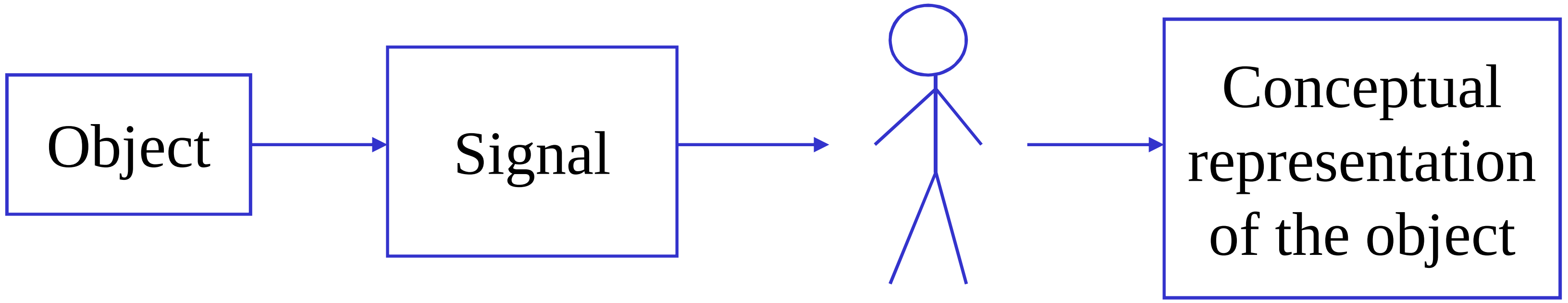
Pattern Recognition.

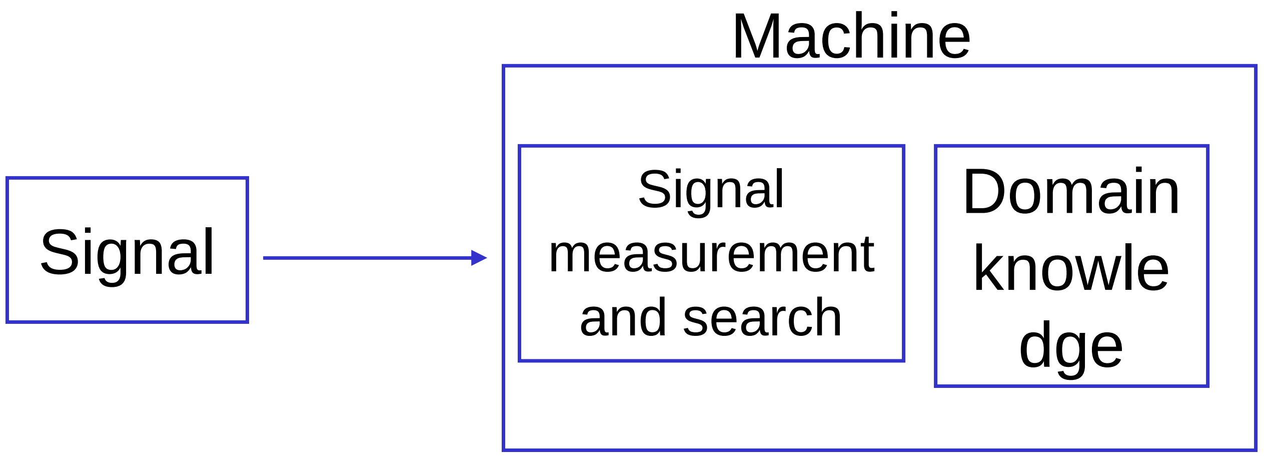
Introduction. Definitions.

Recognition process.

- **Recognition process relates input signal to the stored concepts about the object.**



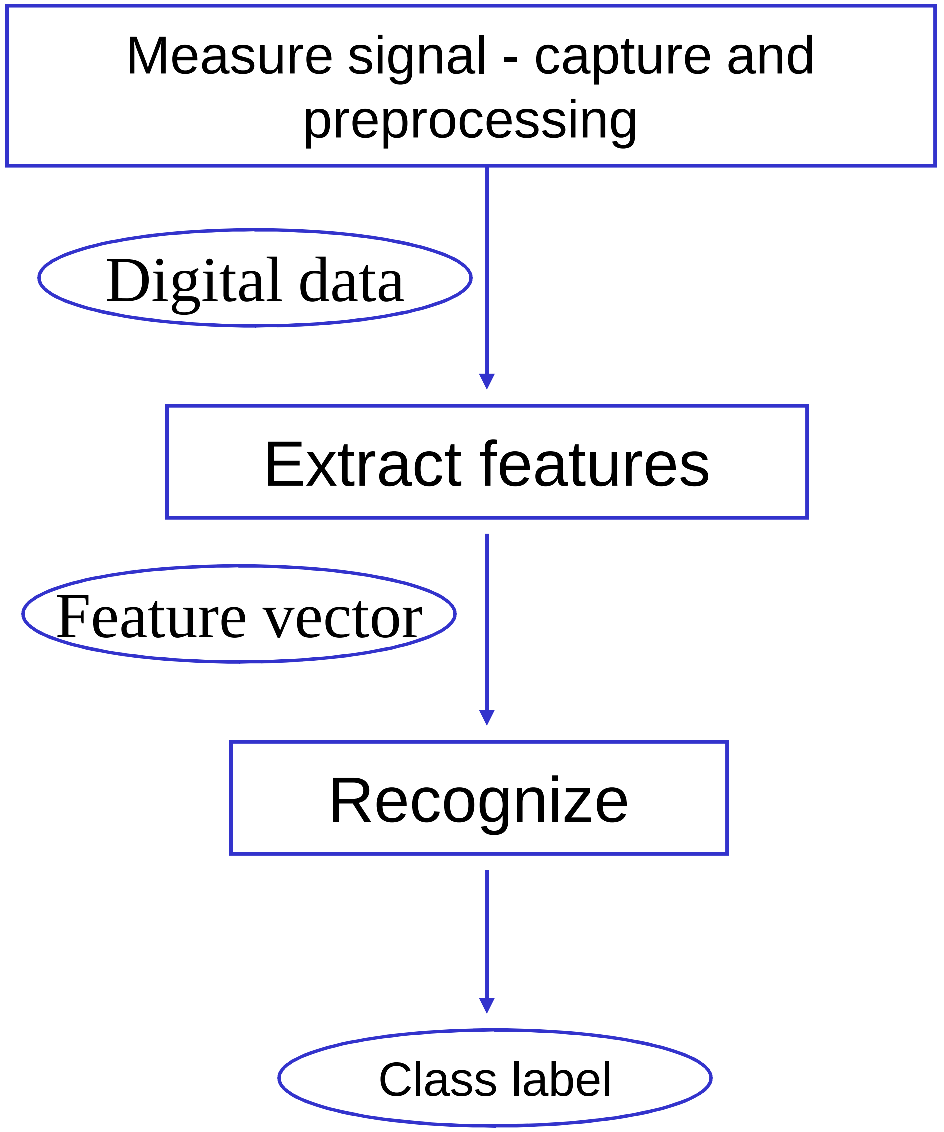
- **Machine recognition relates signal to the stored domain knowledge.**



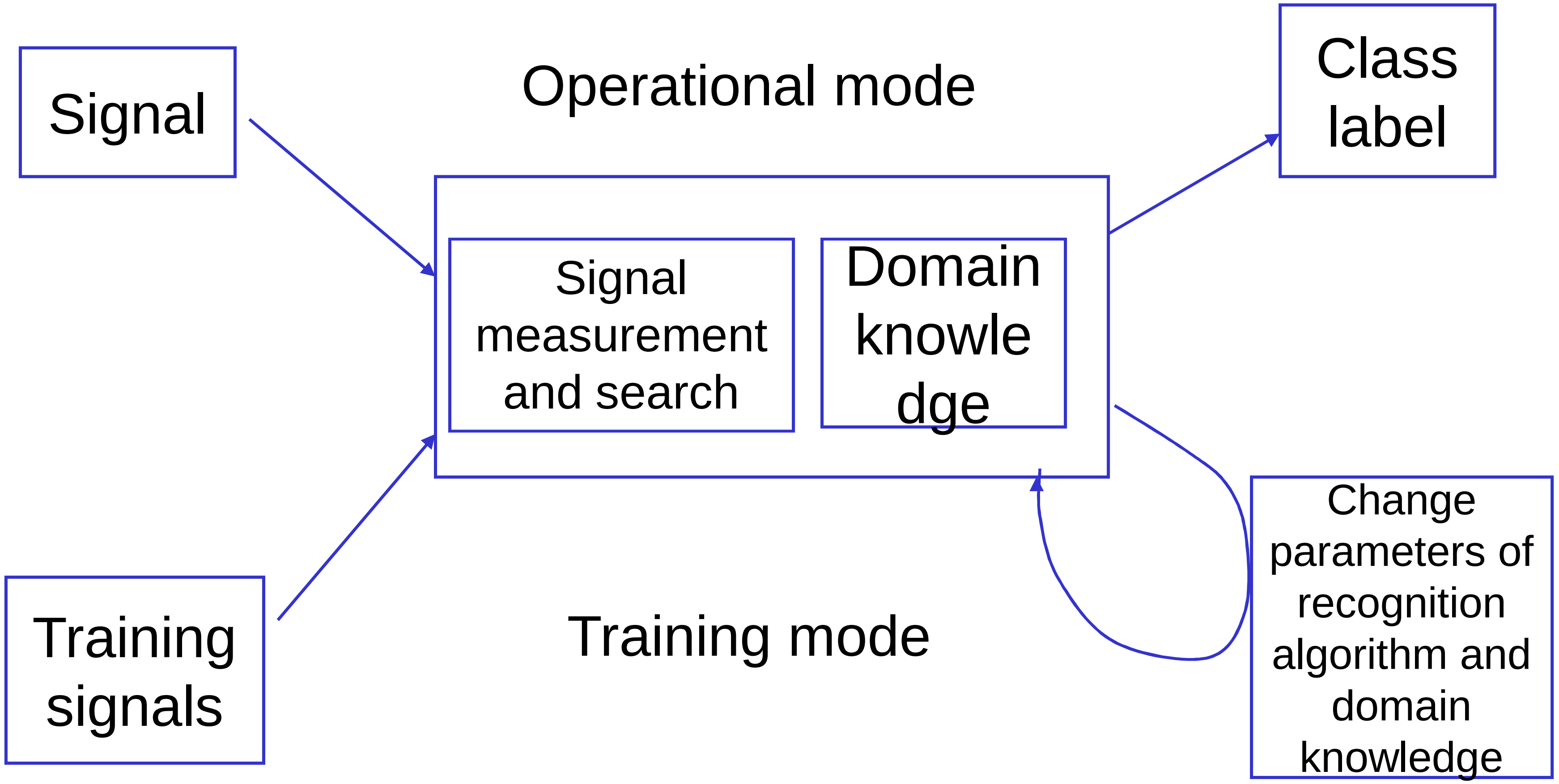
Definitions.

- Similar objects produce similar signals.
- <u>Class</u> is a set of similar objects.
- <u>Patterns</u> are collections of signals originating from similar objects.
- <u>Pattern recognition</u> is the process of identifying signal as originating from particular class of objects.

Pattern recognition steps.



Training of the recognizer.


Signal
Operational mode
Class label
Signal measurement and search
Domain knowle dge
Training signals
Training mode
Change parameters of recognition algorithm and domain knowledge

Types of Training

• Supervised training – uses training samples with associated class labels.

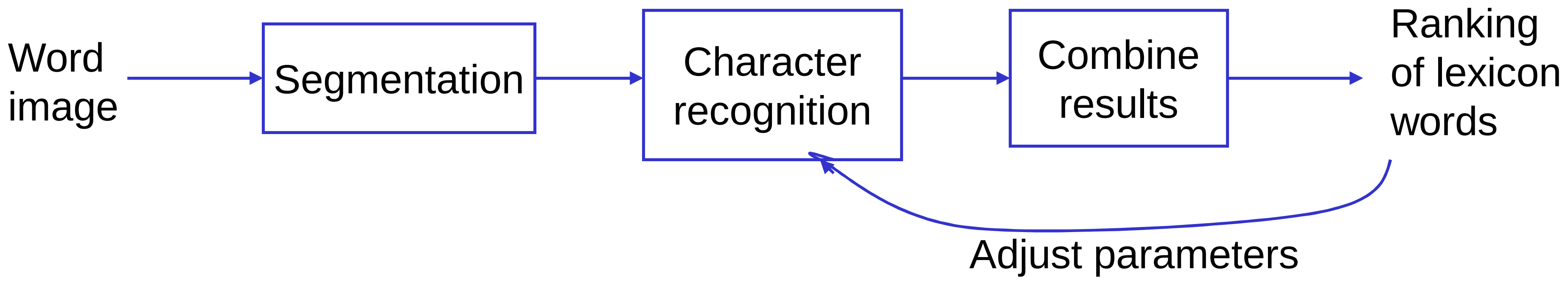
-Character images with corresponding labels.

• Unsupervised training – training samples are not labeled.

-Character images: cluster images and assign labels to clusters later.

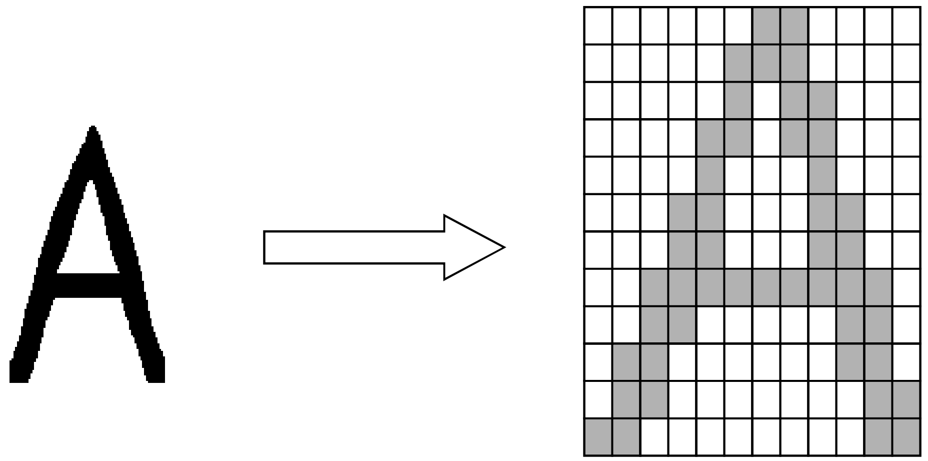
• Reinforcement training – feedback is provided during recognition to adjust system parameters.

- Use word images to train character recognizer.



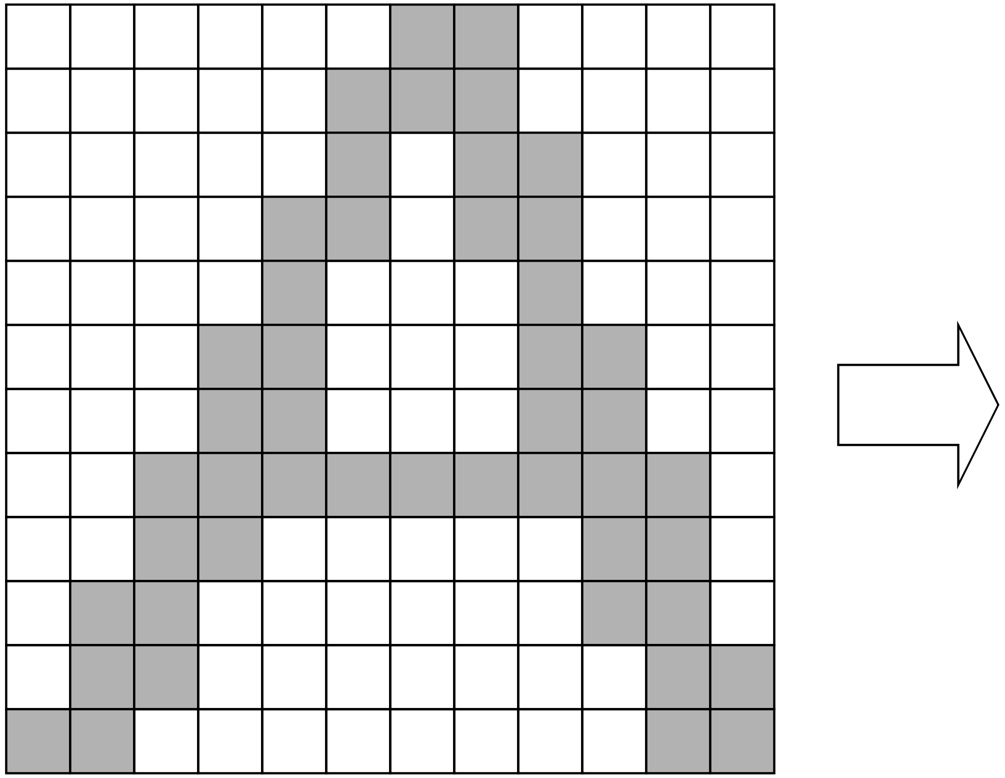
Template Matching(1)

Image is converted into 12x12 bitmap.



Template Matching(2)

Bitmap is represented by 12x12-matrix or by 144-vector with 0 and 1 coordinates.



0	0	0	0	0	0	1	1	0	0	0	0
0	0	0	0	0	1	1	1	0	0	0	0
0	0	0	0	0	1	0	1	1	0	0	0
0	0	0	0	1	1	0	1	1	0	0	0
0	0	0	0	1	0	0	0	1	0	0	0
0	0	0	1	1	0	0	0	1	1	0	0
0	0	0	1	1	0	0	0	1	1	0	0
0	0	1	1	1	1	1	1	1	1	1	0
0	0	1	1	0	0	0	0	0	1	1	0
0	1	1	0	0	0	0	0	0	1	1	0
0	1	1	0	0	0	0	0	0	0	1	1
1	1	0	0	0	0	0	0	0	0	1	1

Template Matching(3)

Training samples – templates with corresponding class:

$$t_1 = \{(0,0,0,0,1,1,...,0), \text{'A'}\}$$

$$t_2 = \{(0,0,0,0,0,1,...,0), \text{'A'}\}$$

....

$$t_k = \{(0,0,1,1,1,1,...,0), \text{'B'}\}$$

....

Template of the image to be recognized:

$$T = \{(0,0,0,0,1,1,...,0), \text{'A'}\}$$

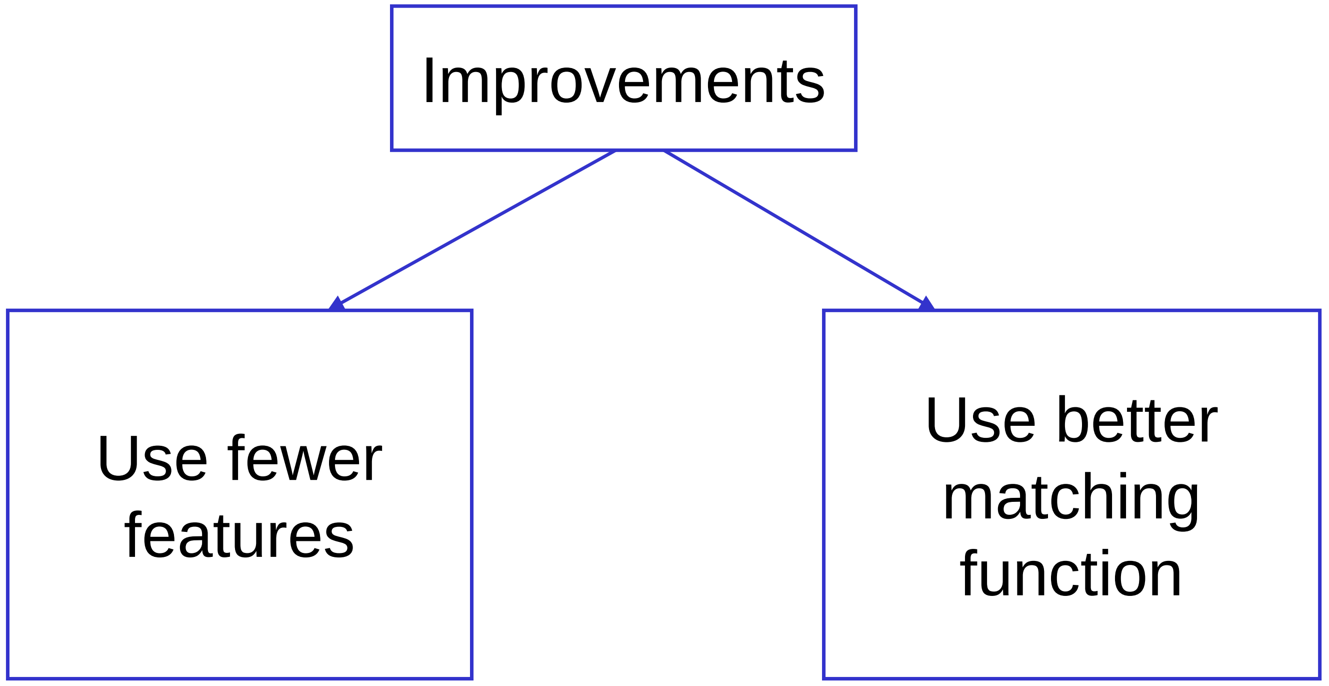
Algorithm:

1. Find t_i, so that $t_i = T$.
2. Assign image to the same class as t_i.

Template Matching(4)

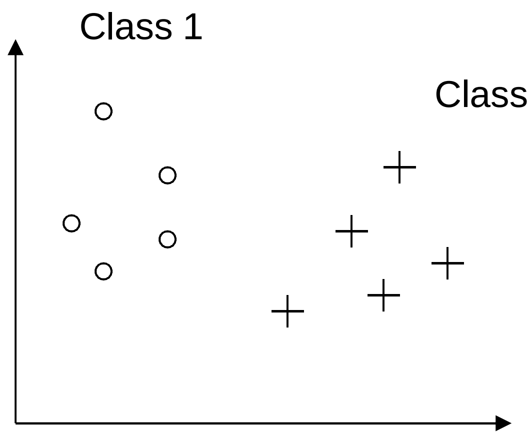
Number of templates to store: 2^{144}

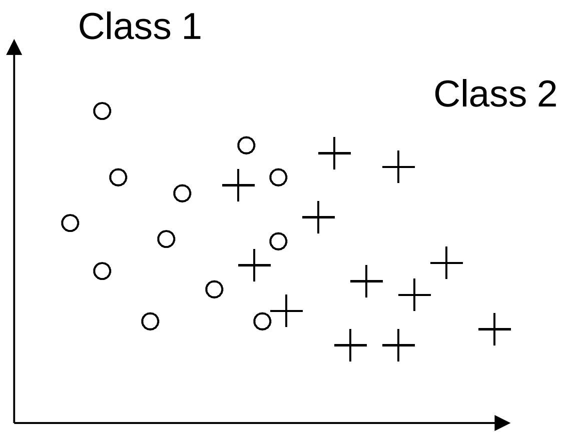
If fewer templates are stored, some images might not be recognized.



Features

- Features are numerically expressed properties of the signal.
- The set of features used for pattern recognition is called feature vector. The number of used features is the dimensionality of the feature vector.
- n-dimensional feature vectors can be represented as points in n-dimensional feature space.





Guidelines for Features

- Use fewer features if possible

Effects:

  – Reducing number of required training samples.
  – Improving quality of recognizing function.

- Use features that differentiate classes well

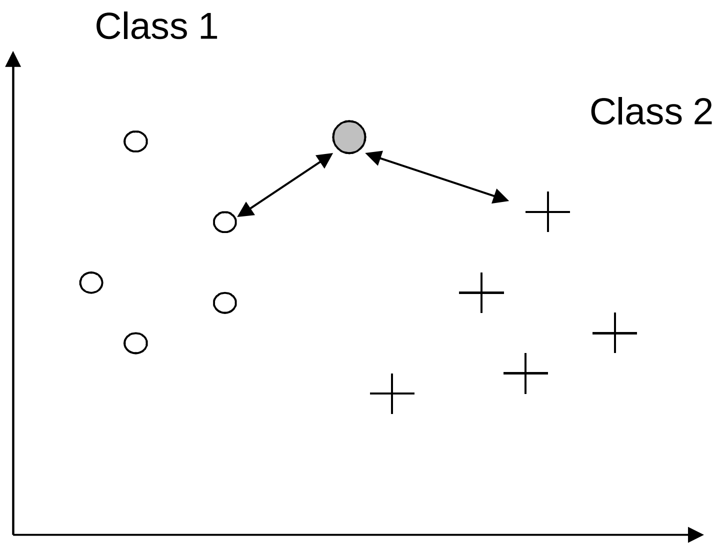
Example for character recognition:

  – Good features: elongation of the image, presence of large loops or strokes.
  – Bad features: number of black pixels, number of connected components.

Distance between feature vectors

- Instead of finding template exactly matching input template look at how close feature vectors are.
- Nearest neighbor classification algorithm:

1. Find template closest to the input pattern.
2. Classify pattern to the same class as closest template.



Center for Unified Biometrics and Sensors
University at Buffalo The State University of New York

Examples of distances in feature space.

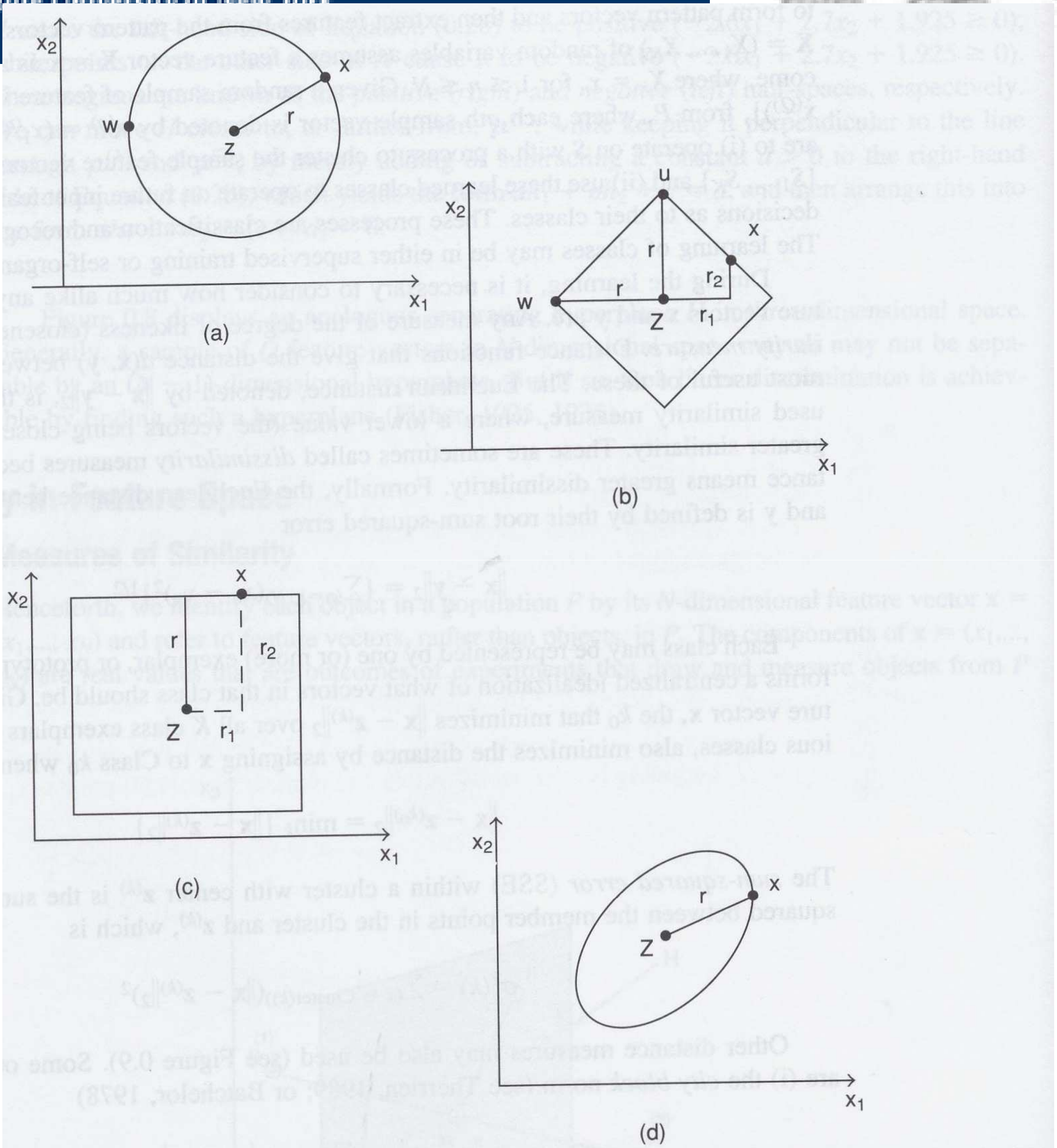
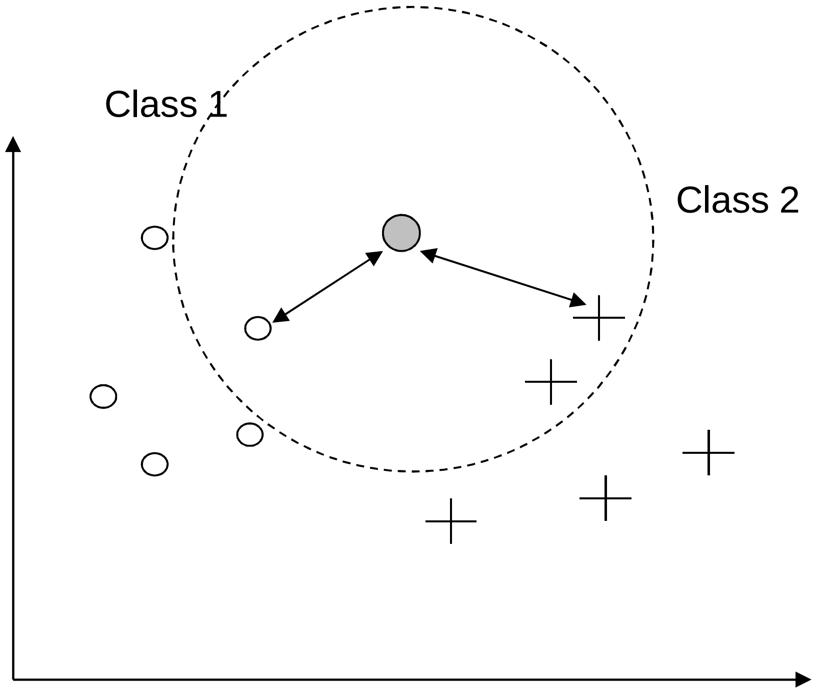


Figure 0.9. Norms as similarity measures and their r-hyperballs with (a) Euclidean Norm, (b) City Block Norm, (c) Supremum Norm, and (d) Mahalanobis Distance.

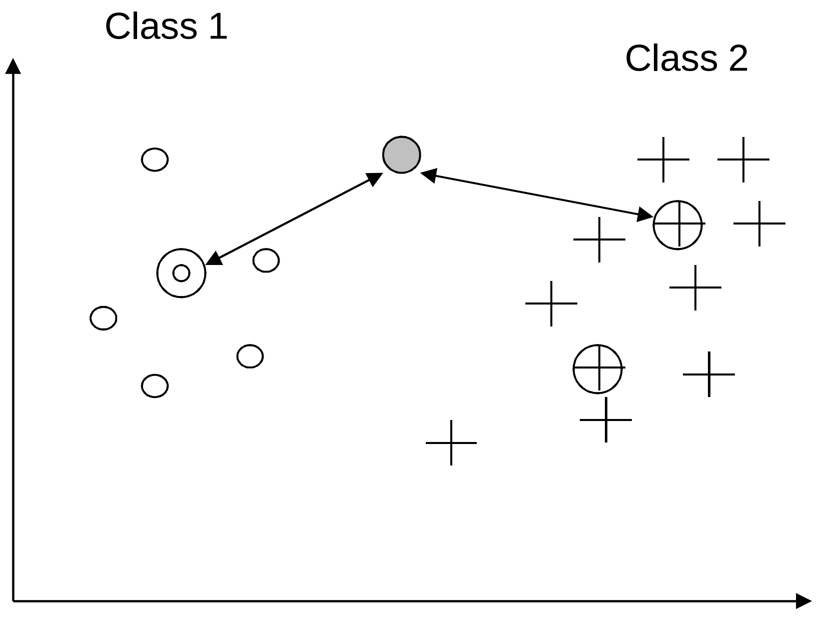
K-nearest neighbor classifier

Modification of nearest neighbor classifier: use k nearest neighbors instead of 1 to classify pattern.



Clustering

Reduce the number of stored templates – keep only cluster centers.

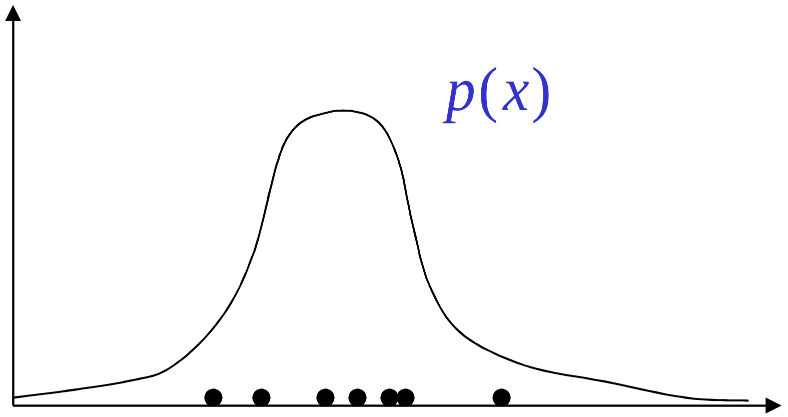


Clustering algorithms reveal the structure of classes in feature space and are used in unsupervised training.

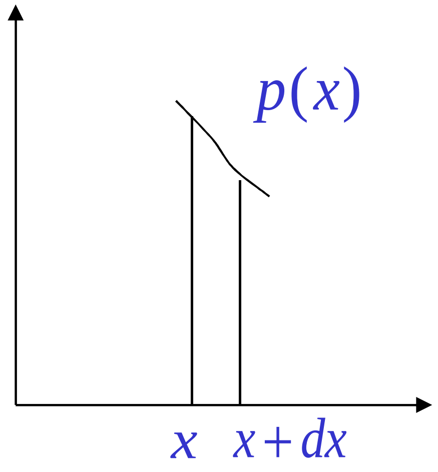
Statistical pattern recognition

•Treat patterns (feature vectors) as observations of random variable (vector).

• Random variable is defined by the probability density function.



Probability density function of random variable and few observations.

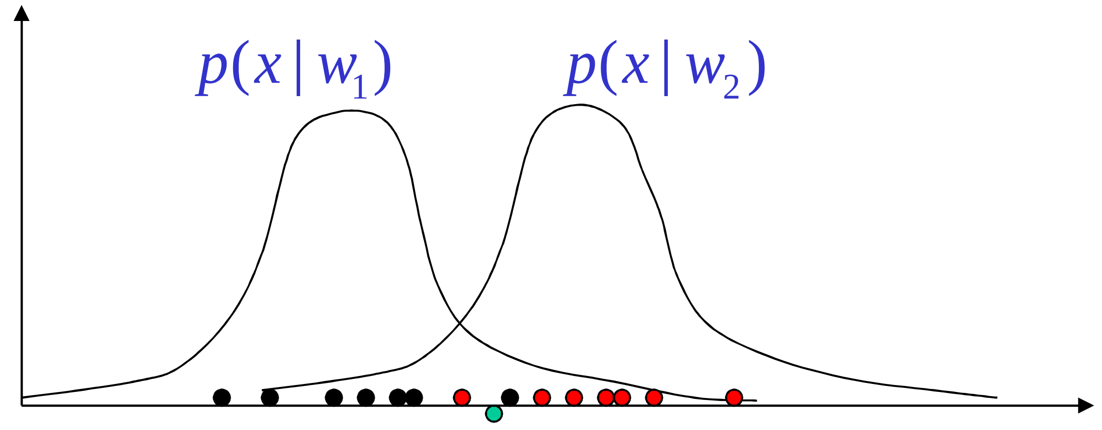


Probability of random variable to fall in the interval $[x, x+dx]$:

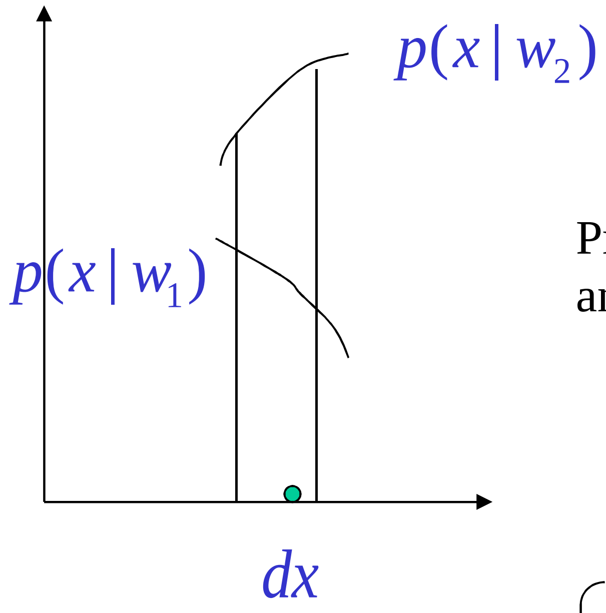
$$\int_{x}^{x+dx} p(x)dx \approx p(x) \times dx$$

Bayes classification rule(1)

• Suppose we have 2 classes and we know probability density functions of their feature vectors. How some new pattern should be classified?



Bayes classification rule(2)



Suppose, sample x can belong to only two classes: w_1 and w_2 .

Probability of random variable to fall in the interval dx and be from class w_1:

$$P(w_1)\times P((x\in dx) \text{ if } (x \text{ is from } w_1))$$
$$= P(w_1)p(x|w_1)\times dx$$

Probability of random variable to fall in the interval dx :

$$\begin{cases} P(w_1)\times P((x\in dx) \text{ if } (x \text{ is from } w_1)) \\ +P(w_2)\times P((x\in dx) \text{ if } (x \text{ is from } w_2)) \\ = P(w_1)p(x|w_1)\times dx + P(w_2)p_2(x|w_2)\times dx \end{cases}$$

Probability of random variable to be from class w_1 when it falls in the interval dx:

$$P((x \text{ is from } w_1) \text{ if } (x\in dx)) = P(w_1|x) = \frac{P(w_1)p(x|w_1)\times dx}{P(w_1)p(x|w_1)\times dx + P(w_2)p_2(x|w_2)\times dx}$$

Bayes classification rule(2)

- Bayes formula:

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

$$p(x) = \sum_{i=1}^{2} p(x \mid w_i) P(w_i)$$

$P(w_i)$ - prior class probability

$P(w_i \mid x)$ - posterior class probability

$p(x \mid w_i)$ - likelihood of sample x (or density of class w_i)

Above formula is a consequent of following probability theory equations:

$$P(A, B) = P(A \mid B) P(B) = P(B \mid A) P(A)$$

$$P(C) = P(C, A) + P(C, B), \text{ if } A \cap B = 0, A \cup B = 1$$

Bayes classification rule(3)

- Bayes classification rule: classify x to the class w_i which has biggest posterior probability $P(w_i \mid x)$

$$P(w_1 \mid x) > P(w_2 \mid x) \;?\quad w_1 \quad : \quad w_2$$

Using Bayes formula, we can rewrite classification rule:

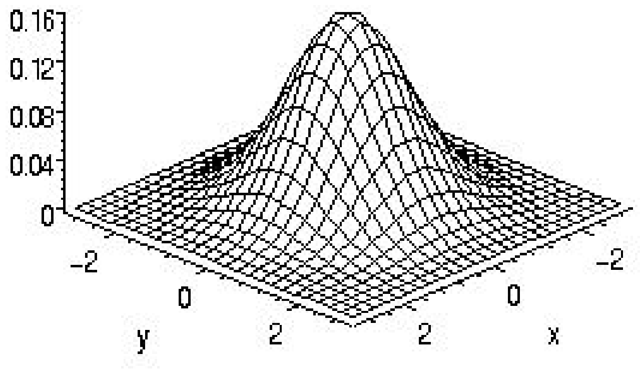
$$p(x \mid w_1)P(w_1) > p(x \mid w_2)P(w_2) \;?\quad w_1 \quad : \quad w_2$$

Estimating probability density function.

• In applications, probability density function of class features is unknown.

• Solution: model unknown probability density function $p(x|w_i)$ of class w_i by some parametric function $p_i(x;\theta)$ and determine parameters based on training samples.

Example: model pdf as a Gaussian function with unitary covariance matrix and unknown mean



$$p(x;\mu) = \frac{1}{(2\pi)^{l/2}} e^{-\frac{1}{2}(x-\mu)^2}$$

Maximum likelihood parameter estimation

- What is the criteria for estimating parameters θ?
- Maximum likelihood parameter estimation:

Parameter θ should maximize the likelihood of observed training samples

$$p(X;\theta) = p(x_1, x_2, \ldots, x_N \mid \theta) = \prod_{k=1}^{N} p(x_k;\theta)$$

- Equivalently, parameter θ should maximize loglikelihood function:

$$\ln(p(X;\theta)) = \ln(p(x_1, x_2, \ldots, x_N \mid \theta)) = \sum_{k=1}^{N} \ln(p(x_k;\theta))$$

ML-estimate for Gaussian pdf

$$\ln(p(X;\mu)) = \sum_{k=1}^{N} \ln(p(x_k;\mu)) = \sum_{k=1}^{N} \ln\left(\frac{1}{(2\pi)^{l/2}} e^{-\frac{1}{2}(x_k-\mu)^2}\right)$$

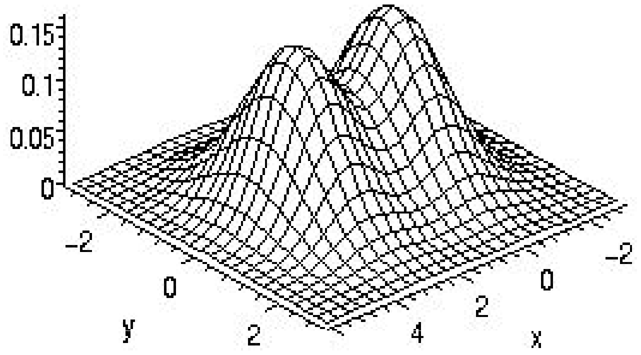
$$= -N\ln((2\pi)^{l/2}) - \frac{1}{2}\sum_{k=1}^{N}(x_k-\mu)^2$$

To find an extremum of function $\ln(p(X;\theta))$ (with respect to θ) we equal its gradient to 0:

$$\nabla_\theta \ln(p(X;\mu)) = \begin{bmatrix} \frac{\partial \ln(p(X;\mu))}{\partial \mu_1} \\ \vdots \\ \frac{\partial \ln(p(X;\mu))}{\partial \mu_l} \end{bmatrix} = \sum_{k=1}^{N}(x_k-\mu) = 0$$

Thus, estimate for parameter μ is: $\mu = \frac{1}{N}\sum_{k=1}^{N} x_k$

Mixture of Gaussian functions



$$p(x;\mu) = \sum_{i=1}^{N} \frac{P_i}{(2\pi\sigma_i^2)^{l/2}} e^{-\frac{1}{2}\frac{(x-\mu_i)^2}{\sigma_i^2}}$$

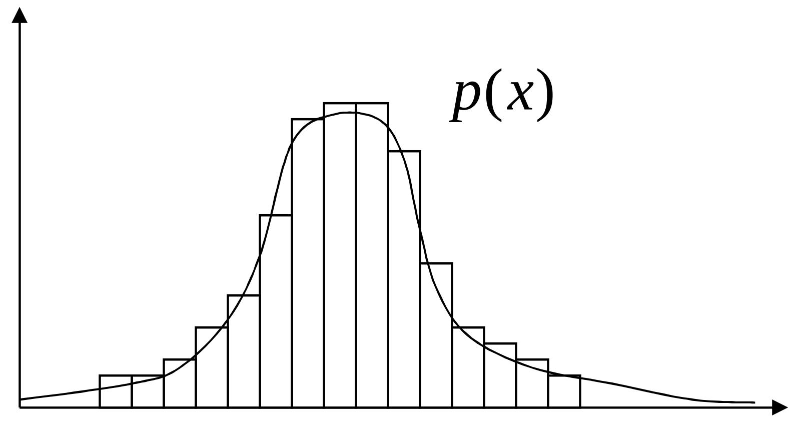
• No direct computation of optimal values of parameters P_i, μ_i, σ_i is possible.

• Generic methods for finding extreme points of non-linear functions can be used: gradient descent, Newton's algorithm, Lagrange multipliers.

• Usually used: expectation-maximization (EM) algorithm.

Nonparametric pdf estimation

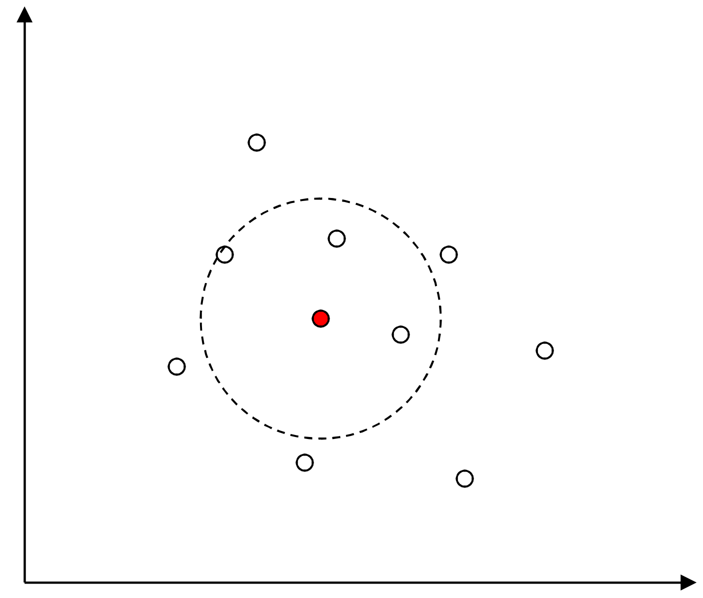
Histogram method:



Split feature space into bins of width h.
Approximate p(x) by:

$$\hat{p}(x) = \frac{1}{h} \frac{\text{Number of training samples inside bin}}{\text{Total number of training samples}}$$

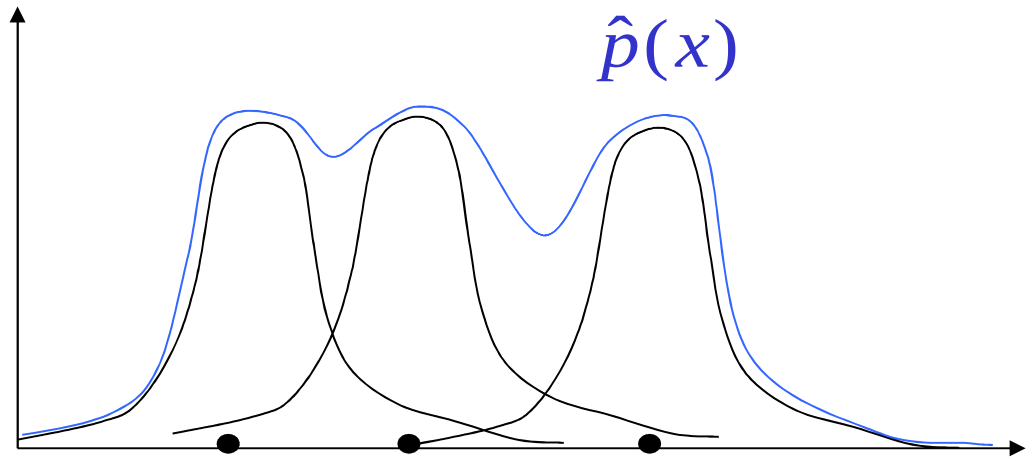
Nearest neighbor pdf estimation



Find k nearest neighbors. Let V be the volume of the sphere containing these k training samples. Then approximate pdf by:

$$\hat{p}(x) = \frac{1}{V}\frac{k}{N}$$

Parzen windows.



Each training point contributes one Parzen kernel function to pdf construction:

$$\hat{p}(x) = \frac{1}{N}\sum_{i=1}^{N}\left(\frac{1}{h}\varphi\left(\frac{x_i - x}{h}\right)\right)$$

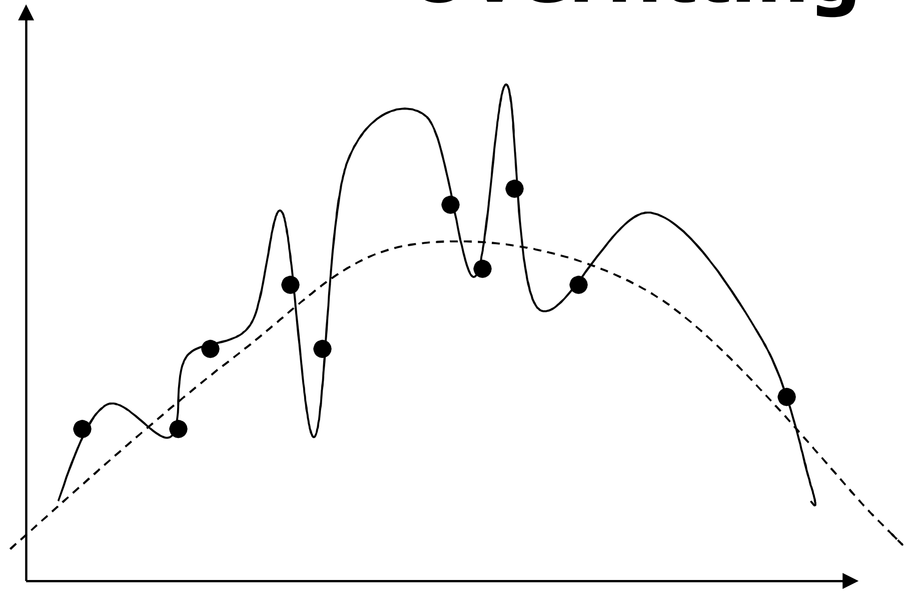
- *Important to choose proper h.*

Parzen windows for cluster centers

- Take cluster centers as centers for Parzen kernel functions.
- Make contribution of the cluster proportional to the number of training samples cluster has.

$$\hat{p}(x) = \frac{1}{N} \sum_{i=1}^{N} \left(\frac{N_i}{h} \varphi \left(\frac{c_i - x}{h} \right) \right)$$

Overfitting

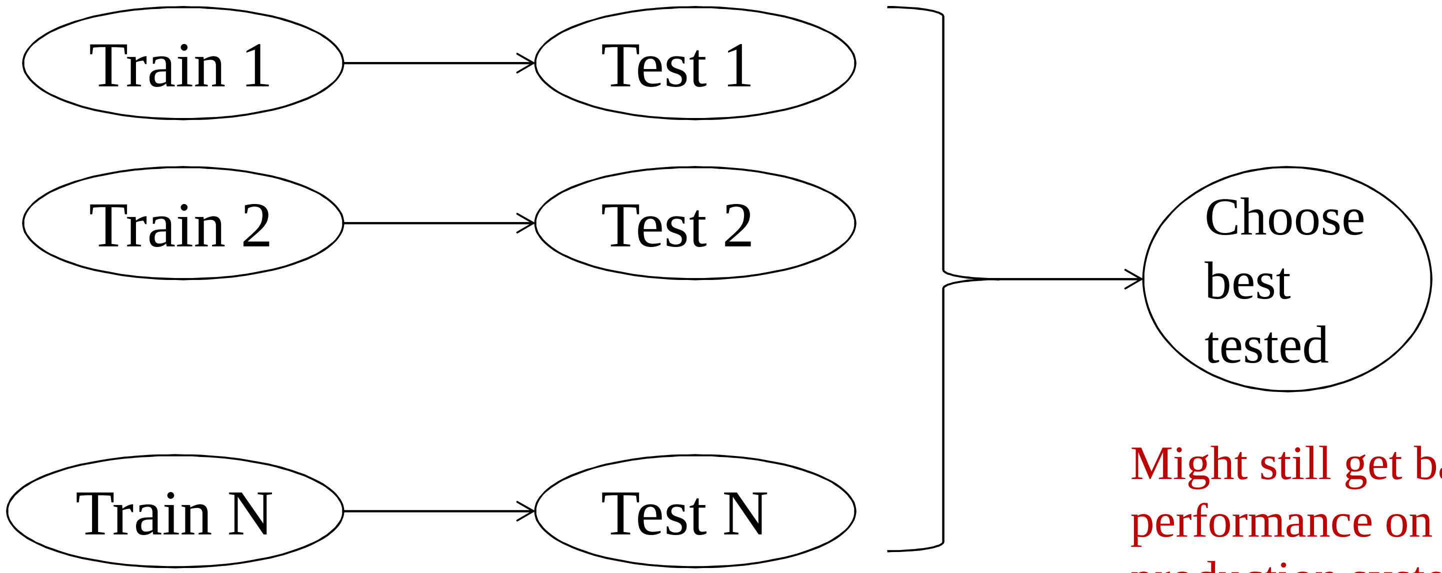


When number of trainable parameters is comparable to the number of training samples, overfitting problem might appear. Approximation might work perfectly on training data, but not well on testing data.

Avoiding overfitting

- No good error estimates from training methods
- Use simpler classifier (with fewer trainable parameters)
- Use separate validation/test data to test the training

Possible training/testing procedure resulting in overfitting effect:



Might still get bad performance on production system