Discriminant Analysis

$$Y \sim \text{Multinom}(p_1, \ldots, p_K)$$
$$X \mid Y=k \sim f_k$$

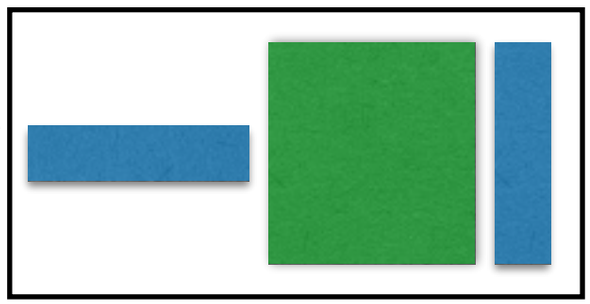
For prediction, compute the decision function
$d_k(x) = \log(p_k) + \log(f_k(x))$ - Constant

QDA : N(mu_k, Sigma_k)
LDA : N(mu_k, Sigma)
NB : Independent pdf over p-dim of x

QDA

Given $Y = k$, let's model the p-dim feature $\mathbf{x}$ by a multivariate normal distribution with mean $\boldsymbol{\mu}_k$ and covariance matrix Σ_k.

$$\boldsymbol{\mu}_k = \begin{pmatrix} \mu_{k,1} \\ \mu_{k,2} \\ \vdots \\ \mu_{k,p} \end{pmatrix}_{p\times 1} \quad , \quad \Sigma_k^{-1} = \begin{pmatrix} \theta_{k,11} & \cdots & \theta_{k,1p} \\ & & \\ \theta_{k,p1} & \cdots & \theta_{k,pp} \end{pmatrix}_{p\times p}$$



Its pdf is given by

$$f_k(\mathbf{x}) = \frac{1}{(\sqrt{2\pi})^p} \frac{1}{|\Sigma_k|^{1/2}} \exp\Big[-\frac{1}{2}(\mathbf{x}-\boldsymbol{\mu}_k)^t \Sigma_k^{-1} (\mathbf{x}-\boldsymbol{\mu}_k)\Big]$$

$$(\mathbf{x}-\boldsymbol{\mu}_k)^t \Sigma_k^{-1} (\mathbf{x}-\boldsymbol{\mu}_k) = \sum_{j=1}^{p}\sum_{l=1}^{p} \theta_{k,jl}(x_j-\mu_{k,j})(x_l-\mu_{k,l})$$

LDA

- If further assume $\Sigma_k = \Sigma$, all we need to compute is

$$d_k(\mathbf{x}) = (\mathbf{x} - \boldsymbol{\mu}_k)^t \Sigma^{-1} (\mathbf{x} - \boldsymbol{\mu}_k) + \log |\Sigma| - 2 \log \pi_k \qquad (1)$$

  which is a linear function of $\mathbf{x}$:

$$\begin{aligned} & (\mathbf{x} - \boldsymbol{\mu}_k)^t \Sigma^{-1} (\mathbf{x} - \boldsymbol{\mu}_k) \\ = & \quad \mathbf{x}^t \Sigma^{-1} \mathbf{x} - 2\mathbf{x}^t \Sigma^{-1} \boldsymbol{\mu}_k + \boldsymbol{\mu}_k^T \Sigma^{-1} \boldsymbol{\mu}_k \end{aligned}$$

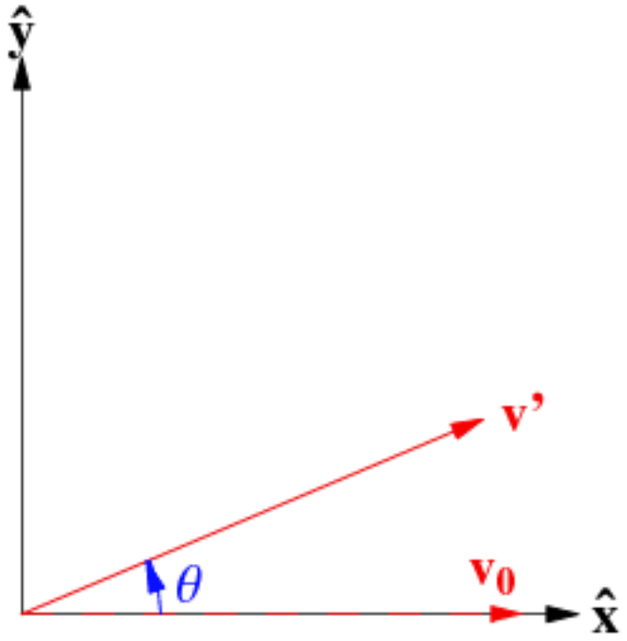
- Estimate Σ by the pooled sample covariance matrix:

$$\hat{\Sigma} = \frac{1}{n-K} \sum_{k=1}^{K} \sum_{i:y_i=k} (\mathbf{x}_i - \hat{\boldsymbol{\mu}}_k)(\mathbf{x}_i - \hat{\boldsymbol{\mu}}_k)^t.$$

What if $\hat{\Sigma}^{-1}$ does not exist? Replace it by $(\hat{\Sigma} + \eta \mathbf{I}_p)^{-1}$ where η is a small number, or compute $\hat{\Sigma}^{-1}$ as follows (assume $p = 3$)

$$\hat{\Sigma} = U_{p\times 3} \begin{pmatrix} d_1 & 0 & 0 \\ 0 & d_2 & 0 \\ 0 & 0 & 0 \end{pmatrix} U^t, \quad \hat{\Sigma}^{-1} = U \begin{pmatrix} 1/d_1 & 0 & 0 \\ 0 & 1/d_2 & 0 \\ 0 & 0 & 0 \end{pmatrix} U^t.$$

3rd-dim (after rotation) is the null space for all classes.



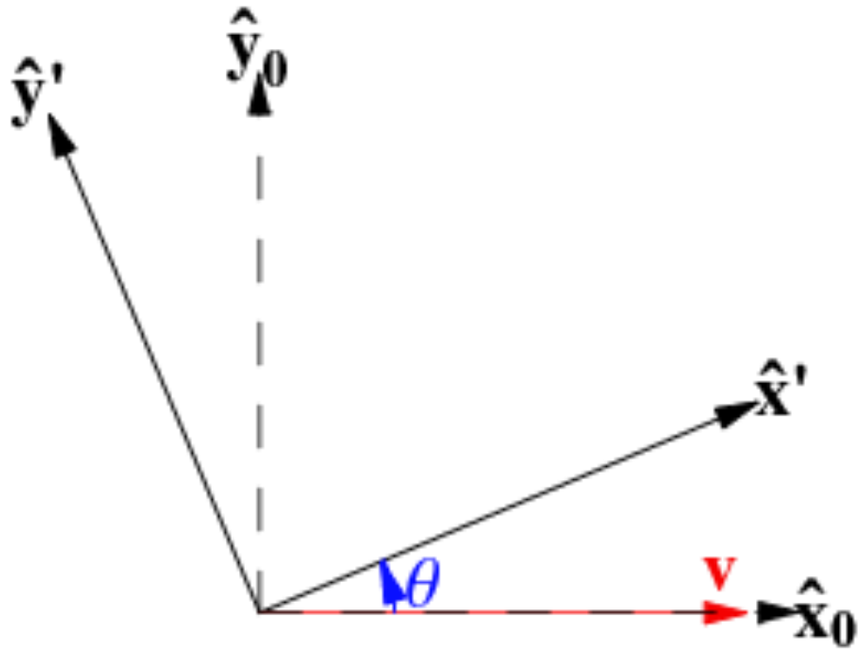
$$\mathsf{R}_\theta = \begin{bmatrix} \cos\theta & -\sin\theta \\ \sin\theta & \cos\theta \end{bmatrix},$$

$$\mathbf{v}' = \mathsf{R}_\theta \, \mathbf{v}_0 .$$

What if $\hat{\Sigma}^{-1}$ does not exist? Replace it by $(\hat{\Sigma} + \eta \mathbf{I}_p)^{-1}$ where η is a small number, or compute $\hat{\Sigma}^{-1}$ as follows (assume $p = 3$)

$$\hat{\Sigma} = U_{p\times 3} \begin{pmatrix} d_1 & 0 & 0 \\ 0 & d_2 & 0 \\ 0 & 0 & 0 \end{pmatrix} U^t, \quad \hat{\Sigma}^{-1} = U \begin{pmatrix} 1/d_1 & 0 & 0 \\ 0 & 1/d_2 & 0 \\ 0 & 0 & 0 \end{pmatrix} U^t.$$

3rd-dim (after rotation) is the null space for all classes.



$$R_\theta = \begin{bmatrix} \cos\theta & -\sin\theta \\ \sin\theta & \cos\theta \end{bmatrix},$$

Reduced Rank LDA

- Suppose Σ is an identity matrix $\mathbf{I}_p$. Then, we can write the discriminant function for LDA as

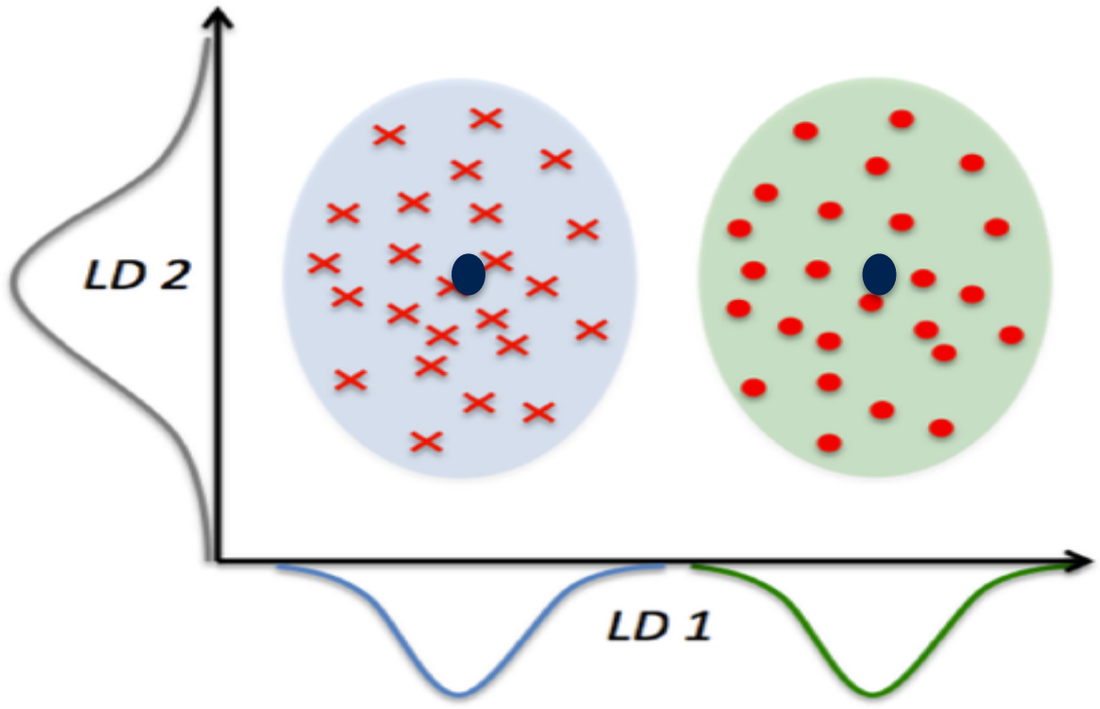
$$d_k(\mathbf{x}) = \|\mathbf{x} - \boldsymbol{\mu}_k\|^2 - 2\log \pi_k. \tag{2}$$

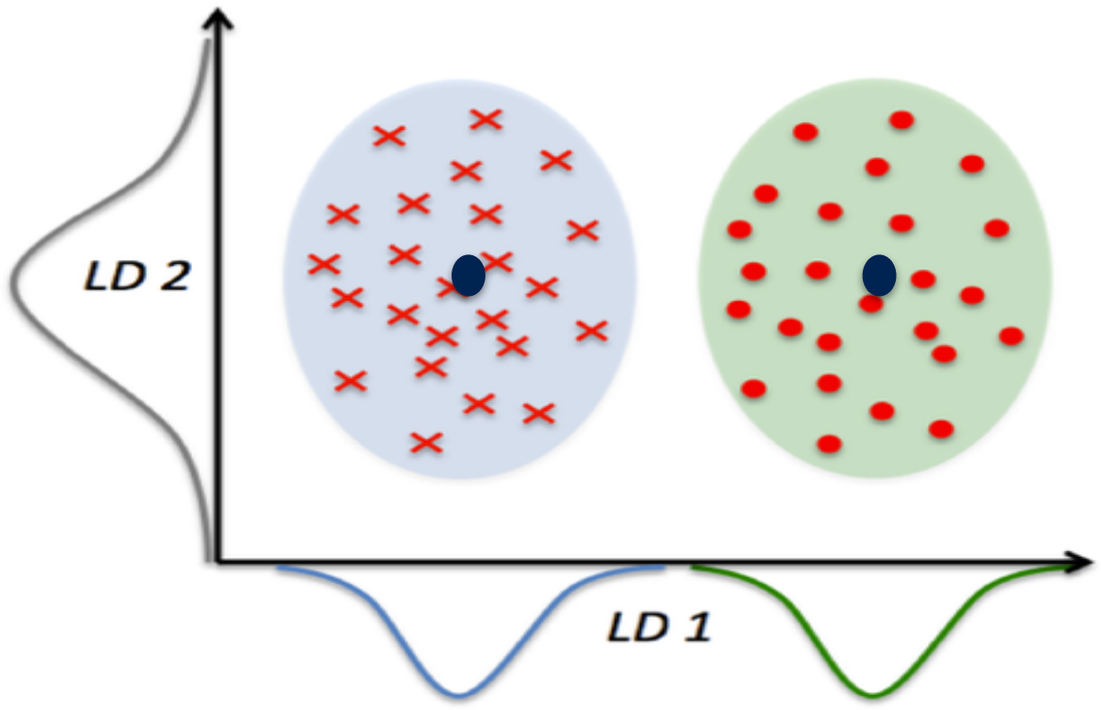
  The feature vector $\mathbf{x}$ only appears in the 1st term which is the squared distance from $\mathbf{x}$ to $\boldsymbol{\mu}_k$, the center of the kth class.

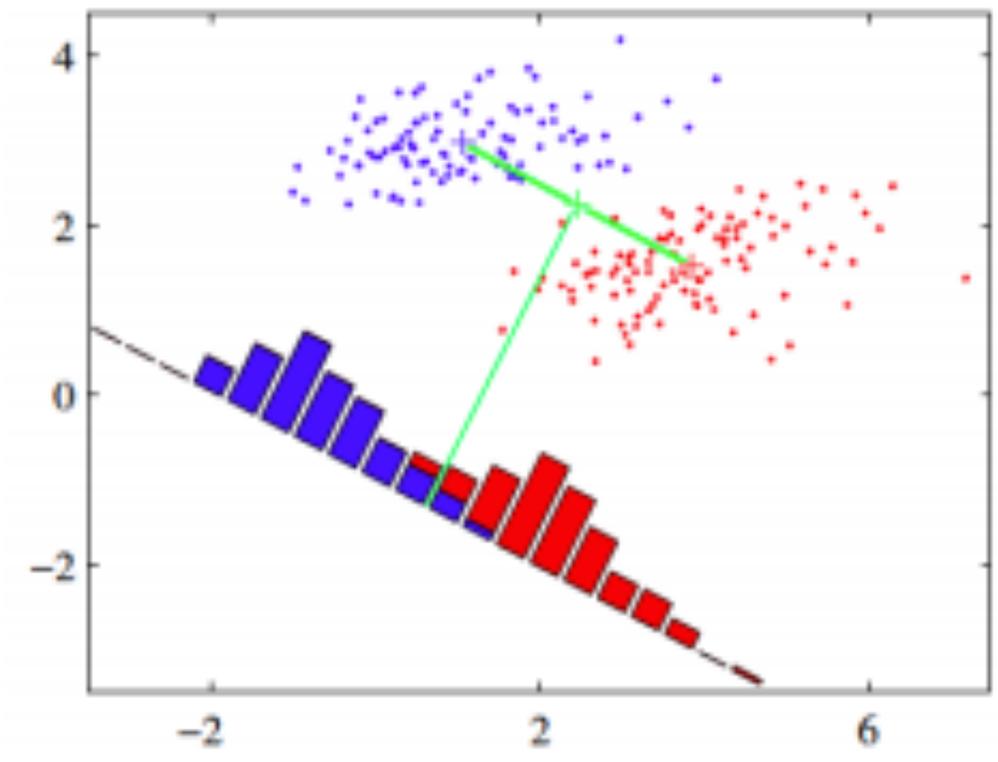
- Two points determine a line; three points determine a plane; K class centers determine a $(K-1)$-dim subspace.

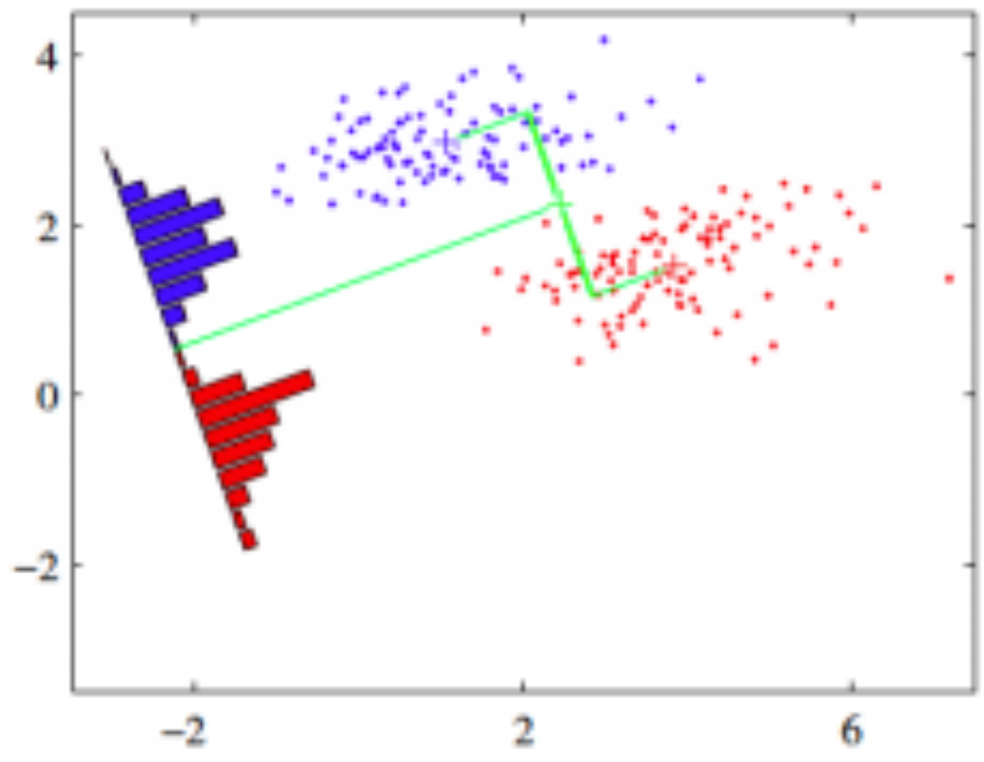
- Next we'll show that we can replace the squared distance $\|\mathbf{x} - \boldsymbol{\mu}_k\|^2$ in the original p-dim space by a square distance in a $(K-1)$-dim subspace. That is, LDA naturally leads to dimension reduction from p to $K-1$.

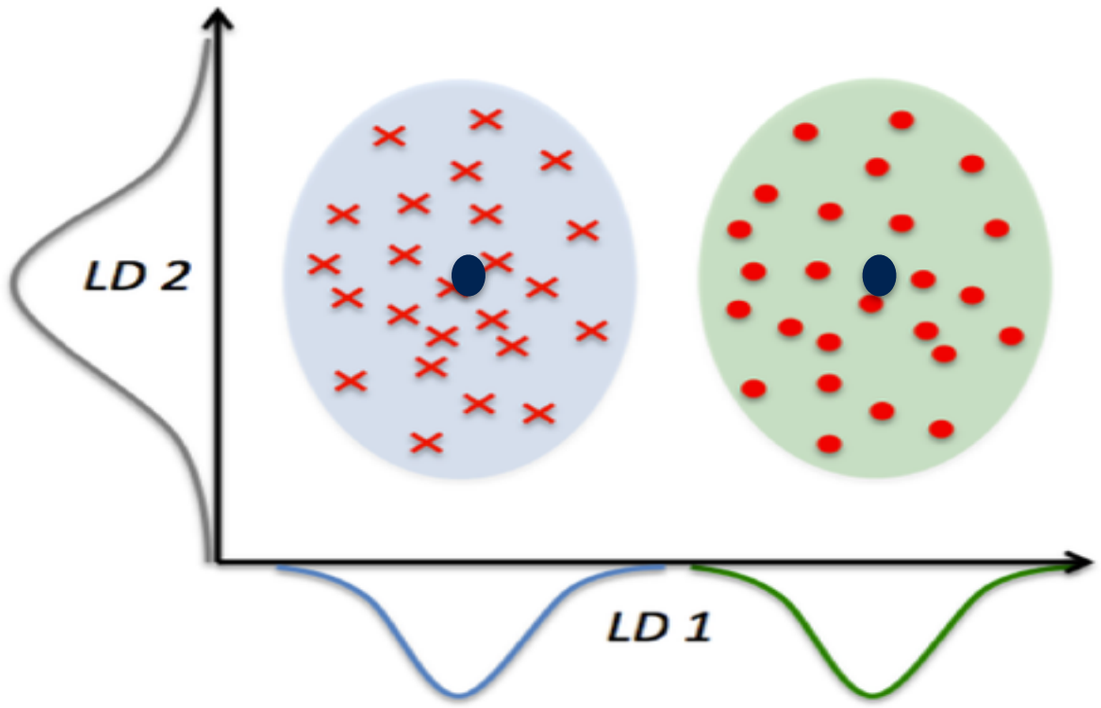
when (K-1)<p


LD 2
LD 1

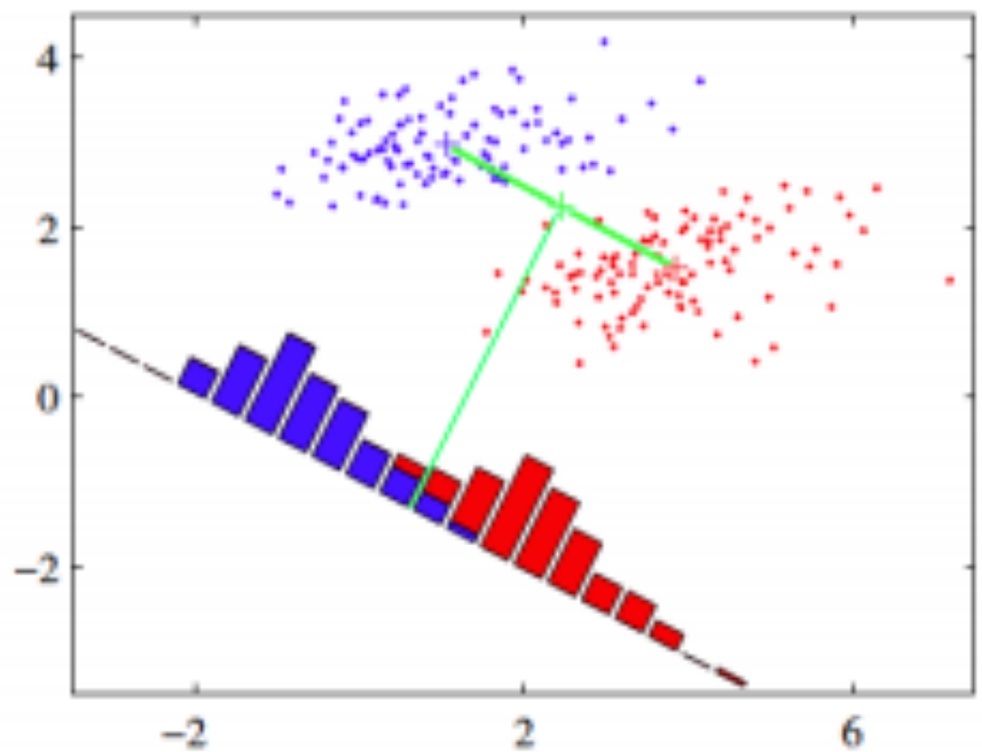

LD 2
LD 1

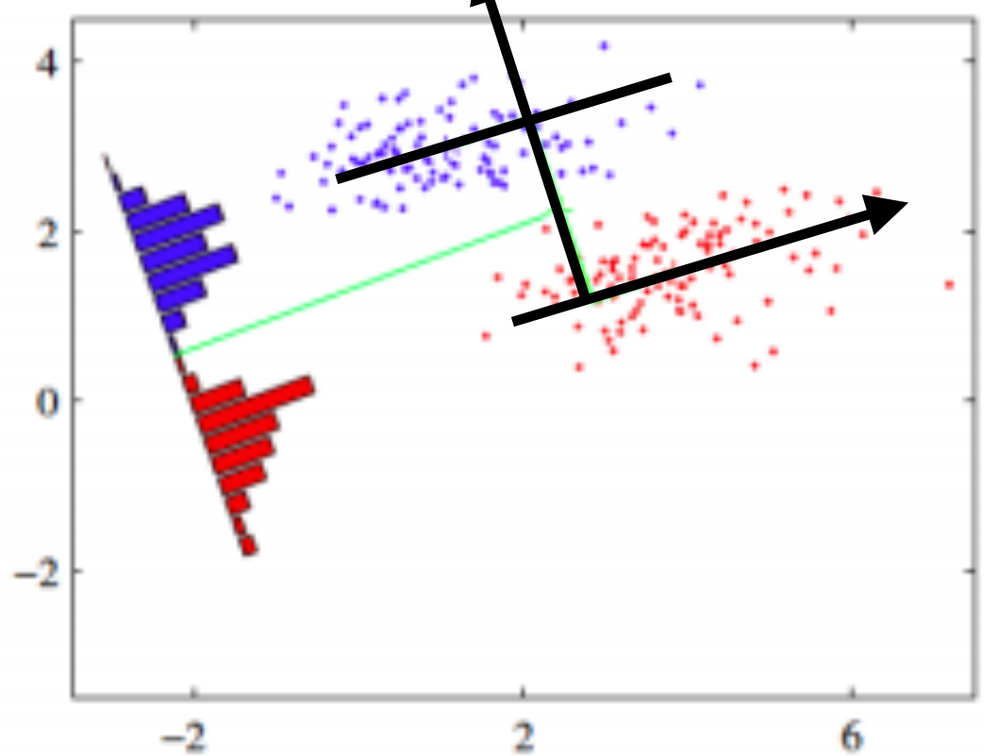

4
2
0
−2
−2
2
6


4
2
0
−2
−2
2
6



new coordinates





Output: $(\pi_k, \boldsymbol{\mu}_k)_{k=1}^K$ and Σ.

- Make Prediction
  - Input: test point $\mathbf{x}^*$ and output from Training
  - Compute SVD of $\Sigma = UD^2U^t$, and $A = UD^{-1}U^t$

    A = $1/\sigma$
  - Compute the projection matrix $\mathbf{H}$ based on $(A\boldsymbol{\mu}_1, \ldots, A\boldsymbol{\mu}_K)$

$$B_{p\times(K-1)} = \Big[A(\boldsymbol{\mu}_1 - \bar{\boldsymbol{\mu}}), \ldots, A(\boldsymbol{\mu}_{K-1} - \bar{\boldsymbol{\mu}})\Big], \quad \mathbf{H} = B(B^tB)^{-1}B.$$

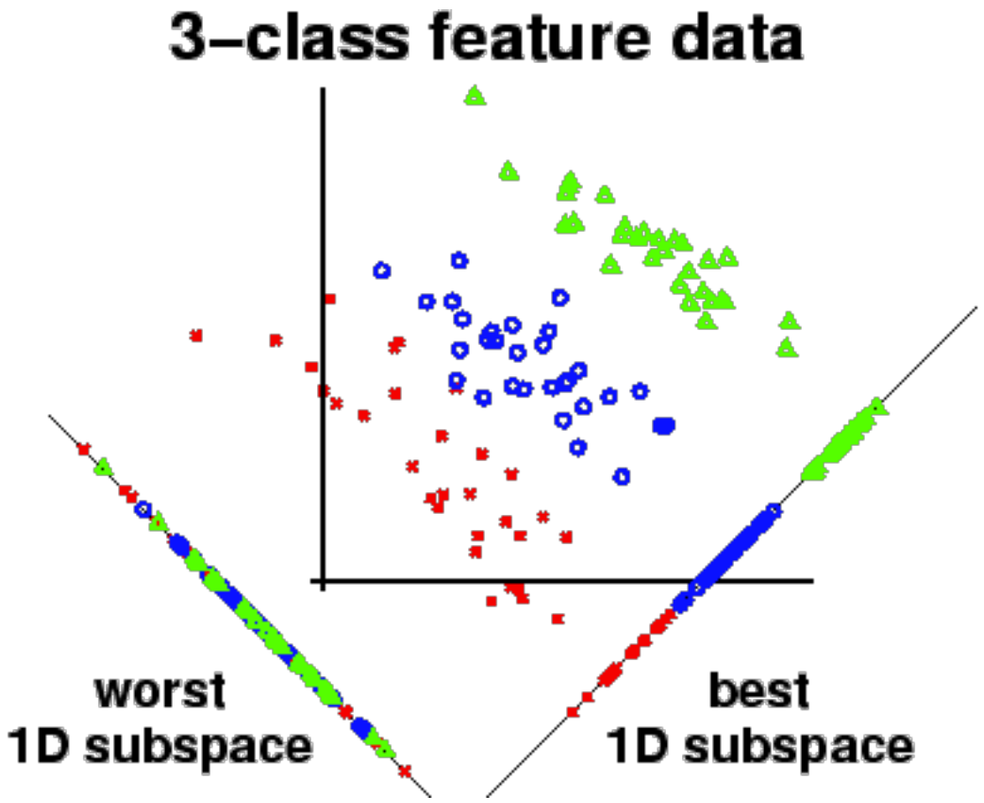
  - Dimension reduction: $\tilde{\mathbf{x}}^* = \mathbf{H}A\mathbf{x}^*$
  - For k = 1:K, compute

$$d_k(\mathbf{x}^*) = \|\tilde{\mathbf{x}}^* - \boldsymbol{\mu}_k\|^2 - 2\log\pi_k.$$

  - Output: $\arg\min_k d_k(\mathbf{x}^*)$.

How we compute the projection of x onto the K centers? Run a regression of x wrt a design matrix formed by (K-1) centers.

Fisher Discriminant Analysis



Supervised dimension reduction

Find a direction such that the projection of data (of multiple classes) onto this direction is well separated.

What's being well separated? The group means of u_i's are far apart from each other, and within each group, the variation/spread is small, i.e., minimize the following ratio

should be maximize

$$\frac{\text{Between group variation}}{\text{Within group variation}} = \frac{\mathbf{a}^t B \mathbf{a}}{\mathbf{a}^t W \mathbf{a}}.$$

The within-class and between-class sample covariance matrices

$$
\begin{aligned}
W_{p\times p} &= \frac{1}{n-K}\sum_{k=1}^{K}\sum_{i:y_i=k}(\mathbf{x}_i-\hat{\boldsymbol{\mu}}_k)(\mathbf{x}_i-\hat{\boldsymbol{\mu}}_k)^t \\
B_{p\times p} &= \frac{1}{K-1}\sum_{k=1}^{K} n_k(\hat{\boldsymbol{\mu}}_k-\bar{\hat{\boldsymbol{\mu}}})(\hat{\boldsymbol{\mu}}_k-\bar{\hat{\boldsymbol{\mu}}})^t
\end{aligned}
$$

The following equalities are trivial.

$$
\begin{aligned}
\mathbf{x}_1 &= \mathbf{x}_1-\hat{\boldsymbol{\mu}}_{y_1}+\hat{\boldsymbol{\mu}}_{y_1} \\
\mathbf{x}_2 &= \mathbf{x}_2-\hat{\boldsymbol{\mu}}_{y_2}+\hat{\boldsymbol{\mu}}_{y_2} \\
\ldots &= \ldots \\
\mathbf{x}_n &= \mathbf{x}_n-\hat{\boldsymbol{\mu}}_{y_n}+\hat{\boldsymbol{\mu}}_{y_n}
\end{aligned}
$$

You can view W as the sample covariance matrix over $\mathbf{x}_i-\hat{\boldsymbol{\mu}}_{y_i}$ (same as the pooled sample covariance matrix $\hat{\Sigma}$ in LDA), and B as the sample covariance matrix over the K class centers $\hat{\boldsymbol{\mu}}_{y_i}$.

Generalized Eigenvalue Problem

$$\max_{\mathbf{a}} \frac{\mathbf{a}^t B \mathbf{a}}{\mathbf{a}^t W \mathbf{a}} \Longrightarrow \max_{\mathbf{a}} \mathbf{a}^t B \mathbf{a} \text{ subj to } \mathbf{a}^t W \mathbf{a} = 1.$$

Assume $W = UD^2U^t$. Define $\mathbf{b} = W^{1/2}\mathbf{a}$, where $W^{1/2} := UDU^t$ is symmetric and write its inverse as $W^{-1/2} = UD^{-1}U^t$.

$$\begin{aligned} \mathbf{a}^t B \mathbf{a} &= \mathbf{a}^t W^{1/2} W^{-1/2} B W^{-1/2} W^{1/2} \mathbf{a} \\ &= (W^{1/2}\mathbf{a})^t W^{-1/2} B W^{-1/2} (W^{1/2}\mathbf{a}) \\ &= \mathbf{b}^t W^{-1/2} B W^{-1/2} \mathbf{b}, \quad \text{subj to } \|\mathbf{b}\|^2 = 1. \end{aligned}$$

The optimization above is a classical eigenvalue problem!

Different from PCA.

We can solve the directions sequentially as follows

- $\mathbf{b}_1 =$ the 1st eigen-vector of matrix $W^{-1/2}BW^{-1/2}$, then solve $\mathbf{a}_1 = W^{-1/2}\mathbf{b}_1$.
- $\mathbf{b}_2 =$ the 2nd eigen-vector of matrix $W^{-1/2}BW^{-1/2}$, then solve $\mathbf{a}_2 = W^{-1/2}\mathbf{b}_2$.
- Note that although $\mathbf{b}_j$'s are orthonormal, but $\mathbf{a}_j$'s are not:

$$\mathbf{b}_j^t\mathbf{b}_j = 1, \quad \mathbf{a}_j^t W \mathbf{a}_j = 1$$

$$\mathbf{b}_j^t\mathbf{b}_l = 0, \quad \mathbf{a}_j^t W \mathbf{a}_l = 0.$$

- We can extract at most $(K-1)$ directions, since the rank of B is $(K-1)$. The $(K-1)$ directions span exactly the same space as the one from the reduced rank LDA.

Connection with reduced rank LDA
Consider a simple case :W = **I**, then what's the (K-1)-dim space spanned by eigenvectors of W?

should be B

LDA vs FDA

- LDA: a classifier.
- FDA: a dimension reduction method, i.e., the output of FDA is a set of directions, but not a classification rule.
- The normal assumption is never mentioned in FDA, but why the space from FDA is similar to the reduced space from LDA?

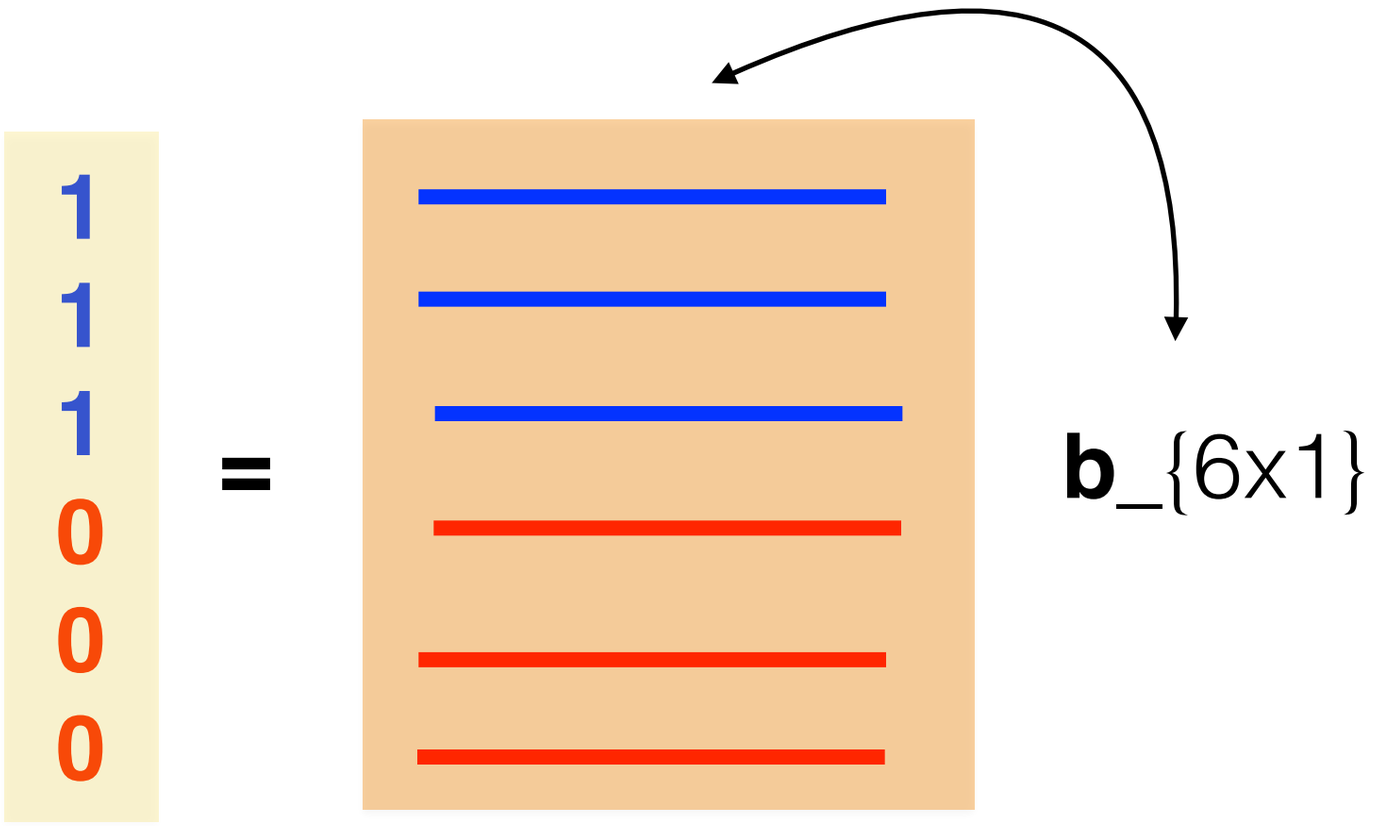
FDA implicitly assumes the data from each group follows or approximately follows a normal distribution with the same covariance matrix.

because we are only looking at the 2nd moment

Dimension reduction algorithm
FDA is a supervised.
PCA is unsupervised.
FDA can be applied on regression too.

What out for overfitting.

transform continuous y data into quantiles



We can project the 6-dim feature space into a one-dim space, such as the projections for the blue = 1, and red = 0.

So two-classes are well-separated.

LDA & QDA in High Dimension

- Singularity of the Covariance Matrix When dimension p is large, the inverse of $\hat{\Sigma}$ for $\hat{\Sigma}_k$ may not exist. For example, if $p > n$, $\hat{\Sigma}$, the $p \times p$ covariance matrix for LDA, is of rank less than n. So we cannot compute $\hat{\Sigma}^{-1}$.

  But singularity is not a serious issue, and we have discussed how to fix singularity for LDA and QDA. A more serious issue is overfitting.

- When dimension p is large, even LDA could end up overfitting the data: one can show that when p gets large, LDA could behave like random guessing (i.e., classification error $= 0.5$).

- Regularization: restrict matrices/vectors to be sparse (e.g., sparse LDA or regularization DA), or restrict features to be independent (NaiveBayes).

Naive Bayes

- Recall: for multi-class problems the optimal decision rule is

$$\arg\max_k \mathbb{P}(Y = k|X = \mathbf{x}) = \arg\max_k \pi_k f_k(\mathbf{x}).$$

- Require $f_k(\mathbf{x})$ to be

$$f_k(\mathbf{x}) = f_{k1}(x_1) \times f_{k2}(x_2) \cdots \times f_{kp}(x_p),$$

  i.e., each dim of $\mathbf{x}$ is independent (You can view independence as regularization for high-dimensional problems).

- Then each density f_{kj} $(j = 1 : p, k = 1 : K)$ is estimated separately within each class. E.g., discrete features via histograms; numerical features via kernel density estimates (nonparametric NB) or normal densities (parametric NB).

How many parameters for parametric **f_k**? A product of independent one-dim normals: **p means, p variances.**

Summary of Parametric Naive Bayes

- Training
  - `Input`: $(\mathbf{x}_i, y_i)_{i=1}^n$
  - `Output`: $\left(\pi_k, (\mu_{kj}, \sigma_{kj}^2)_{j=1:p}\right)_{k=1}^K$
- Make Prediction
  - `Input`: test point $\mathbf{x}^*$ and output from Training
  - For `k = 1:K`, compute

$$
\begin{aligned}
d_k(\mathbf{x}^*) &= -2\log\left[\pi_k \frac{1}{\sqrt{\sigma_{k1}^2}} e^{-\frac{(x_1^*-\mu_{k1})^2}{2\sigma_{k1}^2}} \cdots \frac{1}{\sqrt{\sigma_{kp}^2}} e^{-\frac{(x_p^*-\mu_{kp})^2}{2\sigma_{kp}^2}}\right] \\
&= -2\log\pi_k + \sum_{j=1}^{p}\left[\log\sigma_{kj}^2 + \frac{(x_j^*-\mu_{kj})^2}{\sigma_{kj}^2}\right]
\end{aligned}
$$

  - `Output`: $\arg\min_k d_k(\mathbf{x}^*)$.

Parameters:
pi: K-by-1
mu: K-by-p
sigmasq: K-by-p

Summary: Discriminant Analysis

In Discriminant Analysis (DA), we estimate the joint

$$P(X = \mathbf{x}, Y = k) = P(X = \mathbf{x}|Y = k) \times P(Y = k),$$

and then obtain $P(Y = k|X = \mathbf{x})$.

DA is conceptually simple and works for some low-dimensional problems,

but not an effective way of building classifiers.

P(x, y) = P(Y=y | X=x) P(X=x)
= P(X=x | Y=y) P(Y=y)

Joint dist **Conditions of XIY** **Marginal of Y**

Dist of p-dim X given Y=k: **QDA, LDA (FDA), NB**

For example, for binary LDA with discriminant function (1):

$$d_k(\mathbf{x}) = -2\mathbf{x}^t\Sigma^{-1}\boldsymbol{\mu}_k + \boldsymbol{\mu}_k^T\Sigma^{-1}\boldsymbol{\mu}_k - 2\log\pi_k$$

What matters is the decision boundary which is a linear function has $(p+1)$ parameters:

$$d_1(\mathbf{x}) - d_2(\mathbf{x}) = -2\mathbf{x}^t\Sigma^{-1}(\boldsymbol{\mu}_1 - \boldsymbol{\mu}_2) + \beta_0 = \mathbf{x}^t\boldsymbol{\beta} + \beta_0.$$

However, we estimate $(\boldsymbol{\beta}, \beta_0)$ by learning a much larger collection of parameters such as Σ, $\boldsymbol{\mu}_1$, $\boldsymbol{\mu}_2$ and π_1.

- Next we'll discuss how to directly learn $P(Y = k|X = \mathbf{x})$ (e.g., logistic regression, tree models) or directly learn the decision boundary (e.g., SVM).