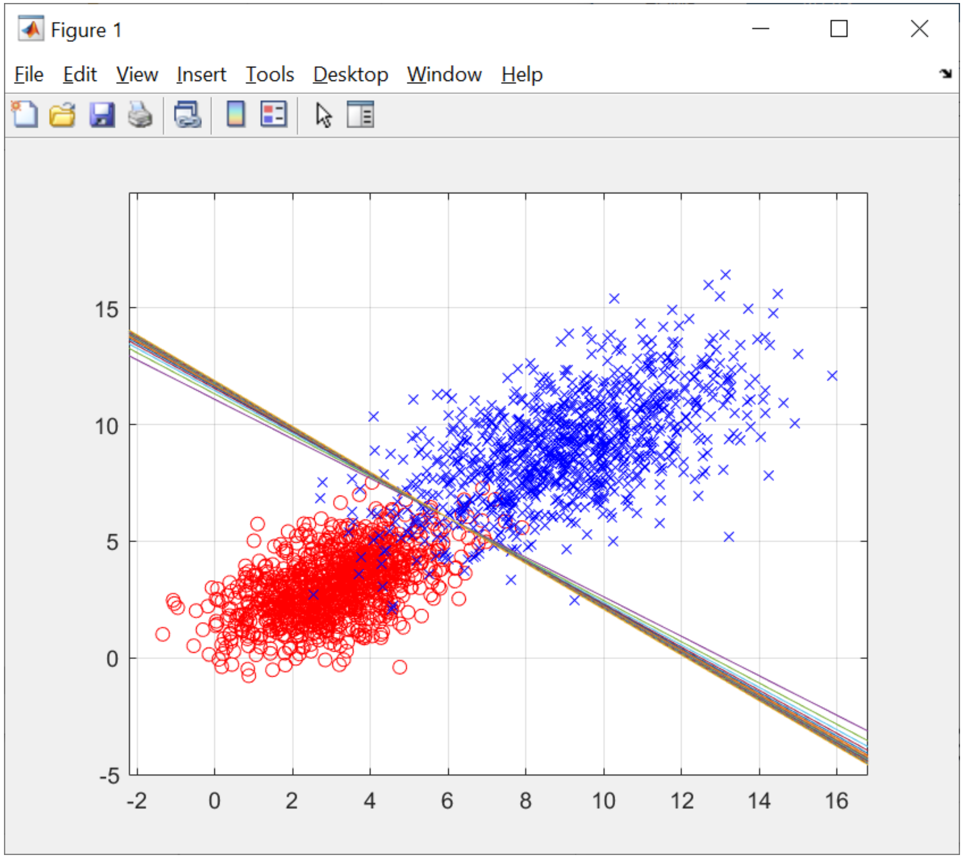



```
% Combine data of two classes
data = vertcat (data_C1, data_C2);
label = vertcat (label_C1, label_C2);

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

```
hold on;
fimplicit(g)
```

What if we still use LDA?

```
% Use regularization to shrink the
cov_all of all data
cov_all = cov(data);
% Too large -- need to shrink
% Scan through the whole range of
% values of the shrinkage parameter
for gamma = 0: 0.05: 1;
    diag = cov_all;
    diag(1,2) = 0;
    diag(2,1) = 0;
    cov_est = (1-gamma)*cov_all +
gamma*diag;
    w1 = inv(cov_est)*(m1-m2);
    f1 = w1.'*x - 0.5*w1.'*(m1+m2);

    hold on;
    fimplicit(f1)
end
```



Fit discriminant analysis classifier fitcdiscr ( )

```
>> Model = fitcdiscr(data, label);
Model.DiscrimType
ans =
   'linear'
>> K = Model.Coeffs(1,2).Const
K =
  23.5780
>> L = Model.Coeffs(1,2).Linear
L =
  -1.8358
  -2.1190
```

```
>> Model.Gamma
ans =
    0

>> K = Model.Coeffs(2,1).Const
K =
  -23.5780
>> K = Model.Coeffs(2,1).Linear
K =
   1.8358
   2.1190
```

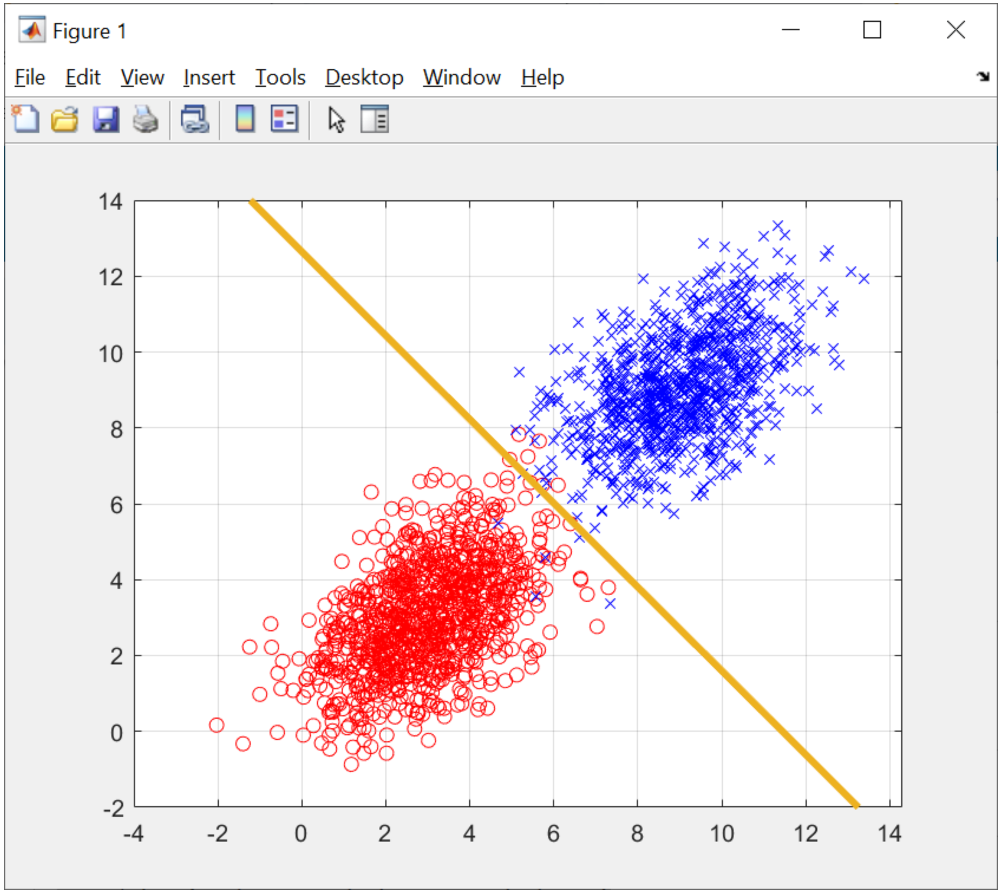
$$\mathbf{w} = \mathbf{C}^{-1}(\mathbf{m_1} - \mathbf{m_2}) = \frac{1}{3}\begin{bmatrix} 2 & -1 \\ -1 & 2 \end{bmatrix}\begin{bmatrix} -6 \\ -6 \end{bmatrix} = \begin{bmatrix} -2 \\ -2 \end{bmatrix}$$

$$\frac{1}{2}\mathbf{w}^{\mathrm{T}}(\mathbf{m_1} + \mathbf{m_2}) = \frac{1}{2}\begin{bmatrix} -2 & -2 \end{bmatrix}\begin{bmatrix} 12 \\ 12 \end{bmatrix} = -24$$

Classification Performance

```
>> resubLoss(Model)
ans =
   0.0065
>> L = predict(Model, data);
>> diff = abs(label - L);
>> length(find(diff~=0))/length(data)
ans =
   0.0065
```

```
f = @(x1,x2) K + L(1)*x1 + L(2)*x2;

gscatter(data(:,1),data(:,2),label,'rb','ox')
grid
hold on;
fimplicit(f);
```



$$K + \begin{bmatrix} x_1 & x_2 \end{bmatrix} L = 0.$$

K = Model.Coeffs(1,2).Const

L = Model.Coeffs(1,2).Linear

Varying Covariance Matrices

```
% Class 1
m1 = [3, 3]';   % Mean vector
cov1 = [2 1; 1 2]; % Covariance matrix
```

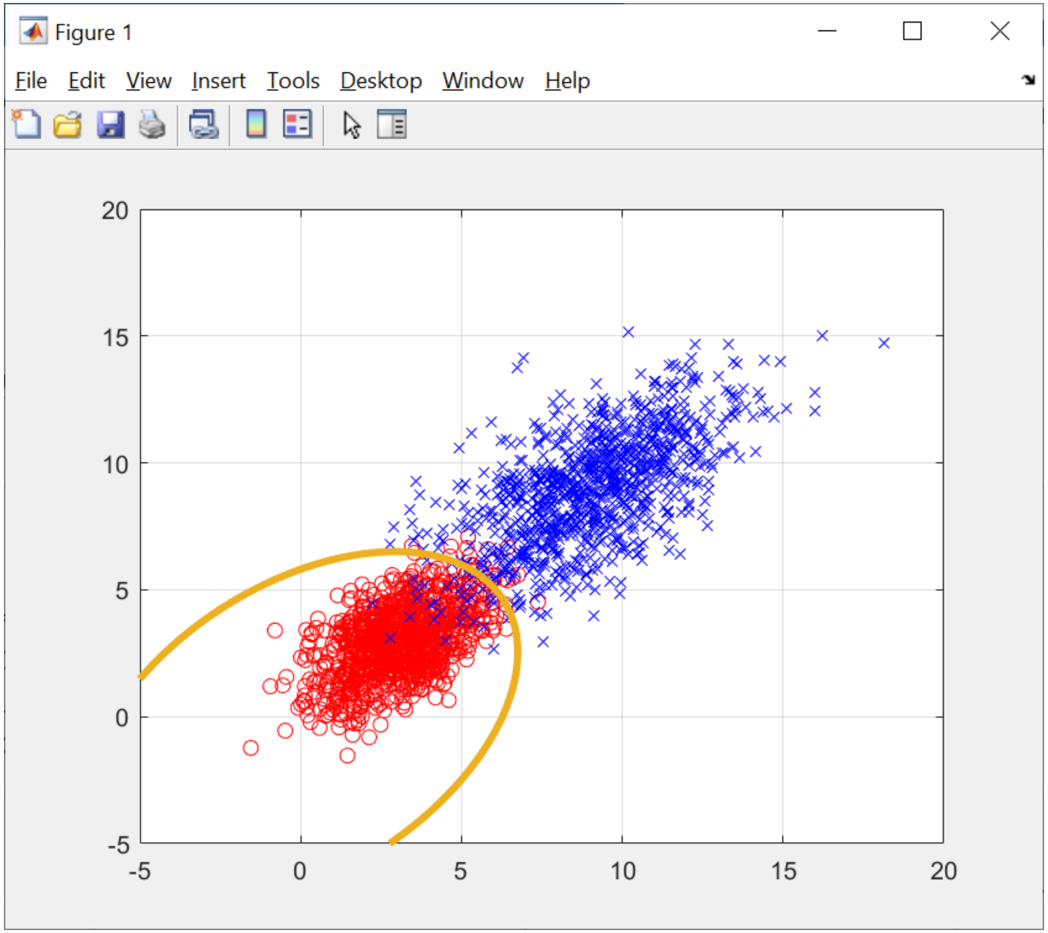
```
% Class 2
m2 = [9, 9]';
cov2 = [5,3; 3,5];
```

```
>> Model = fitcdiscr(data, label);
Model.DiscrimType
ans =
   'linear‘
K = Model.Coeffs(1,2).Const
K =
  13.3013
>> L = Model.Coeffs(1,2).Linear
L =
  -1.0374
  -1.1830

>> resubLoss(Model)
ans =
   0.0380
```

```
>> Model_QDA = fitcdiscr(data, label, 'DiscrimType', 'quadratic');
>> Model_QDA.DiscrimType
ans =
   'quadratic‘

>> resubLoss(Model_QDA)
ans =
   0.0280
```

QDA

```
>> K = Model_QDA.Coeffs(1,2).Const;
L = Model_QDA.Coeffs(1,2).Linear;
Q = Model_QDA.Coeffs(1,2).Quadratic;

K =
   7.7810
L =
   0.0890
  -0.1940
Q =
  -0.2119    0.0886
   0.0886   -0.1971

f = @(x1,x2) K + L(1)*x1 + L(2)*x2 + Q(1,1)*x1.^2 + ...
   (Q(1,2)+Q(2,1))*x1.*x2 + Q(2,2)*x2.^2;

gscatter(data(:,1),data(:,2),label,'rb','ox')
grid
hold on;
fimplicit(f);
```



$$K + \begin{bmatrix} x_1 & x_2 \end{bmatrix} L + \begin{bmatrix} x_1 & x_2 \end{bmatrix} Q \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} = 0.$$

‘lda_demo.py’

```
import numpy as np
infile = r"C:\...\lda_data.csv"
dataset = np.loadtxt(infile, delimiter=',')

X = dataset[:, 0:2]
y = dataset[:,2]  # labels

from sklearn.discriminant_analysis import LinearDiscriminantAnalysis as LDA

clf = LDA()
clf.fit(X, y)
clf.intercept_
clf.coef_
clf.score(X,y)

y_pred = clf.predict(X)
num_errors = np.sum(y != y_pred)
num_errors/np.size(y)
```

Summary

- Prototype Matching (minimum-distance classifier)
  - Assign the unknown pattern to the class of its closest prototype (mean vectors of various classes)
- Bayes Classifier (if multivariate normality is assumed)
  - Becomes the minimum-distance classifier
    - If all covariance matrices are equal to identity matrix.
    - All classes are equally likely.
  - Becomes the LDA
    - If all covariance matrices are assumed to be the same.
  - Becomes the QDA
    - If there are varying covariance matrices for different classes
- LDA can be viewed as a minimum-distance classifier, with the distance being the Mahalanobis distance (between a point $\mathbf{x}$ and the sample mean of a distribution), instead of the Euclidean distance.

$$(\mathbf{x}-\mathbf{m_1})^{\mathrm{T}}\mathbf{C}^{-1}(\mathbf{x}-\mathbf{m_1}) = (\mathbf{x}-\mathbf{m_2})^{\mathrm{T}}\mathbf{C}^{-1}(\mathbf{x}-\mathbf{m_2})$$