Bayes Classification

- Based on
  - Quantifying tradeoffs between various classification decisions.
- Uses conditional probabilities and the costs associated with those decisions.
- State of nature:
  - Let ω denote the *state of nature*.
  - We have different "states", say ω_1, ω_2, …, ω_c
  - We call these states "classes".
- For example, in the fish classification problem:

  ω_1 = "sea bass", ω_2 = "salmon"

- Let ω be a "random variable"

  with some probability distribution P(.)
- In the Bayesian context, P(.) is called *a priori probability*, where:
  - $P(\omega_1)$ is the prob. that next fish is a "sea bass"
  - $P(\omega_2)$ " " "salmon"
- Then: $P(\omega_1) + P(\omega_2) = 1$
- These probs tell us how likely is a particular type of fish to appear before it actually appears.
- What is the classification problem?

  Decide on a type of fish by means of a decision rule

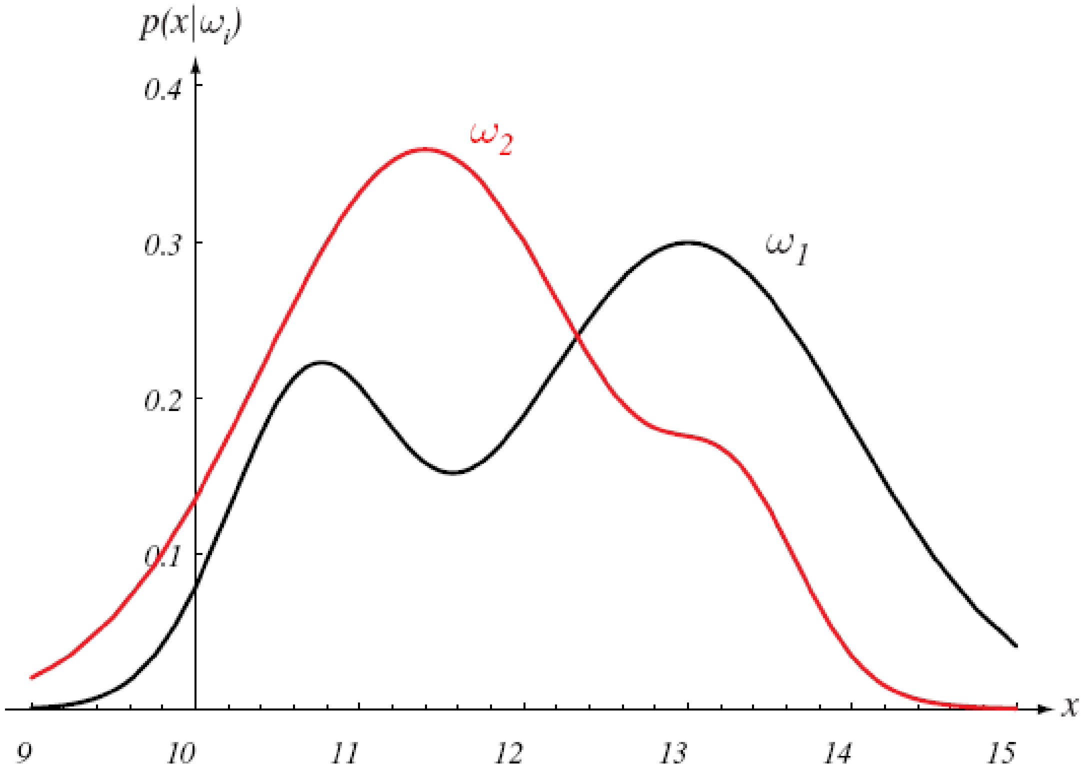
Bayes Classifier

- Suppose the only info we have is $P(\omega_1)$ and $P(\omega_2)$
- Decision rule:

  ω_1 if $P(\omega_1) > P(\omega_2)$

  ω_2 otherwise
- Then, what if $P(\omega_1) = P(\omega_2)$?
  - we have a 50-50 chance of being right!
  - or… our PR system has a 50% accuracy rate! (the worst case)
- Can we do better?
  - Use the *class-conditional information*…
  - By means of the *class-conditional* probability density function: $p(x \mid \omega)$

- Say, what is the prob of *x* given it becomes from a certain class?
- If we use the *lightness* of the fish:



- This *a posteriori* information be used in our decision rule
- For class ω_j:

$$p(\omega_j, x) = P(\omega_j|x)\, p(x) = p(x|\omega_j)\, P(\omega_j)$$

- Rearranging:

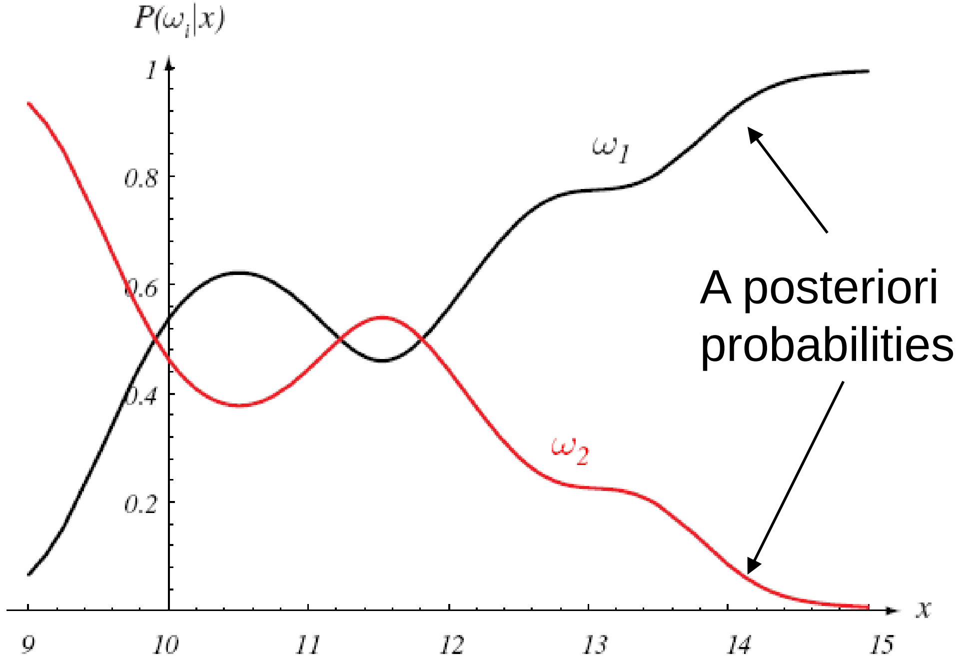
$$P(\omega_j \mid x) = \frac{p(x \mid \omega_j) P(\omega_j)}{p(x)}$$

- where

$$p(x) = \sum_{j=1}^{c} p(x \mid \omega_j) P(\omega_j)$$

- This is *Bayes formula*!
- Go from $P(\omega_j)$ to *a posteriori* prob $P(\omega_j|x)$

- $p(x|\omega_j)$ is called the *likelihood* of ω_j wrt x
- $p(x)$ is an *evidence* factor that only "scales" the final result.
- Example: Lightness, $P(\omega_1)$ = 2/3 and $P(\omega_2)$ = 1/3



A posteriori probabilities

Interpretation of Bayes theorem

- What is the decision rule then?
- Consider prob. of error:

$$P(error \mid x) = \begin{cases} P(\omega_1 \mid x) & \text{if we decide } \omega_2 \\ P(\omega_2 \mid x) & \text{if we decide } \omega_1 \end{cases}$$

- How can we minimize it?

- **Bayes decision rule:**

  Decide ω_1 if $P(\omega_1|x) > P(\omega_2|x)$
  ω_2 otherwise

- Under this rule, the error becomes:

$$P(error \mid x) = \min\{P(\omega_1 \mid x), P(\omega_2 \mid x)\}$$

- Rule expressed in terms of conditional and *a priori* probs
- The evidence, $p(x)$, is unimportant and omitted

- Thus, the decision rule is re-written:

  Decide ω_1 if $p(x|\omega_1)\, P(\omega_1) > p(x|\omega_2)\, P(\omega_2)$
  ω_2 otherwise

- $p(x|\omega_1) = p(x|\omega_2)$ gives us no information about the class
- $P(\omega_1) = P(\omega_2)$ means the classes are equally likely
- In that case, decision based entirely on $p(x|\omega_j)$
- Rule can be generalized to 3+ classes and incorporate risk

Feature space:

- Instead of simply using a scalar x, each object represented by a vector:

  $\mathbf{x} = [x_1\ x_2\ \ldots\ x_d]^t \in \Re^d$

- $\Re^d$ is the d-dimensional Euclidean space… called the *feature* space

Risk - Loss function

- Based on prob theory + utility theory
- How costly is an action?
- Let $\{\omega_1, \omega_2, \ldots, \omega_c\}$ be a set of classes, $\{\alpha_1, \alpha_2, \ldots, \alpha_a\}$ be a set of possible actions
- Loss function: $\lambda(\alpha_i, \omega_j)$
  - loss incurred for taking action α_i when the state of nature is ω_j
- Bayes formula:

$$P(\omega_j \mid \mathbf{x}) = \frac{p(\mathbf{x} \mid \omega_j) P(\omega_j)}{p(\mathbf{x})}$$

- Given **x**, taking an action implies a risk:

$$R(\alpha_i \mid \mathbf{x}) = \sum_{j=1}^{c} \lambda(\alpha_i \mid \omega_j) P(\omega_j \mid \mathbf{x})$$

- Called the *conditional risk*

- Aim: minimize the *expected* loss by taking the action that *minimizes* the risk

Decision rule:

- A function $\alpha(\mathbf{x})$ that tells us which action to take when **x** is given
- The Bayes decision rule minimizes the overall risk:

$$R = \int_{\mathbf{x}} R(\alpha(\mathbf{x}) \mid \mathbf{x}) \, p(\mathbf{x}) \, d\mathbf{x}$$

- Compute the *conditional risk*:

$$R(\alpha_i \mid \mathbf{x}) = \sum_{j=1}^{c} \lambda(\alpha_i \mid \omega_j) P(\omega_j \mid \mathbf{x})$$

  for every value of $i = 1, \ldots, a$
- *The resulting minimum risk is called the Bayes risk*

Two-class Classification

- Two classes: ω_1 and ω_2

  and two actions: α_1 and α_2
- Notation: $\lambda_{ij} = \lambda(\alpha_i|\omega_j)$ is the loss function
- The conditional risk is given by:

$$R(\alpha_1 \,|\, \mathbf{x}) = \lambda_{11} P(\omega_1 \,|\, \mathbf{x}) + \lambda_{12} P(\omega_2 \,|\, \mathbf{x})$$

$$R(\alpha_2 \,|\, \mathbf{x}) = \lambda_{21} P(\omega_1 \,|\, \mathbf{x}) + \lambda_{22} P(\omega_2 \,|\, \mathbf{x})$$

- Aim: decide ω_1 if

$$(\lambda_{21} - \lambda_{11}) P(\omega_1 \,|\, \mathbf{x}) > (\lambda_{12} - \lambda_{22}) P(\omega_2 \,|\, \mathbf{x})$$

- In practice:

  loss after making a mistake **>** loss after being correct
- Meaning that $\lambda_{21} - \lambda_{11}$ and $\lambda_{12} - \lambda_{22}$ **>** 0

- Using Bayes rule, decide ω_1 if

$$(\lambda_{21} - \lambda_{11})\, p(\mathbf{x} \,|\, \omega_1) P(\omega_1) >$$
$$(\lambda_{12} - \lambda_{22})\, p(\mathbf{x} \,|\, \omega_2) P(\omega_2)$$

- Alternatively, decide ω_1 if

$$\frac{p(\mathbf{x} \,|\, \omega_1)}{p(\mathbf{x} \,|\, \omega_2)} > \frac{(\lambda_{12} - \lambda_{22})}{(\lambda_{21} - \lambda_{11})} \frac{P(\omega_2)}{P(\omega_1)}$$

- where the LHS is called

  the *likelihood ratio*

Example: Spam filter (email classification)

2 classes

$$\omega_1 = \text{spam}$$

$$\omega_2 = \text{ham}$$

ham = good

2 actions

$$\alpha_1 = \text{delete}$$

$$\alpha_2 = \text{keep}$$

$$0 = \lambda_{11} = \lambda(\alpha_1, \omega_1) \equiv \text{Email is spam - delete it}$$

$$10 = \lambda_{12} = \lambda(\alpha_1, \omega_2) \equiv \text{Email is ham - delete it}$$

$$5 = \lambda_{21} = \lambda(\alpha_2, \omega_1) \equiv \text{Email is spam - keep it}$$

$$0 = \lambda_{22} = \lambda(\alpha_2, \omega_2) \equiv \text{Email is ham - keep it}$$

- Bayes rule that minimizes the risk:

$$\frac{p(\mathbf{x} \mid \omega_1)}{p(\mathbf{x} \mid \omega_2)} > \frac{(10-0)}{(5-0)} \frac{P(\omega_2)}{P(\omega_1)} = 2.0 \frac{P(\omega_2)}{P(\omega_1)}$$

- Suppose that:

$$P(\omega_1) = 0.9$$

$$P(\omega_2) = 0.1$$

- Then, decide ω_1 if

$$\frac{p(\mathbf{x} \mid \omega_1)}{p(\mathbf{x} \mid \omega_2)} > 2.0 \frac{0.1}{0.9} \approx 0.22$$

Minimum-error-rate Classification

- Two *loss* values: “correct” and “incorrect”
- *Symmetrical* or 0-1 loss function

$$\lambda(\alpha_i \mid \omega_j) = \begin{cases} 0 & \text{if } i = j \\ 1 & \text{if } i \neq j \end{cases} \qquad i, j = 1, \ldots, c$$

- The conditional risk:

$$R(\alpha_i \mid \mathbf{x}) = \sum_{j=1}^{c} \lambda(\alpha_i \mid \omega_j) P(\omega_j \mid \mathbf{x})$$

$$= \sum_{j \neq i} P(\omega_j \mid \mathbf{x})$$

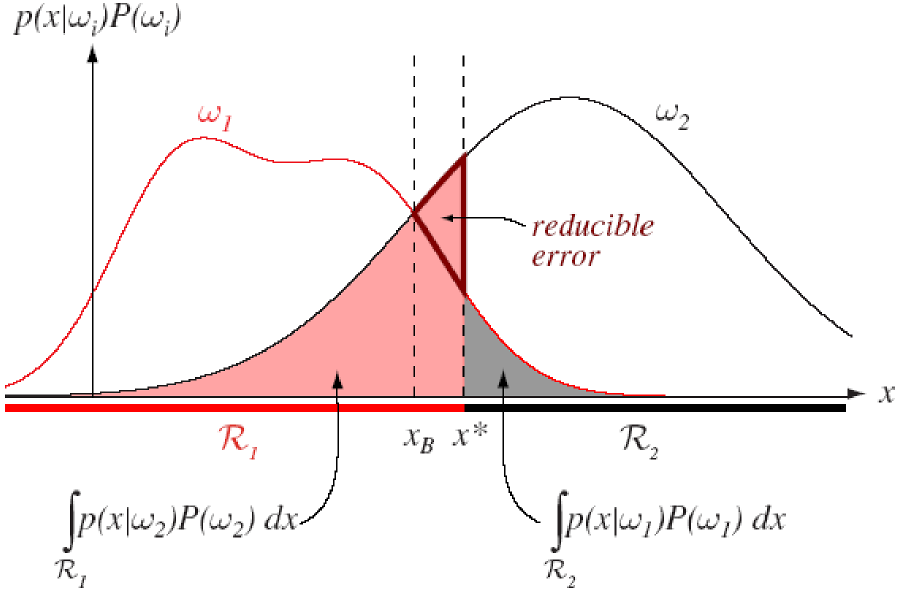
$$= 1 - P(\omega_i \mid \mathbf{x})$$

- *minimizing* the risk is equivalent to *maximizing* $P(\omega_i \mid \mathbf{x})$
- General rule for minimizing the error:
  - Decide ω_i if $P(\omega_i|\mathbf{x}) > P(\omega_j|\mathbf{x})$ for all $j \neq i$.

2 classes: Decide

ω_1 if $P(\omega_1|x) > P(\omega_2|x)$

ω_2 otherwise



Example

$$\lambda(\alpha_1,\omega_1)=\lambda(\alpha_2,\omega_2)=0$$
$$\lambda(\alpha_1,\omega_2)=\lambda(\alpha_2,\omega_1)=1$$

We don't get penalized for being correct!

$$\frac{p(\mathbf{x}\,|\,\omega_1)}{p(\mathbf{x}\,|\,\omega_2)} > \frac{(1-0)}{(1-0)}\,\frac{P(\omega_2)}{P(\omega_1)} = \frac{P(\omega_2)}{P(\omega_1)}$$

$$\equiv p(\mathbf{x}\,|\,\omega_1)P(\omega_1) > p(\mathbf{x}\,|\,\omega_2)P(\omega_2)$$

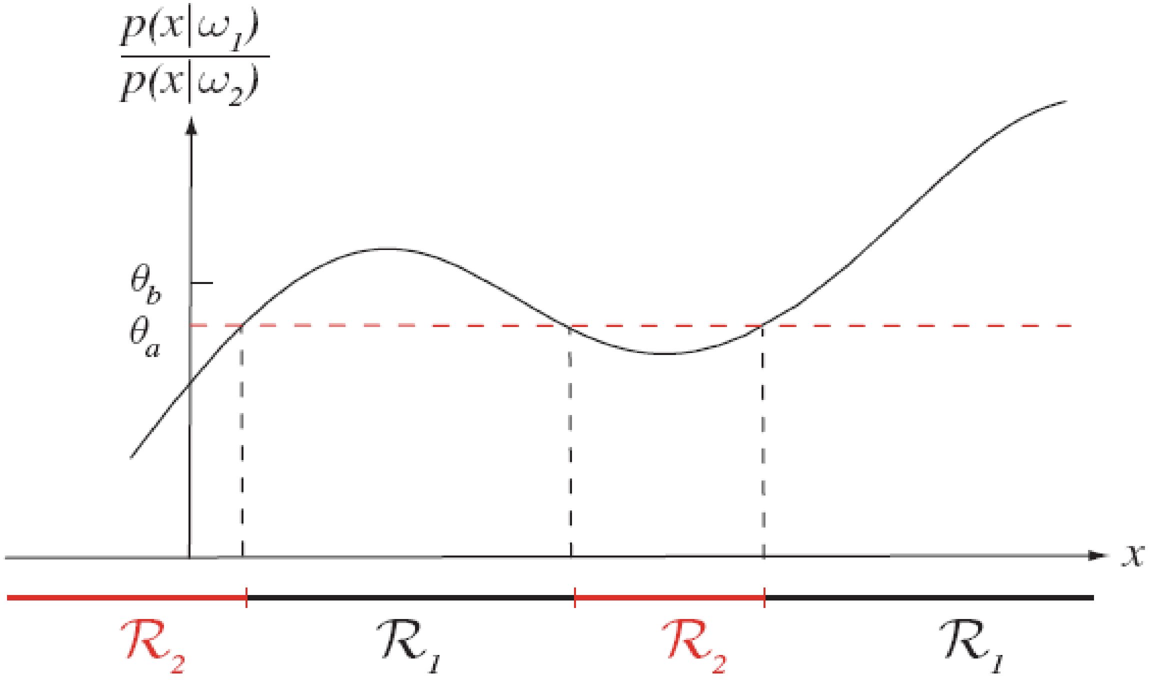
or

$$\frac{p(\mathbf{x}\,|\,\omega_1)}{p(\mathbf{x}\,|\,\omega_2)} > \frac{P(\omega_2)}{P(\omega_1)}$$

Suppose $P(\omega_1)=P(\omega_2)$

$$\theta_a = \frac{P(\omega_2)}{P(\omega_1)} = 1, \text{ since } P(\omega_1)=P(\omega_2)$$

$$\frac{p(\mathbf{x}\,|\,\omega_1)}{p(\mathbf{x}\,|\,\omega_2)} > 1 = \theta_a$$



Classifiers: Multi-class case

- Common representation: in terms of *discriminant functions* $g_i(\mathbf{x})$
- Assign **x** to class ω_i if

$$g_i(\mathbf{x}) > g_j(\mathbf{x}) \qquad \text{for all } j \neq i$$

- If we let $g_i(\mathbf{x}) = -R(\alpha_i \mid \mathbf{x})$, the **maximum** discriminant corresponds to the one that **minimizes** the conditional risk
- For the minimum error rate:

$$g_i(\mathbf{x}) = P(\omega_i \mid \mathbf{x})$$

- Different representations:

$$g_i(\mathbf{x}) = P(\omega_i \mid \mathbf{x}) = \frac{p(\mathbf{x} \mid \omega_i) P(\omega_i)}{p(\mathbf{x})}$$

$$g_i(\mathbf{x}) = p(\mathbf{x} \mid \omega_i) P(\omega_i)$$

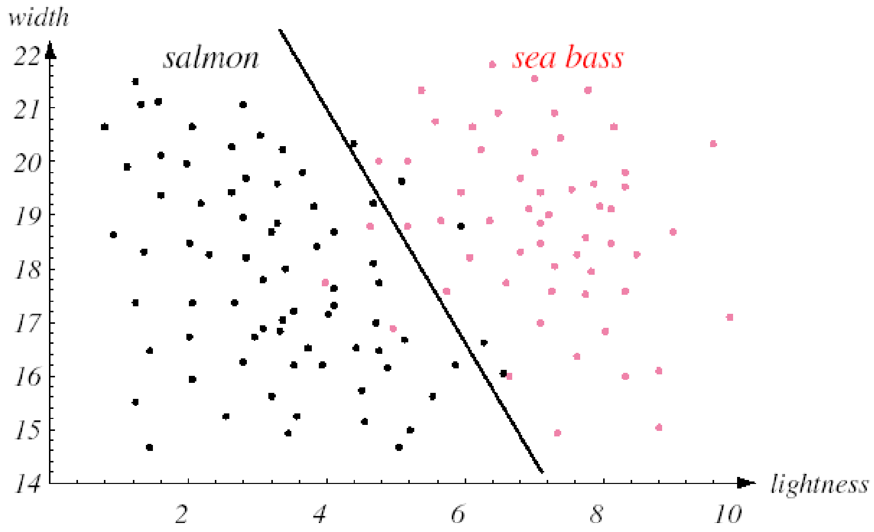
$$g_i(\mathbf{x}) = \ln p(\mathbf{x} \mid \omega_i) + \ln P(\omega_i)$$

- All equivalent
- Divide the feature space into *c* regions
- separated by *decision boundaries*
- 2 classes:
  - A discriminant function (also called *dichotomizer*): $g(\mathbf{x})$
- Decide ω_1 if $g(\mathbf{x}) = g_1(\mathbf{x}) - g_2(\mathbf{x}) > 0$

Dichotomizer

- A machine that:
  - computes the sign of the result of $g(\mathbf{x})$
  - assigns a class depending on that sign
- Many forms for writing $g(\mathbf{x})$
- Two of them are: $g(\mathbf{x}) = P(\omega_1 | \mathbf{x}) - P(\omega_2 | \mathbf{x})$

$$g(\mathbf{x}) = \ln \frac{p(\mathbf{x} | \omega_1)}{p(\mathbf{x} | \omega_2)} + \ln \frac{P(\omega_1)}{P(\omega_2)}$$



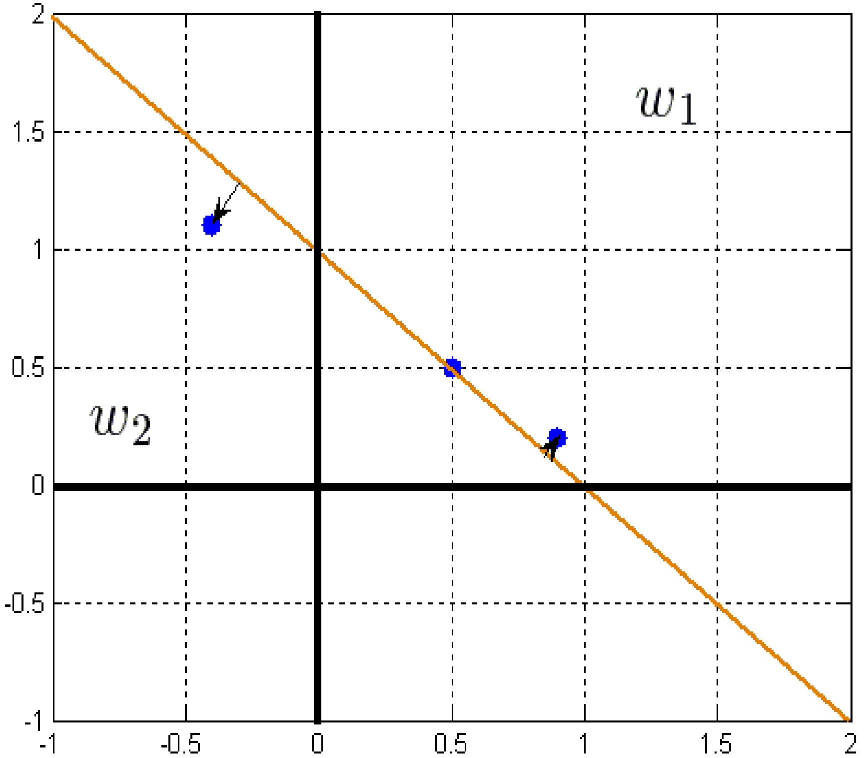
Example

Linear discriminant functions:

$$g_1(\mathbf{x}) = \begin{bmatrix} 2 & 2 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} + 1$$

$$g_2(\mathbf{x}) = \begin{bmatrix} 1 & 1 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} + 2$$

$$g(\mathbf{x}) = g_1(\mathbf{x}) - g_2(\mathbf{x}) = \begin{bmatrix} 1 & 1 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} - 1$$



Rule:

decide ω_1 if $g(\mathbf{x}) > 0$

Classify $\mathbf{x} = \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} = \begin{bmatrix} 0.5 \\ 0.5 \end{bmatrix}$

$$g(\mathbf{x}) = \begin{bmatrix} 1 & 1 \end{bmatrix} \begin{bmatrix} 0.5 \\ 0.5 \end{bmatrix} - 1 = 0.5 + 0.5 - 1 = 0$$

decide arbitrarily

Classify $\mathbf{x} = \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} = \begin{bmatrix} 0.9 \\ 0.2 \end{bmatrix}$

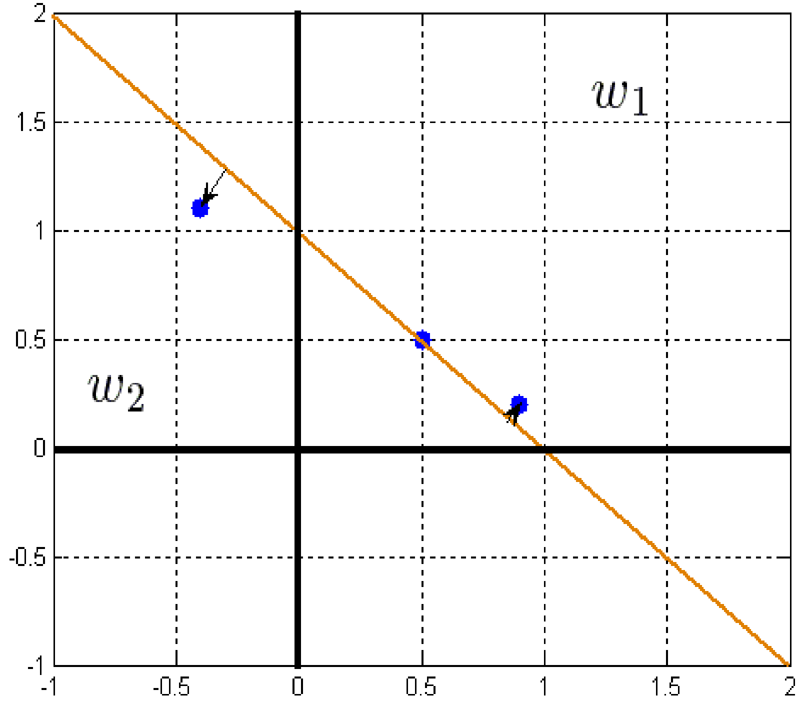
$$g(\mathbf{x}) = \begin{bmatrix} 1 & 1 \end{bmatrix} \begin{bmatrix} 0.9 \\ 0.2 \end{bmatrix} - 1 = 0.9 + 0.2 - 1 = 0.1 > 0$$

(decide ω_1)

Classify $\mathbf{x} = \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} = \begin{bmatrix} -0.4 \\ 1.1 \end{bmatrix}$

$$g(\mathbf{x}) = \begin{bmatrix} 1 & 1 \end{bmatrix} \begin{bmatrix} -0.4 \\ 1.1 \end{bmatrix} - 1 = -0.4 + 1.1 - 1 = -0.3 < 0$$

(decide ω_2)



Example: Quadratic classifier

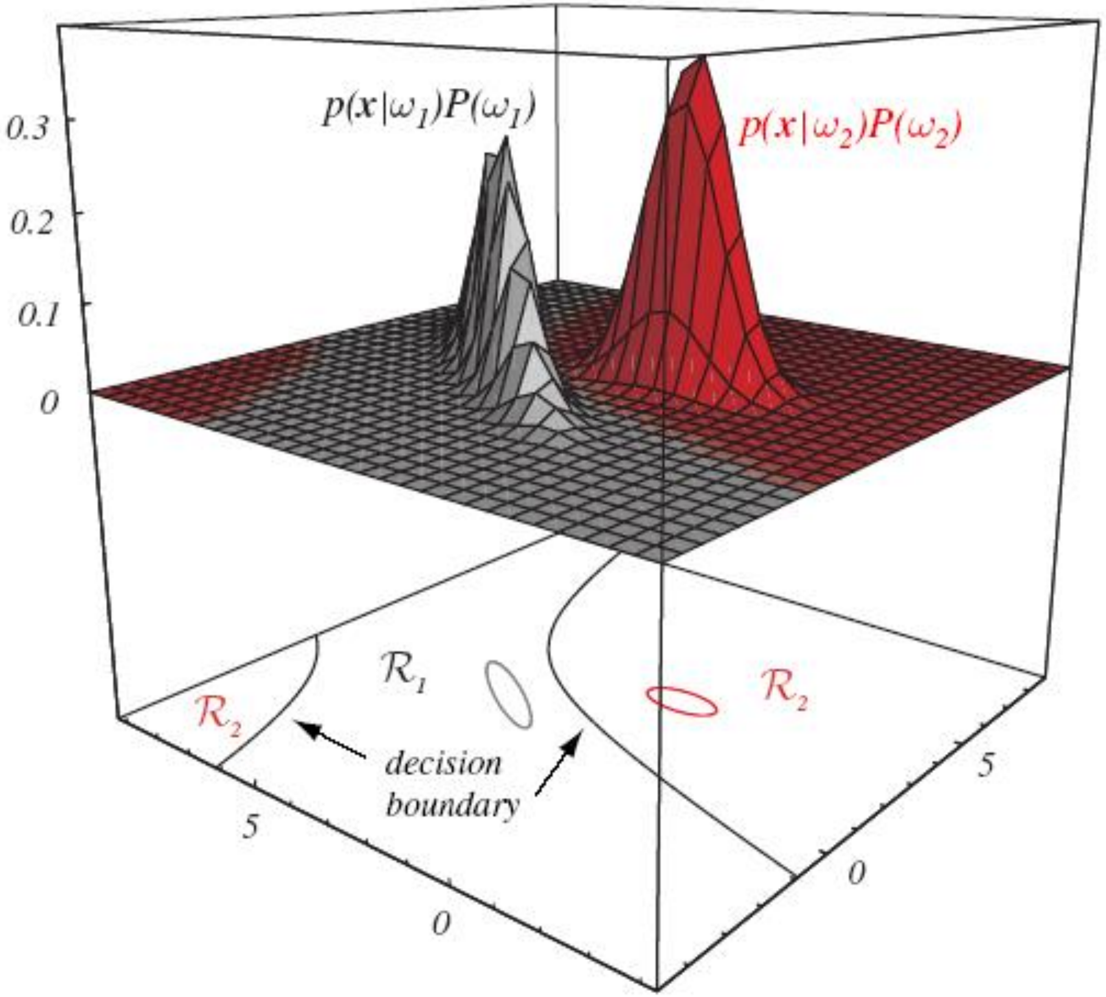


Figure from Duda et al.

The Normal Distribution

Justification:

- Two parameters (moments) *fully* specify the distribution
- Uncorrelated **iff** independent
- Normal *marginal* densities and normal *conditional* densities
- Normal characteristic function
- Linear transformation ⇒ *normal* distribution
- Physical justification: The central limit theorem
- Maximum entropy distribution

$$p(\mathbf{x} \mid \omega_i) = \frac{1}{(2\pi)^{\frac{d}{2}} |\Sigma_i|^{\frac{1}{2}}} e^{-\frac{1}{2}(\mathbf{x}-\mu_i)^t \Sigma_i^{-1} (\mathbf{x}-\mu_i)}$$

Discriminant function

$$p(\mathbf{x} \mid \omega_i) = \frac{1}{(2\pi)^{\frac{d}{2}} |\Sigma_i|^{\frac{1}{2}}} e^{-\frac{1}{2}(\mathbf{x}-\mu_i)^t \Sigma_i^{-1} (\mathbf{x}-\mu_i)}$$

- Minimum error-rate achieved by:

  choose max of:

$$g_i(\mathbf{x}) = \ln p(\mathbf{x} \mid \omega_i) + \ln P(\omega_i)$$

- Now, $p(\mathbf{x} \mid \omega_i)$ known: multivariate normal…

  That is, $p(\mathbf{x} \mid \omega_i) \sim N(\boldsymbol{\mu}_i, \boldsymbol{\Sigma}_i)$
- Thus, the discriminant function results in:

$$g_i(\mathbf{x}) = -\frac{1}{2}(\mathbf{x} - \boldsymbol{\mu}_i)^t \boldsymbol{\Sigma}_i^{-1} (\mathbf{x} - \boldsymbol{\mu}_i) - \frac{d}{2} \ln 2\pi - \frac{1}{2} \ln |\boldsymbol{\Sigma}_i| + \ln P(\omega_i)$$

- Two cases…

Case 1: $\Sigma_i = \sigma^2$ I

- Each feature has the same variance, σ^2
- The features are statistically independent (uncorrelated)
- The samples fall in equal-size hyperspherical clusters
- Points with the same prob. are in a *hypersphere*.
- The classification is based on the *means only*
- The discriminant function results in:

$$g_i(\mathbf{x}) = -\frac{(\mathbf{x}-\boldsymbol{\mu}_i)^t(\mathbf{x}-\boldsymbol{\mu}_i)}{2\sigma^2} + \ln P(\omega_i)$$

- Expanding the quadratic term:

$$g_i(\mathbf{x}) = -\frac{1}{2\sigma^2}[\mathbf{x}^t\mathbf{x} - 2\boldsymbol{\mu}_i^t\mathbf{x} + \boldsymbol{\mu}_i{}^t\boldsymbol{\mu}_i] + \ln P(\omega_i)$$

- Looks like it is a quadratic function on **x**, but…
- $\mathbf{x}^t\,\mathbf{x}$ is the same for all *i*, and is omitted, obtaining:

$$g_i(\mathbf{x}) = \mathbf{w}_i{}^t\mathbf{x} + w_{i0}$$

- where (the *direction* of the hyperplane)

$$\mathbf{w}_i = \frac{1}{\sigma^2}\boldsymbol{\mu}_i$$

- and (the *threshold* or *bias* of the classifier)

$$w_{i0} = -\frac{1}{2\sigma^2}\boldsymbol{\mu}_i{}^t\boldsymbol{\mu}_i + \ln P(\omega_i)$$

- When dealing with *two* classes:

$$g_1(\mathbf{x}) = \frac{1}{\sigma^2}\boldsymbol{\mu}_1^t\mathbf{x} - \frac{1}{2\sigma^2}\boldsymbol{\mu}_1^t\boldsymbol{\mu}_1 + \ln P(\omega_1)$$

$$g_2(\mathbf{x}) = \frac{1}{\sigma^2}\boldsymbol{\mu}_2^t\mathbf{x} - \frac{1}{2\sigma^2}\boldsymbol{\mu}_2^t\boldsymbol{\mu}_2 + \ln P(\omega_2)$$

$$g(\mathbf{x}) = g_1(\mathbf{x}) - g_2(\mathbf{x}) = \frac{1}{\sigma^2}(\boldsymbol{\mu}_1 - \boldsymbol{\mu}_2)^{\mathbf{t}}\mathbf{x} + \frac{1}{2\sigma^2}(\boldsymbol{\mu}_2^t\boldsymbol{\mu}_2 - \boldsymbol{\mu}_1^t\boldsymbol{\mu}_1) + \ln\frac{P(\omega_1)}{P(\omega_2)}$$

$$\text{Decide } \omega_1 \text{ if } \mathrm{g}(\mathbf{x}) > 0, \text{else decide } \omega_2 \text{ when } \mathrm{g}(\mathbf{x}) \leq 0$$

Ties are decided arbitrarily

Example – 2D

$$\boldsymbol{\mu}_1 = \begin{bmatrix} 3 \\ 2 \end{bmatrix} \qquad \boldsymbol{\mu}_2 = \begin{bmatrix} 1 \\ 1 \end{bmatrix} \qquad \sigma^2 = 1$$

$$P(\omega_1) = P(\omega_2) = 0.5$$

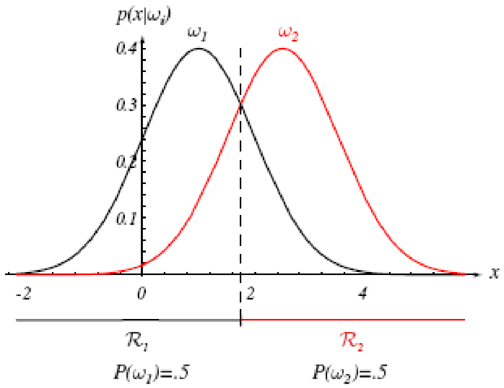
$$g(\mathbf{x}) = \begin{bmatrix} 2 \\ 1 \end{bmatrix}^t \mathbf{x} + \frac{1}{2}(2 - 13) = \begin{bmatrix} 1 & 2 \end{bmatrix} \mathbf{x} - \frac{11}{2}$$

Classify $\mathbf{x} = \begin{bmatrix} 4 \\ 2 \end{bmatrix}$

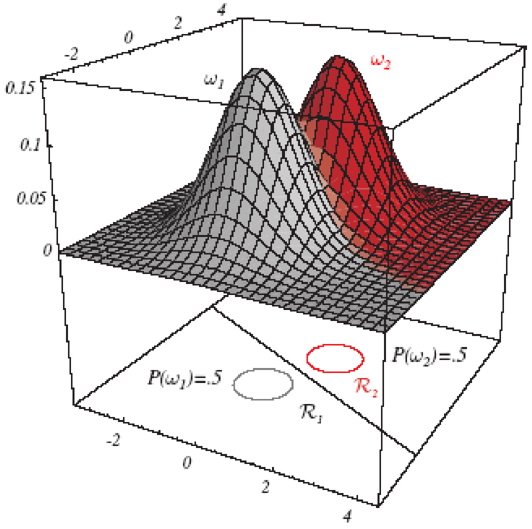
$$g(\mathbf{x}) = \begin{bmatrix} 1 & 2 \end{bmatrix} \begin{bmatrix} 4 \\ 2 \end{bmatrix} - \frac{11}{2} = 8 - \frac{11}{2} = \frac{5}{2} = 2.5 > 0$$

(decide ω_1)

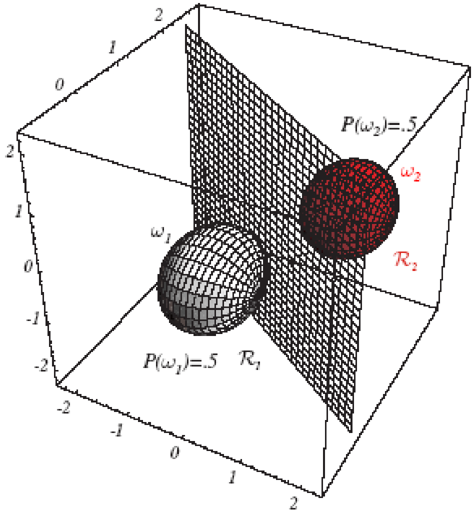
- Examples for different dimensions:



1D



2D



3D

Figure from Duda et al.

- What if $P(\omega_i)$ changes …
- 1D:

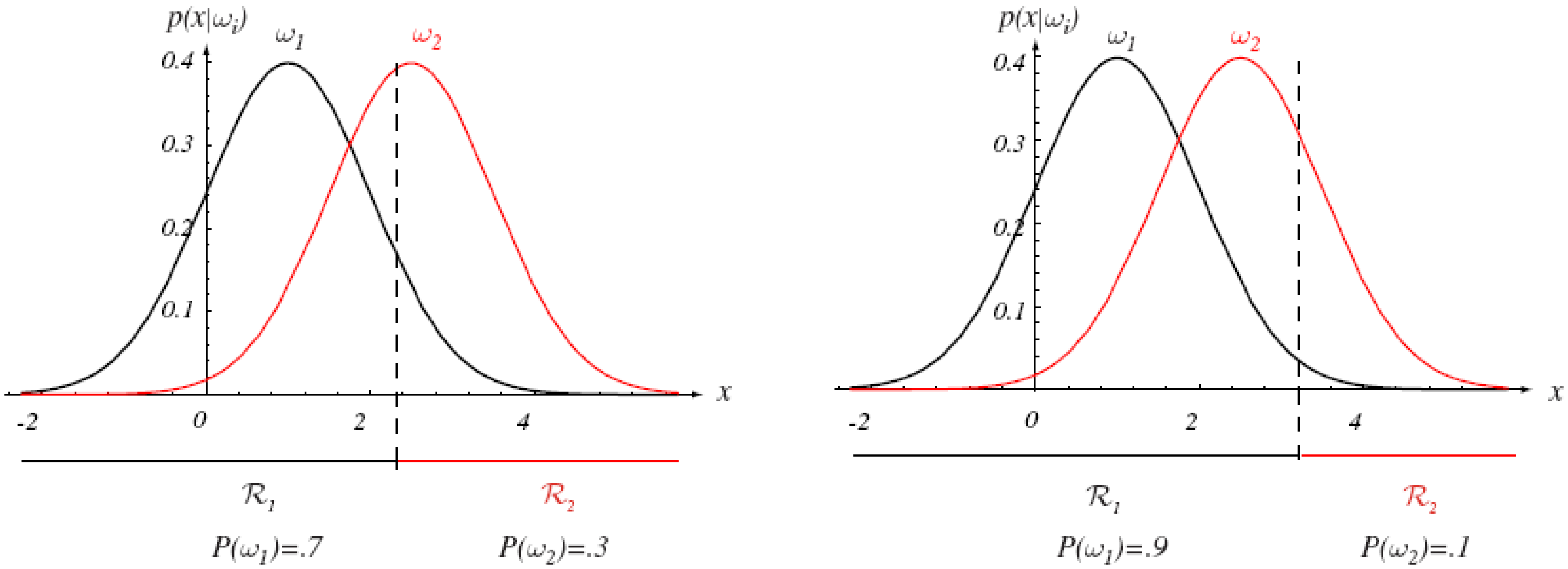


Figure from Duda et al.

- 2D:

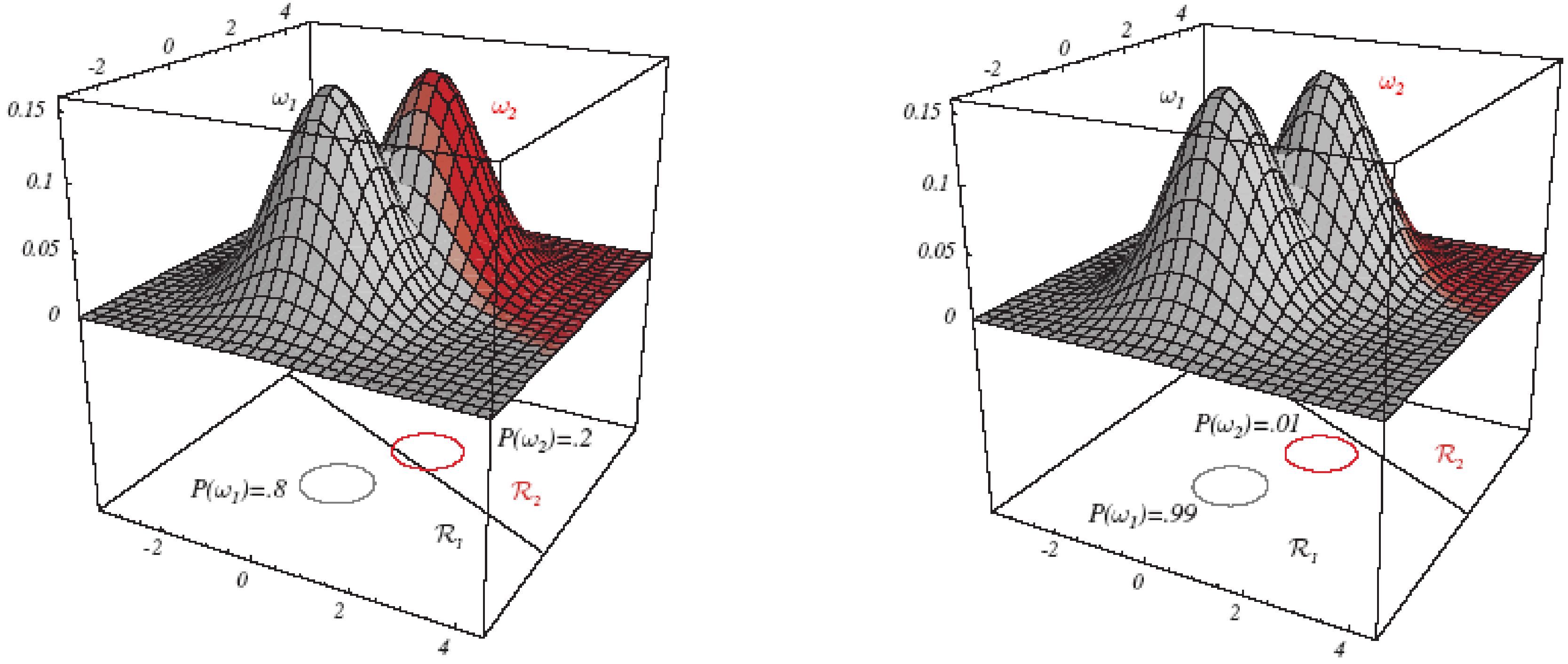


Figure from Duda et al.

- 3D:

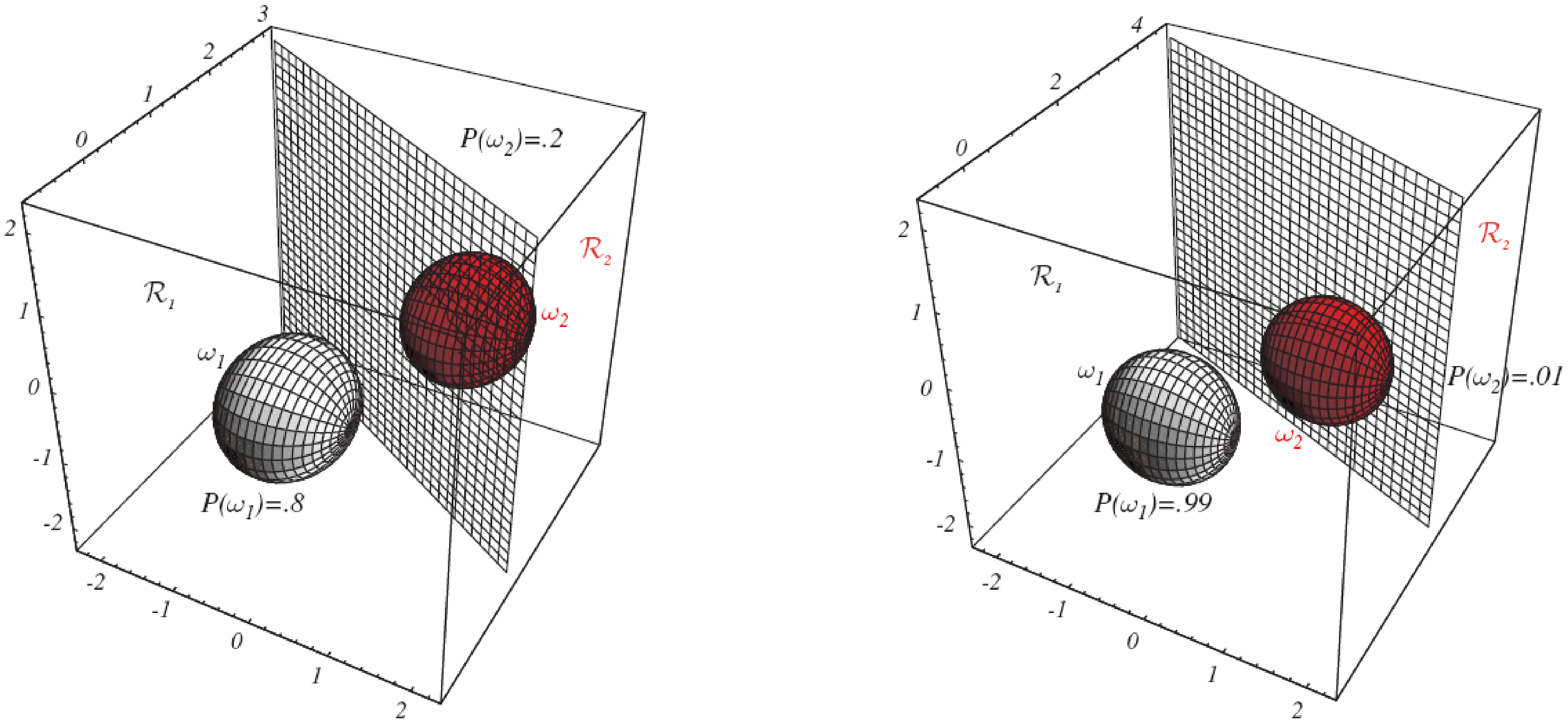


Figure from Duda et al.

Case 2: Σ_i arbitrary

- The covariance matrices are different for all classes
- The only term that can be cancelled out is $\frac{d}{2}\ln 2\pi$
- The resulting classifier is a **quadratic** function:

$$g_i(\mathbf{x}) = \mathbf{x}^t \mathbf{W}_i \mathbf{x} + \mathbf{w}_i{}^t \mathbf{x} + w_{i0}$$

where

$$\mathbf{W}_i = -\frac{1}{2}\mathbf{\Sigma}_i^{-1} \quad , \quad \mathbf{w}_i = \mathbf{\Sigma}_i^{-1}\boldsymbol{\mu}_i$$

and

$$w_{i0} = -\frac{1}{2}\boldsymbol{\mu}_i{}^t \mathbf{\Sigma}_i^{-1}\boldsymbol{\mu}_i - \frac{1}{2}\ln|\mathbf{\Sigma}_i| + \ln P(\omega_i)$$

For two classes

- Decide ω_1 if $g_1(\mathbf{x}) > g_2(\mathbf{x})$ or $g(\mathbf{x}) = g_1(\mathbf{x}) - g_2(\mathbf{x}) > 0$
  ω_2 otherwise
- Decision boundary:

$$g_1(\mathbf{x}) = g_2(\mathbf{x})$$

$$\mathbf{x}^t\mathbf{W}_1\mathbf{x} + \mathbf{w}_1{}^t\mathbf{x} + w_{10} = \mathbf{x}^t\mathbf{W}_2\mathbf{x} + \mathbf{w}_2{}^t\mathbf{x} + w_{20}$$

- Resulting in:

$$g(\mathbf{x}) = \mathbf{x}^t\left(\mathbf{\Sigma}_2^{-1} - \mathbf{\Sigma}_1^{-1}\right)\mathbf{x} + 2\left(\mathbf{\Sigma}_1^{-1}\boldsymbol{\mu}_1 - \mathbf{\Sigma}_2^{-1}\boldsymbol{\mu}_2\right)^t\mathbf{x}$$

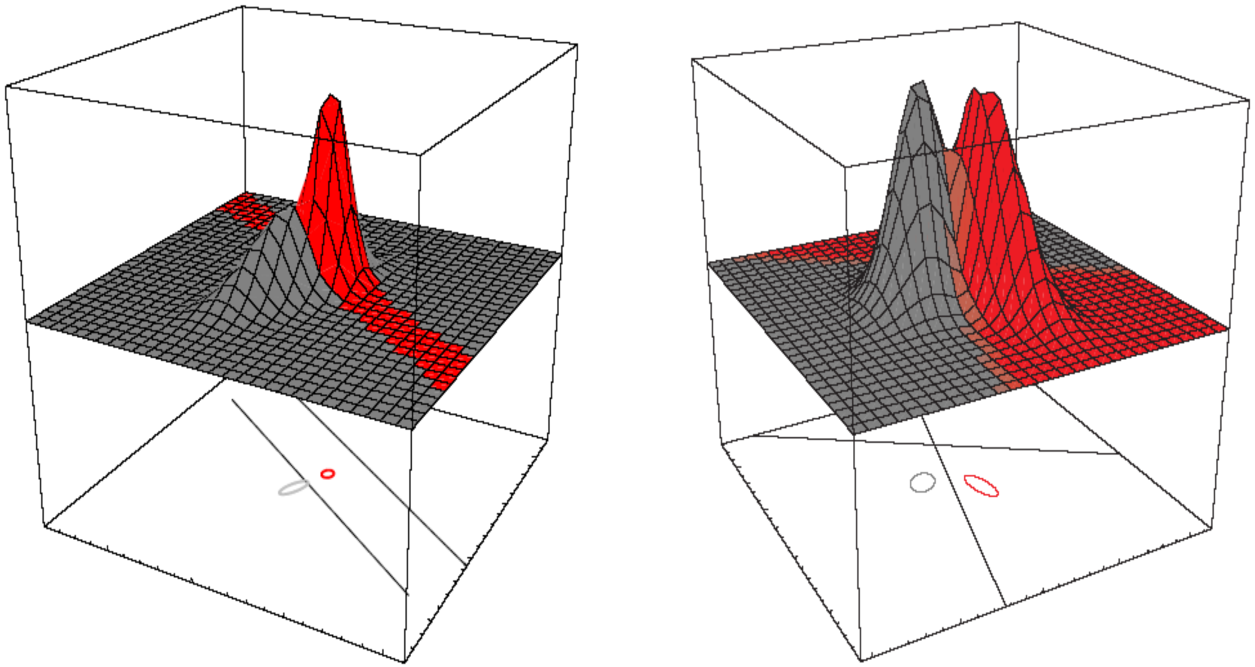
$$-\boldsymbol{\mu}_1^t\mathbf{\Sigma}_1^{-1}\boldsymbol{\mu}_1 + \boldsymbol{\mu}_2^t\mathbf{\Sigma}_2^{-1}\boldsymbol{\mu}_2 - \ln\frac{|\,\mathbf{\Sigma}_1\,|}{|\,\mathbf{\Sigma}_2\,|} + 2\ln\frac{P(\omega_1)}{P(\omega_2)} = 0$$

… a *hyperquadric* in the *d*-dimensional space

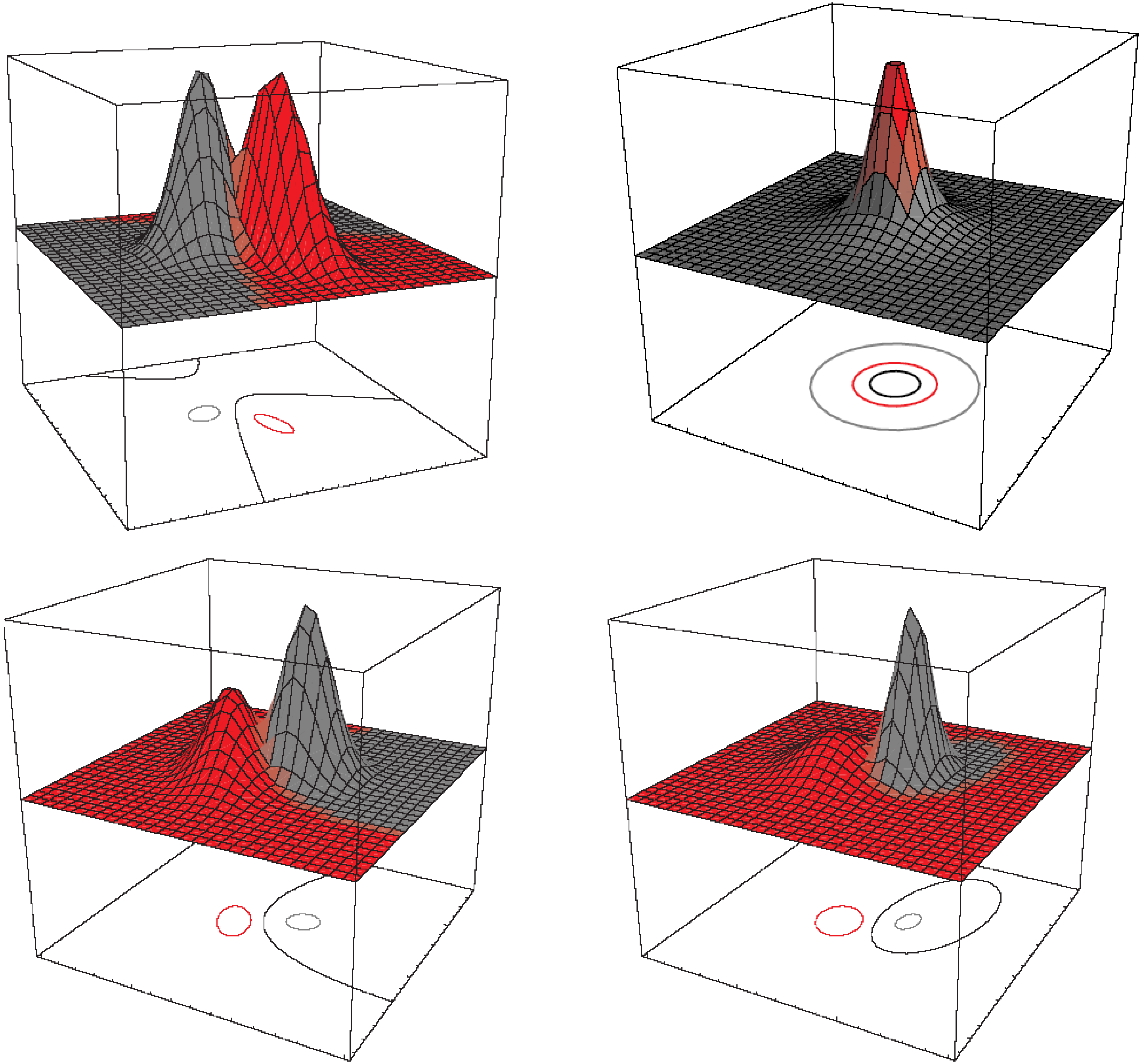
Shapes - various types

- Hyperplane
- Hypersphere
- Hiperellipsoid
- Hyperparaboloid
- Hyperhyperboloid
- Pair of Hyperplanes
- Pair of lines
- A single point
- … and many more

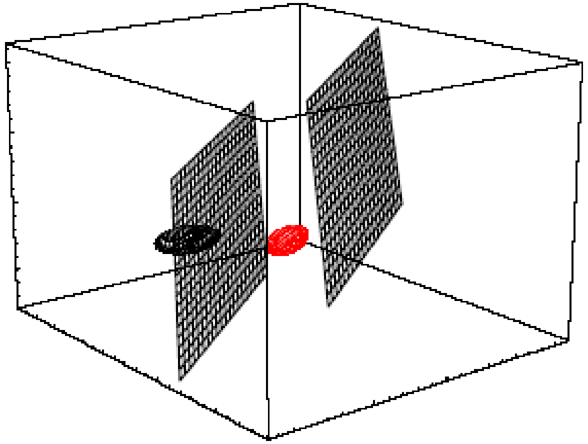
Example: 2D

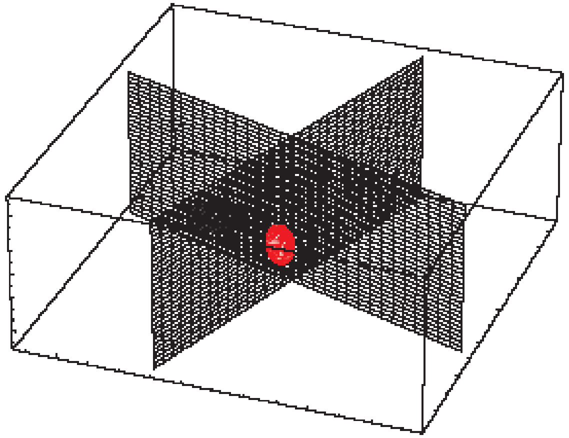


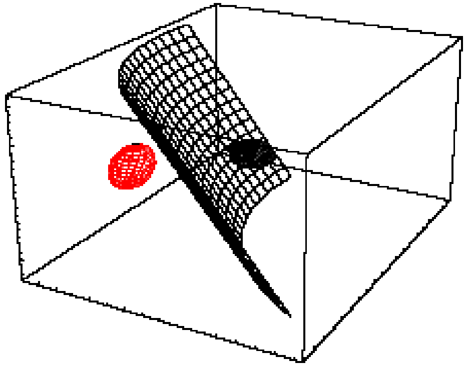
Examples: 2D

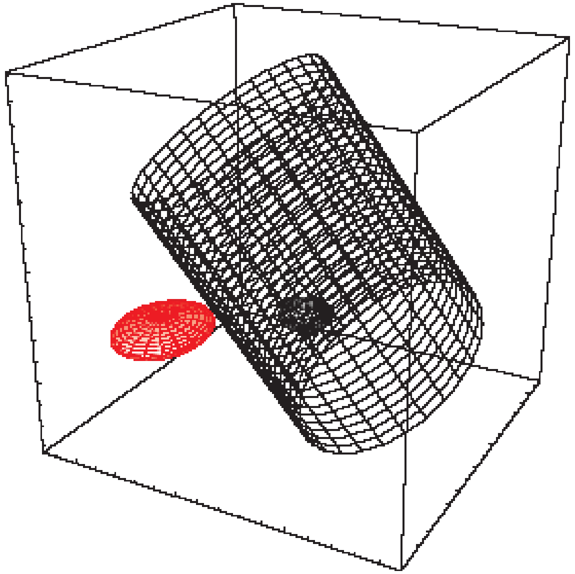


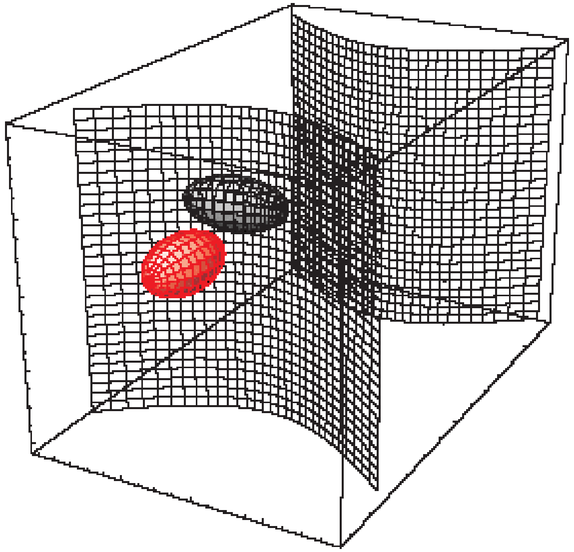
Examples: 3D feature space

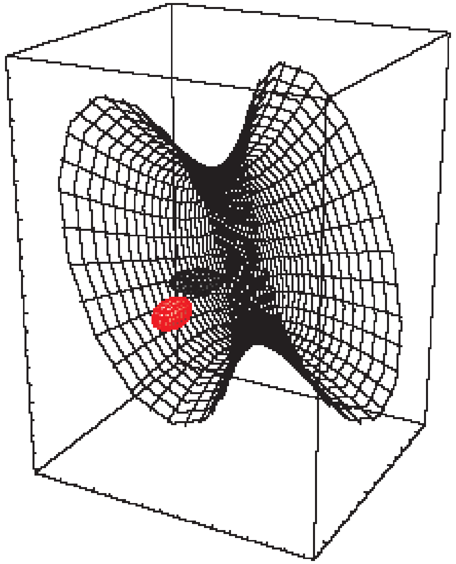

pair of parallel planes


pair of planes


parabolic cylinder

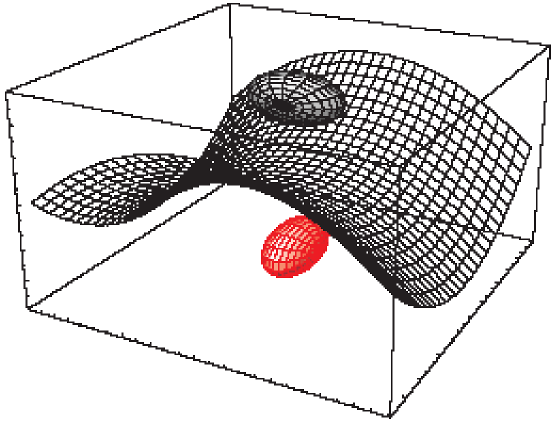

cylinder


hyperbolic cylinder

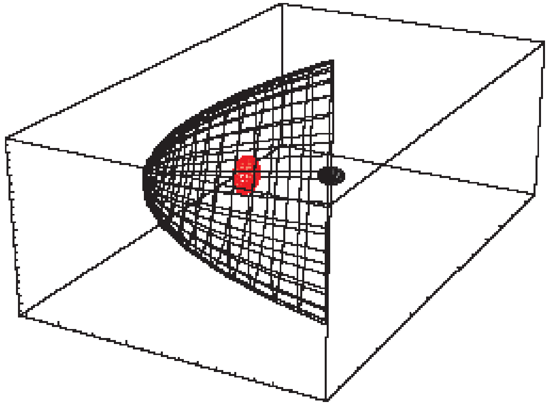

hyperboloid

Examples: 3D… (cont'd)

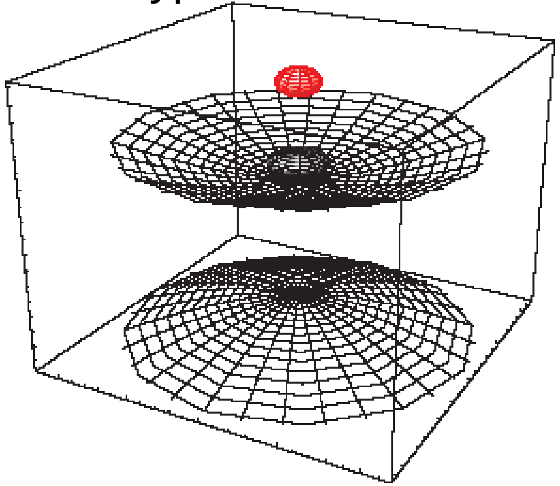
hyperbolic paraboloid



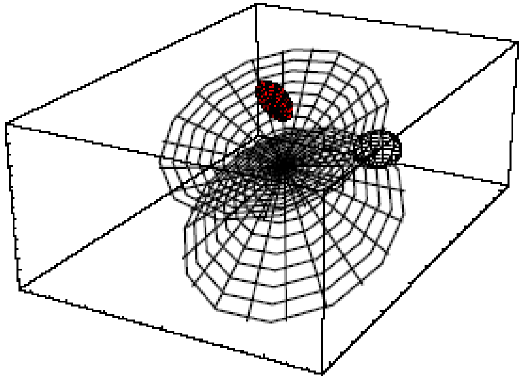
elliptic paraboloid



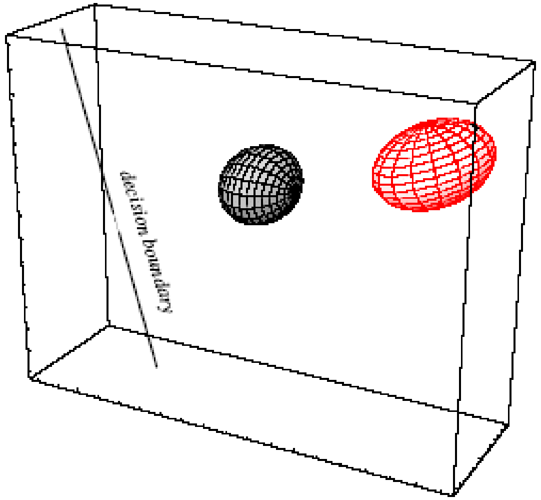
hyperboloid



hyperboloid



single straight line



Example: 4 classes, 2D feature space

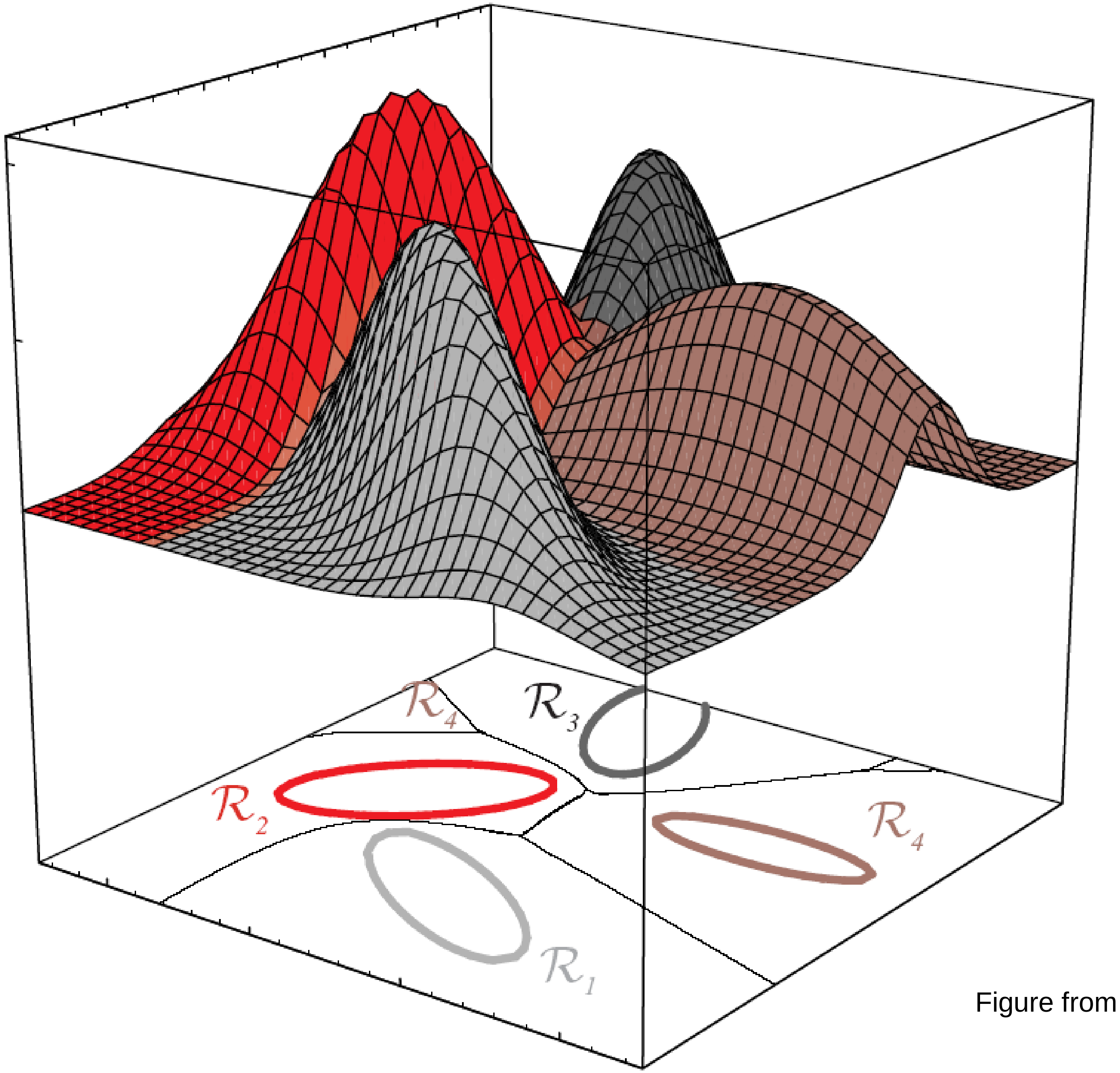
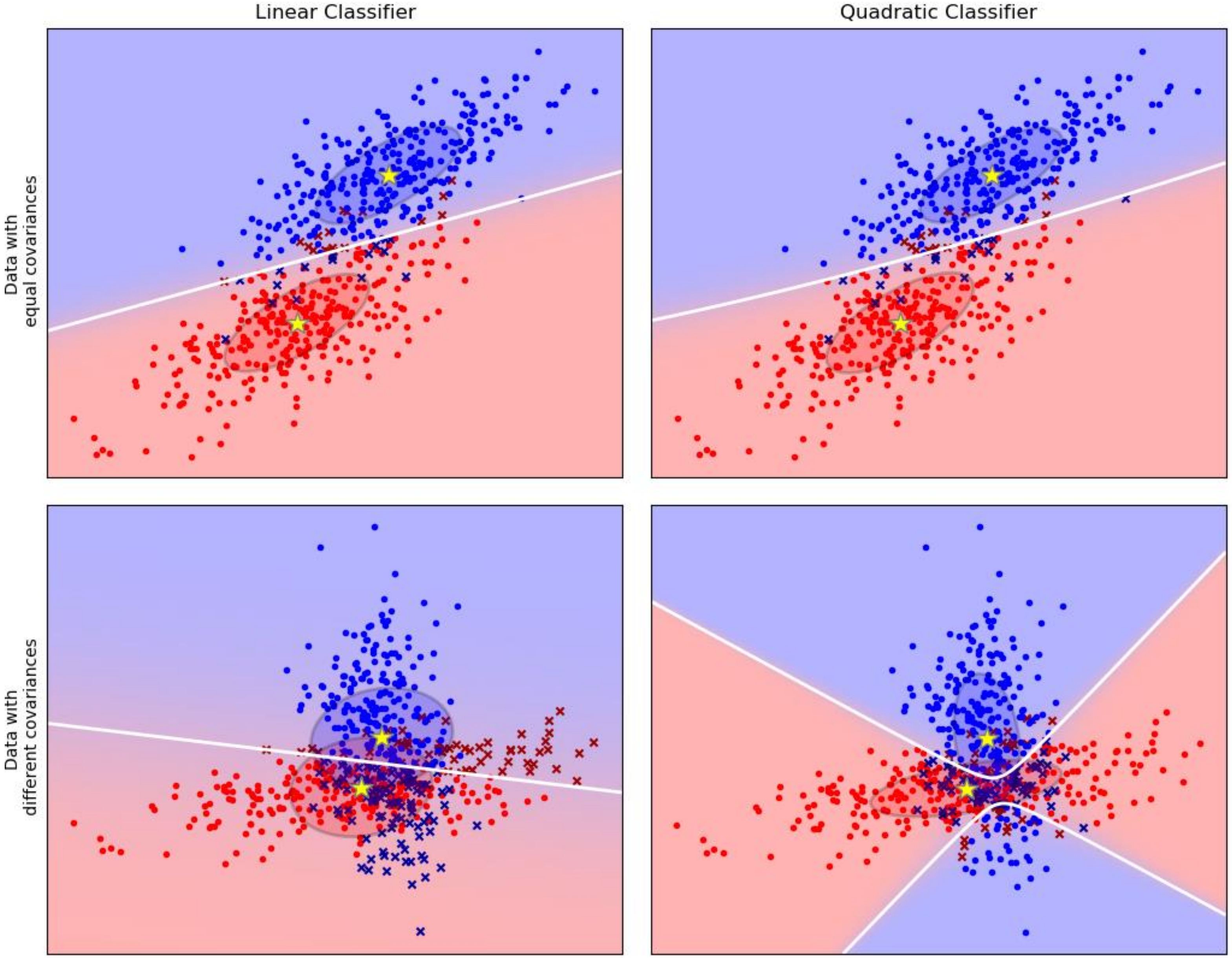


Figure from Duda et al.

Example: Linear vs Quadratic Classifier



Parameter Estimation

- We studied how to obtain the Bayesian classifier.

Assumptions:

Known:

  - $P(\omega_j)$, the a priori probabilities
  - $p(\mathbf{x} \mid \omega_j)$, the class-conditional densities

- In general:

These two are **unknown**, and have to be estimated

- To estimate the priors:
  - Assume a multinomial random variable, ω, and use maximum likelihood estimate

Parameter Estimation

Prior probabilities

Assume we have *c* classes:

- What is: $P(\omega_1), P(\omega_2), \ldots, P(\omega_c)$?
- Say ω is a multinomial r.v.
- Suppose from samples we have seen:

$$\begin{array}{lll} n_1 & \text{samples of class} & \omega_1 \\ n_2 & \text{samples of class} & {}_{+}\,\omega_2 \\ \vdots & & \\ n_c & \text{samples of class} & \underline{\omega_c} \\ n & \text{total number of samples} & \end{array}$$

Then

$$\mathrm{P}(\omega_i) = \frac{n_i}{n}$$

250	cats	(ω_1)
350	dogs	(ω_2)
100	horses	(ω_3)
700	total	

$$P(\omega_1) = \frac{250}{700} \approx 0.357$$

$$P(\omega_2) = \frac{350}{700} \approx 0.5$$

$$P(\omega_3) = \frac{100}{700} \approx 0.143$$

Maximum Likelihood Estimation (MLE)

- **Assume:**
  - Class-conditional probs have a *parametric* form.
- **Aim:**
  - Estimate the parameters of the pdf.
- **Schemes:**
  - MLE views these parameters as **quantities**, while
  - Bayesian estimation views them as **random variables**.

- **Given:**
  - c classes: $\omega_1, \omega_2, \ldots, \omega_c$
  - Labeled dataset: D = $D_1 \cup D_2 \cup \ldots \cup D_c$
  - where each sample in D_j is drawn independently based on the pdf $p(\mathbf{x} \mid \omega_j)$
- i.e., the samples are *independent and identically distributed* (iid) random vectors (or variables).
- **Assumption:**

  $p(\mathbf{x} \mid \omega_j)$ has a *parametric* form

  the aim is to estimate the parameters, vector $\boldsymbol{\theta}_j$

The Problem

- Extract the information from D_j to obtain a good estimate of θ_j
- **Another assumption:**
  - D_i does not give any info about θ_j where $i \neq j$
- Each D_j is independent from each other
- Assume we deal with one dataset: D_j, Rename it D, as follows

$$D = \{\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_n\}$$

- We then have c separate problems:
  - Given D, and the parametric form $p(\mathbf{x} \mid \omega_j)$, estimate the unknown parameter θ

- Examples:
  - Normal distribution: $\theta = [\mu \;\; \Sigma]^t$
  - Exponential distribution: $\theta = [\lambda]^t$
- The parametric forms of the normal distribution…
- Considering the k^{th} sample (it is the same for all samples):

$$p(\mathbf{x}_k \mid \boldsymbol{\mu}) = \frac{1}{(2\pi)^{\frac{d}{2}} |\boldsymbol{\Sigma}|^{\frac{1}{2}}} e^{-\frac{1}{2}(\mathbf{x}_k - \boldsymbol{\mu})^t \boldsymbol{\Sigma}^{-1} (\mathbf{x}_k - \boldsymbol{\mu})}$$

Aim of MLE

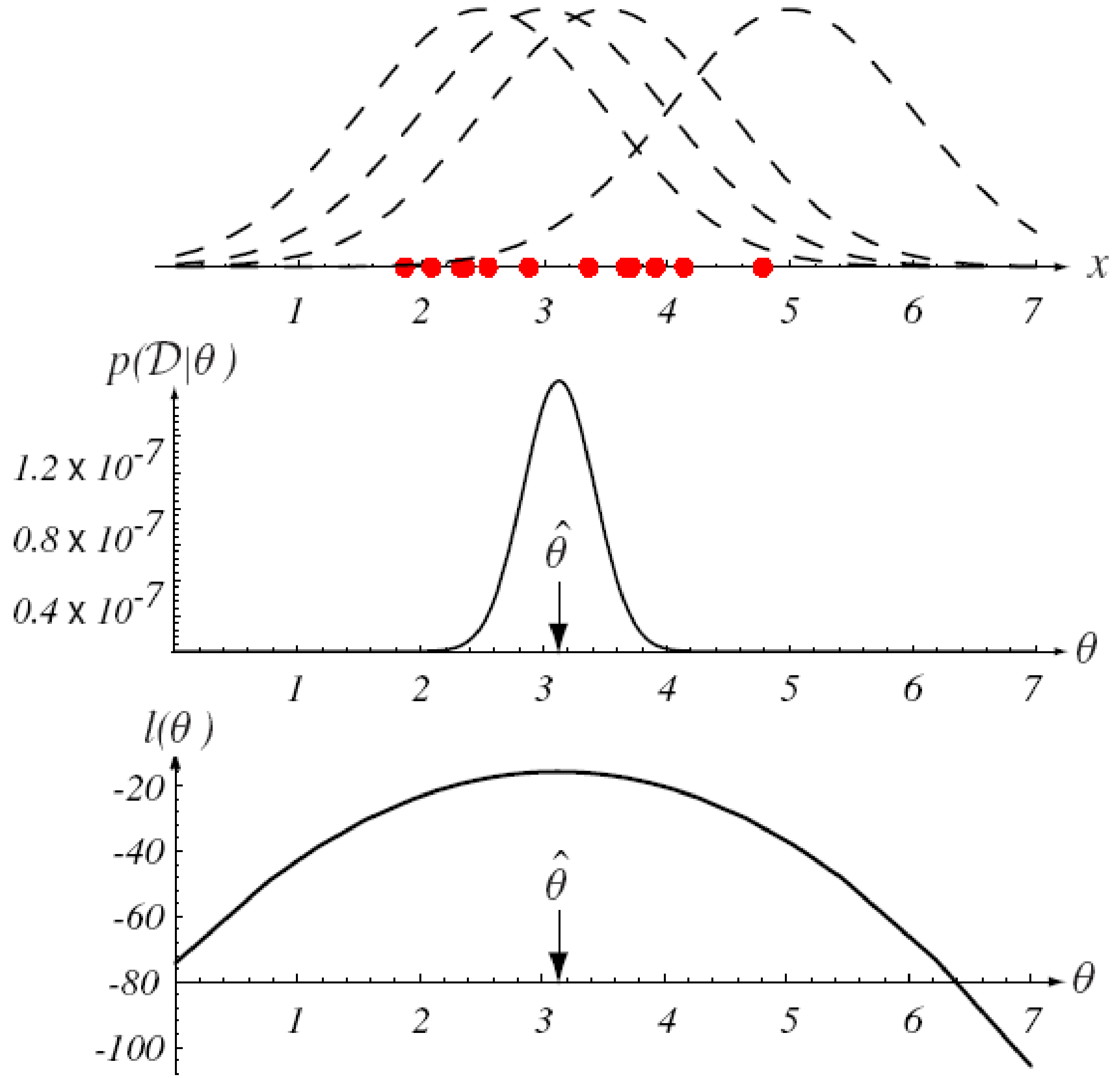
Maximize the *likelihood* that *D* is generated from **θ**.

- Since each sample $\mathbf{x}_i$ in *D* is drawn independently, then, the likelihood is given by:

$$p(\mathcal{D} \mid \boldsymbol{\theta}) = \prod_{k=1}^{n} p(\mathbf{x}_k \mid \boldsymbol{\theta})$$

- The *maximum likelihood estimate* of **θ** is the value $\hat{\boldsymbol{\theta}}$ that maximizes $p(\mathcal{D} \mid \boldsymbol{\theta})$
- For convenience:
  - work with the *logarithm* of the likelihood.
- Maximizing *log* of likelihood

  ⇒ Maximizing likelihood



- If $p(\mathcal{D} \mid \boldsymbol{\theta})$ is differentiable on $\boldsymbol{\theta}$, then..

  $\hat{\boldsymbol{\theta}}$ can be found using differential calculus…

- **How?**
  - Let $\boldsymbol{\theta}$ be a *t*-component vector, $\boldsymbol{\theta} = [\theta_1, \ldots, \theta_t]^t$, and
  - let $\nabla_{\boldsymbol{\theta}}$ be the *gradient* operator:

$$\nabla_{\boldsymbol{\theta}} = \begin{bmatrix} \dfrac{\partial}{\partial \theta_1} \\ \vdots \\ \dfrac{\partial}{\partial \theta_t} \end{bmatrix}$$

  - Define $l(\boldsymbol{\theta})$ as the *log-likelihood* function:

$$l(\boldsymbol{\theta}) \equiv \ln p(\mathcal{D} \mid \boldsymbol{\theta})$$

- Formally, the argument of $l(\boldsymbol{\theta})$ that maximizes the log-likelihood is:

$$\hat{\boldsymbol{\theta}} = \arg\max_{\boldsymbol{\theta}} \ln p(\mathcal{D} \mid \boldsymbol{\theta})$$

- We can write the log-likelihood as:

$$l(\boldsymbol{\theta}) = \ln \prod_{k=1}^{n} p(\mathbf{x}_k \mid \boldsymbol{\theta}) = \sum_{k=1}^{n} \ln p(\mathbf{x}_k \mid \boldsymbol{\theta})$$

and

$$\nabla_{\boldsymbol{\theta}} l = \sum_{k=1}^{n} \nabla_{\boldsymbol{\theta}} \ln p(\mathbf{x}_k \mid \boldsymbol{\theta})$$

- Obtaining a set of **necessary** conditions (*t* equations): $\nabla_{\boldsymbol{\theta}} l = \mathbf{0}$

- A solution $\hat{\boldsymbol{\theta}}$ to this can be:
  - A *true* global maximum or minimum,
  - A *local* maximum or minimum, or
  - An *inflection* point.
- Then, check *second* derivative.
- Note:
  - $\hat{\boldsymbol{\theta}}$ is an *estimate*, and then
  - we must have an *infinite* number of samples to obtain the *true* parameter.

Normal Distribution

- **First case:** Unknown $\boldsymbol{\mu}$
- Suppose the samples are drawn from a multivariate normal distribution…
- Then, $\boldsymbol{\theta} = [\boldsymbol{\mu}\ \ \boldsymbol{\Sigma}]^t$
- Suppose also that $\boldsymbol{\Sigma}$ is known
- Consider one of the samples (it is the same for all samples):

$$p(\mathbf{x}_k \mid \boldsymbol{\mu}) = \frac{1}{(2\pi)^{\frac{d}{2}} |\boldsymbol{\Sigma}|^{\frac{1}{2}}} e^{-\frac{1}{2}(\mathbf{x}_k - \boldsymbol{\mu})^t \boldsymbol{\Sigma}^{-1} (\mathbf{x}_k - \boldsymbol{\mu})}$$

- Then:

$$\ln p(\mathbf{x}_k \mid \boldsymbol{\mu}) = -\frac{1}{2}\ln\left[(2\pi)^d |\boldsymbol{\Sigma}|\right] - \frac{1}{2}(\mathbf{x}_k - \boldsymbol{\mu})^t \boldsymbol{\Sigma}^{-1} (\mathbf{x}_k - \boldsymbol{\mu})$$

- Differentiating wrt $\boldsymbol{\mu}$, we have:

$$\nabla_{\boldsymbol{\theta}} \ln p(\mathbf{x}_k \mid \boldsymbol{\mu}) = \boldsymbol{\Sigma}^{-1}(\mathbf{x}_k - \boldsymbol{\mu})$$

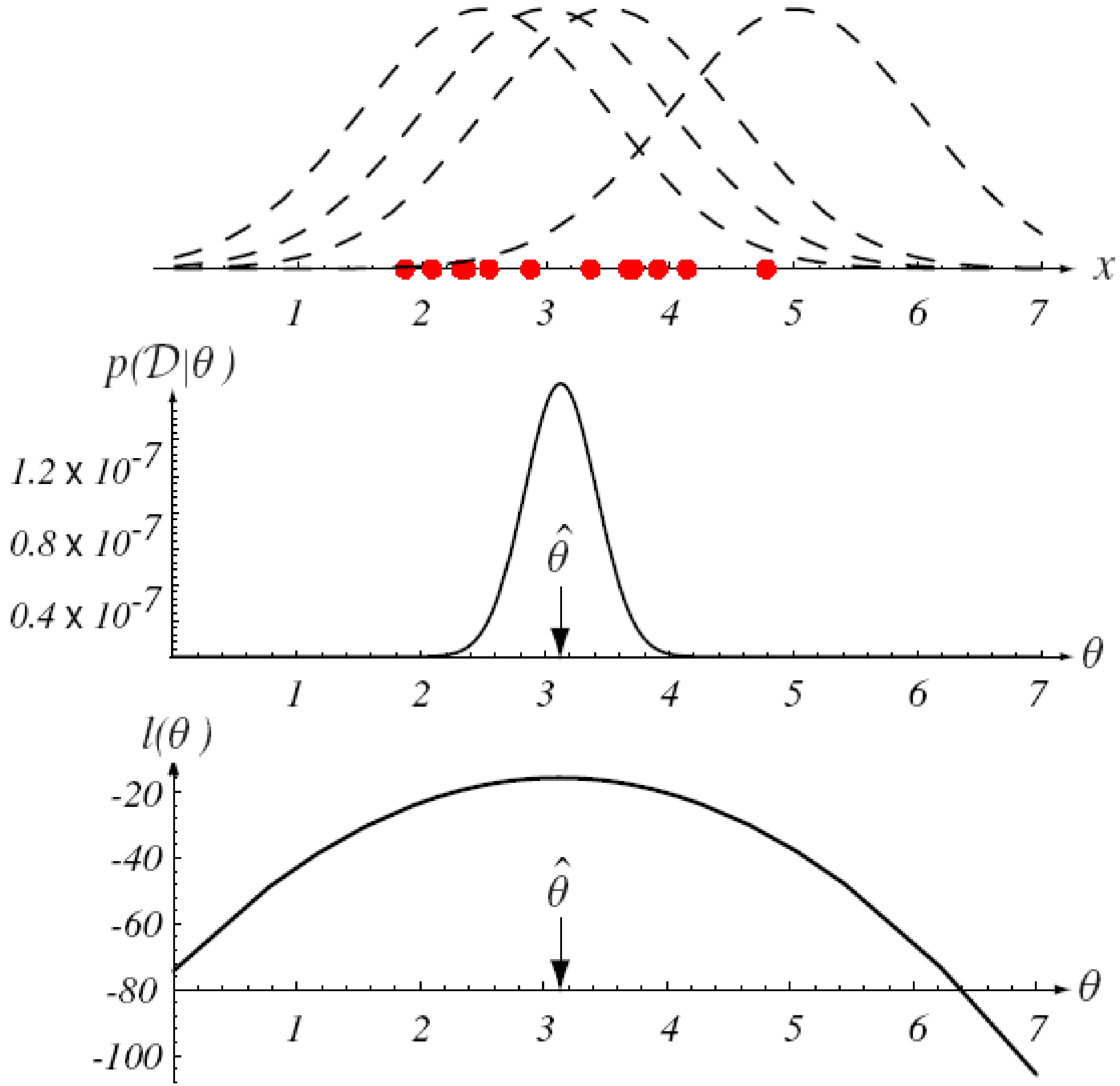
- We know that this is true for all *k*, and thus the necessary condition for a maximum is:

$$\sum_{k=1}^{n} \boldsymbol{\Sigma}^{-1}(\mathbf{x}_k - \boldsymbol{\mu}) = \mathbf{0}$$

- Pre-multiplying by Σ, and rearranging: $\hat{\boldsymbol{\mu}} = \frac{1}{n}\sum_{k=1}^{n} \mathbf{x}_k$

  where $\hat{\boldsymbol{\mu}}$ is called the *sample mean*
- If we view the samples as a "*cloud*"…
  $\hat{\boldsymbol{\mu}}$ is the *centroid* of the cloud

Pictorially (mean):



Second case: Unknown μ and Σ:

- This is the *typical* case in normal distributions
- Consider the *univariate* normal distribution
- Thus, $\boldsymbol{\theta} = [\theta_1 \ \ \theta_2] = [\mu \ \ \sigma^2]^t$
- The log-likelihood function:

$$\ln p(x_k \mid \boldsymbol{\theta}) = -\frac{1}{2}\ln(2\pi\theta_2) - \frac{1}{2\theta_2}(x_k - \theta_1)^2$$

- Taking derivatives and solving, we have:

$$\hat{\mu} = \frac{1}{n}\sum_{k=1}^{n} x_k \quad \text{and} \quad \hat{\sigma}^2 = \frac{1}{n}\sum_{k=1}^{n}(x_k - \hat{\mu})^2$$

- Multivariate case:

$$\hat{\boldsymbol{\mu}} = \frac{1}{n}\sum_{k=1}^{n} \mathbf{x}_k \quad \text{and} \quad \hat{\boldsymbol{\Sigma}} = \frac{1}{n}\sum_{k=1}^{n} (\mathbf{x}_k - \hat{\boldsymbol{\mu}})(\mathbf{x}_k - \hat{\boldsymbol{\mu}})^t$$

- Example for normal distribution:

- Let: $D = \left\{ \begin{bmatrix} 1 \\ 1 \end{bmatrix}, \begin{bmatrix} 2 \\ 2 \end{bmatrix}, \begin{bmatrix} 1.5 \\ 1.8 \end{bmatrix}, \begin{bmatrix} 2.2 \\ 3.3 \end{bmatrix}, \begin{bmatrix} 3.7 \\ 4.2 \end{bmatrix} \right\}$

- We have:

$$\hat{\boldsymbol{\mu}} = \frac{1}{5}\left(\begin{bmatrix} 1 \\ 1 \end{bmatrix} + \begin{bmatrix} 2 \\ 2 \end{bmatrix} + \begin{bmatrix} 1.5 \\ 1.8 \end{bmatrix} + \begin{bmatrix} 2.2 \\ 3.3 \end{bmatrix} + \begin{bmatrix} 3.7 \\ 4.2 \end{bmatrix} \right) = \frac{1}{5}\begin{bmatrix} 10.4 \\ 12.3 \end{bmatrix} = \begin{bmatrix} 2.08 \\ 2.46 \end{bmatrix}$$

$$\hat{\mathbf{\Sigma}} = \frac{1}{5}\left\{\left(\begin{bmatrix}1\\1\end{bmatrix}-\begin{bmatrix}2.08\\2.46\end{bmatrix}\right)\left(\begin{bmatrix}1\\1\end{bmatrix}-\begin{bmatrix}2.08\\2.46\end{bmatrix}\right)^t + \left(\begin{bmatrix}2\\2\end{bmatrix}-\begin{bmatrix}2.08\\2.46\end{bmatrix}\right)\left(\begin{bmatrix}2\\2\end{bmatrix}-\begin{bmatrix}2.08\\2.46\end{bmatrix}\right)^t\right\} +$$

$$\frac{1}{5}\left\{\left(\begin{bmatrix}1.5\\1.8\end{bmatrix}-\begin{bmatrix}2.08\\2.46\end{bmatrix}\right)\left(\begin{bmatrix}1.5\\1.8\end{bmatrix}-\begin{bmatrix}2.08\\2.46\end{bmatrix}\right)^t + \left(\begin{bmatrix}2.2\\3.3\end{bmatrix}-\begin{bmatrix}2.08\\2.46\end{bmatrix}\right)\left(\begin{bmatrix}2.2\\3.3\end{bmatrix}-\begin{bmatrix}2.08\\2.46\end{bmatrix}\right)^t\right\} +$$

$$\frac{1}{5}\left\{\left(\begin{bmatrix}3.7\\4.2\end{bmatrix}-\begin{bmatrix}2.08\\2.46\end{bmatrix}\right)\left(\begin{bmatrix}3.7\\4.2\end{bmatrix}-\begin{bmatrix}2.08\\2.46\end{bmatrix}\right)^t\right\}$$

$$\hat{\mathbf{\Sigma}} = \frac{1}{5}\left\{\begin{bmatrix}1.16 & 1.57\\1.57 & 2.13\end{bmatrix}+\begin{bmatrix}0.01 & 0.03\\0.03 & 0.21\end{bmatrix}+\begin{bmatrix}0.33 & 0.38\\0.38 & 0.48\end{bmatrix}+\begin{bmatrix}0.01 & 0.10\\0.10 & 0.70\end{bmatrix}+\begin{bmatrix}2.62 & 2.81\\2.81 & 3.02\end{bmatrix}\right\}$$

$$\hat{\Sigma} = \begin{bmatrix}0.82 & 0.98\\0.98 & 1.30\end{bmatrix}$$

Naïve Bayes Classifier

- We've seen how to estimate the pdf for the class-conditional probs, $p(\mathbf{x}|\omega_j)$, from a dataset D
- We've used random vectors to represent $\mathbf{x}$:

$$\mathbf{x} = [x_1\ x_2\ \dots\ x_d]^t \in \Re^d$$

- Normal distribution is a good option for many reasons, as discussed
- Estimate the mean and covariance and we're done
- But data may not be normally distributed.

Moreover, in practice:

- We may not have a large number of training samples
- Example: Gene expression data with n = 30 samples (patients) and d = 10,000+ features (gene expressions or transcript measures)
- Roughly speaking, if n is a good number of points, we should have n^d points for d dimensions
- But that's not always the case

Naïve Bayes Classifier

- Solution:
    - Assume that x_1 x_2 … x_d are independent
- Thus, $p(\mathbf{x}|\omega_j)$ will be:

$$p(\mathbf{x} \mid \omega_j) = \prod_{i=1}^{d} p(x_i \mid \omega_j) \quad j = 1,2,...,c$$

- Then, given an unknown sample, we assign the class as follows:

$$\omega_j = \arg\max_{\omega_j} \prod_{i=1}^{d} p(x_i \mid \omega_j) \quad j = 1,2,...,c$$

- This is the Naïve Bayes Classifier!
- It's simple, and it's been shown to work well in many real classification problems

Bayes Classification in Scikit

Bayes:

- Uses Gaussian distributions
- Called Quadratic Discriminant Analysis

https://scikit-learn.org/stable/modules/lda_qda.html

Naïve Bayes:

- Gaussian:
  - Based on normal distributions of independent random variables
- Multinomial Naïve Bayes :
  - Also considers discrete attributes (features).
  - Suitable for text classification.
- Bernoulli Naïve Bayes:
  - Considers multiple features, binary-valued
  - Also suitable for text classification

http://scikit-learn.org/stable/modules/naive_bayes.html

Classifier Evaluation

Cross validation:

- Given a dataset, $D = D_1 \cup D_2 \cup \ldots \cup D_c$ where each D_i has n_i labeled samples
- Split D_i into two subsets D_{i1} and D_{i2}, (not necessarily of the same cardinality),
- Obtaining:

  training set: $D_{11} \cup D_{21} \cup \ldots \cup D_{c1}$

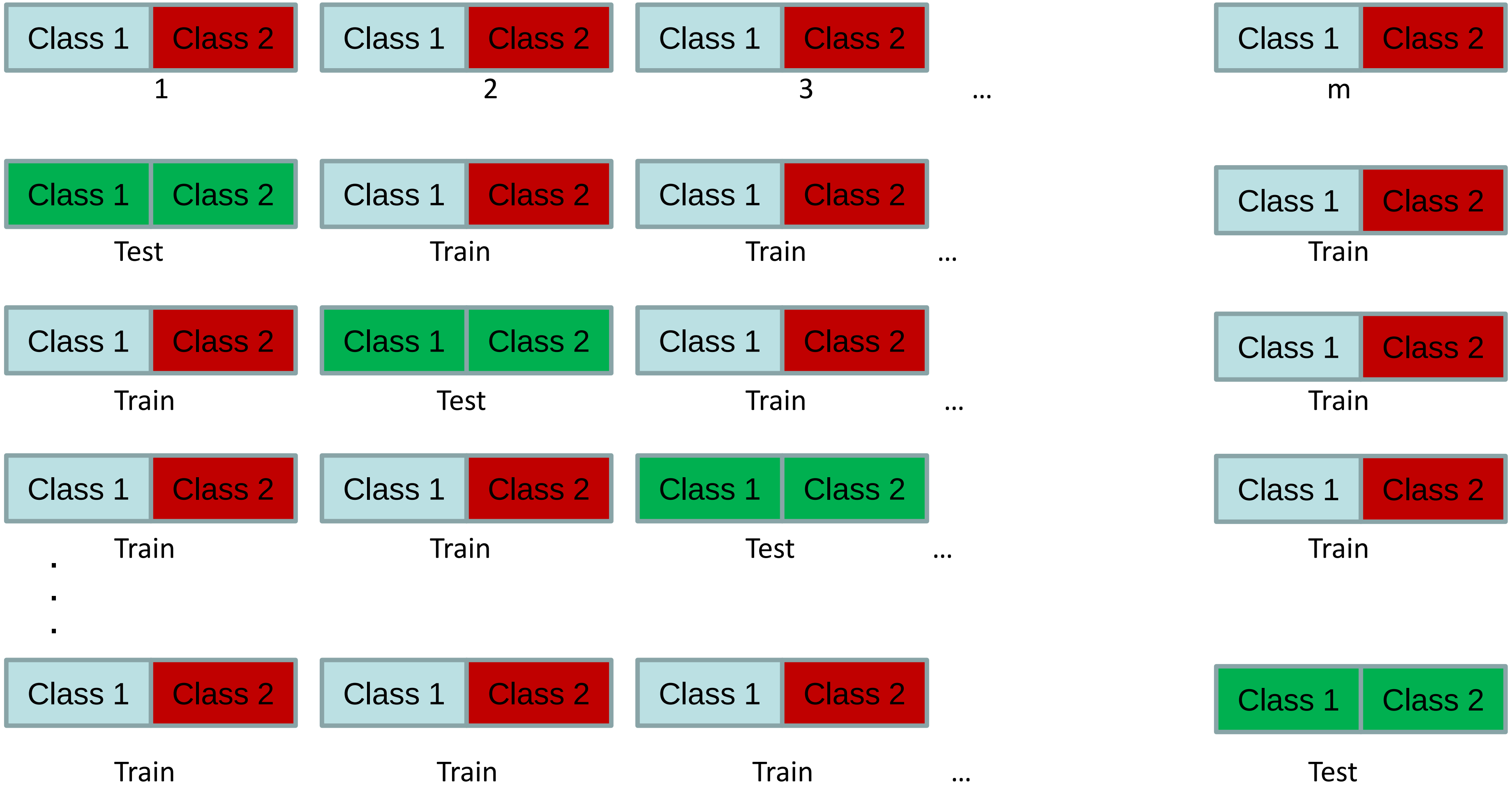
  test set: $D_{12} \cup D_{22} \cup \ldots \cup D_{c2}$
- **Train** classifier with $D_{11} \cup D_{21} \cup \ldots \cup D_{c1}$ **Test** classifier with $D_{12} \cup D_{22} \cup \ldots \cup D_{c2}$
- Do the same by swapping training and test sets

- It is a generalization of *cross validation*.
- Using this method, D_i is divided into m disjoint sets of equal size, say $\lfloor n_i/m \rfloor$

  (distributing remaining samples over some sets) where $n_i \geq m$
- The classifier is trained m times, and
  - at each time, one different set is left out for *testing*,
  - remaining samples used for *training*.

Cross Validation graphically

m-fold cross validation



Class 1 Class 2 Class 1 Class 2 Class 1 Class 2 Class 1 Class 2

1 2 3 … m

Class 1 Class 2 Class 1 Class 2 Class 1 Class 2 Class 1 Class 2

Test Train Train … Train

Class 1 Class 2 Class 1 Class 2 Class 1 Class 2 Class 1 Class 2

Train Test Train … Train

Class 1 Class 2 Class 1 Class 2 Class 1 Class 2 Class 1 Class 2

Train Train Test … Train

.
.
.

Class 1 Class 2 Class 1 Class 2 Class 1 Class 2 Class 1 Class 2

Train Train Train … Test

Confusion Matrix

- Given *c* classes, a $c \times c$ matrix **A** where:
  - a_{ij} contains the # of samples that are classified as ω_j when their **true** class is ω_i
- For $i = j$,
  - a_{ij} refers to how *accurate* is the classifier for ω_i
- whereas for $i \neq j$,
  - a_{ij} refers to how the classifier *misclassifies* samples

Example, 4 classes:

Predicted class

True class

	ω_1	ω_2	ω_3	ω_4
ω_1	18	2	0	1
ω_2	3	33	8	1
ω_3	0	2	12	1
ω_4	4	1	2	28

Performance Measures – 2 classes

- Suppose *two* classes: *positive* and *negative*
- Let:
  - *TP* = true positives (positive classified as positive)
  - *TN* = true negatives (negative classified as negative)
  - *FP* = false positives (negative classified as positive)
  - *FN* = false negatives (positive classified as negative)
  - *P* = total number of positives
  - *N* = total number of negatives
  - *n* = total number of samples

Performance Measures

- True positive TP rate:

$$TPrate = \frac{TP}{TP + FN} = \frac{TP}{P}$$

- Positive predicted value (PPV) or precision
  aka hit rate

$$PPV = \frac{TP}{TP + FP}$$

- False positive rate or FP rate
  aka false alarm rate

$$FP\,rate = \frac{FP}{TN + FP} = \frac{FP}{N}$$

- Negative predicted value (NPV)

$$NPV = \frac{TN}{TN + FN}$$

Performance Measures

- Specificity:

$$SP = \frac{TN}{TN + FP} = 1 - FP\,rate$$

- Sensitivity (recall):

$$SE = \frac{TP}{TP + FN} = TP\,rate$$

- Accuracy:

$$\frac{TP + TN}{n}$$

- Geometric mean:

$$G_m = \sqrt{SE \times SP}$$

Summary of Performance Measures

		Predicted class		**Metrics**
		Class1(+)	**Class2 (-)**	
True class	**Class1(+)**	TP	FN	**P =** TP+FN **True positive rate** (**Sensitivity**, **Recall**)= TP/P **False negative rate=** FN/P
	Class2 (-)	FP	TN	**N =** TN+FP **True negative rate** (**Specificity**)= TN/N **False positive rate=** FP/N
Metrics		**Positive predicted value (PPV) Precision** TP/(TP+FP)	**Negative predictive value(NPV)** TN/(TN+FN)	**n** = TP+FN+FP+TN **Accuracy=** (TP+TN)/n **Error rate =** (FP+FN)/n=1-Accuracy **Geometric mean=** $\sqrt{SE \times SP}$

Other Performance Measures

- Geometric mean

$$G_m = \sqrt{SE \times SP}$$

- Matthews Correlation Coefficient
  - +1: perfect classification
  - 0: average random classification
  - -1: inverse classification

$$MCC = \frac{TP \times TN - FP \times FN}{\sqrt{(TP + FP)(TP + FN)(TN + FP)(TN + FN)}}$$

- F-measure

$$F = \frac{(\beta^2 + 1)PPV \times SE}{\beta^2 PPV + SE}$$

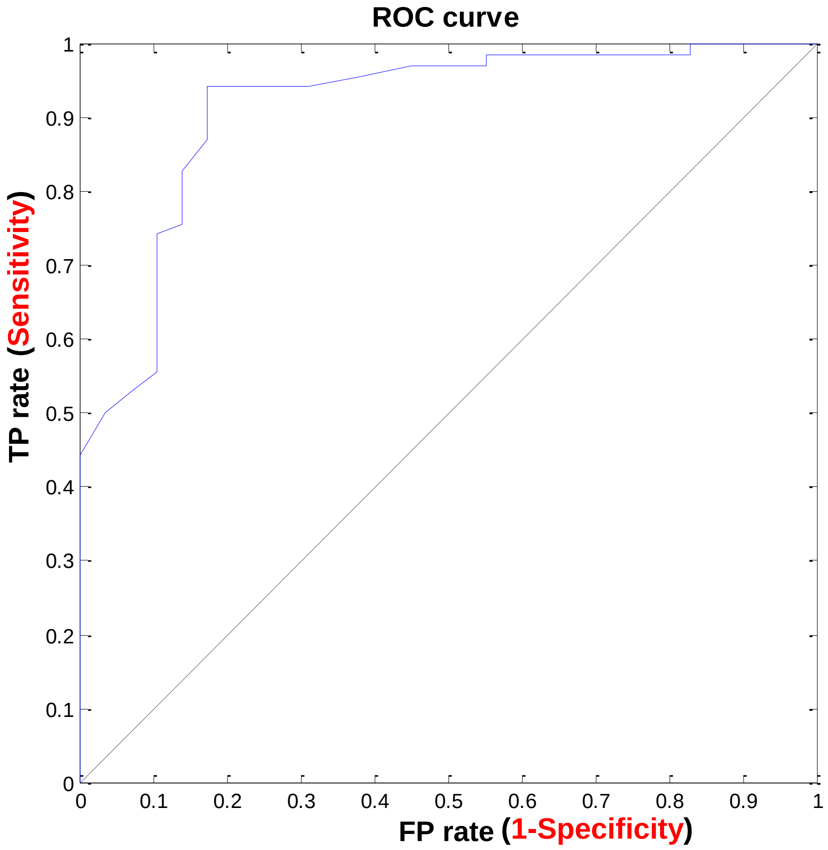
  - Common value of β=1 gives F1 measure:

$$F1 = 2(PPV \times SE)/(PPV + SE)$$

- ROC curve analysis
  - Involves visual and numerical analysis of the tradeoff between TP and FP rates

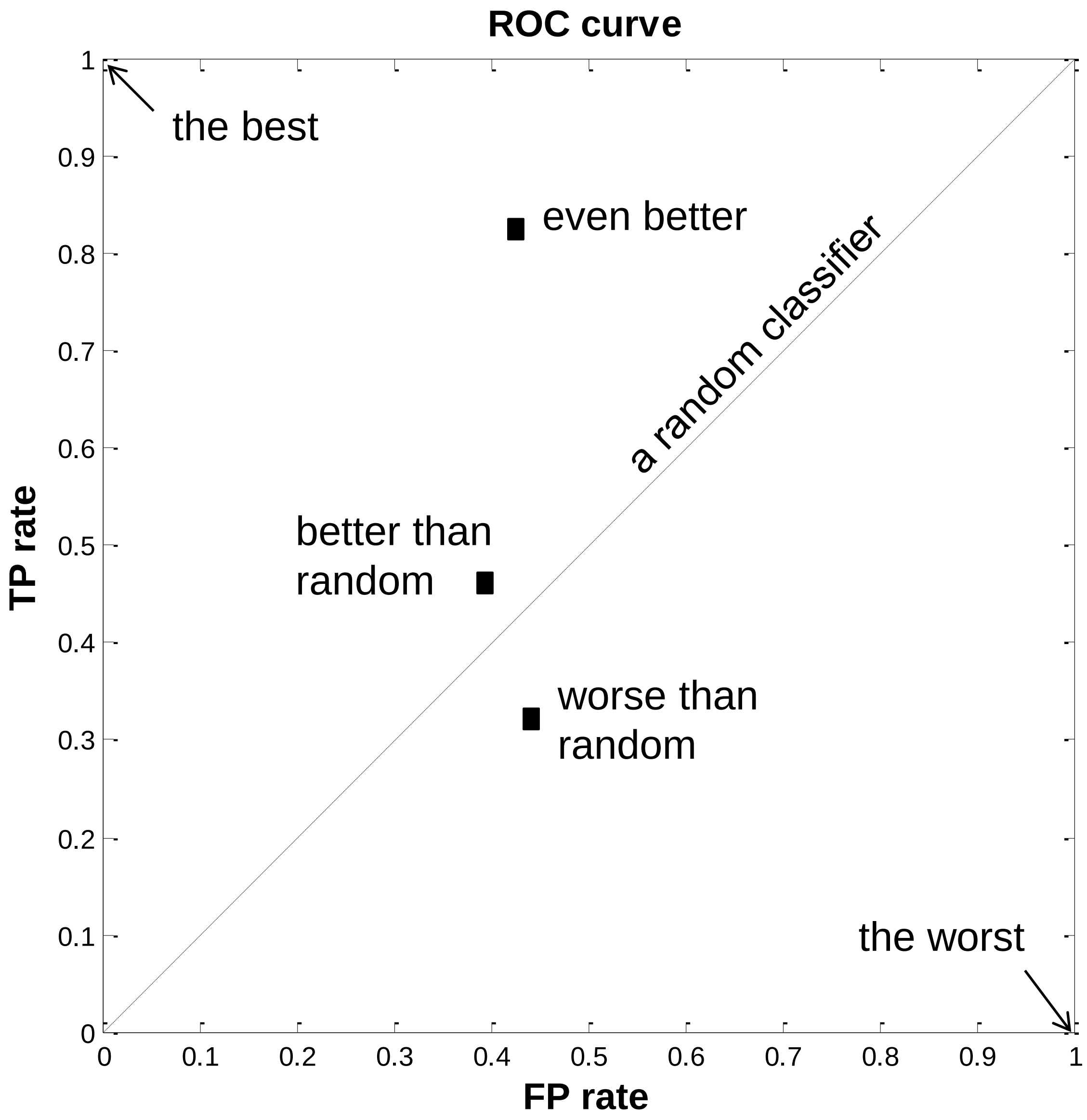
ROC Graphs

- Receiver Operating Characteristic (ROC) graph: visual tools for analysis
- Depicts tradeoff between TP/FP rates
- Better insight than simple metrics
- Suitable for unbalanced class problems
- Drawback: not easy to be constructed in practice



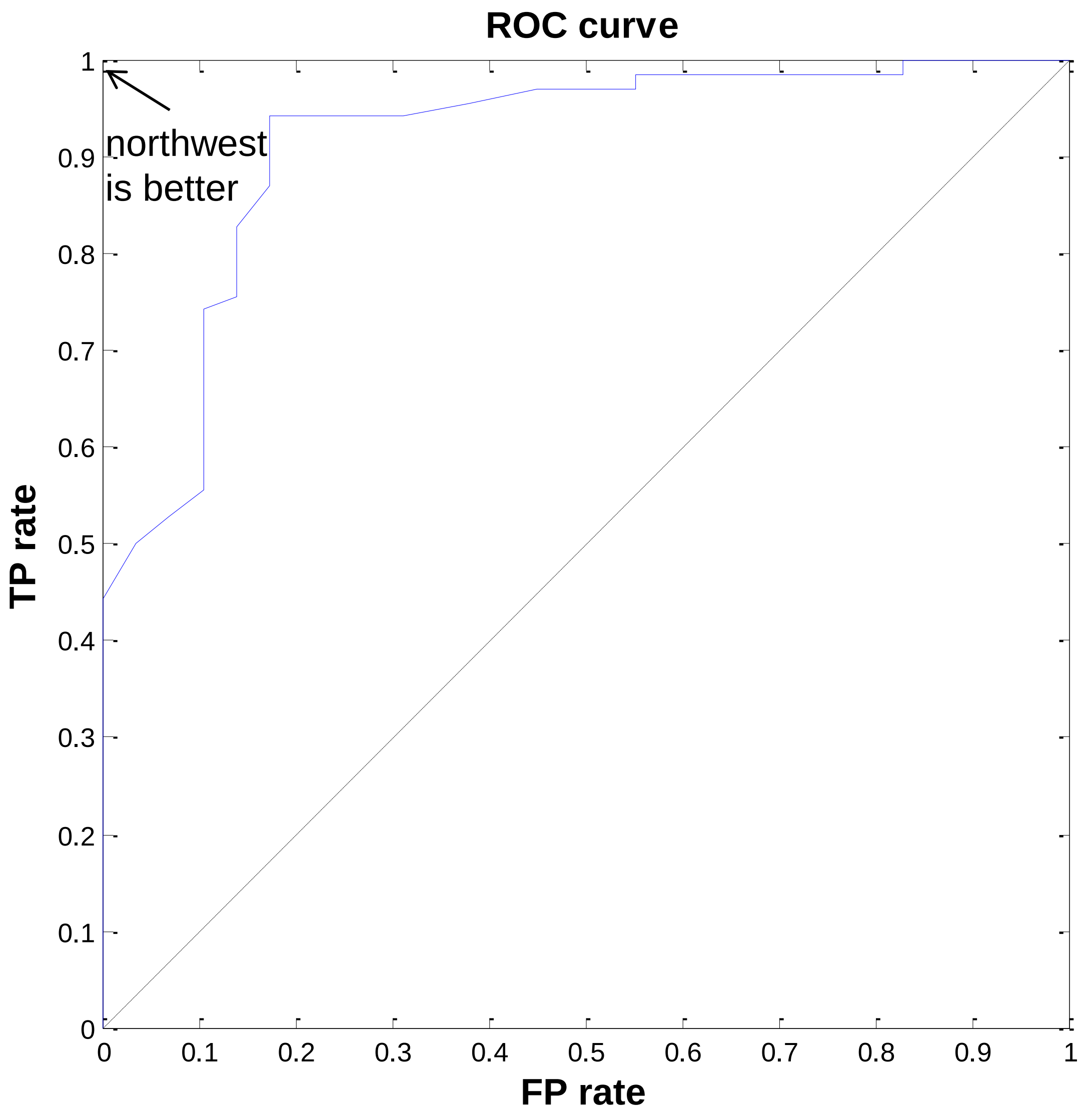
ROC Space

- One point corresponds to the result of a classifier
- Northwest is better
- Southeast is worse
- Diagonal $x = y$ corresponds to a random classifier



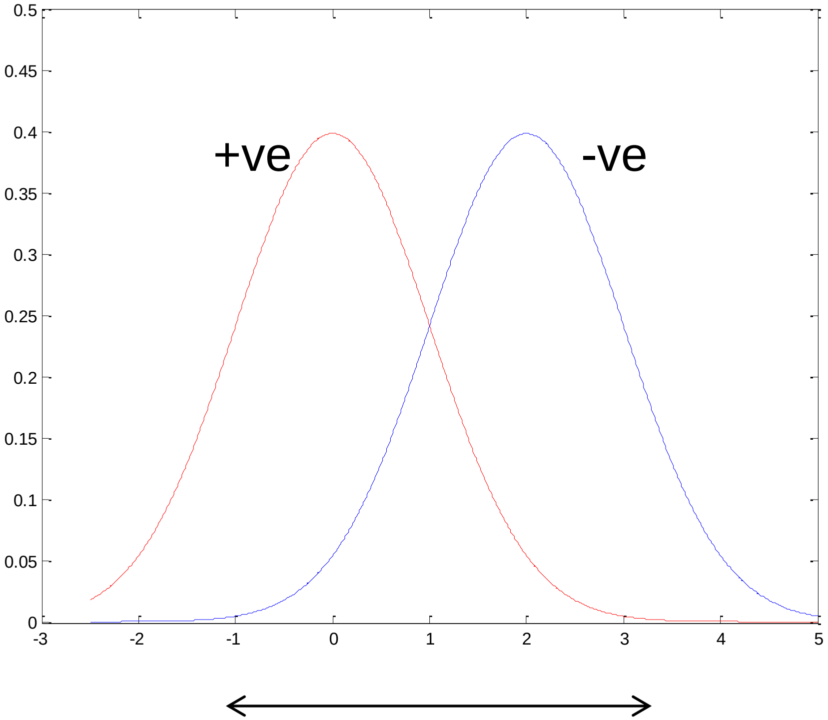
ROC Curve

- A curve in ROC space
- Northwest is better
- In theory:
  - Exact curve
- In practice:
  - Approximate
  - Varies on parameters
  - Varies on samples

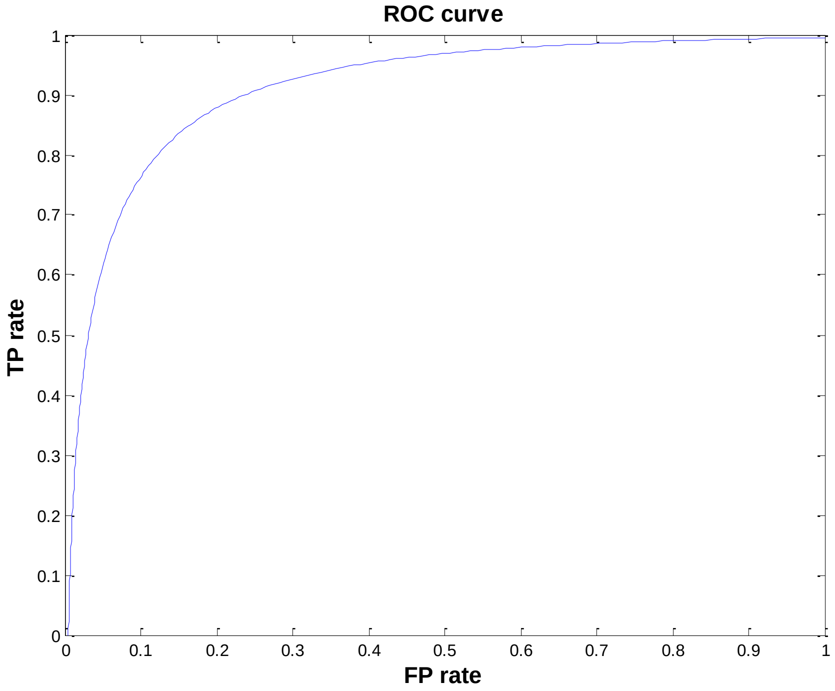


ROC in Theory

- Example: Two normally distributed classes

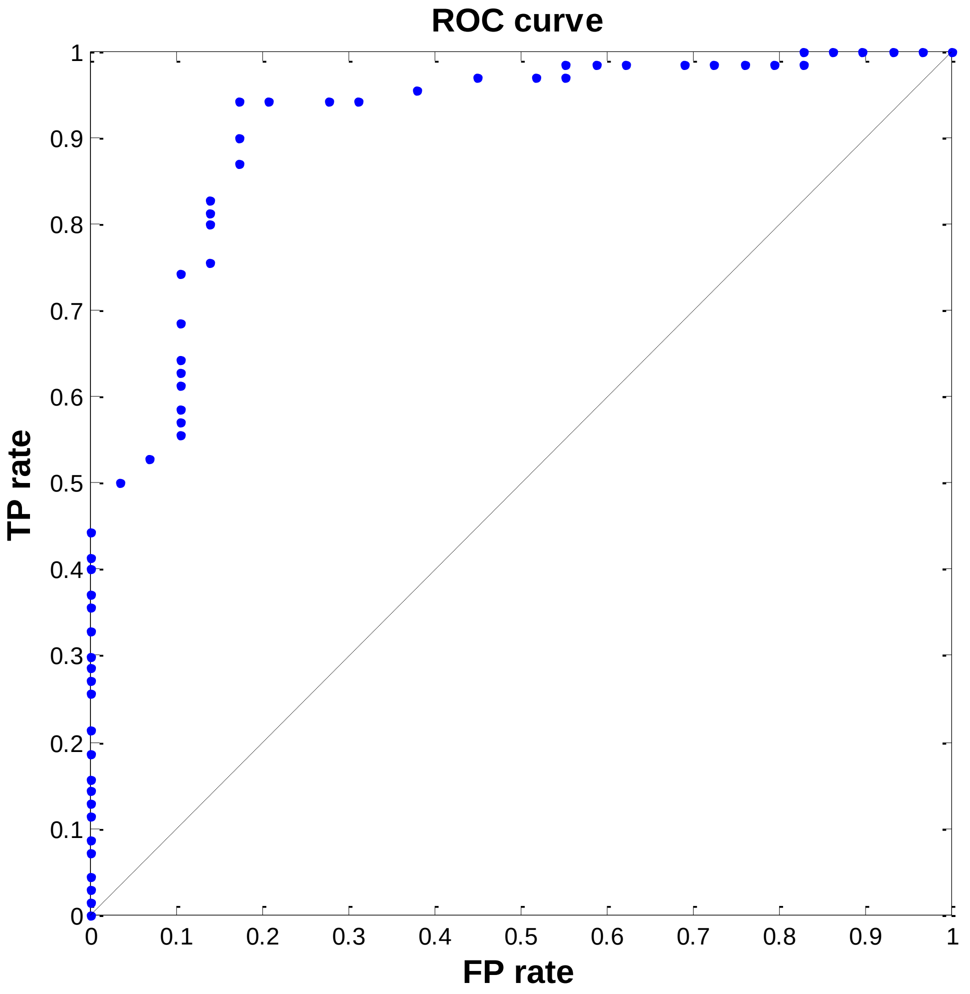


slide threshold from left to right
and find TP/FP rates



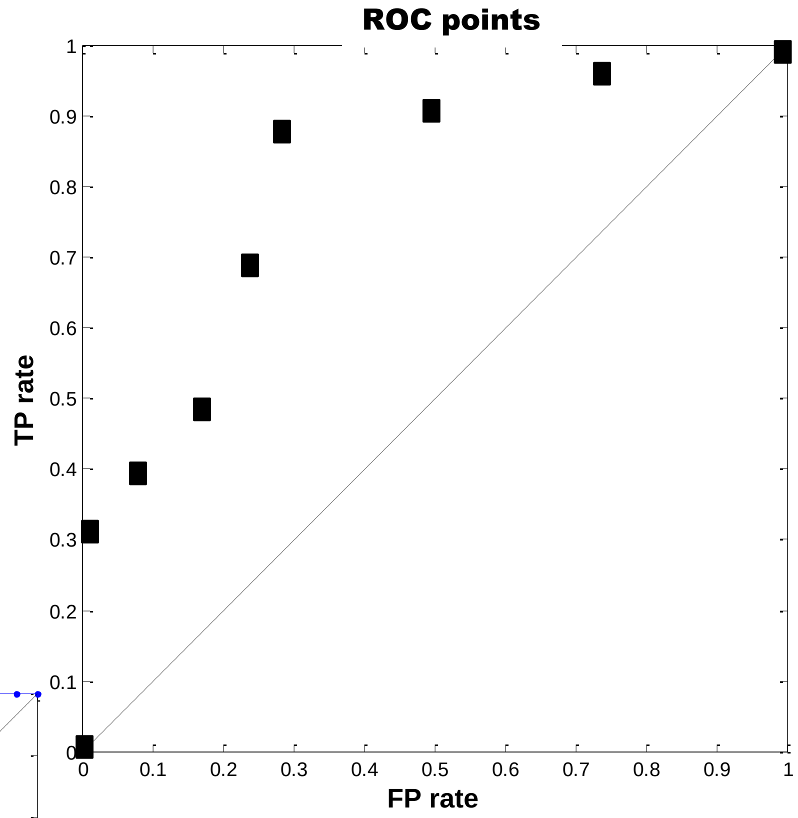
Empirical ROC Curve

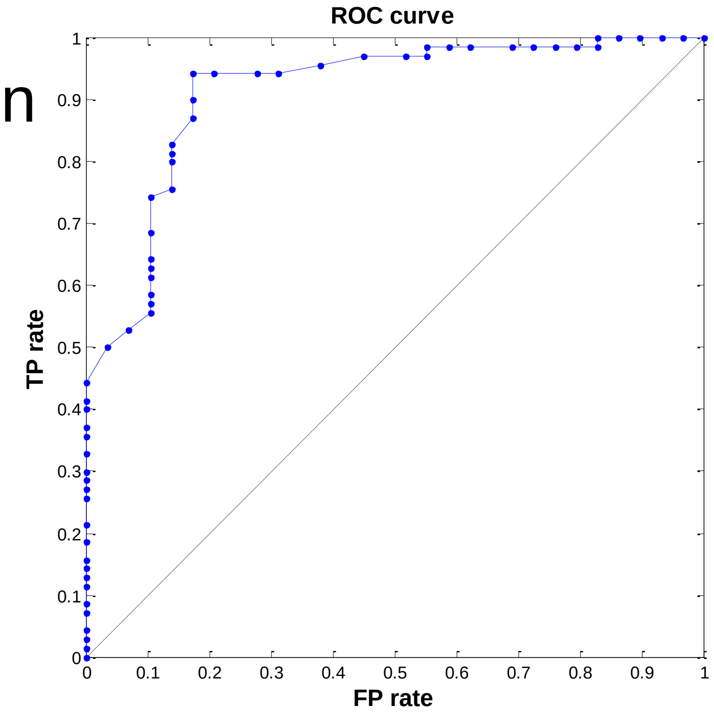
- Vary parameters of classifier:
  - Obtain a set of points in ROC space
  - SVM: Vary parameters of kernel and/or margin
  - *k*-NN: Vary *k*
  - Bayesian/LDR: vary thresholds, weights or dimensions
  - NN: vary hyperparamters
- Use scores from classifier
  - Scikit:
  - https://scikit-learn.org/0.15/auto_examples/plot_roc.html



From Points to ROC Curve

- Points from different classification scenarios
- Smooth or fitted ROC curve
- Connect points using cutoff levels
- Convex hull
- Regression
- Stepwise

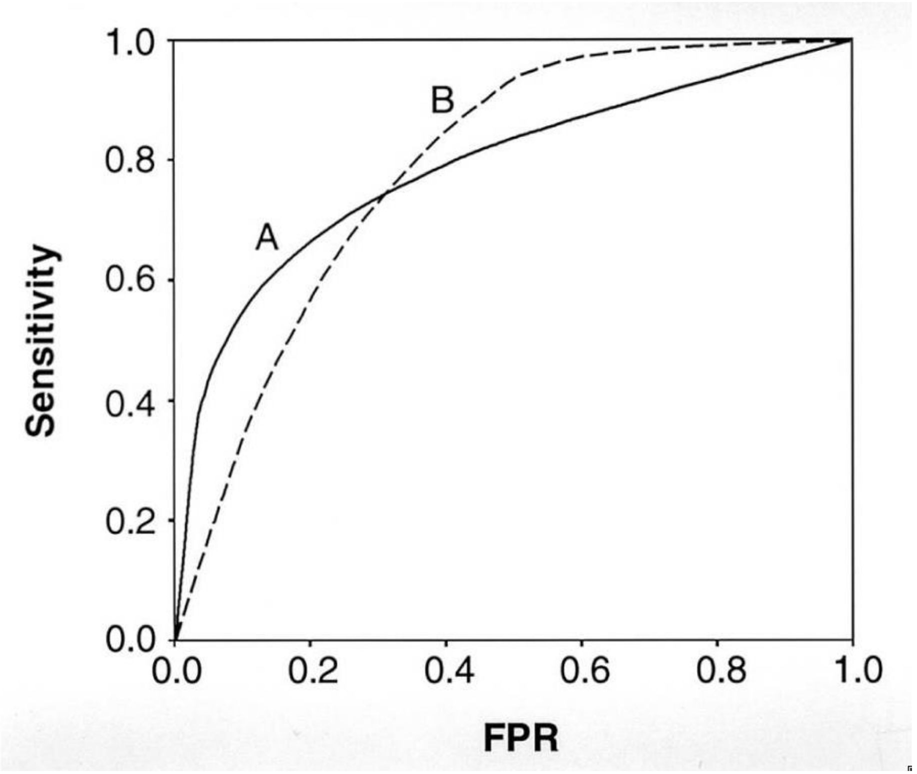




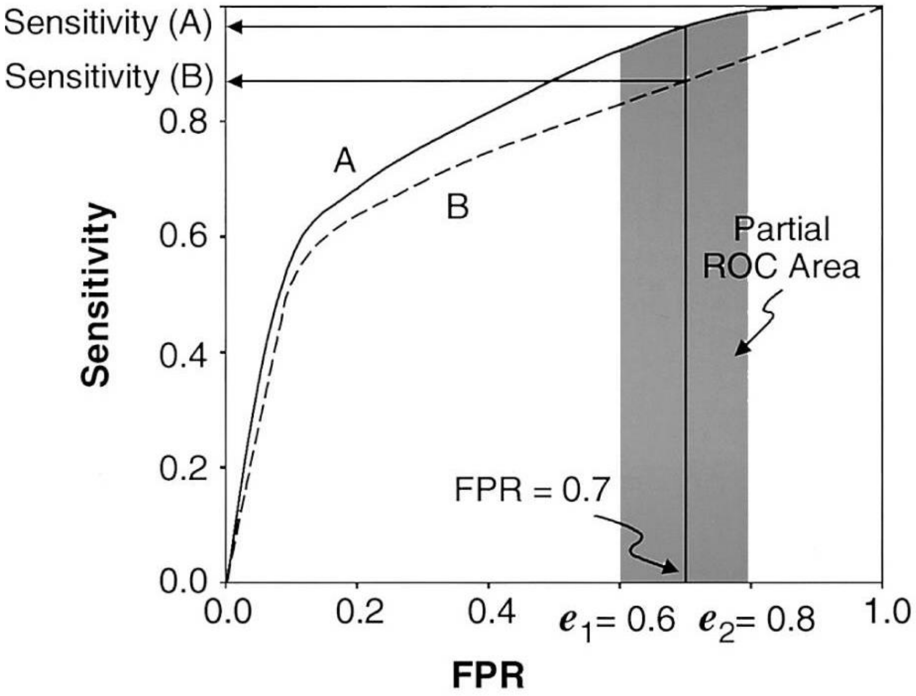
Comparison of Classifiers

Both AUC are equal!

Compare classifiers A and B



Compare sensitivities and AUC



Figures from Park *et al.*

Performance Measures – Multi-class

- a_{ii} is the # of samples classified as ω_j when their **true** class is ω_i
- n is the total number of samples
- n_i is the total number of samples of class i
- True positives: $TP_i = a_{ii}$
- False positives: $FP_i = \sum_{j=1}^{c} a_{ji}$
- False negatives: $FN_i = \sum_{j=1}^{c} a_{ij}$

- Precision:
$$Precision_i = \frac{TP_i}{TP_i + FP_i}$$
- Recall:
$$Recall_i = \frac{TP_i}{TP_i + FN_i}$$
- Accuracy:
$$Acc_i = \frac{a_{ii}}{n_i}$$
- Macro Average Geometric (MAvG):
$$\mathrm{MAvG} = \sum_{i=1}^{c} (Acc_i)^{1/c}$$

Performance Measures – Multi-class

- F-measure of class i :

$$\text{F-measure}_i = \frac{2 \times Recall_i \times Precision_i}{Recall_i + Precision_i}$$

- Mean F-measure (MFM):

$$\text{MFM} = \frac{1}{c} \sum_{i=1}^{c} \text{F} - \text{measure}_i$$

- Macro average accuracy (MAvA):

$$\text{MFM} = \frac{1}{c} \sum_{i=1}^{c} Acc_i$$

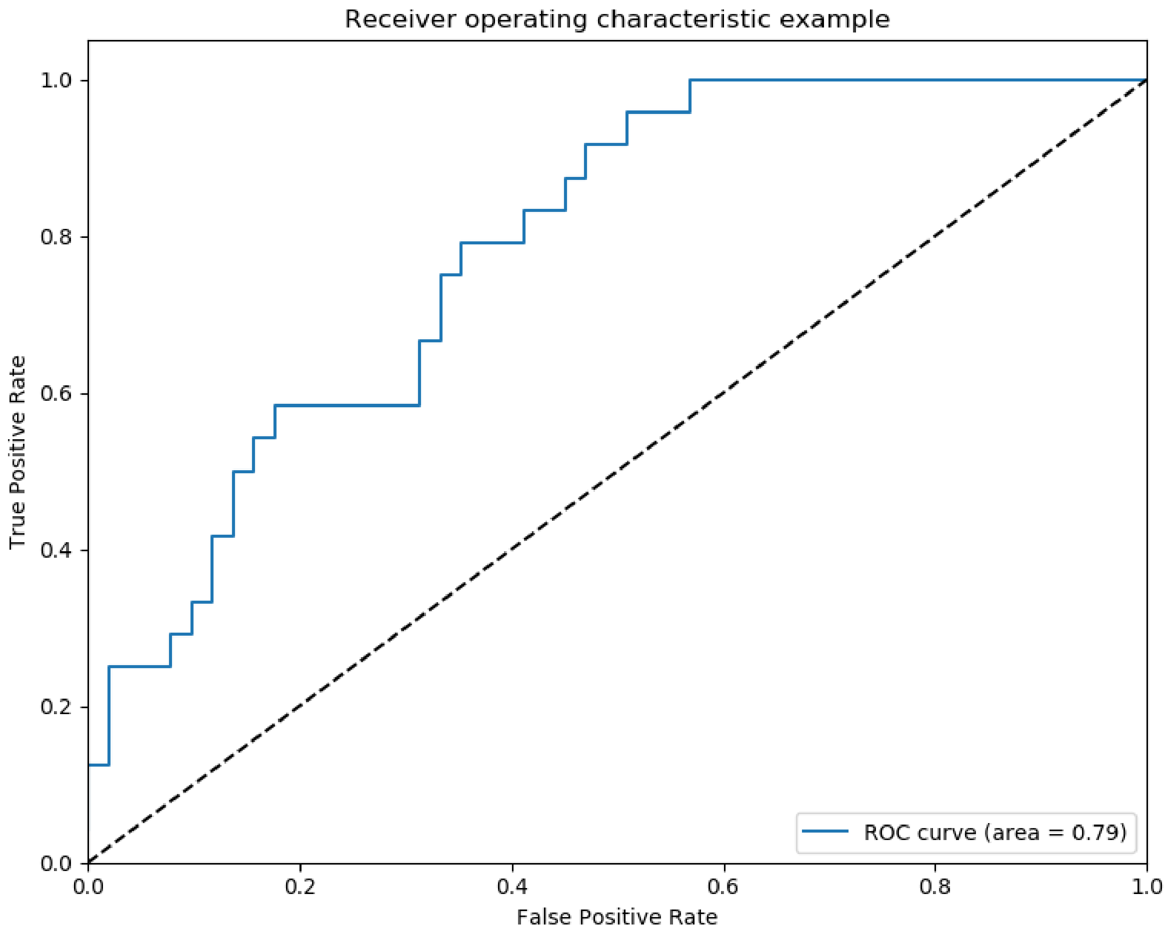
- AUC Uniform (one-against-one) :

$$\text{AUCU} = \frac{2}{c(c-1)} \sum_{\substack{i,j=1 \\ i \neq j}}^{c} AUC_{ij}$$

where AUC_{ij} is the AUC for classes i and j

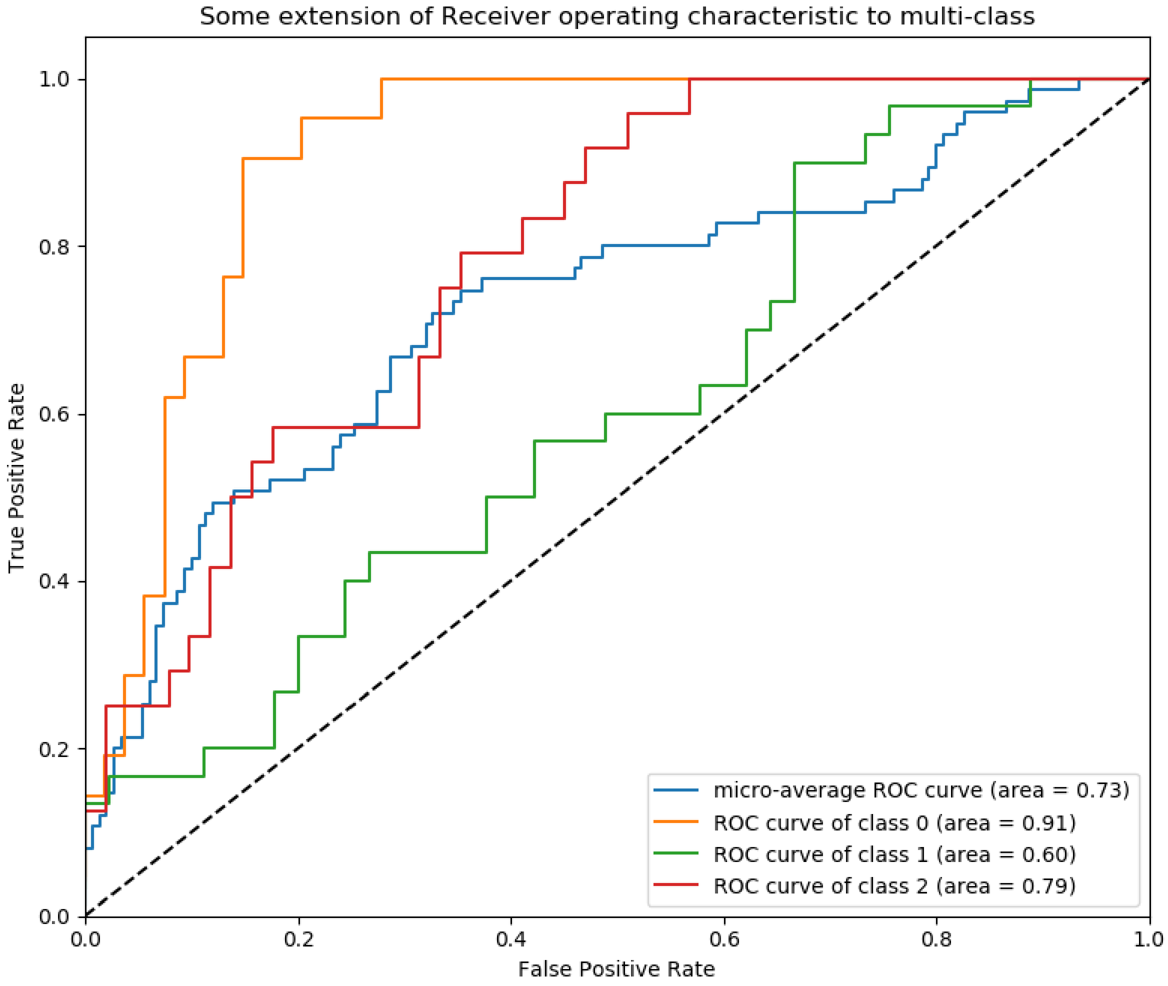
Example: SVM Linear kernel

2 classes



Receiver operating characteristic example

3 classes



Some extension of Receiver operating characteristic to multi-class

References

1. R. Duda et al, Pattern Classification, 2nd Edition, Wiley, 2000.
2. Linear and Quadratic classifiers in Scikit: https://scikit-learn.org/stable/modules/lda_qda.html
3. Naïve Bayes in Scikit: http://scikit-learn.org/stable/modules/naive_bayes.html
4. S. Park, J. Goo, C. Jo. Receiver Operating Characteristic (ROC) Curve: Practical Review for Radiologists, Korean J Radiol, 2004, 5:11-18.
5. T. Fawcett, An Introduction to ROC Analysis, Pattern Recognition Letters, 2006, 861-874.
6. J. Sanchez-Crisostomo et al. Empirical Analysis of Assessments Metrics for Multi-class Imbalance Learning on the Back-Propagation Context. ICSI 2014, Springer LNCS 8795, pp. 17-23.
7. ROC in Scikit: https://scikit-learn.org/0.15/auto_examples/plot_roc.html