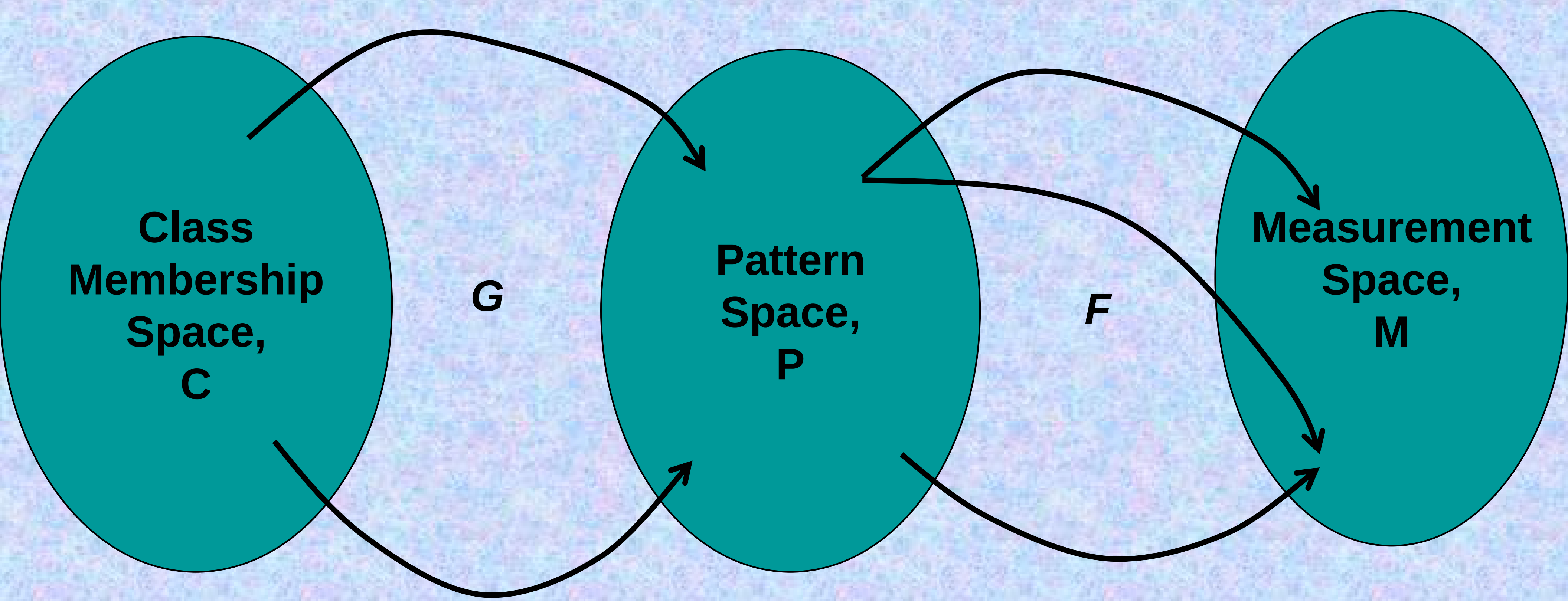
Pattern Recognition

Pattern Recognition is a branch of science that concerns the description or classification (or identification) of measurements. It is an important component of intelligent systems and are used for both data processing and decision making.



Statistical Features

The features used in pattern recognition and segmentation are generally geometric or intensity gradient based.

One approach is to work directly with regions of pixels in the image, and to describe them by various statistical measures. Such measures are usually represented by a single value. These can be calculated as a simple by-product of the segmentation procedures previously described.

Such *statistical descriptions* may be divided into two distinct classes. Examples of each class are given below:

- *Geometric descriptions*: area, length, perimeter, elongation, average radius, compactness and moment of inertia.
- *Topological descriptions*: connectivity and Euler number.

Elongation

- sometimes called *eccentricity*. This is the ratio of the maximum length of line or *chord* that spans the region to the minimum length chord. We can also define this in terms of moments, as we will see shortly.

Compactness

- this is the ratio of the square of the perimeter to the area of the region

Connectivity -

- the number of neighboring features adjoining the region.

Euler Number

- for a single region, one minus the number of holes in that region. The Euler number for a set of connected regions can be calculated as the number of regions minus the number of holes.

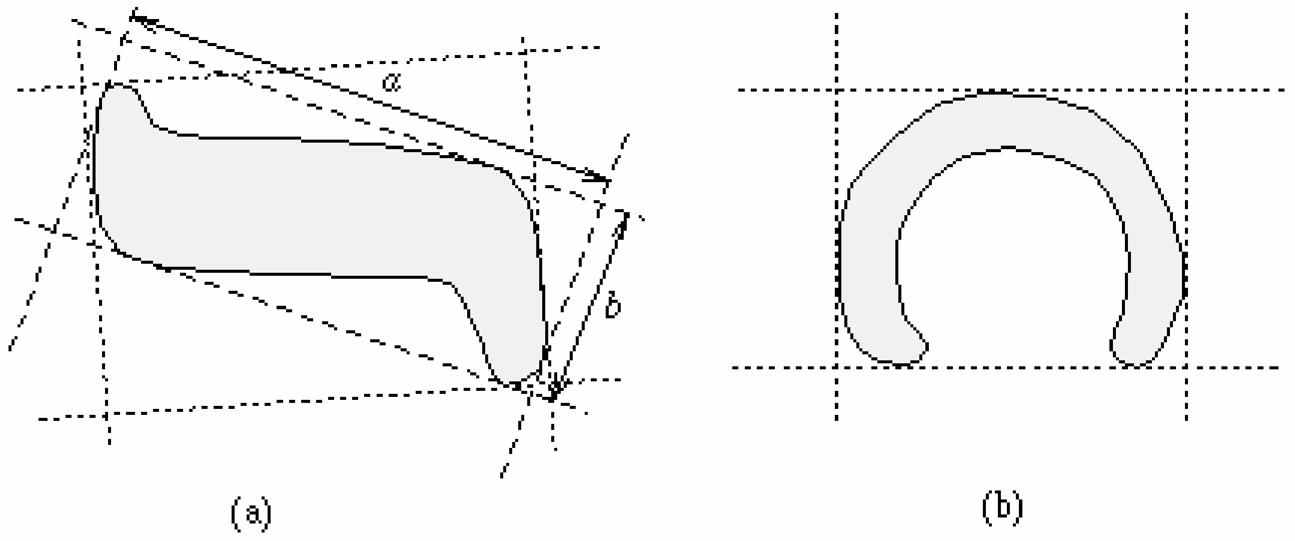


Figure 6.24 *Elongatedness: (a) Bounding rectangle gives acceptable results, (b) bounding rectangle cannot represent elongatedness.*

Elongatedness:

A ratio between the length and width of the region bounding rectangle = a/b = Area/sqr(thickness).

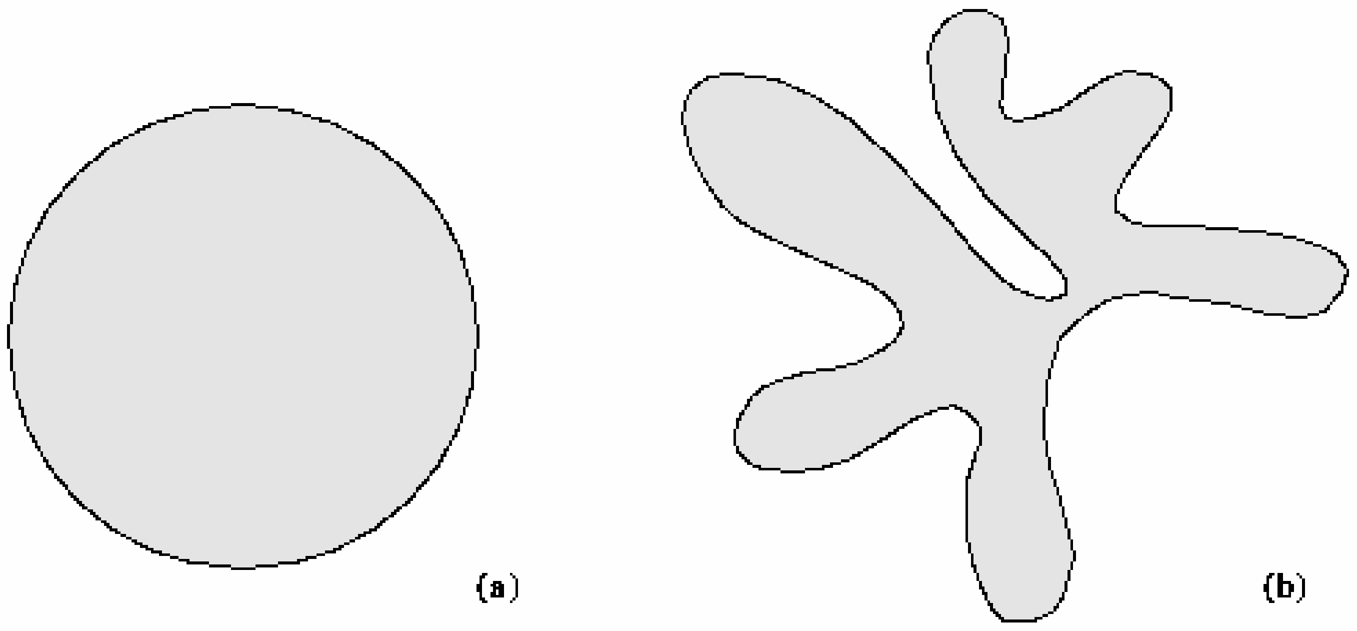


Figure 6.25 *Compactness: (a) Compact, (b) non-compact.*

Compactness

Compactness is independent of linear transformations

= sqr(perimeter)/Area

Moments of Inertia

The *ij*-th discrete central moment m_{ij}, of a region is defined by:

$$m_{ij} = \sum (x - \tilde{x})^i (y - \tilde{y})^j$$

where the sums are taken over all points (*x*, *y*) contained within the region and ($x^{\sim}$, $y^{\sim}$) are the center of gravity of the region:

$$\tilde{x} = \frac{1}{n}\sum_i x_i \quad and \quad \tilde{y} = \frac{1}{n}\sum_i y_i$$

Note that, *n*, the total number of points contained in the region, is a measure of its area.

We can form seven new moments from the central moments that are invariant to changes of position, scale and orientation (RTS) of the object represented by the region, although these new moments are *not* invariant under perspective projection. For moments of order up to seven, these are:

$$
\begin{aligned}
M_1 &= m_{20} + m_{02} \\
M_2 &= (m_{20} - m_{02})^2 + 4m_{11}^2 \\
M_3 &= (m_{30} - 3m_{12})^2 + (3m_{21} - m_{03})^2 \\
M_4 &= (m_{30} + m_{12})^2 + (m_{21} + m_{03})^2 \\
M_5 &= (m_{30} - 3m_{12})(m_{30} + m_{12})\left[(m_{30} + m_{12})^2 - 3(m_{21} + m_{03})^2\right] \\
&\quad + (3m_{21} - m_{03})(m_{21} + m_{03})\left[3(m_{30} + m_{12})^2 - (m_{21} + m_{03})^2\right] \\
M_6 &= (m_{20} + m_{02})\left[(m_{30} + m_{12})^2 - 3(m_{21} + m_{03})^2\right] \\
&\quad + 4m_{11}(m_{30} + m_{12})(m_{03} + m_{21}) \\
M_7 &= (3m_{21} - m_{03})(m_{12} + m_{30})\left[(m_{30} + m_{12}^2 - 3(m_{21} + m_{03})^2\right] \\
&\quad -(m_{30} - 3m_{12})(m_{12} + m_{03})\left[3(m_{30} + m_{12})^2 - (m_{21} + m_{03})^2\right]
\end{aligned}
$$

We can also define *eccentricity*, using moments as

$$
\text{eccentricity} = \frac{m_{20} + m_{02} + \sqrt{(m_{20} - m_{02})^2 + 4m_{11}^2}}{m_{20} + m_{02} - \sqrt{(m_{20} - m_{02})^2 + 4m_{11}^2}}.
$$

We can also find *principal axes of inertia* that define a natural coordinate system for a region. It is given by:

$$
\theta = \frac{1}{2}\tan^{-1}\left[\frac{2m_{11}}{m_{20} - m_{02}}\right]
$$

Geometric properties in terms of moments:

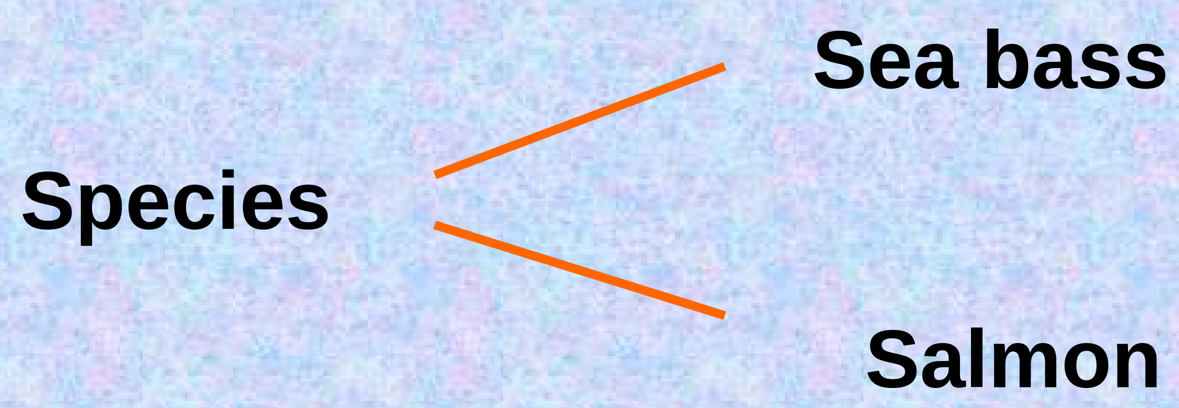
$$Area = m_{00};\ \tilde{x} = \frac{m_{10}}{m_{00}};\ \tilde{y} = \frac{m_{01}}{m_{00}}$$

Some Terminologies:

- Pattern
- Feature
- Feature vector
- Feature space
- Classification
- Decision Boundary
- Decision Region
- Discriminant function
- Hyperplanes and Hypersurfaces
- Learning
- Supervised and unsupervised
- Error
- Noise
- PDF
- Baye’s Rule
- Parametric and Non-parametric approaches

An Example

- “Sorting incoming Fish on a conveyor according to species using optical sensing



- **Some properties that could be possibly used to distinguish between the two types of fishes is**

  - **Length**
  - **Lightness**
  - **Width**
  - **Number and shape of fins**
  - **Position of the mouth, etc…**

  Features

- **This is the set of all suggested features to explore for use in our classifier!**

Feature is a property of an object (quantifiable or non quantifiable) which is used to distinguish between or classify two objects.

Feature vector

- A Single feature may not be useful always for classification

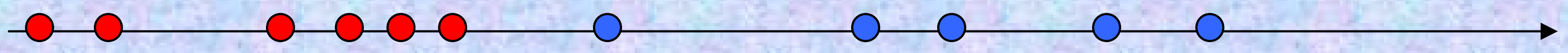
- A set of features used for classification form a feature vector

$$\begin{tikzcd}
\text{Fish} \arrow[rr] & & x^T = [x_1, x_2] \arrow[d] \arrow[dr] & \\
& & \text{Lightness} & \text{Width}
\end{tikzcd}$$

Feature space

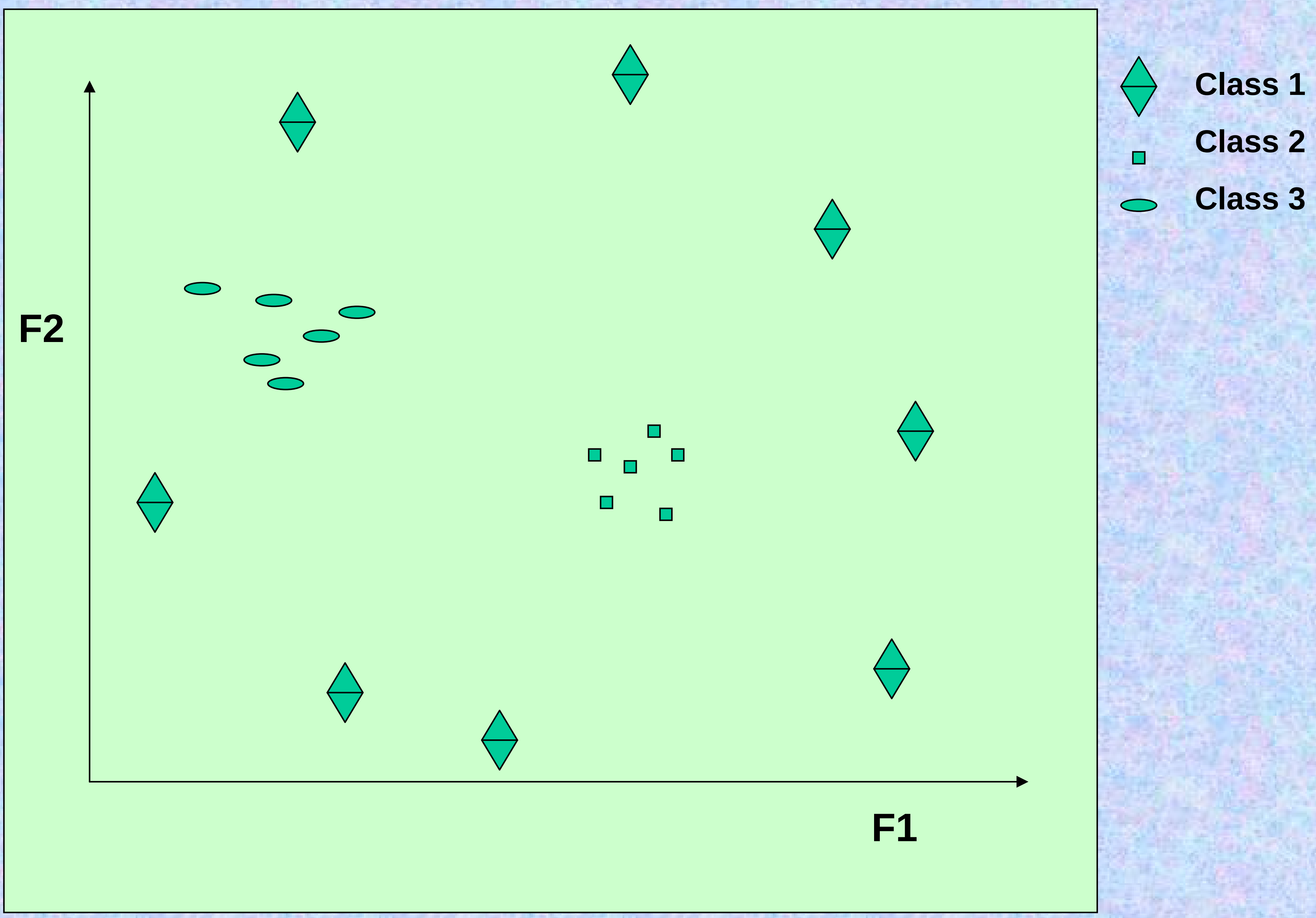
- The samples of input (when represented by their features) are represented as points in the feature space

- If a single feature is used, then work on a one- dimensional feature space.



Point representing samples

- If number of features is 2, then we get points in 2D space as shown in next page.
- We can also have an n-dimensional feature space



Sample points in a two-dimensional feature space

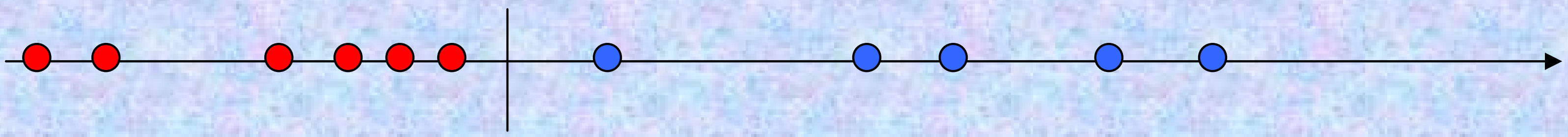
Decision region and Decision Boundary

- Our goal of pattern recognition is to reach an optimal decision rule to categorize the incoming data into their respective categories

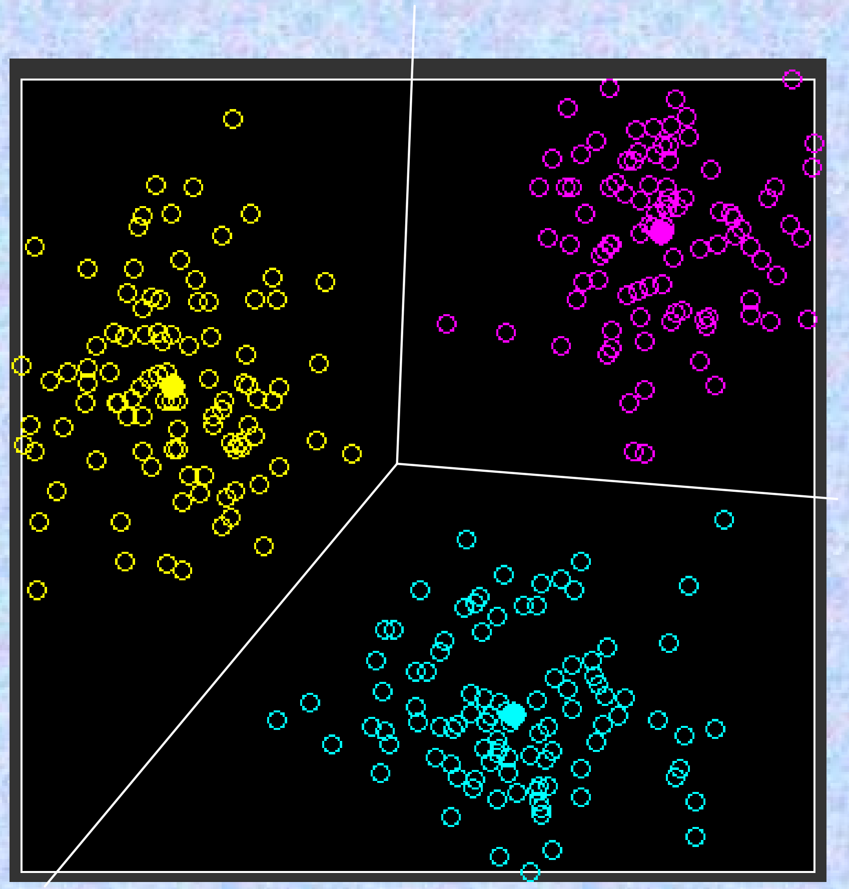
- The decision boundary separates points belonging to one class from points of other

- The decision boundary partitions the feature space into decision regions.

- The nature of the decision boundary is decided by the discriminant function which is used for decision. It is a function of the feature vector.



Decision boundary in one-dimensional case with two classes.



Decision boundary in two dimensional case with three classes

Hyper planes and Hyper surfaces

- **For two category case, a positive value of discriminant function decides class 1 and a negative value decides the other.**
- **If the number of dimensions is three. Then the decision boundary will be a plane or a 3-D surface. The decision regions become semi-infinite volumes**
- **If the number of dimensions increases to more than three, then the decision boundary becomes a hyper-plane or a hyper-surface. The decision regions become semi-infinite hyperspaces.**

Learning

- The classifier to be designed is built using input samples which is a mixture of all the classes.

- The classifier learns how to discriminate between samples of different classes.

- If the Learning is offline i.e. Supervised method then, the classifier is first given a set of training samples and the optimal decision boundary found, and then the classification is done.

- If the learning is online then there is no teacher and no training samples (Unsupervised). The input samples are the test samples itself. The classifier learns and classifies at the same time.

Error

- The accuracy of classification depends on two things
  - The optimality of decision rule used: The central task is to find an optimal decision rules which can generalize to unseen samples as well as categorize the training samples as correctly as possible. This decision theory leads to a minimum error-rate classification.
  - The accuracy in measurements of feature vectors: This inaccuracy is because of presence of noise. Hence our classifier should deal with noisy and missing features too.

Some necessary elements of

Probability theory and Statistics

Normal Density:

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

Bivariate Normal Density:

$$p(x,y) = \frac{e^{-\frac{1}{2(1-\rho_{xy}^2)}[(\frac{x-\mu_x}{\sigma_x})^2 - \frac{2\rho_{xy}(x-\mu_x)(y-\mu_y)}{\sigma_x\sigma_y} + (\frac{y-\mu_y}{\sigma_y})^2]}}{2\pi\sigma_x\sigma_y\sqrt{(1-\rho_{xy}^2)}}$$

μ - Mean; σ - S.D.; ρ_{xy} - Correlation Coefficient

Visualize ρ as equivalent to the orientation of the 2-D Gabor filter.

For x as a discrete random variable, the expected value of x:

$$E(x) = \sum_{i=1}^{n} x_i P(x_i) = \mu_x$$

E(x) is also called the first moment of the distribution.
The kth moment is defined as:

$$E(x^k) = \sum_{i=1}^{n} x_i^k P(x_i)$$

$P(x_i)$ is the probability of $x = x_i$.

Second, third,… moments of the distribution p(x) ae the expected values of: x^2, x^3,…

The kth central moment is defined as:

$$E[(x-\mu_x)^k] = \sum_{i=1}^{n} (x-\mu_x)^k P(x_i)$$

Thus, the second central moment (also called Variance) of a random variable x is defined as:

$$\sigma_x^2 = E[\{x - E(x)\}^2] = E[(x-\mu_x)^2]$$

S.D. of x is σ_x.

$$\sigma_x^2 = E[\{x - E(x)\}^2] = E[(x-\mu_x)^2]$$

$$= E(x^2) - 2\mu_x^2 + \mu_x^2 = E(x^2) - \mu_x^2$$

$$Thus$$

$$E(x^2) = \sigma^2 + \mu^2$$

If z is a new variable: z= ax + by; Then E(z) = E(ax + by)=aE(x) + bE(y).

Covariance of x and y, is defined as: $\sigma_{xy} = E[(x-\mu_x)(y-\mu_y)]$

Covariance indicates how much x and y vary together. The value depends on how much each variable tends to deviate from its mean, and also depends on the degree of association between x and y.

Correlation between x and y: $\rho_{xy} = \dfrac{\sigma_{xy}}{\sigma_x \sigma_y} = E[(\dfrac{x-\mu_x}{\sigma_x})(\dfrac{y-\mu_y}{\sigma_y})]$

Property of correlation coefficient: $-1 \le \rho_{xy} \le 1$

For Z:

$$E[(z-\mu_z)^2] = a^2\sigma_x^2 + 2ab\sigma_{xy} + b^2\sigma_y^2;$$

$$If \quad \sigma_{xy} = 0, \ \ \sigma_x^2 = a^2\sigma_x^2 + b^2\sigma_y^2$$

Multi-variate Case: $X = [x_1 \ x_2 \ \ldots\ldots \ x_d]^T$

Mean vector:

$$\boldsymbol{\mu} = E(X) = \begin{bmatrix} \mu_1 \\ \mu_2 \\ . \\ . \\ \mu_d \end{bmatrix}$$

Covariance matrix (symmetric):

$$\Sigma = \begin{bmatrix} \sigma_{11} & \sigma_{12} & . & . & \sigma_{1d} \\ \sigma_{21} & \sigma_{22} & . & . & \sigma_{2d} \\ . & . & . & . & . \\ . & . & . & . & . \\ \sigma_{d1} & \sigma_{d2} & . & . & \sigma_{dd} \end{bmatrix} = \begin{bmatrix} \sigma_1^2 & \sigma_{12} & . & . & \sigma_{1d} \\ \sigma_{12} & \sigma_2^2 & . & . & \sigma_{2d} \\ . & . & . & . & . \\ . & . & . & . & . \\ \sigma_{1d} & \sigma_{2d} & . & . & \sigma_d^2 \end{bmatrix}$$

d-dimensional normal density is:

$$p(X) = \frac{1}{\sqrt{\det(\Sigma)(2\pi)^d}} \exp[-\frac{(X-\boldsymbol{\mu})^T \Sigma^{-1} (X-\boldsymbol{\mu})}{2}]$$

$$= \frac{1}{\sqrt{\det(\Sigma)(2\pi)^d}} \exp[-\frac{1}{2} \sum_{ij} (x_i - \mu_i) s_{ij} (x_j - \mu_j)]$$

$$p(X) = \frac{1}{\sqrt{\det(\Sigma)(2\pi)^d}} \exp[-\frac{(X-\mu)^T \Sigma^{-1}(X-\mu)}{2}]$$

$$= \frac{1}{\sqrt{\det(\Sigma)(2\pi)^d}} \exp[-\frac{1}{2}\sum_{ij}(x_i-\mu_i)s_{ij}(x_j-\mu_j)]$$

where s_{ij} is the ijth component of Σ^{-1} (the inverse of covariance matrix Σ).

Special case, d = 2; where X = $(x\ y)^T$; Then:

$$\mu = \begin{pmatrix} \mu_x \\ \mu_y \end{pmatrix}$$

and

$$\Sigma = \begin{pmatrix} \sigma_x^2 & \sigma_{xy} \\ \sigma_{xy} & \sigma_y^2 \end{pmatrix} = \begin{pmatrix} \sigma_x^2 & \rho_{xy}\sigma_x\sigma_y \\ \rho_{xy}\sigma_x\sigma_y & \sigma_y^2 \end{pmatrix}$$

Can you now obtain this, as given earlier:

$$p(x,y) = \frac{e^{-\frac{1}{2(1-\rho_{xy}^2)}[(\frac{x-\mu_x}{\sigma_x})^2 - \frac{2\rho_{xy}(x-\mu_x)(y-\mu_y)}{\sigma_x\sigma_y} + (\frac{y-\mu_y}{\sigma_y})^2]}}{2\pi\sigma_x\sigma_y\sqrt{(1-\rho_{xy}^2)}}$$

Sample mean is defined as: $\tilde{x} = \frac{1}{n}\sum_{i=1}^{n} x_i P(x_i) = \frac{1}{n}\sum_{i=1}^{n} x_i$ where, $P(x_i) = 1/n$.

Sample Variance is: $\sigma_x^2 = \frac{1}{n}\sum_{i=1}^{n}(x_i - \tilde{x})^2$

Higher order moments may also be computed: $E(x_i - \tilde{x})^3;\ E(x_i - \tilde{x})^4$

Covariance of a bivariate distribution:

$$\sigma_{xy} = E[(x-\mu_x)(y-\mu_y)] = \frac{1}{n}\sum_{i=1}^{n}(x-\tilde{x})(y-\tilde{y})$$

MAXIMUM LIKELIHOOD ESTIMATE

The ML estimate of a parameter is that value which, when substituted into the probability distribution (or density), produces that distribution for which the probability of obtaining the entire observed set of samples is maximized.

Problem: Find the maximum likelihood estimate for μ in a normal distribution.

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

Assuming all random samples to be independent:

$$p(x_1,,,,x_n) = p(x_1).....p(x_n) = \prod_{i=1}^{n} p(x_i)$$

$$= \frac{1}{\sigma^n (2\pi)^{n/2}} \exp[-\frac{1}{2\sigma^2} \sum_{i=1}^{n} (\frac{x-\mu}{\sigma})^2]$$

Taking derivative (w.r.t. μ) of the LOG of the above:

$$\frac{1}{2\sigma^2} \sum_{i=1}^{n} (x_i - \mu).2 = \frac{1}{\sigma^2}[\sum_{i=1}^{n} x_i - n\mu]$$

Setting this term = 0, we get:

$$\mu = \frac{1}{n} \sum_{i=1}^{n} x_i = \tilde{x}$$

Parametric Decision making (Statistical) - Supervised

Goal of most classification procedures is to estimate the probabilities that a pattern to be classified belongs to various possible classes, based on the values of some feature or set of features.

In most cases, we decide which is the most likely class. We need a mathematical decision making algorithm, to obtain classification.

Bayesian decision making or Bayes Theorem

This method refers to choosing the most likely class, given the value of the feature/s. Bayes theorem calculates the probability of class membership.

Define:

$P(w_i)$ - Prior Prob. for class w_i ; $P(X)$ - Prob. for feature vector X

$P(w_i|X)$ - Measured-conditioned or posteriori probability

$P(X | w_i)$ - Prob. Of feature vector X in class w_i

Bayes Theorem:

$$P(w_i \mid \vec{X}) = \frac{P(\vec{X} \mid w_i)P(w_i)}{P(\vec{X})}$$

P(X) is the probability distribution for feature X in the entire population. Also called *unconditional density function*.

$P(w_i)$ is the *prior probability* that a random sample is a member of the class C_i.

$P(X \mid w_i)$ is the *class conditional probability* of obtaining feature value X given that the sample is from class w_i. It is equal to the number of times (occurrences) of X, if it belongs to class w_i.

The goal is to measure: $P(w_i \mid X)$ –
***Measured-conditioned or posteriori probability*, from the above three values.**

This is the Prob. of any vector X being assigned to class w_i.

$$\begin{tikzcd}
 & P(X|w) \arrow[d] & \\
P(w) \arrow[r] & \boxed{\text{BAYES RULE}} \arrow[r] & P(w|X) \\
 & X, P(X) \arrow[u] &
\end{tikzcd}$$

Take an example:

Two class problem: Cold (C) and not-cold (C’). Feature is fever (f).

Prior probability of a person having a cold, P(C) = 0.01.

Prob. of having a fever, given that a person has a cold is,
P(f|C) = 0.4. Overall prob. of fever P(f) = 0.02.

Then using Bayes Th., the Prob. that a person has a cold, given that she (or he) has a fever is:

$$P(C \mid f) = \frac{P(f \mid C)P(C)}{P(f)} = \frac{0.4 * 0.01}{0.02} = 0.2$$

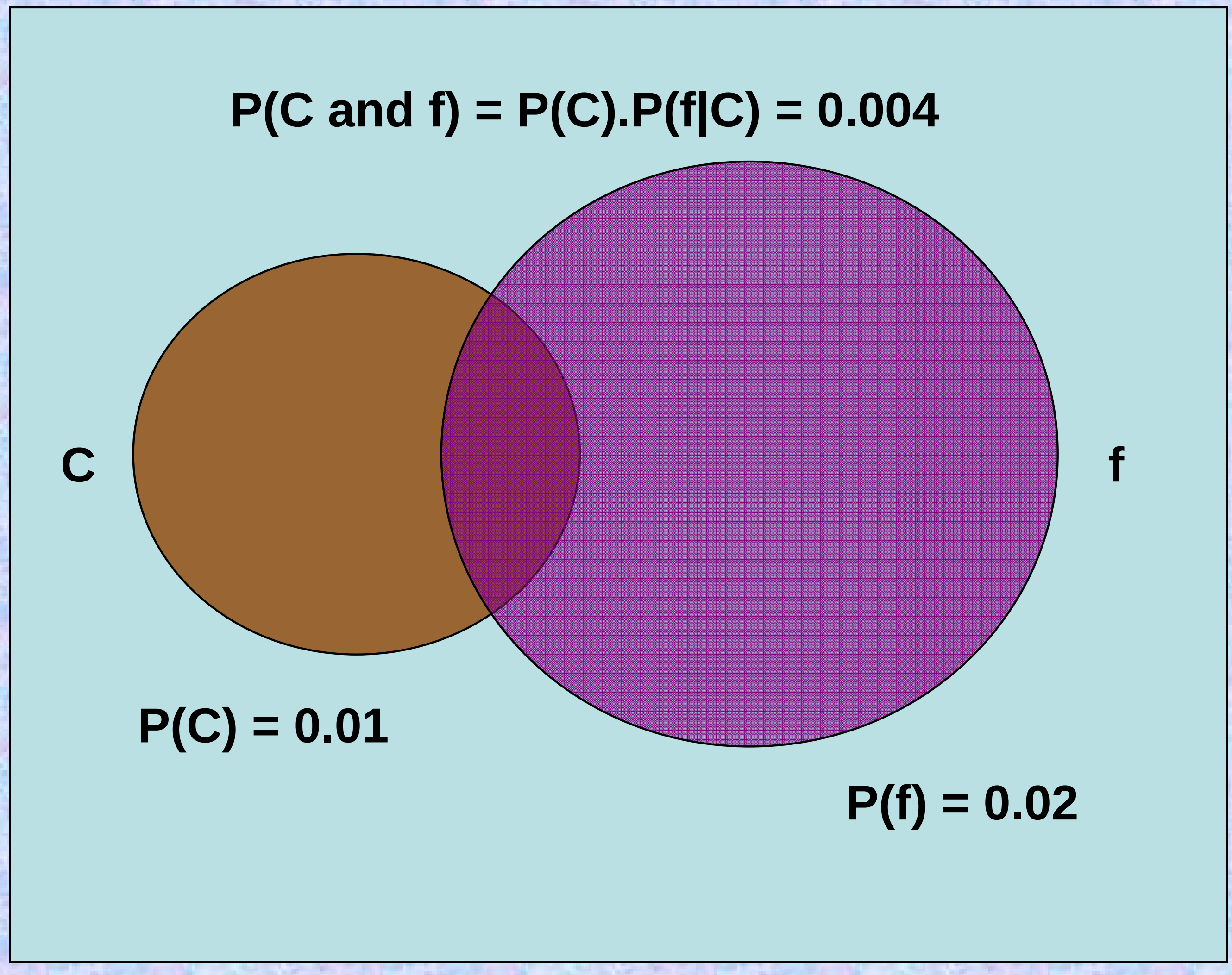
Not convinced that it works?
let us take an example with values to verify:

Total Population =1000. Thus, people having cold = 10. People having both fever and cold = 4. Thus, people having only cold = 10 – 4 = 6.
People having fever (with and without cold) = 0.02 * 1000 = 20.
People having fever without cold = 20 – 4 = 16 (*may use this later*).

So, probability (percentage) of people having cold along with fever, out of all those having fever, is: 4/20 = 0.2 (20%).

IT WORKS, GREAT

A Venn diagram, illustrating the two class, one feature problem.



Probability of a joint event - a sample comes from class C and has the feature value X:

$$P(C \text{ and } X) = P(C).P(X|C) = P(X).P(C|X)$$
$$= 0.01*0.4 = 0.02*0.2$$

Also verify, for a K class problem:

$P(X) = P(w_1)P(X|w_1) + P(w_2)P(X|w_2) + \ldots\ldots + P(w_k)P(X|w_k)$

Thus:

$$P(w_i \mid \overrightarrow{X}) = \frac{P(\overrightarrow{X} \mid w_i)P(w_i)}{P(w_1)P(X \mid w_1) + P(w_2)P(X \mid w_2) + \ldots + P(w_k)P(X \mid w_k)}$$

With our last example:

$P(f) = P(C)P(f|C) + P(C')P(f|C')$

$= 0.01 * 0.4 + 0.99 * 0.01616 = 0.02$

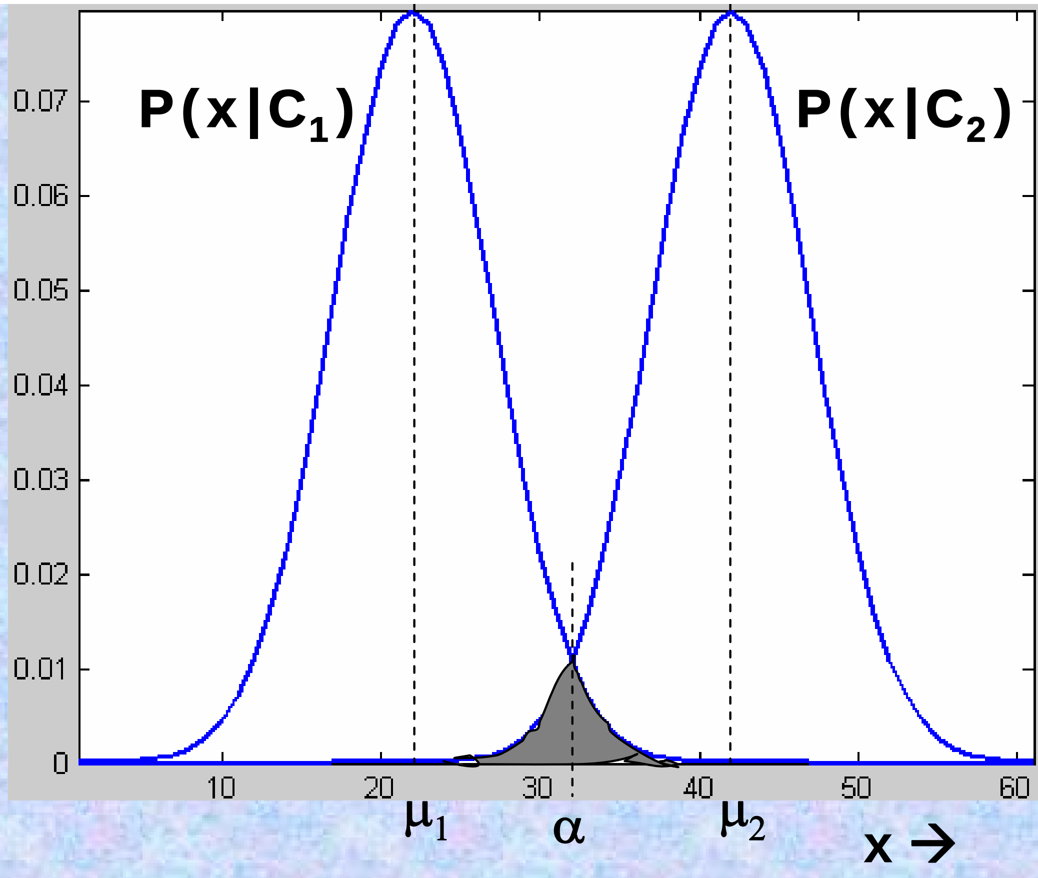
Decision or Classification algorithm according to Baye's Theorem:

$$Choose \quad \begin{cases} w_1; & \text{if } p(X|w_1)p(w_1) > p(X|w_2)p(w_2) \\ w_2; & \text{if } p(X|w_2)p(w_2) > p(X|w_1)p(w_1) \end{cases}$$

Errors in decision making:

Let d = 1, C = 2,
P(C_1) = P(C_2) =

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



Bayes decision rule:

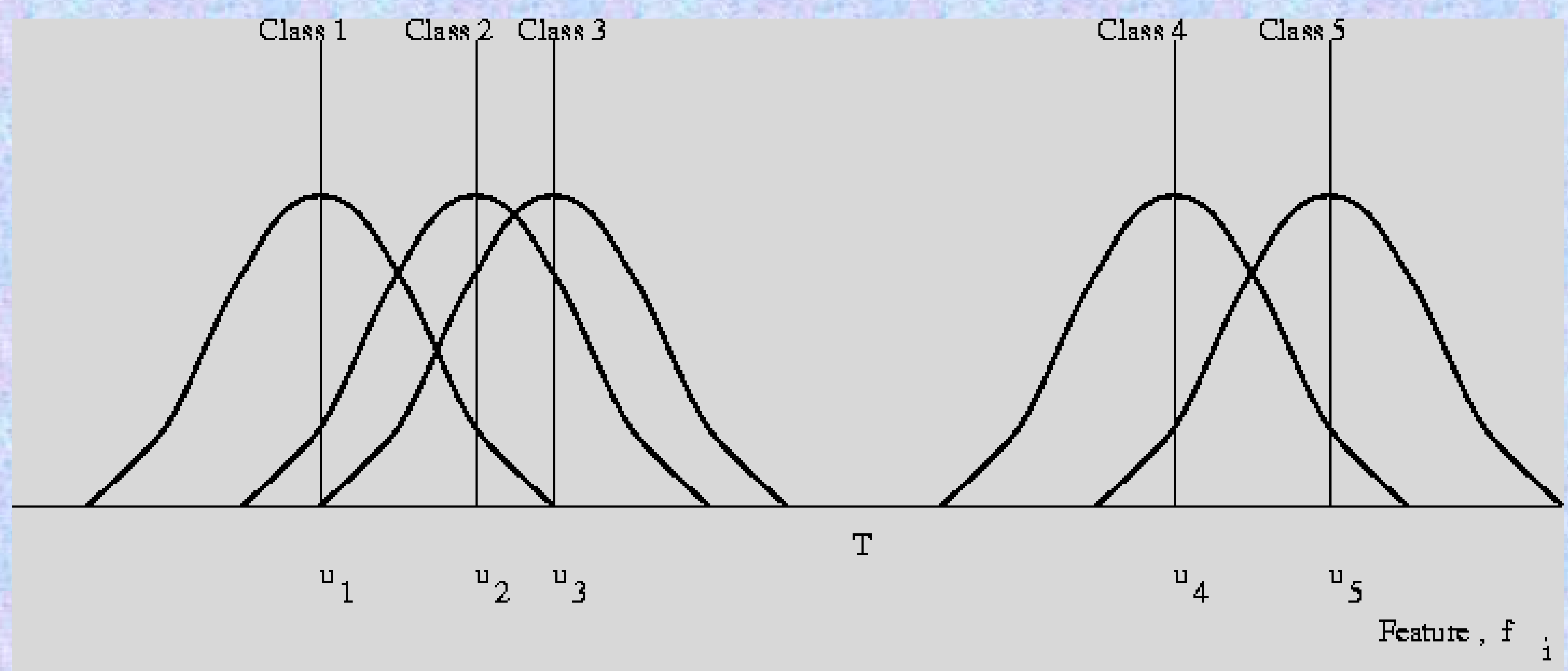
Choose C_1 , if $P(x|C_1) > P(x|C_2)$

This gives α, and hence the two decision regions.

Classification error (the shaded region):

P(E) = P(Chosen C_1, when x belongs to C_2) +
P(Chosen C_2, when x belongs to C_1)

$$= P(C_2)\int_{-\infty}^{\alpha} P(\gamma | C_2)d\gamma + P(C_1)\int_{\alpha}^{-\infty} P(\gamma | C_1)d\gamma$$



Normal distributions of feature measurement for a 5-class problem, equal variance.

A minimum distance classifier

Rule: Assign X to R_i, where X is closest to μ_i.

K-means Clustering

- **Given a fixed number of k clusters, assign observations to those clusters so that the means across clusters for all variables are as different from each other as possible.**
- **Input**
  - **Number of Clusters, k**
  - **Collection of n, d dimensional vectors x_j , j=1, 2, …, n**
- **Goal: find the k mean vectors $\mu_1, \mu_2, \ldots, \mu_k$**
- **Output**
  - **k x n binary membership matrix U where**

$$u_{ij} = \begin{cases} 1 & \text{if } x_i \in G_i \\ 0 & \text{else} \end{cases}$$

& G_j, j=1, 2, …, k represent the k clusters

If n is the number of known patterns and k the desired number of clusters, the k-means algorithm is:

```
Begin
    initialize n, c, $\mu_1, \mu_2, \ldots, \mu_c$(randomly selected)
        do
            1.classify n samples according
              to nearest $\mu_i$
            2.recompute $\mu_i$
        until no change in $\mu_i$
    return $\mu_1, \mu_2, \ldots, \mu_c$
End
```

Classification Stage

- The samples have to be assigned to clusters in order to minimize the cost function which is:

$$J = \sum_{i=1}^{c} J_i = \sum_{i=1}^{c} \left[\sum_{k,\, x_k \in G_i} \left\| x_k - \mu_i \right\|^2 \right]$$

- This is the Euclidian Distance of the samples from its cluster center for all clusters should be minimum
- The classification of a point x_k is done by:

$$u_i = \begin{cases} 1 & \text{if } \left\| x_k - \mu_i \right\|^2 \geq \left\| x_k - \mu_j \right\|^2 \ , \forall\, k \neq i \\ 0 & \text{otherwise} \end{cases}$$

Re-computing the Means

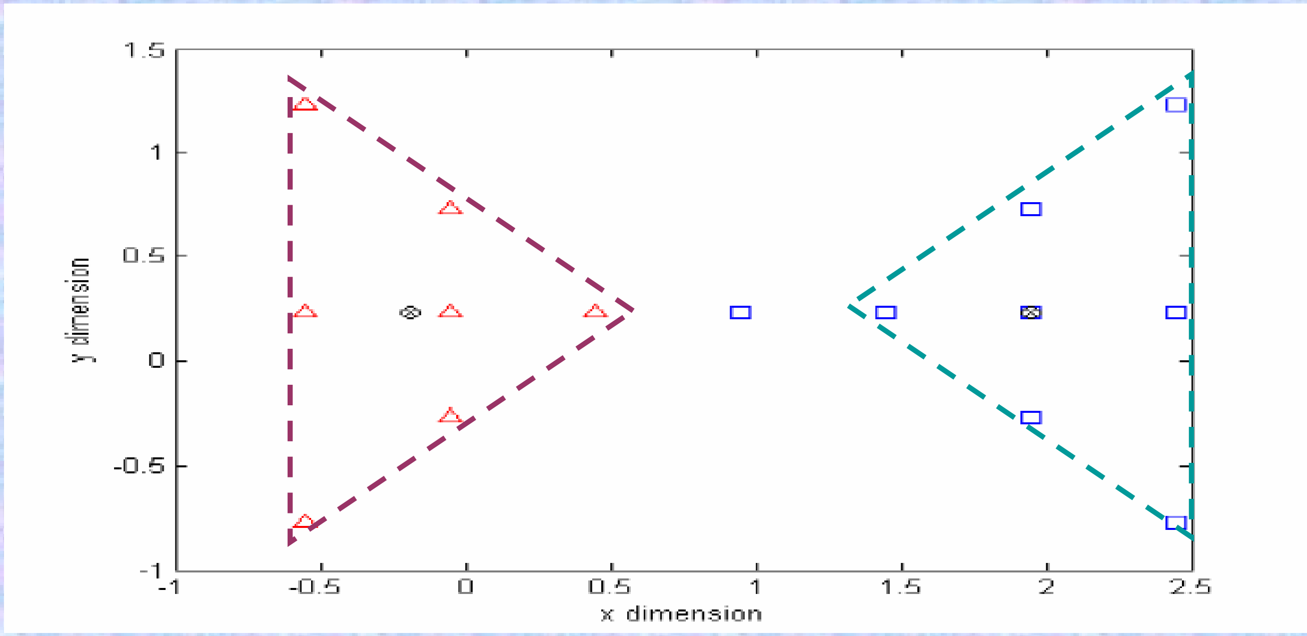
- The means are recomputed according to:

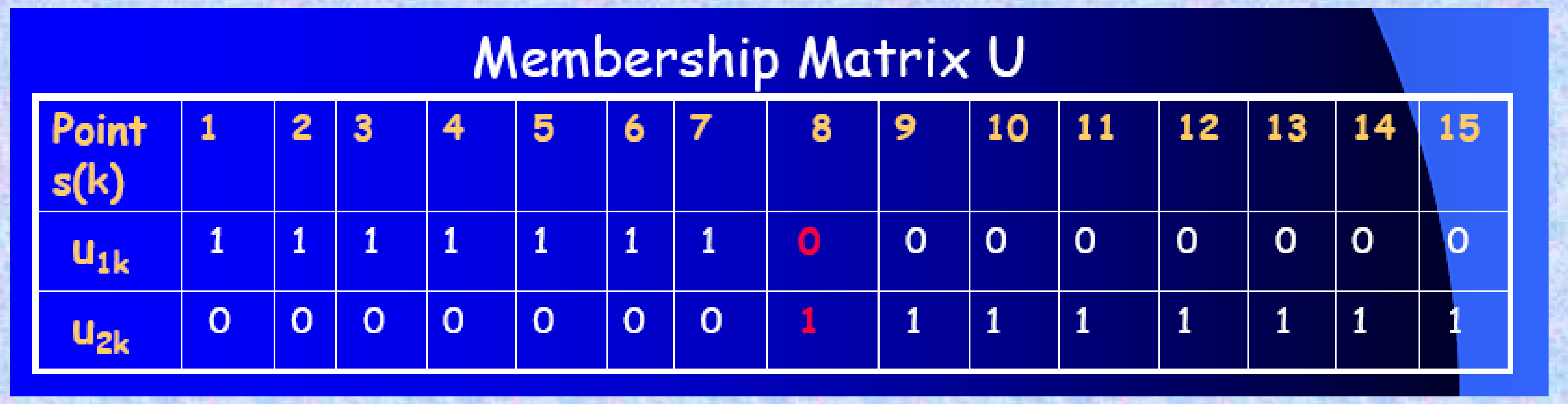
$$\mu_i = \frac{1}{|G_i|}\left(\sum_{k,\, x_k \in G_i} x_k\right)$$

- Disadvantages
  - What happens when there is overlap between classes… that is a point is equally close to two cluster centers…… Algorithm will not terminate
  - The Terminating condition is modified to “Change in cost function (computed at the end of the Classification) is below some threshold rather than 0”.

An Example

- The no of clusters is **two** in this case.
- But still there is some overlap





Membership Matrix U

Point s(k)	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
u_{1k}	1	1	1	1	1	1	1	0	0	0	0	0	0	0	0
u_{2k}	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1

Decision Regions and Boundaries

A classifier partitions a feature space into class-labeled ***decision regions (DRs)***.

If decision regions are used for a possible and unique class assignment, the regions must cover R^d and be disjoint (non-overlapping. In Fuzzy theory, decision regions may be overlapping.

The border of each decision region is a ***Decision Boundary (DBs)***.

Typical classification approach is as follows:

Determine the decision region (in R^d) into which X falls, and assign X to this class.

This strategy is simple. But determining the DRs is a challenge.

It may not be possible to visualize, DRs and DBs, in a general classification task with a large number of classes and higher feature space (dimension).

Classifiers are based on *Discriminant functions*.

In a C-class case, Discriminant functions are denoted by: $g_i(X)$, i = 1,2,…,C.

This partitions the R^d into C distinct (disjoint) regions, and the process of classification is implemented using the *Decision Rule*:

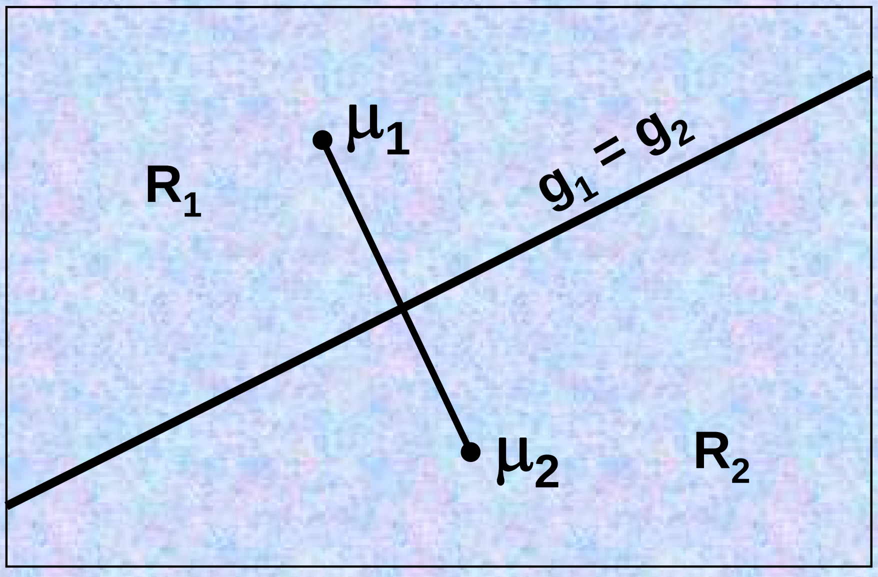
Assign X to class C_m (or region m), where: $g_m(X) > g_i(X), \forall i, i \neq m.$

***Decision Boundary* is defined by the locus of points, where:**

$$g_k(X) = g_l(X), k \neq l$$

Minimum distance classifier:

Discriminant function is based on the distance to the class mean:



$$g_1(X) = \left\| \vec{X} - \vec{\mu_1} \right\|; \quad g_2(X) = \left\| \vec{X} - \vec{\mu_2} \right\|$$

Let the discrimination function for the ith class be:

$$g_i(\overrightarrow{X}) = P(C_i \mid \overrightarrow{X}), \quad \text{and assume } P(C_i) = P(C_j), \forall i, j; i \neq j.$$

Remember, multivariate Gaussian density?

$$g_i(X) = P(X \mid C_i) = \frac{1}{\sqrt{\det(\Sigma)(2\pi)^d}} \exp[-\frac{(X-\mu_i)^T \Sigma^{-1} (X-\mu_i)}{2}]$$

Define:

$$G_i(X) = \log[P(X \mid C_i)] = \log[\frac{1}{\sqrt{\det(\Sigma)(2\pi)^d}}] - \frac{(X-\mu_i)^T \Sigma^{-1} (X-\mu_i)}{2}$$

$$= k.\overrightarrow{d}_i^{\,2} + q$$

Thus the classification is now influenced by the square distance (hyper-dimensional) of X from μ_i, weighted by the Σ^{-1}.

Let us examine:

$$\overrightarrow{d}_i^{\,2} = (X-\mu_i)^T \Sigma^{-1} (X-\mu_i)$$

This quadratic term (scalar) is known as the *Mahalanobis distance* (the distance from X to μ_i in feature space).

$$\vec{d}_i^{\,2} = (X - \mu_i)^T \Sigma^{-1} (X - \mu_i)$$

For a given X, some $G_m(X)$ is largest and also $(d_m)^2$ is the smallest, for a class i = m.

***<u>Simplest case</u>*: Σ = I, the criteria becomes the Euclidean distance norm.**

This is equivalent to obtaining the mean μ_m, for which X is the nearest, for all μ_i. The distance function is then:

$$\vec{d}_i^{\,2} = \|X - \mu_i\|^2 = X^T X - 2\mu_i^T X + \mu_i^T \mu \quad \text{(all vector notations)}$$

$$\text{Thus,} \quad G_i(X) = d_i^2/2 = (X^T X)/2 - \mu_i^T X + (\mu_i^T \mu)/2$$

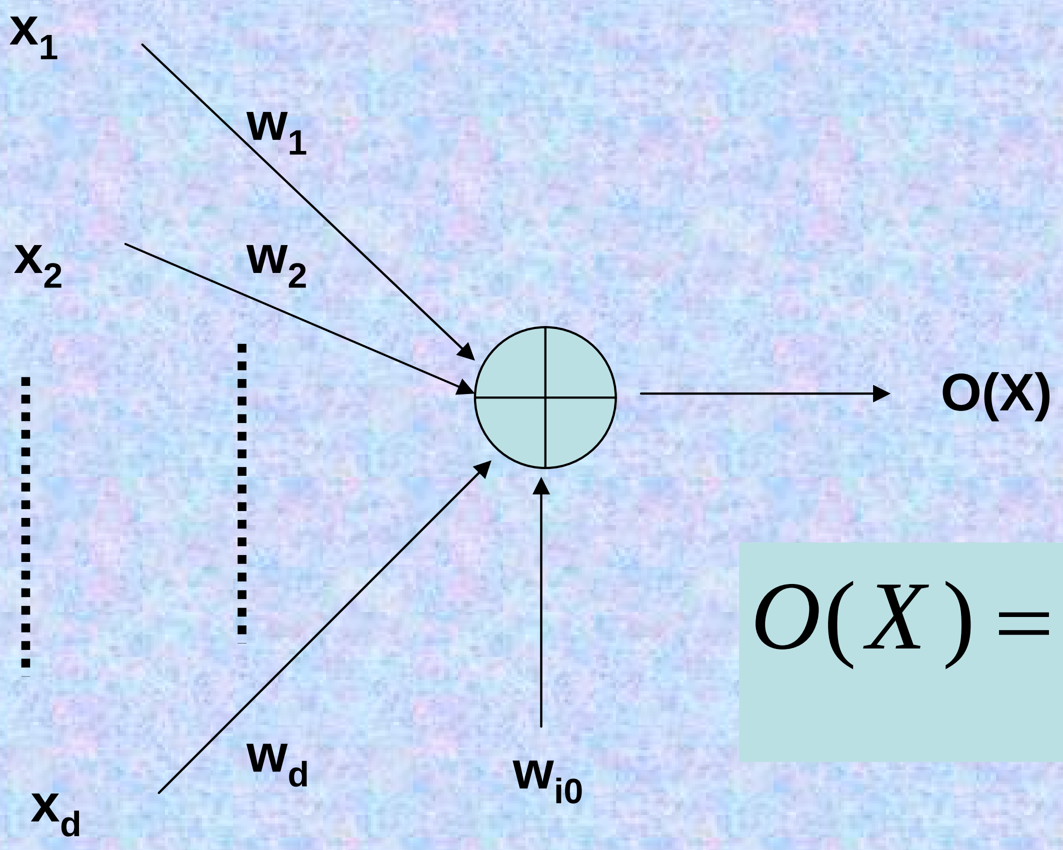
$$= \omega_i^T X + \omega_{i0}$$

Neglecting the class-invariant term.

$$\textit{where,} \ \omega_i^T = \mu_i \ \text{and} \ \omega_{i0} = -\frac{\mu_i^T \mu}{2}$$

This gives the simplest linear discriminant function or correlation detector.

The perceptron (ANN) built to form the linear discriminant function



$$O(X) = (\sum_i w_i x_i) + w_{i0}$$

View this as (in 2-D space):

$$Y = MX + C$$

The decision region boundaries are determined by solving :

$$G_i(X) = G_j(X), \text{ which gives} : (\omega_{\mathrm{i}}^{\mathrm{T}} - \omega_{\mathrm{j}}^{\mathrm{T}})X + (\omega_{i0} - \omega_{j0}) = 0$$

This is an expression of a hyperplane separating the decision regions in R^d. The hyperplane will pass through the origin, if:

$$\omega_{i0} = \omega_{j0}$$

Generalized results (Gaussian case) of a discriminant function:

$$G_i(X) = \log[P(X \mid C_i)] = \log[\frac{1}{\sqrt{\det(\Sigma_i)(2\pi)^d}}] - \frac{(X - \mu_i)^T \Sigma_i^{-1}(X - \mu_i)}{2}$$

$$= -\frac{1}{2}(X - \mu_i)^T \Sigma_i^{-1}(X - \mu_i) - (\frac{d}{2})\log(2\pi) - \frac{1}{2}\log(\Sigma_i)$$

The mahalanobis distance (quadratic term) spawns a number of different surfaces, depending on Σ^{-1}. It is basically a vector distance using a Σ^{-1} norm. It is denoted as:

$$\left\| X - \mu_i \right\|_{\Sigma_i^{-1}}^2$$

Make the case of Baye's rule more general for class assignment. Earlier we has assumed that:

$$g_i(\overrightarrow{X}) = P(C_i \mid \overrightarrow{X}), \quad \text{assuming } P(C_i) = P(C_j), \forall i, j; i \neq j.$$

Now, $G_i(\overrightarrow{X}) = \log[P(C_i \mid \overrightarrow{X}).P(\overrightarrow{X})] = \log[P(\overrightarrow{X} \mid C_i)] + \log[\mathrm{P}(C_i)]$

$$G_i(X) = \log[\frac{1}{\sqrt{\det(\Sigma_i)(2\pi)^d}}] - \frac{(X-\mu_i)^T \Sigma_i^{-1} (X-\mu_i)}{2} + \log[P(C_i)]$$

$$= -\frac{1}{2}(X-\mu_i)^T \Sigma_i^{-1} (X-\mu_i) - (\frac{d}{2})\log(2\pi) - \frac{1}{2}\log(\sum_i) + \log[P(C_i)]$$

$$= -\frac{1}{2}(X-\mu_i)^T \Sigma_i^{-1} (X-\mu_i) - \frac{1}{2}\log(\sum_i) + \log[P(C_i)]$$ **Neglecting the constant term**

<u>Simpler case</u>: $\Sigma_i = \sigma^2 I$, **and eliminating the class-independent bias, we have:**

$$G_i(X) = -\frac{1}{2\sigma^2}(X-\mu_i)^T (X-\mu_i) + \log[P(C_i)]$$

These are loci of constant hyper-spheres, centered at class mean.

If Σ is a diagonal matrix, with equal/unequal σ_{ii}^2:

$$\Sigma = \begin{bmatrix} \sigma_1^2 & 0 & . & . & 0 \\ 0 & \sigma_2^2 & . & . & 0 \\ . & . & . & . & . \\ . & . & . & . & . \\ 0 & 0 & . & . & \sigma_d^2 \end{bmatrix} \quad and \quad \Sigma^{-1} = \begin{bmatrix} 1/\sigma_1^2 & 0 & . & . & 0 \\ 0 & 1/\sigma_2^2 & . & . & 0 \\ . & . & . & . & . \\ . & . & . & . & . \\ 0 & 0 & . & . & 1/\sigma_d^2 \end{bmatrix}$$

Considering the discriminant function:

$$G_i(X) = -\frac{1}{2}(X-\mu_i)^T \Sigma^{-1}(X-\mu_i) - \frac{1}{2}\log(\Sigma) + \log[P(C_i)]$$

This now will yield a weighted distance classifier. Depending on the covariance term (*more spread/scatter or not*), we tend to put more emphasis on some feature vector components than the other.

Check out the following:

This will give hyper-elliptical surfaces in R^d, for each class.

It is also possible to linearise it.

More general decision boundaries

Take $P(C_i)$ = K for all i, and eliminating the class independent terms yield:

$$G_i(X) = (X - \mu_i)^T \Sigma^{-1} (X - \mu_i)$$

$$\vec{d}_i^{\,2} = (X - \mu_i)^T \Sigma^{-1} (X - \mu_i) = -X^T \Sigma^{-1} X + 2\mu_i^T \Sigma^{-1} X - \mu_i^T \Sigma^{-1} \mu_i$$

$$G_i(X) = (\Sigma^{-1}\mu_i)^T X - \frac{1}{2}\mu_i^T \Sigma^{-1}\mu_i$$ **as Σ is symmetric.**

$$\text{Thus,} \quad G_i(X) = \omega_i^T X + \omega_{i0}$$

$$where \ \ \omega_i = \Sigma^{-1}\mu_i \text{ and } \omega_{i0} = -\frac{1}{2}\mu_i^T \Sigma^{-1}\mu_i$$

Thus the decision surfaces are hyperplanes and decision boundaries will also be linear (use $G_i(X) = G_j(X)$, as done earlier)

The discriminant function for linearly separable classes is:

$$g_i(X) = \omega_i^T X + \omega_{i0}$$

where, ω_i is a dx1 vector of weights used for class i.

This function leads to DBs that are hyperplanes. It's a point in 1D, line in 2-D, planar surfaces in 3-D, and ……. .

3-D case: $$(\omega_1 \omega_2 \omega_3)\begin{pmatrix} x_1 \\ x_2 \\ x_3 \end{pmatrix} = 0$$ **is a plane passing through the origin.**

In general, the equation: $$\omega^T(\overrightarrow{X} - \overrightarrow{X_d}) = 0; \Rightarrow \omega^T \overrightarrow{X} - d = 0$$

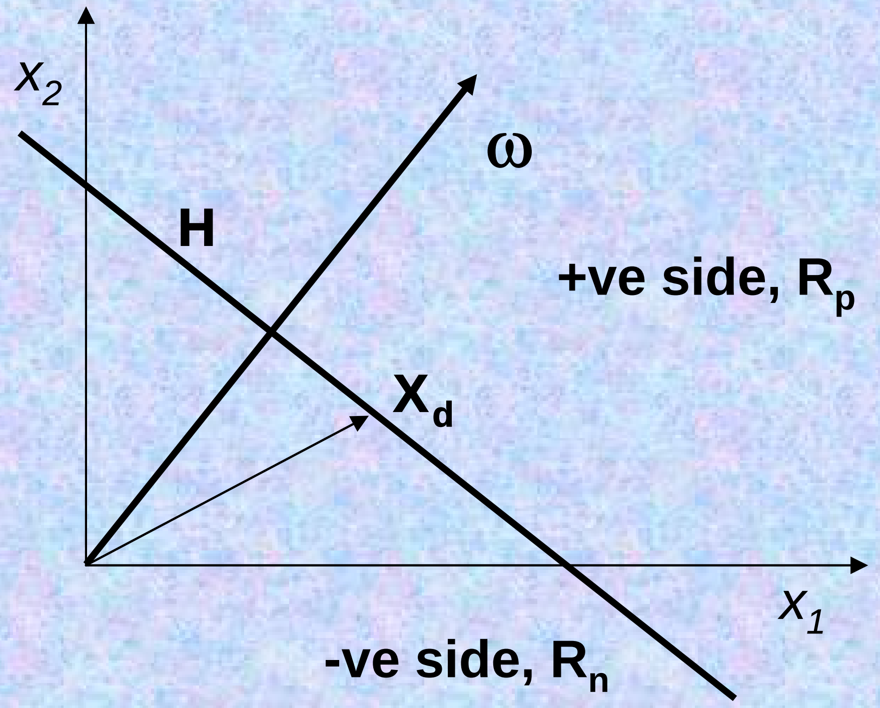
represents a plane H passing through any point (position vector) X_d.

This plane partitions the space into two mutually exclusive regions, say R_p and R_n. The assignment of the vector X to either the +ve side, or –ve side or along H, can be implemented by:

$$\omega^T \overrightarrow{X} - d \begin{cases} > 0 & \text{if } X \in R_p \\ = 0 & \text{if } X \in H \\ < 0 & \text{if } X \in R_n \end{cases}$$

Linear Discriminant Function g(X):

$$g(X) = \omega^T \overrightarrow{X} - d$$



Orientation of H is determined by ω.

Location of H is determined by d.

H is a hyperplane for d > 3.   The figure shows a 2D representation.

Quadratic Decision Boundaries

In R^d with $X = (x_1, x_2, \ldots, x_d)^T$, consider the equation:

$$\sum_{i=1}^{d} w_{ii} x_i^2 + \sum_{i=1}^{d-1} \sum_{j=i+1}^{d} w_{ij} x_i x_j + \sum_{i=1}^{d} w_i x_i + w_o = 0 \qquad ..1$$

The above equation is defined by a quadric discriminant function, which yields a quadric surface.

If d=2, $X = (x_1, x_2)^T$ equation (1) becomes:

$$w_{11} x_1^2 + w_{22} x_2^2 + w_{12} x_1 x_2 + w_1 x_1 + w_2 x_2 + w_0 = 0 \qquad ..2$$

Special cases of equation:

$$w_{11}x_1^2 + w_{22}x_2^2 + w_{12}x_1x_2 + w_1x_1 + w_2x_2 + w_0 = 0 \qquad ..2$$

Case 1:

$w_{11} = w_{22} = w_{12} = 0$; Eqn. (2) defines a line.

Case 2:

$w_{11} = w_{22} = K$; $w_{12} = 0$; defines a circle.

Case 3:

$w_{11} = w_{22} = 1$; $w_{12} = w_1 = w_2 = 0$; defines a circle whose center is at the origin.

Case 4:

$w_{11} = w_{22} = 0$; defines a bilinear constraint.

Case 5:

$w_{11} = w_{12} = w_2 = 0$; defines a parabola with a specific orientation.

Case 6:

$$w_{11} \neq 0, w_{22} \neq 0, w_{11} \neq w_{22}; w_{12} = w_1 = w_2 = 0$$

defines a simple ellipse.

Selecting suitable values of w_i's, gives other conic sections.

For d $\geq$ 3, we define a family of hyper-surfaces in R^d.

$$\sum_{i=1}^{d} w_{ii} x_i^2 + \sum_{i=1}^{d-1} \sum_{j=i+1}^{d} w_{ij} x_i x_j + \sum_{i=1}^{d} w_i x_i + \omega_o = 0 \qquad ..1$$

In the above equation, the number of parameters:

2d + 1 + d(d-1)/2 = (d+1)(d+2)/2.

Organize these parameters, and manipulate the equation to obtain:

$$\overline{X}^T W \overline{X} + w^T \overline{X} + \omega_o = 0 \qquad ..3$$

w has d terms, ω_o has one term, and W (ω_{ij}) is a dxd matrix as:

$$\omega_{ij} = \begin{cases} w_{ii} & \text{if i = j} \\ \frac{1}{2} w_{ij} & \text{if i} \neq \text{j} \end{cases}$$

(d²-d) non-diagonal terms of the matrix W,
is obtained by duplicating (split into two parts):
d(d-1)/2 w_{ij}s.

In equation 3, the symmetric part of matrix W, contributes to the Quadratic terms. Equation 3 generally defines a hyperhyperboloidal surface. If W = I, we get a hyperplane.

Example of linearization:

$$g(X) = x_2 - x_1^2 - 3x_1 + 6 = 0$$

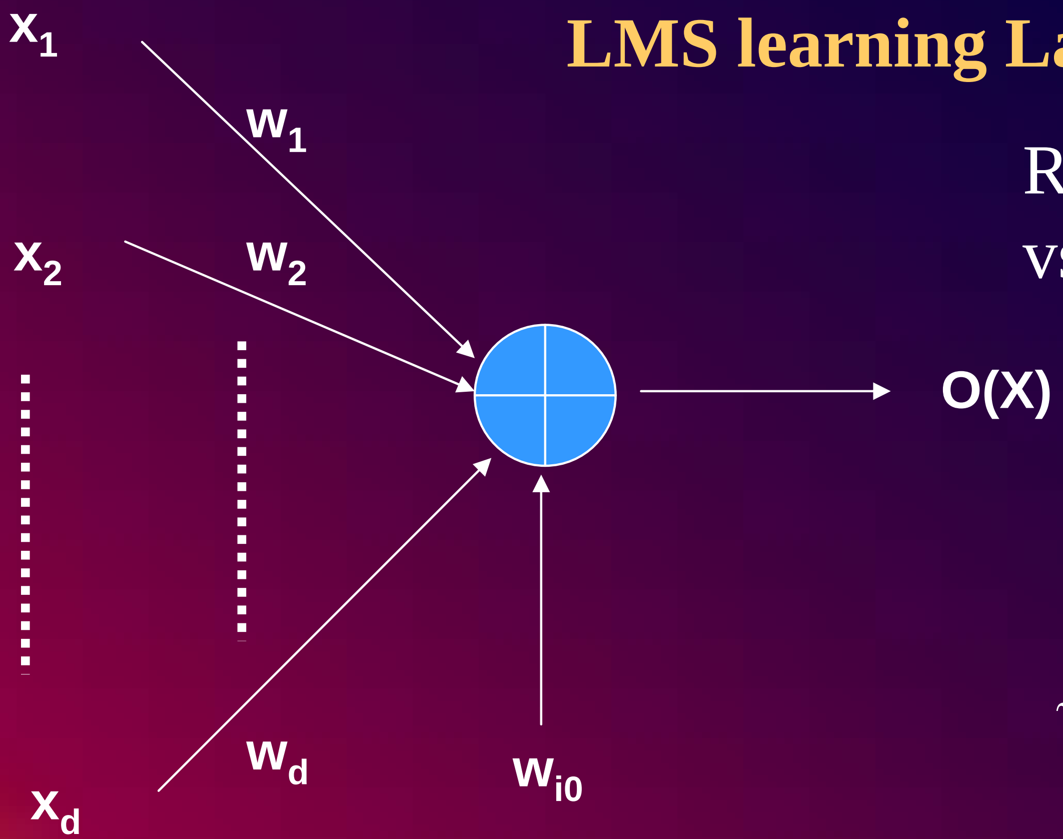
To **Linearize**, let $x_3 = x_1^2$. Then:

$$g(X) = x_2 - x_3 - 3x_1 + 6 = W^T X + w_o$$

where, $X = [x_1 x_2 x_3]^T$

and $W^T = [-3, 1, -1]$

LMS learning Law in BPNN or FFNN models

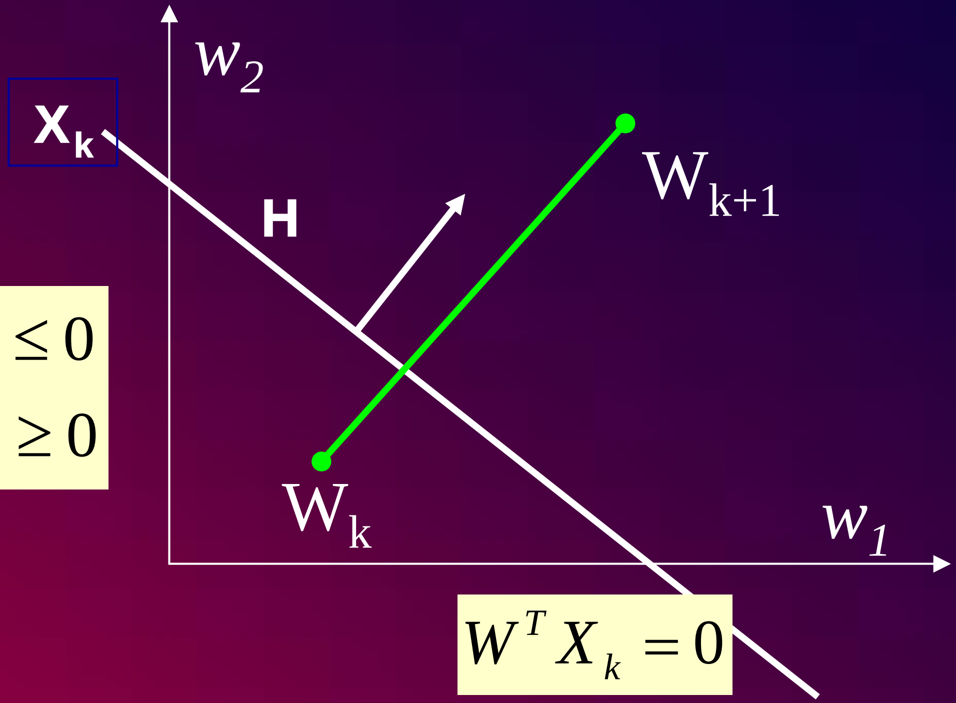


Read about perceptron
vs. multi-layer feedforward network

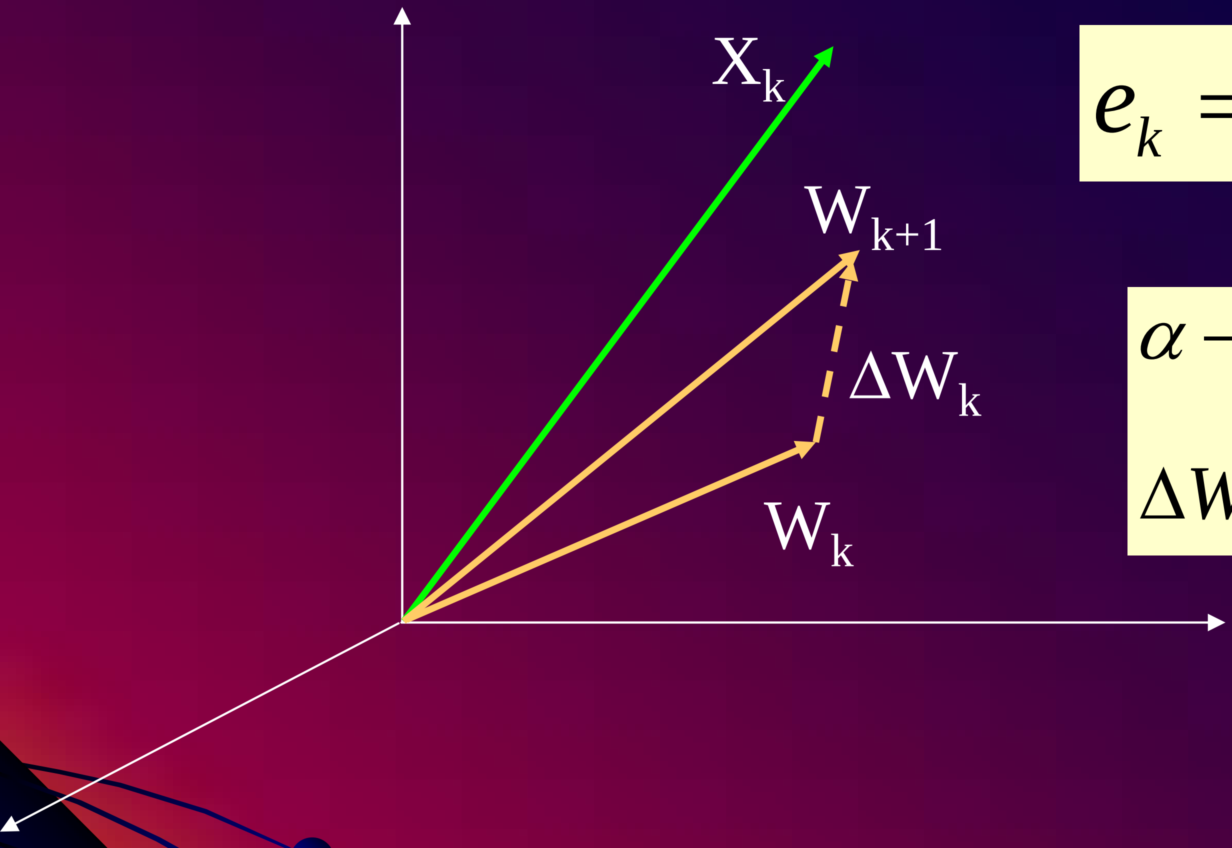
$$W_{k+1} = \begin{cases} W_k + \eta_k X_k & \text{if } X_k^T W_k \le 0 \\ W_k & \text{if } X_k^T W_k \ge 0 \end{cases}$$

η_κ is the learning rate parameter

$$W_{k+1} = \begin{cases} W_k + \eta_k X_k & \text{if } X_k \in X_1 \text{ and } X_k^T W_k \le 0 \\ W_k - \eta_k X_k & \text{if } X_k \in X_0 \text{ and } X_k^T W_k \ge 0 \end{cases}$$



In case of FFNN, the objective is to minimize the error term:



$$e_k = d_k - s_k = d_k - X_k^T W_k$$

$\alpha - LMS$ Learning Algorithm:

$$\Delta W_k = \eta e_k \hat{X}_k$$

MSE error surface:

$$\xi_k = \frac{1}{2}[d_k - X_k^T W_k]^2 = E/2 - P^T W + (1/2) W^T R W.$$

$$P^T = E[d_k X_k^T];$$

$$R = E[X_k X_k^T] = E[\begin{bmatrix} 1 & x_1^k & x_n^k \\ x_1^k & x_1^k x_1^k & x_1^k x_n^k \\ & & \\ x_n^k & x_n^k x_1^k & x_n^k x_n^k \end{bmatrix}$$

$$\nabla \xi = \left(\frac{\delta\xi}{\delta w_0}, \frac{\delta\xi}{\delta w_1}, \ldots\ldots, \frac{\delta\xi}{\delta w_n}\right)^T = -P + RW$$

Thus,

$$\hat{W} = R^{-1} P$$

Some of the latest advancements in Pattern recognition technology deal with:

- **Neuro-fuzzy (soft computing) concepts**
- **Reinforcement learning**
- **Learning from small data sets**
- **Generalization capabilities**
- **Evolutionary Computations**
- **Genetic algorithms**
- **Pervasive computing**
- **Neural dynamics**
- **Support Vector machines**