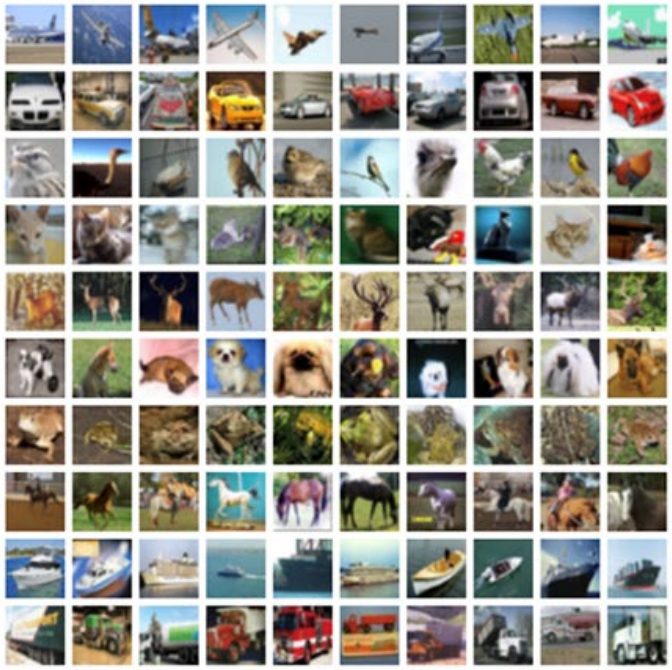
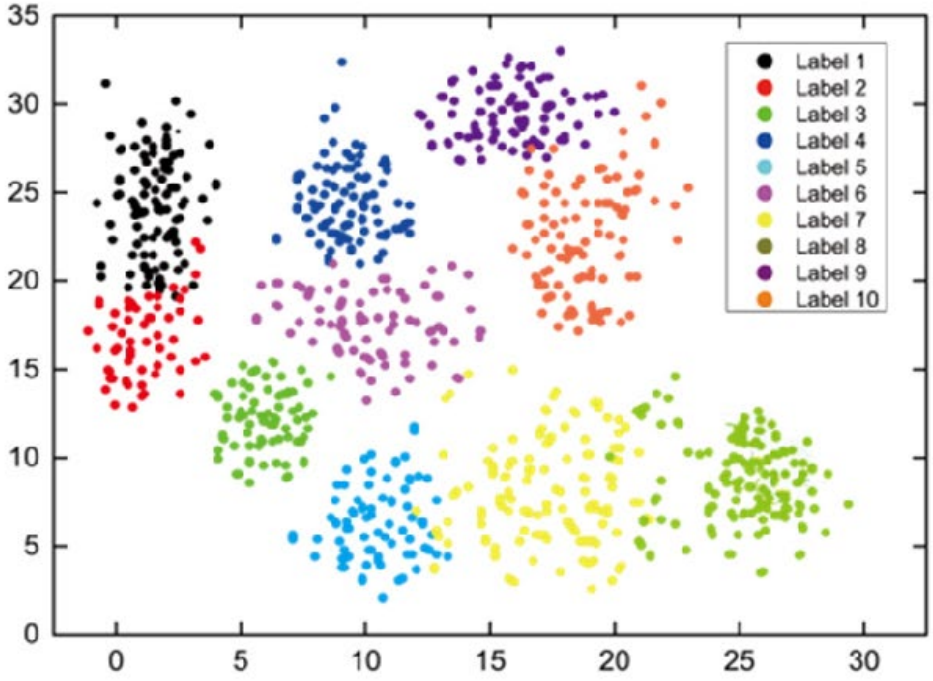
Multiple classes

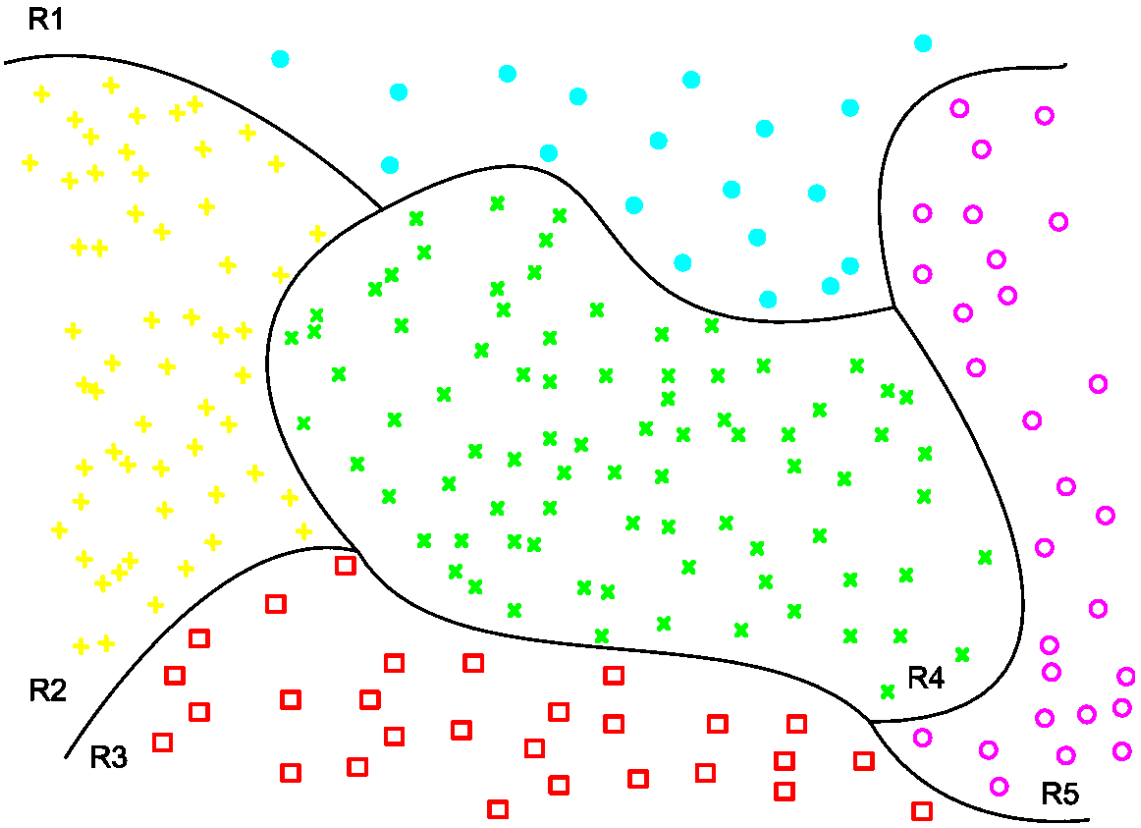


mapping



Classification rule setup

Equivalent to a partitioning of the feature space

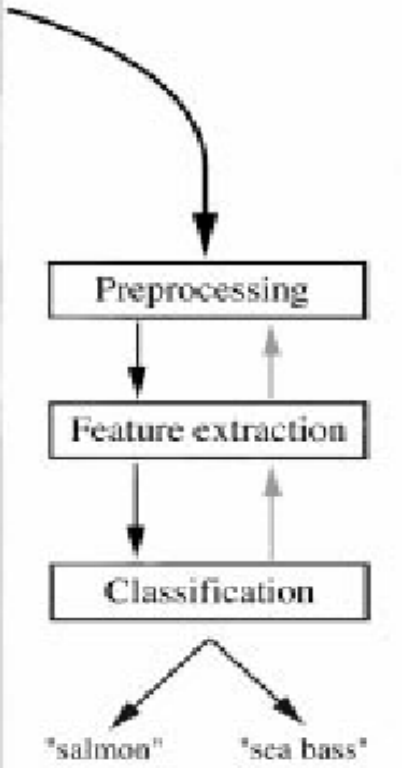


Independent of the particular application

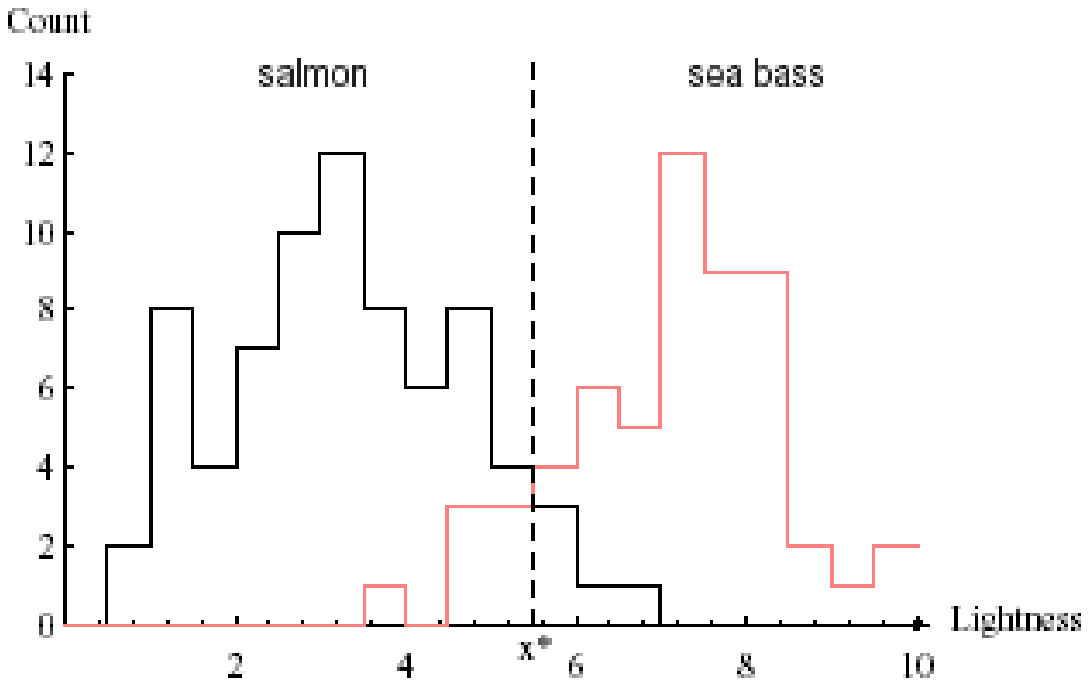
Desirable properties of the training set

- It should contain typical representatives of each class including intra-class variations
- Reliable and large enough
- Should be selected by the domain experts

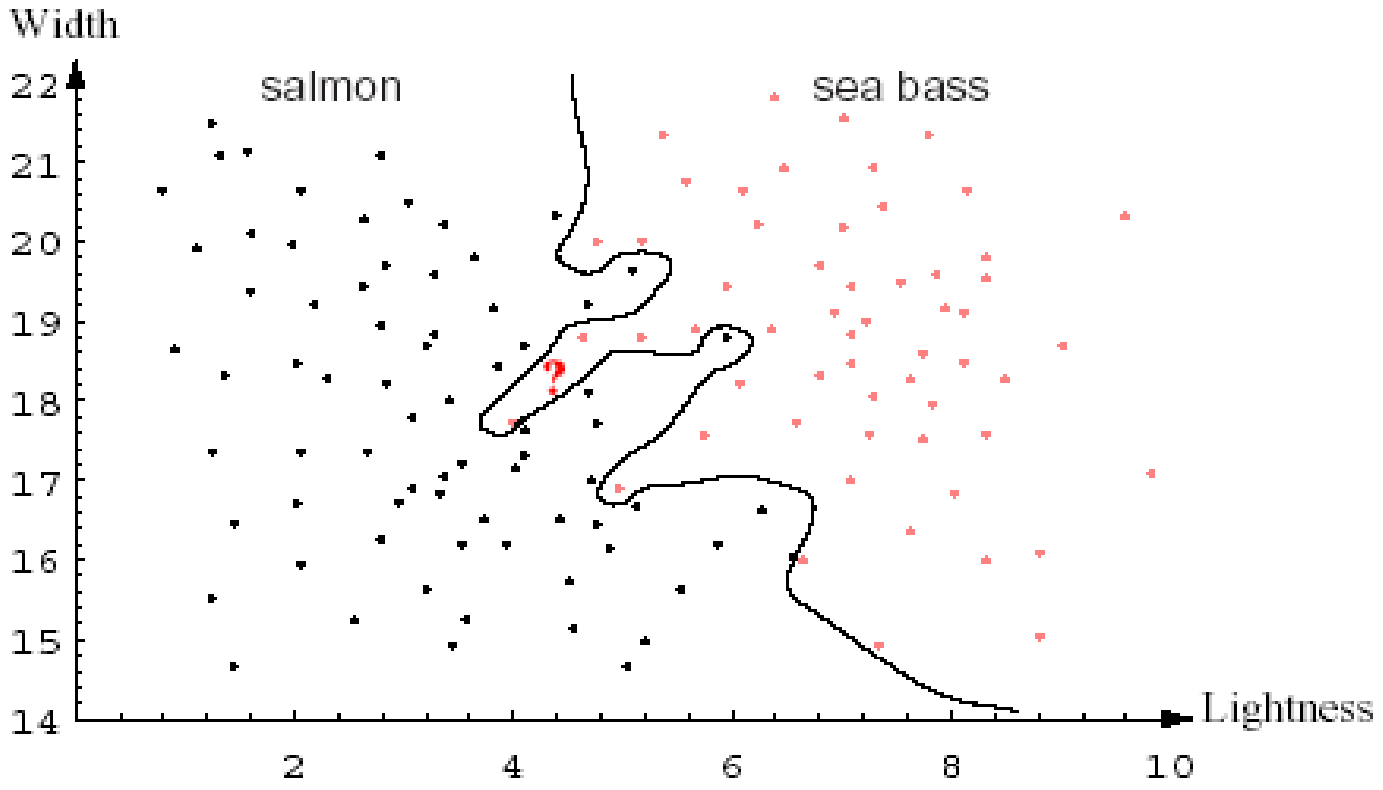
An example – Fish classification



The features: Length, width, brightness



2-D feature space



Empirical observation

- For a given training set, we can have several classifiers (several partitioning of the feature space)

Empirical observation

- For a given training set, we can have several classifiers (several partitioning of the feature space)
- The training samples are not always classified correctly

Empirical observation

- For a given training set, we may have several classifiers (several partitioning of the feature space)
- The training samples are not always classified correctly
- We should avoid overtraining of the classifier

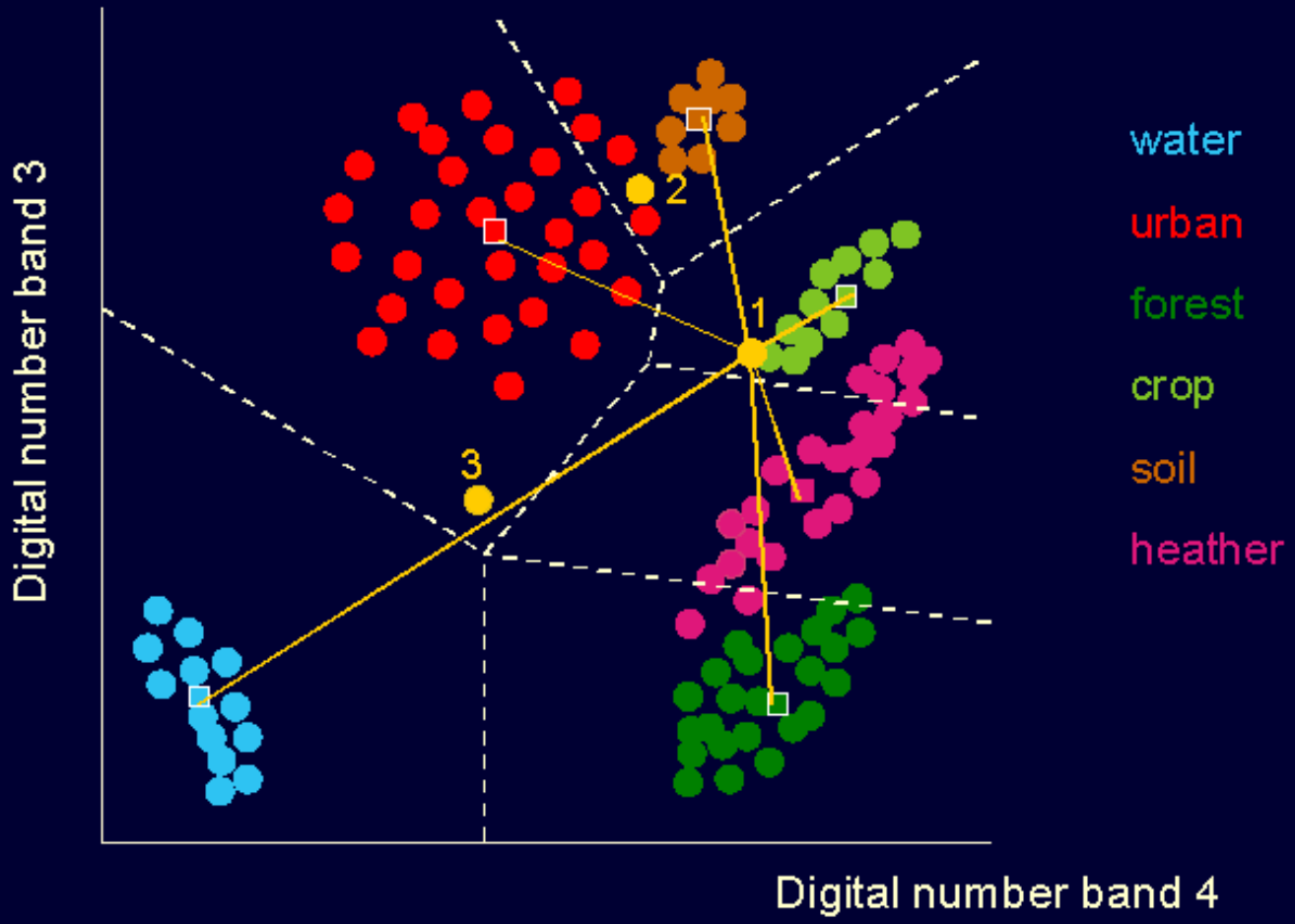
Minimum distance (NN) classifier

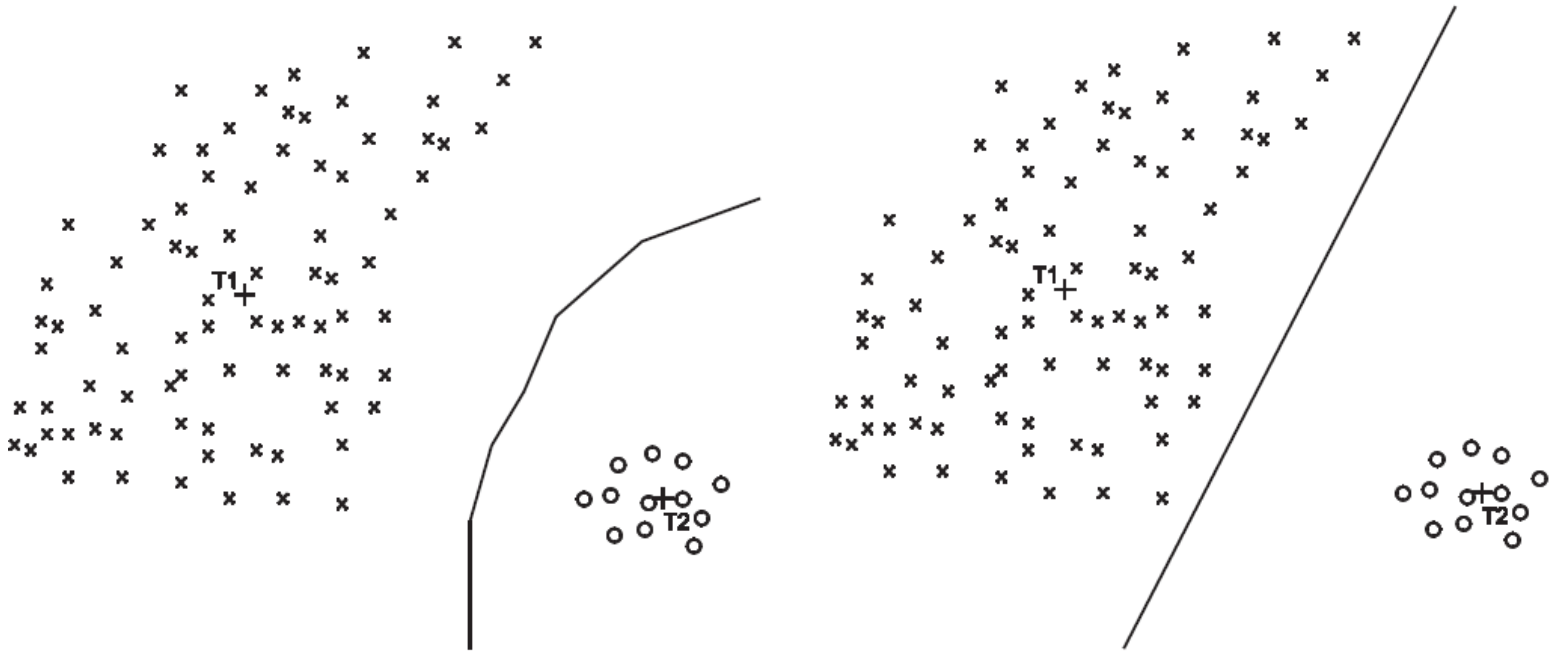
- Depending on $d(\mathbf{x}, \omega_i)$, NN classifier may not be linear

- NN classifier is sensitive to outliers →

  k-NN classifier

Minimum distance to means classification



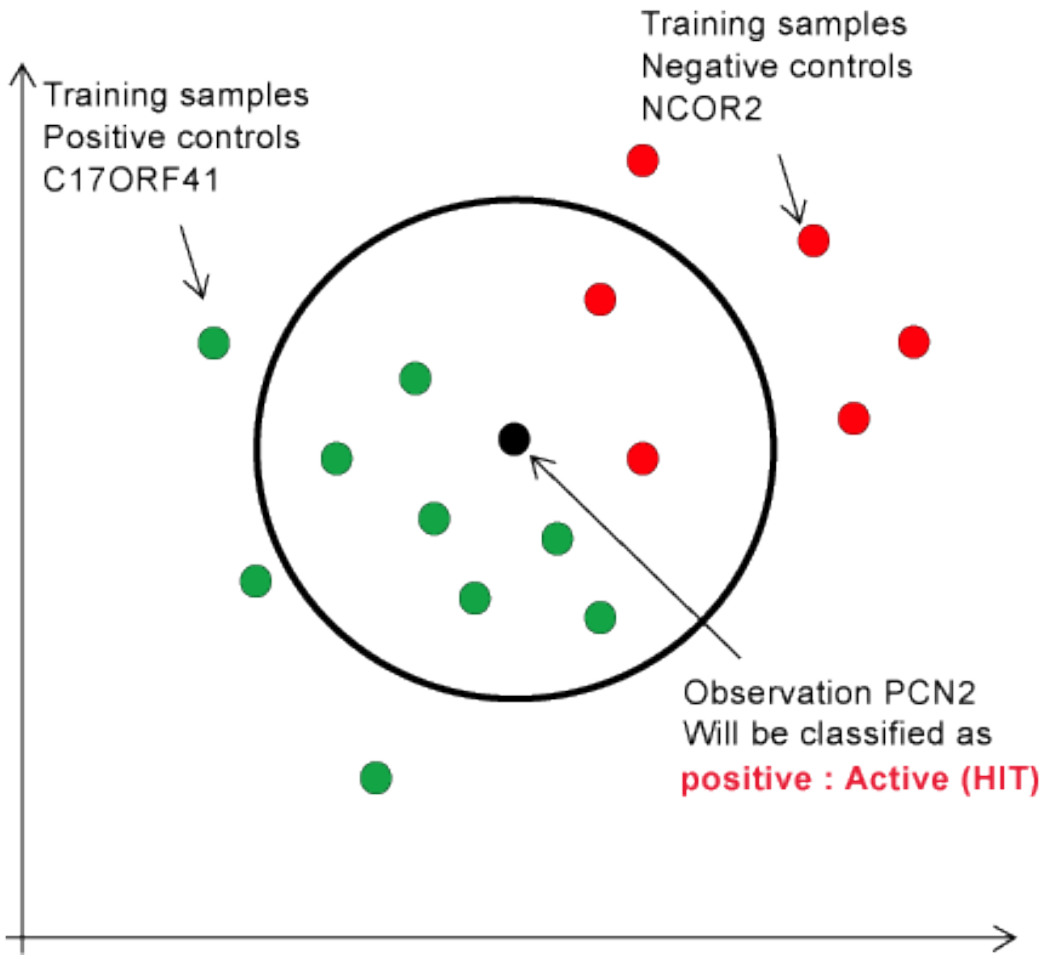

T1
T2
T1
T2

k- NN classifier

- NN classifier is sensitive to outliers → k-NN classifier

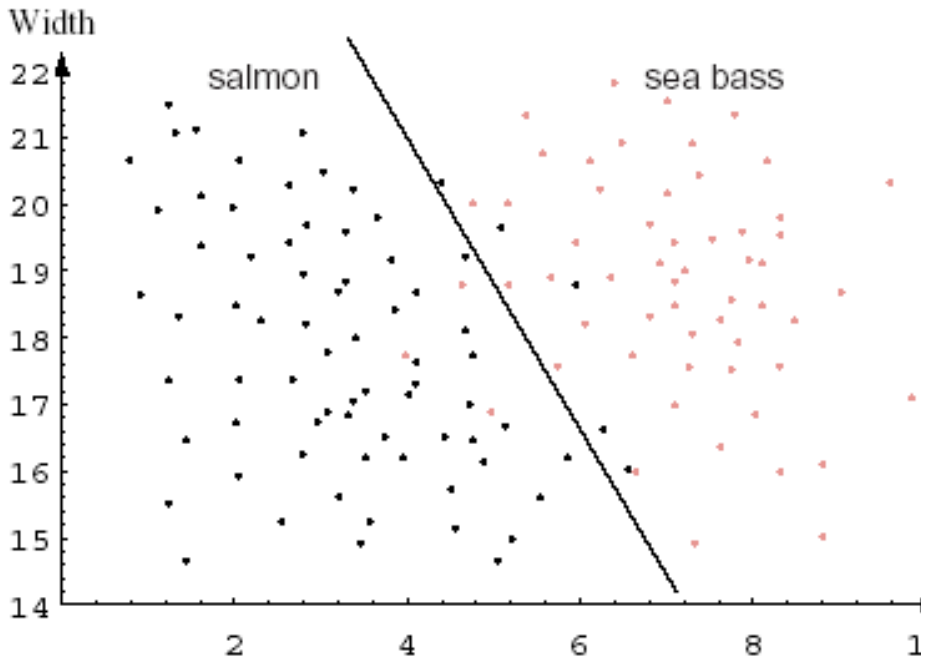
- It finds the nearest training points unless *k* samples belonging to one class has been reached

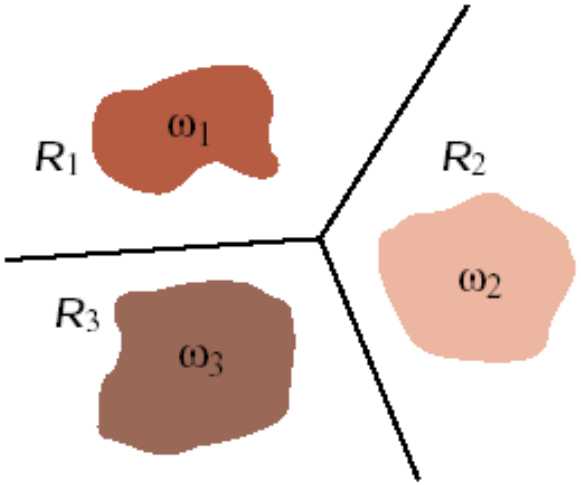
k-NN classifier



Linear classifier

Discriminant functions g(x) are hyperplanes

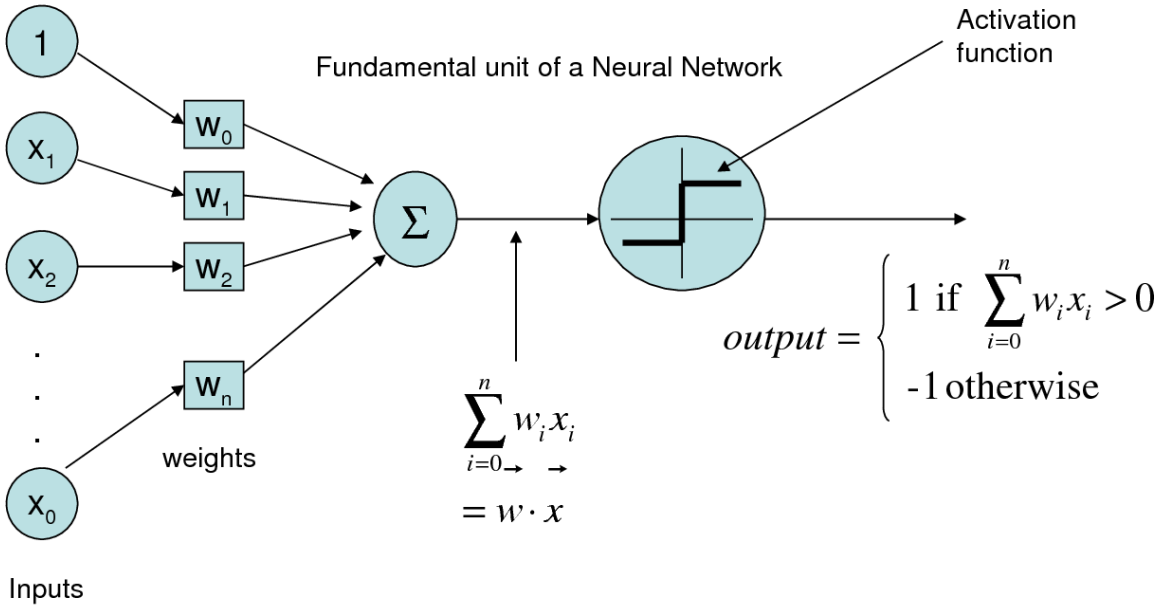




Perceptron

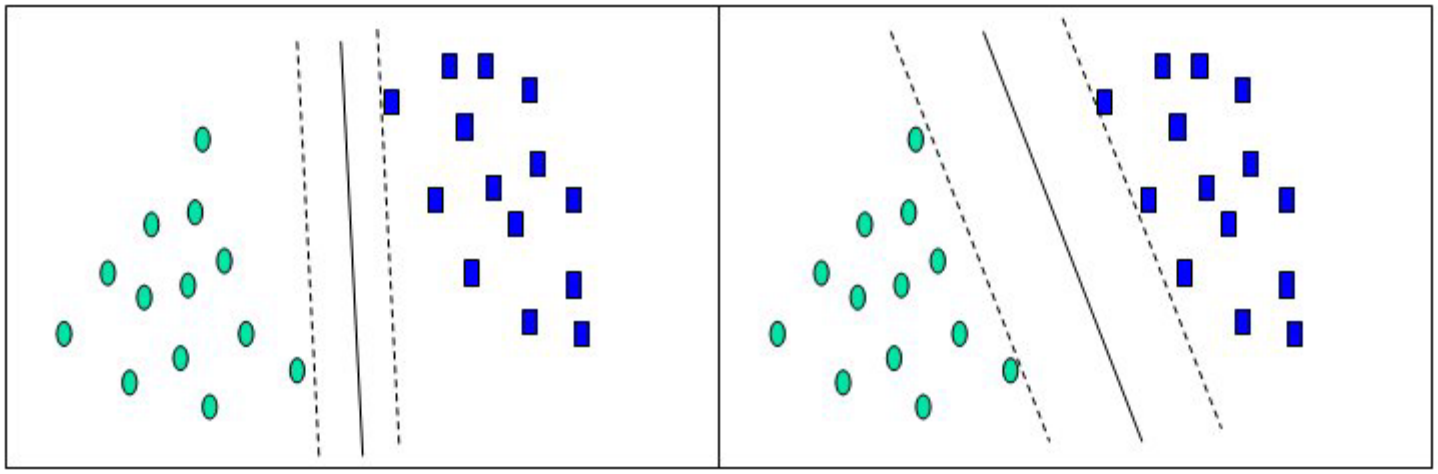
Artificial Neural Networks

The Perceptron



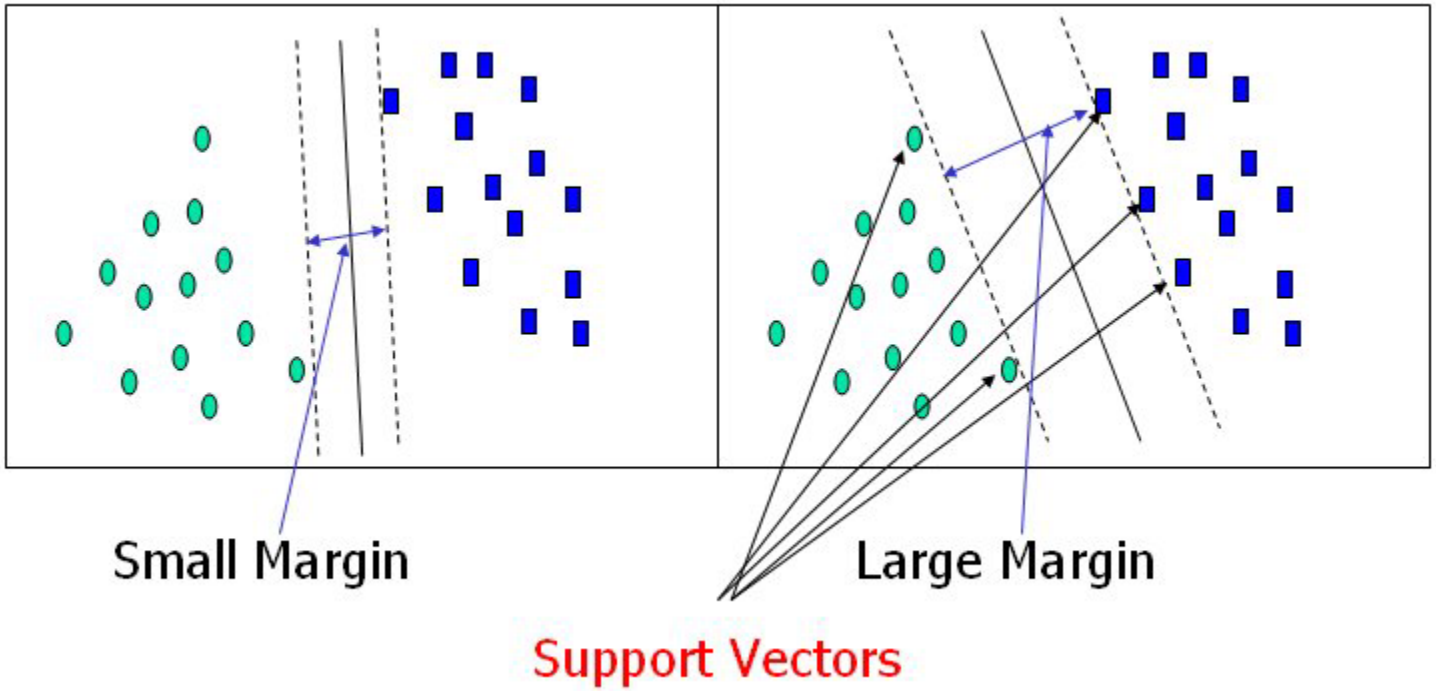
“Optimal” linear classifier:

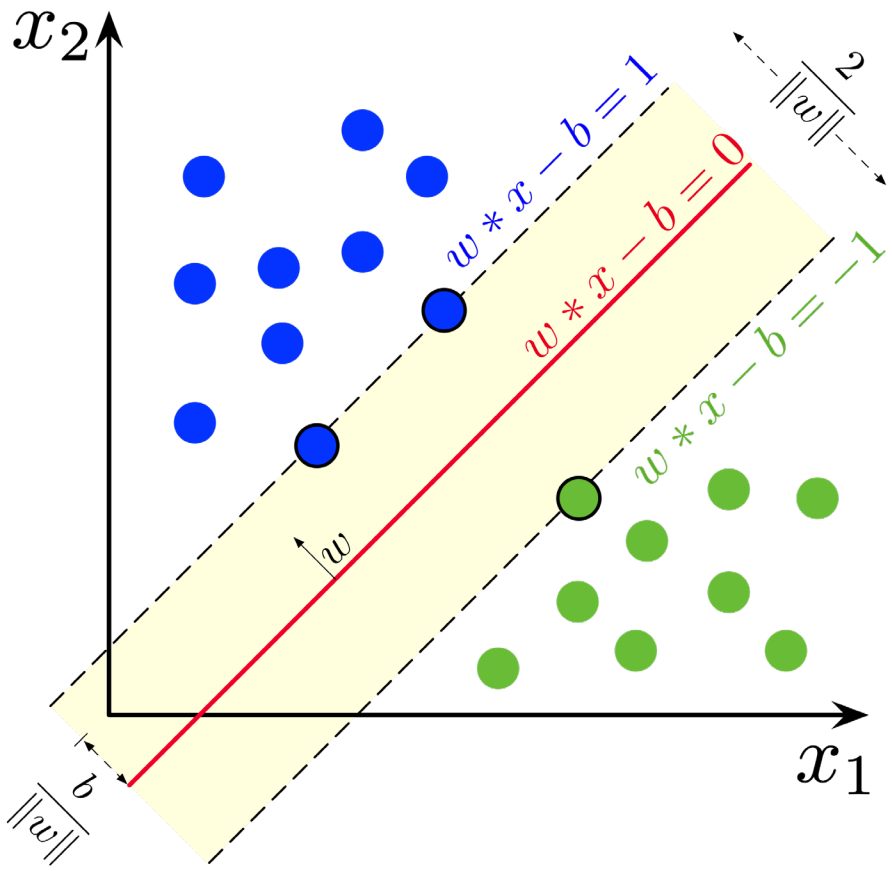
Maximizing the margin



Support vector machine (SVM)

"Optimal" linear classifier

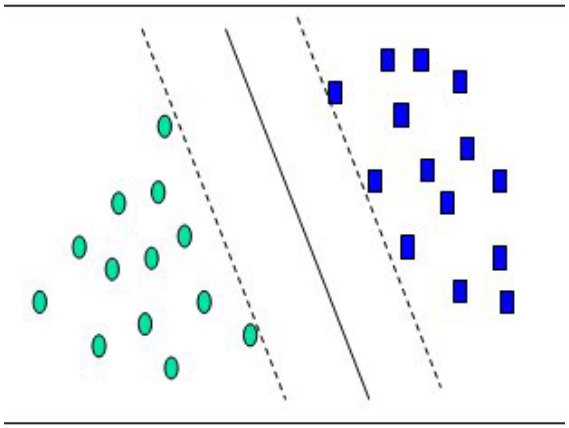



x_2
x_1
$w * x - b = 1$
$w * x - b = 0$
$w * x - b = -1$
w
$\frac{2}{\|w\|}$
$\frac{b}{\|w\|}$

Support vector machine (SVM)

Training algorithm: Discrete optimization

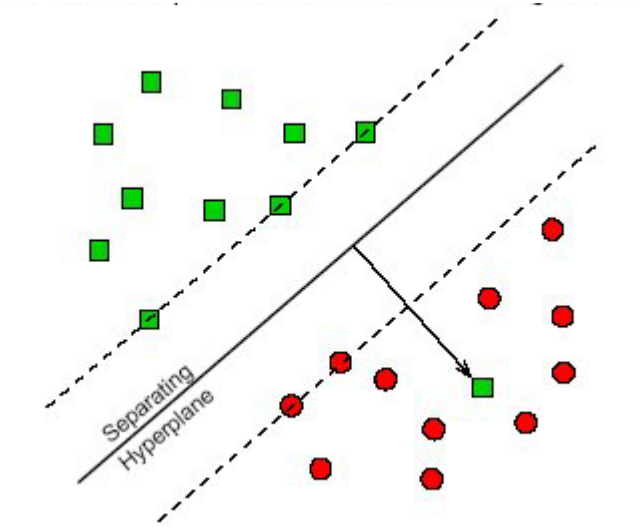
- many versions



Drawbacks:

- very sensitive to outliers and noise
- does not consider the shape and the size of the training set

Admitting training errors



Cost function penalizes the errors

Attn: Definition of weights

Bayesian classifier

Assumption: feature values are random variables

Statistic classifier, the decision is probabilistic

It is based on the **Bayes rule**

Bayesian classifier

Main idea: maximize posterior probability

$$P(\omega_j|\mathbf{x})$$

Since it is hard to do directly, we rather maximize

$$p(\mathbf{x}|\omega_j)P(\omega_j)$$

In case of equal priors, we maximize only

$$p(\mathbf{x}|\omega_j)$$

The Bayes rule

Class-conditional probability

A posteriori probability

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

A priori probability

Total probability

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

How to estimate $p(\mathbf{x}|\omega_j)P(\omega_j)$?

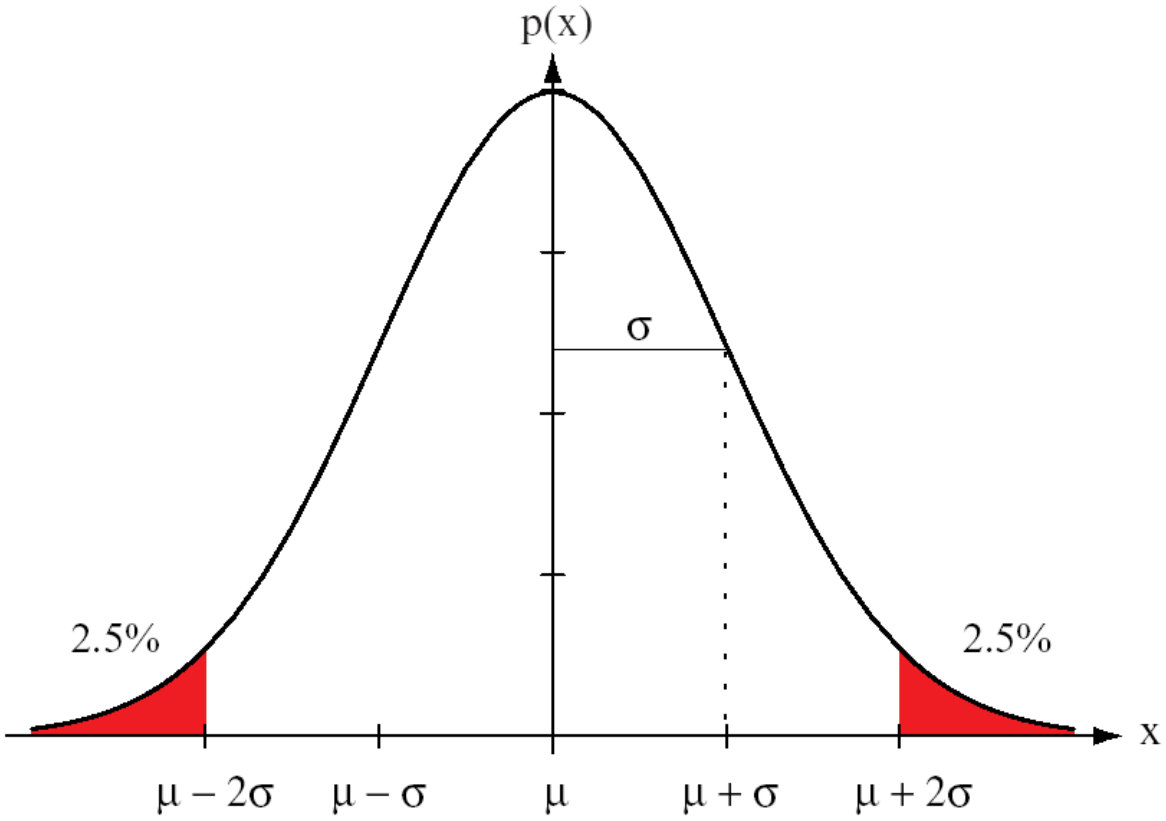
$P(\omega_j)$

- From the case studies performed before (OCR, speech recognition)
- From the occurence in the training set
- Assumption of equal priors

$p(\mathbf{x}|\omega_j)$

- Parametric estimate (assuming pdf is of a known form, e.g. Gaussian)
- Non-parametric estimate (pdf is unknown or too complex)

Parametric estimate of Gaussian $p(\mathbf{x}|\omega_j)$.

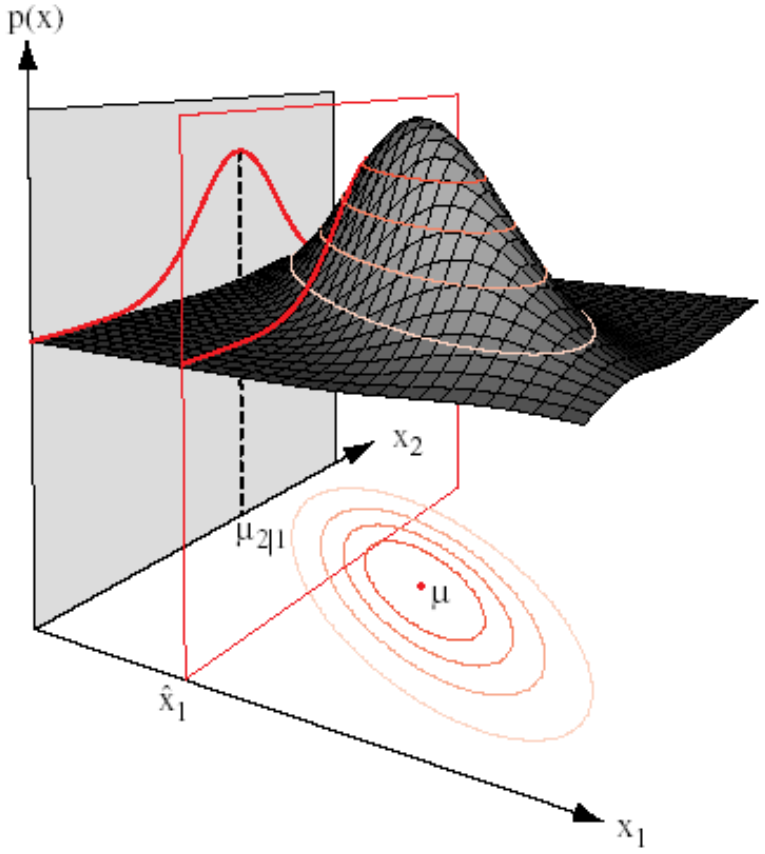


$$\hat{\mu} = \frac{1}{n}\sum_{k=1}^{n} x_k$$

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

$$p(x) = \frac{1}{\sqrt{2\pi}\sigma}\exp\left[-\frac{1}{2}\left(\frac{x-\mu}{\sigma}\right)^2\right]$$

d-dimensional Gaussian pdf

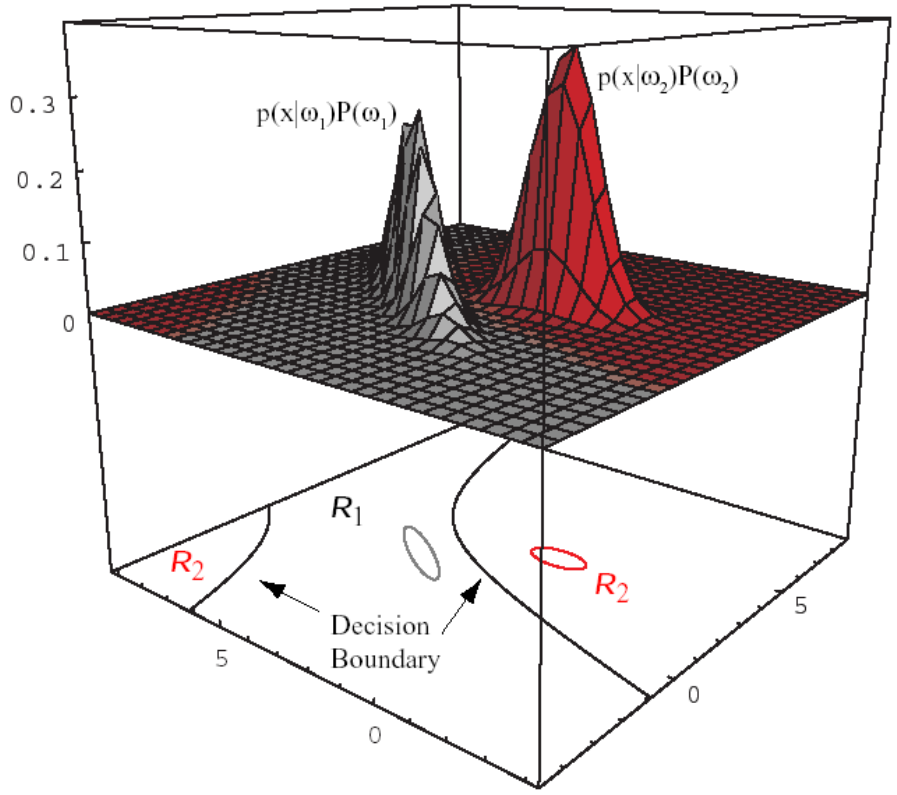


$$\hat{\boldsymbol{\mu}} = \frac{1}{n}\sum_{k=1}^{n}\mathbf{x}_k$$

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

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

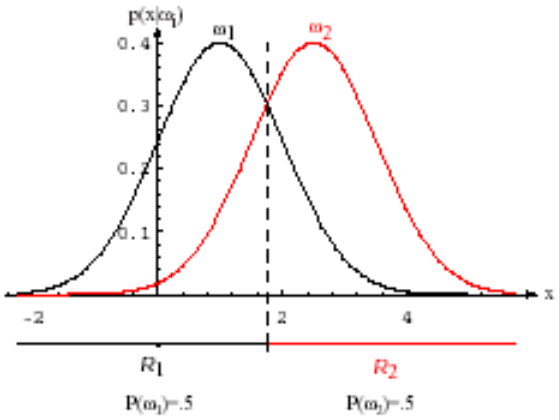
Two-class Gaussian case in 2D

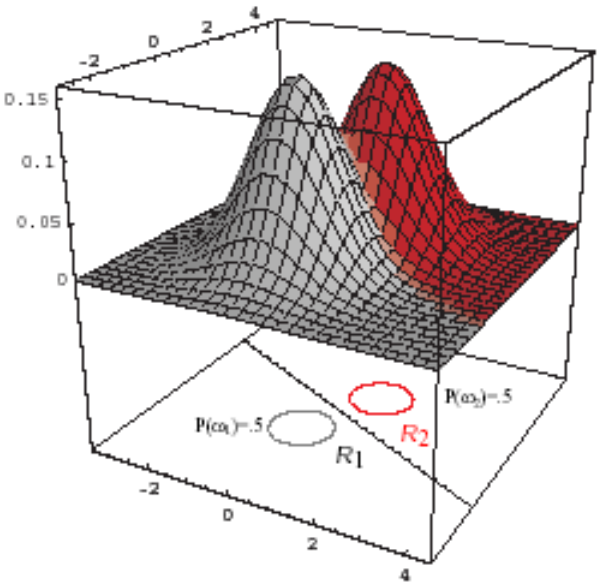


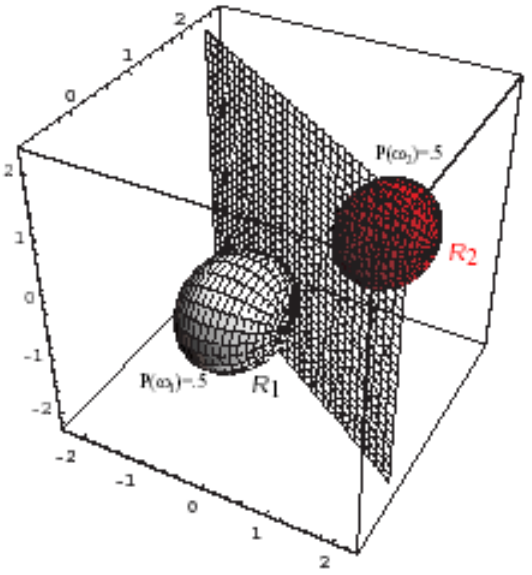
$$g_i(\mathbf{x}) = -\frac{1}{2}(\mathbf{x}-\boldsymbol{\mu}_i)^t\boldsymbol{\Sigma}_i^{-1}(\mathbf{x}-\boldsymbol{\mu}_i) - \frac{d}{2}\ln 2\pi - \frac{1}{2}\ln|\boldsymbol{\Sigma}_i| + \ln P(\omega_i).$$

Classification = comparison of two Gaussians

Two-class Gaussian case – Equal cov. mat.







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

Linear decision boundary

Equal priors $P(\omega_j)$

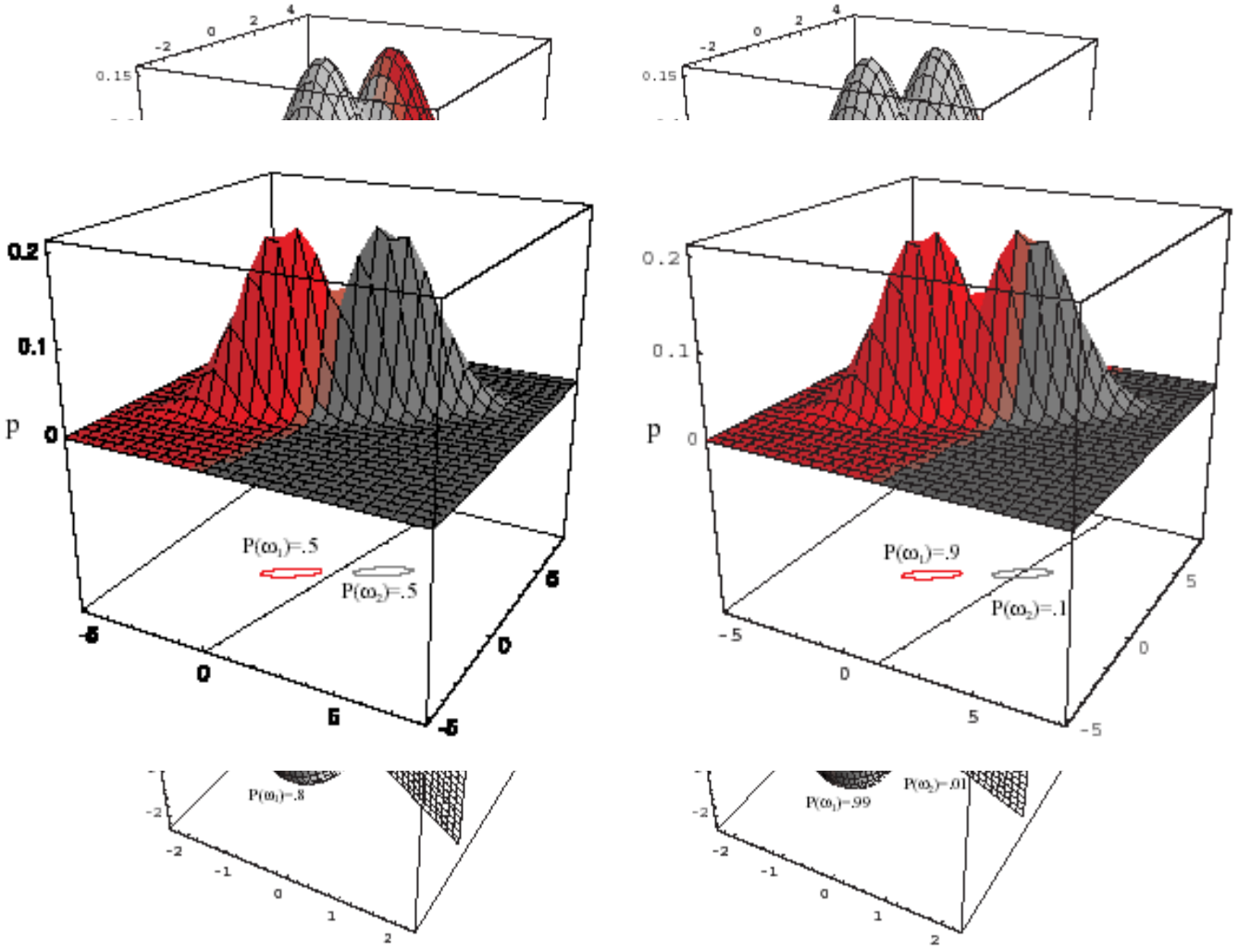
max $g_i(\mathbf{x}) = -\frac{1}{2}(\mathbf{x}-\boldsymbol{\mu}_i)^t\boldsymbol{\Sigma}^{-1}(\mathbf{x}-\boldsymbol{\mu}_i)$

min $(\mathbf{x}-\boldsymbol{\mu}_i)^t\boldsymbol{\Sigma}^{-1}(\mathbf{x}-\boldsymbol{\mu}_i)$

Classification by minimum Mahalanobis distance

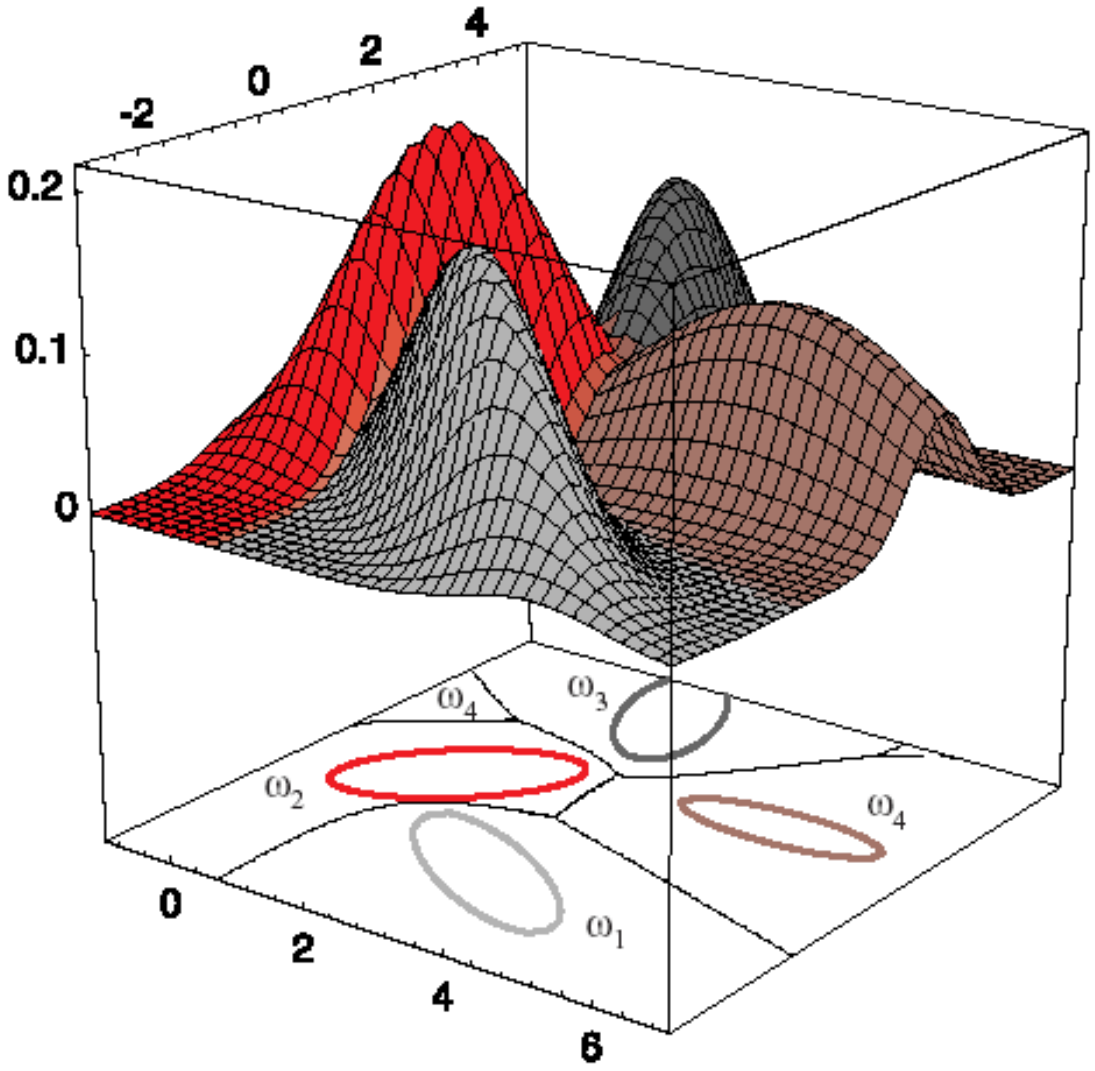
If the cov. mat. is diagonal with equal variances then we get “standard” minimum distance rule

Non-equal priors $P(\omega_j)$



Linear decision boundary still preserved

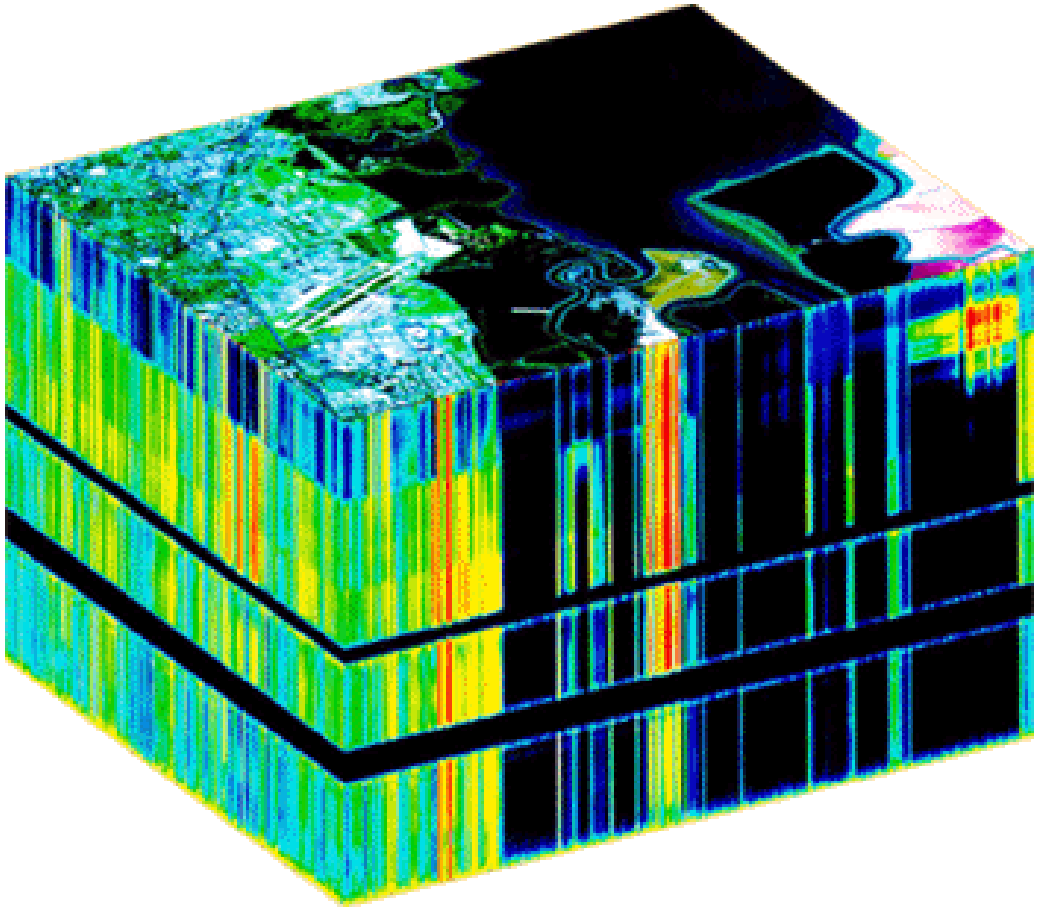
More classes, Gaussian case in 2D



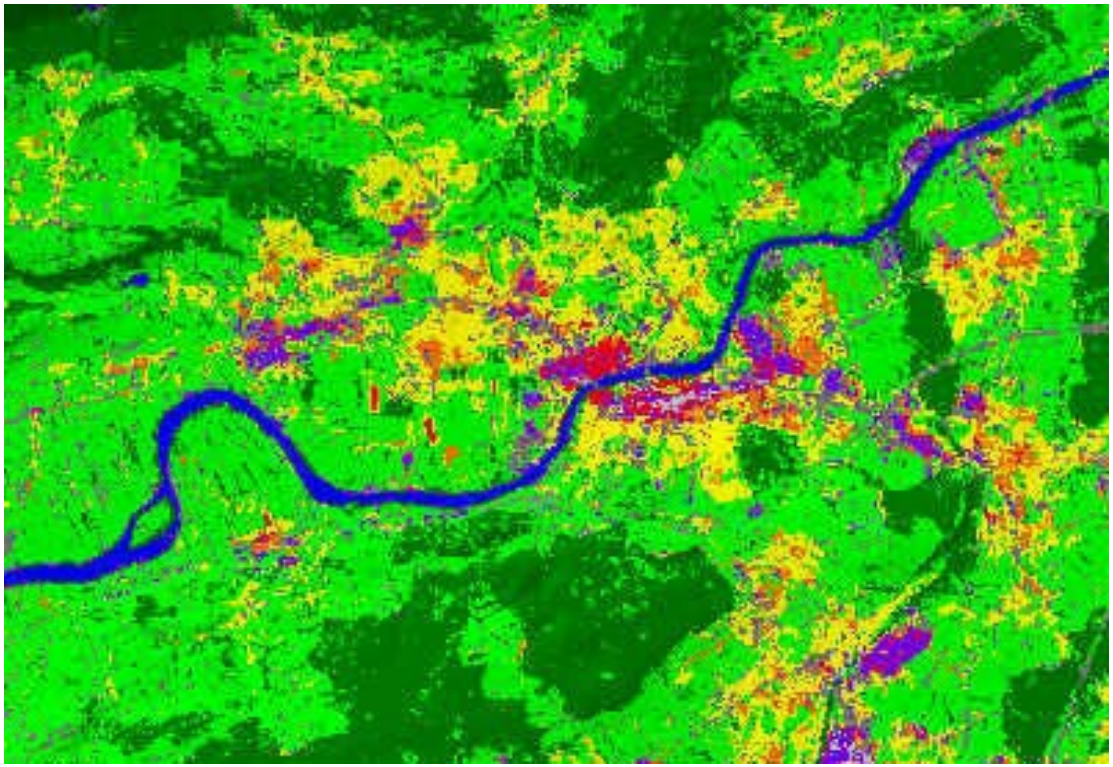
Applications of Bayesian classifier in multispectral remote sensing

- Objects = pixels
- Features = pixel values in the spectral bands (from 4 to several hundreds)
- Training set – selected manually by means of thematic maps (GIS), and on-site observation
- Number of classes – typically from 2 to 16

Applications of Bayesian classifier in multispectral remote sensing



Applications of Bayesian classifier in multispectral remote sensing



Classification performance

- **How to evaluate the performance of the classifiers?**
  - evaluation on the training set (optimistic error estimate)
  - evaluation on the test set (pessimistic error estimate). Intuitive evaluation is misleading, different errors may have different significance

Confusion matrix

		Predicted		
		Greyhound	Mastiff	Samoyed
Actual	Greyhound	250	25	18
	Mastiff	21	320	24
	Samoyed	22	12	180

Confusion matrix 2 x 2

Machine learing \ Manual counting	True	False
True	True Positive (TP)	False Positive (FP)
False	False Negative (FN)	True Negative (TN)

Equations:

$$\text{False positive rate (FPR)} = \frac{FP}{FP+TN}$$

$$\text{False negative rate (FNR)} = \frac{FN}{FN+TP}$$

$$\text{Sensitivity} = \frac{TP}{TP+FN}$$

$$\text{Specificity} = \frac{TN}{TN+FP}$$

$$\text{Youden index} = \text{Sensitivity} + \text{Specificity} - 1$$

$$\text{Accuracy} = \frac{TP+TN}{TP+TN+FP+FN}$$

Thank you !

Any questions ?