Aprenentatge i Reconeixement de Formes
Pattern Recognition and Machine Learning

Francesc J. Ferri

Dept. d'Informàtica. Universitat de València

November 2009

Syllabus

1. Statistical Approach to PR
   - Bayes rule
   - Discriminant functions
2. Non parametric methods
   - distance-based techniques
   - Nearest neighbors
3. Connexionist approaches
   - Perceptrons
   - Autoorganizative maps
   - Prototype-based networks
4. Unsupervised Learning
   - Clustering
   - Vectorial Quantization
   - Semi-supervised approaches
5. Complexity Reduction
   - Feature extraction
   - Feature Selection
   - Prototype selection/generation

Bibliography

- Duda, Hart, Stork “Pattern Classification”, Wiley 2001.
- Webb “Statistical Pattern Recognition”, 2002.
- Hastie, Tibshirani, Friedman “The Elements of Statistical Learning”, Springer 2001.
- Devroye, Gyorfi, Lugosi “A Probabilistic Theory of Pattern Recognition”, Springer 1996.
- Vapnik “The Nature of Statistical Learning”, Springer 1995.
- Theodoridis, Koutroumbas “Pattern Recognition” Academic Press, 1999. Statistical pattern recognition: a review
- Jain, A.K.; Duin, R.P.W.; Jianchang Mao; Statistical pattern recognition: a review IEEE Trans. on Pattern Analysis and Machine Intelligence, Volume 22, Issue 1, Jan. 2000 Page(s):4 - 37 `http://dx.doi.org/10.1109/34.824819`

Introduction

- Machine Perception
- Pattern Analysis
- Classification/Recognition
- Computacional/Algorithmic/Machine Learning
- Data Mining

Recognition/Perception Systems

signals ⟶ recognition ⟶ *interpretation*

A **pattern** is any kind of stimulus with a certain type of "regularity" which enables effective discrimination with regard to other similar stimuli.

The easiest case of **interpretation** is **classification** (also **categorization**).

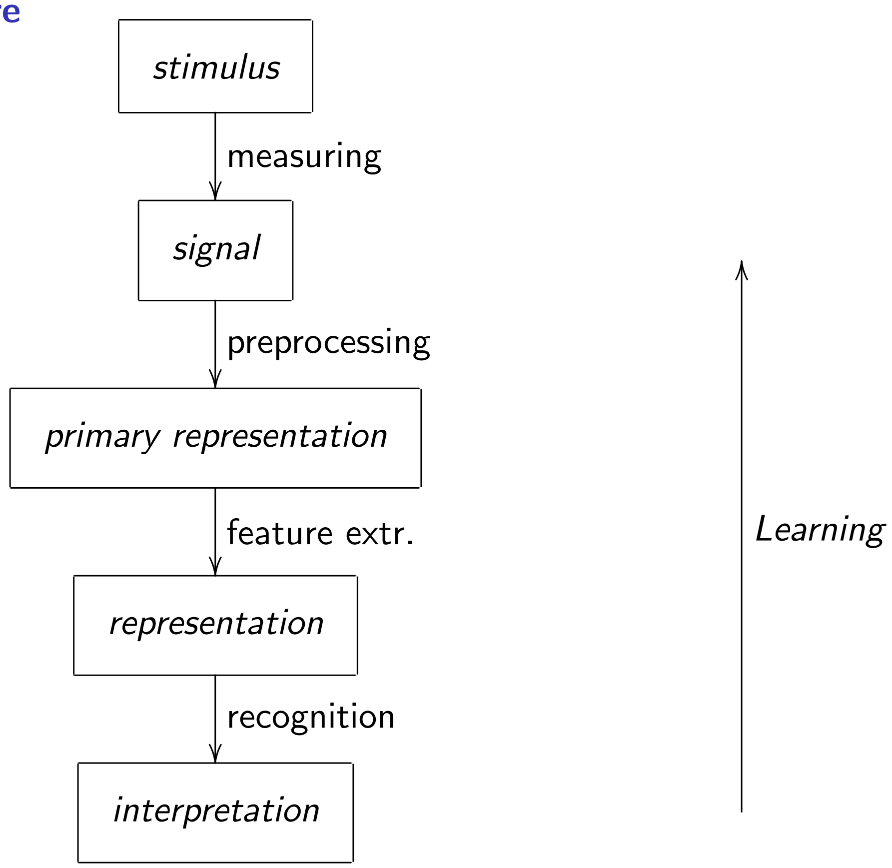
A **class** is an entity representing a certain kind of patterns. The elements of a class share certain "properties" while still present differences.

The goal of a pattern recognition system consists of **modelling** or **identifying** the different "classes" of objects in a particular application of a certain domain.

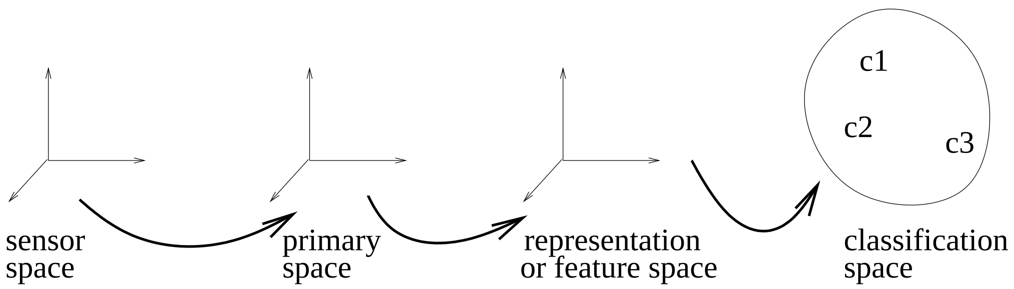
EXAMPLE: *softdrink automatic machine*

- stimulus: introducing bill notes (or coins).
- preprocessing: using mechanical/optic/magnetic sensors to achieve different signals.
- representation: a set of measures about the stimulus.
- result: a money amount (or error).

Architecture



Representation



In general, representation spaces are **arbitrary spaces**. But in most cases, we will deal with **Hilbert spaces** (i.e. **vector spaces** with inner product) from which **Euclidean spaces** are a particular case. In some cases, **metric** or **pseudometric spaces** can be considered.

Structurally rich patterns may require arbitrary complex spaces like graph spaces, tree spaces or string spaces (structural/syntactic pattern recognition)

Preprocessing

An appropriate description of the phenomenon under study is obtained from the sensors. Examples of these descriptions are:

OCR:	a binary matrix (binary image representing a printed symbol).
Speech recognition:	an (audio) signal.
Spam filtering:	a text file.
Face recognition:	an integer matrix (grey level image).
Textile Quality Control:	three matrices (a color image).
Softdrink machine:	a vector of different values.

Preprocessing

Usually, some preprocessing is done in the primary representation in order to improve further processing in the system.

Noise reduction, relevant information emphasis, redundance removing are particular examples.

Preprocessing is **much** more important when object under study are images or *n*-dimensional signals in general.

Primary representation and features

Preprocessing does not usually implies a change of domain (e.g. clean images, emphasized audio signals, text files after trimming, etc. are the results).

On the other hand, in order to analyze and recognize objects, a convenient representation (features) needs to be obtained. Usually some a priori knowledge has to be known in order to obtain such representations.

Example:

a) speech recognition of "yes" or "no" answers on the phone, and b) tone detection for melody recognition are two applications with the same sensor information, similar primary representations but radically different features.

Problem definition

Given an (abstract) representation space (feature space), find a **mapping** from objects in this onto the **classification space** (a finite number of class labels) or the **regression space** another vectorial representation (usually).

In the case of classification, the mapping $\mathcal{F}$ is referred to as the classifier

$$\mathcal{F} : X \longrightarrow W = \{w_1, \ldots, w_c\}$$

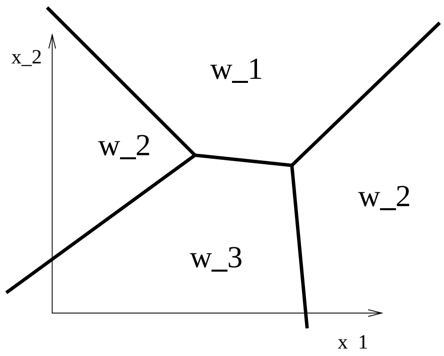
The information in order to obtain $\mathcal{F}$ is given as examples, i.e. pairs (x, w) where $x \in X$ and $w \in W$.

It must be fulfilled that $\mathcal{F}(x) = w$ for all examples (x, w).

Geometric representation

In particular, if there is a vector space for classification, the mapping $\mathcal{F}$ can be seen as a **partition**

Under certain circumstances, blocks in this partition are compact regions separated by **decision boundaries**.



Examples in Matlab and PRtools

Lets consider a very well known illustrative example used in the literature

Iris plants

```
>> A=iris
Iris plants, 150 by 4 dataset with 3 classes:  [50 50 50]
>> disp(getfeatlab(A))
sepal length
sepal width
petal length
petal width
>> disp(getlablist(A))
Iris Setosa
Iris Versicolour
Iris Virginica
```

These are objects in a 4D vector space. The meaning of each feature are measures over petals and sepals of 3 species of flowers. $W = \{w_1, w_2, w_3\}$

Examples in Matlab and PRtools

We can visualize 2D and 3D projections of the objects and define mappings for classification or regression.

```
>> scatterd(A(:,[1 3])) ; scatterd(A(:,[1 3 4]),3)
>> wm=fisherm(A) % this is a 4D to 2D regression mapping
Fisher mapping, 4 to 2 trained mapping --> affine
>> disp(A(11,:)*wm) % mapping applied to 11th object
Iris plants, 1 by 2 dataset with 3 classes:  [1 0 0]
-8.3974 0.6474
>> wc=nmc(A)*labeld % this is a classification mapping
4 to 3 trained mapping --> sequential
>> disp(A(11,:)*wc) % mapping applied to 11th object
Iris Setosa
>> B=A*wm; % Projected data from 4D to 2D
>> wcc=fisherc(B)*labeld % another classification example in 2D
>> scatterd(B),plotc(wcc) % shows objets and partition
```


Performance assessment

The solution of a particular problem is the result of a learning process: a particular mapping between X and W.

Usually the learning is done from **examples** and the goodness of the solution found has to be "measured" also using examples.

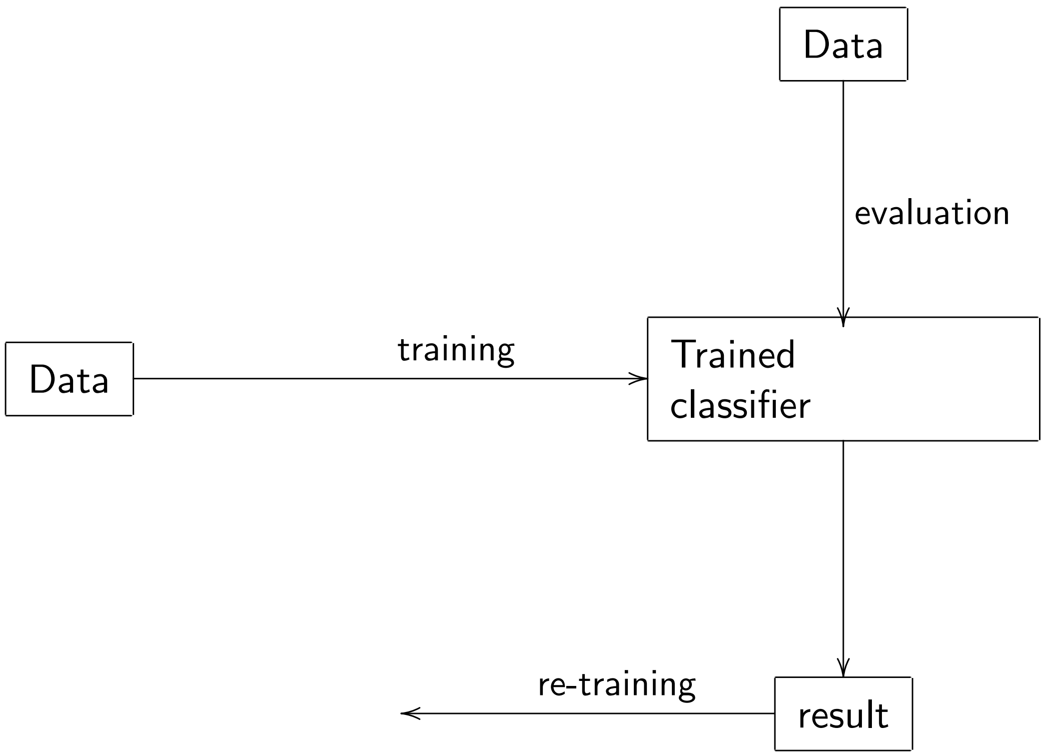
Resubstitution estimate

We can for example use all available samples to obtain (learn) the mapping and estimate the performance on the **same** samples. This is also referred to as "apparent" error rate.

It is usual to distinguish between examples used for learning (Training set or training samples) and examples to evaluate the mapping (Test set or testing samples).

Resubstitution estimation corresponds to the particular case in which all samples are both training and testing samples.

Error Rate Estimation



Measuring Error Rate

Resubstitution usually gives rise to very optimistic results (in some cases even unacceptable optimistic).

Probability of error

In the opposite side we can measure performance as the probability of the mapping giving an error. This is a function on X but can be appropriately averaged. This is also referred to as "true" error.

All other possibilities give feasible estimates of the true error

Error estimation methods

Resubstitution: All samples are training and testing.

Holdout: Training and Test sets are different and (statistically) independent.

Cross Validation: Multiple estimate in which samples are used (in turn) for training and testing. Final estimation is computed by averaging.

Bootstrap: The available samples are randomly sampled (with substitution) to extract (independent) subsets that are used for training and testing. Final estimation is done by a (corrected) average of several trials.

Cross-validation. Particular cases

Leaving-one-out

All examples but one are used to learn a mapping that is applied to the only example. The process is repeated for all examples and the number of errors is counted.

Rotation or m-fold (stratified) cross validation

Available examples are partitioned into m blocks. All blocks but one are used to learn a mapping that is applied to the left block. The process is repeated for the m blocks and the number of errors is counted. Stratified refers to the fact that proportions of different classes are preserved in each block.

Error estimation

The quality of error estimation depends on several different facts:

- the effective number of examples used to learn or train the mapping.
- the (statistical) independence of the samples used for train and test.
- the number and variability of the examples used for testing.

Under some assumptions, the quality of different error estimators can be formally analyzed and characterized in terms of **bias** (deviation between the target and the estimate) and **variance** (the variability of the estimate).

Examples

```
A=nist16                                          % a 10-digits image dataset
imshow(uint8(reshape(+A(700,:),16,16)))                     % have a look
[Tr,Ts]=gendat(A,.6)                                   % split data in 60-40%
wc=nmc(Tr)                                  % a classifier (obtained from Tr)
testc(Tr*wc)                                   % resubstitution error (on Tr)
testc(Ts*wc)                              % holdout error (for this partition)
(size(Ts,1)*testc(Ts*nmc(Tr)) + size(Tr,1)*testc(Tr*nmc(Ts)))/size(A,1)
                                                      % cross validation error
sum(labeld(Tr*wc)==getlabels(Tr))/size(Tr,1)                  % same as testc
```

Conditional error evaluation

In order to particularize errors for different classes one considers the probability of a classifier assigning class j to an i-class example, e_{ij}. This forms the so-called **confusion matrix**:

$$\begin{pmatrix} e_{11} \cdots e_{1c} \\ \vdots \quad\quad \vdots \\ e_{c1} \cdots e_{cc} \end{pmatrix}$$

Sometimes this matrix is formed with error counts instead of probabilities.

Conditional error evaluation. Two classes

Change of notation:

$c_A + e_A = n_A$ examples in class A (possitive). The same for class B.

$n_A + n_B = n$ total number of examples

$$\begin{pmatrix} c_A & e_A \\ e_B & c_B \end{pmatrix} \qquad\qquad \begin{pmatrix} TP & FN \\ FP & TN \end{pmatrix}$$

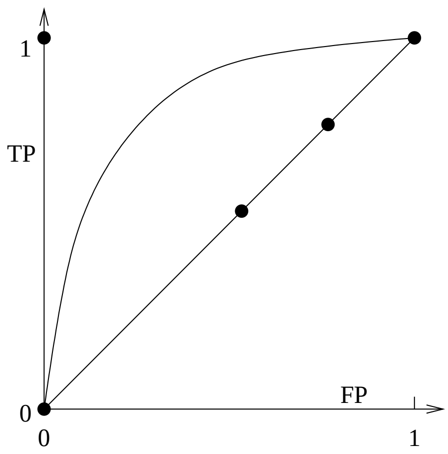
- $\frac{c_A}{n_A}$ *hit, true positives (TP).*
- $\frac{e_B}{n_B}$ *false alarm, false positives (FP).*
- $\frac{e_A}{n_A}$ *miss, false negatives.*
- $\frac{c_B}{n_B}$ *correct reject, true negatives.*

Conditional error evaluation. Two classes

As $TP + FN = 1$ and $FP + TN = 1$ the complete (conditional) behavior of a 2-class classifier can be characterized by 2 parameters: e.g. (FP, TP).

This 2D parameter space is referred to as ROC space (from Receiver Operating Curves).

We can identify (the behavior of) some particular classifiers in this space.



Examples

```
A=breast                                  % a 2-class dataset
wc=nmc(A)                                 % a classifier
testc(A*wc)                               % resubstitution error
confmat(A*wc)                             % confusion matrix
e = roc(a)                                % a ROC curve
plotr(e)
e.error = 1-e.error;
e.xlabel = 'False Positive';
e.ylabel = 'True positive';
plotr(e)                                  % a more standard one
```