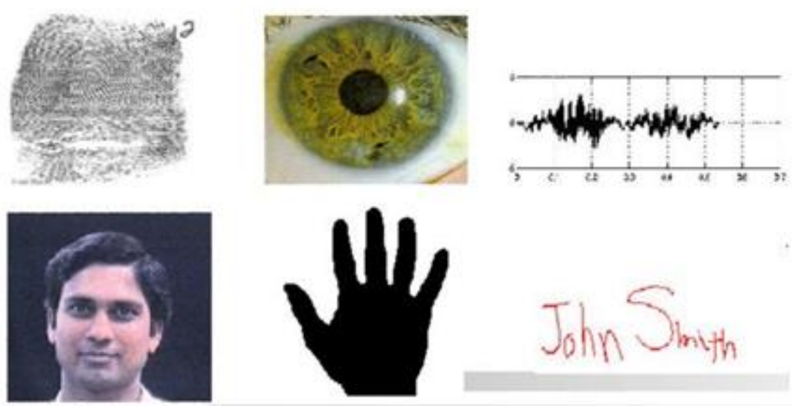
PATTERN RECOGNITION FUNDAMENTALS:

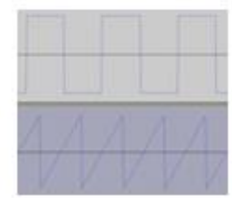
Topic to be covered

- **Basic concepts of pattern recognition and pattern classes**
- **Issues in pattern recognition system and evaluation**
- **Design concepts and methodologies**
- **Pattern recognition applications**

Basic concepts of pattern recognition

- **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, …





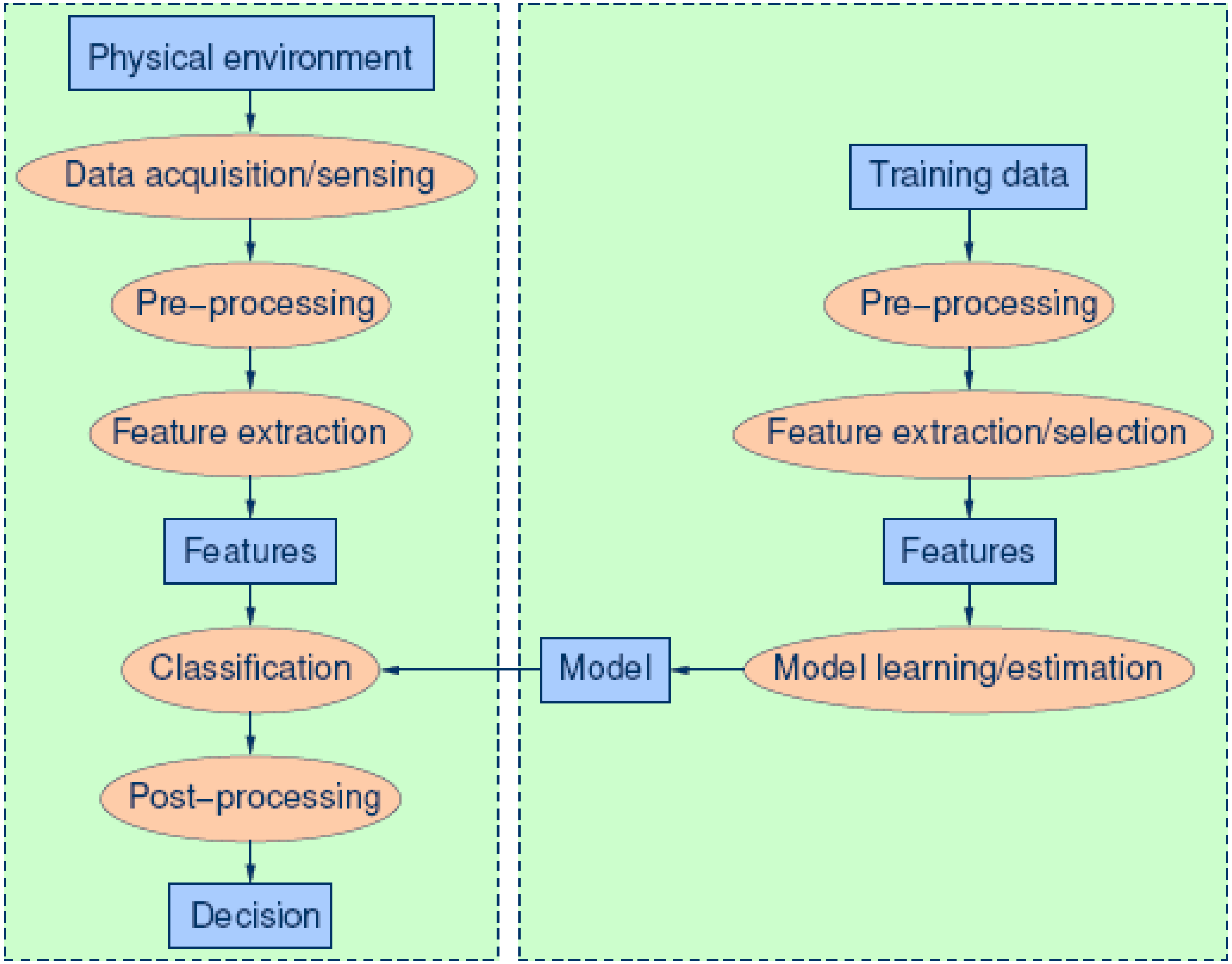
Basic concepts of pattern recognition

- A ***pattern*** is an **object**, process or event that can be given a name.
- A ***pattern class*** (or category) is a set of patterns sharing common attributes and usually originating from the same source.
- During ***recognition*** (or ***classification***) given objects are assigned to prescribed classes.
- A ***classifier*** is a machine which performs classification.

Pattern recognition approaches

- **Statistical PR:** based on underlying statistical model of patterns and pattern classes.
- **Neural networks:** classifier is represented as a network of cells modeling neurons of the human brain (connectionist approach).
- **Structural (or syntactic) PR:** pattern classes represented by means of formal structures as grammars, automata, strings, etc.

Basic Components of a Pattern Recognition System



Components of Pattern Recognition (Cont'd)

- **Data acquisition and sensing**
- **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.

    (Better for further processing)

Components of Pattern Recognition (Cont'd)

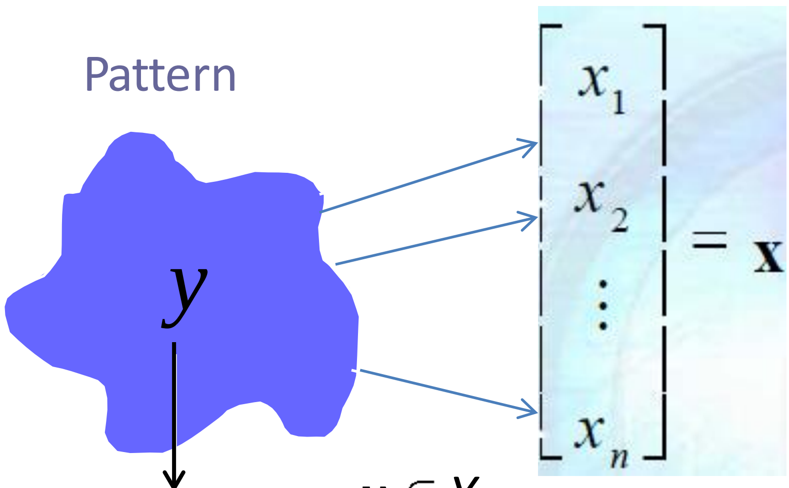
- **Model learning and estimation**
  - Learning a mapping between features and pattern groups.
- **Classification**
  - Using learned models to assign a pattern to a predefined category
- **Post-processing**
  - Evaluation of confidence in decisions.
  - Exploitation of context to improve performances.

Pattern Representation

- A pattern is represented by a set of *d* features, or attributes, viewed as a *d*-dimensional *feature vector*.

$$\mathbf{x} = (x_1, x_2, \ldots, x_d)^T$$

Basic concepts



Feature vector $\mathbf{x} \in X$

- A vector of observations (measurements).

- $\mathbf{x}$ is a point in feature space X.

Hidden state $y \in Y$

- Cannot be directly measured.

- Patterns with equal hidden state belong to the same class.

Task

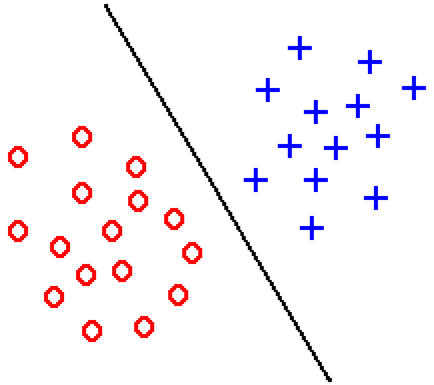
- To design a classifer (decision rule) $q : X \rightarrow Y$

  which decides about a hidden state based on an observation.

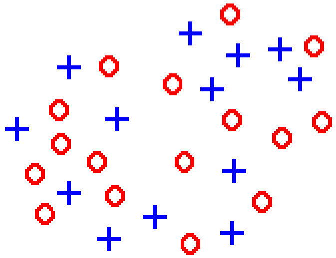
Feature Extraction

Task: to extract features which are good for classification.

Good features:
- Objects from the same class have similar feature values.
- Objects from different classes have different values.



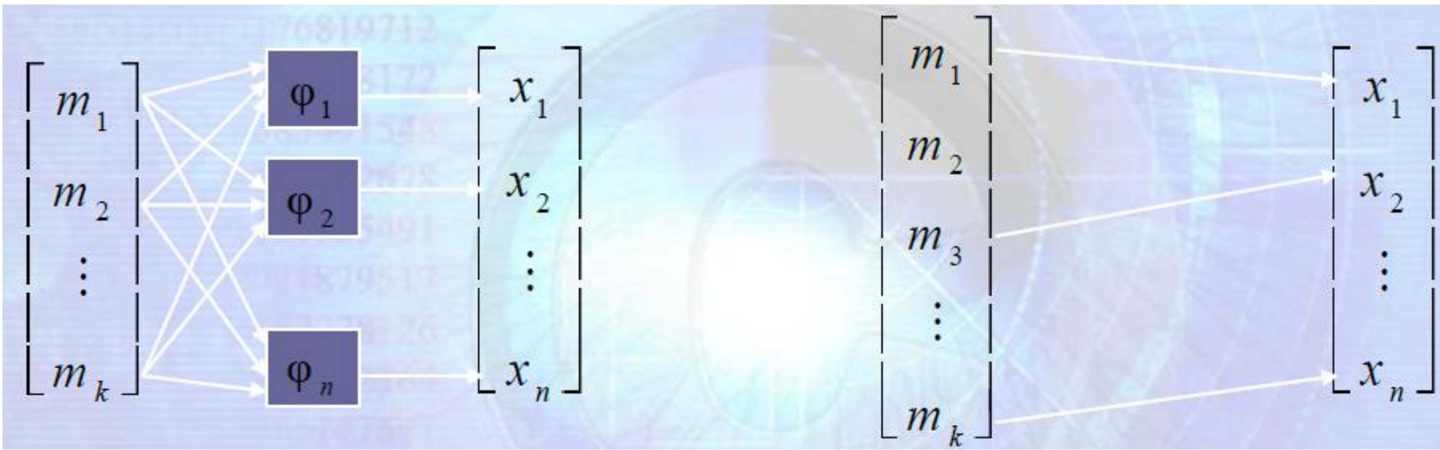
"Good" features



"Bad" features

Feature Extraction Methods

Feature extraction Feature selection



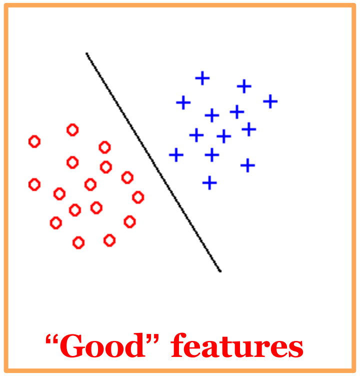
Problem can be expressed as optimization of parameters of featrure extractor $\varphi(\theta)$

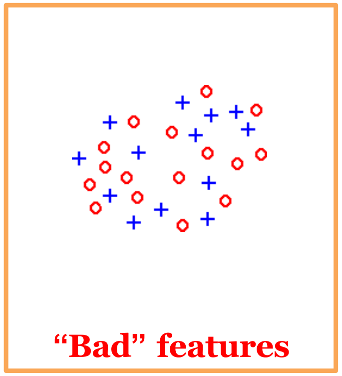
Supervised methods: objective function is a criterion of separability (discriminability) of labeled examples, e.g., linear discriminat analysis (LDA).

Unsupervised methods: lower dimesional representation which preserves important characteristics of input data is sought for, e.g., principal component analysis (PCA).

Feature Extraction

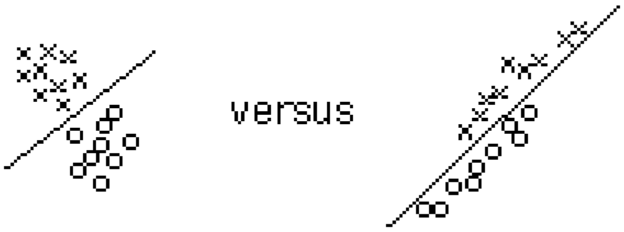
- Problem: Inadequate Features
  - Features simply do not contain the information needed to separate the classes, it doesn't matter how much effort you put into designing the classifier.
  - **Solution:** go back and design better features.





Feature Extraction

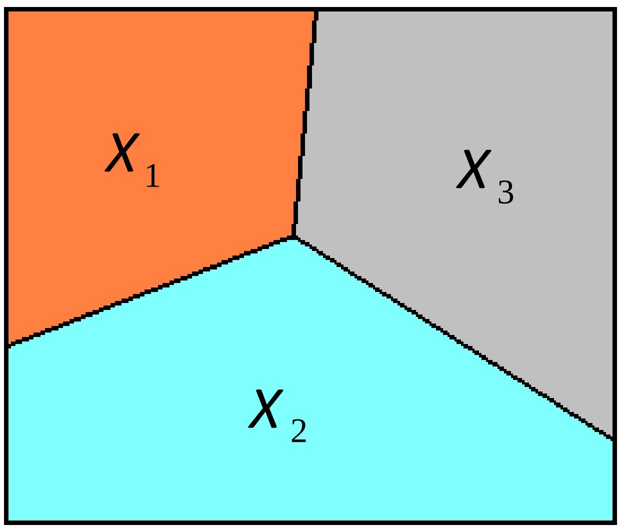
- Problem: Correlated Features
  - Often happens that two features that were meant to measure different characteristics are influenced by some common mechanism and tend to vary together.
    - E.g. the perimeter and the maximum width of a figure will both vary with scale; larger figures will have both 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!

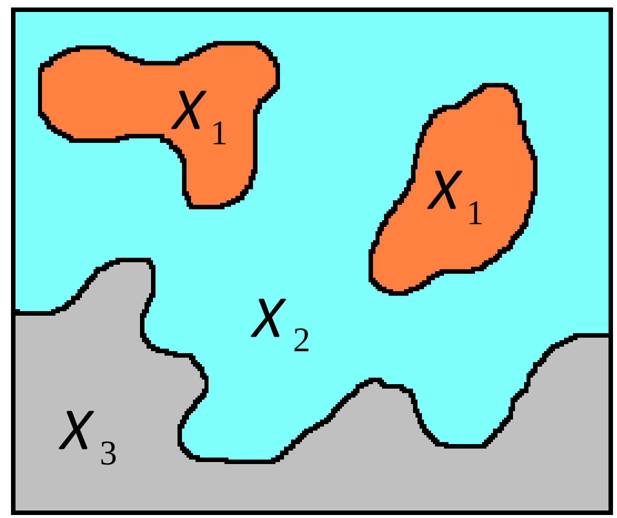


Classifier

A classifier partitions feature space X into **class-labeled regions** such that

$$X = X_1 \cup X_2 \cup \cdots \cup X_{|Y|} \qquad X_1 \cap X_2 \cap \cdots \cap X_{|Y|} = \{0\}$$



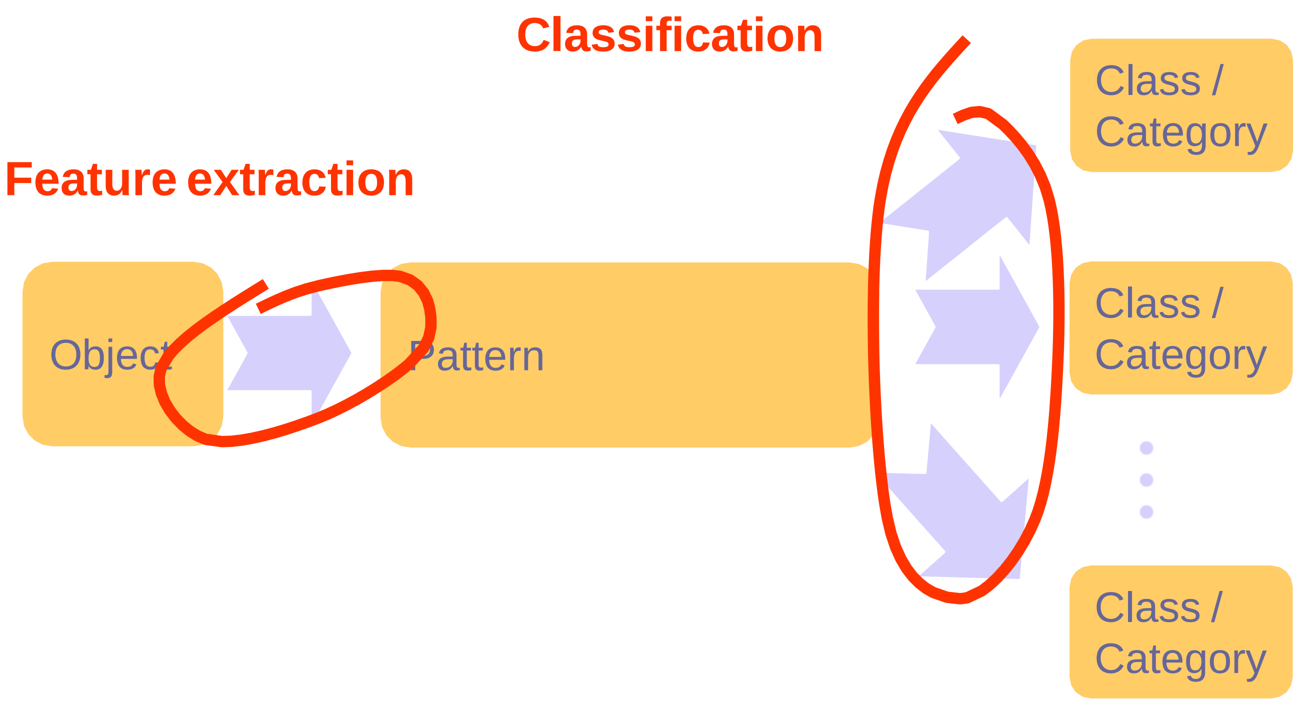


The classification consists of determining to which region a feature vector **x** belongs to.

Borders between **decision boundaries** are called decision regions.

Block diagram

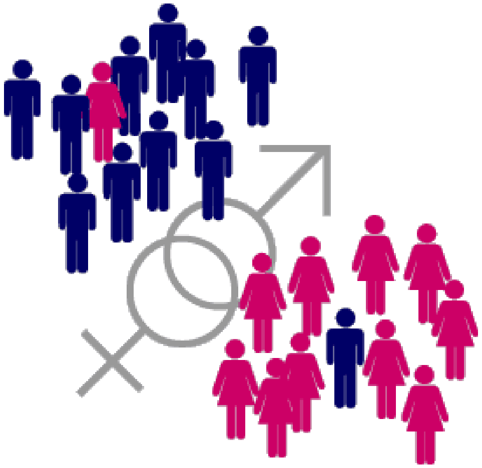
The process consists of two major operations:



Example : Gender

Assume an algorithm to recognize the gender of a student in a university, where the available input is several features of the students (of course, the gender cannot be one of the features).

The student to be classified is the ***Object***, The gender (Male or Female) are the ***Classes***, and the input which is referred to the student is the *Pattern*.



What is a *Feature*?

Feature is a scalar *x* which is quantitatively describes a property of the *Object*.

Example: Possible features of a student:

• Number of eyes	$x \in \{0, 1, 2\}$
• Hair color	$x \in \{0$ =Black, 1 =Blond, 2 =Red ,. . .$\}$
• Wear glasses or not	$x \in \{0, 1\}$
• Hair length [cm]	$x \in [0..100]$
• Shoe size [U.S]	$x \in [3.5, 4, 4.5, .. ,14]$
• Height [cm]	$x \in [40..240]$
• Weight [kg]	$x \in [30..600]$

Feature Extraction?

"When we have two or more classes, feature extraction consist of choosing those features which are most effective for preserving class separability"

Assume we choose the shoe size of the student as a feature. The selection is heuristically and seems reasonable.

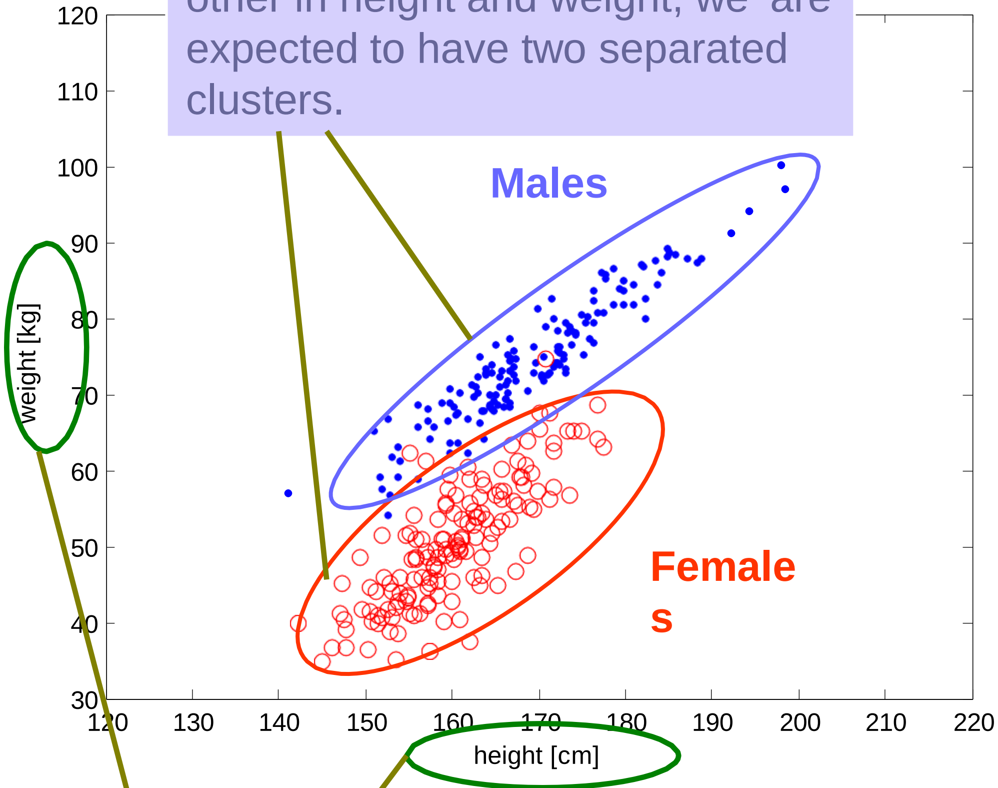
Example

Assume we are using the height and the weight of each of the students in the university as a ***pattern*.**

The height and the weight are both *features*, which span a *feature space* *V* of dimension 2.

Each of the students is represented as a point in the ***feature space*.** Patterns of male students are depicted blue and those of female students – in red.

Since the male students and the female students differ from each other in height and weight, we are expected to have two separated clusters.



Each student is characterized by a vector of two feature: (height, weight).

What is a *Class*?

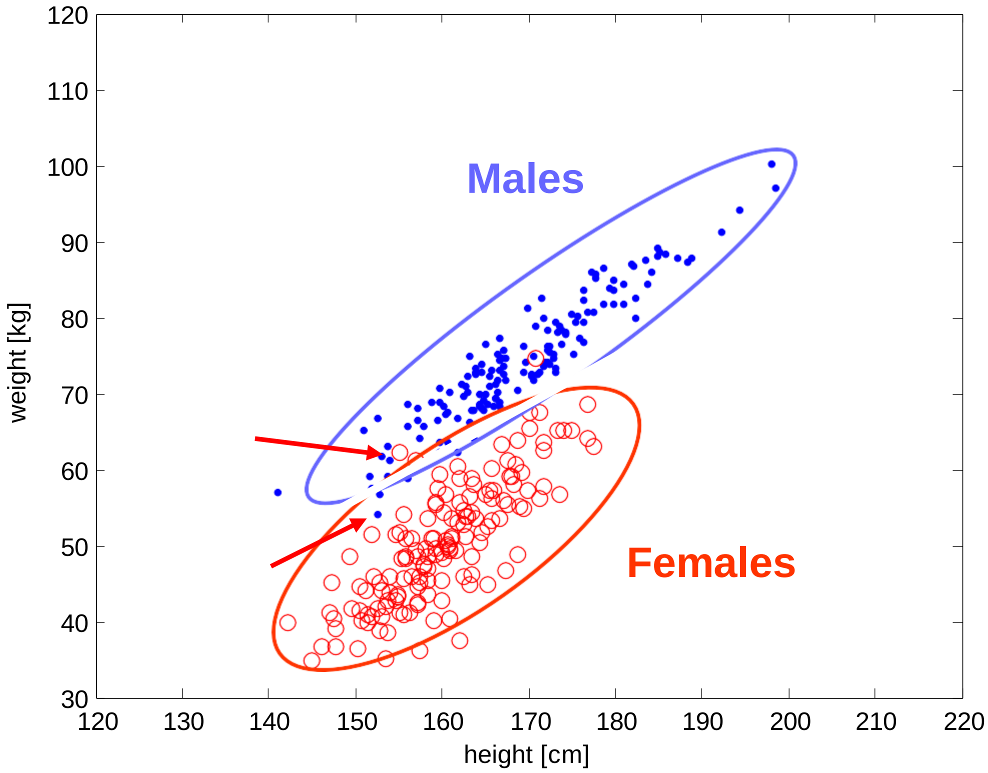
"Class is a set of ***patterns*** that share some common properties"

In example, the **Male students** and the **Female students** are two classes of objects that share a common gender.

What is Classification?

Classification is a mathematical function or algorithm which assigns a ***feature*** to one of the ***classes***.

Example:
We can draw a line between the two clusters in the gender example and every student will be classified as a female or male according to this line.

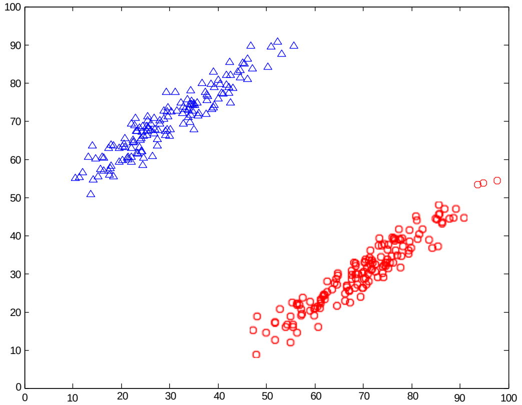


Clusters Separation

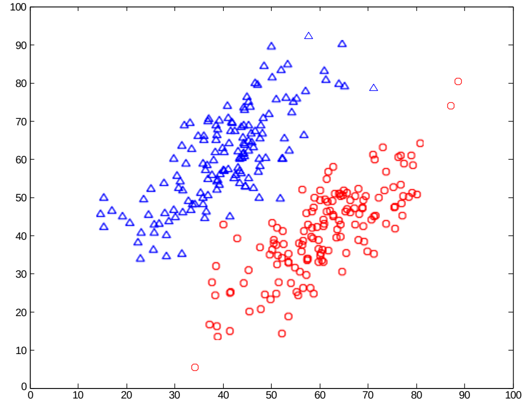
Misclassifications are a consequence of the separation of the clusters.

The separation of clusters is quantified using two major methods:

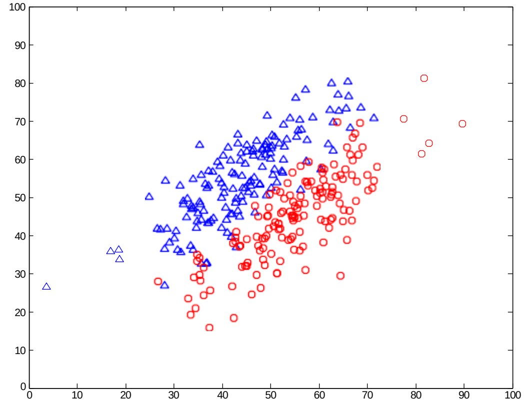
1. Mathematically: there are several *separation criteria*'s.
2. "Intuitively": overlapping of the clusters.



Separable clusters



Almost separable clusters



Non-separable clusters

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

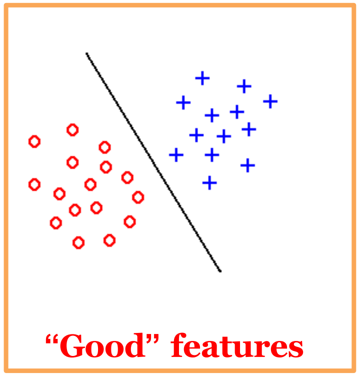
- Multiple Features
  - Does adding more features always improve the results?
    - No!! So we must:
      - Avoid unreliable features.
      - Be careful about correlations with existing features.
      - Be careful about measurement costs.
      - Be careful about noise in the measurements.
  - **Is there some curse for workingin very high dimensions?**
    - YES THERE IS! ==> **CURSE OF DIMENSIONALITY**

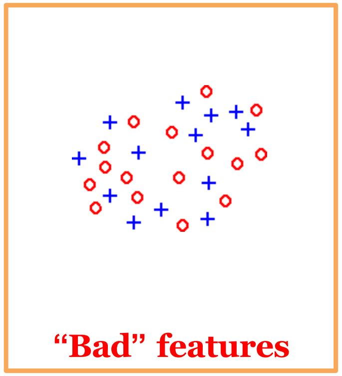
➡ **thumb rule: $n >= d(d-1)/2$**

n = nr of examples in training dataset d = nr of features

Feature Extraction

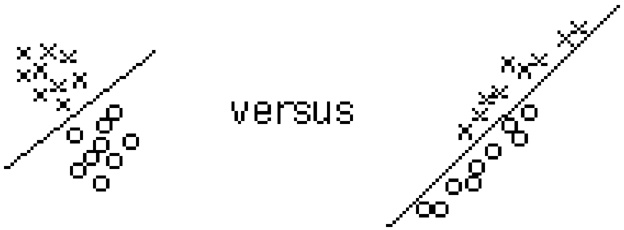
- Problem: Inadequate Features
  - features simply do not contain the information needed to separate the classes, it doesn't matter how much effort you put into designing the classifier.
  - **Solution:** go back and design better features.





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!



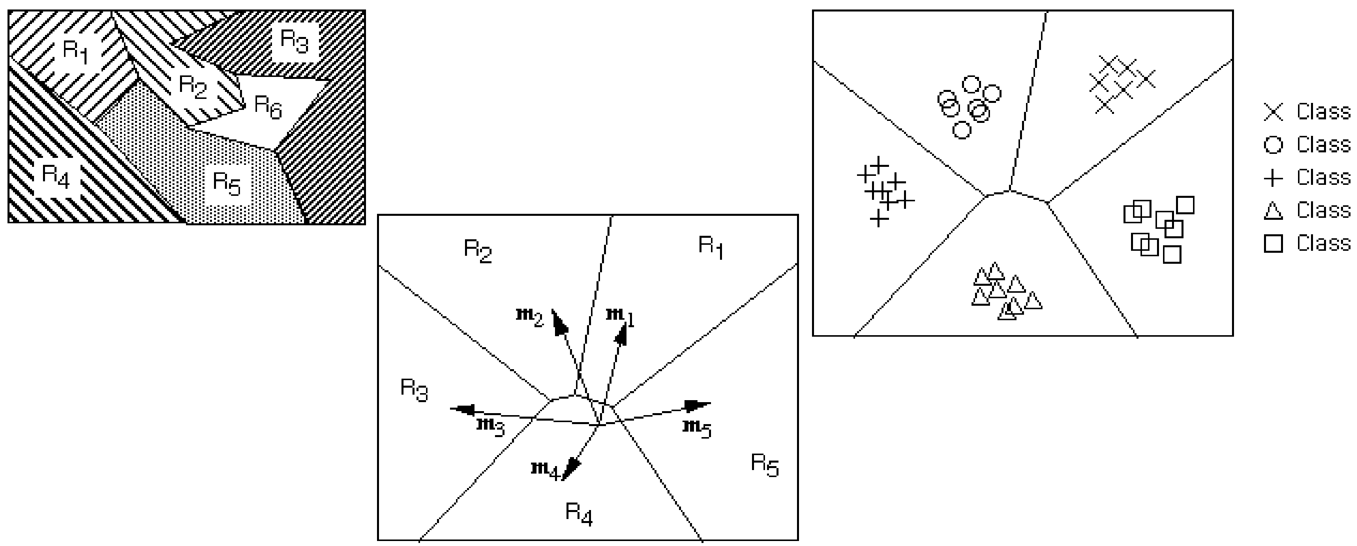
Designing a Classifier

- 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?

Designing a Classifier

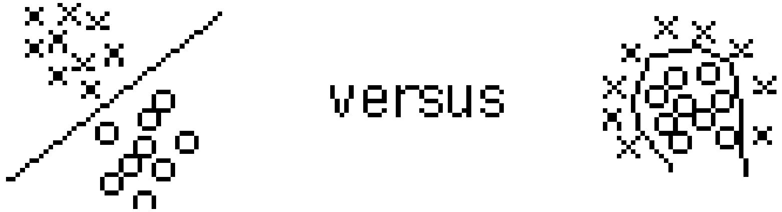
- Designing a **Classifier**
  - How can we manage the tradeoff between complexity of decision rules and their performance to unknown samples?

Different criteria lead to different decision boundaries



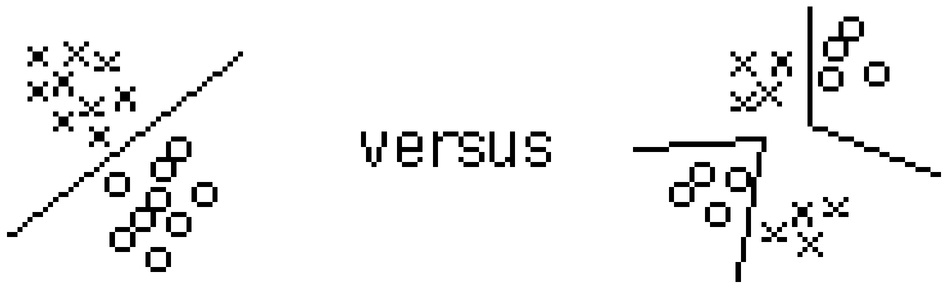
Designing a Classifier

- 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*



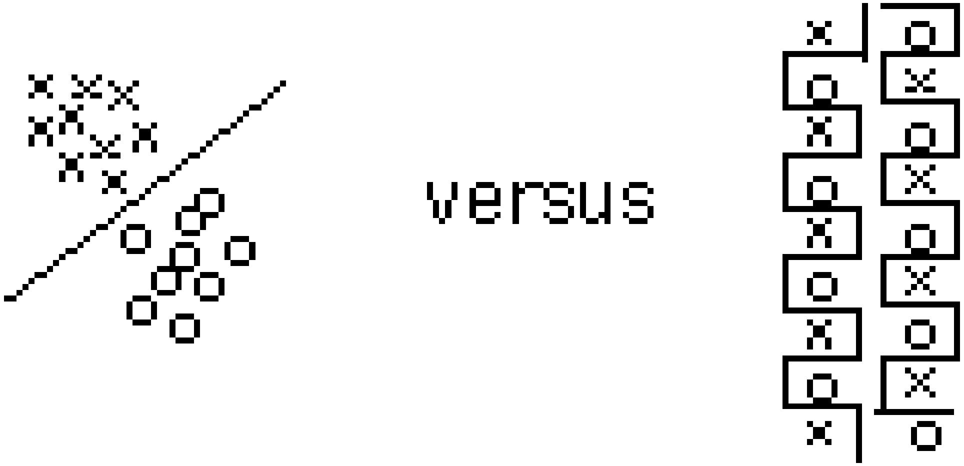
Designing a Classifier

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



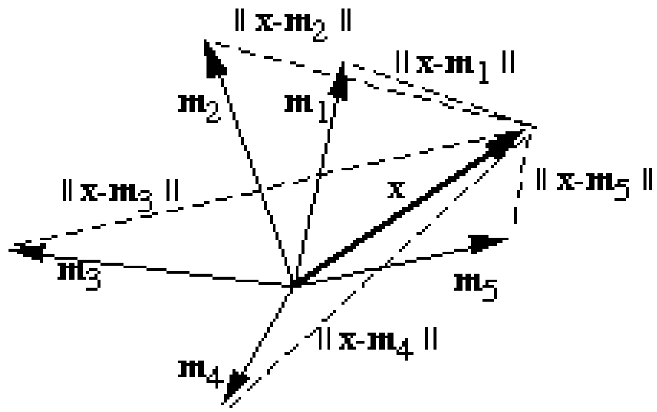
Designing a Classifier

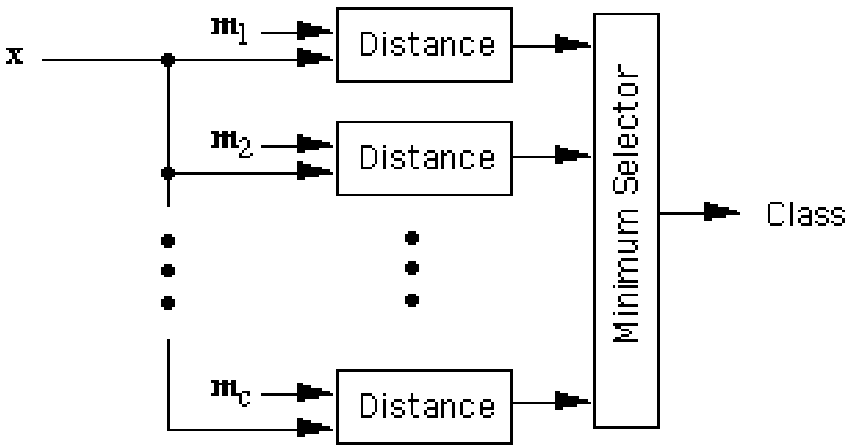
- Problem: Complex Feature Space
  - **Solution:** use different type of Classifier...



Simple Classifiers

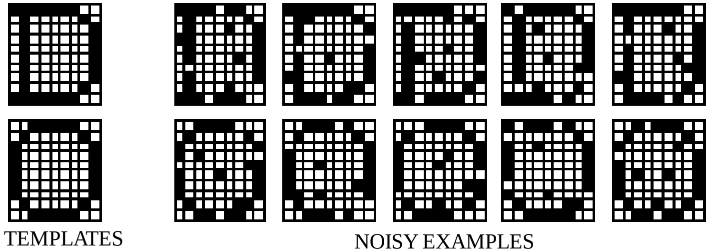
- Minimum-distance Classifiers
  - based on some specified **“metric”** **||x-m||**
  - e.g.Template Matching





Simple Classifiers

- Template Matching

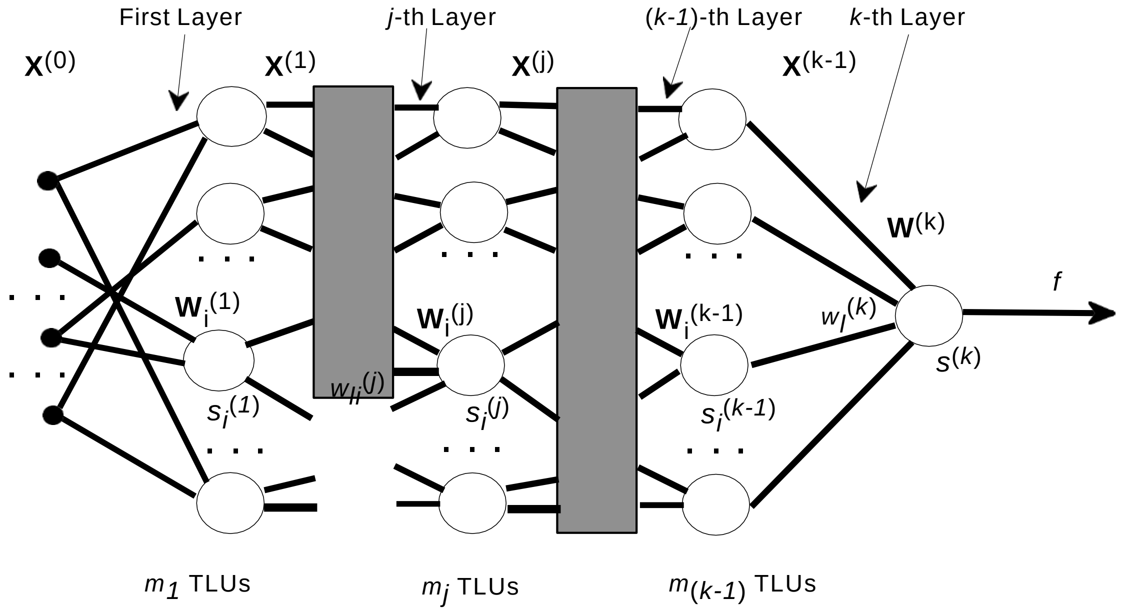


  - To classify one of the noisy examples, simply compare it to the two templates. This can be done in a couple of equivalent ways:
    1. Count the number of agreements. Pick the class that has the maximum number of agreements. **This is a maximum correlation approach.**
    2. Count the number of disagreements. 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

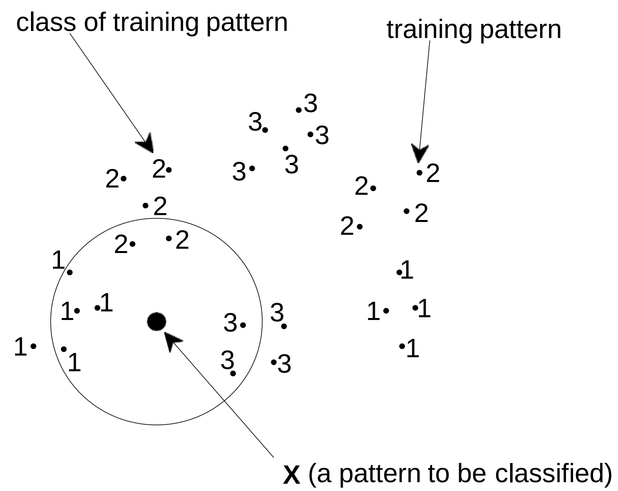
- Metrics
  - different ways of measuring distance:
    - **Euclidean metric:**
      - $||u|| = sqrt(u12 + u22 + \ldots + ud2)$
- **Manhattan (or taxicab) metric:**
      - $|| u || = |u1| + |u2| + \ldots + |ud|$
    - Contours of constant...
      - ...Euclidean distance are circles (or spheres)
 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

Classifiers: Neural Networks



Classifiers: kNN

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



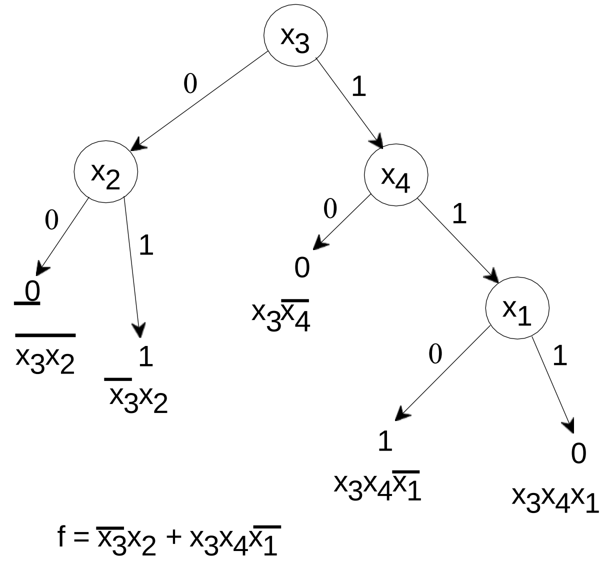
k = 8

four patterns of category 1
two patterns of category 2
two patterns of category 3

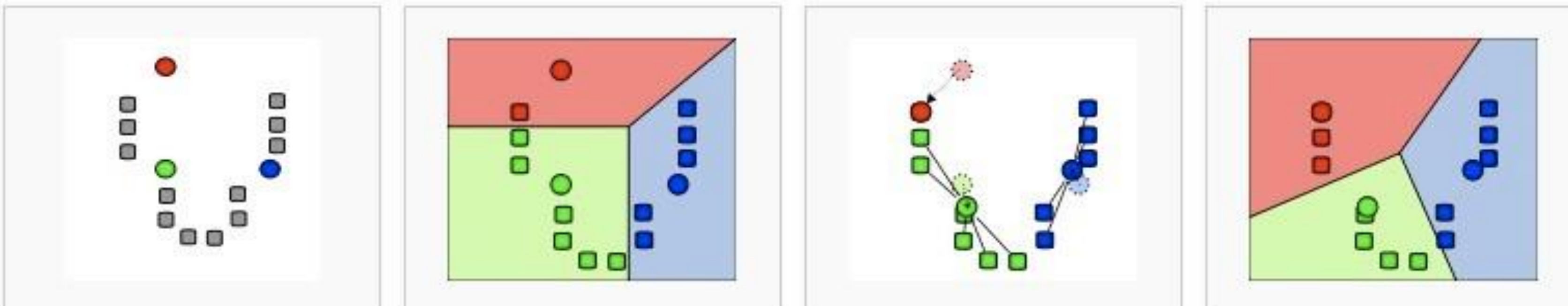
plurality are in category 1,
so decide **X** is in category 1

Decision Trees

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



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.

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.

Pattern Recognition Applications

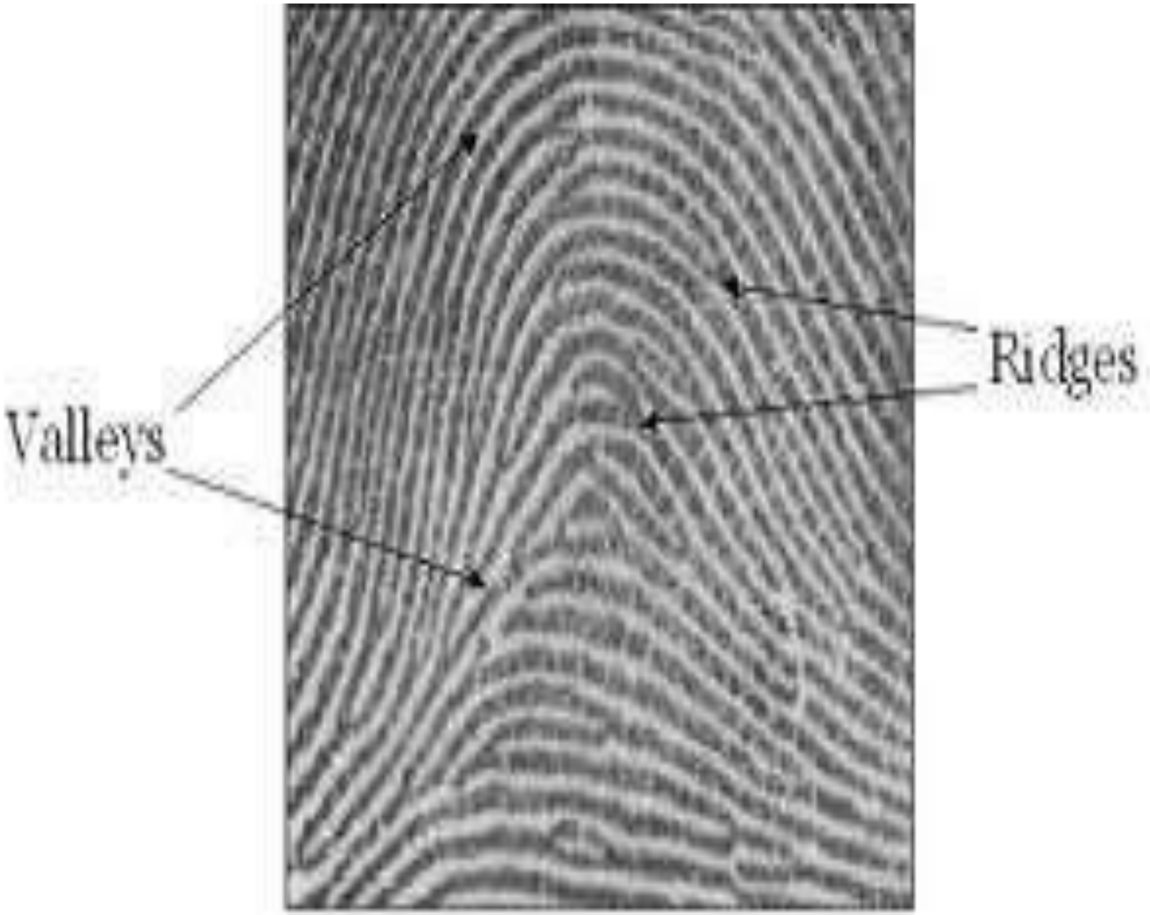
Problem Domain	Application	Input Pattern	Pattern Classes
Document image analysis	Optical character recognition	Document image	Characters, words
Document classification	Internet search	Text document	Semantic categories
Document classification	Junk mail filtering	Email	Junk/non-junk
Multimedia database retrieval	Internet search	Video clip	Video genres
Speech recognition	Telephone directory assistance	Speech waveform	Spoken words
Natural language processing	Information extraction	Sentences	Parts of speech
Biometric recognition	Personal identification	Face, iris, fingerprint	Authorized users for access control
Medical	Computer aided diagnosis	Microscopic image	Cancerous/healthy cell
Military	Automatic target recognition	Optical or infrared image	Target type
Industrial automation	Printed circuit board inspection	Intensity or range image	Defective/non-defective product
Industrial automation	Fruit sorting	Images taken on a conveyor belt	Grade of quality
Remote sensing	Forecasting crop yield	Multispectral image	Land use categories
Bioinformatics	Sequence analysis	DNA sequence	Known types of genes
Data mining	Searching for meaningful patterns	Points in multidimensional space	Compact and well-separated clusters

Fingerprint Recognition

Fingerprint

- The popular Biometric used to authenticate person is Fingerprint which is unique and permanent throughout a person's life
- Fingerprint is the pattern of ridges and valleys
- The ridges have characteristics, called minutiae, are the ridge ending and the ridge bifurcation
- Ridge ending is defined as the point where ridge ends abruptly
- Ridge bifurcation is defined as the point where a ridge forks into branch ridges

Valleys and Ridges



Ridge Ending and Bifurcation

Ridge ending

Bifurcation

Short ridge

Fingerprint Recognition

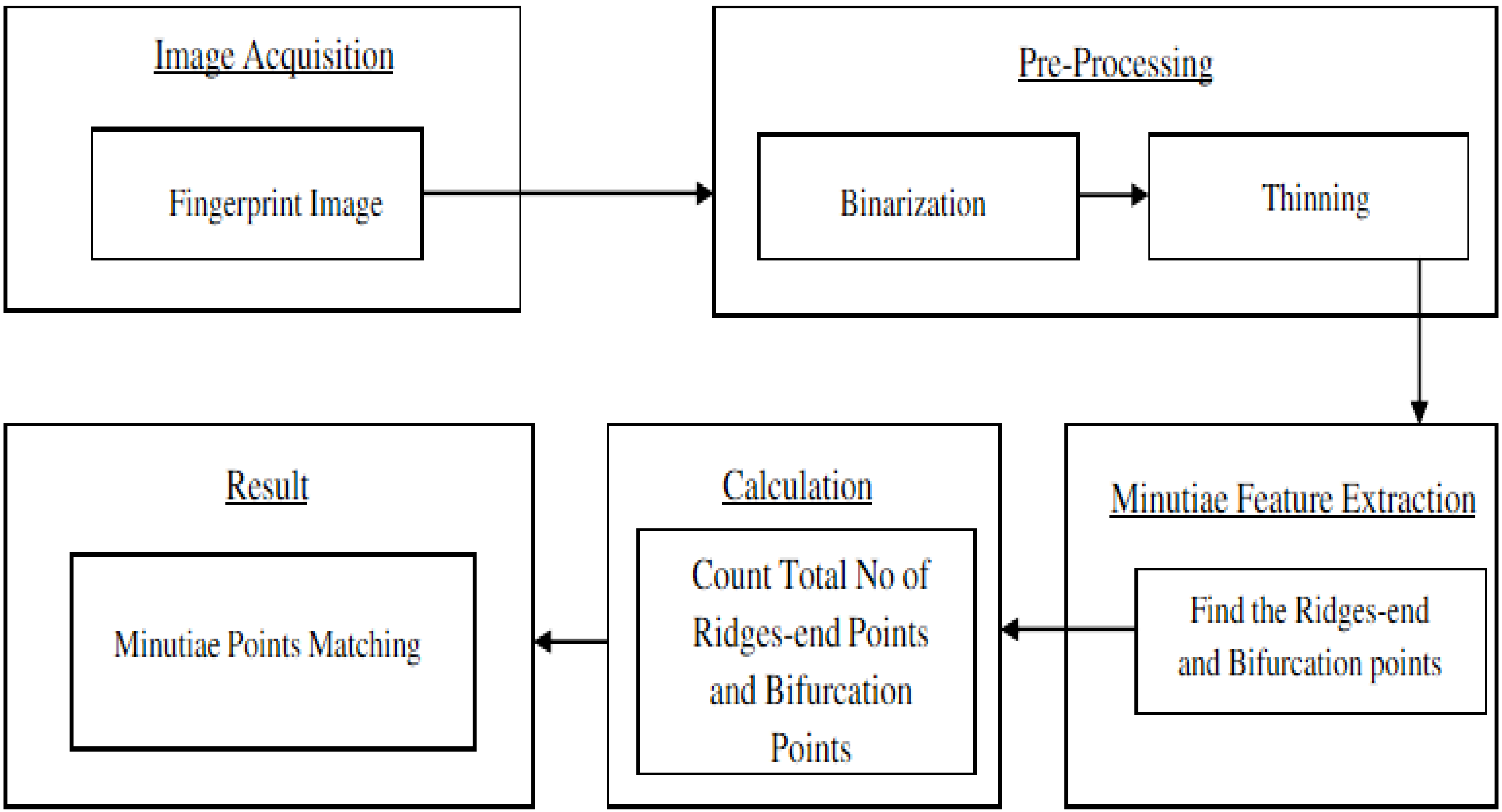
Fingerprint recognition or fingerprint authentication refers to the method of verifying a match between two human fingerprint

Fingerprint recognition techniques have the advantage to use low- cost standard capturing device

However , recognition of the fingerprint becomes a complex computer vision problem , especially when dealing with noisy and low quality images

A minutia matching is widely used for fingerprint recognition and can be classified as ridge ending and ridge bifurcation

Fingerprint Matching using Ridge-End and Bifurcation Points



Fingerprint Image

- The input fingerprint image is the gray scale image of a person, which has intensity values ranging from 0 to 255
- A number of methods are used to acquire fingerprints
- The inked impression method remains the most popular one
- Inkless fingerprint scanners are also present

Inked method

Inkless method

Binarization

Binarization is used to convert gray scale image into binary image by fixing the threshold value

The pixel values above the threshold are set to ‘1’ and the pixel values below the threshold are set to ‘0’ respectively

Original Fingerprint

Binarized Fingerprint

Thinning

The binarized image is thinned using Block Filter

To reduce the thickness of all ridge lines to a single pixel width to extract minutiae points effectively

Thinning does not change the location of minutiae points compared to original fingerprint

Binarized Fingerprint **Image after Thining**

Minutiae Extraction

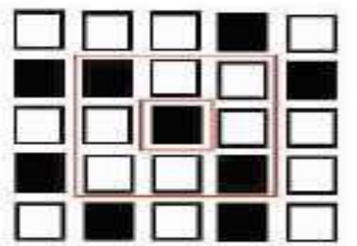
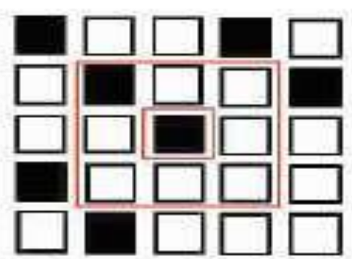
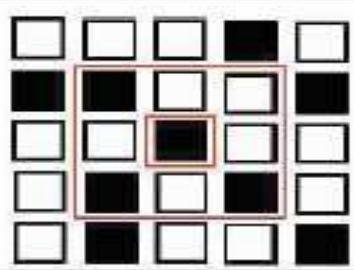
Classification of ridge-end and ridge bifurcation points is done by creating matrix

Crossing Number is used to locate the minutiae points in fingerprint image

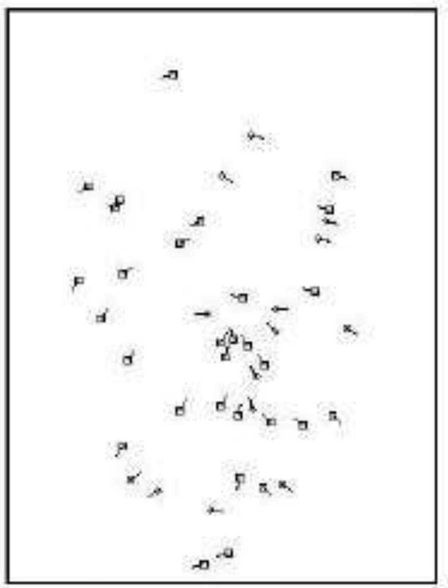
Crossing Number is defined as half of the sum of differences between intensity values of two adjacent pixels

- If crossing Number is 1 → minutiae points are classified as Termination
- If crossing Number is 2 → minutiae points are classified as Normal ridge
- If crossing Number 3 or greater than 3 → minutiae points are classified as Bifurcation

Crossing Number and Type of Minutiae

	Crossing Number =2. Normal ridge pixel.
	Crossing Number =1. Termination point.
	Crossing Number =3. Bifurcation point.

Gray-scale Fingerprint



Minutiae points

Minutiae Matching

- Image Acquisition
- Computation of Points
- Location Detection of Points
- Amount and Location Matching

Image Acquisition

Iimage.jpg = Input Image acquisition from reader.

Timage.jpg = Template Image retrieve from database.

Computation of Points

After the detection of minutiae points, matching algorithm require to calculate total number of available points in the fingerprint image separately

To perform this computation two counter variables are used to count both ridge-end and bifurcation points

Minutiae Point Calculation

Sr. No	Images	Ridges Points	Bifurcation Points
1.	Iimage.jpg	545	2858
2.	Timage.jpg	161	860

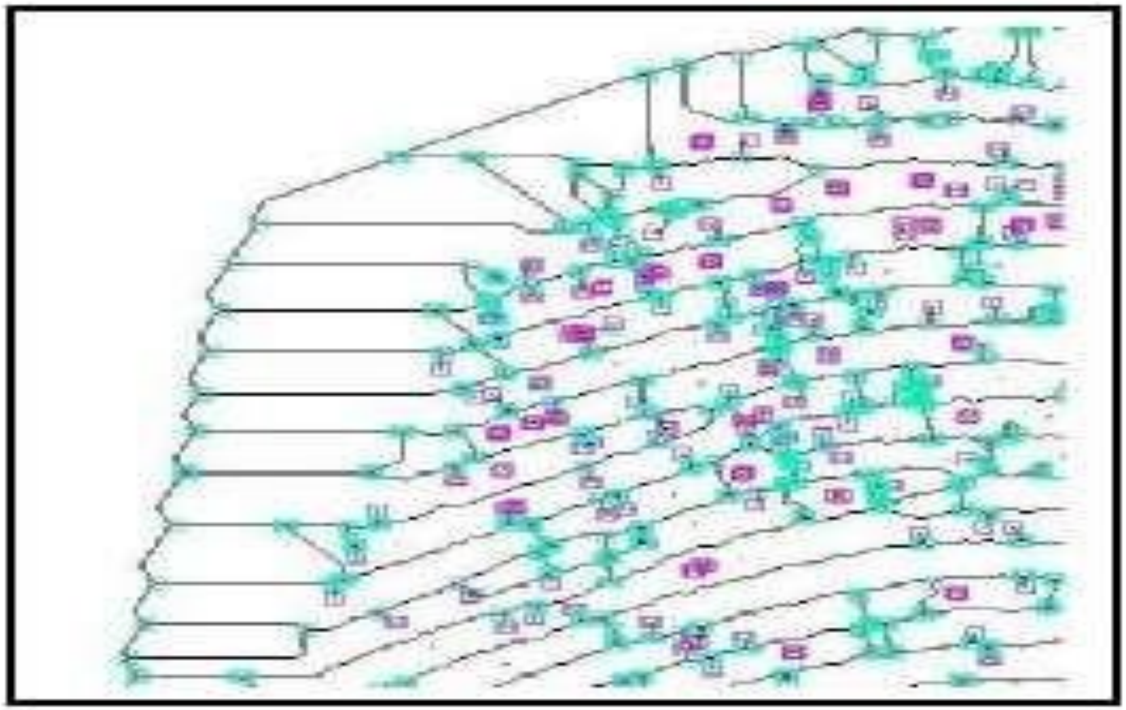
Location Detection of Points

Each minutiae point in the fingerprint image has a specific location.

This location information of particular point is significant to store for further matching of fingerprints.

The location of every point in the digital image is given by pixel position, so that it can be taken and stored separately for both ridge-end and bifurcation points.

Minutiae Point Extracted in Input Image



Amount and Location Matching

In the previous steps, all the required information about points is computed and stored

Now, this is the matching step, here the algorithm compares the computed values with the stored values

This algorithm first, compares the combination of both amounts of ridge-end and bifurcation points with stored data

If the match occurs, the algorithm then compares the location of ridge points with stored location data