Lecture 5
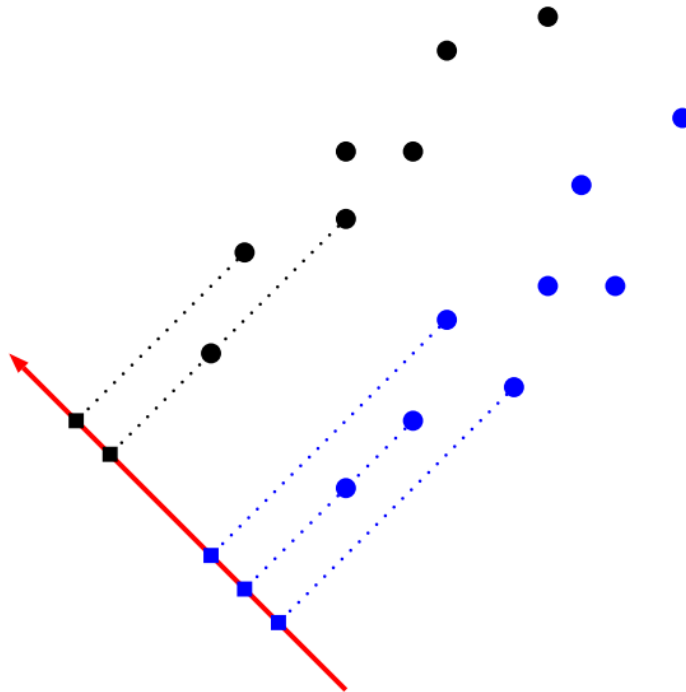
Linear Discriminant Analysis

Linear Discriminant Analysis [Fisher, 1936]

Given a training data set $x_1, \ldots, x_n \in \mathbb{R}^d$ consisting of two classes, find a (unit-vector) direction that "best" discriminates between the two classes.

LDA is a supervised dimensionality reduction method that tries to preserve the discriminatory information between classes.

# Linear Discriminant Analysis [Fisher, 1936]

- First, observe that projections of the two classes onto parallel lines always have "the same amount of separation".

- This time we are going to focus on lines that pass through the origin.

- Consider any unit vector $\boldsymbol{w} \in \mathbb{R}^d$, the 1D projections of the points are

$$a_i = \boldsymbol{w}^T \boldsymbol{x}_i, i = 1, \dots, n$$
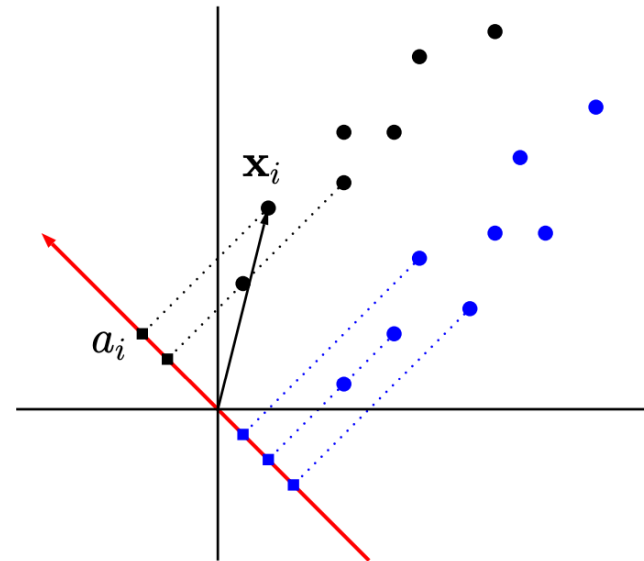
Linear Discriminant Analysis [Fisher, 1936]

Now the data look like this:
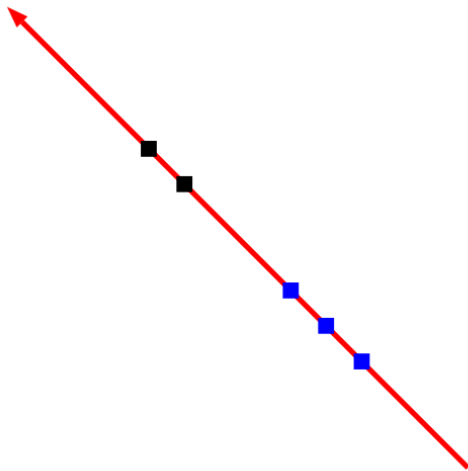
How do we quantify the separation between the two classes (in order to compare different directions v and select the best one)?

One (naive) idea is to measure the distance between the two class means in the 1D projection space: $|\mu_1 - \mu_2|$, where

$$\mu_1 = \frac{1}{n_1} \sum_{\mathbf{x}_i \in X_1} a_i = \frac{1}{n_1} \sum_{\mathbf{x}_i \in X_1} \mathbf{w}^T \mathbf{x}_i$$

$$= \mathbf{w}^T \cdot \frac{1}{n_1} \sum_{\mathbf{x}_i \in X_1} \mathbf{x}_i = \mathbf{w}^T \mathbf{m}_1$$

and similarly,

$$\mu_2 = \mathbf{w}^T \mathbf{m}_2, \quad \mathbf{m}_2 = \frac{1}{n_2} \sum_{\mathbf{x}_i \in X_2} \mathbf{x}_i$$

Linear Discriminant Analysis [Fisher, 1936]
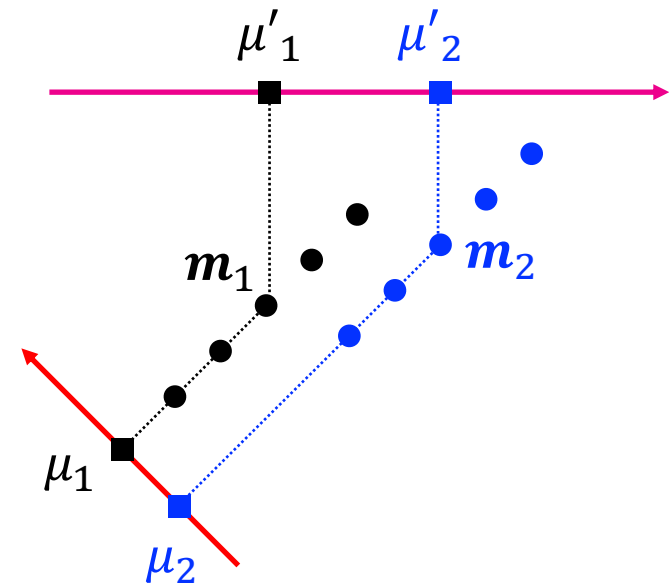
That is, we solve the following problem

$$\max_{\mathbf{w}:\|\mathbf{w}\|=1} |\mu_1 - \mu_2|$$

where

$$\mu_j = \mathbf{w}^T \mathbf{m}_j, \quad j = 1, 2.$$

However, this criterion does not always work (as shown in the right plot).

What else do we need to control?

Linear Discriminant Analysis [Fisher, 1936]

It turns out that we should also pay attention to the variances of the projected classes:

$$s_1^2 = \sum_{\mathbf{x}_i \in X_1} (a_i - \mu_1)^2, \quad s_2^2 = \sum_{\mathbf{x}_i \in X_2} (a_i - \mu_2)^2$$

Ideally, the projected classes have both faraway means and small variances.

This can be achieved through the following modified formulation:

$$\max_{\mathbf{w}:\|\mathbf{w}\|=1} \frac{(\mu_1 - \mu_2)^2}{s_1^2 + s_2^2}$$

The optimal w should be such that

- $(u_1 - u_2)^2$: large
- s_1^2, s_2^2: both small

Linear Discriminant Analysis [Fisher, 1936]

First, we derive a formula for the distance between the two projected centroids:

$$(\mu_1 - \mu_2)^2 = \left(\mathbf{w}^T\mathbf{m}_1 - \mathbf{w}^T\mathbf{m}_2\right)^2 = \left(\mathbf{w}^T(\mathbf{m}_1 - \mathbf{m}_2)\right)^2$$
$$= \mathbf{w}^T(\mathbf{m}_1 - \mathbf{m}_2) \cdot (\mathbf{m}_1 - \mathbf{m}_2)^T\mathbf{w}$$
$$= \mathbf{w}^T\mathbf{S}_b\mathbf{w}$$

where

$$\mathbf{S}_b = (\mathbf{m}_1 - \mathbf{m}_2)(\mathbf{m}_1 - \mathbf{m}_2)^T \in \mathbb{R}^{d \times d}$$

is called the **between-class scatter matrix**.

Remark. Clearly, $\mathbf{S}_b$ is square, symmetric and positive semidefinite. Moreover, $\text{rank}(\mathbf{S}_b) = 1$, which implies that it only has 1 positive eigenvalue!

Linear Discriminant Analysis [Fisher, 1936]

Next, for each class $j = 1, 2$, the variance of the projection (onto $\boldsymbol{w}$) is

$$
\begin{aligned}
s_j^2 &= \sum_{\mathbf{x}_i \in X_j} (a_i - \mu_j)^2 = \sum_{\mathbf{x}_i \in X_j} \left(\mathbf{w}^T \mathbf{x}_i - \mathbf{w}^T \mathbf{m}_j \right)^2 \\
&= \sum_{\mathbf{x}_i \in X_j} \mathbf{w}^T (\mathbf{x}_i - \mathbf{m}_j)(\mathbf{x}_i - \mathbf{m}_j)^T \mathbf{w} \\
&= \mathbf{w}^T \left[\sum_{\mathbf{x}_i \in X_j} (\mathbf{x}_i - \mathbf{m}_j)(\mathbf{x}_i - \mathbf{m}_j)^T \right] \mathbf{w} \\
&= \mathbf{w}^T \mathbf{S}_j \mathbf{w}
\end{aligned}
$$

where

$$
\mathbf{S}_j = \sum_{\mathbf{x}_i \in X_j} (\mathbf{x}_i - \mathbf{m}_j)(\mathbf{x}_i - \mathbf{m}_j)^T \in \mathbb{R}^{d \times d}
$$

is called the **within-class scatter matrix** for class j.

Linear Discriminant Analysis [Fisher, 1936]

The total within-class scatter of the two classes in the projection space is

$$s_1^2 + s_2^2 = \mathbf{w}^T \mathbf{S}_1 \mathbf{w} + \mathbf{w}^T \mathbf{S}_2 \mathbf{w} = \mathbf{w}^T (\mathbf{S}_1 + \mathbf{S}_2) \mathbf{w} = \mathbf{w}^T \mathbf{S}_w \mathbf{w}$$

where

$$\mathbf{S}_w = \mathbf{S}_1 + \mathbf{S}_2 = \sum_{\mathbf{x}_i \in X_1} (\mathbf{x}_i - \mathbf{m}_1)(\mathbf{x}_i - \mathbf{m}_1)^T + \sum_{\mathbf{x}_i \in X_2} (\mathbf{x}_i - \mathbf{m}_2)(\mathbf{x}_i - \mathbf{m}_2)^T$$

is called the **total within-class scatter matrix** of the original data.

Remark. $S_w \in \mathbb{R}^{d \times d}$ is also square, symmetric, and positive semidefinite.

Linear Discriminant Analysis [Fisher, 1936]

Putting everything together, we have derived the following optimization problem:

$$\max_{\mathbf{w}:\|\mathbf{w}\|=1} \frac{\mathbf{w}^T\mathbf{S}_b\mathbf{w}}{\mathbf{w}^T\mathbf{S}_w\mathbf{w}}$$

Generalized Rayleigh quotient
(广义瑞利商)

Theorem. Suppose $\boldsymbol{S}_w$ is nonsingular. The maximizer of the problem is given by the largest eigenvector $\boldsymbol{w}_1$ of $\boldsymbol{S}_w^{-1}\boldsymbol{S}_b$, i.e.,

$$\mathbf{S}_w^{-1}\mathbf{S}_b\mathbf{w}_1 = \lambda_1\mathbf{w}_1 \quad \longleftarrow \quad \mathbf{S}_b\boldsymbol{w} = \lambda\mathbf{S}_w\boldsymbol{w}$$

Remark. $\text{rank}(\boldsymbol{S}_w^{-1}\boldsymbol{S}_b) = \text{rank}(\boldsymbol{S}_b) = 1$, so λ_1 is the only nonzero (positive) eigenvalue that can be found. It represents the largest amount of separation between the two classes along any single direction.

Linear Discriminant Analysis [Fisher, 1936]

- Generalized Rayleigh quotient (广义瑞利商)

$$J = \frac{\boldsymbol{w}^{\mathrm{T}}\mathbf{S}_b\boldsymbol{w}}{\boldsymbol{w}^{\mathrm{T}}\mathbf{S}_w\boldsymbol{w}}$$

- Let $\boldsymbol{w}^{\mathrm{T}}\mathbf{S}_w\boldsymbol{w} = 1$, maximizing generalized Rayleigh quotient is equivalent to

$$\min_{\boldsymbol{w}} \ - \boldsymbol{w}^{\mathrm{T}}\mathbf{S}_b\boldsymbol{w}$$

$$\text{s.t.} \ \boldsymbol{w}^{\mathrm{T}}\mathbf{S}_w\boldsymbol{w} = 1$$

- Using the method of Lagrangian multipliers

$$\mathbf{S}_b\boldsymbol{w} = \lambda\mathbf{S}_w\boldsymbol{w}$$

Linear Discriminant Analysis [Fisher, 1936]

Instead of solving a generalized eigenvalue problem, a smartest way is to rewrite as

$$\lambda \mathbf{w} = \mathbf{S}_w^{-1} \underbrace{(\mathbf{m}_1 - \mathbf{m}_2)(\mathbf{m}_1 - \mathbf{m}_2)^T}_{\mathbf{S}_b} \mathbf{w}$$

$$= \mathbf{S}_w^{-1}(\mathbf{m}_1 - \mathbf{m}_2) \cdot \underbrace{(\mathbf{m}_1 - \mathbf{m}_2)^T \mathbf{w}}_{\text{scalar}}$$

This implies that

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

and it can be computed from $\mathbf{w} \propto \mathbf{S}_w^{-1}(\mathbf{m}_1 - \mathbf{m}_2)$ through rescaling!

Remark. Here, inverting $\mathbf{S}_w$ can be done through singular value decomposition $\mathbf{S}_w = \mathbf{U}\boldsymbol{\Sigma}\mathbf{V}^T$

Singular Value Decomposition

Singular Value Decomposition (奇异值分解，SVD)

Any real matrix $\mathbf{A} \in \mathbb{R}^{m \times n}$ can be decomposed as

$$\mathbf{A} = \mathbf{U}\mathbf{\Sigma}\mathbf{V}^{\mathrm{T}}$$

$\mathbf{U} \in \mathbb{R}^{m \times m}$ is a unitary matrix of order m satisfying $\mathbf{U}^{\mathrm{T}}\mathbf{U} = \mathbf{I}$,

$\mathbf{V} \in \mathbb{R}^{n \times n}$ is a unitary matrix of order n satisfying $\mathbf{V}^{\mathrm{T}}\mathbf{V} = \mathbf{I}$,

$\mathbf{\Sigma} \in \mathbb{R}^{m \times n}$ is a $m \times n$ matrix with all of its elements take the value 0 except $(\mathbf{\Sigma})_{ii} = \sigma_i$, where σ_i are non-negative real numbers and $\sigma_1 \geqslant \sigma_2 \geqslant \ldots \geqslant 0$.

Singular Value Decomposition

- Applications of SVD, e.g., low-rank matrix approximation

to approximate a r-rank matrix $\mathbf{A}$ with a k-rank matrix $\widetilde{\mathbf{A}}$, where $k \leqslant r$,

$$\min_{\widetilde{\mathbf{A}} \in \mathbb{R}^{m \times n}} \|\mathbf{A} - \widetilde{\mathbf{A}}\|_F$$

$$\text{s.t. } \operatorname{rank}(\widetilde{\mathbf{A}}) = k.$$

SVD can provide an analytical solution:
- Performing SVD on A;
- obtain a matrix $\mathbf{\Sigma}_k$ by setting the $r - k$ smallest singular values in $\mathbf{\Sigma}$ to zero, (i.e., keep only the k largest singular values)

- the optimal solution: $\mathbf{A}_k = \mathbf{U}_k \mathbf{\Sigma}_k \mathbf{V}_k^{\mathrm{T}}$,

  $\mathbf{U}_k$ and $\mathbf{V}_k$ are, respectively, the first k columns of $\mathbf{U}$ and $\mathbf{V}$

Linear Discriminant Analysis

- **The idea of LDA:** project the same class samples onto a line, while samples of different classes are far away from each other.

  - To make the projection of similar samples as close as possible, we can make the covariance of the projection points of similar samples as small as possible

  - To make the projection of samples from different classes as far away as possible, we can make the distance between the class centers as large as possible

Extend LDA to multiclass

The global scatter matrix:

$$\mathbf{S}_w = \sum_{j=1}^{c} \sum_{\mathbf{x} \in X_j} (\mathbf{x} - \mathbf{m}_j)(\mathbf{x} - \mathbf{m}_j)^T, \quad \mathbf{S}_b = \sum_{j=1}^{c} n_j(\mathbf{m}_j - \mathbf{m})(\mathbf{m}_j - \mathbf{m})^T$$

We have arrived at the same kind of problem:

$$\max_{\mathbf{w}:\|\mathbf{w}\|=1} \frac{\mathbf{w}^T \mathbf{S}_b \mathbf{w}}{\mathbf{w}^T \mathbf{S}_w \mathbf{w}}$$

The solution is given by the largest eigenvector of $\boldsymbol{S}_w^{-1}\boldsymbol{S}_b$ ($\boldsymbol{S}_b$ is nonsingular):

$$\mathbf{S}_w^{-1}\mathbf{S}_b\mathbf{w}_1 = \lambda_1 \mathbf{w}_1$$

However, the formula $\mathbf{w} \propto \mathbf{S}_w^{-1}(\mathbf{m}_1 - \mathbf{m}_2)$ is no longer valid:

$$\lambda_1 \mathbf{w}_1 = \mathbf{S}_w^{-1}\mathbf{S}_b\mathbf{w}_1 = \mathbf{S}_w^{-1} \sum_{j} n_j(\mathbf{m}_j - \mathbf{m})\underbrace{(\mathbf{m}_j - \mathbf{m})^T \mathbf{w}_1}_{\text{scalar}}$$

Extend LDA to multiclass

The solution is given by the largest eigenvector of $\boldsymbol{S}_w^{-1}\boldsymbol{S}_b$ ($\boldsymbol{S}_b$ is nonsingular):

$$\mathbf{S}_w^{-1}\mathbf{S}_b\mathbf{w}_1 = \lambda_1\mathbf{w}_1$$

However, the formula $\mathbf{w} \propto \mathbf{S}_w^{-1}(\mathbf{m}_1 - \mathbf{m}_2)$ is no longer valid:

$$\lambda_1\mathbf{w}_1 = \mathbf{S}_w^{-1}\mathbf{S}_b\mathbf{w}_1 = \mathbf{S}_w^{-1}\sum_j n_j(\mathbf{m}_j - \mathbf{m})\underbrace{(\mathbf{m}_j - \mathbf{m})^T\mathbf{w}_1}_{\text{scalar}}$$

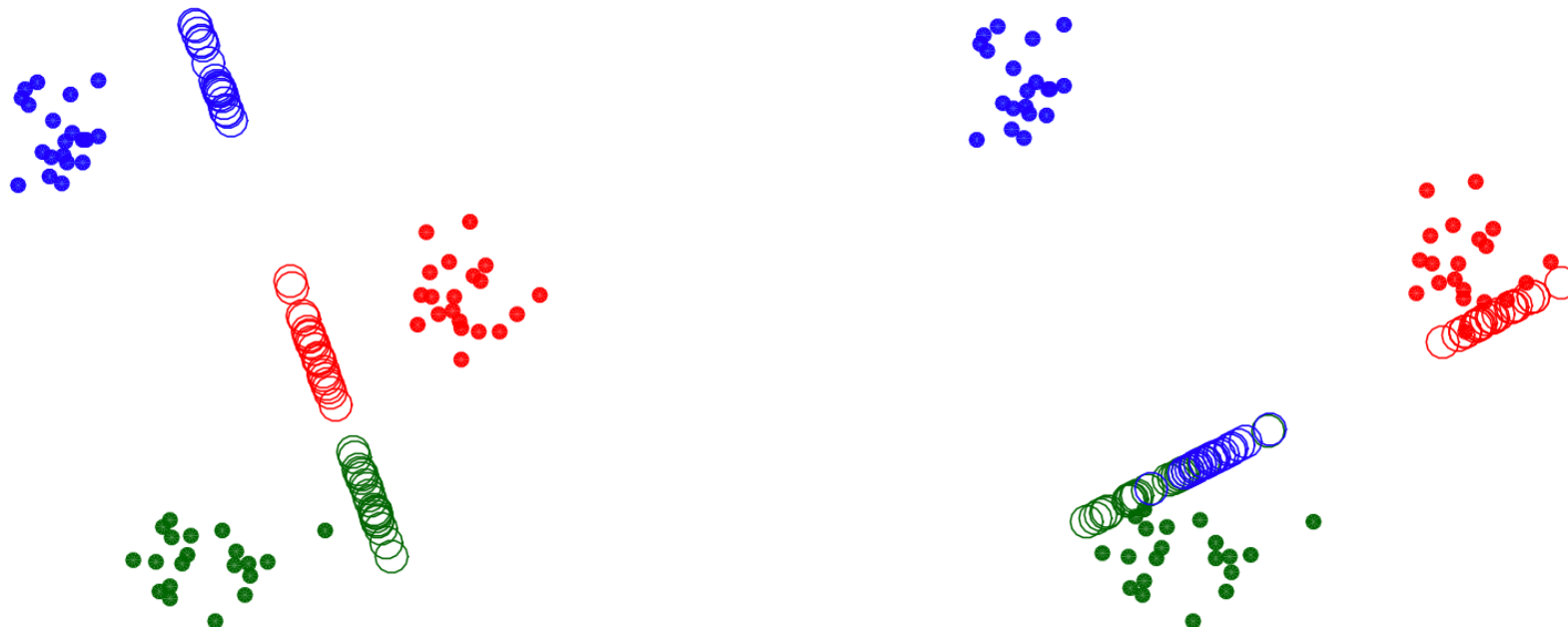So we have to find $\boldsymbol{w}_1$ by solving a generalized eigenvalue problem:

$$\mathbf{S}_b\boldsymbol{w} = \lambda\mathbf{S}_w\boldsymbol{w}$$

Extend LDA to multiclass
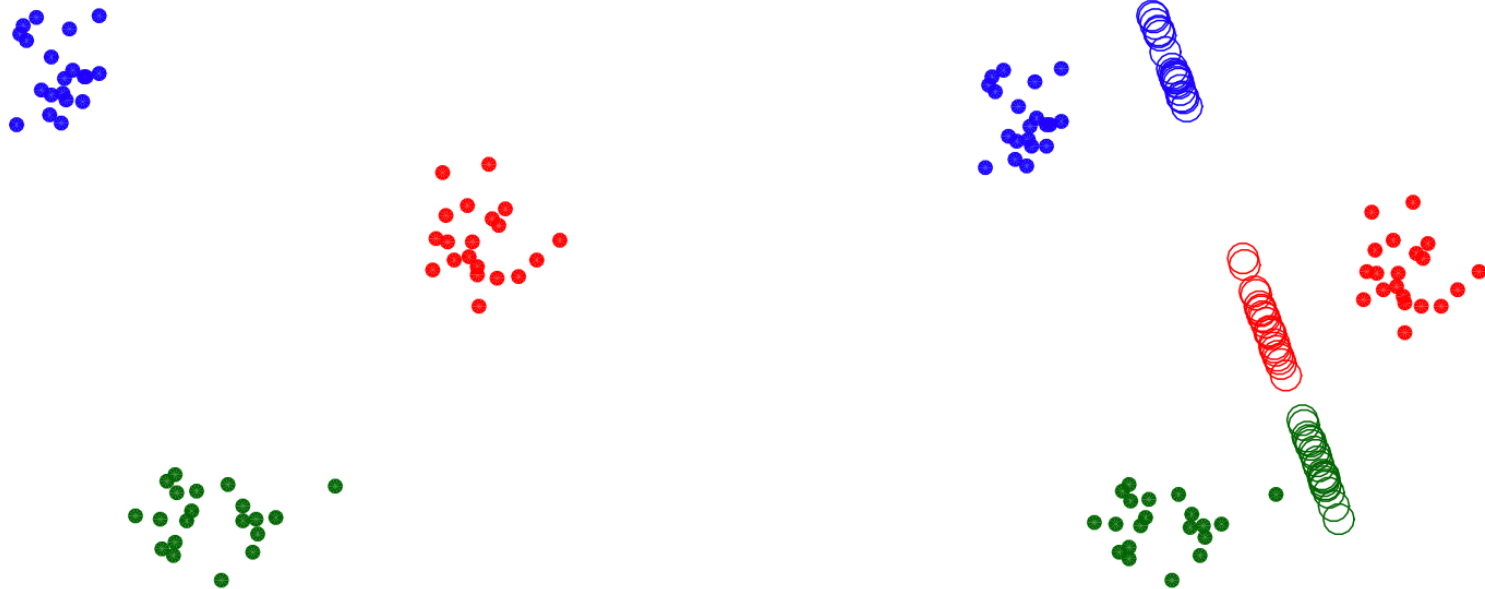
Extend LDA to multiclass

What about the second eigenvector w_2?

Extend LDA to multiclass

How many discriminatory directions can we find?

To answer this question, we just need to count the number of nonzero eigenvalues

$$\mathbf{S}_w^{-1}\mathbf{S}_b\mathbf{w} = \lambda\mathbf{w}$$

since only the nonzero eigenvectors will be used as the discriminatory directions.

In the above equation, the within-class scatter matrix $\boldsymbol{S}_w$ is *assumed to be nonsingular*. However, the between-class scatter matrix $\boldsymbol{S}_b$ is of low rank:

$$\mathbf{S}_b = \sum n_i(\mathbf{m}_i - \mathbf{m})(\mathbf{m}_i - \mathbf{m})^T$$

$$= [\sqrt{n_1}(\mathbf{m}_1 - \mathbf{m}) \cdots \sqrt{n_c}(\mathbf{m}_c - \mathbf{m})] \cdot \begin{bmatrix} \sqrt{n_1}(\mathbf{m}_1 - \mathbf{m})^T \\ \vdots \\ \sqrt{n_c}(\mathbf{m}_c - \mathbf{m})^T \end{bmatrix}$$

Extend LDA to multiclass

Observe that the columns of the matrix

$$[\sqrt{n_1}(\mathbf{m}_1 - \mathbf{m}) \cdots \sqrt{n_c}(\mathbf{m}_c - \mathbf{m})]$$

are linearly dependent:

$$\sqrt{n_1} \cdot \sqrt{n_1}(\mathbf{m}_1 - \mathbf{m}) + \cdots + \sqrt{n_c} \cdot \sqrt{n_c}(\mathbf{m}_c - \mathbf{m})$$

$$= (n_1 \mathbf{m}_1 + \cdots n_c \mathbf{m}_c) - (n_1 + \cdots + n_c)\mathbf{m}$$

$$= n\mathbf{m} - n\mathbf{m}$$

$$= \mathbf{0}.$$

The shows that $\text{rank}(S_b) \leq c - 1$ (where c is the number of training classes).

Therefore, one can only find at most $c - 1$ discriminatory directions.

Extend LDA to multiclass

- Objective

$$\max_{\mathbf{W}} \frac{\text{tr}\left(\mathbf{W}^{\text{T}}\mathbf{S}_b\mathbf{W}\right)}{\text{tr}\left(\mathbf{W}^{\text{T}}\mathbf{S}_w\mathbf{W}\right)}$$

$$\text{tr}(\mathbf{A}) = \sum_{i=1}^{n} A_{ii}$$

$$\frac{\partial}{\partial \mathbf{X}} \text{tr}\left(\mathbf{X}^{\text{T}}\mathbf{B}\mathbf{X}\right) = (\mathbf{B} + \mathbf{B}^{\text{T}})\mathbf{X}$$

where $\mathbf{W} \in \mathbb{R}^{d \times (N-1)}$

$$\mathbf{S}_b\mathbf{W} = \lambda\mathbf{S}_w\mathbf{W}$$

Concatenating the eigenvectors corresponding to the d' largest non-zero eigenvalues of $\mathbf{S}_w^{-1}\mathbf{S}_b$ leads to the closed-form solution of $\mathbf{W}$, where $d' \leq c - 1$

LDA for classification

- Explain from the aspect of Bayesian decision theory

  - It can be proved that we have the optimal solution of LDA when both classes follow Gaussian distribution with the same prior and covariance

Bayes Theorem for Classification

Thomas Bayes was a famous mathematician whose name represents a big subfield of statistical and probabilistic modeling. Here we focus on a simple result, known as Bayes theorem:

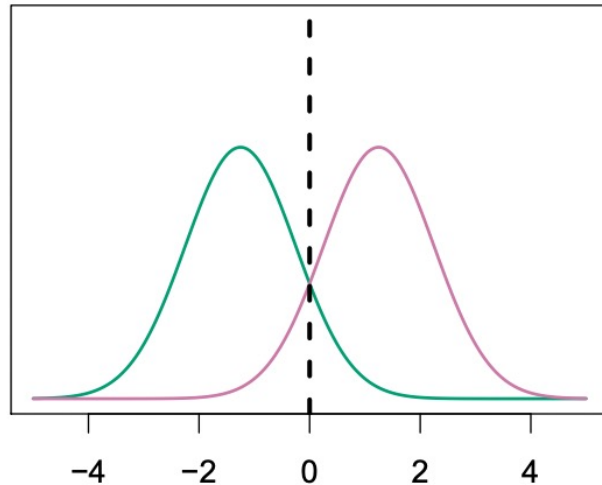$$\Pr(Y = k | X = x) = \frac{\Pr(X = x | Y = k) \cdot \Pr(Y = k)}{\Pr(X = x)}$$

One writes this slightly differently for discriminant analysis:

$$\Pr(Y = k | X = x) = \frac{\pi_k f_k(x)}{\sum_{l=1}^{K} \pi_l f_l(x)}, \quad \text{where}$$
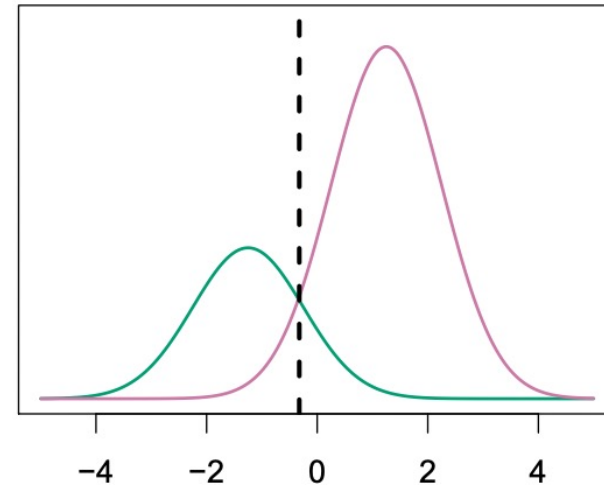
- $f_k(x) = \Pr(X = x \,|\, Y = k)$ is the density for X in class k. Here we will use normal densities for these, separately in each class.

- $\pi_k = \Pr(Y = k)$ is the marginal or prior probability for class k.

Classify to the highest density

$\pi_1 = .5, \quad \pi_2 = .5$

$\pi_1 = .3, \quad \pi_2 = .7$

- We classify a new point according to which density is highest

- When the priors are different, we take them into account as well, and compare $\pi_k \, f_k(x)$. On the right, we favor the pink class — the decision boundary has shifted to the left.

Linear Discriminant Analysis (p = 1)

The Gaussian density has the form

$$f_k(x) = \frac{1}{\sqrt{2\pi}\sigma_k} e^{-\frac{1}{2}\left(\frac{x-\mu_k}{\sigma_k}\right)^2}$$

Here μ_k is the mean, and σ_k^2 the variance (in class k). We will assume that all the $\sigma_k = \sigma$ are the same.

Plugging this into Bayes formula, we get a rather complex expression for $p_k(x) = \Pr(Y = k \mid X = x)$:

$$p_k(x) = \frac{\pi_k \frac{1}{\sqrt{2\pi}\sigma} e^{-\frac{1}{2}\left(\frac{x-\mu_k}{\sigma}\right)^2}}{\sum_{l=1}^{K} \pi_l \frac{1}{\sqrt{2\pi}\sigma} e^{-\frac{1}{2}\left(\frac{x-\mu_l}{\sigma}\right)^2}}$$

Happily, there are simplifications and cancellations.

Discriminant functions

To classify at the value $X = x$, we need to see which of the $p_k(x)$ is largest. Taking logs, and discarding terms that do not depend on k, we see that this is equivalent to assigning x to the class with the largest *discriminant score*:

$$\delta_k(x) = x \cdot \frac{\mu_k}{\sigma^2} - \frac{\mu_k^2}{2\sigma^2} + \log(\pi_k)$$
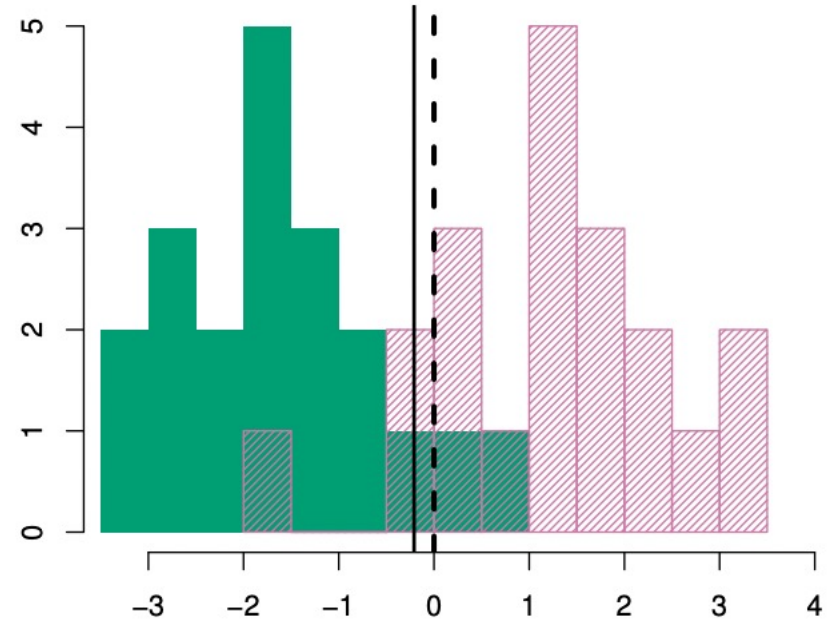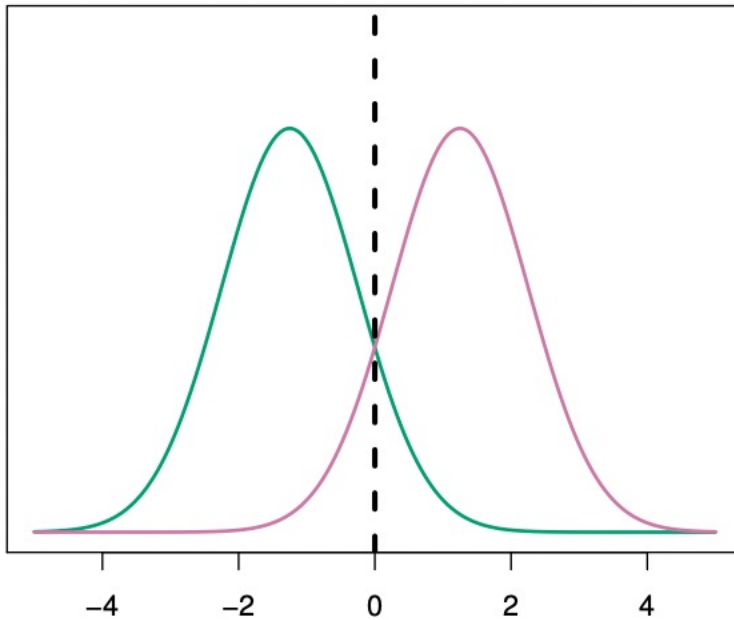
Note that $\delta_k(x)$ is a *linear* function of x.

If there are $K = 2$ classes and $\pi_1 = \pi_2 = 0.5$, then one can see that the decision boundary is at
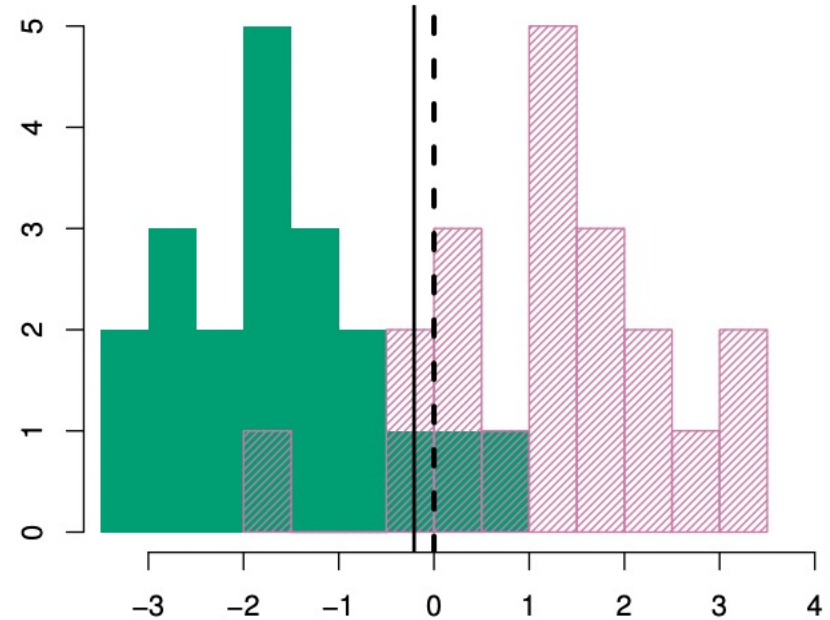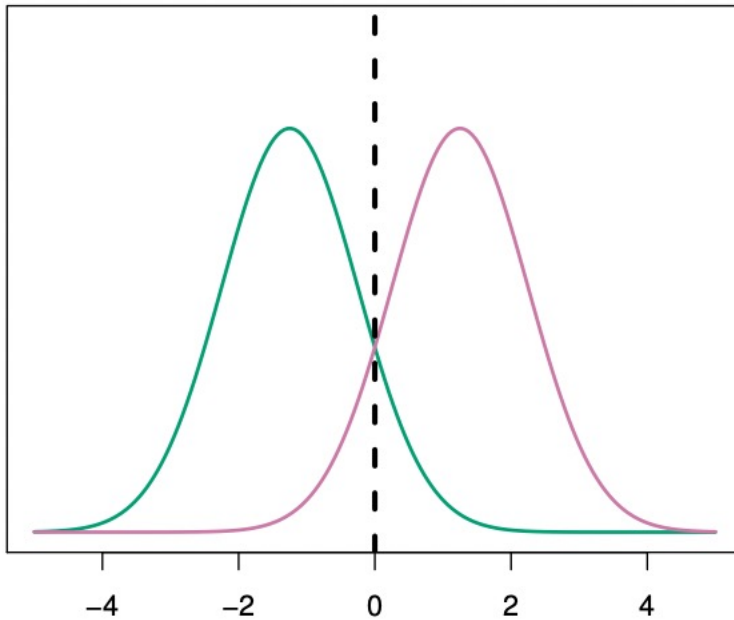
$$x = \frac{\mu_1 + \mu_2}{2}.$$

(See if you can show this)

Discriminant functions

Example with $\mu_1 = -1.5, \mu_2 = 1.5, \pi_1 = \pi_2 = 0.5,$ and $\sigma^2 = 1$

Discriminant functions

- Example with $\mu_1 = -1.5, \mu_2 = 1.5, \pi_1 = \pi_2 = 0.5$, and $\sigma^2 = 1$

- Typically we don't know these parameters; we just have the training data. In that case we simply estimate the parameters and plug them into the rule.

Estimating the parameters

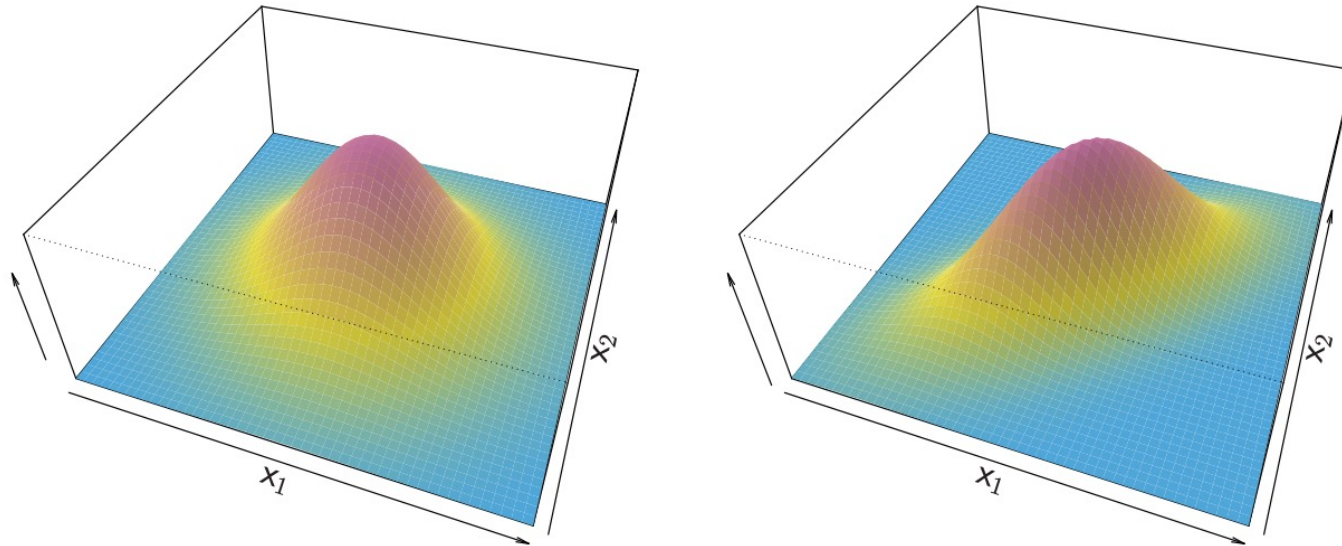$$\hat{\pi}_k = \frac{n_k}{n}$$

$$\hat{\mu}_k = \frac{1}{n_k} \sum_{i:\, y_i = k} x_i$$

$$\hat{\sigma}^2 = \frac{1}{n-K} \sum_{k=1}^{K} \sum_{i:\, y_i = k} (x_i - \hat{\mu}_k)^2$$

$$= \sum_{k=1}^{K} \frac{n_k - 1}{n - K} \cdot \hat{\sigma}_k^2$$

where $\hat{\sigma}_k^2 = \frac{1}{n_k - 1} \sum_{i:y_i=k}(x_i - \hat{\mu}_k)^2$ is the usual formula for the estimated variance in the *k*-th class.

Linear Discriminant Analysis (p > 1)

$$\Pr(Y = k | X = x) = \frac{\pi_k f_k(x)}{\sum_{l=1}^{K} \pi_l f_l(x)},$$

$$\text{Density: } f(x) = \frac{1}{(2\pi)^{p/2} |\boldsymbol{\Sigma}|^{1/2}} e^{-\frac{1}{2}(x-\mu)^T \boldsymbol{\Sigma}^{-1}(x-\mu)}$$

$$\text{Discriminant function: } \delta_k(x) = x^T \boldsymbol{\Sigma}^{-1} \mu_k - \frac{1}{2} \mu_k^T \boldsymbol{\Sigma}^{-1} \mu_k + \log \pi_k$$

The decision rule of LDA

$$\hat{f}^{\text{LDA}}(x) = \underset{k=1,\dots K}{\text{argmax}}\, \delta_k(x)$$

where $\delta_k(x)$ is the estimated discriminant function of class k,

$$\delta_k(x) = x^T \boldsymbol{\Sigma}^{-1} \mu_k - \frac{1}{2}\mu_k^T \boldsymbol{\Sigma}^{-1} \mu_k + \log \pi_k$$

Despite its complex form,

$$\delta_k(x) = c_{k0} + c_{k1}x_1 + c_{k2}x_2 + \dots + c_{kp}x_p \text{ — a linear function.}$$

LDA computations and sphering

Note that LDA equivalently minimizes over $k = 1, \ldots, K$

$$\frac{1}{2}(x - \mu_k)^T \Sigma^{-1} (x - \mu_k) - \log \pi_k$$

It helps to factorize Σ (i.e., compute its eigendecomposition):

$$\Sigma = UDU^T$$

where $U \in \mathbb{R}^{p \times p}$ has orthonormal columns (and rows), and $D = \text{diag}(d_1, \ldots d_p)$ with $d_j \geq 0$ for each j. Then we have $\Sigma^{-1} = UD^{-1}U^T$, and

$$(x - \mu_k)^T \Sigma^{-1} (x - \mu_k) = \| \underbrace{D^{-1/2}U^T x}_{\tilde{x}} - \underbrace{D^{-1/2}U^T \mu_k}_{\tilde{\mu}_k} \|_2^2$$

This is just the squared distance between $\tilde{x}$ and $\tilde{u}_k$

LDA procedure summarized

- Compute the sample estimates π_k, μ_k, Σ

- Factor Σ, as in $\Sigma = UDU^T$

- Transform the class centroids $\tilde{u}_k = D^{-1/2}U^T\mu_k$

- Given any point $x \in \mathbb{R}^p$, transform to $\tilde{x} = D^{-1/2}U^Tx \in \mathbb{R}^p$, and then classify according to the nearest centroid in the transformed space, adjusting for class proportions—this is the class k for which $\frac{1}{2}||\tilde{x} - \tilde{u}_k||_2^2 - \log \pi_k$ is smallest

What is this transformation doing?

$$\tilde{x} = D^{-1/2}U^Tx, \quad i = 1, \dots, n$$

This is basically sphering the data points, because if we think of $x \in \mathbb{R}^p$ were a random variable with covariance matrix Σ, then

$$\mathrm{Cov}\left(D^{-1/2}U^Tx\right) = D^{-1/2}U^T\Sigma UD^{-1/2} = I$$

Linear subspace spanned by sphered centroids

LDA compares the quantity $\frac{1}{2}||\tilde{x} - \tilde{u}_k||_2^2 - \log \pi_k$ across the classes $k = 1, \dots, K$. Consider the affine subspace $M \subseteq \mathbb{R}^p$ spanned by the transformed centroids $\tilde{u}_1, \dots, \tilde{u}_K$, which has dimension $K - 1$

For any $\tilde{x} \in \mathbb{R}^p$, we can decompose $\tilde{x} = P_M \tilde{x} + P_{M^\perp} \tilde{x}$, so

$$\|\tilde{x} - \tilde{\mu}_k\|_2^2 = \|\underbrace{P_M \tilde{x} - \tilde{\mu}_k}_{\in M} + \underbrace{P_{M^\perp} \tilde{x}}_{\in M^\perp}\|_2^2$$

$$= \|P_M \tilde{x} - \tilde{\mu}_k\|_2^2 + \|P_{M^\perp} \tilde{x}\|_2^2$$

The second term doesn't depend on k

What this is telling us: the LDA classification rule is unchanged if we project the points to be classified onto M, since the distances orthogonal to M don't matter

LDA procedure summarized

- Compute the sample estimates π_k, μ_k, Σ

- Make two transformations: first, sphere the data points, based on factoring Σ; second, project down to the affine subspace spanned by the sphered centroids. This can all be summarized a single linear transformation $A \in \mathbb{R}^{(K-1) \times p}$

- Given any point $x \in \mathbb{R}^p$, transform to $\tilde{x} = Ax \in \mathbb{R}^{K-1}$, and classify according to the class $k = 1, \dots, K$ for which

$$\frac{1}{2} ||\tilde{x} - \tilde{u}_k||_2^2 - \log \pi_k$$

is smallest, where $\tilde{u}_k = A\mu_k$

The decision rule of LDA

This way of describing LDA may sound more complicated, but actually, it's much simpler! After applying A, we've reduced the problem from p to $K - 1$ dimensions, and then it's basically nearest centroid classification:

$$\hat{f}^{\text{LDA}}(x) = \underset{k=1,\ldots.K}{\text{argmax}} \, ||\tilde{x} - \tilde{\mu}_k||_2^2 - \log \pi_k$$

(The only distinction being that we adjust for class proportions)

Example

Decision boundaries, using the formula that we derived:

3 classes with 8 features to a 2-dimensional subspace
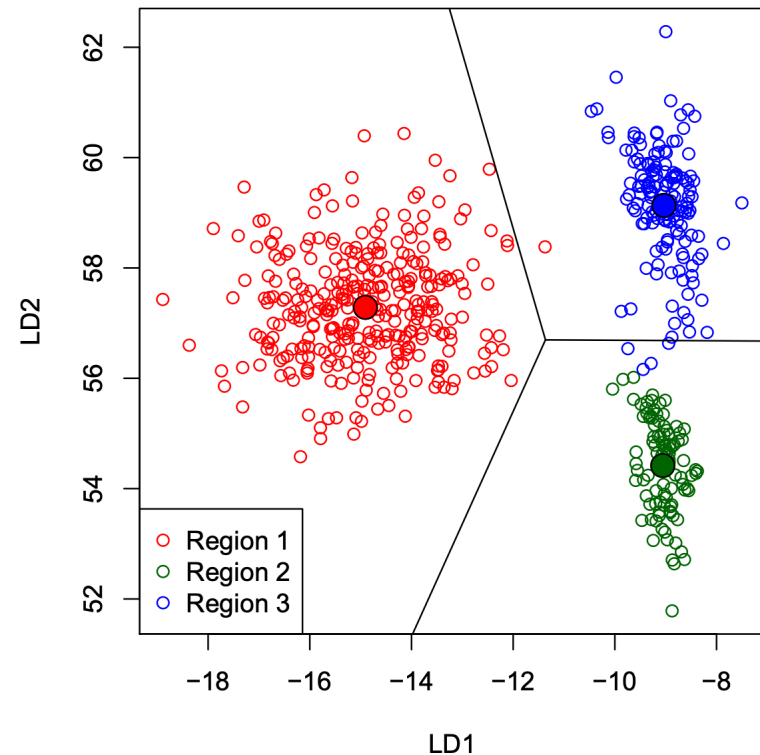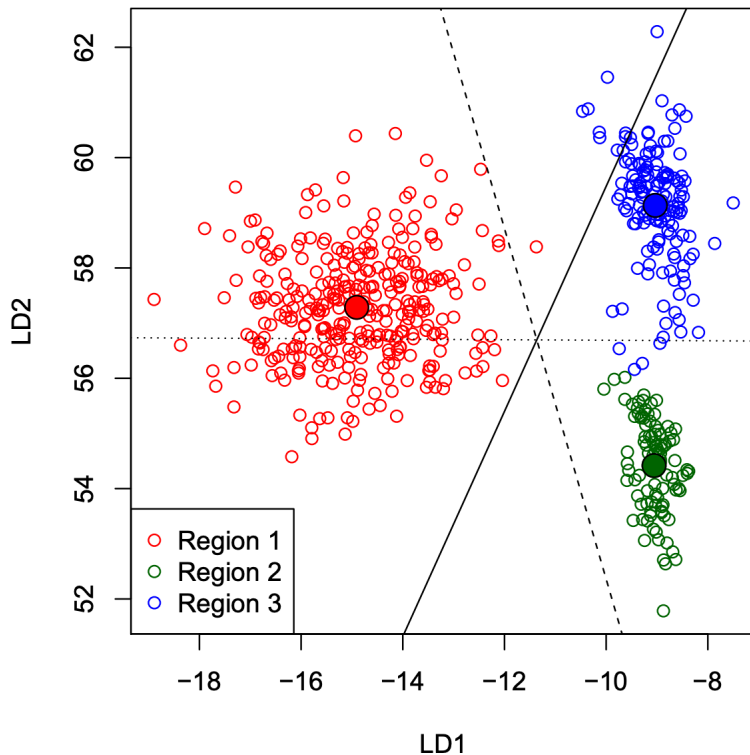
Reduced-rank linear discriminant analysis

- The dimension reduction from p to $K - 1$ was exact, in that we didn't change the LDA rule at all. Why might we want to reduce further to a dimension $L < K - 1$, if K is large?
  - Visualization
  - Regularization

- Reduced-rank linear discriminant analysis is a nice way to project down to lower than $K - 1$ dimensions. It chooses the lower dimensional subspaces so as to spread out the centroids as much as possible

The decision boundary (binary case)

Let us assume that $\boldsymbol{\Sigma}_1 = \boldsymbol{\Sigma}_2 = \boldsymbol{\Sigma}.$ Then:

$$(\mathbf{x} - \boldsymbol{\mu}_1)^\top \boldsymbol{\Sigma}^{-1}(\mathbf{x} - \boldsymbol{\mu}_1) = (\mathbf{x} - \boldsymbol{\mu}_2)^\top \boldsymbol{\Sigma}^{-1}(\mathbf{x} - \boldsymbol{\mu}_2)$$

$$2\mathbf{x}^\top \boldsymbol{\Sigma}^{-1}(\boldsymbol{\mu}_1 - \boldsymbol{\mu}_2) = \underbrace{\boldsymbol{\mu}_1^\top \boldsymbol{\Sigma}^{-1}\boldsymbol{\mu}_1 - \boldsymbol{\mu}_2^\top \boldsymbol{\Sigma}^{-1}\boldsymbol{\mu}_2}_{\text{const}}$$
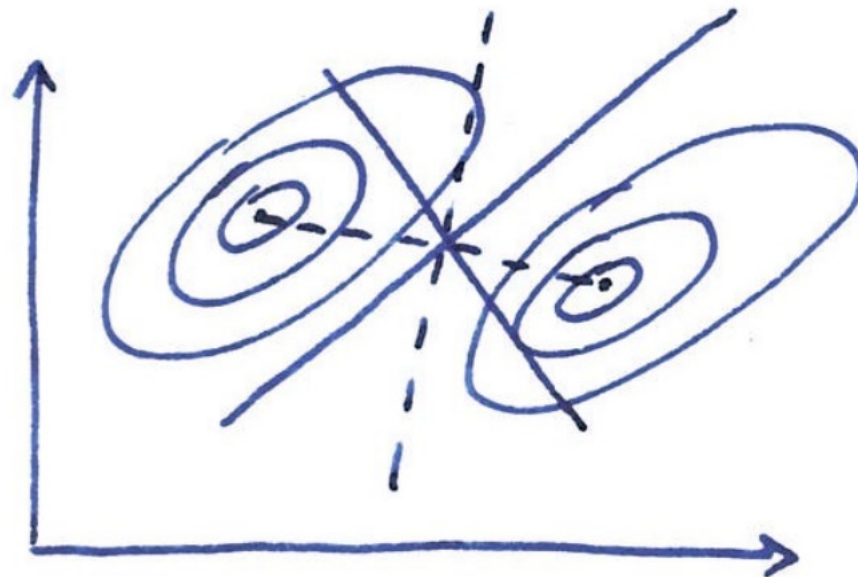
$$\boxed{\mathbf{x}^\top \boldsymbol{\Sigma}^{-1}(\boldsymbol{\mu}_1 - \boldsymbol{\mu}_2) = \text{const}}$$

This is linear projection of $\boldsymbol{x}$ onto the $\Sigma^{-1}(\mu_1 - \mu_2)$ direction.
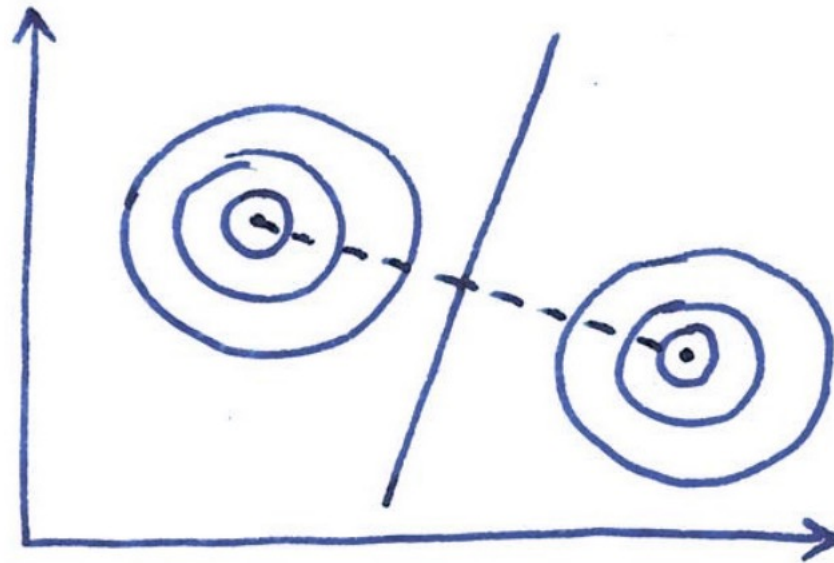
Where did we see this?

The role of Σ^{-1} in LDA

Why does LDA use projection on $\Sigma^{-1} (\mu_1 - \mu_2)$ and not simply on $(\mu_1 - \mu_2)$?

Nearest centroid classifier

Under additional assumption that the covariance matrix is spherical, $\Sigma = \sigma^2 \mathbf{I}$, LDA reduces to the *nearest centroid classifier*:

Quadratic Discriminant Analysis (QDA)

QDA: A classifier with a quadratic decision boundary, the covariance matrix, is not identical.

$$\delta_k(x) = x^T \boldsymbol{\Sigma}_{\boldsymbol{k}}^{-1} \mu_k - \frac{1}{2} \mu_k^T \boldsymbol{\Sigma}_{\boldsymbol{k}}^{-1} \mu_k + \log \pi_k$$

Logistic Regression versus LDA

For a two-class problem, one can show that for LDA

$$\log\left(\frac{p_1(x)}{1 - p_1(x)}\right) = \log\left(\frac{p_1(x)}{p_2(x)}\right) = c_0 + c_1 x_1 + \ldots + c_p x_p$$

So it has the same form as logistic regression.

The difference is in how the parameters are estimated.

- Logistic regression uses the conditional likelihood based on Pr*(Y |X)* (known as *discriminative learning*).
- LDA uses the full likelihood based on Pr*(X, Y)* (known as *generative learning*).
- Despite these differences, in practice the results are often very similar.

LDA for classification

- *When the classes are well-separated*, the parameter estimates for the logistic regression model are surprisingly **unstable**. Linear discriminant analysis does not suffer from this problem.

- *If n is small and the distribution of the predictors X is approximately normal in each of the classes*, the linear discriminant model is again more stable than the logistic regression.

- Linear discriminant analysis is popular when we have more than two classes, because it also provides low-dimensional views of the data.

Multi-class Classification

Multiclass Classification

- Multiclass Classification learning methods
  - Some binary classification methods can be directly extended to accommodate multiclass cases
  - Apply some strategies to solve multiclass classification problems with any existing binary classification methods （more general）
    - Decompose the problem and then train a binary classifier for each divided binary classification problem
    - Ensemble the outputs collected from all binary classifiers into the final multiclass predictions
- Dividing strategies
  - One vs. One (OvO)
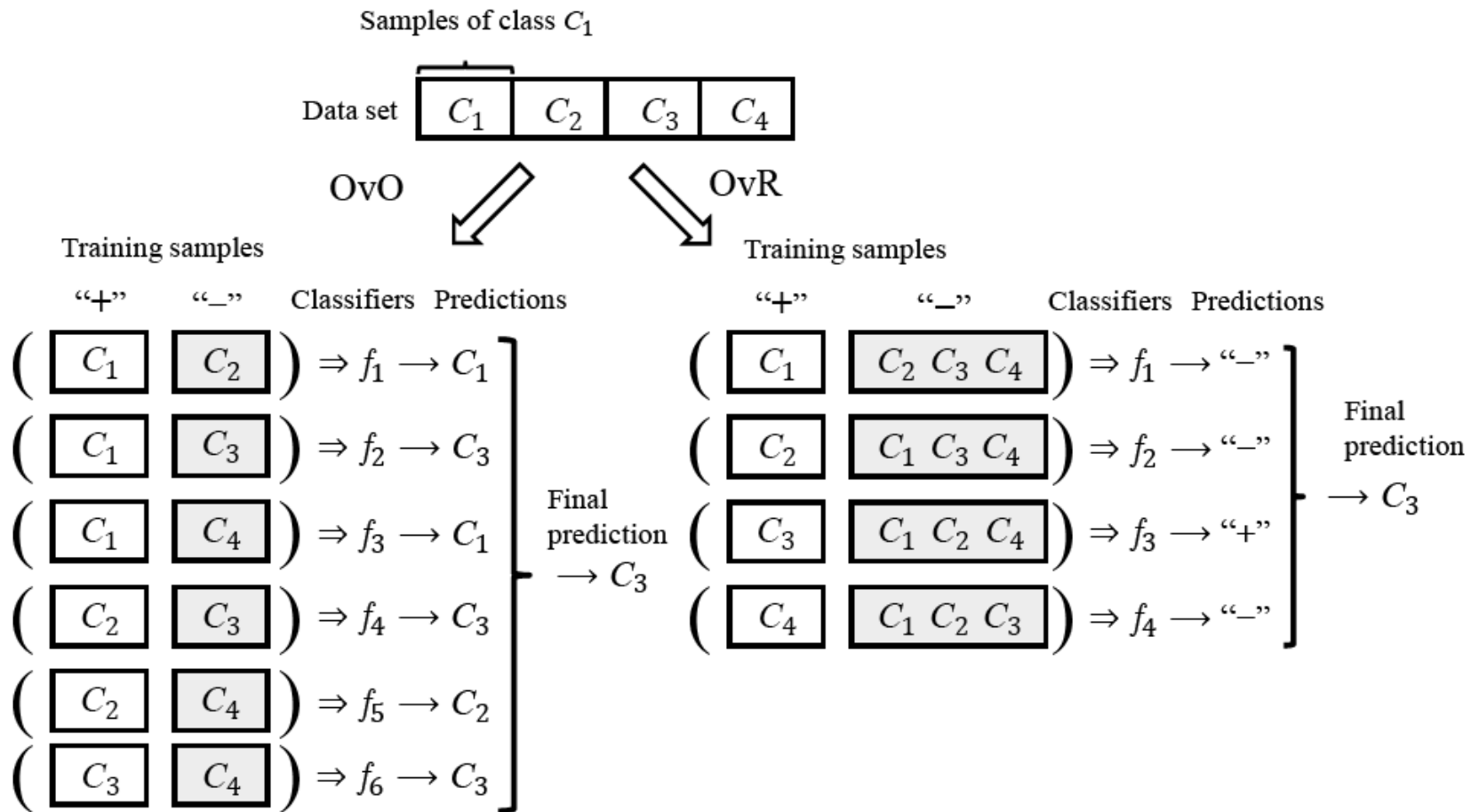 - One vs. Rest (OvR)
  - Many vs. Many (MvM)

Multiclass Classification - OvO

- In the decomposing phase
  - puts the N classes into pairs
    - $N(N-1)/2$ binary classification tasks
  - trains a classifier for each task
    - $N(N-1)/2$ classifiers
- In the testing phase
  - a new sample is classified by all classifiers
    - $N(N-1)/2$ classification outputs
  - the final prediction can be made via voting
    - the predicted class is the one received the most votes

Multiclass Classification - OvR

- ## In the decomposing phase
  - consider each class as positive in turn, and the rest classes are considered as negative
    - *N* binary classification tasks
  - trains a classifier for each task
    - *N* classifiers
- ## In the testing phase
  - a new sample is classified by all classifiers
    - *N* classification outputs
  - the prediction confidences are usually assessed
    - the class with the highest confidence is used as the classification result

Multiclass Classification – A comparison between OvO and OvR

Samples of class C_1

Data set: C_1 | C_2 | C_3 | C_4

OvO   OvR

OvO

Training samples

"+"   "−"   Classifiers   Predictions

$\left(C_1 \quad C_2 \right) \Rightarrow f_1 \rightarrow C_1$

$\left(C_1 \quad C_3 \right) \Rightarrow f_2 \rightarrow C_3$

$\left(C_1 \quad C_4 \right) \Rightarrow f_3 \rightarrow C_1$

$\left(C_2 \quad C_3 \right) \Rightarrow f_4 \rightarrow C_3$

$\left(C_2 \quad C_4 \right) \Rightarrow f_5 \rightarrow C_2$

$\left(C_3 \quad C_4 \right) \Rightarrow f_6 \rightarrow C_3$

Final prediction $\rightarrow C_3$

OvR

Training samples

"+"   "−"   Classifiers   Predictions

$\left(C_1 \quad C_2\ C_3\ C_4 \right) \Rightarrow f_1 \rightarrow \text{"−"}$

$\left(C_2 \quad C_1\ C_3\ C_4 \right) \Rightarrow f_2 \rightarrow \text{"−"}$

$\left(C_3 \quad C_1\ C_2\ C_4 \right) \Rightarrow f_3 \rightarrow \text{"+"}$

$\left(C_4 \quad C_1\ C_2\ C_3 \right) \Rightarrow f_4 \rightarrow \text{"−"}$

Final prediction $\rightarrow C_3$

Multiclass Classification – A comparison between OvO and OvR

OvO

- Train $N(N-1)/2$ classifiers，the memory and testing time costs are often higher

- Each classifier uses only samples of two classes. Hence, the computational cost of training OvO is lower
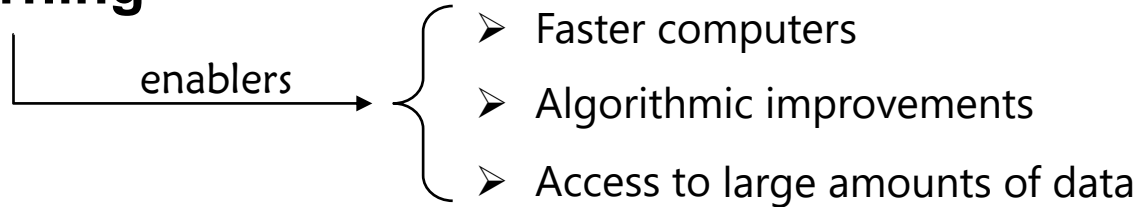
OvR

- Train N classifiers，the memory and testing time costs are often lower

- Each classifier uses all training samples. Hence, the computational cost of training OvO is higher

As for the prediction performance, it depends on the specific data distribution, and in most cases, the two methods have similar performance.
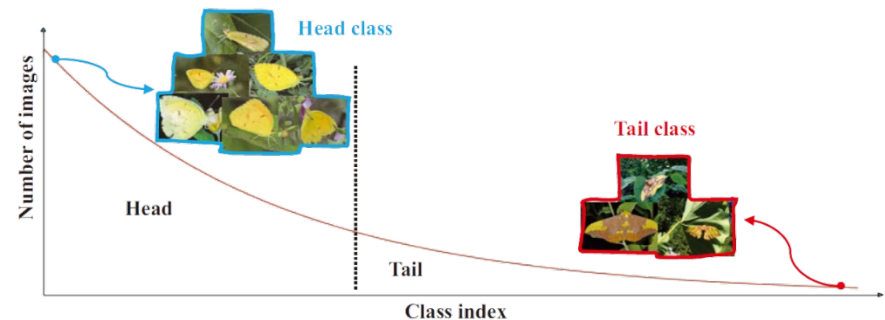
Recent Progress

Class Imbalance Problem

- **Deep Learning**

  enablers →
  - ➤ Faster computers
  - ➤ Algorithmic improvements
  - ➤ Access to large amounts of data

- **Real-world class-imbalanced (long-tailed) class distribution**
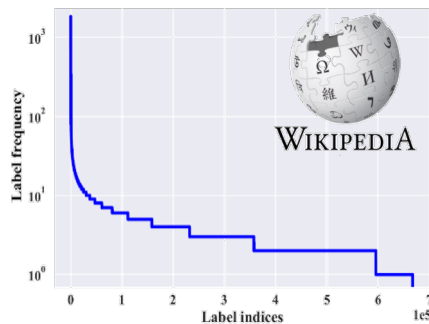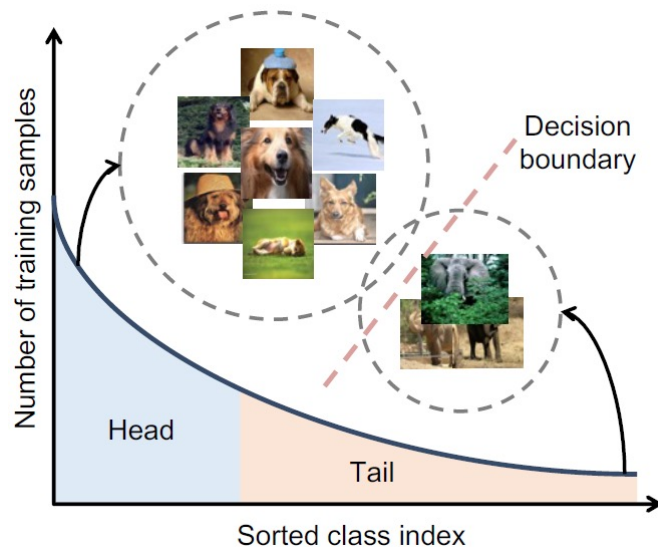

Zipf's law
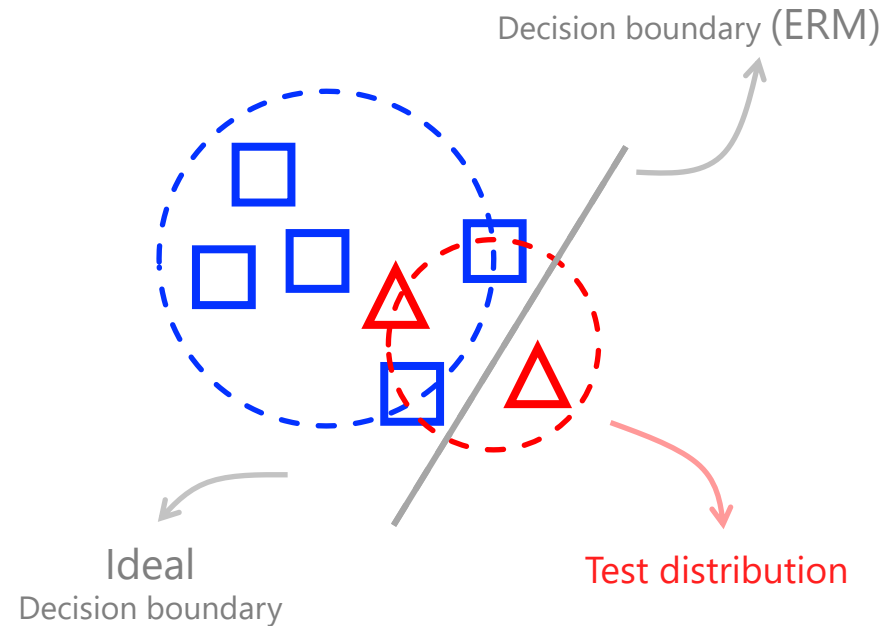
(Power law)

Class Imbalance Problem

Challenge

—— model biased towards head classes
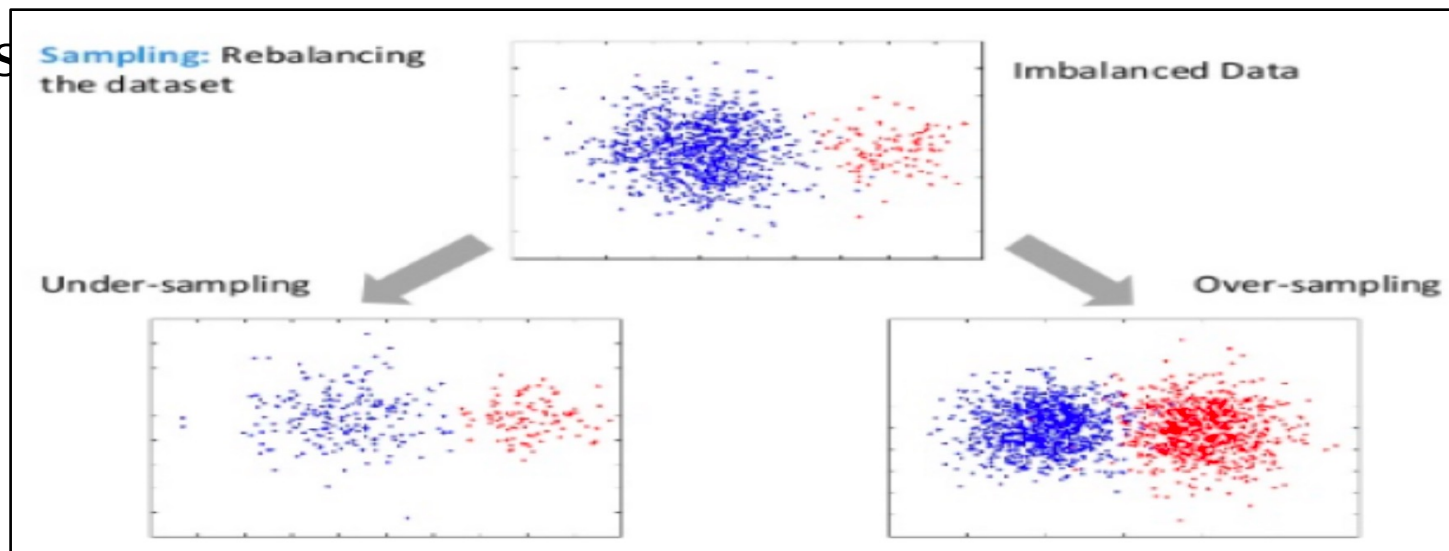

Take an example

Class Imbalance Problem

- ## Class imbalance
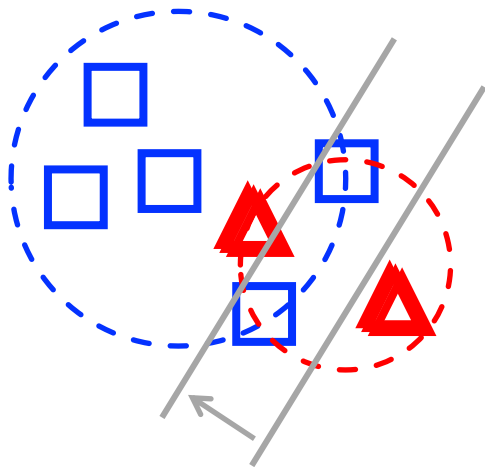 - ❑ a significantly different number of samples for each class.（the positive class is the minority）

    the classes
    are balanced
    $$\frac{y}{1-y} > 1$$
    ➡
    $$\frac{y}{1-y} > \frac{m^+}{m^-}$$
    the observed
    class ratio

- ## Res
  - ❑

    sses
  - ❑

    re
  - ❑

Sampling: Rebalancing the dataset

Imbalanced Data
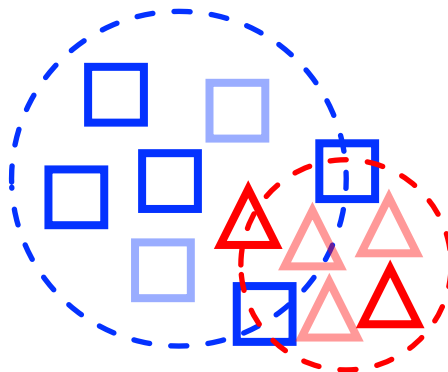
Under-sampling
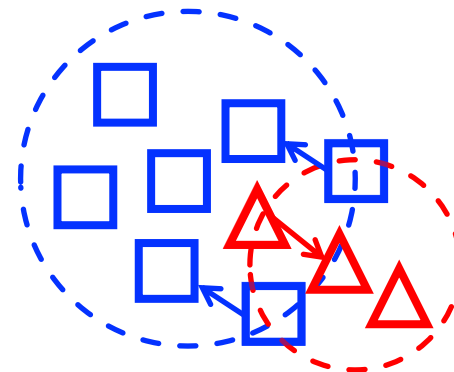
Over-sampling

Class Imbalance Problem

**Class
Re-balancing**

**Data
Augmentation**

**Class-balanced
Loss Function**

Extreme Classification

	Dataset	# of Train Points	# of Labels	# of Test Points
Benchmark*	**LF-AmazonTitles-131K**	**294,805**	**131,073**	**134,835**
	LF-WikiSeeAlsoTitles-320K	**693,082**	**312,330**	**177,515**
	LF-AmazonTitles-1.3M	**2,248,619**	**1,305,265**	**970,237**
Bing	**LF-P2PTitles-300K**	**1,366,429**	**300,000**	**585,602**
	LF-P2PTitles-2M	**2,539,009**	**1,640,898**	**1,088,146**

Label Tree

Linear separators trained inside root node

$\mathbf{w}_b$
$\mathbf{w}_c$
$\mathbf{w}_a$

● Label for **Child a**
■ Label for **Child b**
▲ Label for **Child c**

Branching factor: $K = 3$

Root node
node labels $S = \{1, \ldots, L\}$
label partitions S_a, S_b, S_c
linear separators $\mathbf{w}_a, \mathbf{w}_b, \mathbf{w}_c$

● Non-leaf node
■ Leaf node

Child a
node labels S_a

Child b
node labels S_b

Child c
node labels S_c
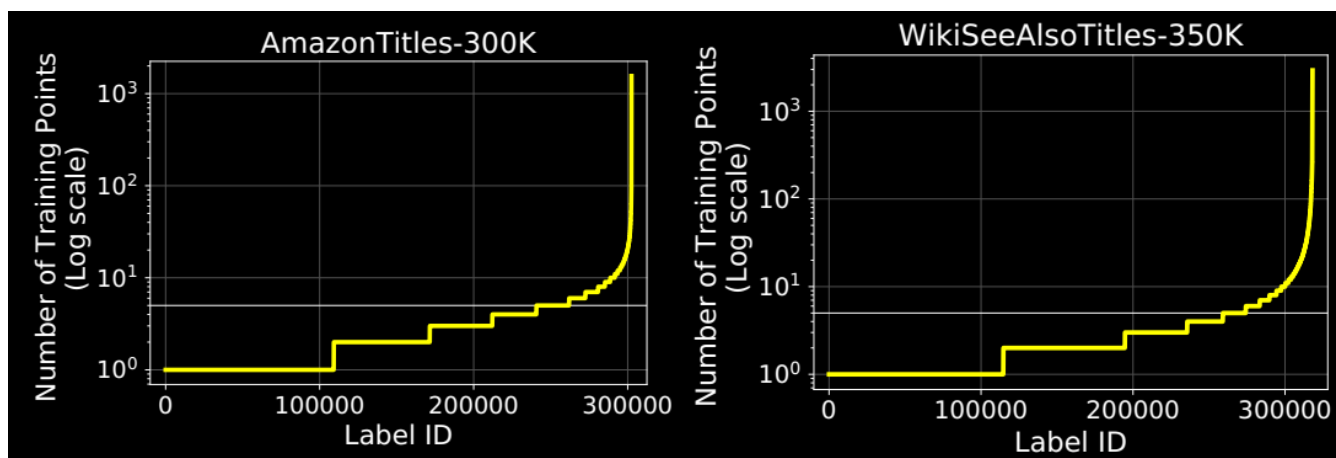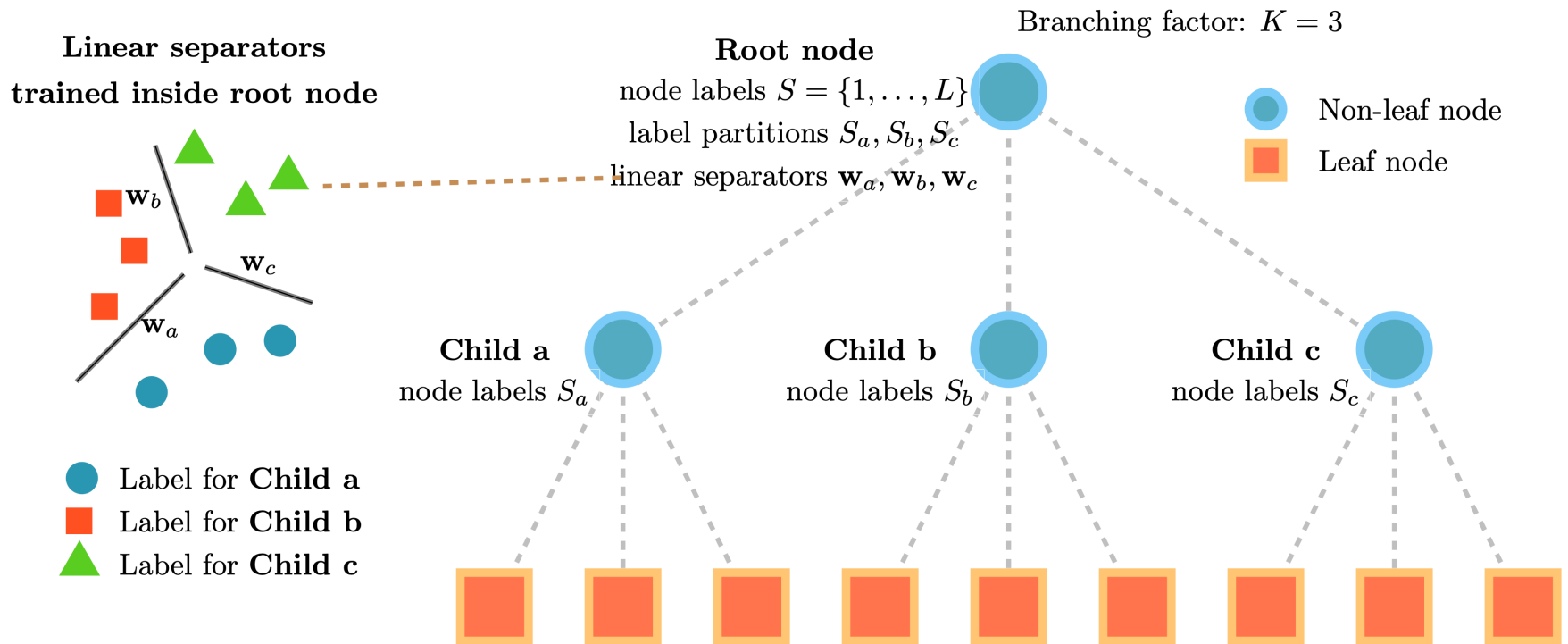
- recursively partition the set of classes into subsets
- train a multi-class (linear) classifier for inference