Introduction to Pattern Recognition

Luís Gustavo Martins - lmartins@porto.ucp.pt
EA-UCP, Porto, Portugal
http://artes.ucp.pt ; http://citar.ucp.pt

Human Perception

- Humans have developed highly sophisticated skills for sensing their environment and taking actions according to what they observe, e.g.,
  - Recognizing a face.
  - Understanding spoken words.
  - Reading handwriting.
  - Distinguishing fresh food from its smell.
  - ...

Human and Machine Perception

- We are often influenced by the knowledge of how patterns are modeled and recognized in nature when we develop pattern recognition algorithms.

- Research on machine perception also helps us gain deeper understanding and appreciation for pattern recognition systems in nature.

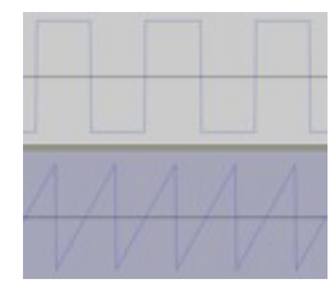
- Yet, we also apply many techniques that are purely numerical and do not have any correspondence in natural systems.

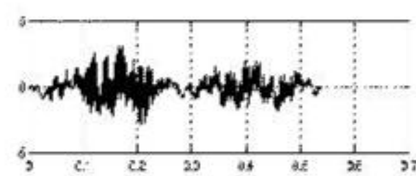
Pattern Recognition (PR)

- Pattern Recognition is the study of how machines can:
  - observe the environment,
  - learn to distinguish patterns of interest,
  - make sound and reasonable decisions about the categories of the patterns.

Pattern Recognition (PR)

- **What is a Pattern?**
  - is an abstraction, represented by a set of measurements describing a "physical" object
- Many types of patterns exist:
  - visual, temporal, sonic, logical, ...

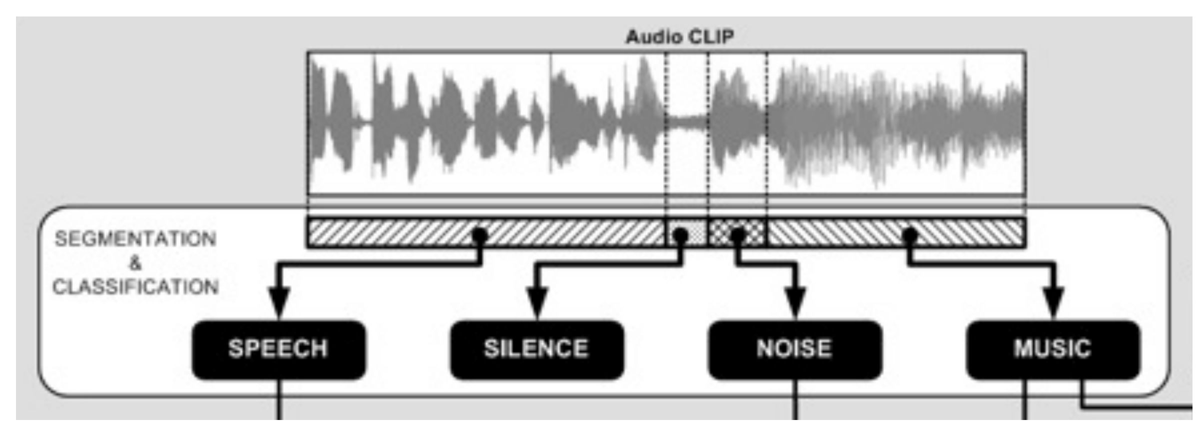





Pattern Recognition (PR)

- **What is a Pattern Class** (or category)?
  - is a set of patterns sharing common attributes
  - a collection of "similar", not necessarily identical, objects
  - During recognition, given objects are assigned to a prescribed class



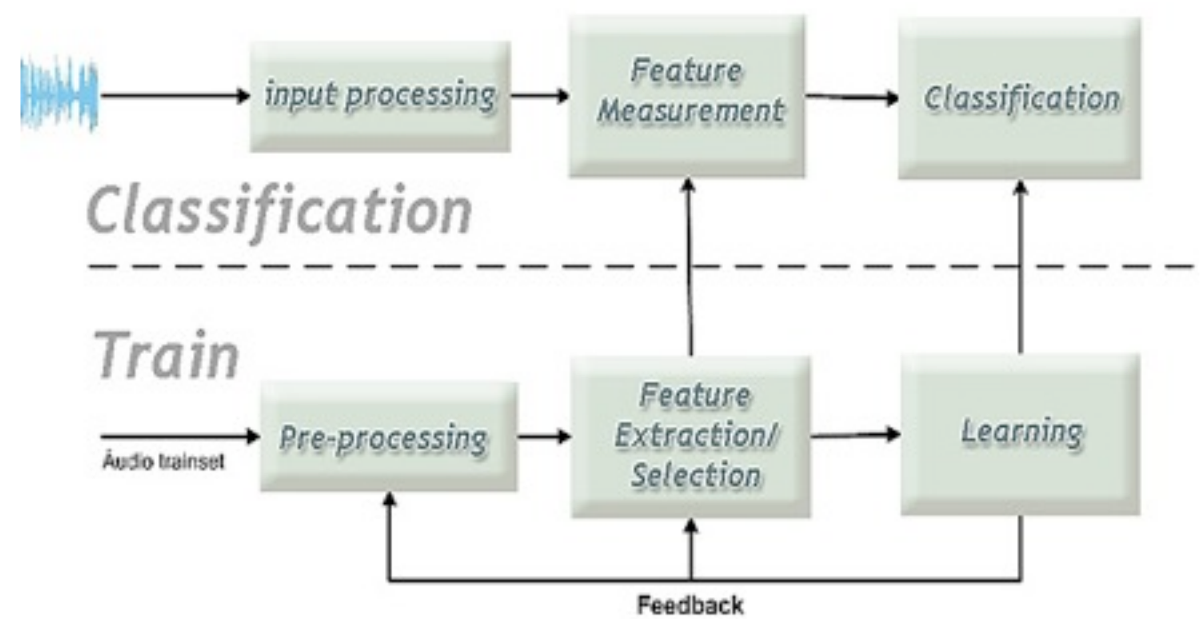
Pattern Recognition (PR)

- No single theory of Pattern Recognition can possibly cope with such a broad range of problems...
- However, there are several standard models, including:
  - **Statistical** or fuzzy pattern recognition (see Fukunaga)
  - Syntactic or structural pattern recognition (see Schalkoff)
  - Knowledge-based pattern recognition (see Stefik)

Luís Gustavo Martins - lmartins@porto.ucp.pt

Pattern Recognition

- Two phase Process
  1. Training/Learning
     - Learning is hard and time consuming
     - System must be exposed to several examples of each class
     - Creates a "model" for each class
     - Once learned, it becomes natural
  2. Detecting/Classifying

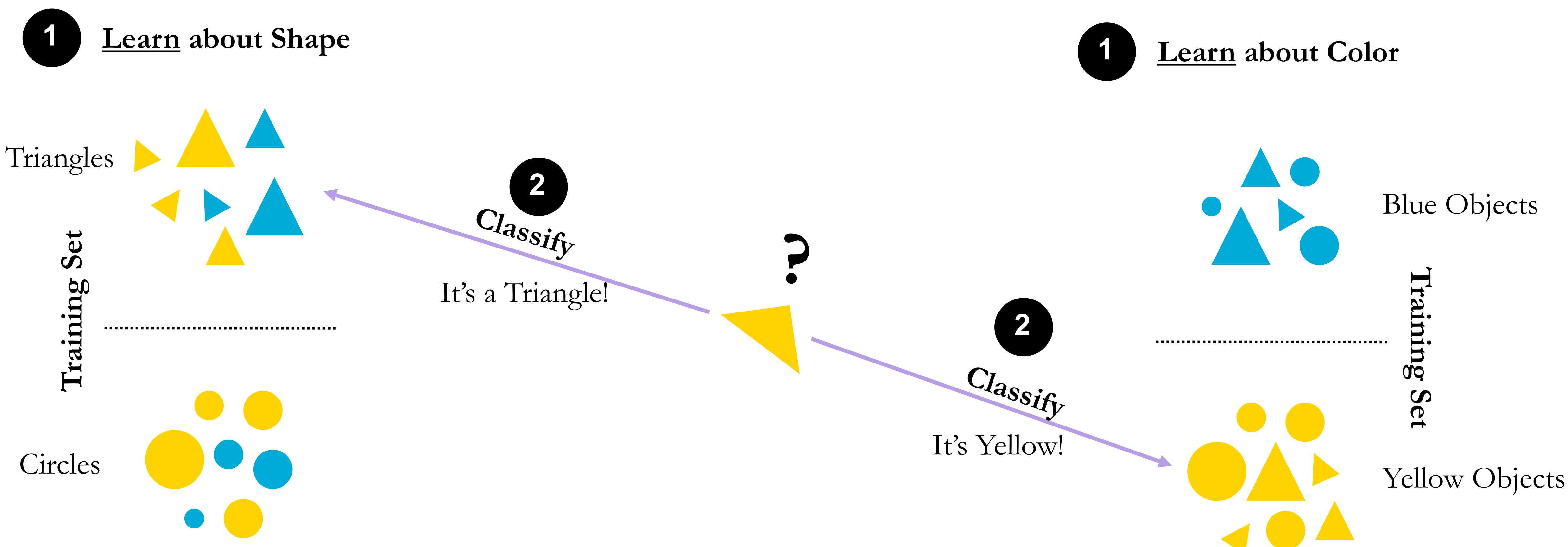


Pattern Recognition

- How can a machine learn the rule from data?
  - **Supervised learning:** a teacher provides a category label or cost for each pattern in the training set.
    - ➡*Classification*
  - **Unsupervised learning:** the system forms clusters or natural groupings of the input patterns (based on some similarity criteria).
    - ➡*Clustering*
  - **Reinforcement learning:** no desired category is given but the teacher provides feedback to the system such as the decision is right or wrong.

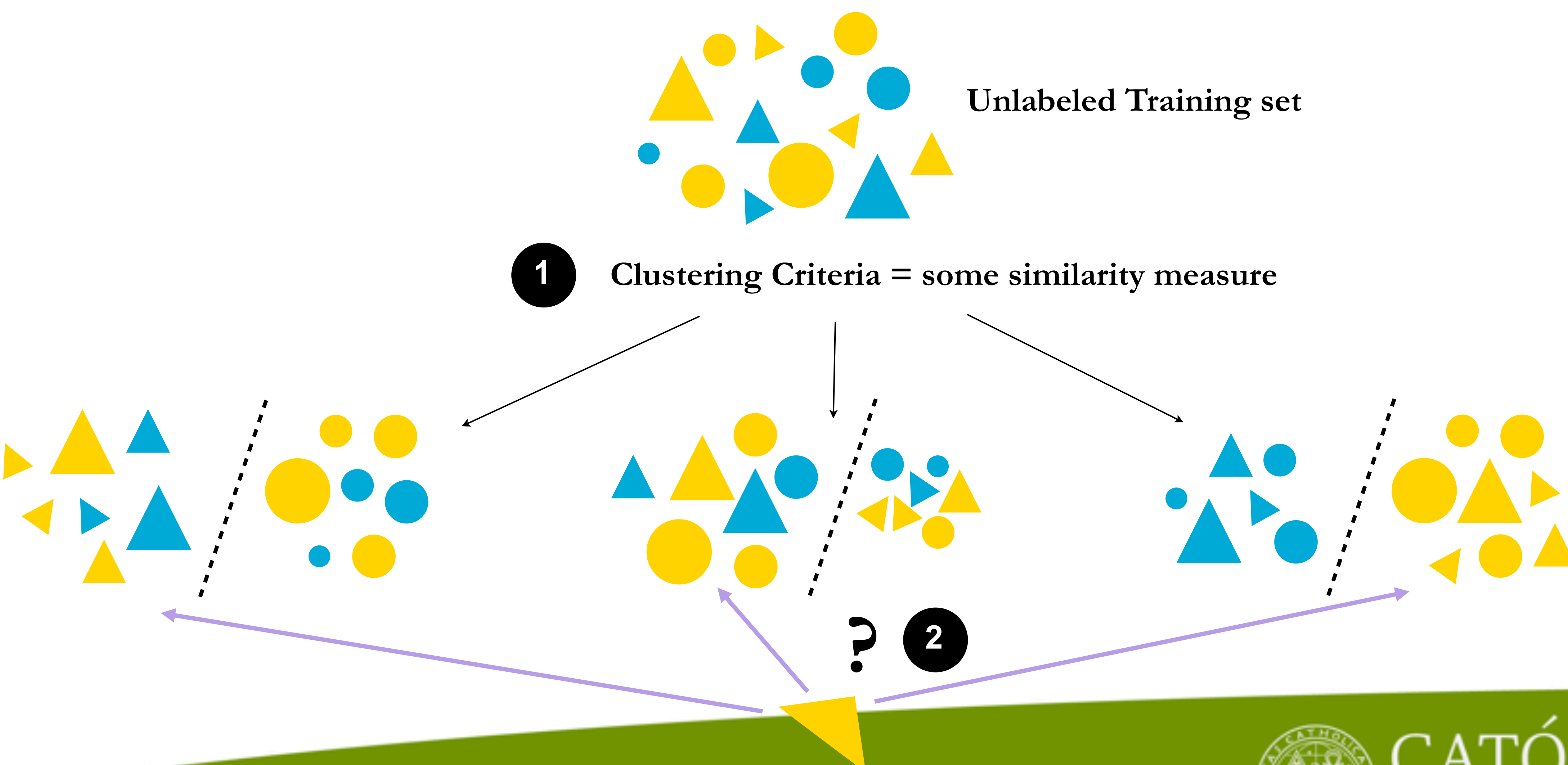
Pattern Recognition

- **Supervised Training/Learning**
  - a "teacher" provides labeled training sets, used to train a classifier



Pattern Recognition

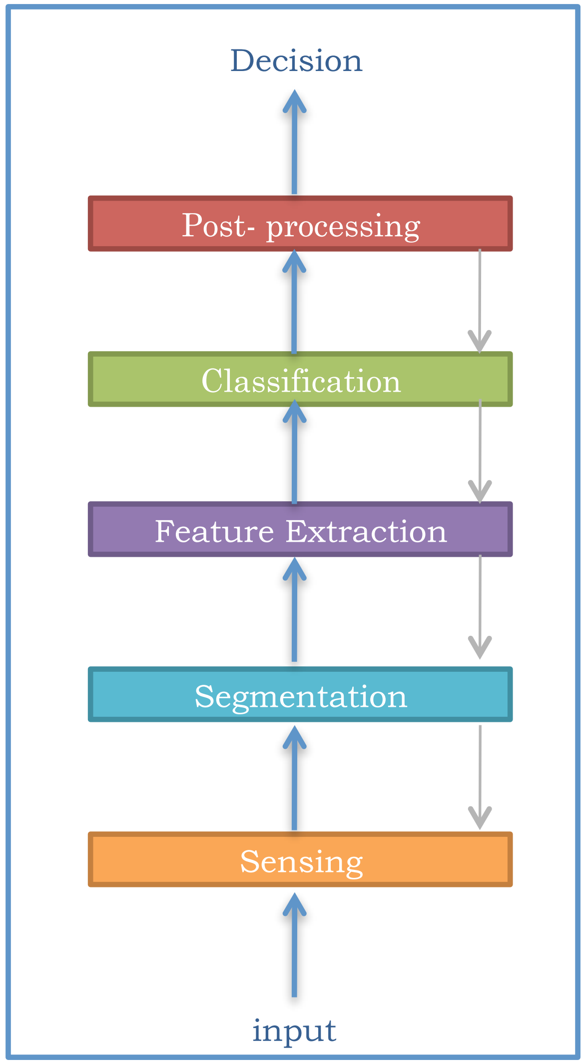
- **Unsupervised Training/Learning**
  - No labeled training sets are provided
  - System applies a specified clustering/grouping criteria to unlabeled dataset
  - Clusters/groups together "most similar" objects (according to given criteria)



Luís Gustavo Martins - lmartins@porto.ucp.pt

Pattern Recognition Process

- **Data acquisition and sensing:**
  - Measurements of physical variables.
  - Important issues: bandwidth, resolution , etc.
- **Pre-processing:**
  - Removal of noise in data.
  - Isolation of patterns of interest from the background.
- **Feature extraction:**
  - Finding a new representation in terms of features.
- **Classification**
  - Using features and learned models to assign a pattern to a category.
- **Post-processing**
  - Evaluation of confidence in decisions.

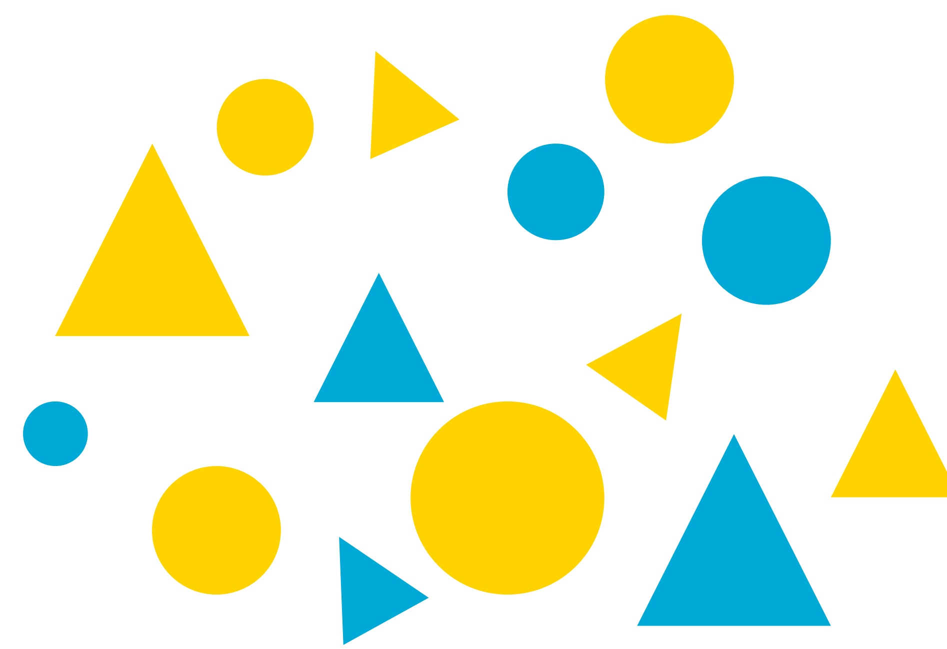


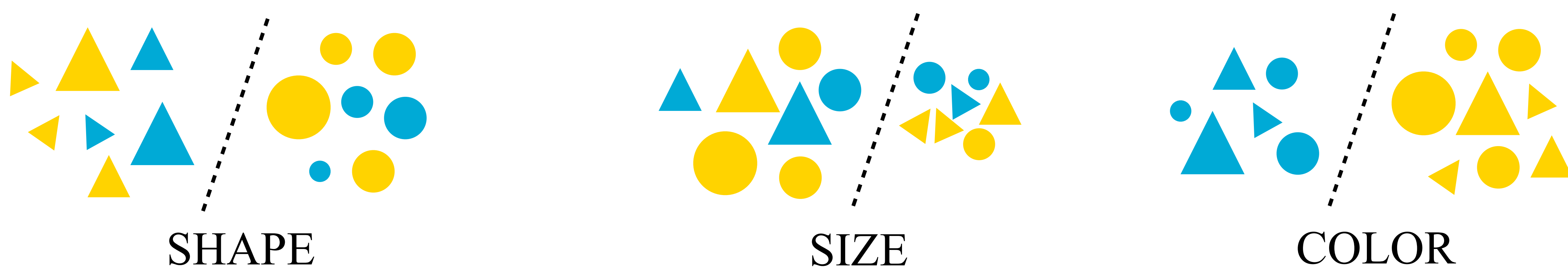
Features

- Features are properties of an object:
  - Ideally representative of a specific type (i.e. class) of objects
  - Compact (memory efficient)
  - Computationally simple (CPU efficient)
  - Perceptual relevant (if trying to implement a human inspired classifier)
- **=> Should pave the way to a good discrimination of different classes of objects!**

Features

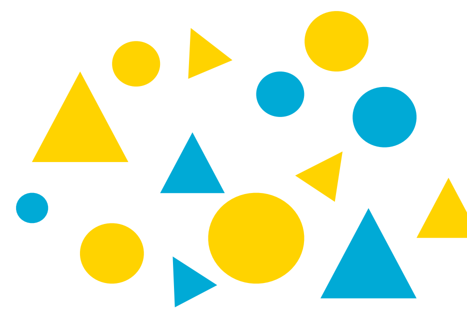
- Take a group of graphical objects
  - Possible features:
    - Shape
    - Color
    - Size
    - ...
  - Allows to group them into different classes:





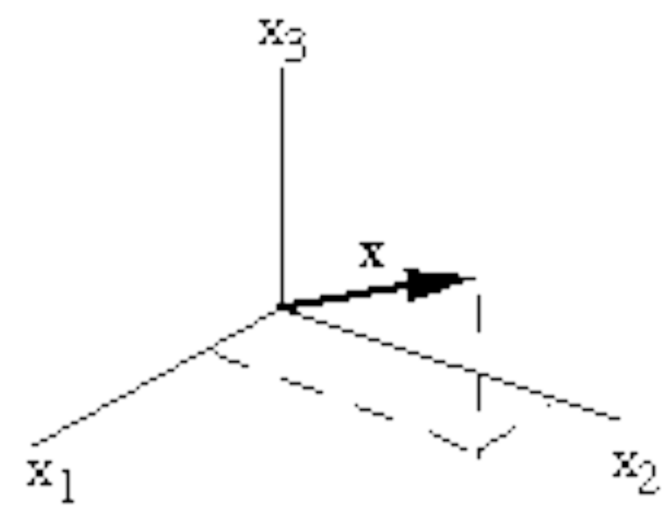
Feature Vectors

- Usually a single object can be represented using several features, e.g.
  - **x1** = shape (e.g. nr of sides)
  - **x2** = size (e.g. some numeric value)
  - **x3** = color (e.g. rgb values)
  - ...
  - **xd** = some other (numeric) feature.



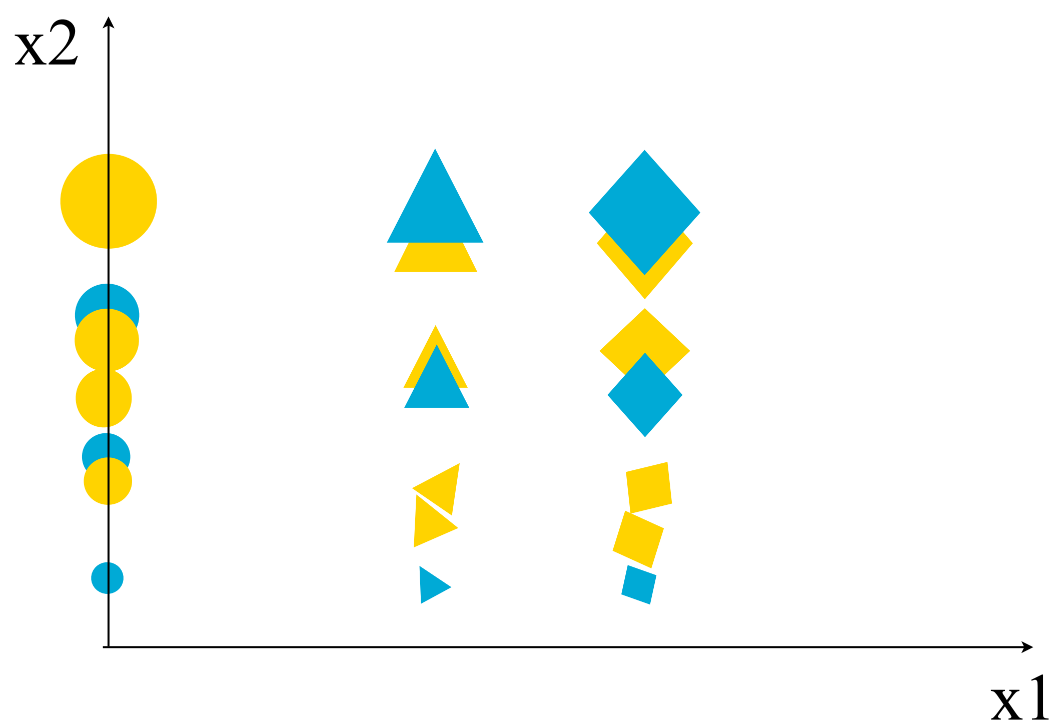
- **X** becomes a feature vector
  - x is a point in a d-dimensional **feature space.**

$$\mathbf{x} = \begin{bmatrix} x_1 \\ x_2 \\ \cdot \\ \cdot \\ \cdot \\ x_d \end{bmatrix}$$



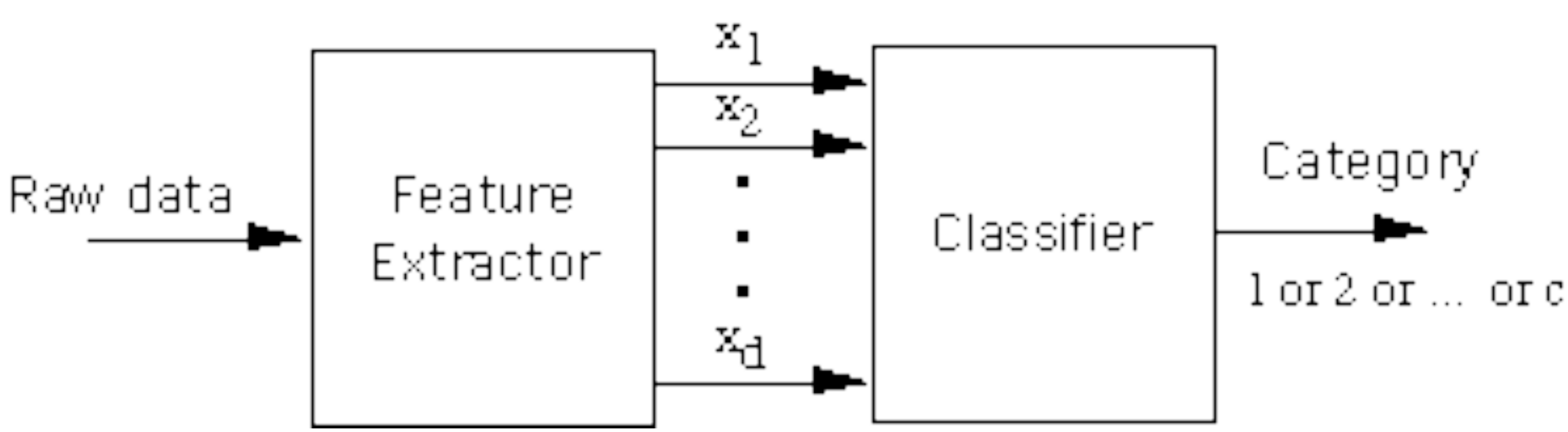
Feature Vectors

- Example of a 2D Feature Space
  - **x1** = shape (e.g. nr of sides)
  - **x2** = size (e.g. some numeric value)



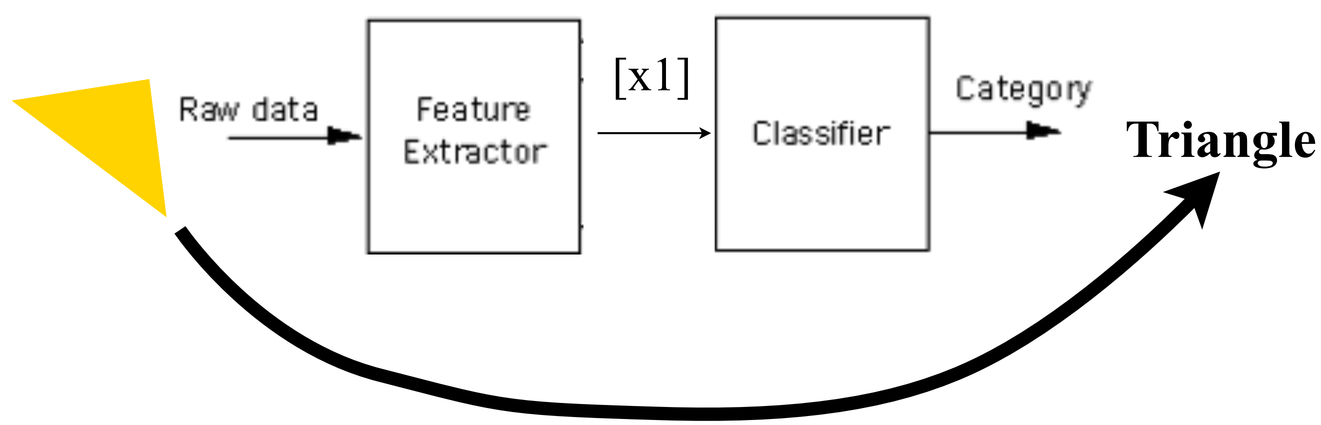
The Classical Model for PR

1. A **Feature Extractor** extracts features from raw data (e.g. audio, image, weather data, etc)
2. A **Classifier** receives X and assigns it to one of c categories, Class 1, Class 2, ..., Class c (i.e. labels the raw data).



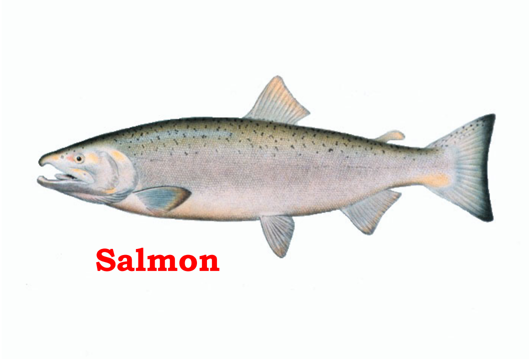
The Classical Model for PR

- Example: classify graphic objects according to their shape
  - Feature extracted:
    - nr. of sides (x1)
  - Classifier:
    - 0 sides => circle
    - 3 sides => triangle
    - (4 sides => rectangle)
  - How does the classifier know that a circle has no sides and that a triangle has 3 sides?!



A Case Study: Fish Classification

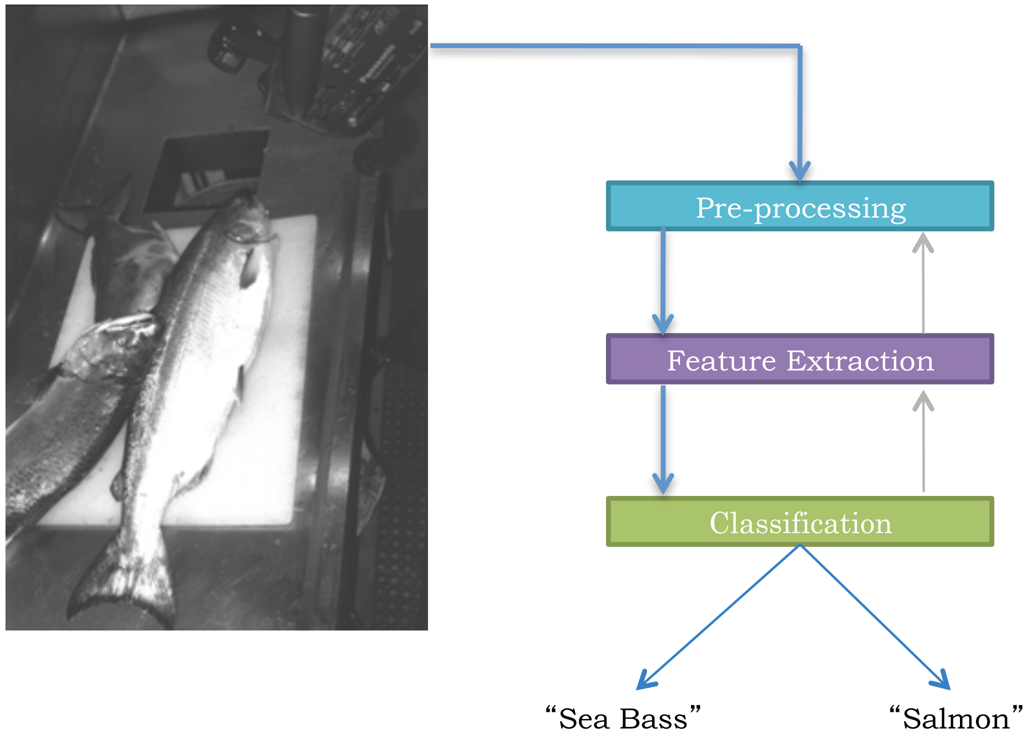
- Problem:
  - sort incoming fish on a conveyor belt according to species
  - Assume only two classes exist:
    - Sea Bass and Salmon



Luís Gustavo Martins - lmartins@porto.ucp.pt

A Case Study: Fish Classification

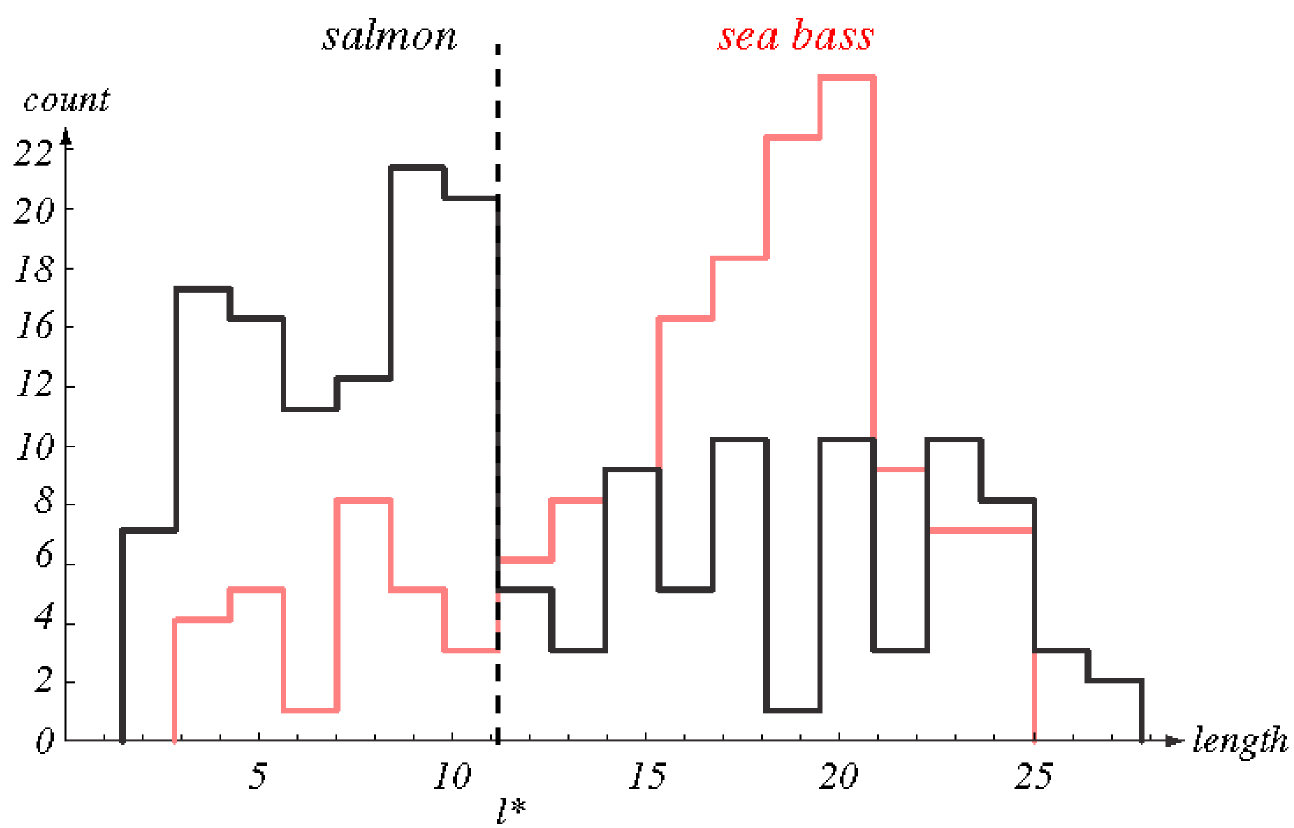
- What kind of information can distinguish one species from the other?
  - length, width, weight, number and shape of fins, tail shape, etc.
- What can cause problems during sensing?
  - lighting conditions, position of fish on the conveyor belt, camera noise, etc.
- What are the steps in the process?
  1. Capture image.
  2. Isolate fish
  3. Take measurements
  4. Make decision



A Case Study: Fish Classification

- Selecting Features
  - Assume a fisherman told us that a sea bass is generally longer than a salmon.
  - We can use length as a feature and decide between sea bass and salmon according to a threshold on length.
  - How can we choose this threshold?

Histograms of the length feature for two types of fish in training samples. How can we choose the threshold l* to make a reliable decision?



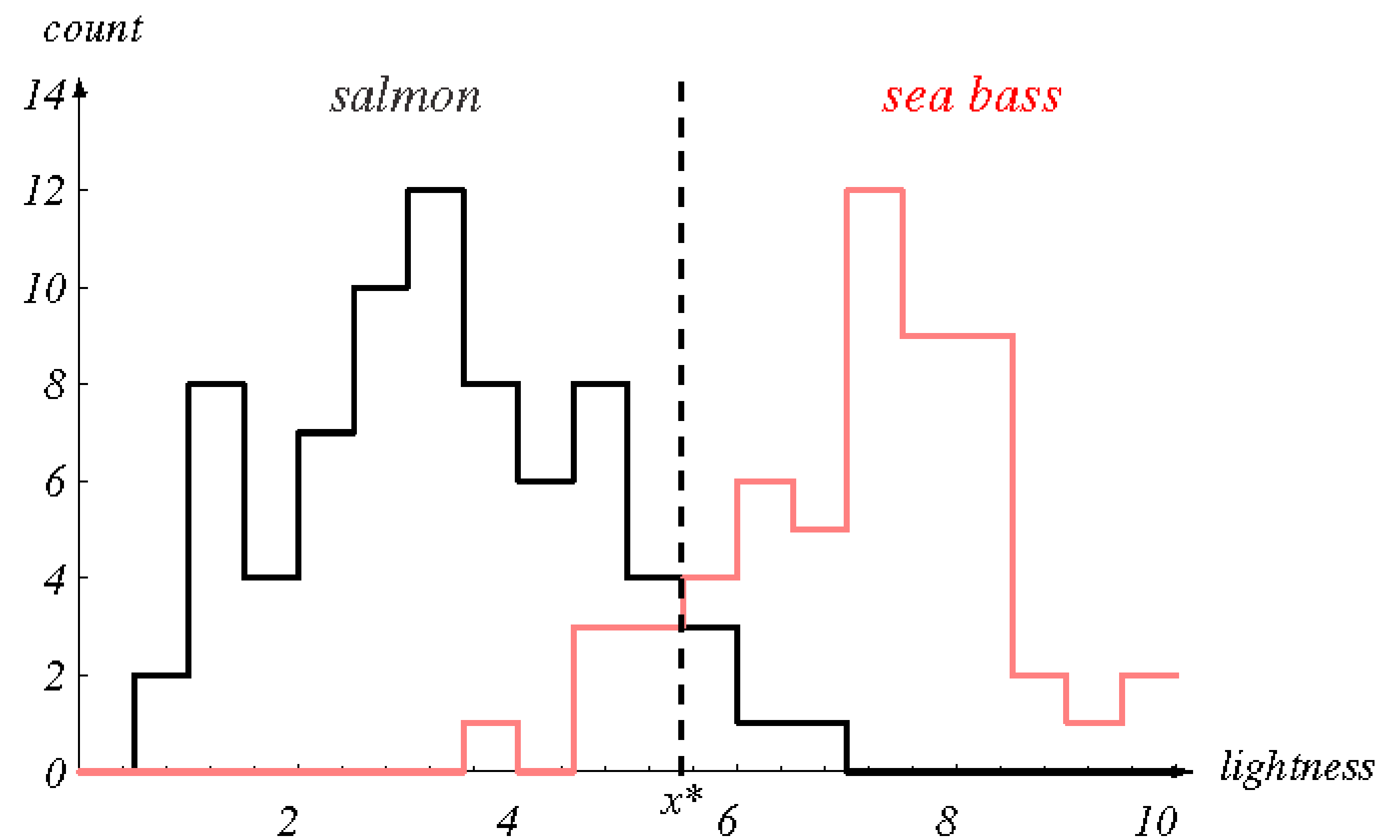
Even though "sea bass" is longer than "salmon" on the average, **there are many examples of fish where this observation does not hold...**

Luís Gustavo Martins - lmartins@porto.ucp.pt

A Case Study: Fish Classification

- Selecting Features
  - Let's try another feature and see if we get better discrimination
    - ➡Average Lightness of the fish scales

Histograms of the lightness feature for two types of fish in training samples.



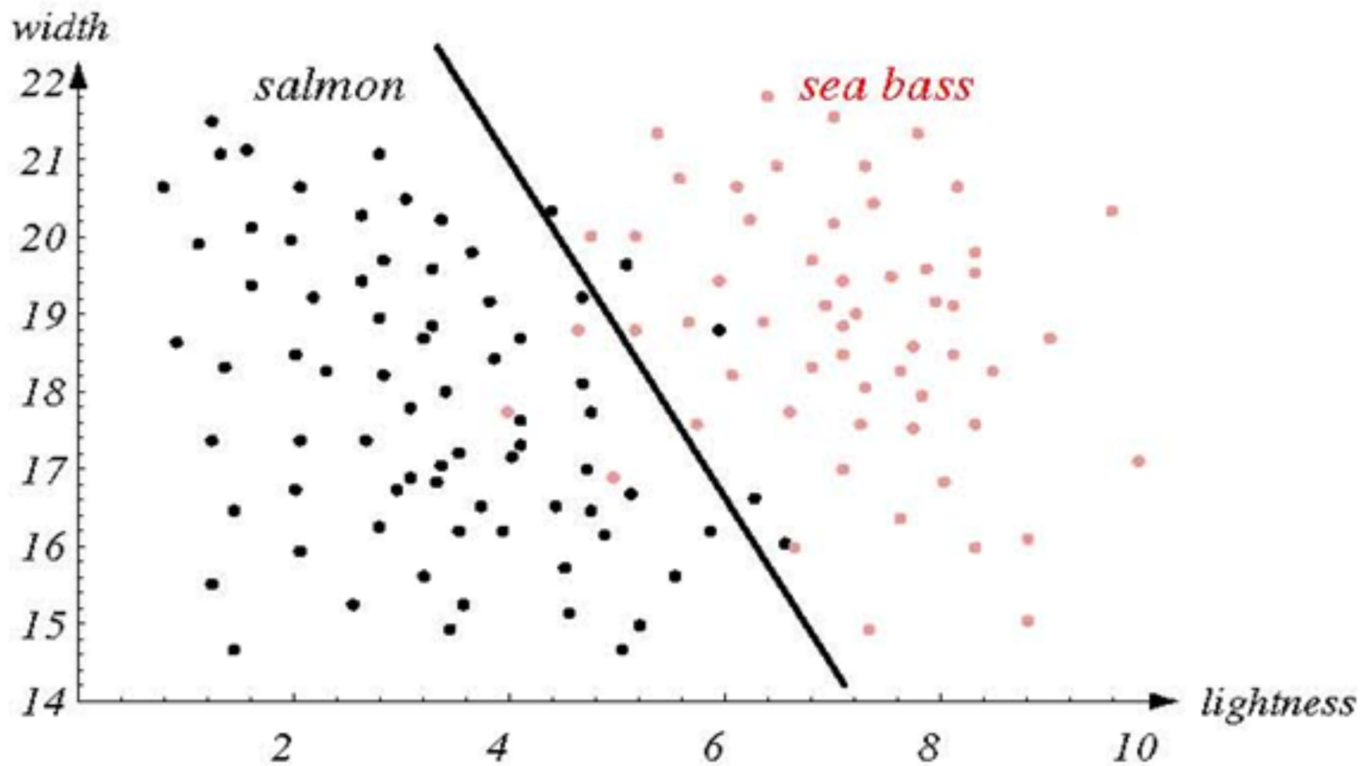
It looks easier to choose the threshold x$*$ but we still cannot make a perfect decision.

A Case Study: Fish Classification

- Multiple Features
  - Single features might not yield the best performance.
  - To improve recognition, we might have to use more than one feature at a time.
  - Combinations of features might yield better performance.
  - Assume we also observed that sea bass are typically wider than salmon.

$$\begin{bmatrix} x_1 \\ x_2 \end{bmatrix} \quad \begin{matrix} x_1 : lightness \\ x_2 : width \end{matrix}$$

Each fish image is now represented by a point in this 2D feature space



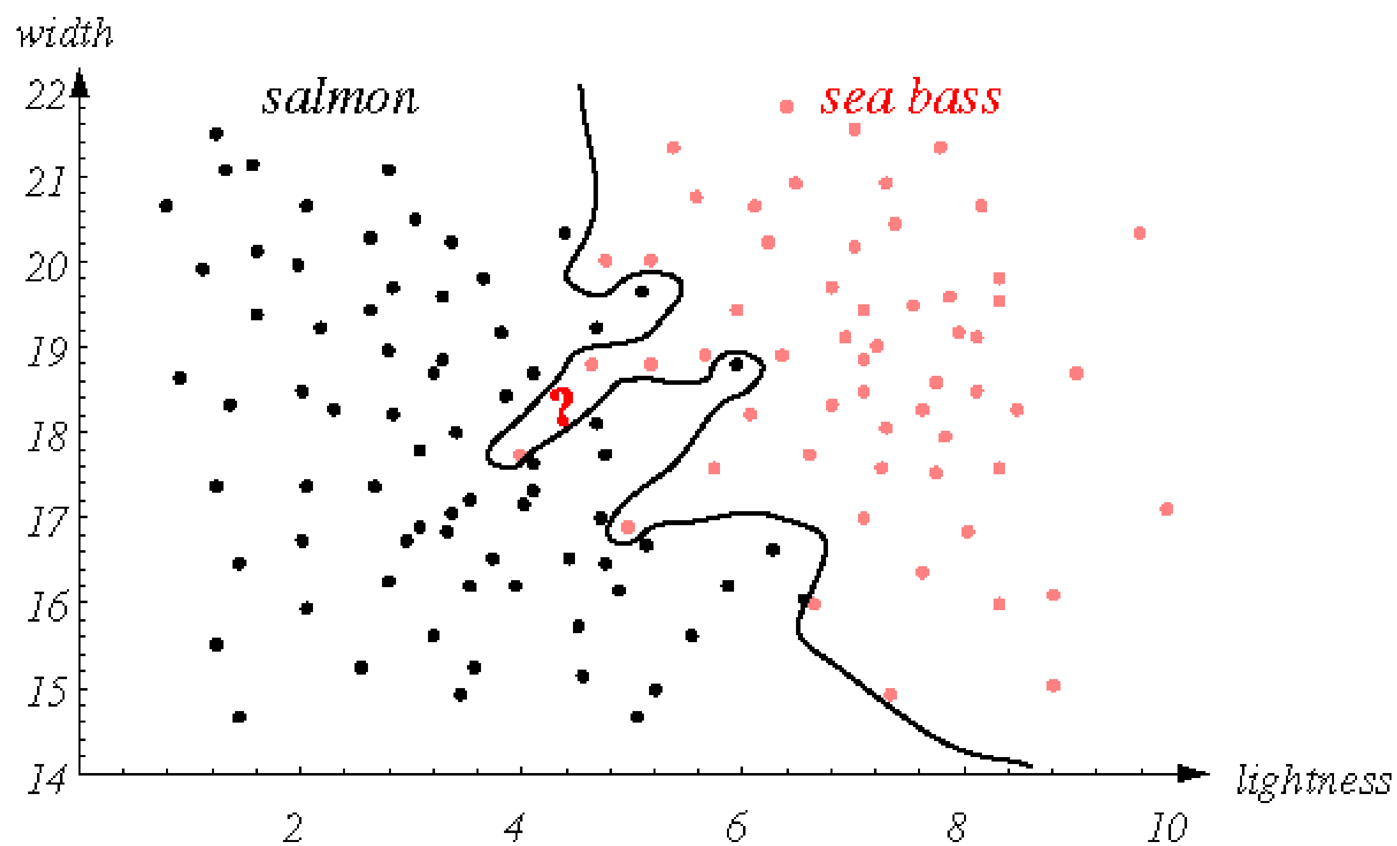
Scatter plot of lightness and width features for training samples. **We can draw a decision boundary to divide the feature space into two regions. Does it look better than using only lightness?**

Luís Gustavo Martins - lmartins@porto.ucp.pt

A Case Study: Fish Classification

- Designing a **Classifier**
  - Can we do better with another decision rule?
  - More complex models result in more complex boundaries.

We may distinguish training samples perfectly but **how can we predict how well we can generalize to unknown samples?**



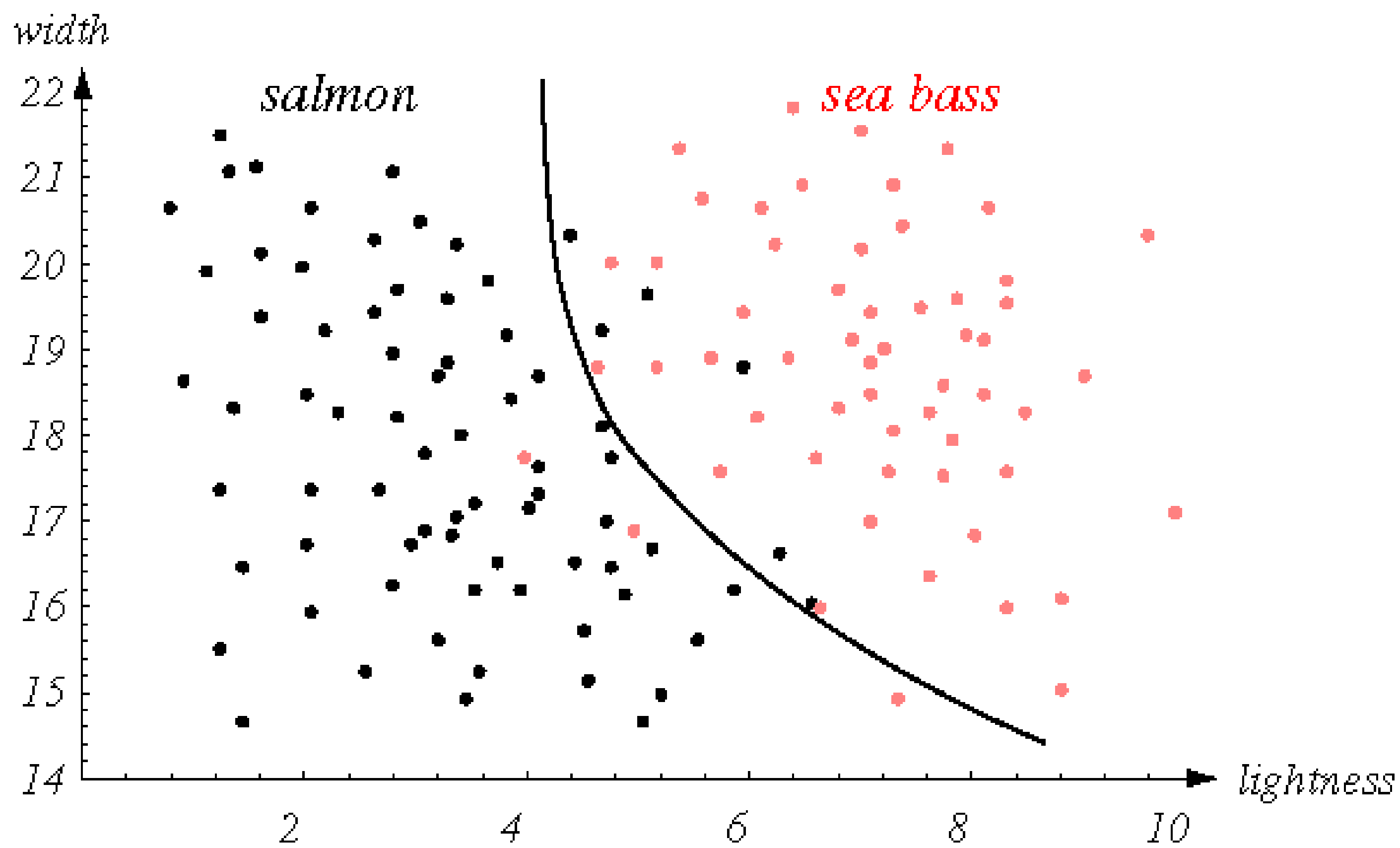
DANGER OF **OVER FITTING!!**

CLASSIFIER WILL FAIL TO GENERALIZE TO NEW DATA...

A Case Study: Fish Classification

- Designing a **Classifier**
  - How can we manage the tradeoff between complexity of decision rules and their performance to unknown samples?

Different criteria lead to different decision boundaries



Feature Extraction

- Designing a **Feature Extractor**
  - Its design is **problem specific** (e.g. features to extract from graphic objects may be quite different from sound events...)
  - The ideal feature extractor would produce the same feature vector X for all patterns in the same class, and different feature vectors for patterns in different classes.
  - In practice, different inputs to the feature extractor will always produce different feature vectors, but we hope that the **within-class variability** is small relative to the **between-class variability**.
- **Designing a good set of features is sometimes "more of an art than a science"...**

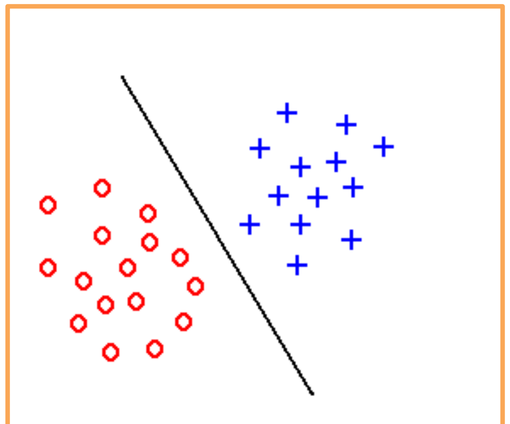
Feature Extraction

- Multiple Features
  - Does adding more features always improve the results?
    - No!! So we must:
      - Avoid unreliable features.
      - Be careful about correlations with existing features.
      - Be careful about measurement costs.
      - Be careful about noise in the measurements.
  - **Is there some curse for working in very high dimensions?**
    - YES THERE IS! ==> **CURSE OF DIMENSIONALITY**
      - ➡**thumb rule: n >= d(d-1)/2** n = nr of examples in training dataset
        d = nr of features

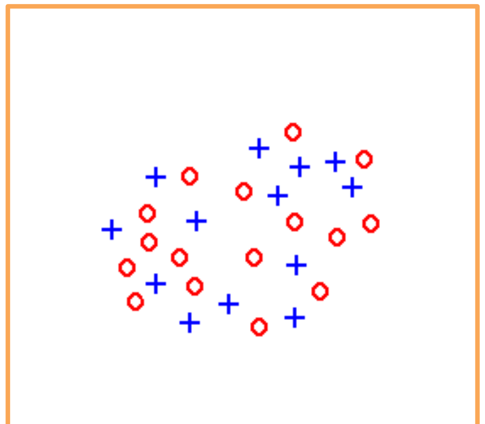
Luís Gustavo Martins - lmartins@porto.ucp.pt

Feature Extraction

- Problem: Inadequate Features
  - features simply do not contain the information needed to separate the classes, it doesn't matter how much effort you put into designing the classifier.
  - **Solution**: go back and design better features.



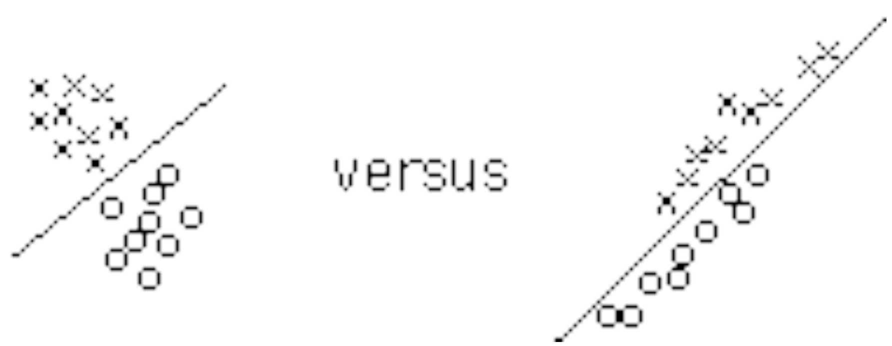
"Good" features



"Bad" features

Feature Extraction

- Problem: Correlated Features
  - Often happens that two features that were meant to measure different characteristics are influenced by some common mechanism and tend to vary together.
    - E.g. the perimeter and the maximum width of a figure will both vary with scale; larger figures will have both larger perimeters and larger maximum widths.
  - This degrades the performance of a classifier based on Euclidean distance to a template.
    - A pattern at the extreme of one class can be closer to the template for another class than to its own template. A similar problem occurs if features are badly scaled, for example, by measuring one feature in microns and another in kilometers.
  - **Solution**: (Use other metrics, e.g. Mahalanobis...) or extract features known to be uncorrelated!



Designing a Classifier

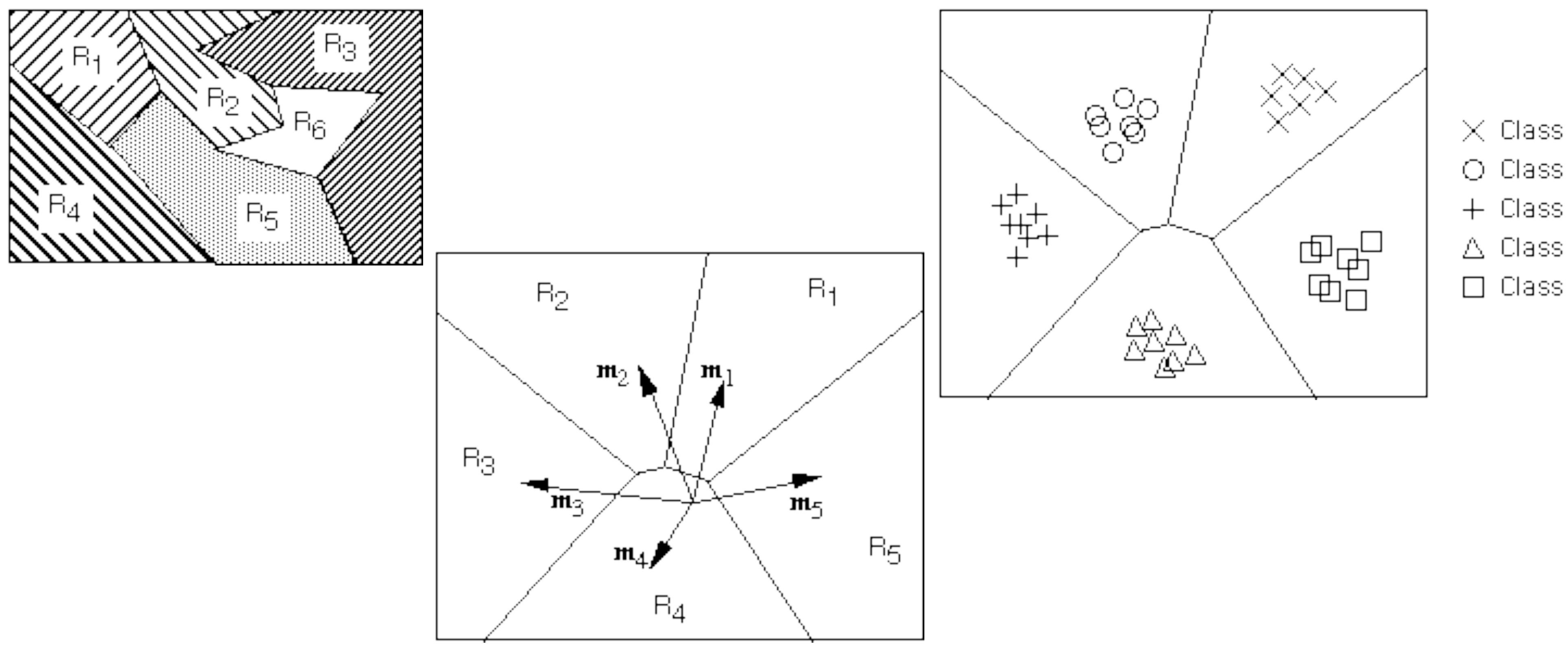
- Model selection:
  - Domain dependence and prior information.
  - Definition of design criteria.
  - Parametric vs. non-parametric models.
  - Handling of missing features.
  - Computational complexity.
  - Types of models: templates, decision-theoretic or statistical, syntactic or structural, neural, and hybrid.
  - How can we know how close we are to the true model underlying the patterns?

Luís Gustavo Martins - lmartins@porto.ucp.pt

Designing a Classifier

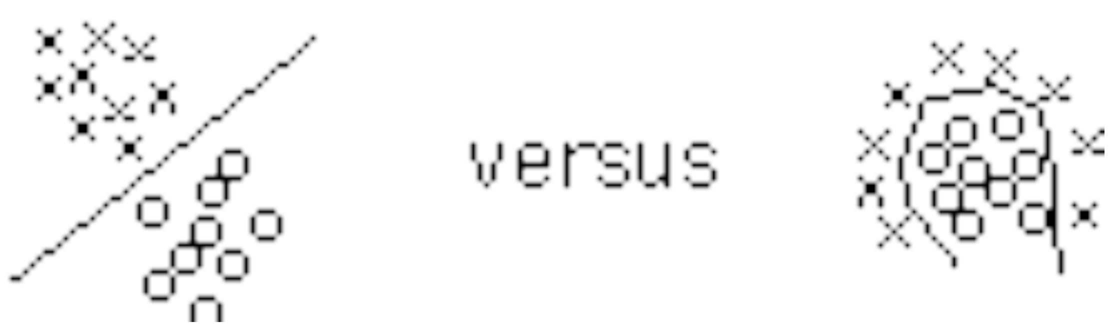
- Designing a **Classifier**
  - How can we manage the tradeoff between complexity of decision rules and their performance to unknown samples?

Different criteria lead to different decision boundaries



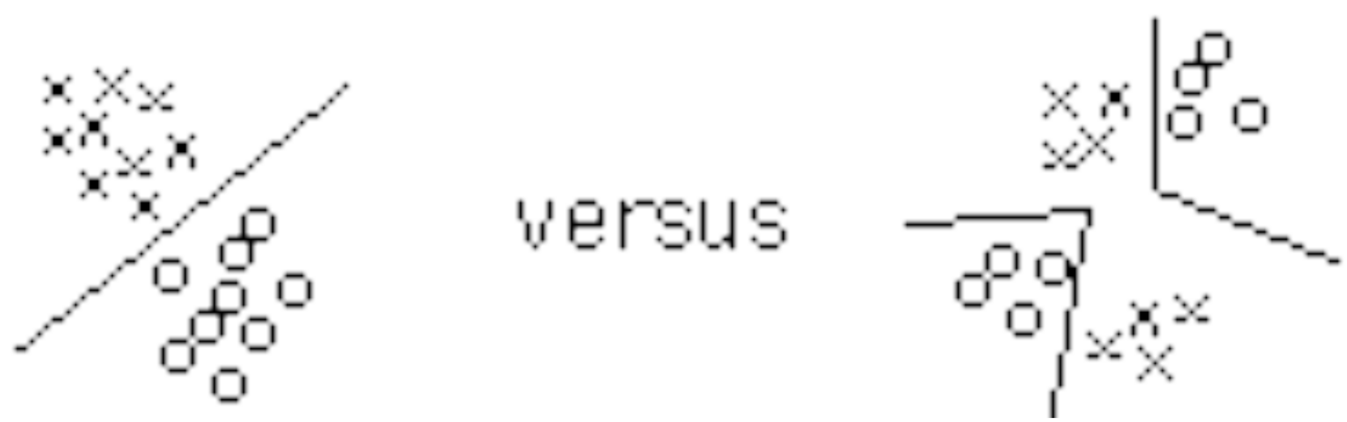
Designing a Classifier

- Problem: Curved Boundaries
  - linear boundaries produced by a minimum-Euclidean-distance classifier may not be flexible enough.
    - For example, if x1 is the perimeter and x2 is the area of a figure, x1 will grow linearly with scale, while x2 will grow quadratically. This will "warp" the feature space and prevent a linear discriminant function from performing well.
  - **Solutions:**
    - Redesign the feature set (e.g., let x2 be the square root of the area)
    - *Try using Mahalanobis distance, which can produce quadratic decision boundaries*
    - *Try using a neural network (beyond the scope of these notes; see Haykin)*



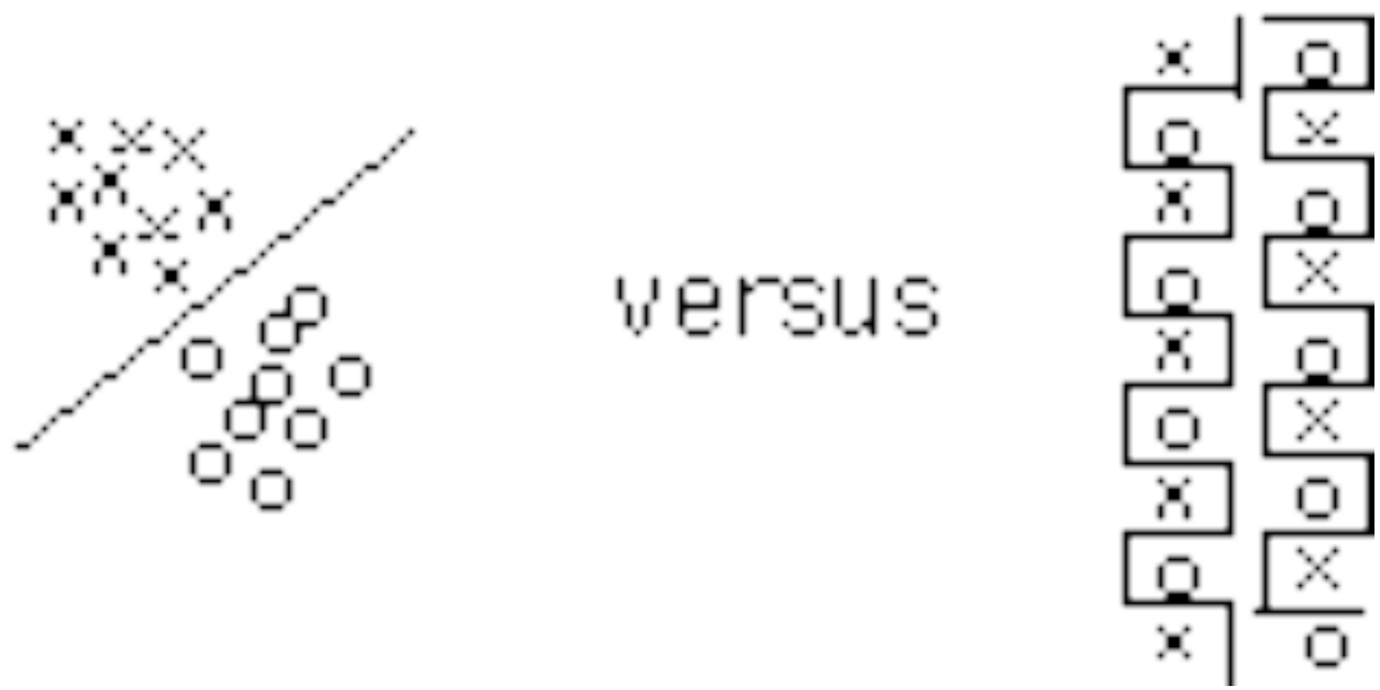
Designing a Classifier

- Problem: Subclasses in the dataset
  - frequently happens that the classes defined by the end user are not the "natural" classes...
  - **Solution**: use CLUSTERING.



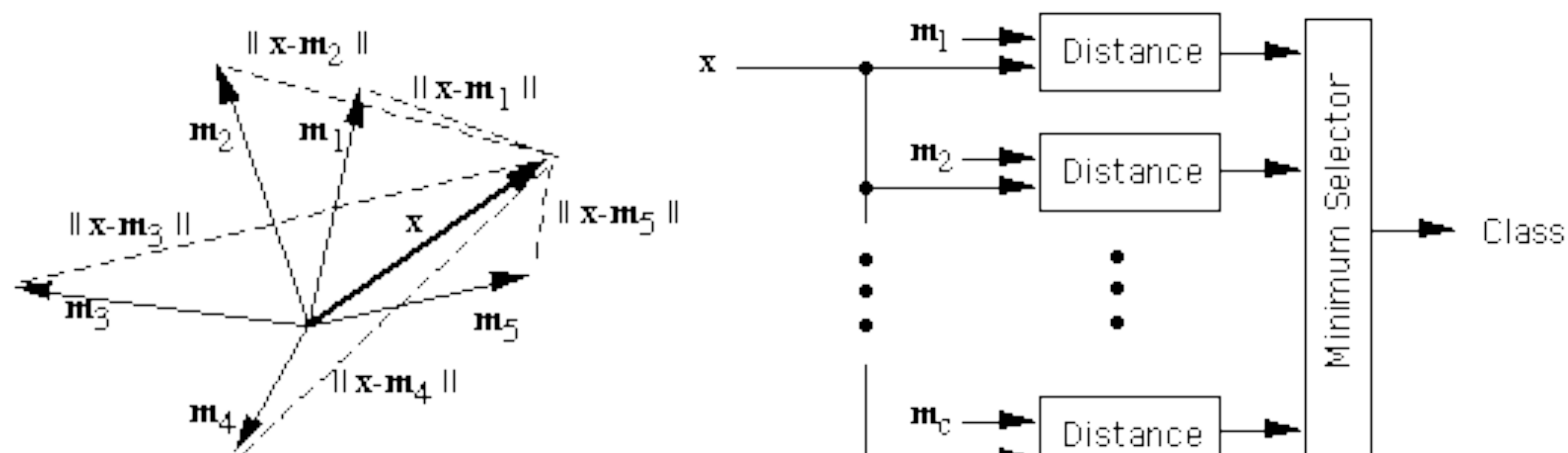
Designing a Classifier

- Problem: Complex Feature Space
  - **Solution**: use different type of Classifier...



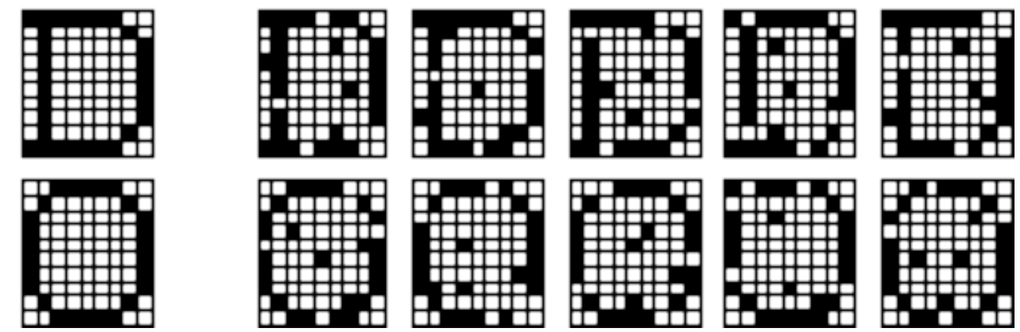
Simple Classifiers

- Minimum-distance Classifiers
  - based on some specified **“metric” ||x-m||**
  - e.g. Template Matching



Simple Classifiers

- Template Matching



TEMPLATES NOISY EXAMPLES

  – To classify one of the noisy examples, simply compare it to the two templates. This can be done in a couple of equivalent ways:

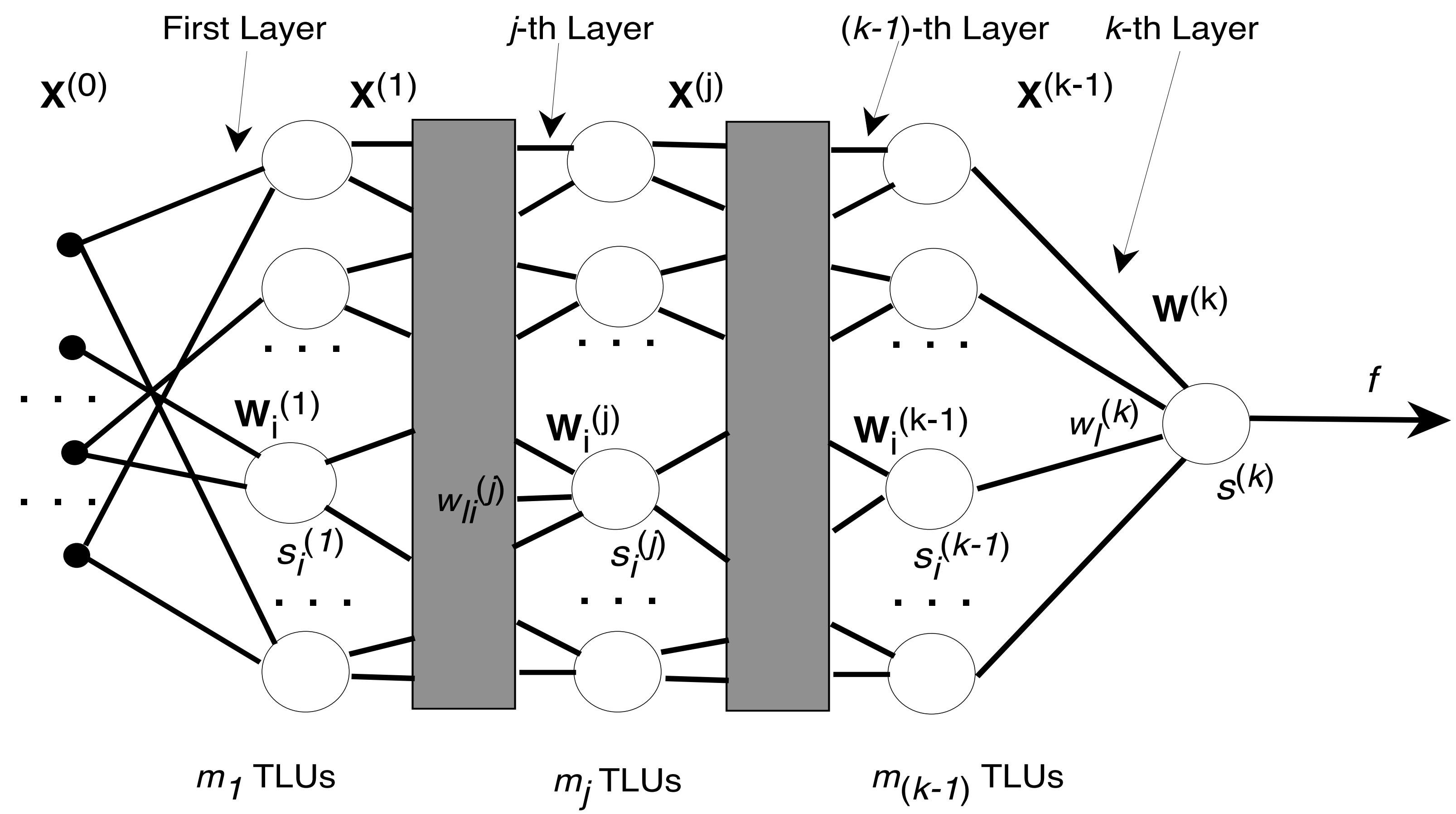
    1. Count the number of agreements. Pick the class that has the maximum number of agreements. **This is a maximum correlation approach**.
    2. Count the number of disagreements. Pick the class with the minimum number of disagreements. **This is a minimum error approach.**

- Works well when the variations within a class are due to "additive noise", and there are no other distortions of the characters -- translation, rotation, shearing, warping, expansion, contraction or occlusion.

Simple Classifiers

- Metrics
  - different ways of measuring distance:
    - **Euclidean metric:**
      - $|| u || = sqrt(u12 + u22 + ... + ud2)$
    - **Manhattan (or taxicab) metric:**
      - $|| u || = |u1| + |u2| + ... + |ud|$
    - Contours of constant...
      - ... Euclidean distance are circles (or spheres)
      - ... Manhattan distance are squares (or boxes)
      - ... Mahalanobis distance are ellipses (or ellipsoids)

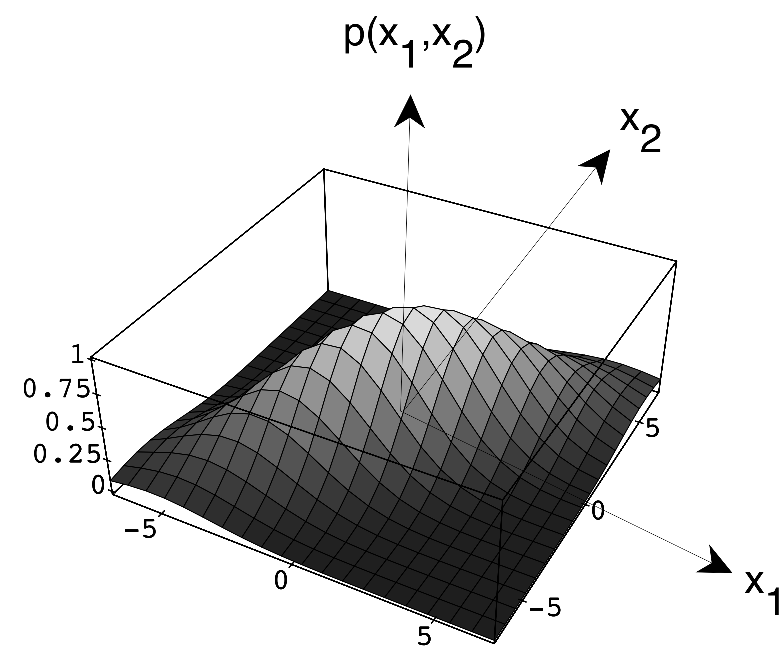
Classifiers: Neural Networks

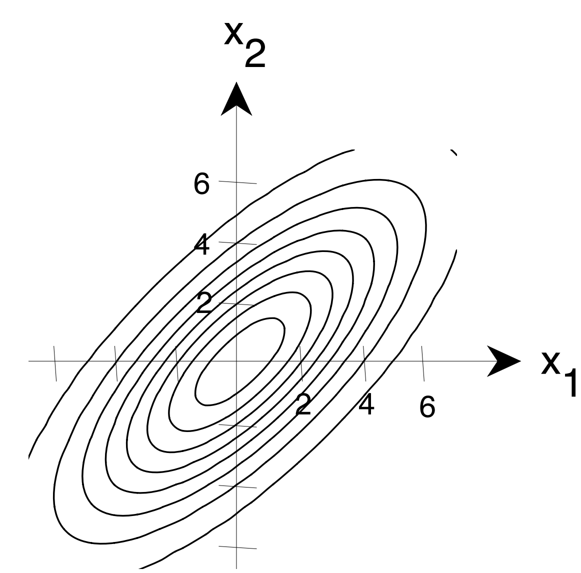


http://robotics.stanford.edu/people/nilsson/MLBOOK.pdf

Gaussian Modeling

$$N(\mu,\Sigma) = \frac{1}{\sqrt{(2\pi)^n.\left|\Sigma\right|}} e^{-\frac{1}{2}(x-\mu)^T \Sigma^{-1}(x-\mu)}, n = 2$$

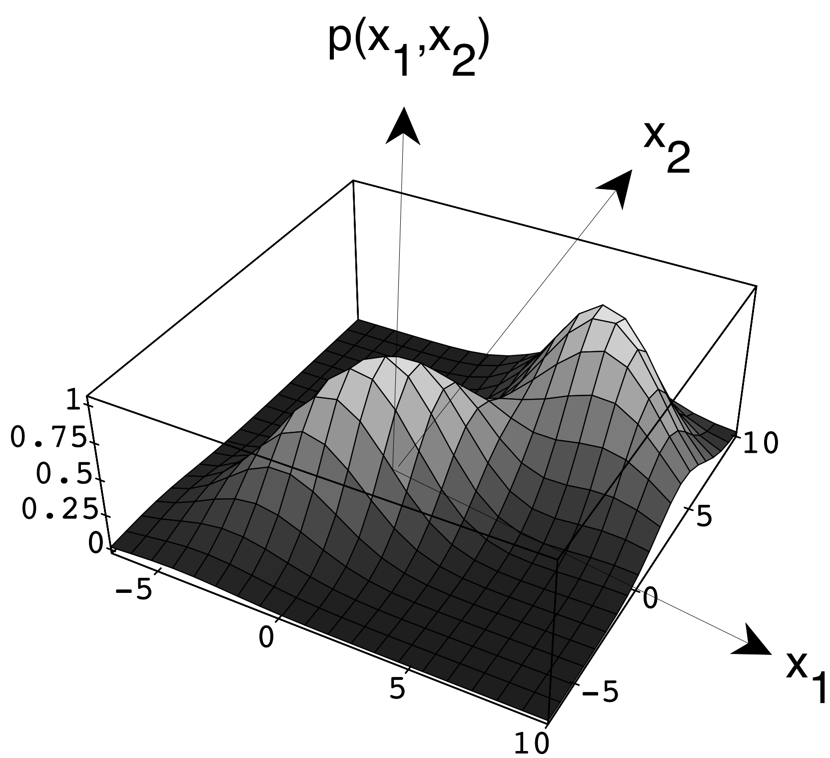


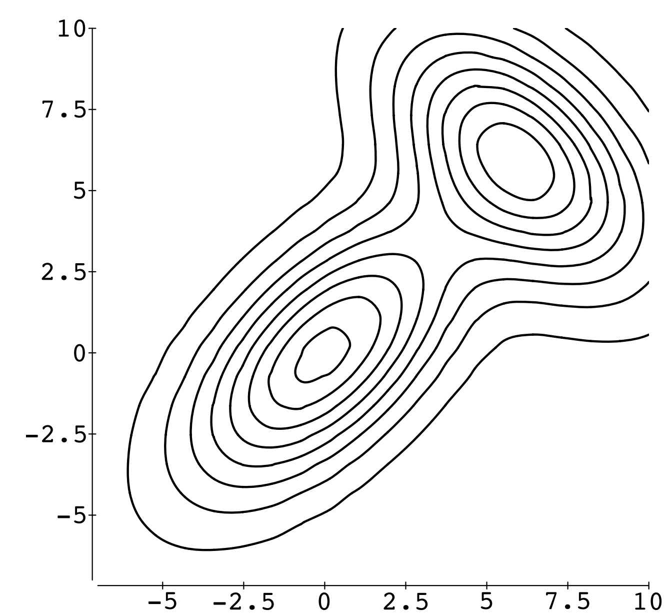


http://robotics.stanford.edu/people/nilsson/MLBOOK.pdf

Gaussian Mixture Models

- Use multiple Gaussians to model the data

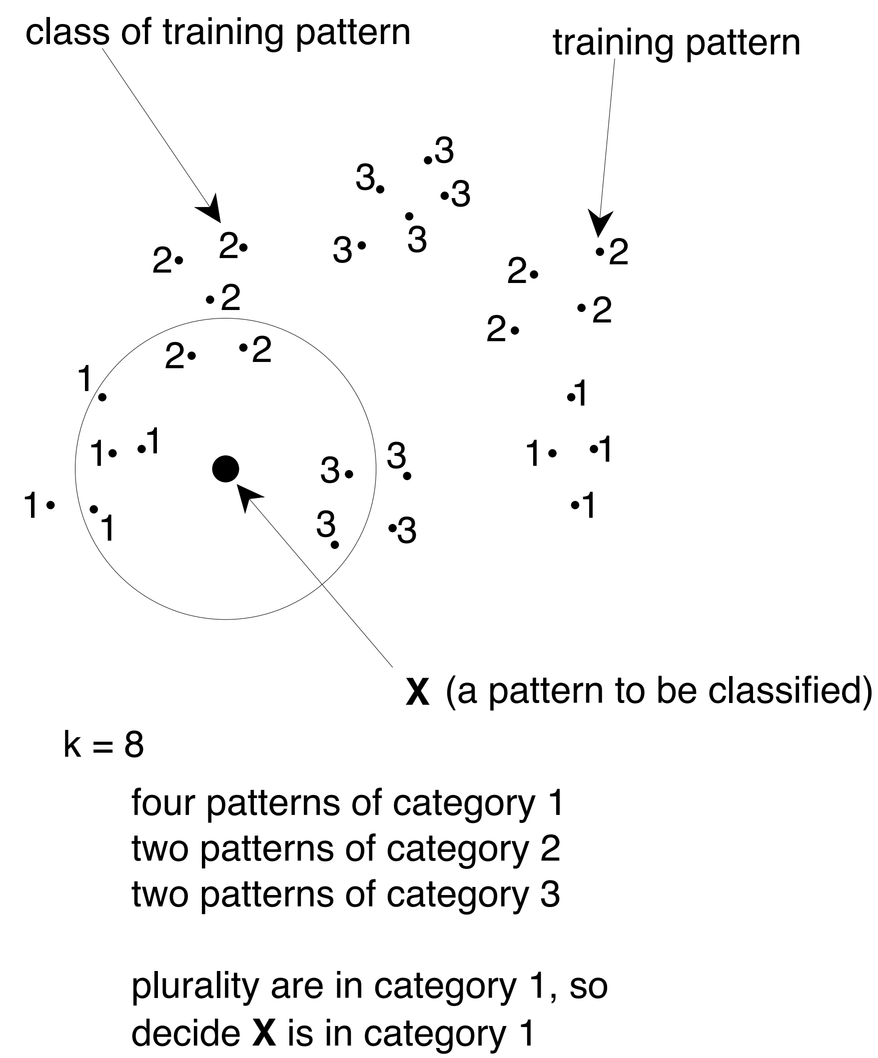




http://robotics.stanford.edu/people/nilsson/MLBOOK.pdf

Classifiers: kNN

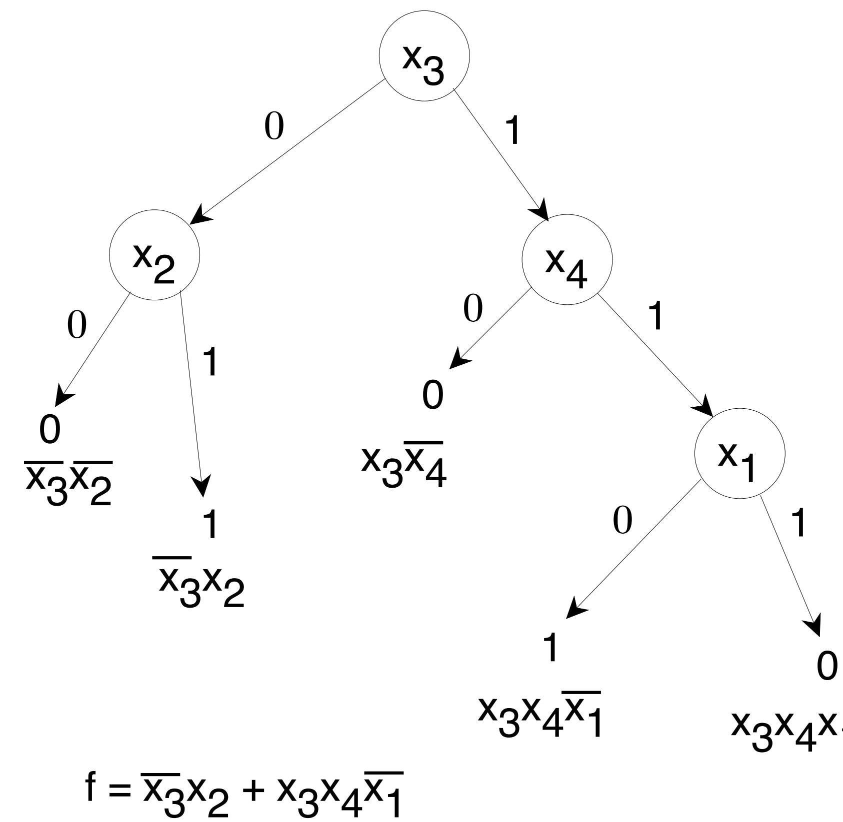
- k-Nearest Neighbours Classifier
  - Lazy Classifier
    - no training is actually performed (hence, lazy ;-))
  - An example of Instance Based Learning



http://robotics.stanford.edu/people/nilsson/MLBOOK.pdf

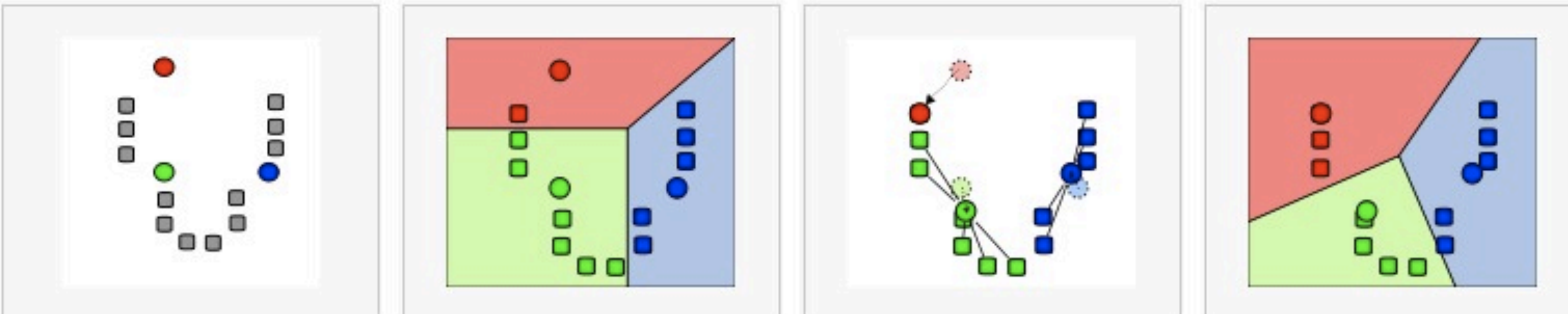
Decision Trees

- Learn rules from data
- Apply each rule at each node
- classification is at the leafs of the tree



http://robotics.stanford.edu/people/nilsson/MLBOOK.pdf

Clustering: k-means



1) *k* initial "means" (in this case *k*=3) are randomly selected from the data set (shown in color).

2) *k* clusters are created by associating every observation with the nearest mean. The partitions here represent the Voronoi diagram generated by the means.

3) The centroid of each of the *k* clusters becomes the new means.

4) Steps 2 and 3 are repeated until convergence has been reached.

http://en.wikipedia.org/wiki/K-means_clustering

Evaluating a Classifier

- **Training Set**
  - used for training the classifier
- **Testing Set**
  - examples not used for training
  - avoids overfitting to the data
  - tests generalization abilities of the trained classifiers
- Data sets are usually hard to obtain...
  - Labeling examples is time and effort consuming
  - Large labeled datasets usually not widely available
  - Requirement of separate training and testing datasets imposes higher difficulties...
  - Use **Cross-Validation** techniques!

Evaluating a Classifier

- **Confusion Matrix**

Example confusion matrix

		Predicted		
		Cat	Dog	Rabbit
Actual	Cat	5	3	0
	Dog	2	3	1
	Rabbit	0	2	11

		prediction outcome		
		p	*n*	total
actual value	*p'*	True Positive	False Negative	P'
	n'	False Positive	True Negative	N'
	total	P	N	

http://en.wikipedia.org/wiki/Confusion_matrix

Evaluating a Classifier

- **Costs of Error**
  - We should also consider costs of different errors we make in our decisions. For example, if the fish packing company knows that:
    - Customers who buy salmon will object vigorously if they see sea bass in their cans.
    - Customers who buy sea bass will not be unhappy if they occasionally see some expensive salmon in their cans.
- **How does this knowledge affect our decision?**

Luís Gustavo Martins - lmartins@porto.ucp.pt

Pattern Recognition References

- Introduction to Machine Learning - Draft of Incomplete Notes, by Nils J. Nilsson (http://robotics.stanford.edu/people/nilsson/MLBOOK.pdf)
- Pattern Recognition general links (http://cgm.cs.mcgill.ca/~godfried/teaching/pr-web.html)
- PR for HCI - Richard O. Duda notes (http://www.cs.princeton.edu/courses/archive/fall08/cos436/Duda/PR_home.htm)
- Talal Alsubaie PR Slides (http://www.slideshare.net/t_subaie/pattern-recognition-presentation)
- Y-H Pao, Adaptive Pattern Recognition and Neural Networks (Addison-Wesley Publishing Co., Reading, MA, 1989). Augments statistical procedures by including neural networks, fuzzy-set representations and self-organizing feature maps.
- R. Schalkoff, Pattern Recognition: Statistical, Structural and Neural Approaches (John Wiley & Sons, New York, 1992). A clear presentation of the essential ideas in three important approaches to pattern recognition.
- M. Stefik, Introduction to Knowledge Systems (Morgan Kaufmann, San Francisco, CA, 1995). Although this excellent book is not oriented towards pattern recognition per se, the methods it presents are the basis for knowledge-based pattern recognition.

Pattern Recognition References

- P.A. Devijver and J. Kittler, Pattern Recognition: A Statistical Approach (Prentice-Hall International, Englewood Cliffs, NJ, 1980). A very clear presentation of the mathematical foundations.
- R. O. Duda and P. E. Hart, Pattern Classification and Scene Analysis (Wiley-Interscience, New York, 1973). Still in print after all these years.
- K. Fukunaga, Introduction to Statistical Pattern Recognition, 2nd Ed. (Academic Press, New York, 1990. A standard, widely-used textbook.
- D. J. Hand, Discrimination and Classification (John Wiley and Sons, Chichester, UK, 1981). A warmly recommended introductory book.
- S. Haykin, Neural Networks (MacMillan, NY, 1993). There are dozens of interesting books on neural networks. Haykin is an excellent, engineering-oriented textbook.
- T. Masters, Advanced Algorithms for Neural Networks (Wiley, NY, 1995). Well described by the title, with a chapter devoted to the often overlooked issue of validation.
- WEKA Data Mining Software in Java (http://www.cs.waikato.ac.nz/ml/weka/)