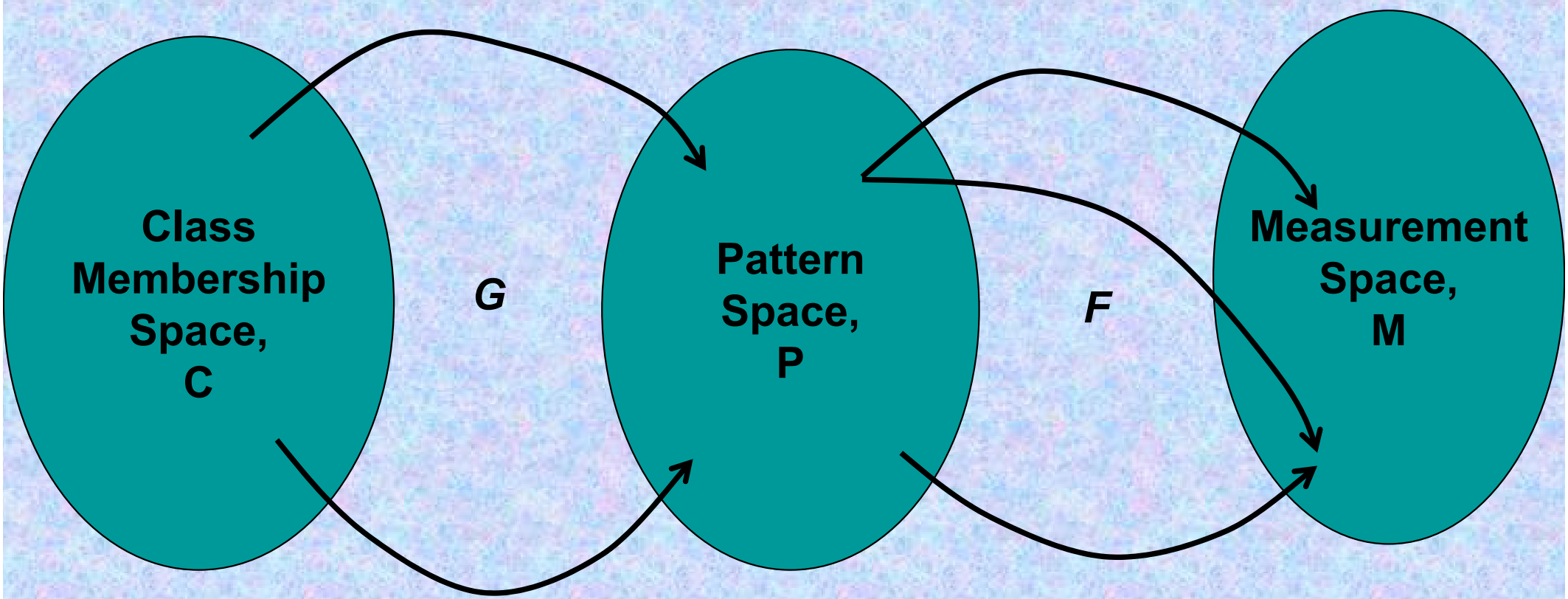
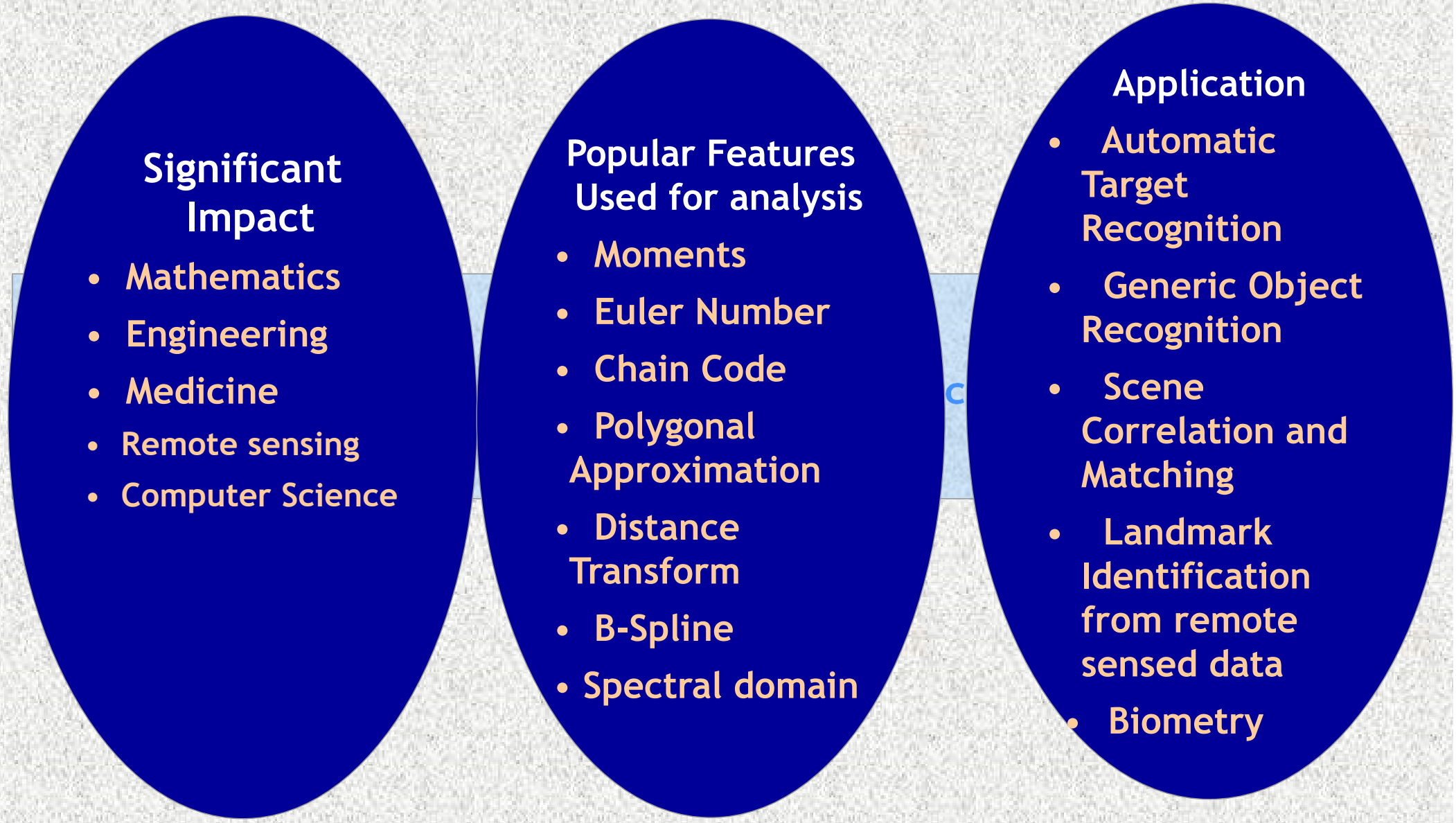
Pattern Recognition

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.



$$C = G^{-1}(P) = G^{-1}(F^{-1}(M))$$



An Important Area of Research in
Computer Vision and Visual Perception

Features must be invariant to:

- **Translation**
- **Rotation**
- **Scale**
- **Noise**
- **Projective (?)**

Computational cost must not be high

Must be distinct and unique for a given shape.

Preferably have graceful degradation due to discontinuities and missing parts

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.

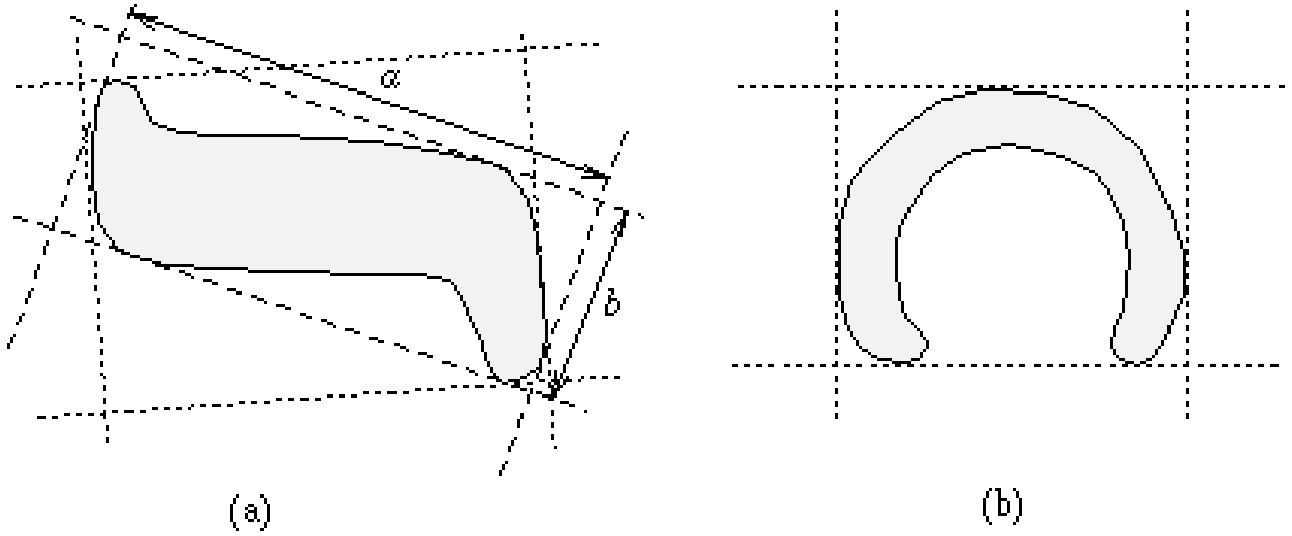


Figure 6.24 *Elongatedness: (a) Bounding rectangle gives acceptable results, (b) bounding rectangle cannot represent elongatedness.*

Elongatedness:

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

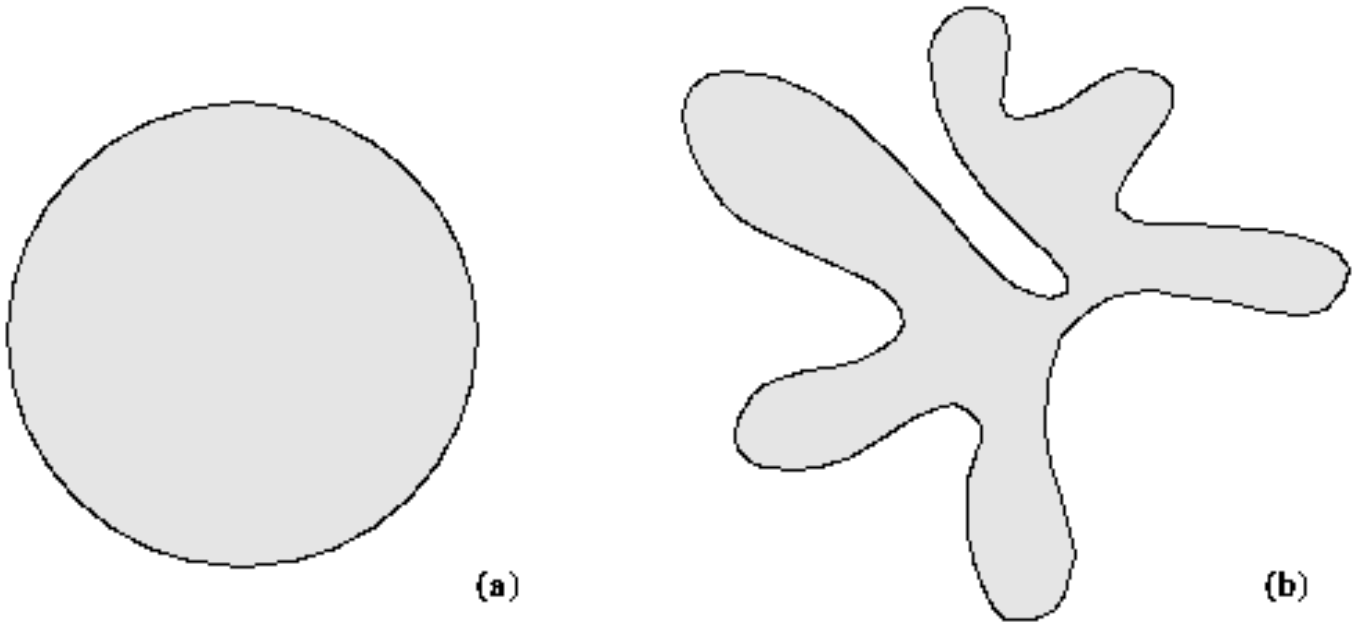


Figure 6.25 *Compactness: (a) Compact, (b) non-compact.*

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:

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

where the sums are taken over all points (*x*, *y*) contained within the region and ($\tilde{x}$, $\tilde{y}$) 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 ($= \mu_{00}$).

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. Let, the normalized unscaled central moments be:

$$m_{pq} = \frac{\mu_{pq}}{(\mu_{oo})^{\gamma}}; \gamma = (\frac{p+q}{2}) + 1$$

Up to order seven, the RST-invariant moments are:

$$M_1 = m_{20} + m_{02}$$

$$M_2 = (m_{20} - m_{02})^2 + 4m_{11}^2$$

$$M_3 = (m_{30} - 3m_{12})^2 + (3m_{21} - m_{03})^2$$

$$M_4 = (m_{30} + m_{12})^2 + (m_{21} + m_{03})^2$$

$$M_5 = (m_{30} - 3m_{12})(m_{30} + m_{12})\left[(m_{30} + m_{12})^2 - 3(m_{21} + m_{03})^2\right] + (3m_{21} - m_{03})(m_{21} + m_{03})\left[3(m_{30} + m_{12})^2 - (m_{21} + m_{03})^2\right]$$

$$M_6 = (m_{20} + m_{02})\left[(m_{30} + m_{12})^2 - 3(m_{21} + m_{03})^2\right] + 4m_{11}(m_{30} + m_{12})(m_{03} + m_{21})$$

$$M_7 = (3m_{21} - m_{03})(m_{12} + m_{30})\left[(m_{30} + m_{12}^2 - 3(m_{21} + m_{03})^2\right] - (m_{30} - 3m_{12})(m_{12} + m_{03})\left[3(m_{30} + m_{12})^2 - (m_{21} + m_{03})^2\right]$$

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:

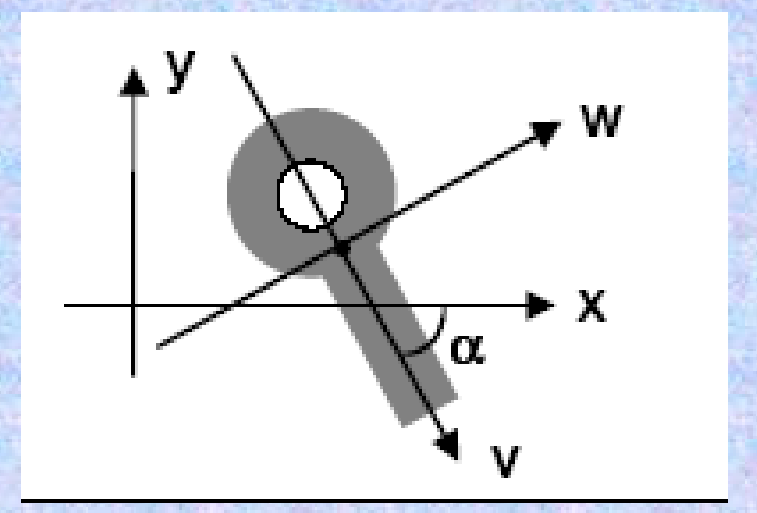
$$\theta = \frac{1}{2}\tan^{-1}\left[\frac{2m_{11}}{m_{20} - m_{02}}\right]$$

Geometric properties in terms of moments:

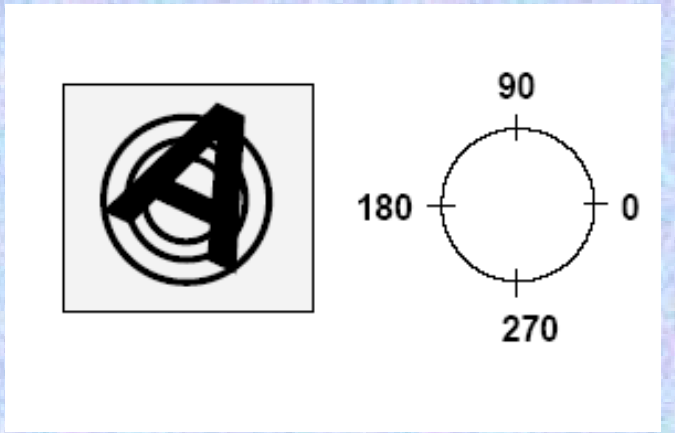
$$Area = m_{00}; \quad \tilde{x} = \frac{m_{10}}{m_{00}}; \tilde{y} = \frac{m_{01}}{m_{00}}$$

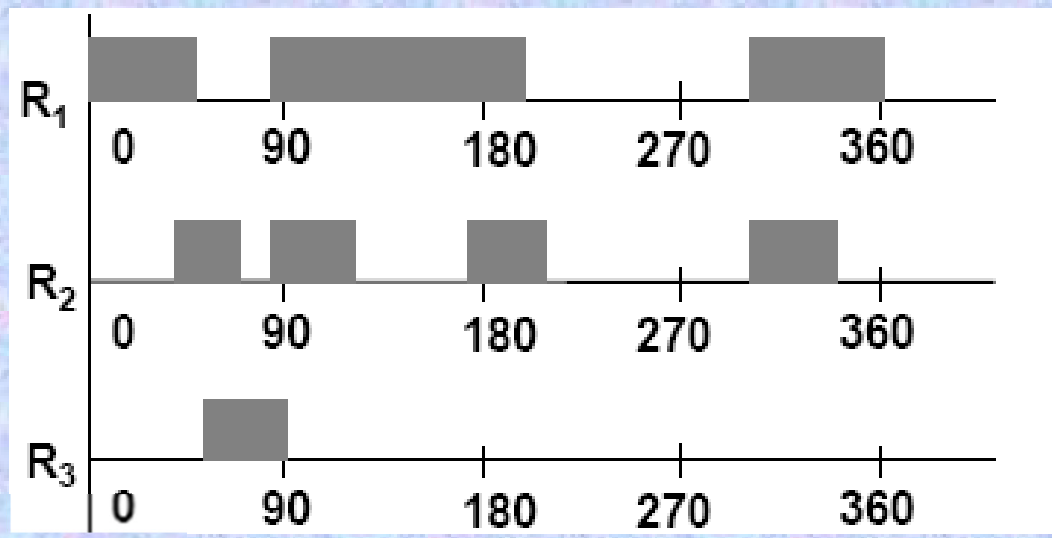
Axis of minimal inertia

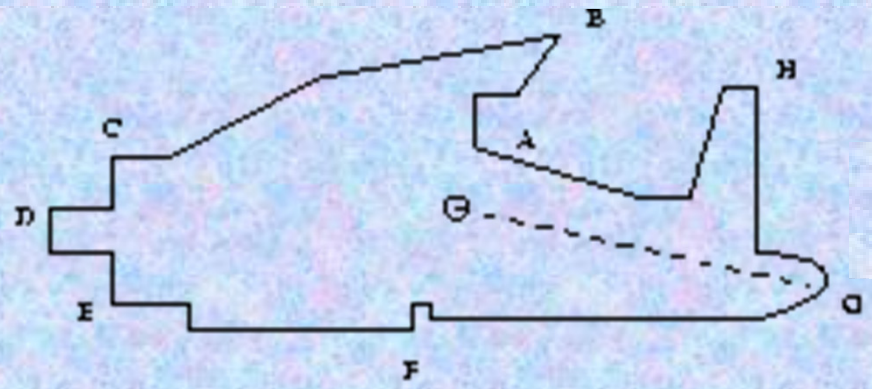
$$\tan 2\alpha = 2m_{xy}/(m_y - m_x)$$



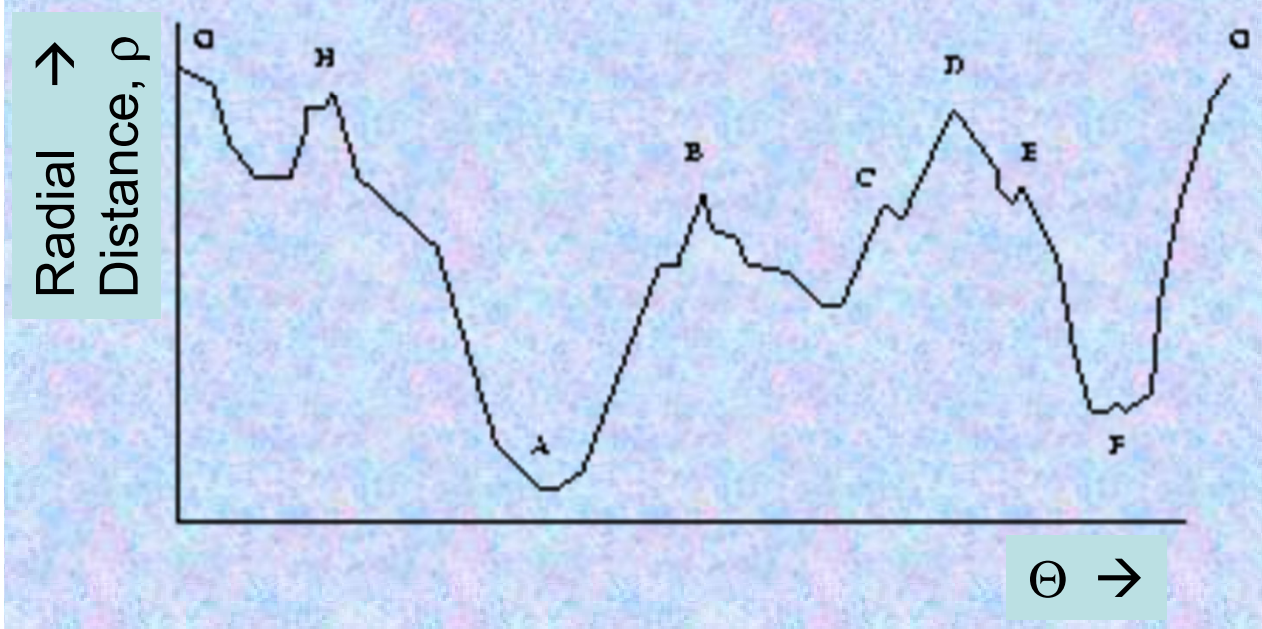
Polar Signature, Skeletons (MAT), B-splines are also used.







Polar Signature



Phase of DFT-based Signature Function

Coordinates of the boundary points of the shape are expressed as: (x_0, y_0), (x_1, y_1)……. (x_N, y_N)

OR

$$z_i = x_i + jy_i$$

Coefficient of the Fourier Transform is given as

$$Z(e^{j\omega}) = \sum_n z_k e^{(-j\omega n)}$$

Shape is rotated by an angle θ, and the starting point by l_0

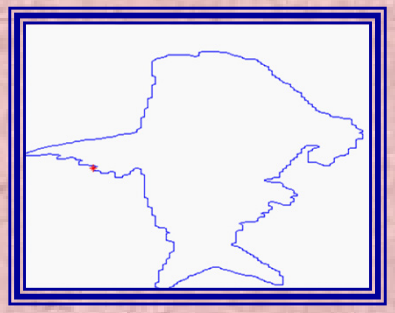
$$Z'_m = Z_m e^{j\theta} e^{(-jm2\pi l_0)}$$

$$Z_m = R_m e^{j\theta_m}$$

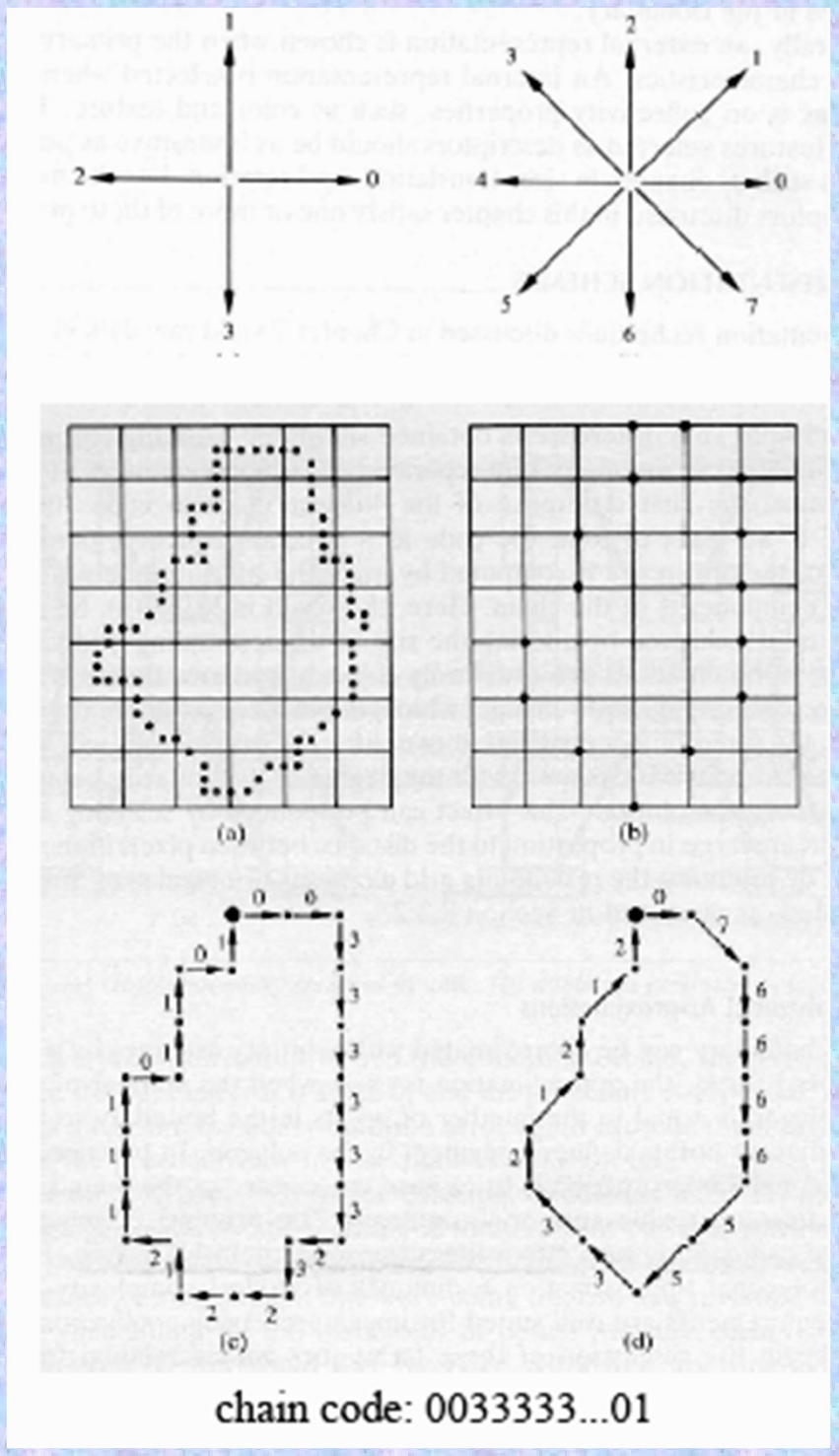
Shift in phase, due to rotation of the shape and change in starting point is

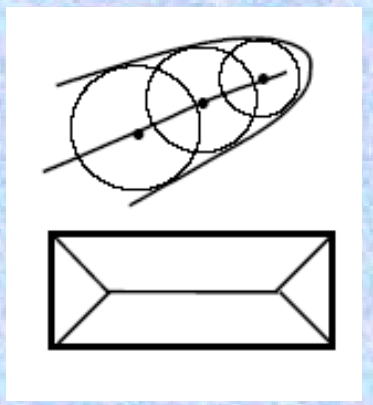
$$\theta'_m = \theta_m + \theta - 2\pi l_0 m / N$$

This leads to the derivation of the modified DFT coefficients for normalization against the scaling, rotation and shift in the starting point as depicted in the table below:



Invariance	Modified coefficients
Translation	$Z'_0 = 0$
Scale	$R'_m = R_m / S$
Rotation	$\Theta'_m = \Theta_m + \theta - (\Theta_{-1} + \Theta_{+1})/2$
Starting point	$\Theta'_m = \Theta_m + m(\Theta_{-1} - \Theta_{+1})/2$





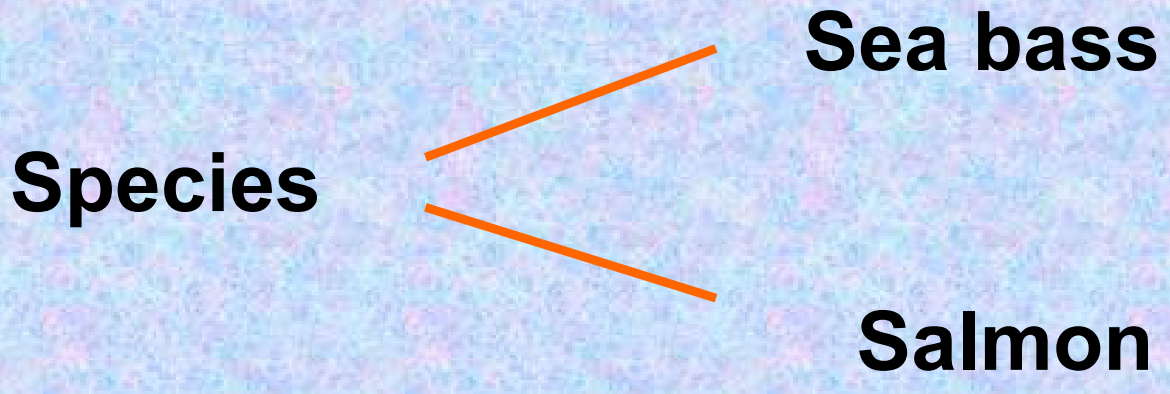
Skeletons - MAT

Read about :

- **CSS, a multi-scale representation;**
- **MCC**
- **Wavelet based descriptors**
- **Contour descriptor moments**
- **Distance functions – Hausdorff**
- **Shape Context**

An Example of Classification

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



Flower: Rose vs Tulips; Fruits: Apple vs Mango; Vehicles: Cars vs Trucks

– **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

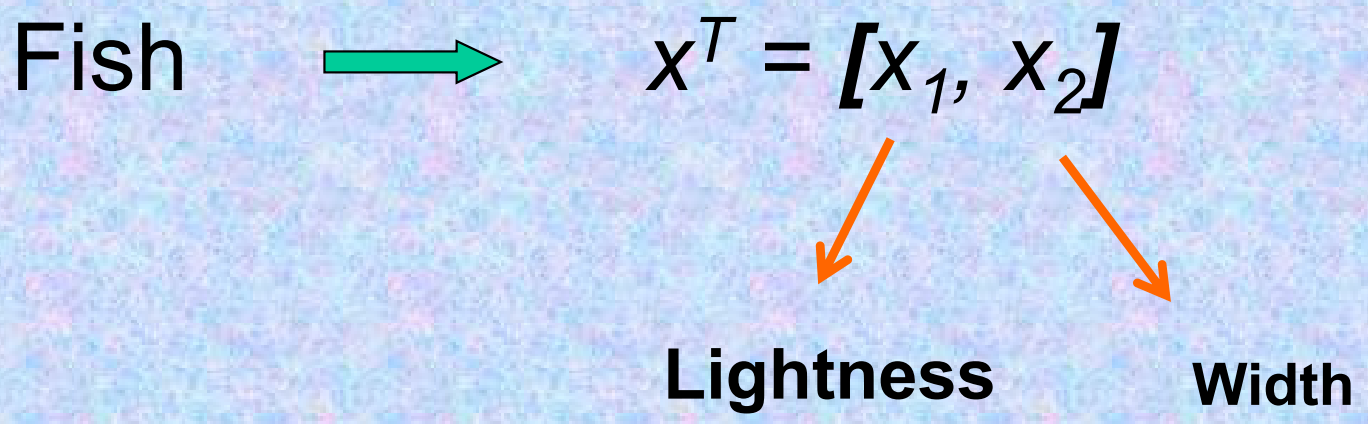
– **This is the set of all suggested features to explore for use in our classifier!**

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

Feature vector

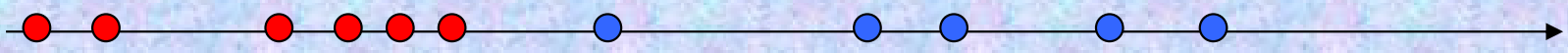
- **A Single feature may not be useful always for classification**

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



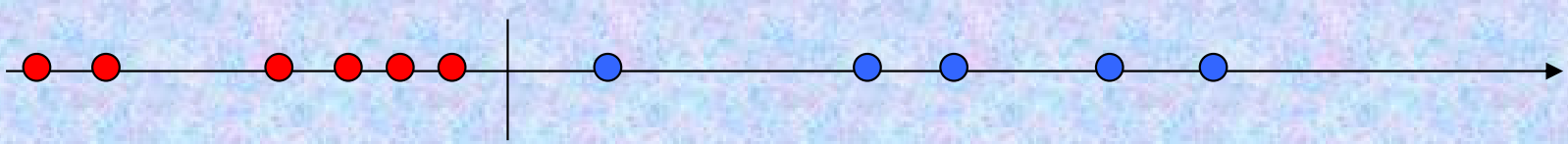
Feature space

- 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.

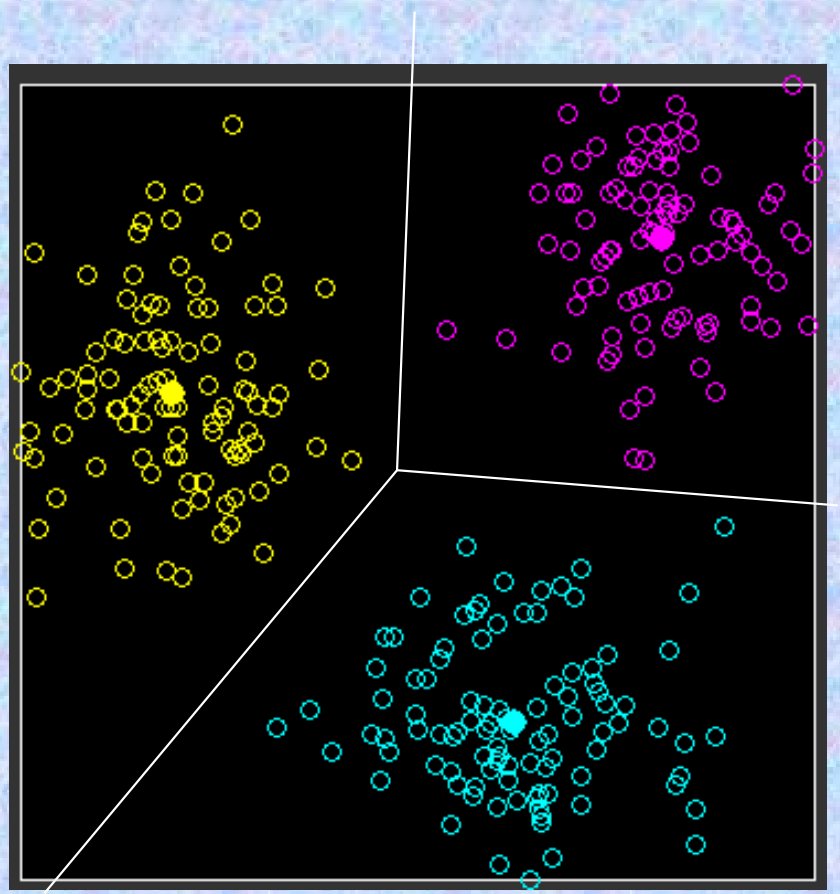


Point representing samples

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

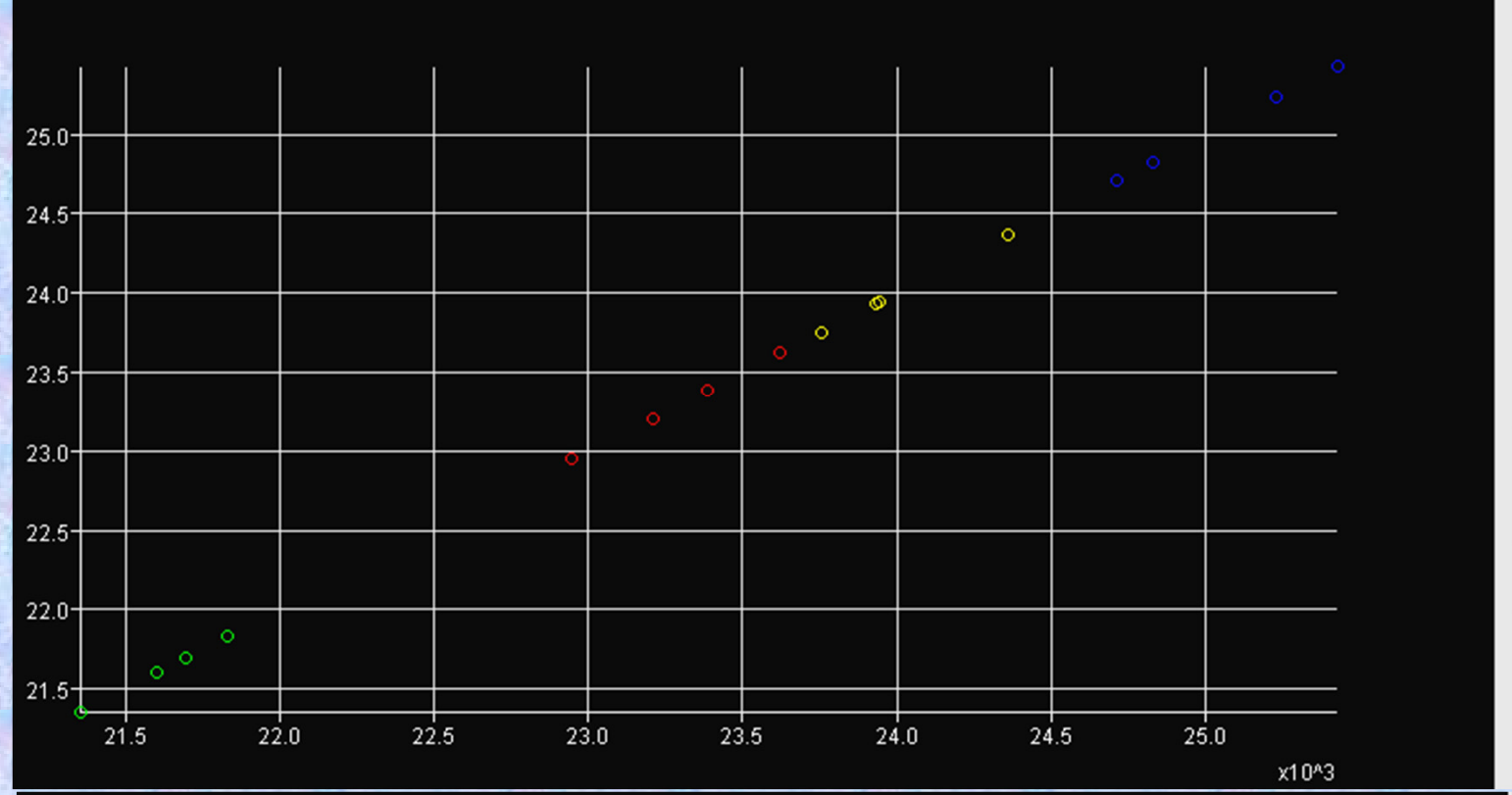


Decision boundary in one-dimensional case with two classes.

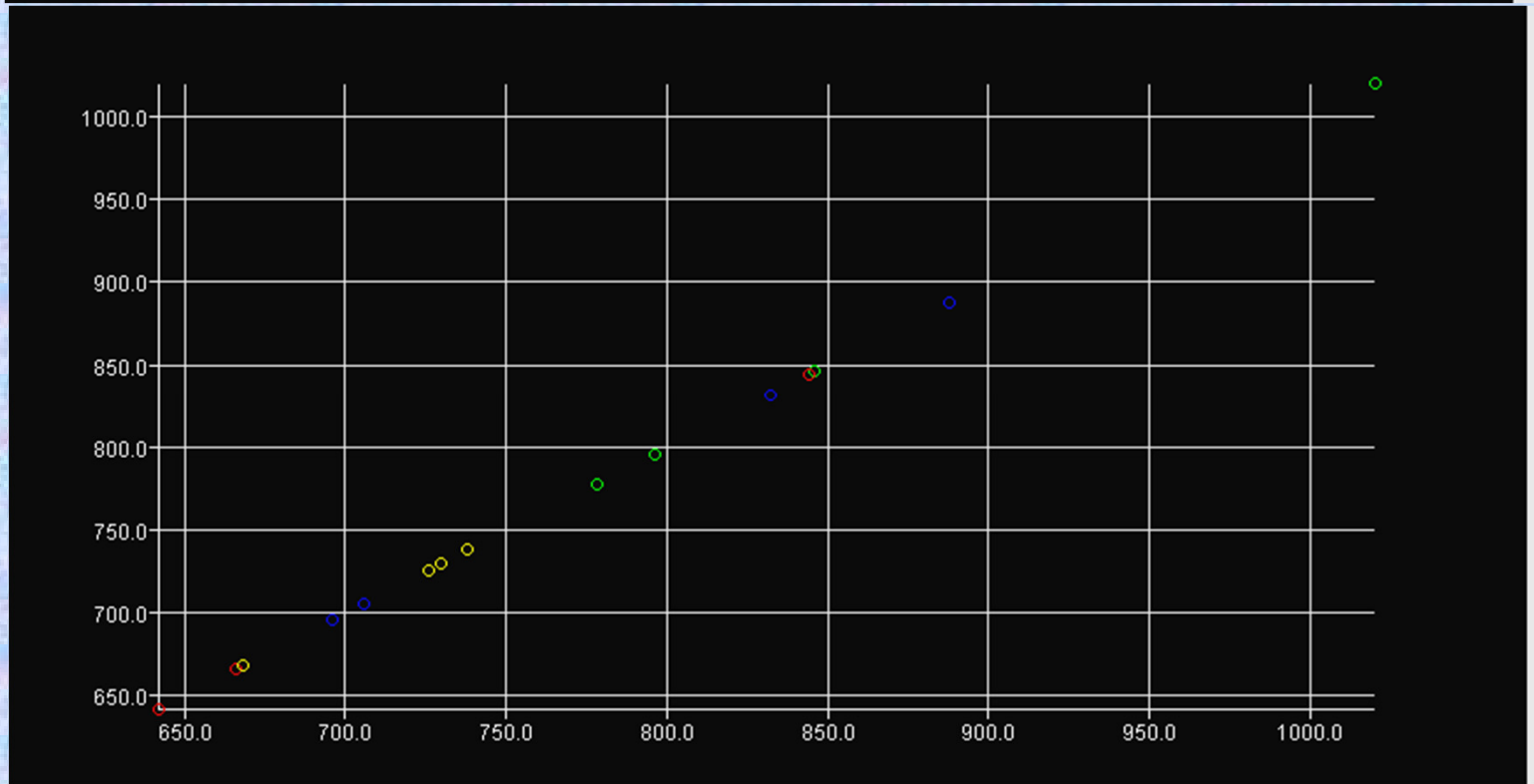


Decision boundary in 2 (or 3) dimensional case with three classes

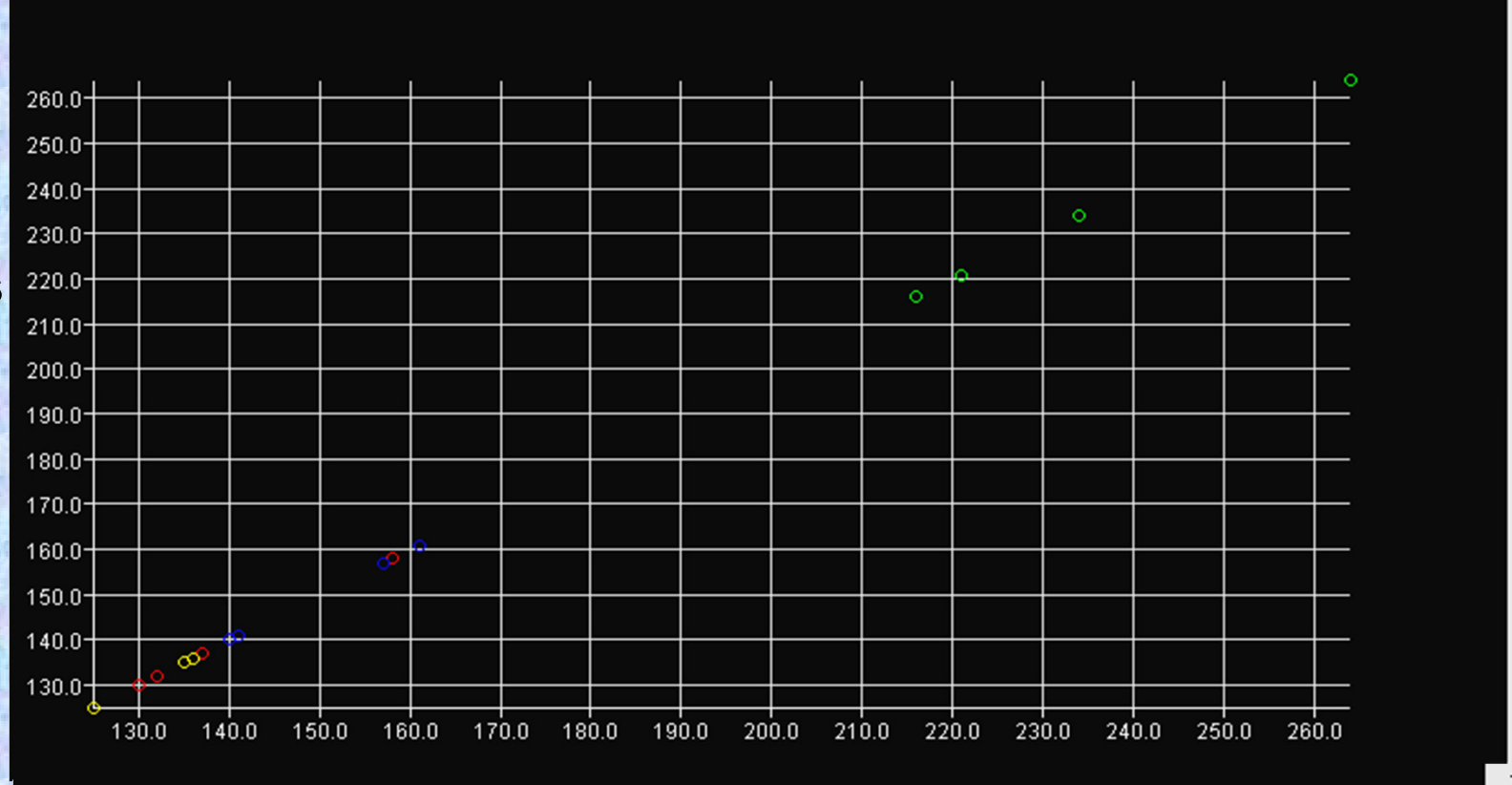
Area



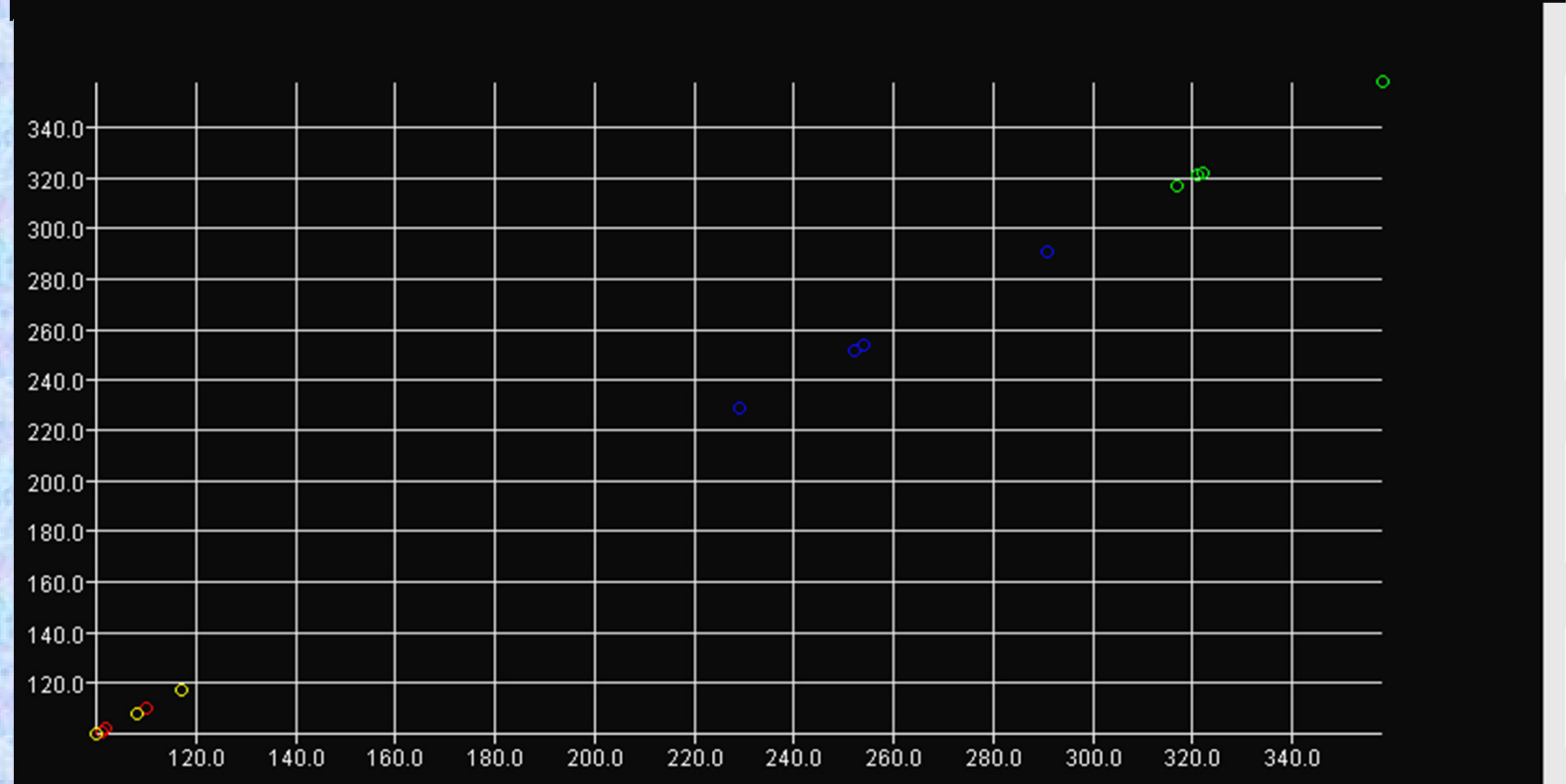
Perimeter



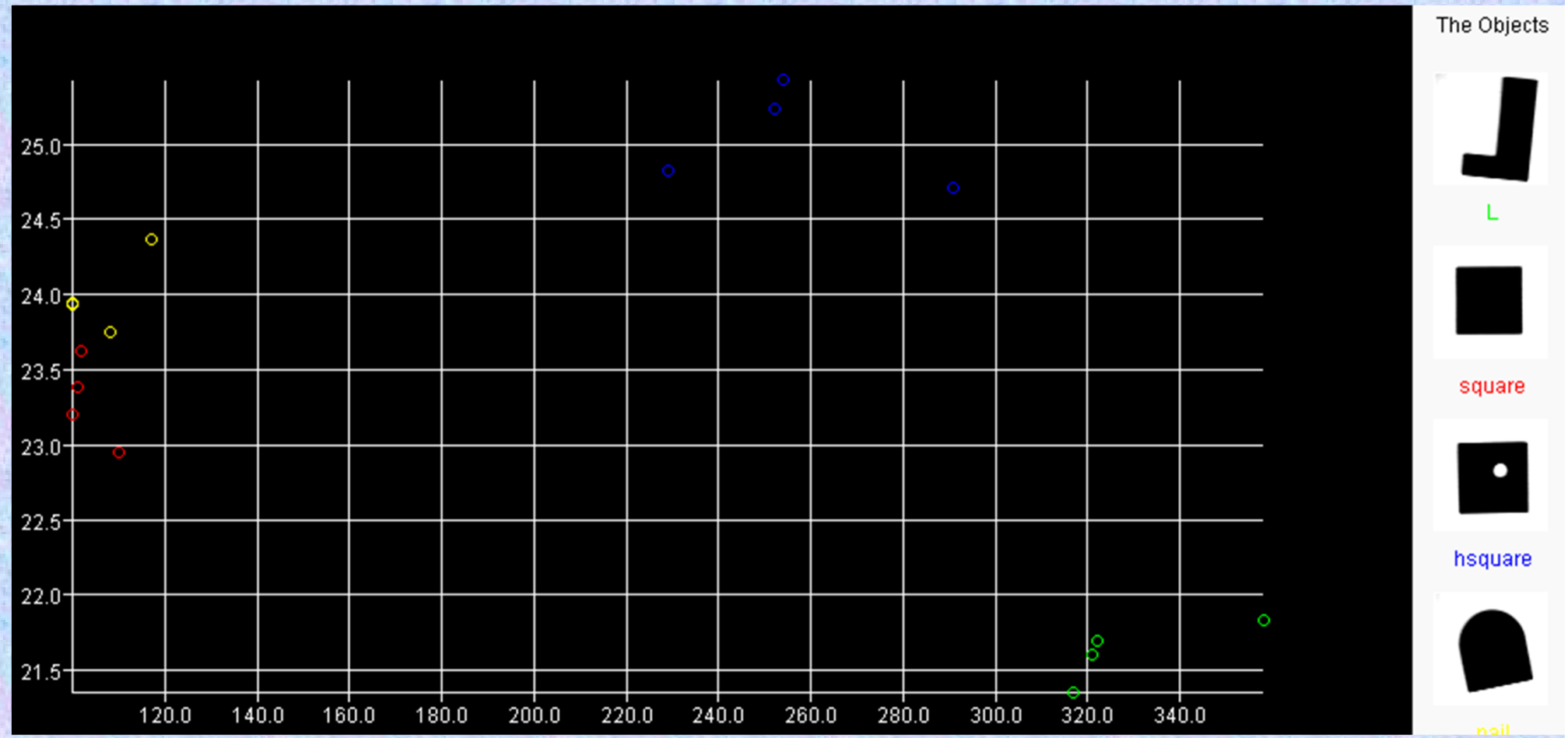
Compactness



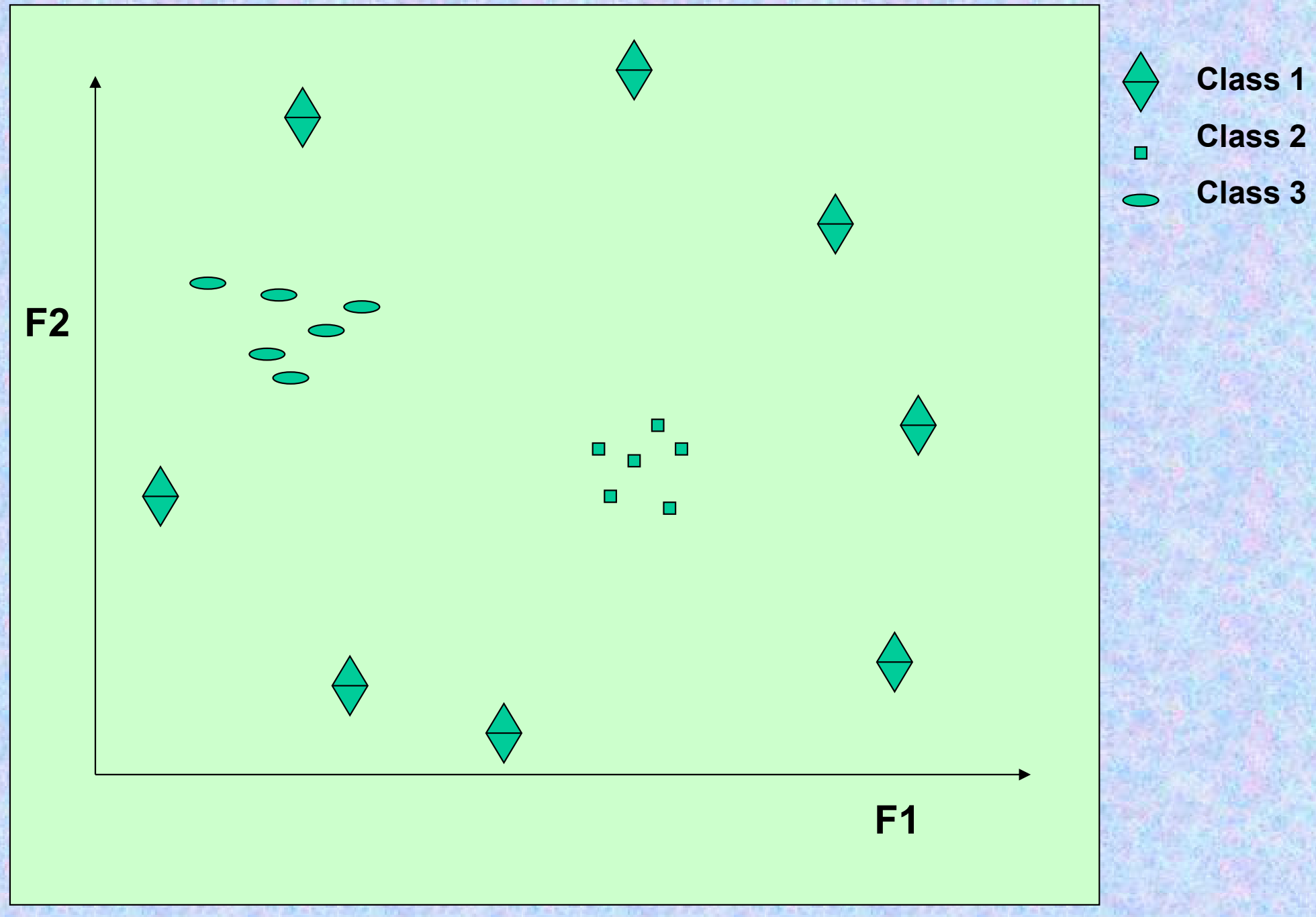
Elongation



Area



Elongation



Sample points in a two-dimensional feature space

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

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.**

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.**

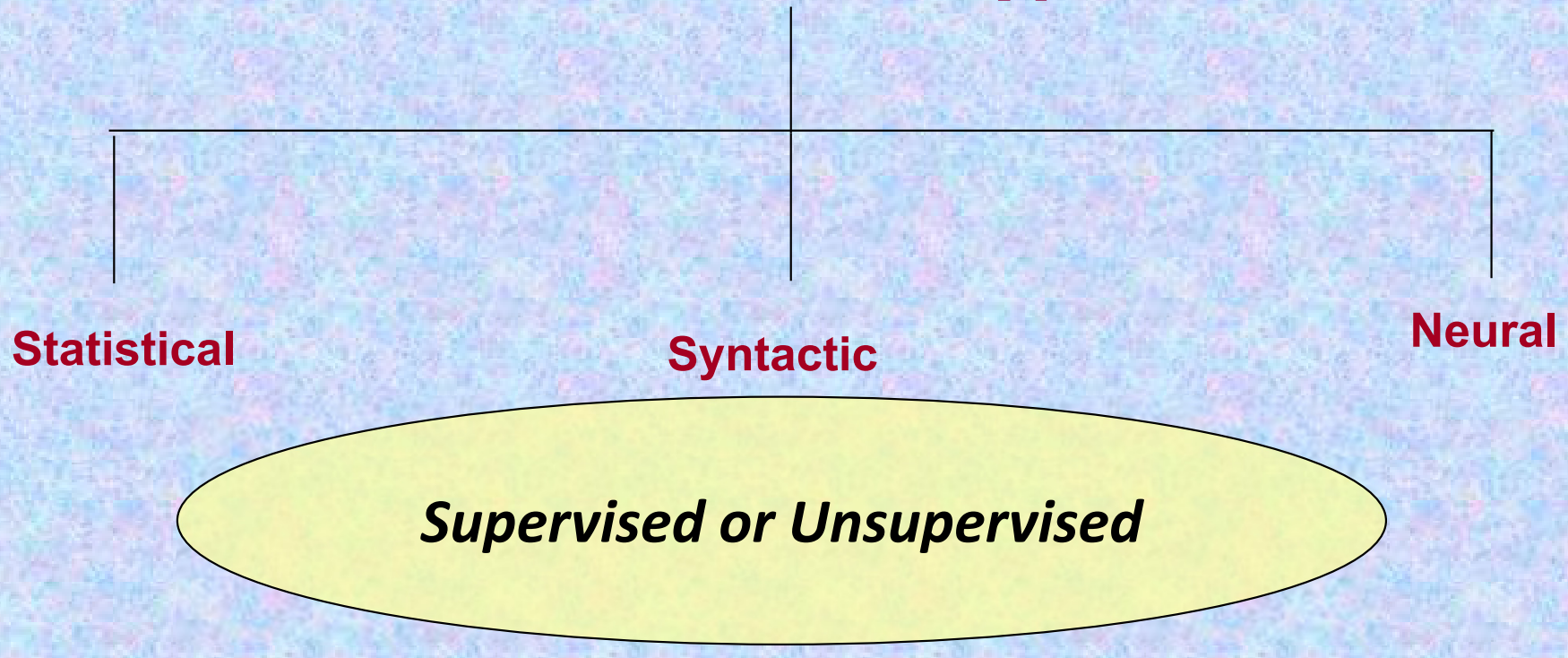
Learning

- **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.**

Classifier Types



Categories of Statistical Classifiers:

- **Linear**
- **Quadratic**
- **Piecewise**
- **Non-parametric**

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. (Uncondl.) for feature vector X.

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

$P(X \mid w_i)$ - Prob. (Class-Condnl.) Of feature vector X in class w_i

Bayes Theorem:

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

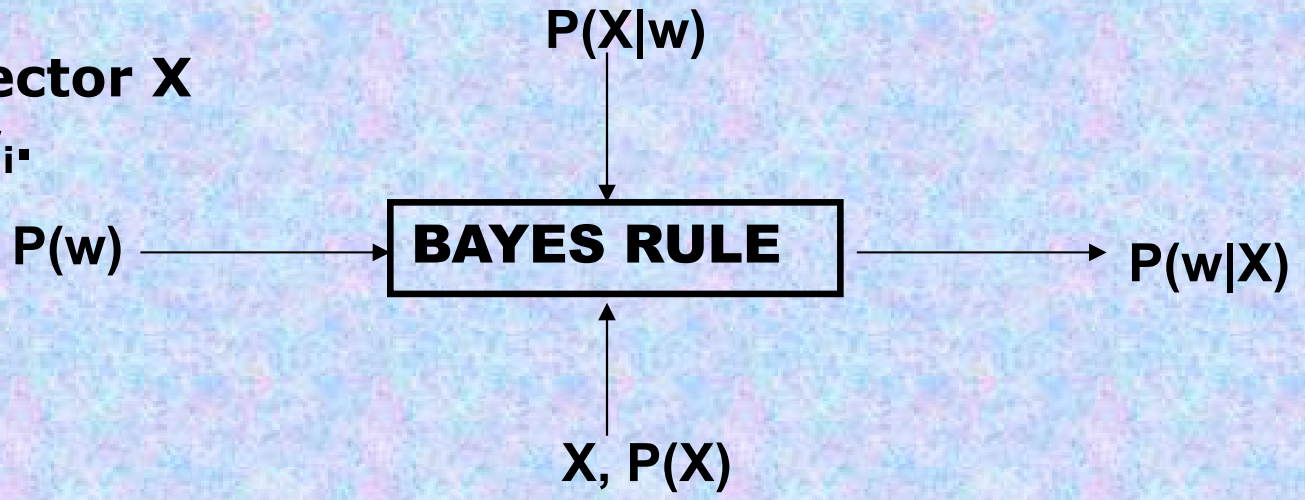
P(X) is the probability distribution for feature X in the entire population. Also called *unconditional density function (or evidence)*.

$P(w_i)$ 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* (or likelihood) 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.



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:

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

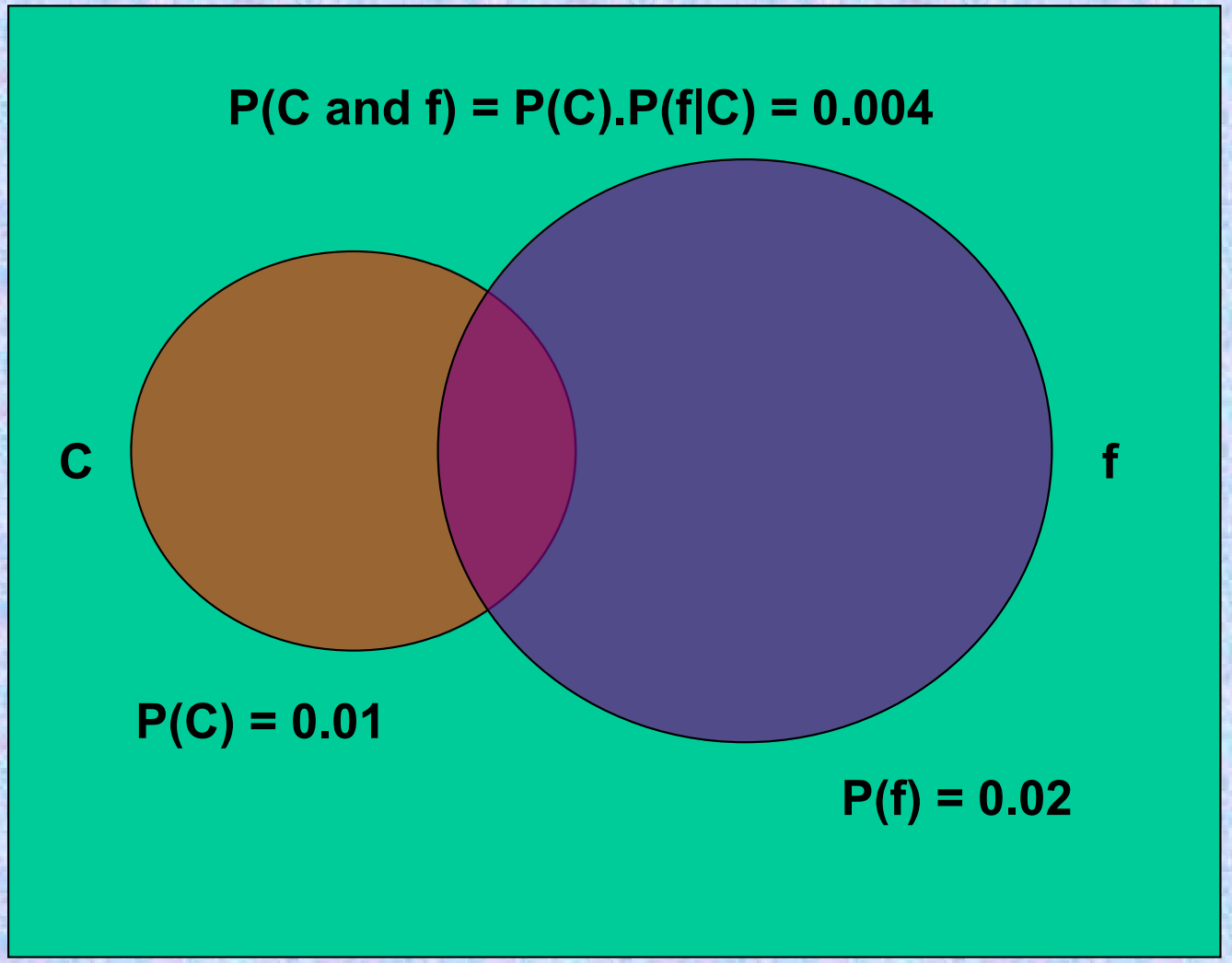
Not convinced that it works?
let us take an example with values to verify:

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:

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

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:

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

Errors in decision making:

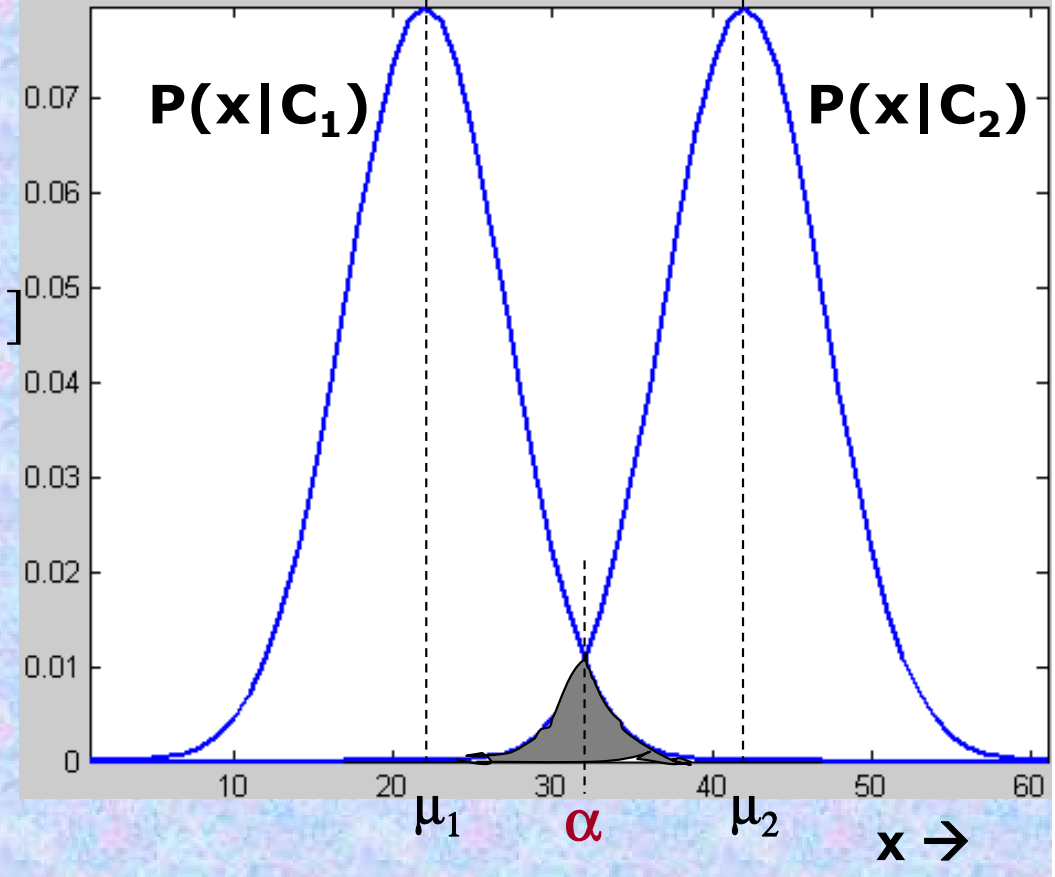
Let d = 1, C = 2,
$P(C_1) = P(C_2) = K$;

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

Bayes decision rule:

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

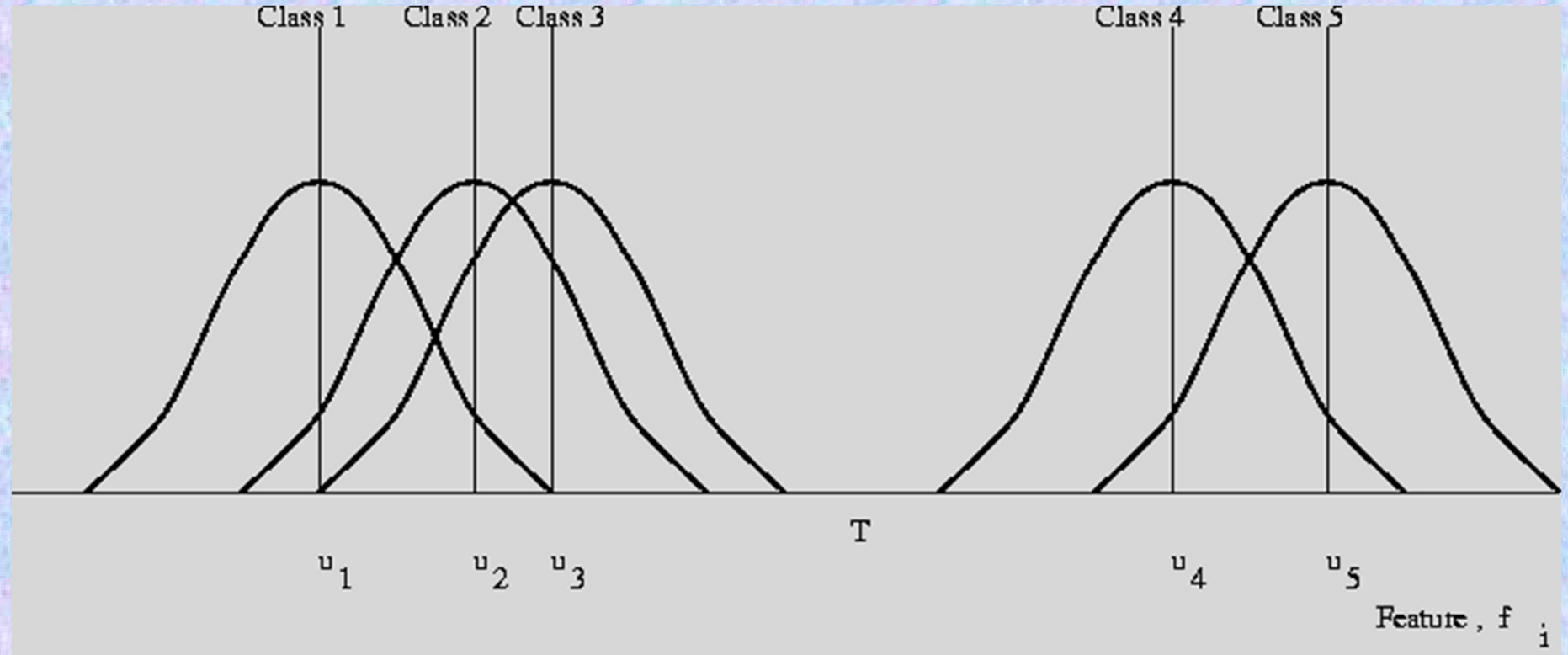
This gives α, and hence the two decision regions.



Classification error (the shaded region – minimum of the two curves):

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

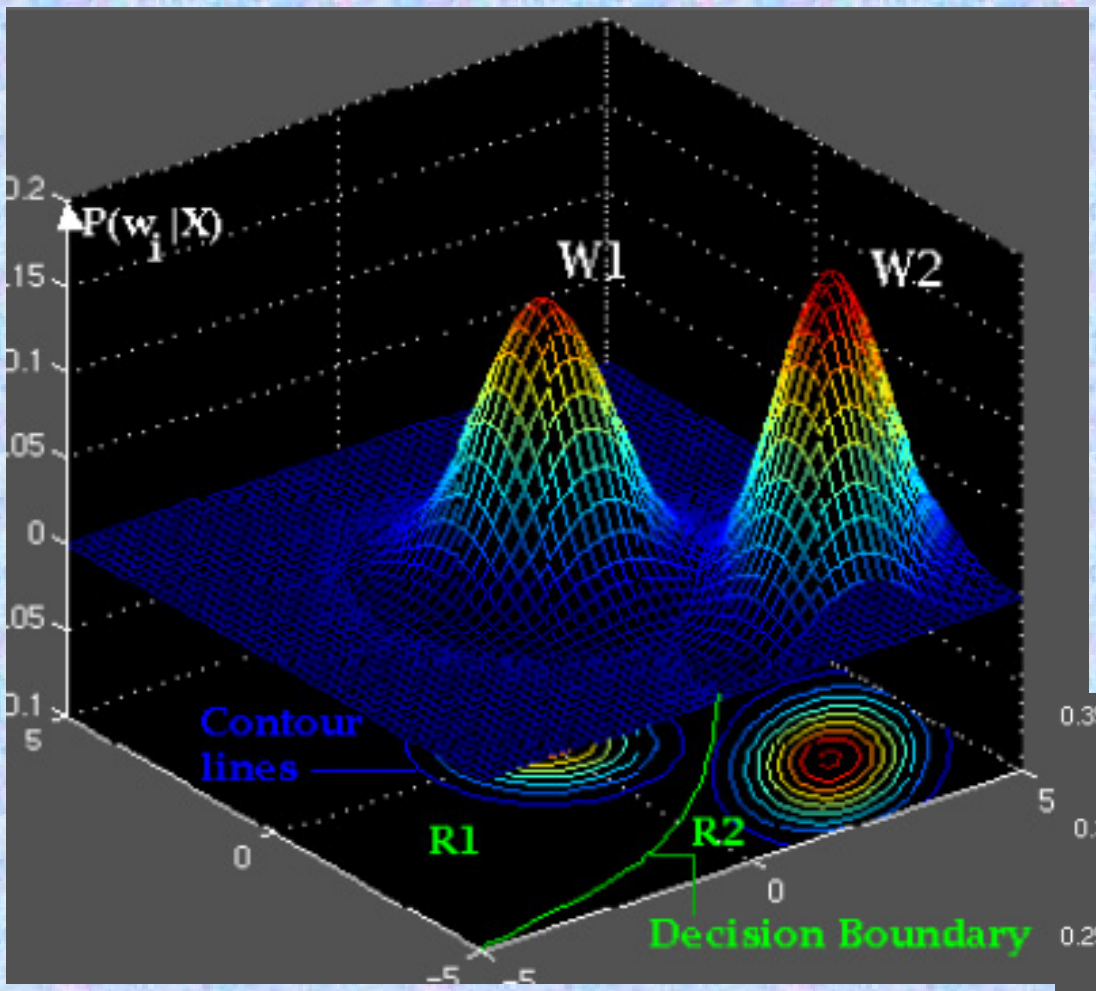
$$= P(C_2)\int_{-\infty}^{\alpha} P(\gamma \mid C_2)d\gamma + P(C_1)\int_{\alpha}^{-\infty} P(\gamma \mid C_1)d\gamma$$



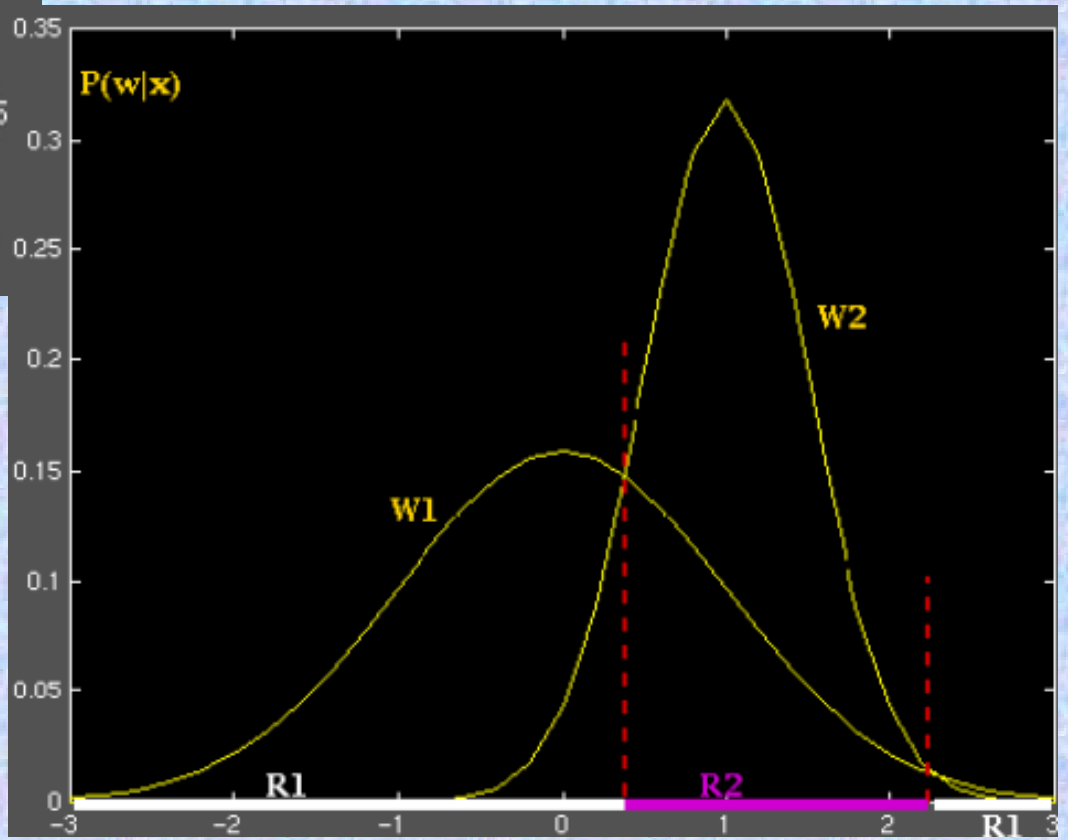
Normal distributions of feature measurement for a 5-class problem, equal variance.

A minimum distance (NN) supervised classifier

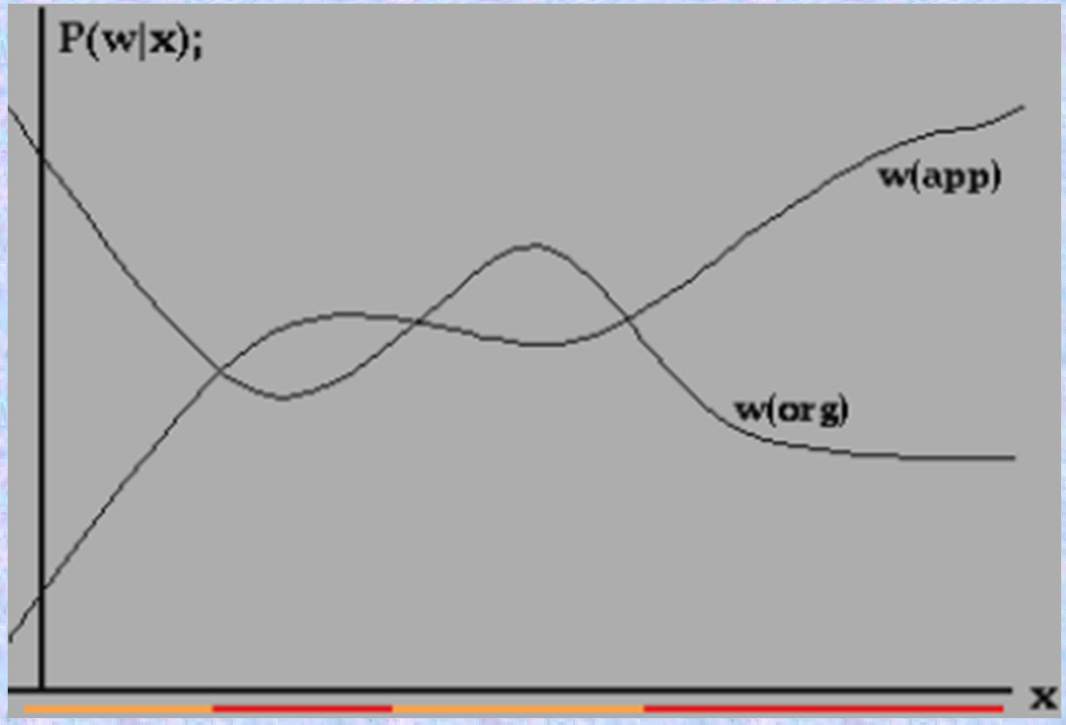
Rule: Assign X to R_i, where X is closest to μ_i.



An example of 2-D DRs:
R1 and R2; with a non-linear DB.



An example of 1-D DRs:
R1 and R2.



Decision based on arbitrary Posteriors, for an example: Apples Vs. Oranges.

Commonly used Discriminant functions based on Baye's decision rule:

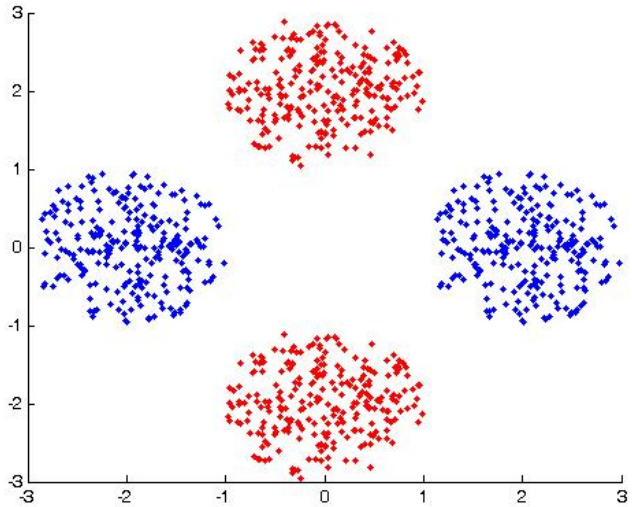
$$g_i(x) = P(w_i|x)$$

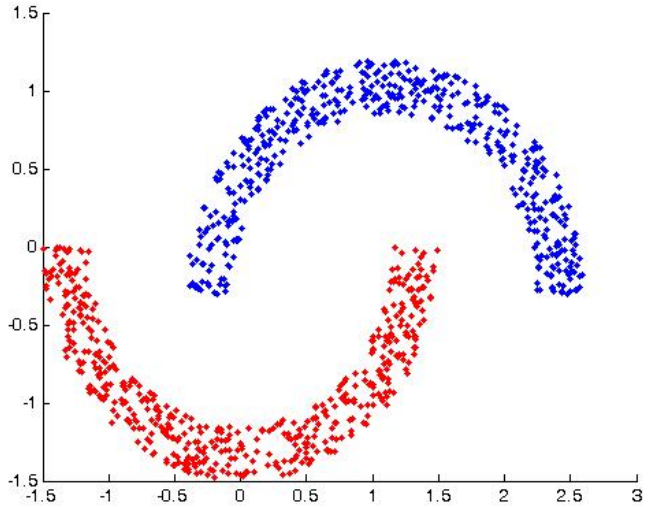
$$g_i(x) = \frac{p(x|w_i)P(w_i)}{\sum_{j=1}^{c} p(x|w_j)P(w_j)}$$

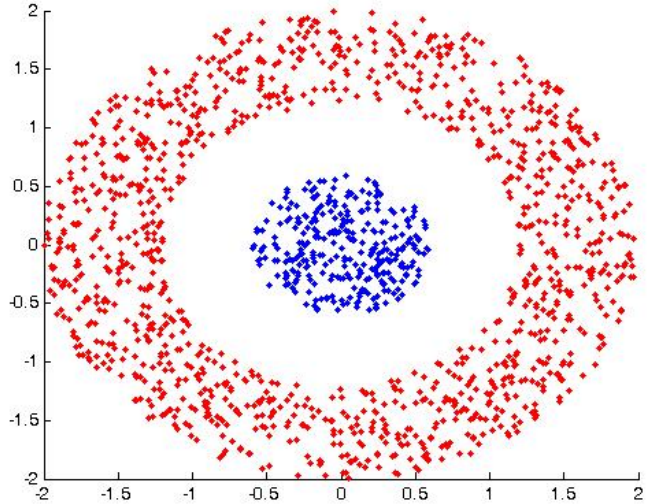
$$g_i(x) = p(x|w_i)P(w_i)$$

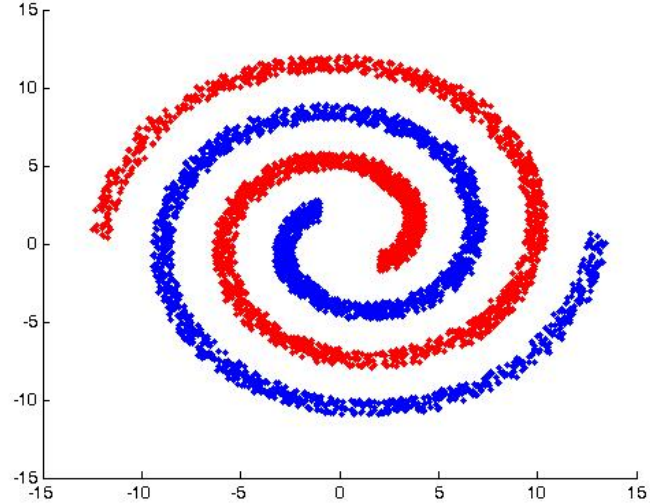
$$g_i(x) = \ln p(x|w_i) + \ln P(w_i)$$

Some examples of dense distribution of instances, with non-linear decision boundaries









K-means Clustering (unsupervised)

- **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 c the desired number of clusters, the k-means algorithm is:

```
Begin
    initialize n, c, μ1,μ2,…,μc(randomly
    selected)
        do
        1.classify n samples according
          to nearest μi
        2.recompute μi
        until no change in μi
    return μ1, μ2, …, μc
End
```

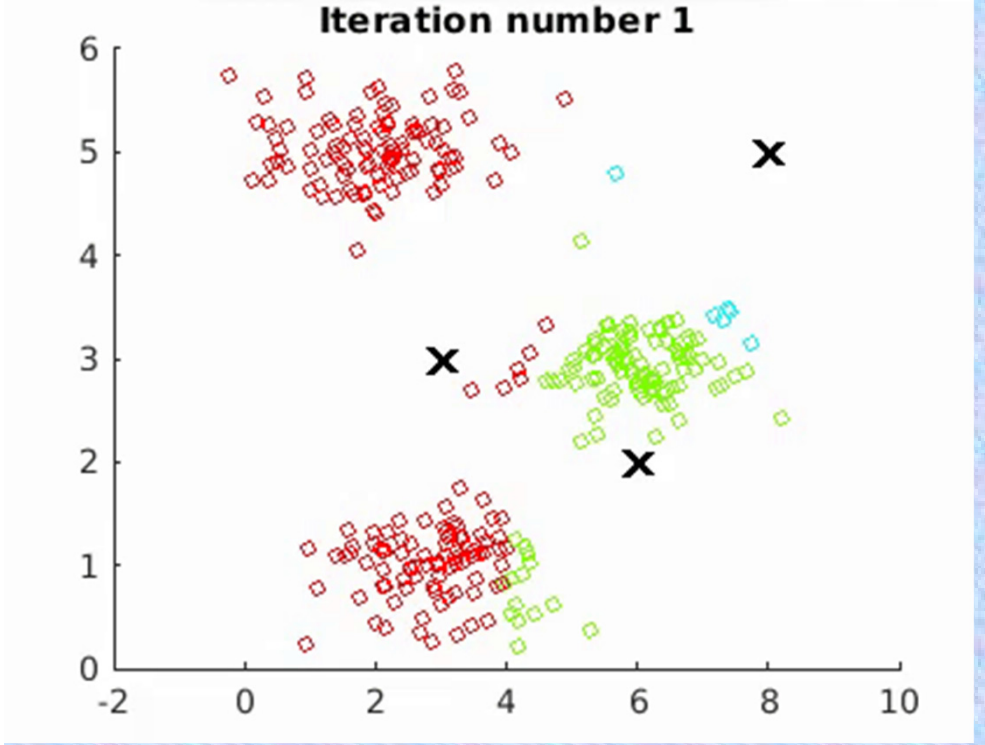
Classification Stage

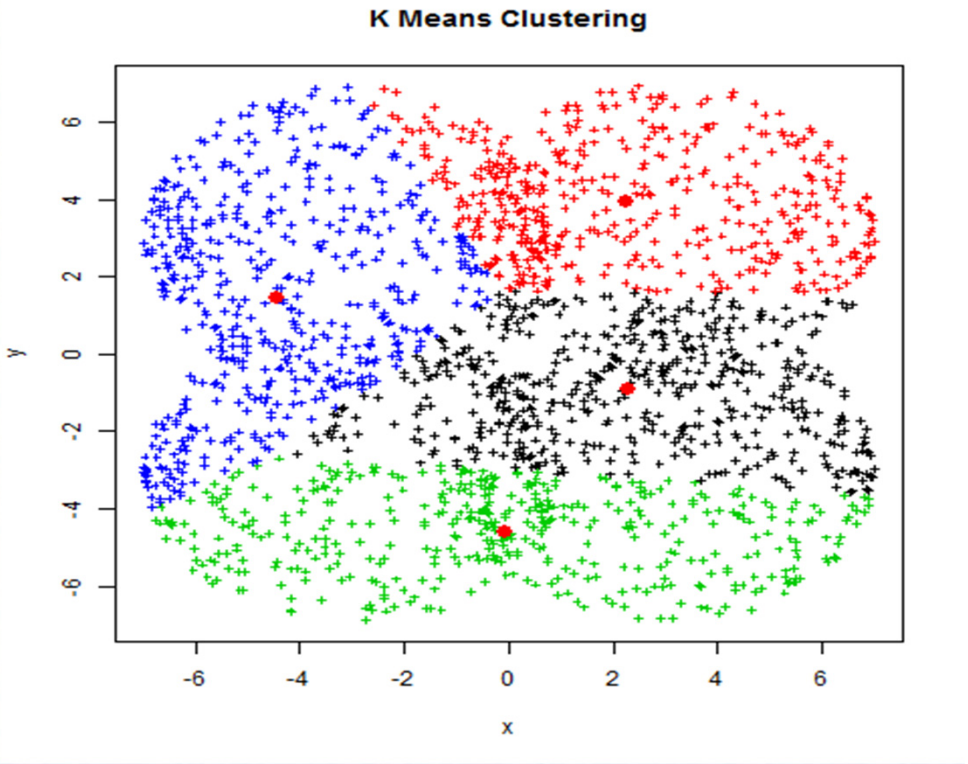
- The samples have to be assigned to clusters in order to minimize the cost function which is:

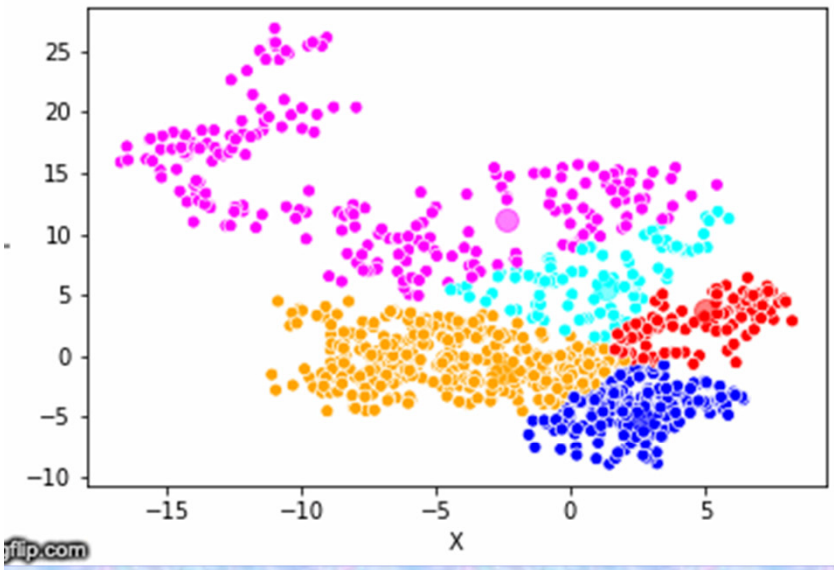
$$J = \sum_{i=1}^{c} J_i = \sum_{i=1}^{c} \left[\sum_{k, x_k \in G_i} \| x_k - \mu_i \|^2 \right]$$

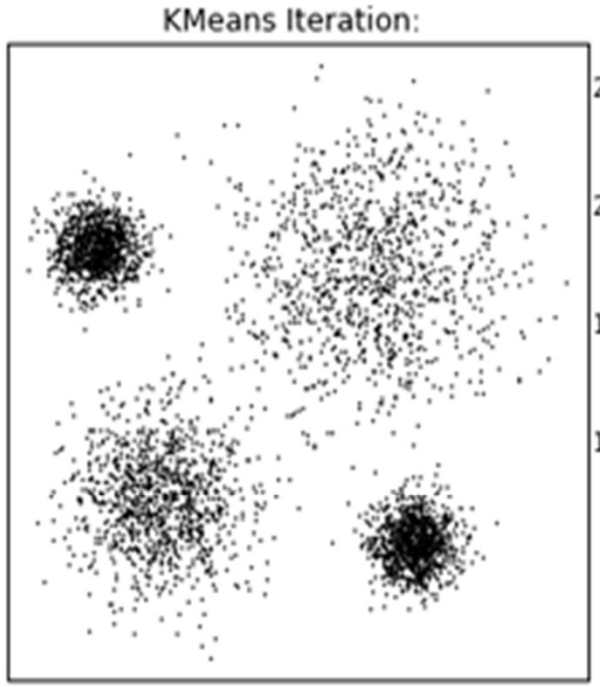
- This is the Euclidian Distance of the samples from its cluster center; for all clusters this sum should be minimum
- The classification of a point x_k is done by:

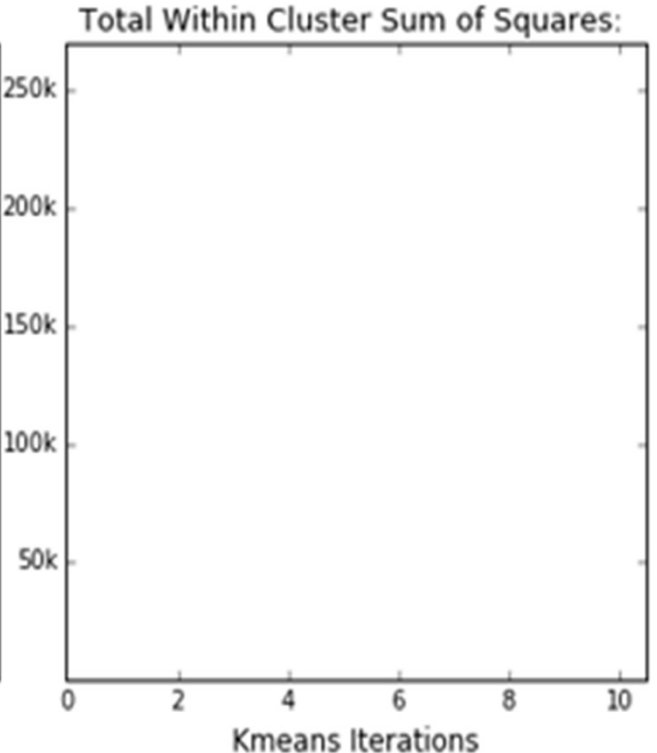
$$u_i = \begin{cases} 1 & \text{if } \| x_k - \mu_i \|^2 \geq \| x_k - \mu_j \|^2 \quad , \forall \, k \neq i \\ 0 & \text{otherwise} \end{cases}$$


Iteration number 1


K Means Clustering
x
y


X


KMeans Iteration:


Total Within Cluster Sum of Squares:
Kmeans Iterations

Re-computing the Means

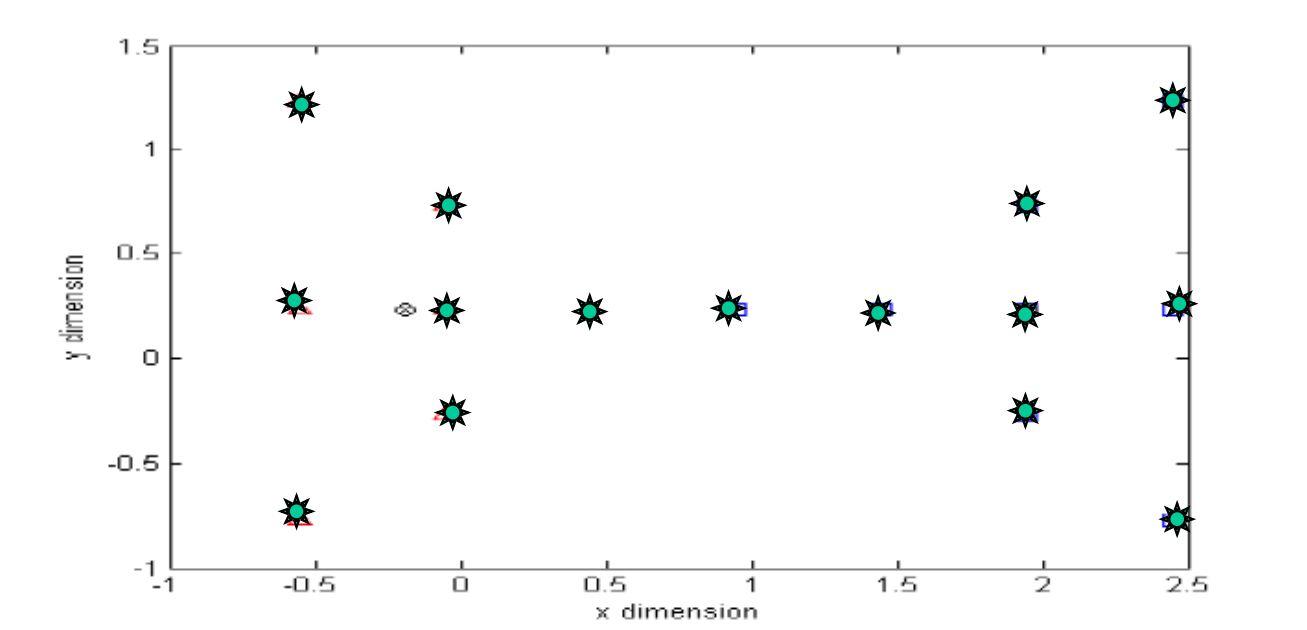
- The means are recomputed according to:

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

- 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”; Recurringly identical re-assignment; #iterations,….

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

Some necessary elements of

Probability theory and Statistics

The NORMAL DISTRIBUTION

The normal (or Gaussian) distribution, is a very commonly used (occurring) function in the fields of probability theory, and has wide applications in the fields of:

- **Pattern Recognition;**
- **Machine Learning;**
- **Artificial Neural Networks and Soft computing;**
- **Digital Signal (image, sound , video etc.) processing**
- **Vibrations, Graphics etc.**

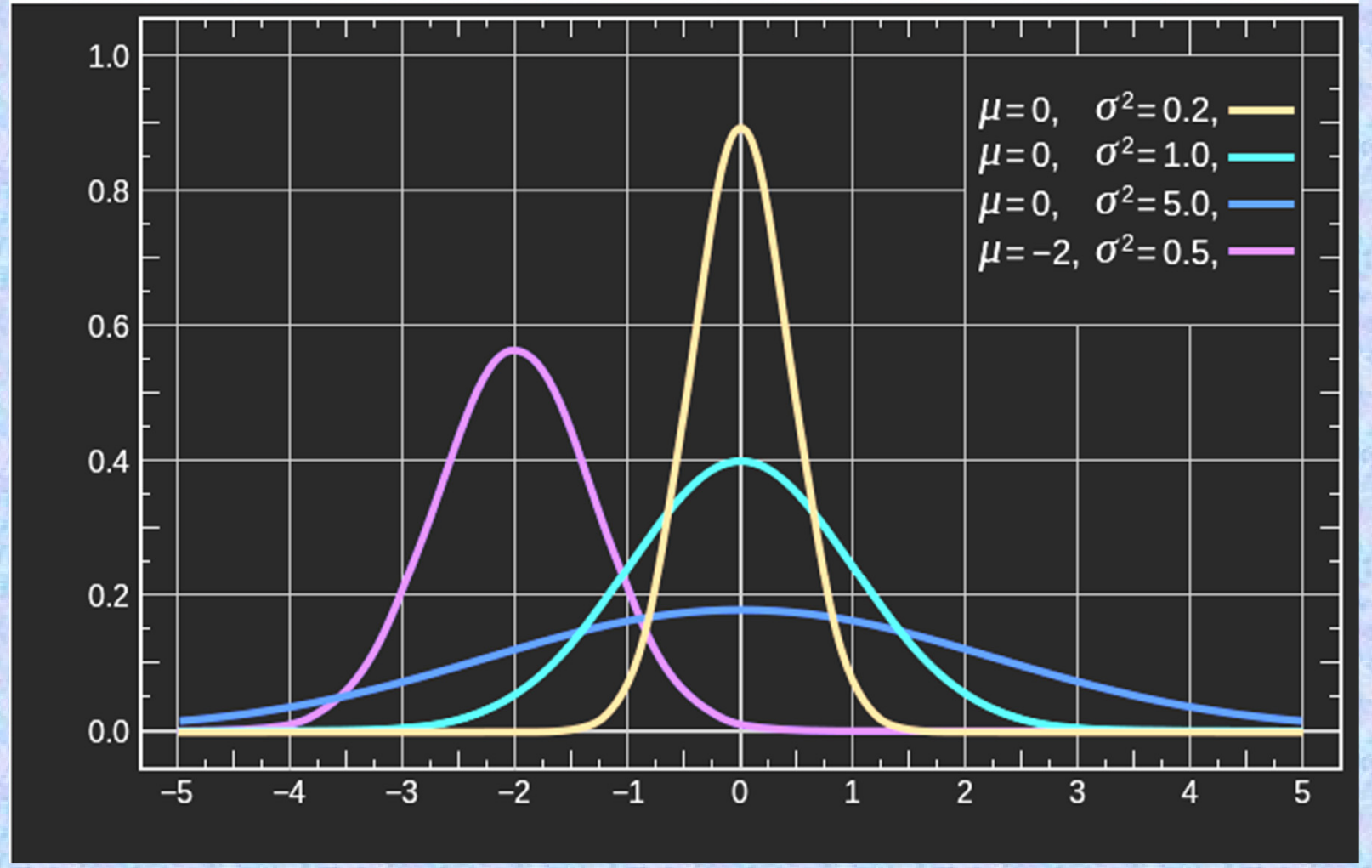
Its also called a BELL function/curve.

The formula for the normal distribution is:

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

The parameter μ is called the mean or expectation (or median or mode) of the distribution.

The parameter σ is the standard deviation;
and variance is thus σ^2.



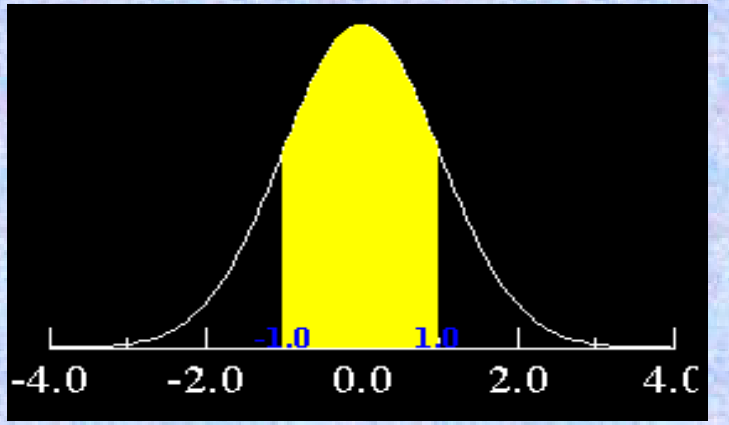
$$p(x) = \frac{1}{\sigma\sqrt{2\pi}} \exp[-\frac{1}{2}(\frac{x-\mu}{\sigma})^2]$$

https://en.wikipedia.org/wiki/File:Normal_Distribution_PDF.svg
(2013)

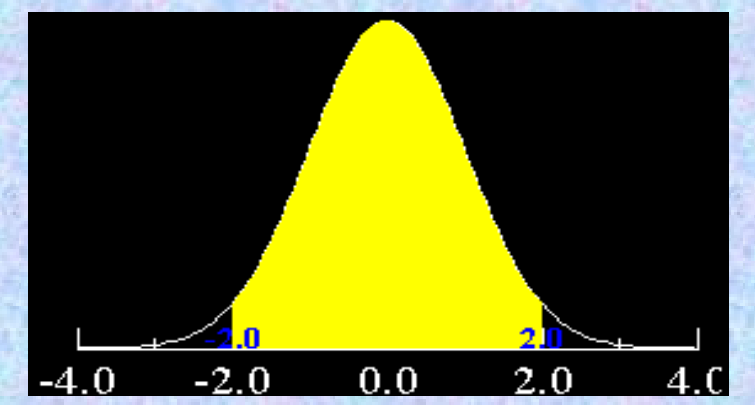
The 68 – 95 - 99.7% Rule:

All normal density curves satisfy the following property which is often referred to as the Empirical Rule:

- 68% of the observations fall within 1 standard deviation of the mean, that is, between $(\mu-\sigma)$ and $(\mu+\sigma)$

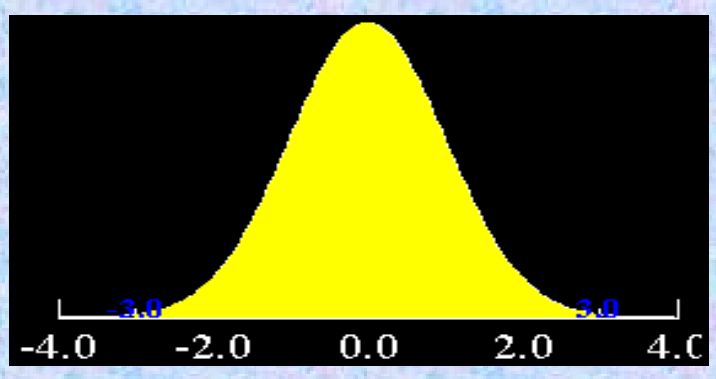


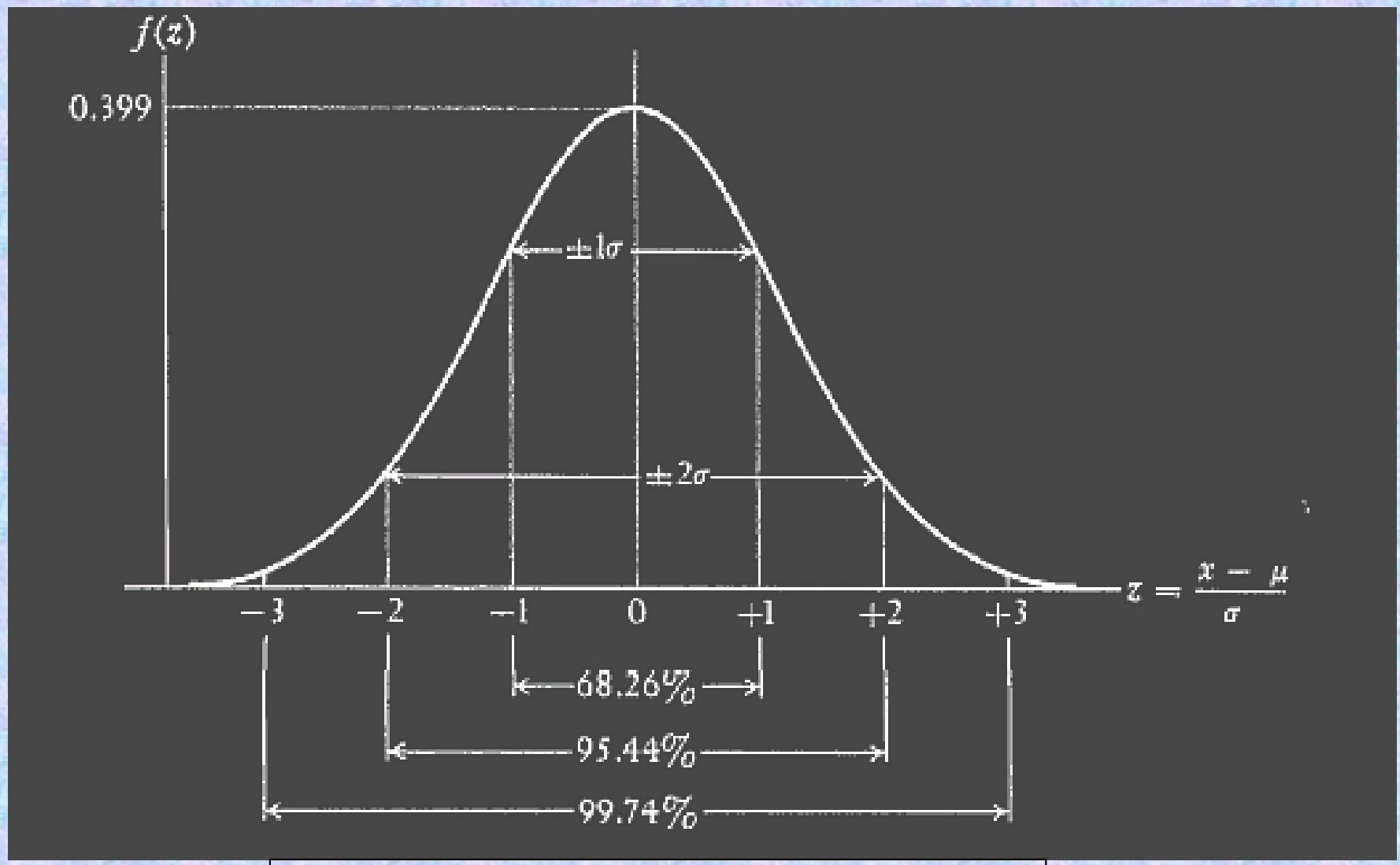
- 95% of the observations fall within 2 standard deviations of the mean, that is, between $(\mu-2\sigma)$ and $(\mu+2\sigma)$



- 99.7% of the observations fall within 3 standard deviations of the mean, that is, between

$(\mu-3\sigma)$ and $(\mu+3\sigma)$





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

The normal distribution $p(x)$, with any mean μ and any positive deviation σ, has the following properties:

- **It is symmetric around the mean (μ) of the distribution.**
- **It is unimodal: its first derivative is positive for $x < \mu$, negative for $x > \mu$, and zero only at $x = \mu$.**
- **It has two inflection points (where the second derivative of f is zero and changes sign), located one standard deviation away from the mean, $x = \mu - \sigma$ and $x = \mu + \sigma$.**
- **It is log-concave.**
- **It is infinitely differentiable, indeed supersmooth of order 2.**

Also, the standard normal distribution p (with $\mu = 0$ and $\sigma = 1$) also has the following properties:

- **Its first derivative $p'(x)$ is: $-x.p(x)$.**
- **Its second derivative $p''(x)$ is: $(x^2 - 1).p(x)$**
- **More generally, its n-th derivative :**

$p^{(n)}(x)$ **is:** $(-1)^n H_n(x)p(x)$,

where, H_n is the Hermite polynomial of order n.

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 k^{th} 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) are the expected values of: x^2, x^3,…

The k^{th} 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.

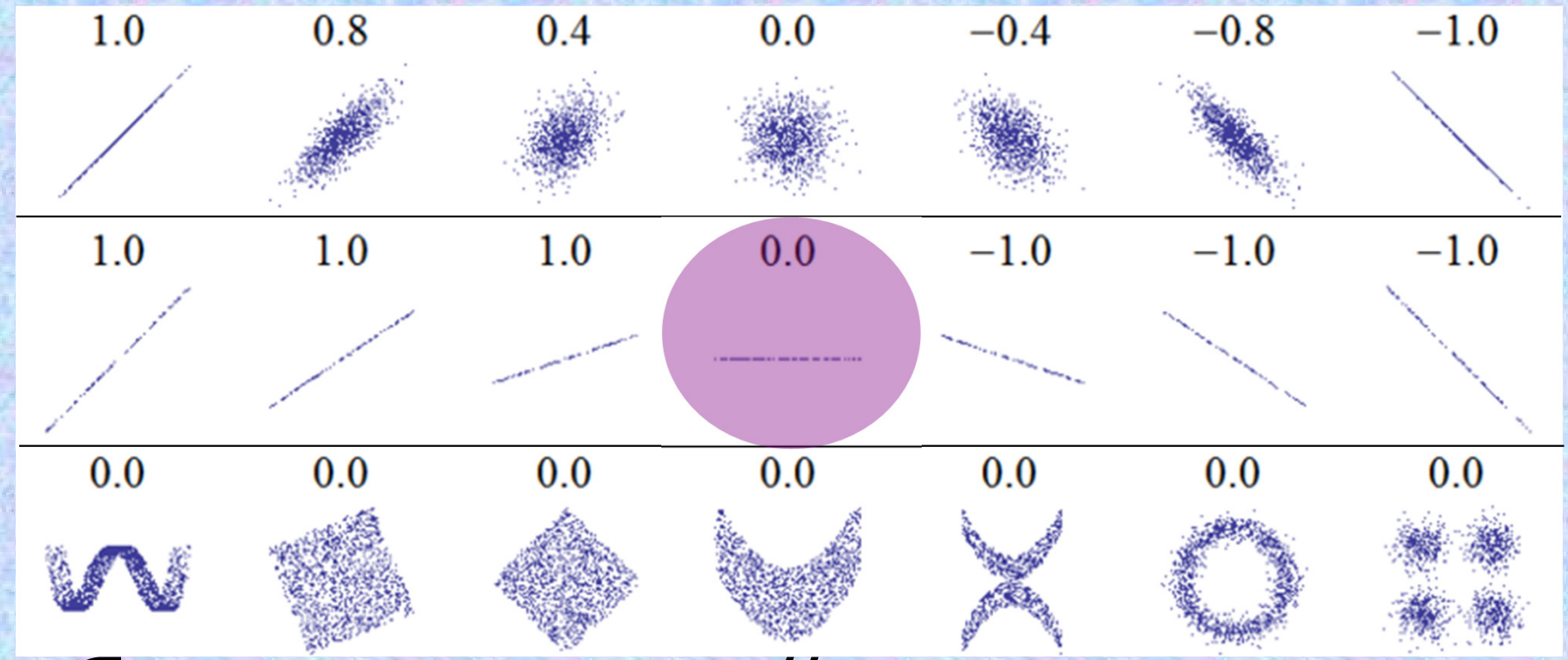
Correlation between x and y: $\rho_{xy} = \frac{\sigma_{xy}}{\sigma_x \sigma_y} = E[(\frac{x-\mu_x}{\sigma_x})(\frac{y-\mu_y}{\sigma_y})]$

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

For Z = ax + by ;

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

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



$$\rho_{xy} = \frac{\sigma_{xy}}{\sigma_x \sigma_y} = E[(\frac{x-\mu_x}{\sigma_x})(\frac{y-\mu_y}{\sigma_y})]$$

$$\rho_{X,Y} = \frac{E(XY) - E(X)E(Y)}{\sqrt{E(X^2) - E^2(X)}\sqrt{E(Y^2) - E^2(Y)}}$$

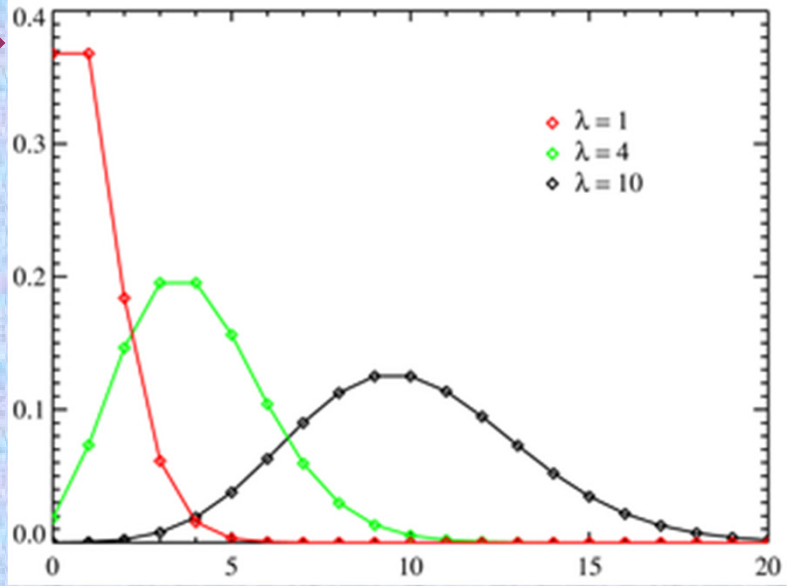
The correlation coefficient can also be viewed as the cosine of the angle between the two vectors ($\mathbb{R}^D$) of samples drawn from the two random variables.

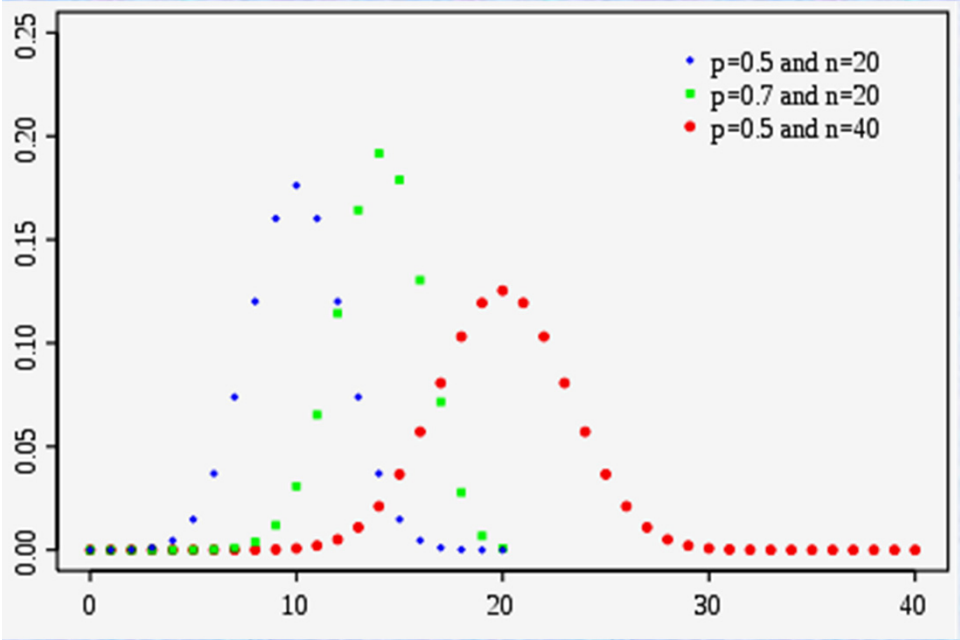
This method only works with centered data, i.e., data which have been shifted by the sample mean so as to have an average of zero.

Other PDFs:

Poisson →

$$P(x) = \frac{\lambda^x}{x!} e^{-\lambda;} \quad \lambda > 0$$





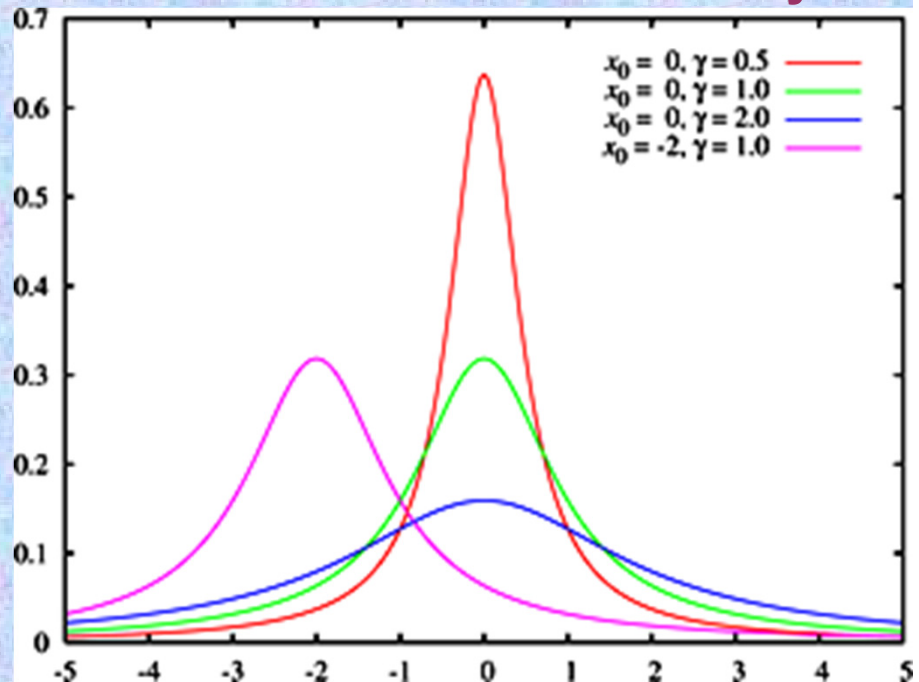
$$\Pr(K = k) = \binom{n}{k} p^k (1-p)^{n-k}$$

← Binomial

Cauchy

$$f(x; x_0, \gamma) = \frac{1}{\pi\gamma \left[1 + \left(\frac{x - x_0}{\gamma}\right)^2\right]}$$

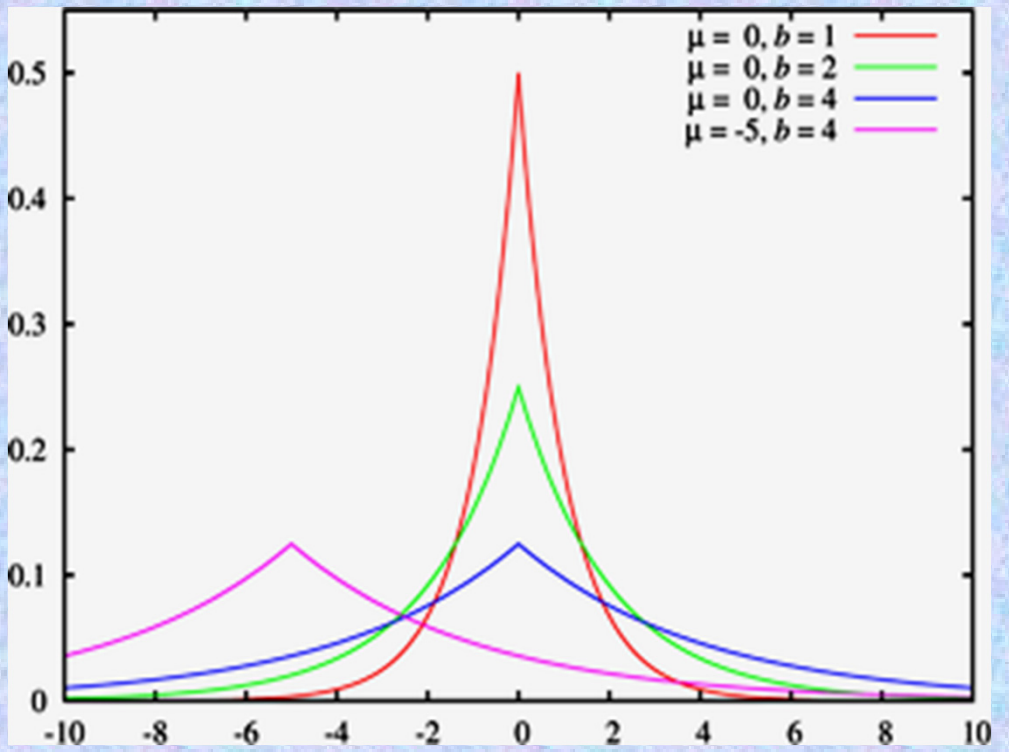
$$= \frac{1}{\pi} \left[\frac{\gamma}{(x - x_0)^2 + \gamma^2}\right]$$



LAPLACE:

$$f(x|\mu,b) = \frac{1}{2b}\exp\left(-\frac{|x-\mu|}{b}\right)$$

$$= \frac{1}{2b}\begin{cases}\exp\left(-\frac{\mu-x}{b}\right) & \text{if } x < \mu \\ \exp\left(-\frac{x-\mu}{b}\right) & \text{if } x \geq \mu\end{cases}$$



Double Exponential Density:

$$P(x) = \frac{1}{2b}e^{-\left|x-a/b\right|};$$

Read about:

- **Central Limit Theorem**
- **Uniform Distribution**
- **Geometric Distribution**
- **Quantile-Quantile (QQ) Plot**
- **Probability-Probability (P-P) Plot**

PROB. & STAT. Contd.

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

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 (MLE)

The ML estimate (MLE) 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:

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

Also read about MAP estimate – Baye's is an example.

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 k^{th} moment is defined as:

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

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

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

Mean vector:

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

Covariance matrix (symmetric):

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

d-dimensional normal density is:

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

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

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

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

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

Special case, d = 2; where X = (x y)^T; **Then:** $\mu = \begin{pmatrix} \mu_x \\ \mu_y \end{pmatrix}$

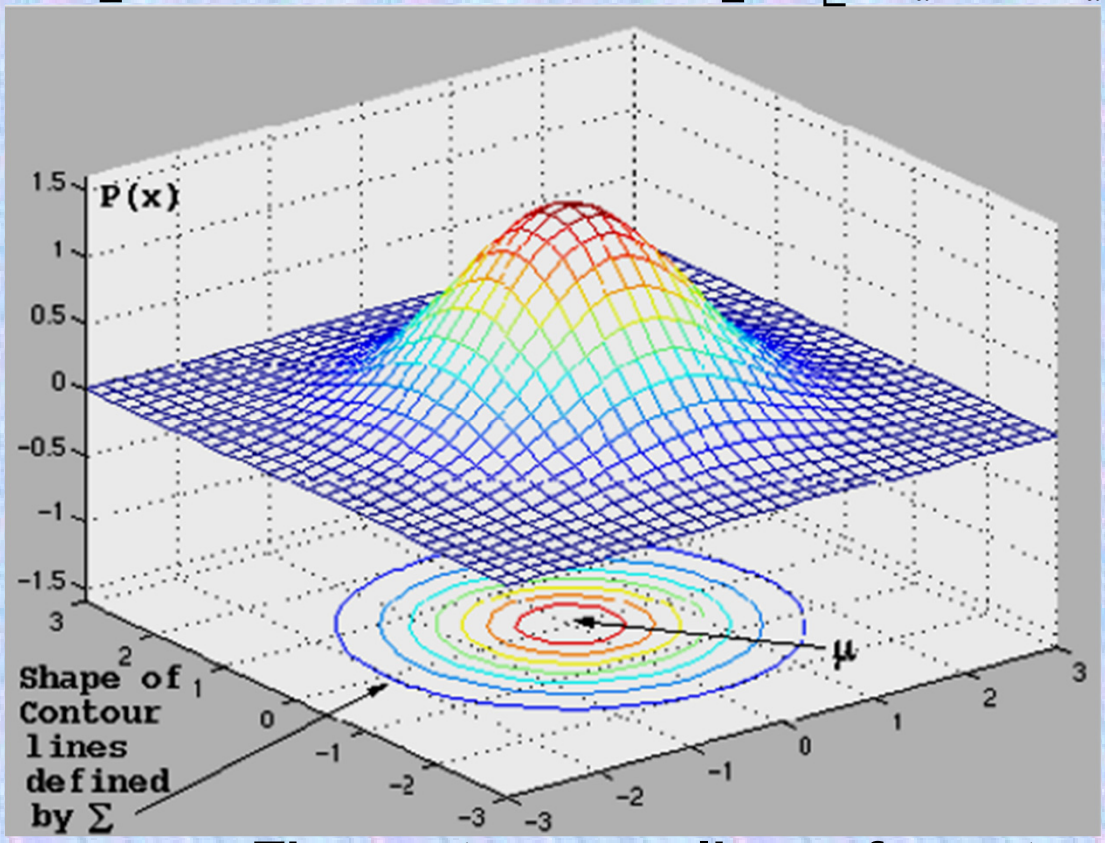
and
$$\Sigma = \begin{pmatrix} \sigma_x^2 & \sigma_{xy} \\ \sigma_{xy} & \sigma_y^2 \end{pmatrix} = \begin{pmatrix} \sigma_x^2 & \rho_{xy}\sigma_x\sigma_y \\ \rho_{xy}\sigma_x\sigma_y & \sigma_y^2 \end{pmatrix}$$

Can you now obtain this, as given earlier:

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

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

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



Contours have constant density of the distant term (d=2):

$$d(X) = (X-\mu)^T \Sigma_D^{-1} (X-\mu);$$

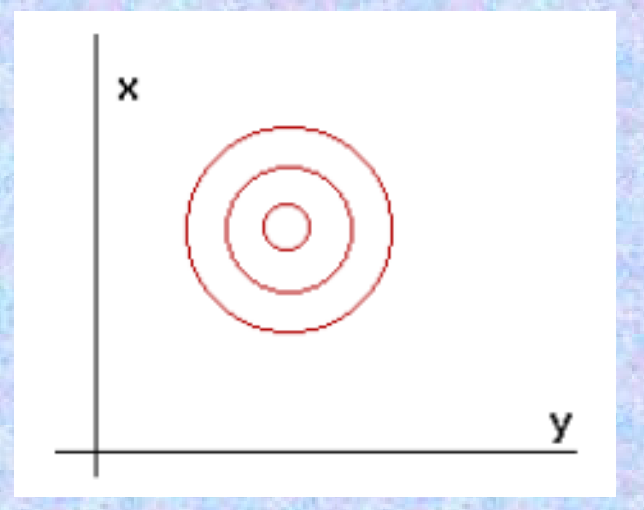
The contours are lines of constant Mahalanobis distance (determined by the matrix Σ), and are quadratic functions.

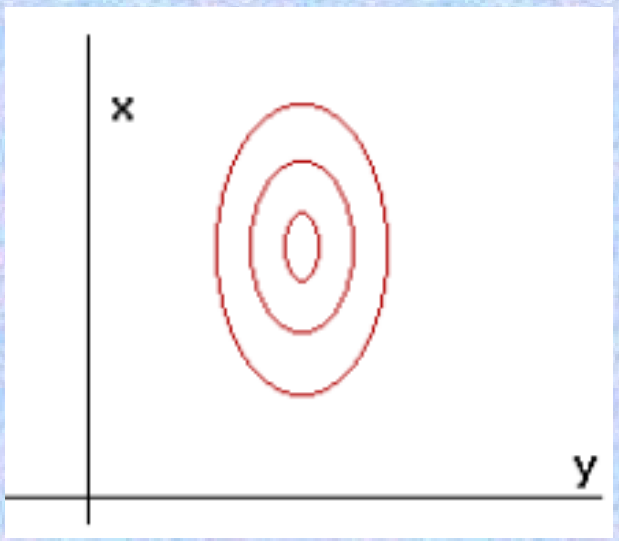
The contours of constant density may also be hyper-ellipsoids (non-diagonal Σ) of constant Mahalanobis distance to μ.

Diagonal covariance;

$$\sigma_x = \sigma_y;$$

$$\rho_{xy} = 0;$$





Diagonal covariance;

$$\sigma_x > \sigma_y;$$

$$\rho_{xy} = 0;$$

Remember, asymmetric and oriented Gaussians

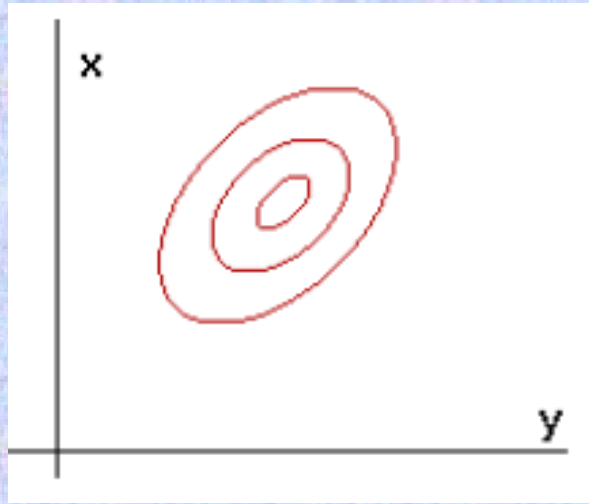
Non-Diagonal covariance;

$$\sigma_x = \sigma_y;$$

$$\rho_{xy} < 0;$$

$$\sigma_x = \sigma_y;$$

$$\rho_{xy} > 0;$$



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*:

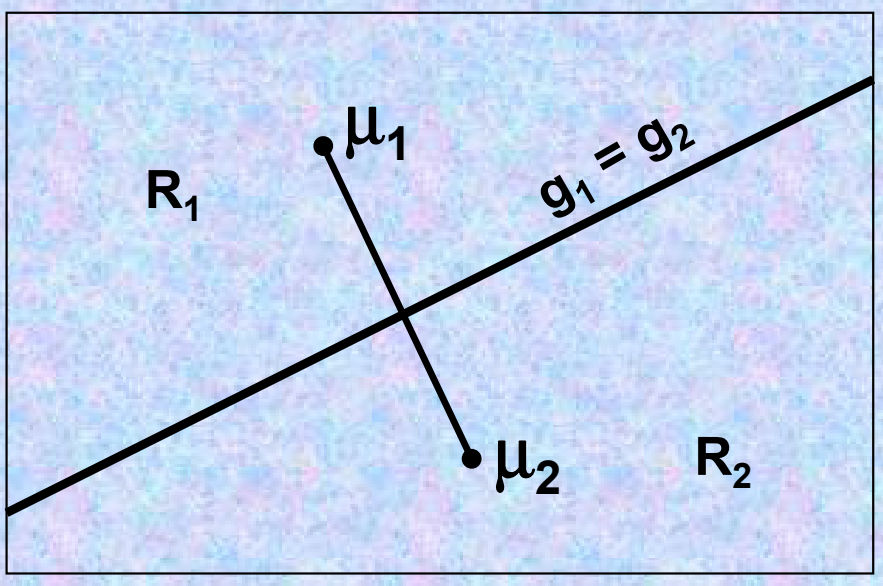
Assign X to class C_m (or region m), where: $g_m(X) > g_i(X), \forall i, i \neq m.$

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

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

Minimum distance (also ND/NN) classifier:

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



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

This does not take into account class PDFs and priors.

Remember Baye's: $P(w_i \mid \vec{X}) = \frac{P(\vec{X} \mid w_i)P(w_i)}{P(\vec{X})}$

Consider discriminant function as: $g_i(x) = \ln p(x|w_i) + \ln P(w_i)$

and class-conditional Prob. as:

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

$g_i(x) =$

Many cases arise, due to the varying nature of Σ:

- **Diagonal (equal or unequal elements);**
- **Off-diagonal (+ve or –ve).**

Let the discrimination function for the i^{th} class be:

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

Remember, multivariate Gaussian density?

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

Define:

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

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

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:

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

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

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

For a given X, some $G_m(X)$ is largest where $(d_m)^2$ is the smallest, for a class i = m (assign X to class m, based on NN Rule) .

***Simplest case*: $\Sigma = I$, the criteria becomes the Euclidean distance norm (and hence the NN classifier).**

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

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

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

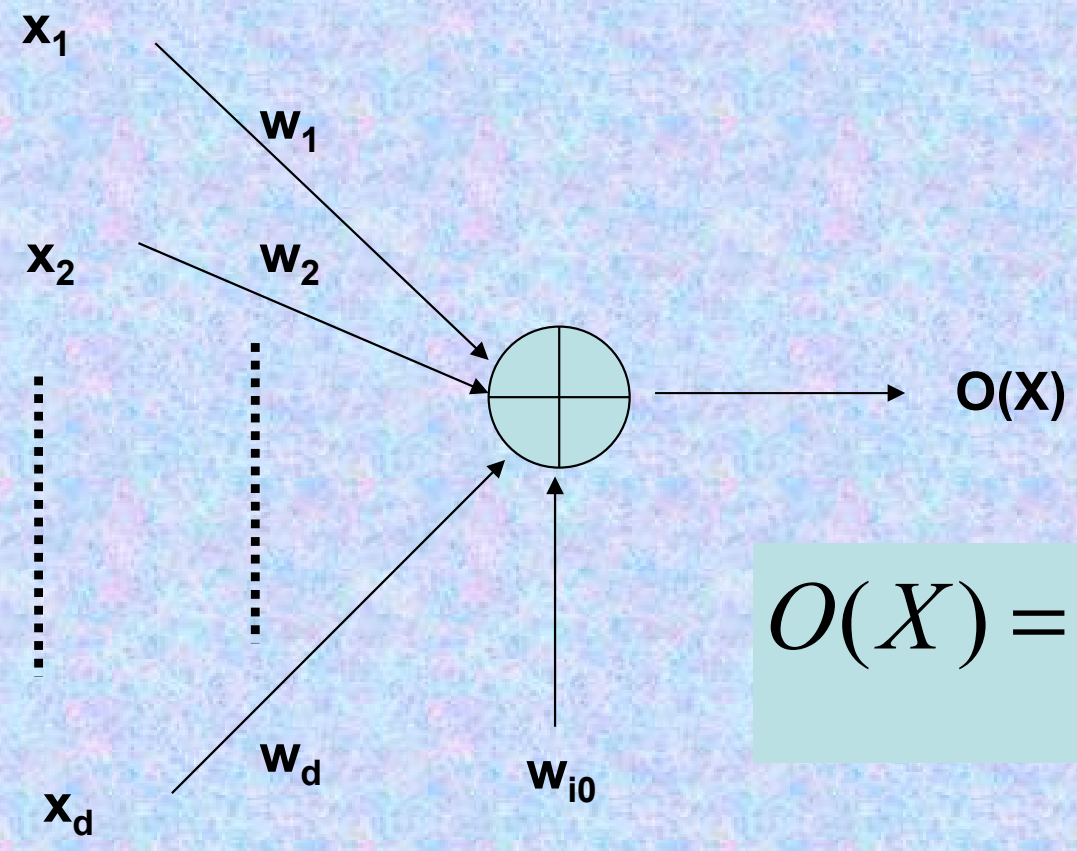
$$= \omega_i^T X + \omega_{i0}$$

Neglecting the class-invariant term.

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

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):

$$G = MX - Y + C$$

The decision region <u>boundaries</u> are determined by solving :

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

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

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

<u>Generalized results (Gaussian case) of a discriminant function:</u>

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

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

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:

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

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

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

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

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

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

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

Neglecting the constant term

Simpler case: $\Sigma_i = \sigma^2 I$, **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.
More on this later on…..

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:

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

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:

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

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

$$G_i(X) = (\Sigma^{-1}\mu_i)^T X - \frac{1}{2}\mu_i^T \Sigma^{-1} \mu_i$$ **as $\Sigma_i = \Sigma$,** and **are 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)

Beyond this, if a diagonal Σ is class-dependent or off-diagonal terms are non-zero, we get non-linear DFs, DRs or DBs.

The discriminant function (DF) 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: $\omega^T(\vec{X} - \vec{X}_d) = 0; \Rightarrow \omega^T \vec{X} - d = 0$

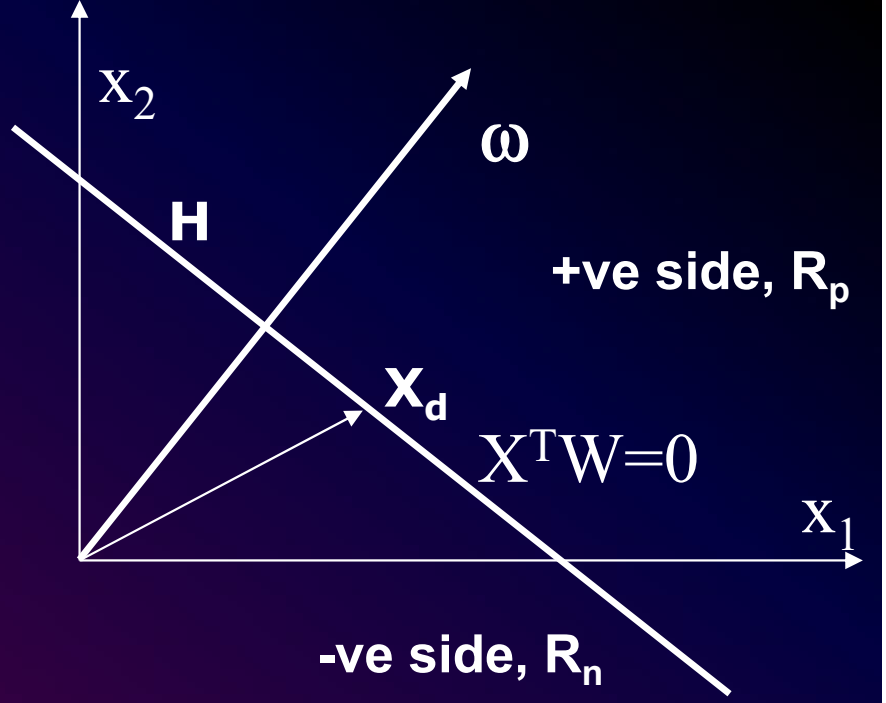
represents a plane H passing through any point (position vector) X_d.

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:

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

A relook at,
Linear Discriminant Function g(X):

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

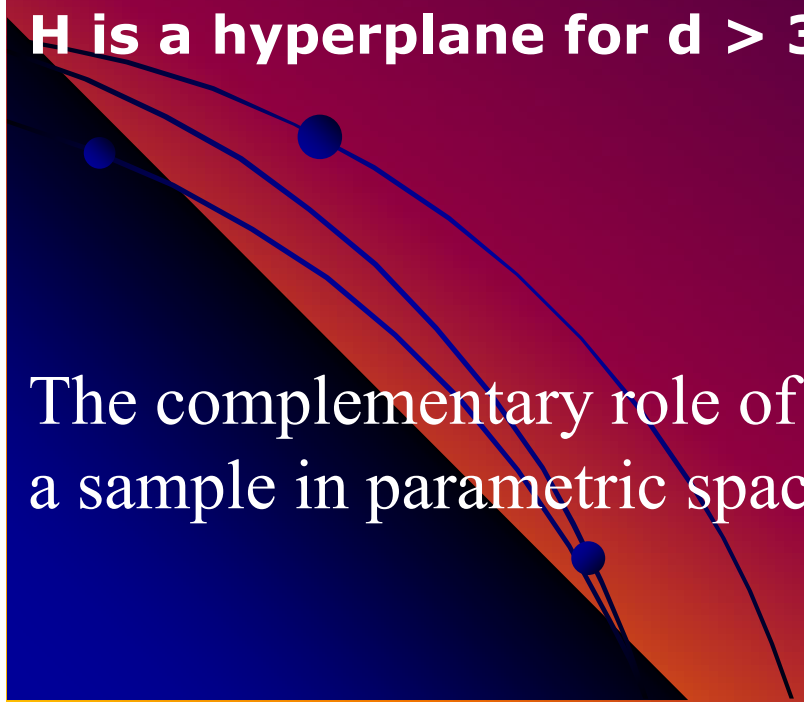


Pattern/feature Space

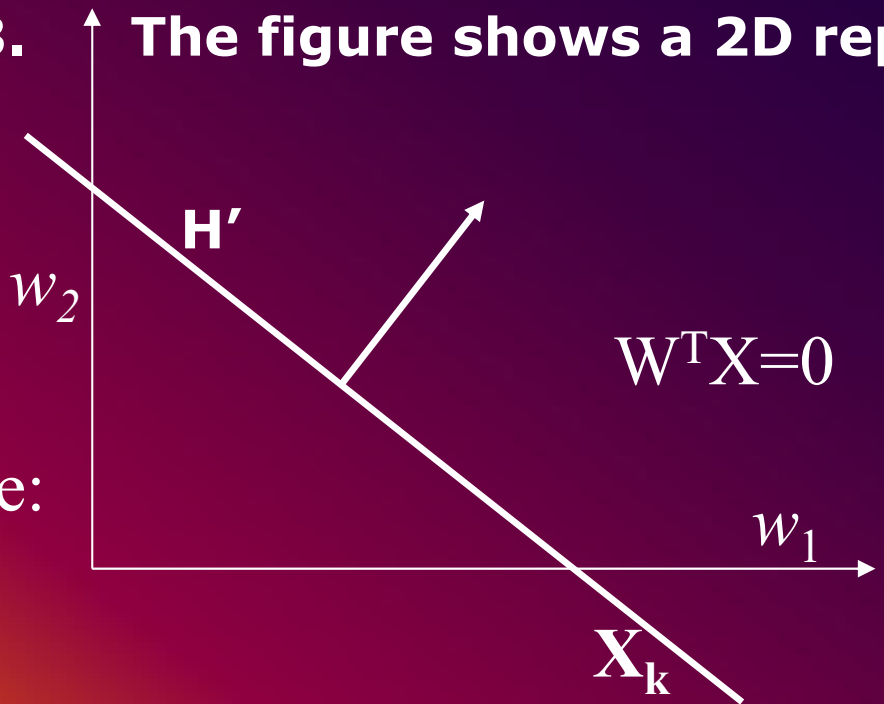
Orientation of H is determined by ω.

Location of H is determined by d.

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

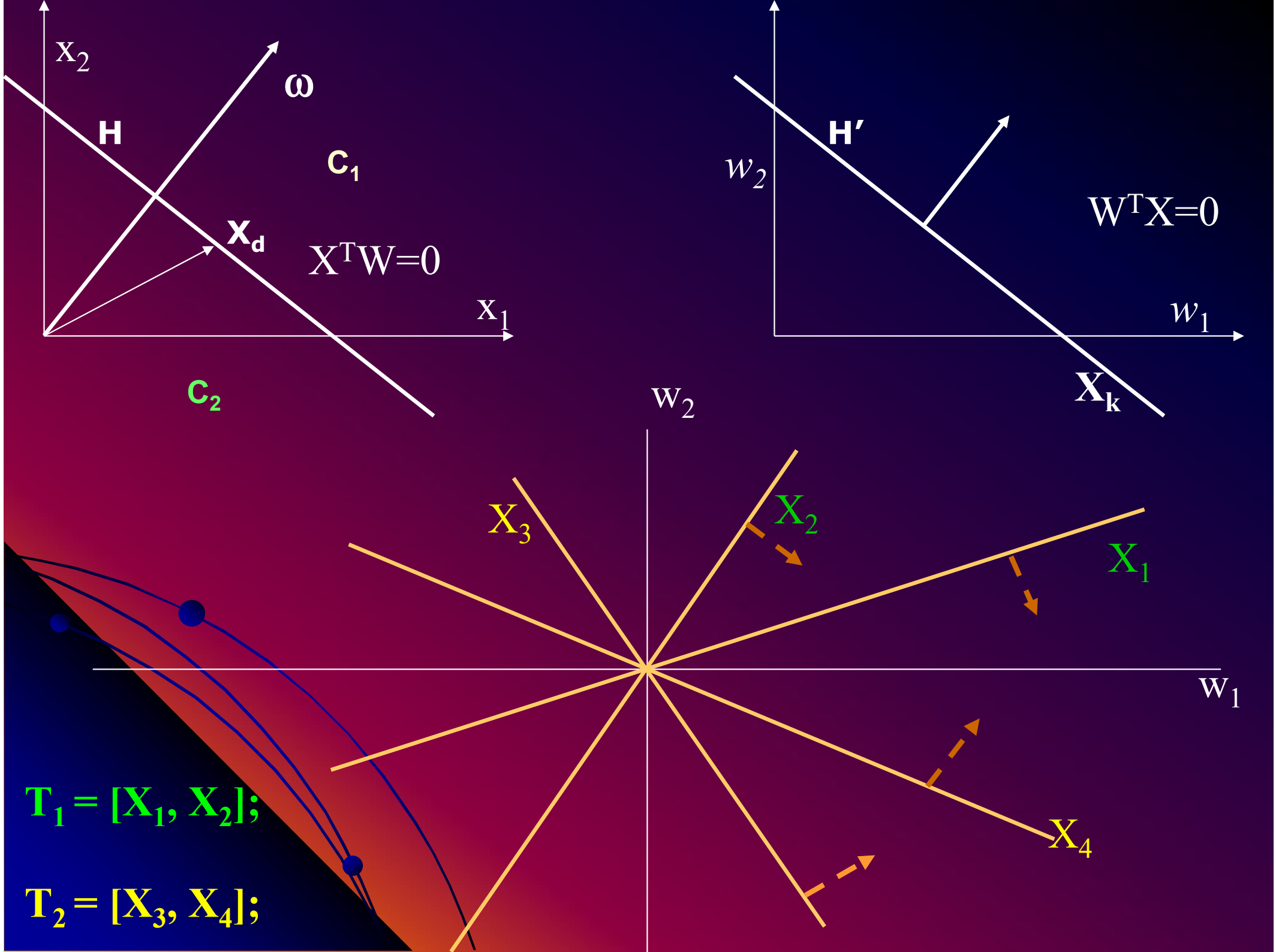


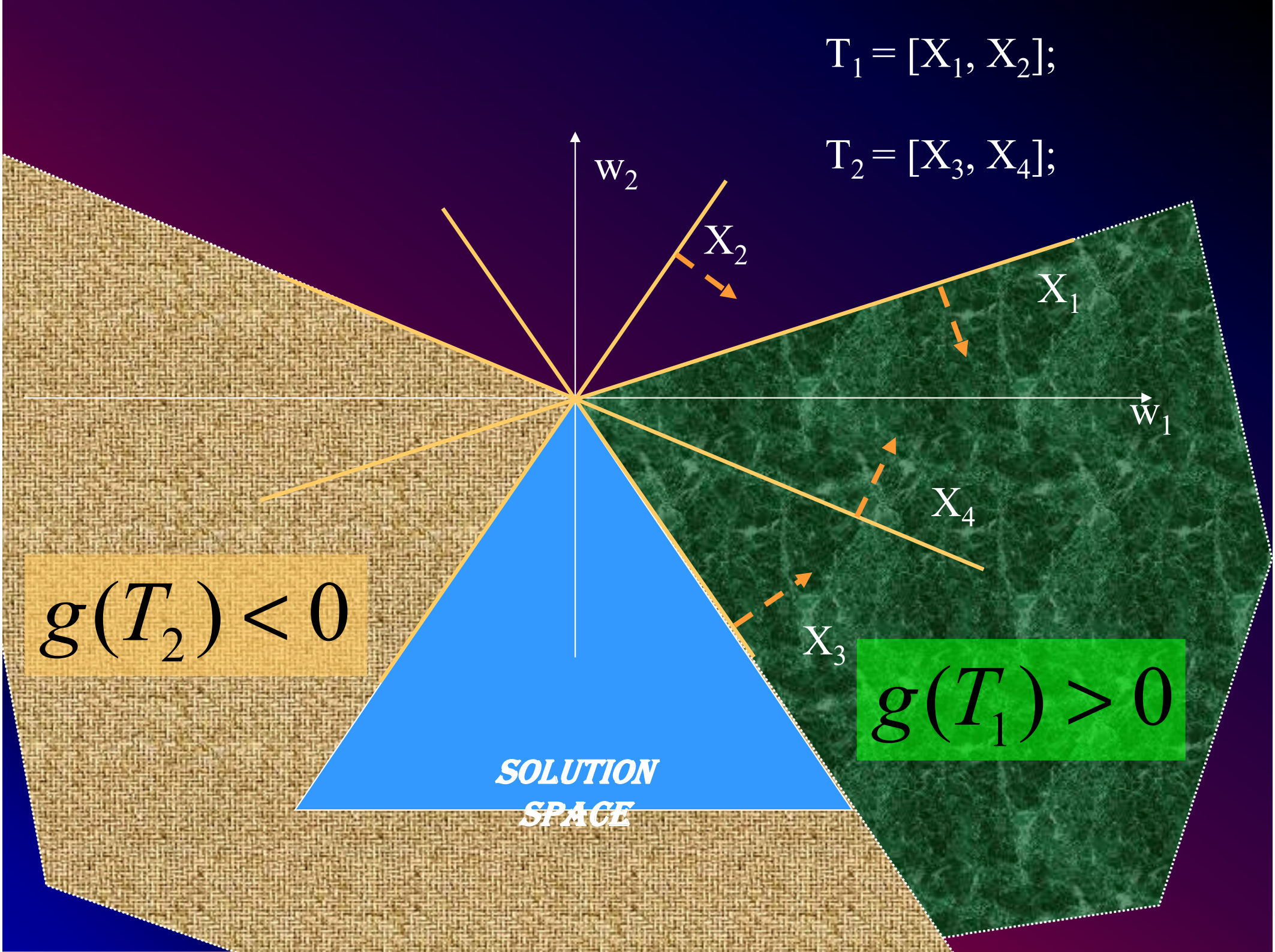
The complementary role of a sample in parametric space:



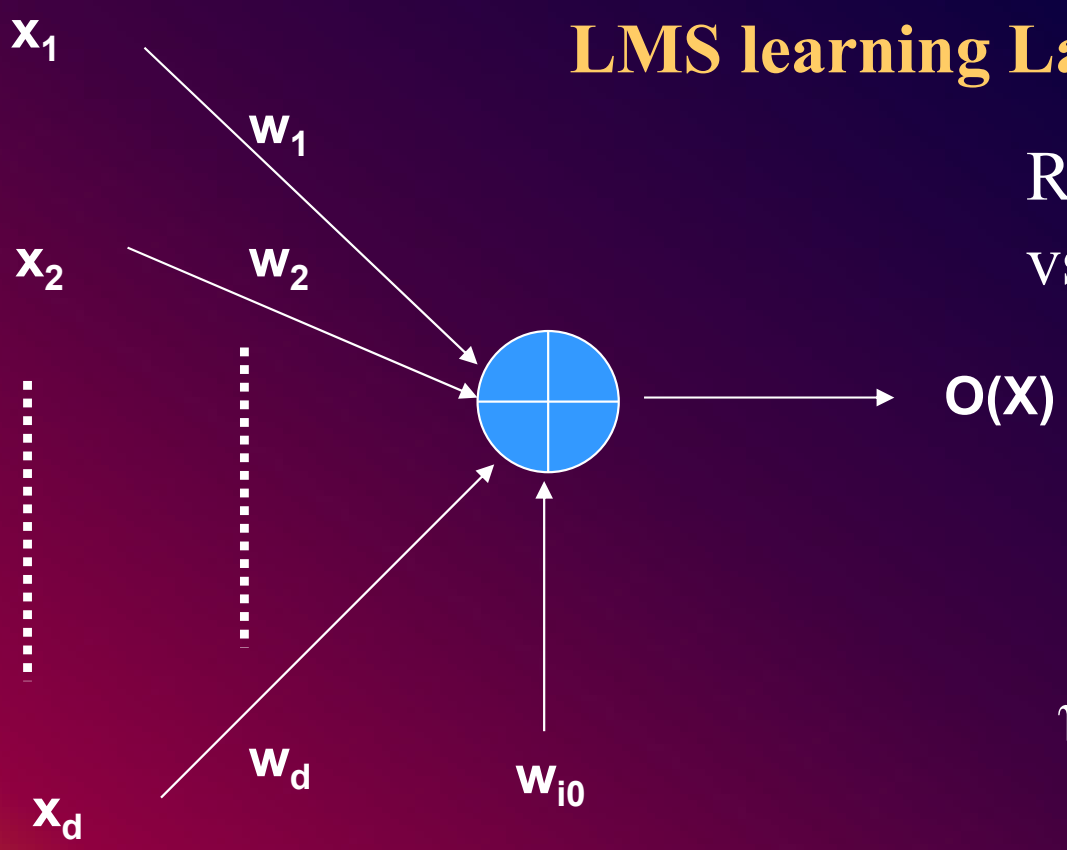
WTX=0

Weight Space


x2
ω
H
c1
xd
XTW=0
x1
c2
H′
w2
WTX=0
w1
Xk
w2
X3
X2
X1
w1
X4
T1 = [X1, X2];
T2 = [X3, X4];


$T_1 = [X_1, X_2]$;
$T_2 = [X_3, X_4]$;
w_2
X_2
X_1
w_1
X_4
X_3
$g(T_2) < 0$
$g(T_1) > 0$
SOLUTION SPACE

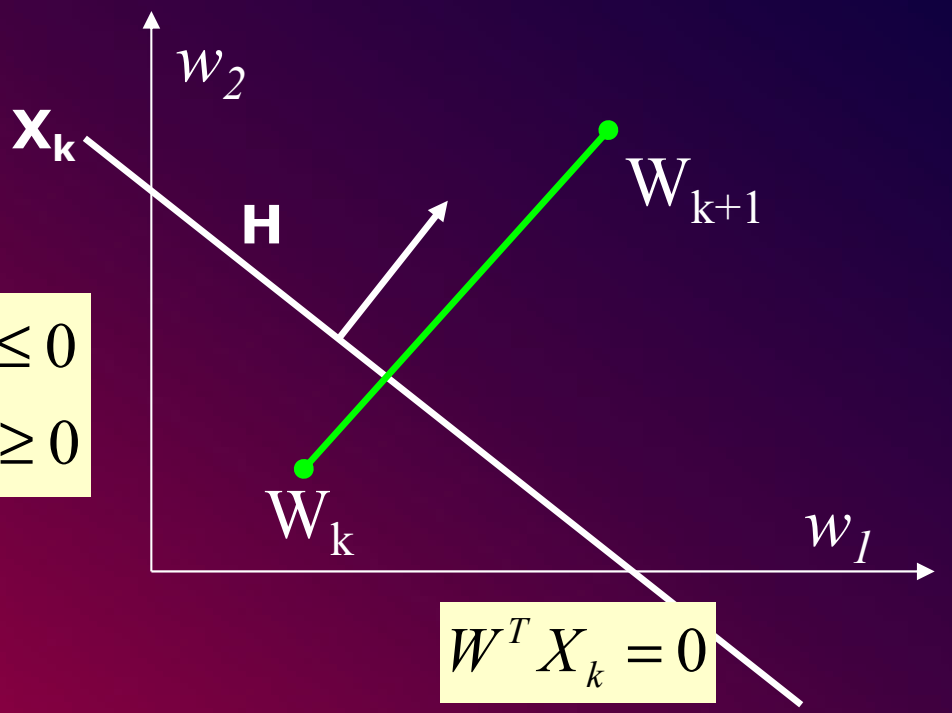
LMS learning Law in BPNN or FFNN models



Read about perceptron
vs. multi-layer feedforward network

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

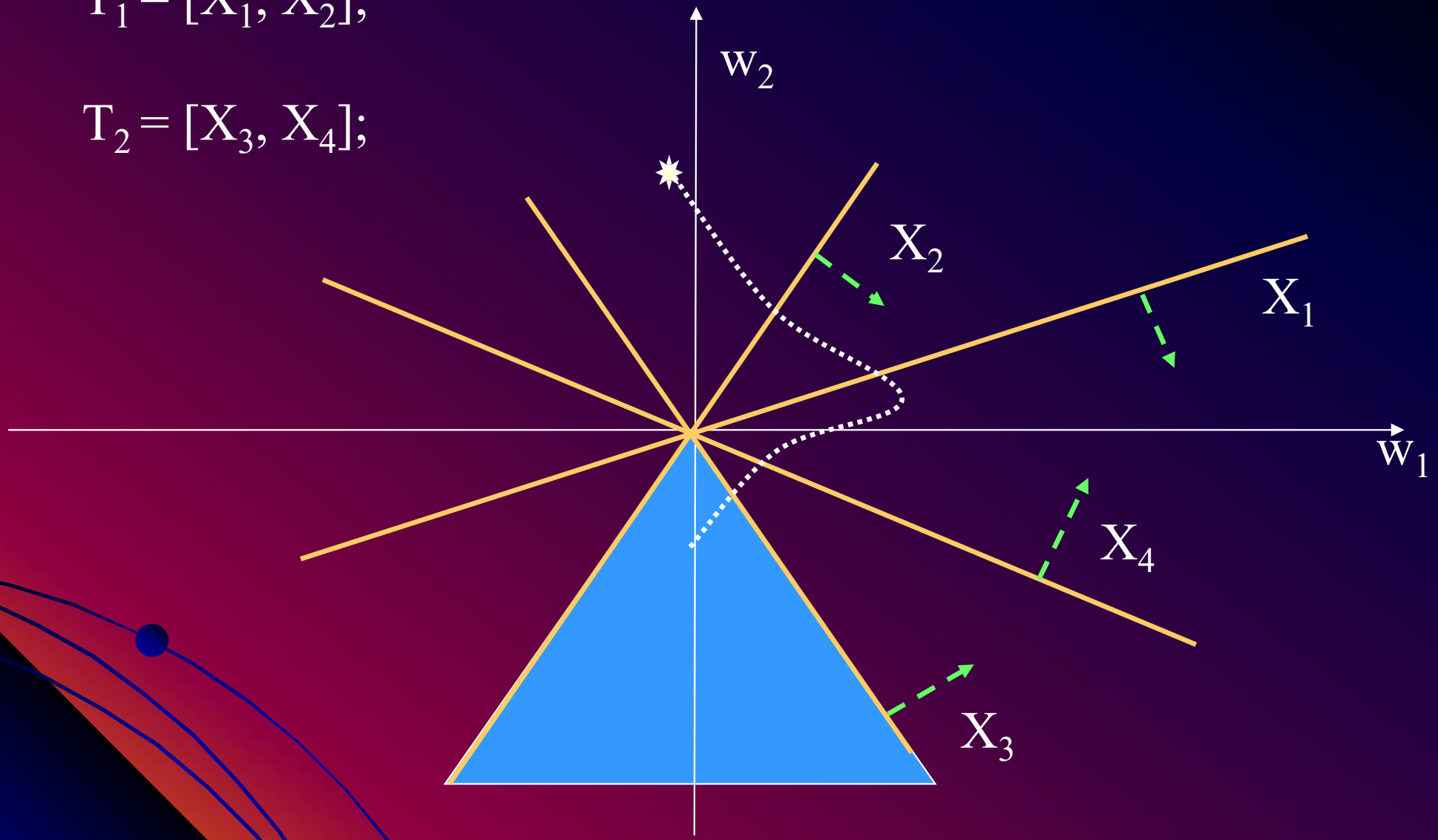
η_κ is the learning rate parameter



$$W_{k+1} = \begin{cases} W_k + \eta_k X_k & \text{if } X_k \in \mathrm{X}_1 \text{ and } X_k^T W_k \le 0 \\ W_k - \eta_k X_k & \text{if } X_k \in \mathrm{X}_0 \text{ and } X_k^T W_k \ge 0 \end{cases}$$

$T_1 = [X_1, X_2]$;

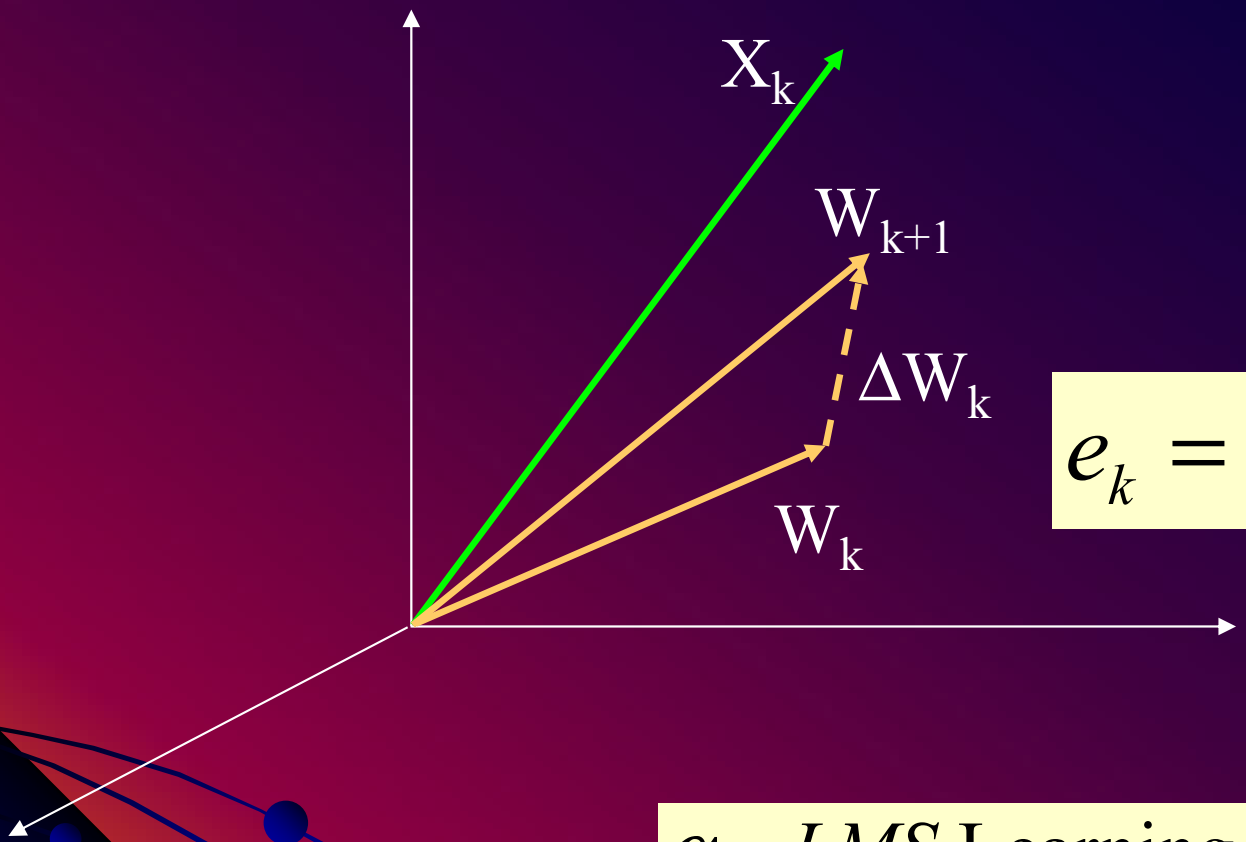
$T_2 = [X_3, X_4]$;



η_κ decreases with each iteration

$$W_{k+1} = \begin{cases} W_k + \eta_k X_k & \text{if } X_k \in \mathrm{X}_1 \text{ and } X_k^T W_k \le 0 \\ W_k - \eta_k X_k & \text{if } X_k \in \mathrm{X}_0 \text{ and } X_k^T W_k \ge 0 \end{cases}$$

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



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

$$\alpha - LMS \text{ Learning Algorithm}:$$

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

Also, Check: Hebb, Delta

MSE error surface (in case of multi-layer perceptron):

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

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

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

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

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

Effect of class Priors – revisiting DBs in a more general case.

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

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

$$g_i(x) = \ln p(x|w_i) + \ln P(w_i)$$

$$g_i(x) = \frac{-1}{2}(x - \mu_i)^t \Sigma_i^{-1}(x - \mu_i) - \frac{d}{2}\ln 2\pi - \frac{1}{2}\ln|\Sigma| + \ln P(w_i)$$

CASE A. – Same diagonal Σ, with identical diagonal elements.

Canceling in class-invariant terms:

$$g_i(X) = \frac{-1}{2\sigma^2}[(X-\mu_i)^T(X-\mu_i)] + \ln P(w_i)$$

$$g_i(X) = \frac{-1}{2\sigma^2}[X^T X - 2\mu_i^T X + \mu_i^T \mu_i] + \ln P(w_i)$$

$$g_i(X) = \frac{-1}{2\sigma^2}[X^T X - 2\mu_i^T X + \mu_i^T \mu_i] + \ln P(w_i)$$

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

where $\omega_i = \mu_i / \sigma^2$ and $\omega_{i0} = -\frac{\mu_i^T \mu_i}{2\sigma^2} + \ln P(w_i)$

The linear DB is thus: $g_k(X) = g_l(X), k \neq l$

which is: $(\omega_k^T - \omega_l^T)X + (\omega_{k0} - \omega_{l0}) = 0;$

Prove that the 2nd constant term:

$$(\omega_{k0} - \omega_{l0}) = (\omega_l - \omega_k)^T X_0; \text{ where}$$

$$X_0 = \frac{1}{2}(\mu_k + \mu_l) - \sigma^2 \frac{\mu_k - \mu_l}{\|\mu_k - \mu_l\|^2} \ln \frac{P(\omega_k)}{P(\omega_l)}$$

Thus the linear DB is: $W^T(X - X_0) = 0;$

where, $W = \mu_k - \mu_l$

Nothing new, seen earlier

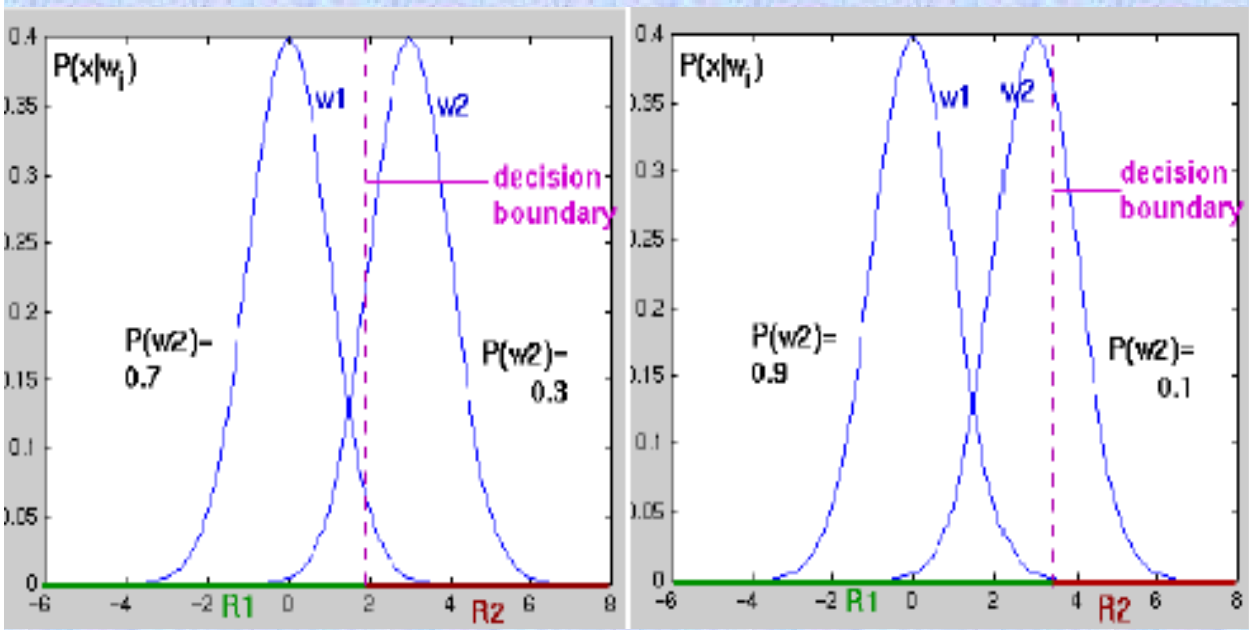
CASE – A. – Same diagonal Σ, with identical diagonal elements (Contd.)

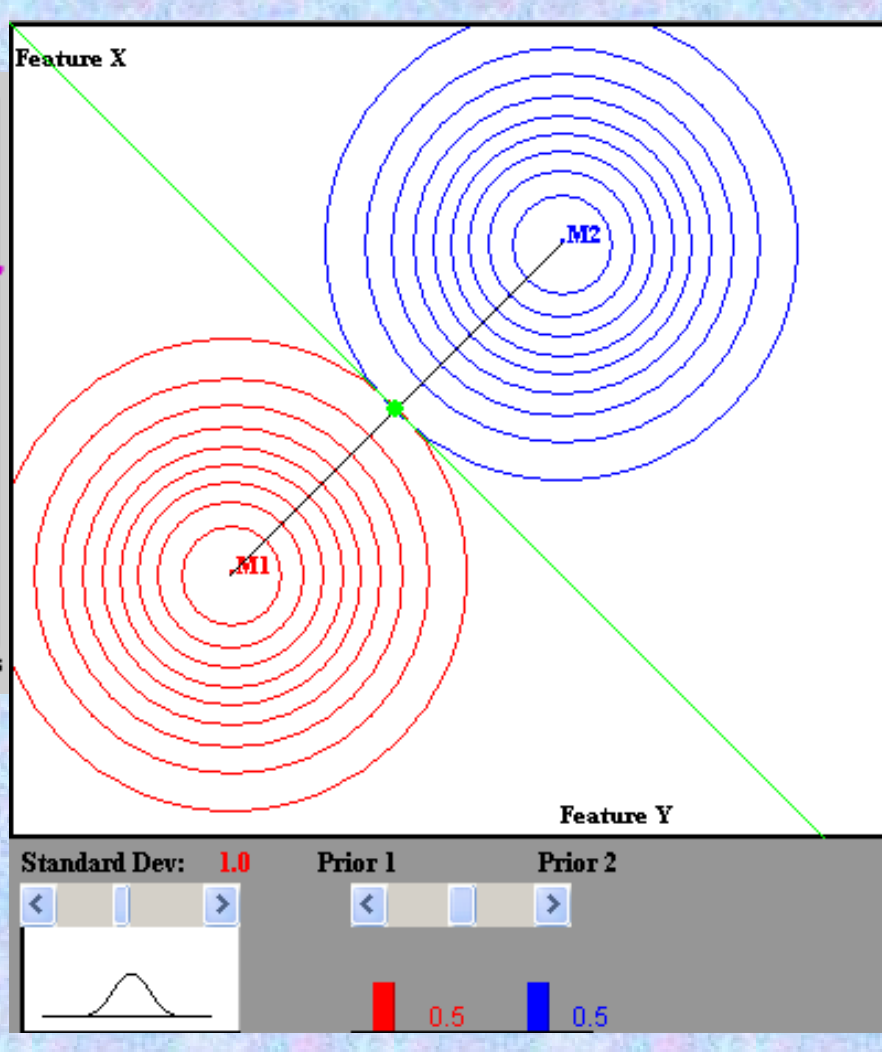
Linear DB:

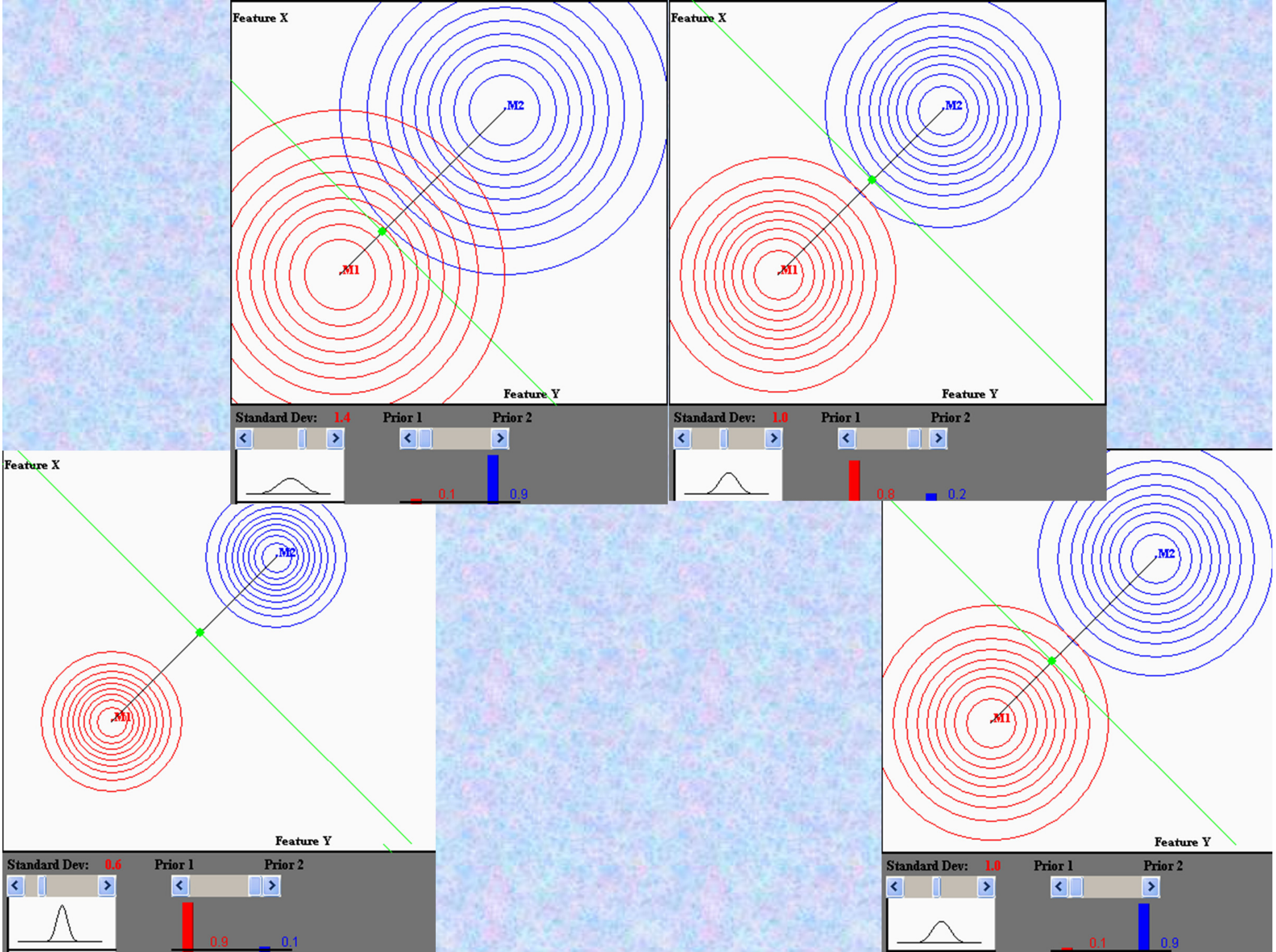
$$W^T(X - X_0) = 0;$$

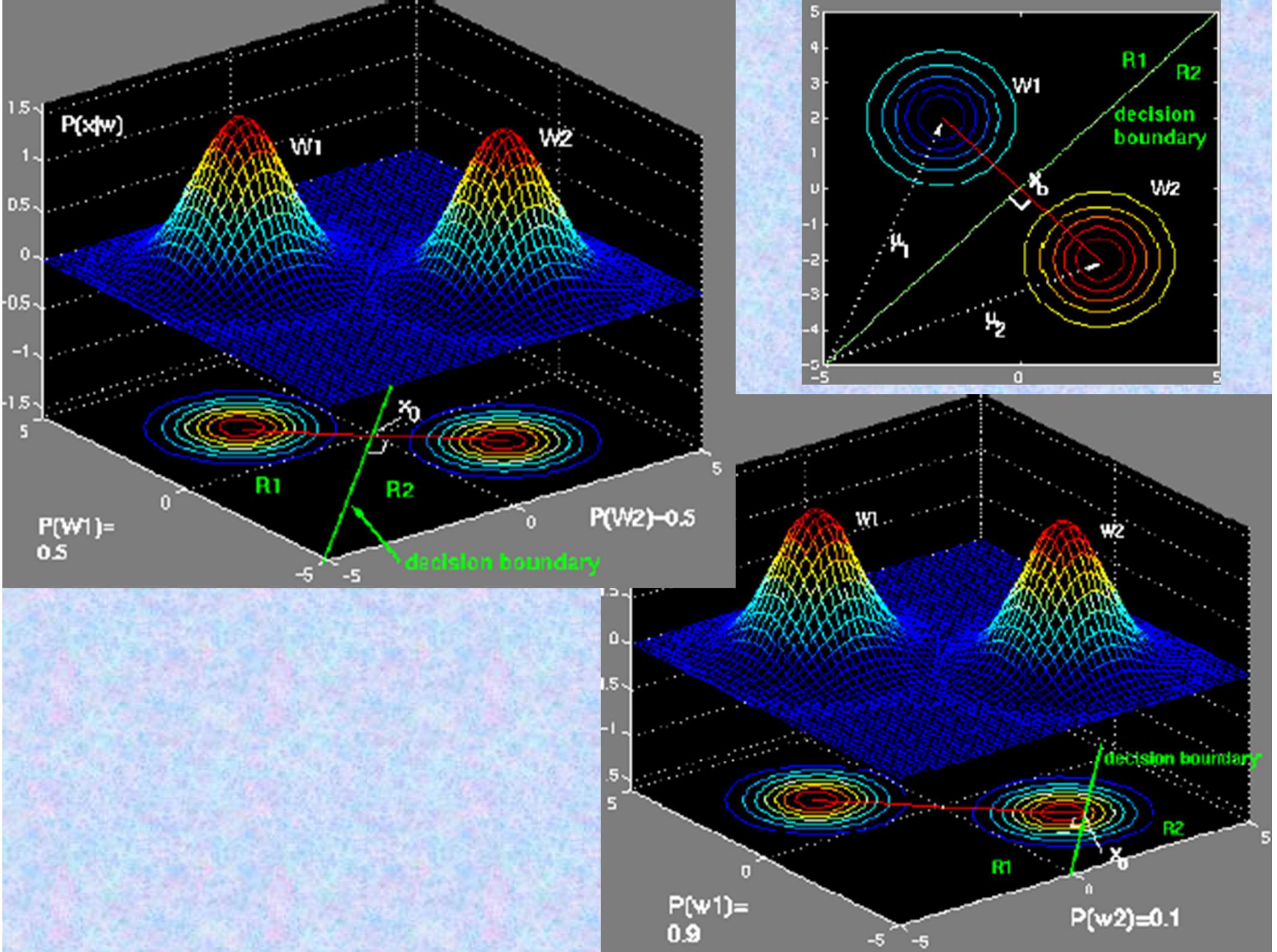
where, $W = \mu_k - \mu_l$

$$X_0 = \frac{1}{2}(\mu_k + \mu_l) - \sigma^2 \frac{\mu_k - \mu_l}{\|\mu_k - \mu_l\|^2} \ln \frac{P(\omega_k)}{P(\omega_l)}$$






Feature X
M2
M1
Feature Y
Standard Dev: 1.4
Prior 1
Prior 2
0.1
0.9
Feature X
M2
M1
Feature Y
Standard Dev: 1.0
Prior 1
Prior 2
0.8
0.2
Feature X
M2
M1
Feature Y
Standard Dev: 0.6
Prior 1
Prior 2
0.9
0.1
Feature X
M2
M1
Feature Y
Standard Dev: 1.0
Prior 1
Prior 2
0.1
0.9


P(x|w)
W1
W2
R1
R2
x_0
decision boundary
P(W1)= 0.5
P(W2)=0.5
W1
W2
R1
R2
x_0
decision boundary
P(w1)= 0.9
P(w2)=0.1
W1
W2
R1
R2
x_0
decision boundary
μ_1
μ_2

CASE – B. – Arbitrary Σ, but identical for all class.

$$g_i(X) = \frac{-1}{2}[(X-\mu_i)^T \Sigma^{-1}(X-\mu_i)] + \ln P(w_i)$$

Removing the class-invariant quadratic term:

$$g_i(X) = \frac{-1}{2}\mu_i^T \Sigma^{-1}\mu_i + (\Sigma^{-1}\mu_i)^T X + \ln P(w_i)$$

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

$$where\ \omega_i = \Sigma^{-1}\mu_i \text{ and } \omega_{i0} = -\frac{1}{2}\mu_i^T \Sigma^{-1}\mu_i + \ln P(w_i)$$

The linear DB is thus: $g_k(X) = g_l(X), k \neq l$

which is: $(\omega_k^T - \omega_l^T)X + (\omega_{k0} - \omega_{l0}) = 0;$

$(\omega_{k0} - \omega_{l0}) = (\omega_l - \omega_k)^T X_0$; where

$$X_0 = \frac{1}{2}(\mu_k + \mu_l) - \frac{\mu_k - \mu_l}{(\mu_k - \mu_l)^T \Sigma^{-1}(\mu_k - \mu_l)} \ln \frac{P(\omega_k)}{P(\omega_l)}$$

← ***Prove it.***

Thus the linear DB is: $W^T(X - X_0) = 0;$

where, $W = \omega_k - \omega_l$ $\quad where\ \omega_i = \Sigma^{-1}\mu_i$

$$Thus,\ W = \Sigma^{-1}(\mu_k - \mu_l);$$

The normal to the DB, "W", is thus the transformed line joining the two means.

The transformation matrix is a symmetric Σ^{-1}.

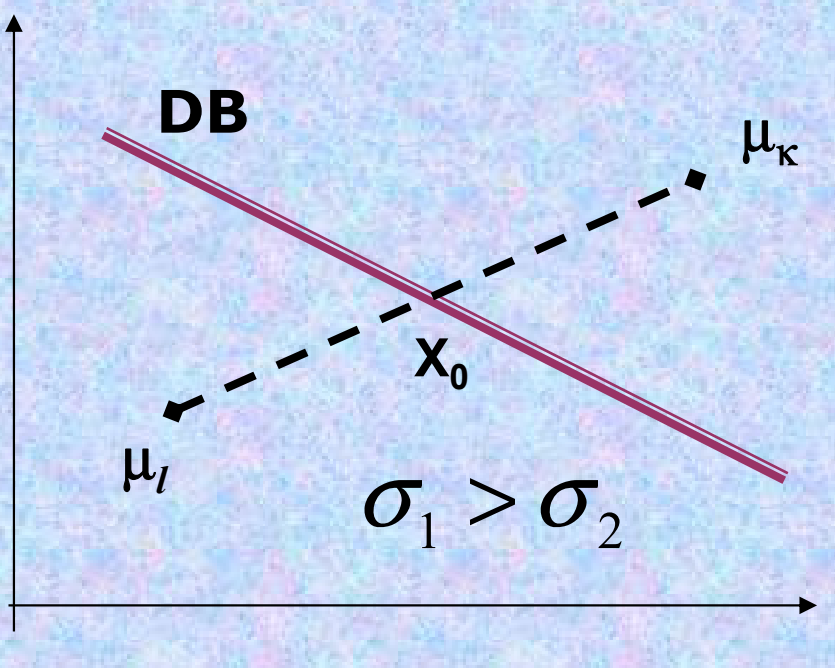
The DB is thus -

a tilted (rotated) vector joining the two means.

Let Σ (2–D) be diagonal, with non-identical diagonal elements: σ_1 and σ_2.

$$Then,\ W_D = \begin{bmatrix} \\ \\ \end{bmatrix};$$

$d = 2$ case. $\quad Direction\ of\ DB = \begin{bmatrix} \\ \\ \end{bmatrix}$



Thus the linear DB is: $W^T(X - X_0) = 0;$

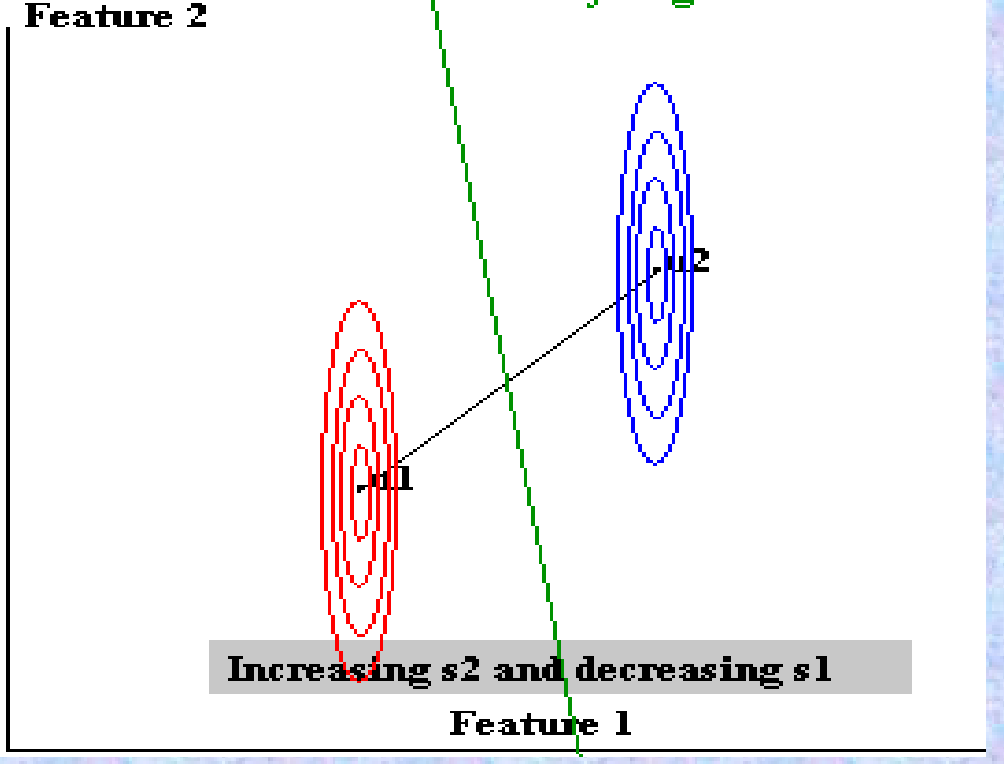
where, $W = \omega_k - \omega_l$ $\quad$ $where\ \omega_i = \Sigma^{-1}\mu_i$

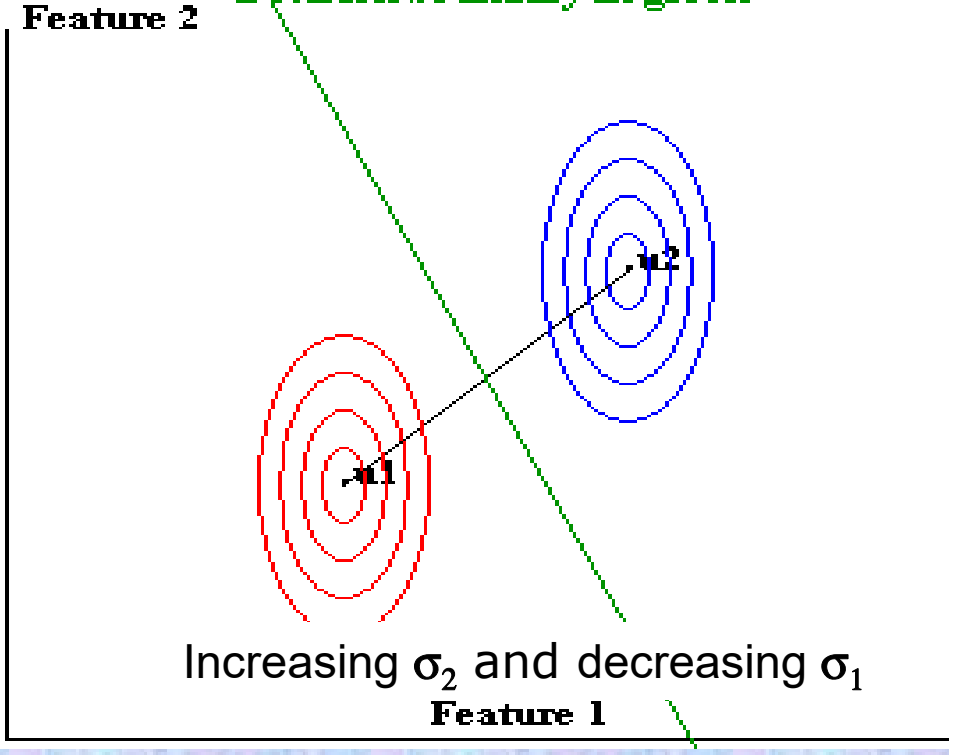
$$Thus,\ W = \Sigma^{-1}(\mu_k - \mu_l);$$

Special case:

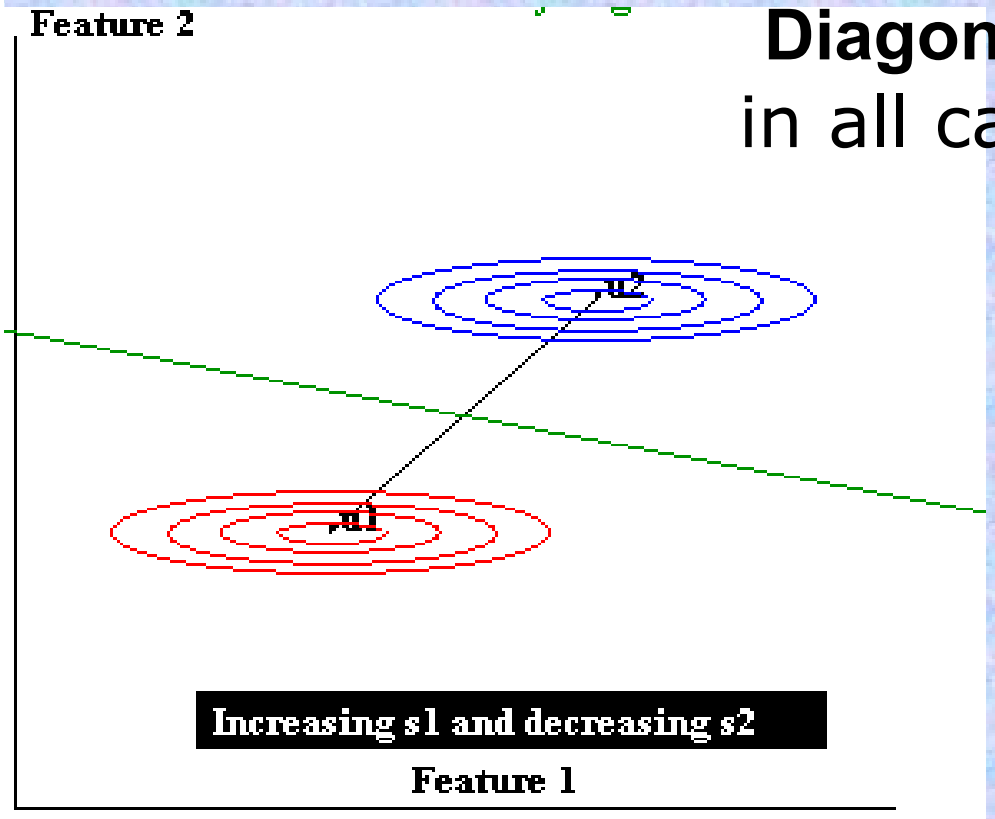
Let, Σ (2–D) be arbitrary, but with diagonal elements (=1).

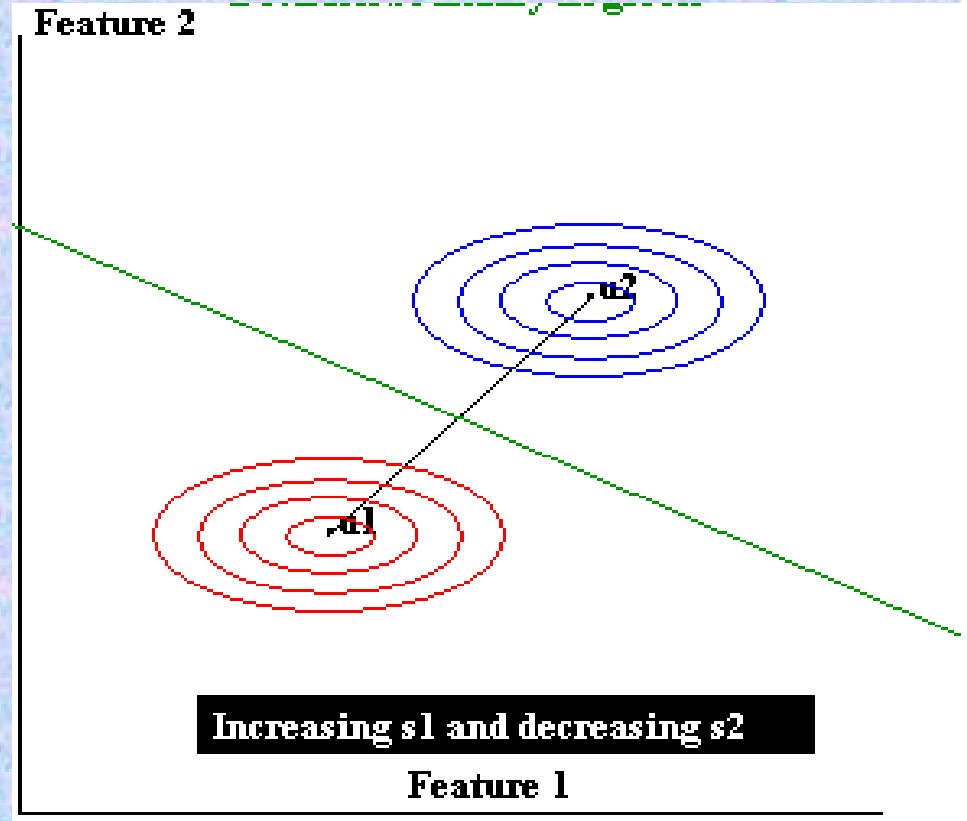
Solve for **W** in this case, and compare with the diagonal Σ case.

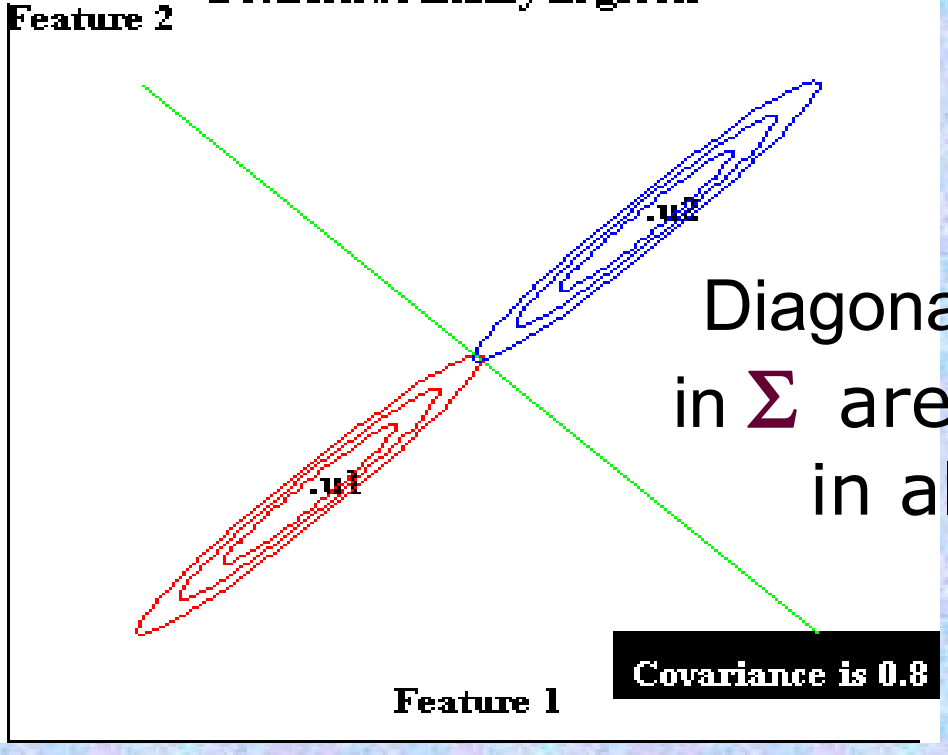


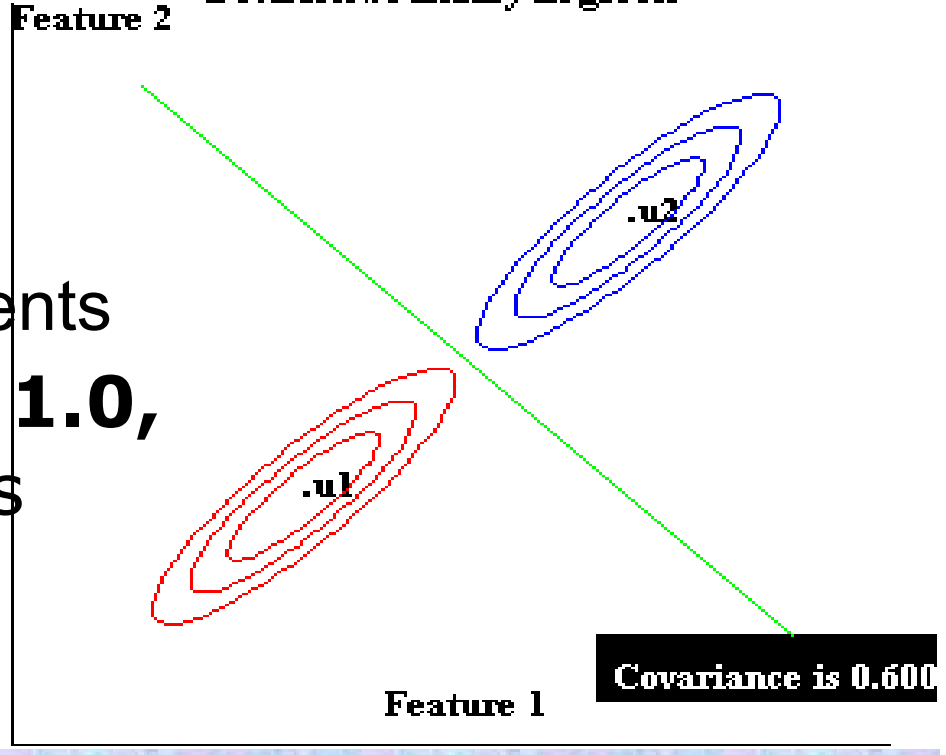


Diagonal Σ
in all cases.

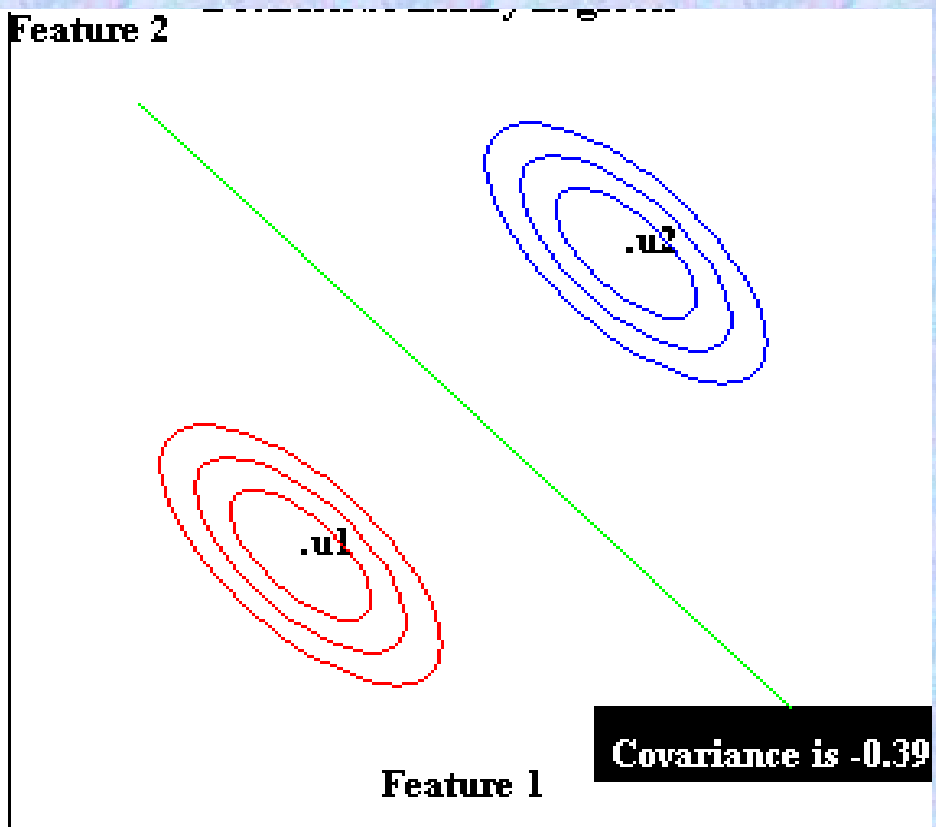


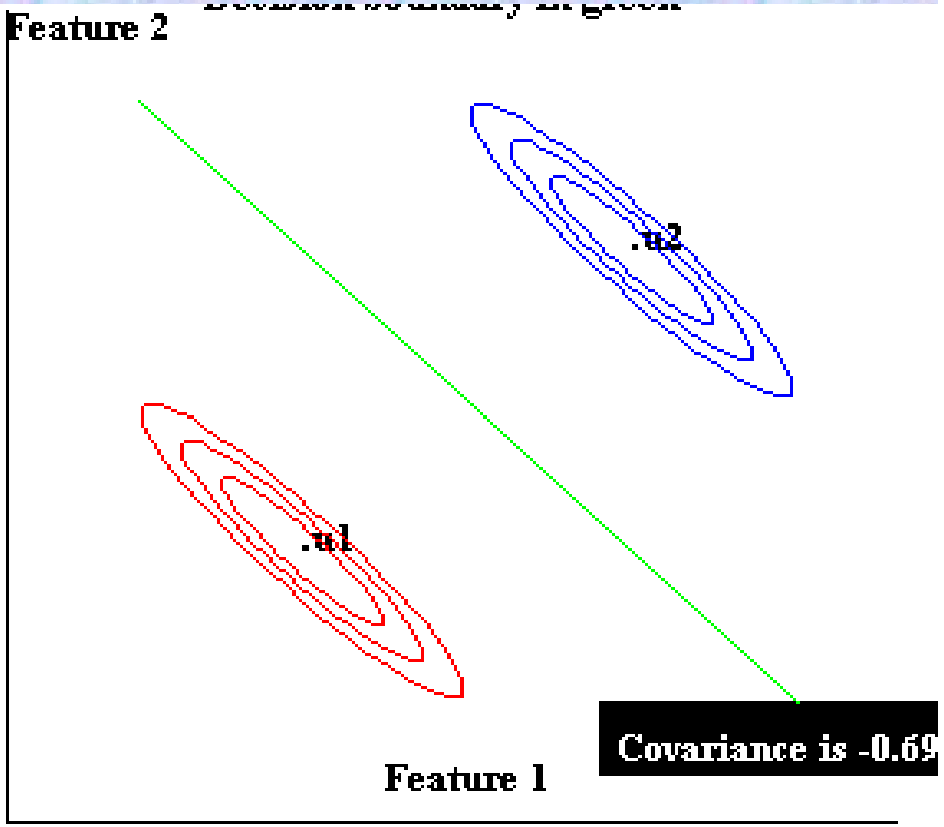


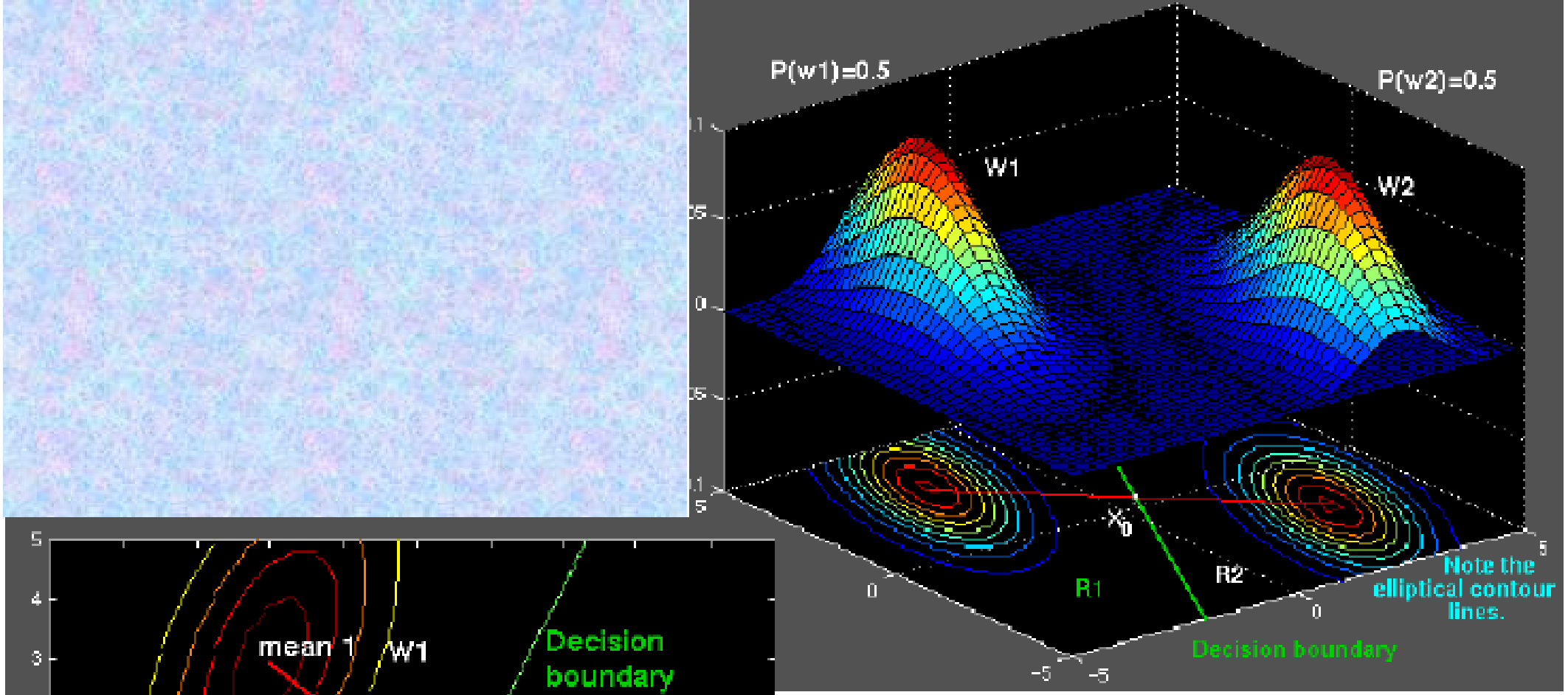


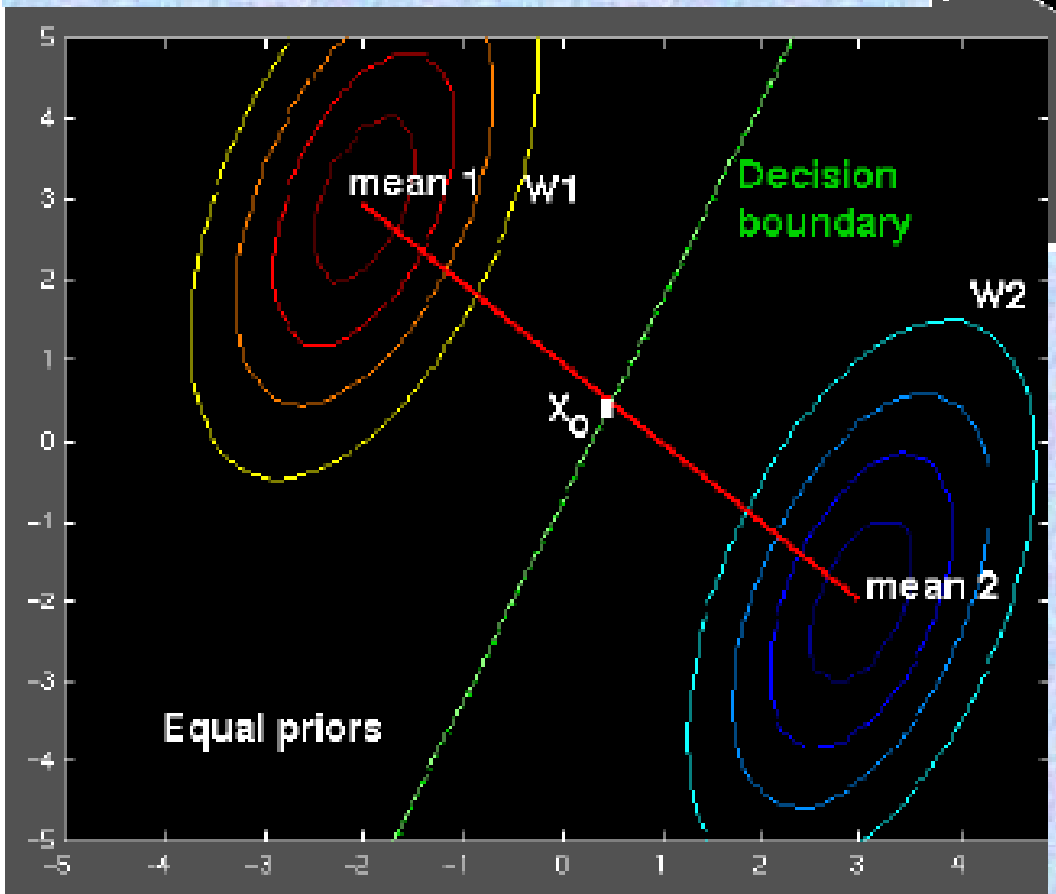


Diagonal elements in Σ are both **1.0,** in all cases



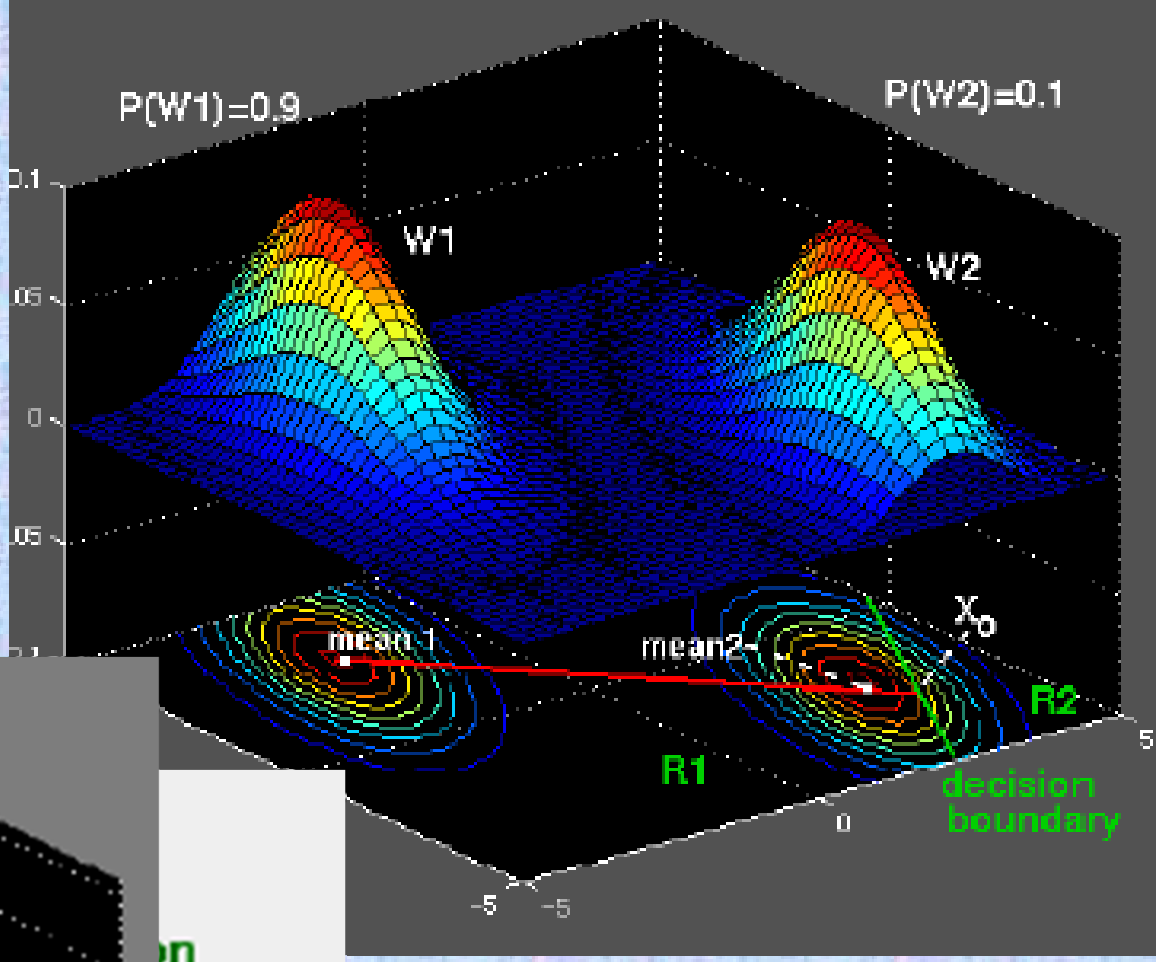


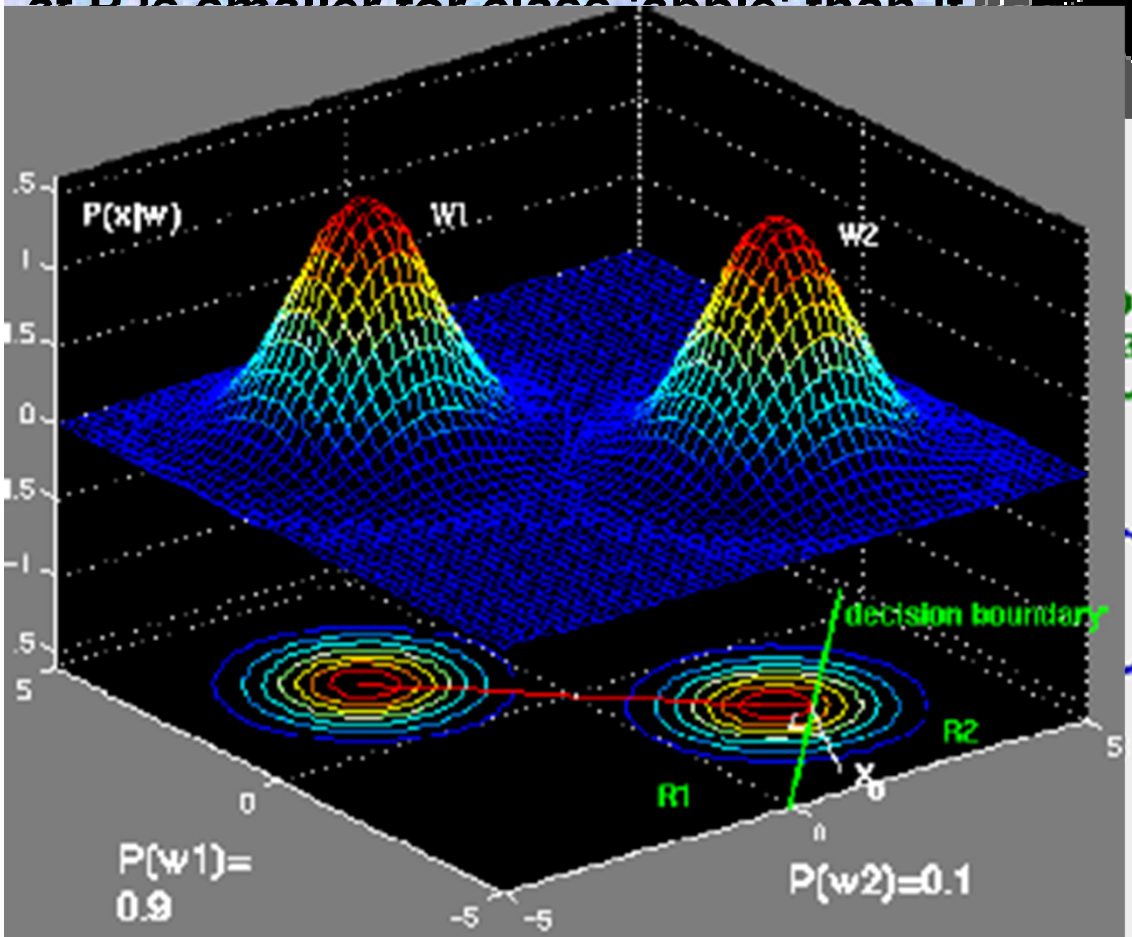

P(w1)=0.5
P(w2)=0.5
W1
W2
x_0
R1
R2
Note the elliptical contour lines.
Decision boundary


mean 1
W1
Decision boundary
W2
x_0
mean 2
Equal priors

Point P is actually closer (in the Euclidean sense) to the mean for the Orange class.

The discriminant function evaluated at P is smaller for class 'apple' than it





CASE C. – Arbitrary Σ, all parameters are class dependent.

$$g_i(X) = \frac{-1}{2}[(X-\mu_i)^T \Sigma_i^{-1}(X-\mu_i)] - \frac{-1}{2}\ln|\Sigma_i| + \ln P(w_i)$$

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

$$\textit{where} \;\; \mathrm{W_i} = \frac{-1}{2}\Sigma_i^{-1};$$

$$\omega_i = \Sigma_i^{-1}\mu_i \text{ and}$$

$$\omega_{i0} = -\frac{1}{2}\mu_i^T\Sigma_i^{-1}\mu_i - \frac{1}{2}\ln|\Sigma_i| + \ln P(w_i)$$

The DBs and DFs are hyper-quadrics. $g_k(X) = g_l(X), k \neq l$

We shall first look into a few cases of such surfaces next.

Example [Duda, Hart]:

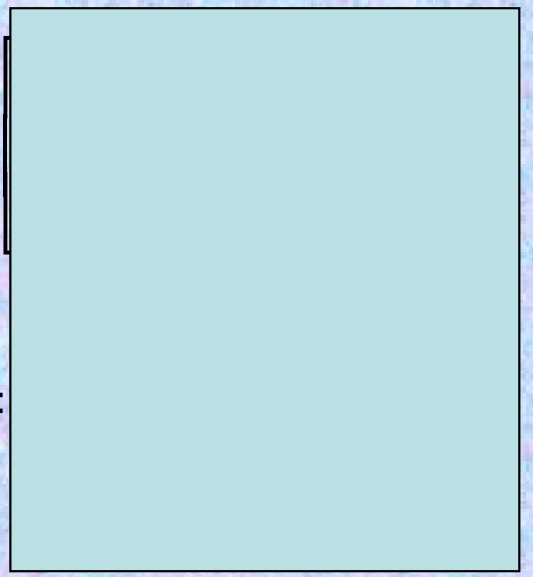
$$\mu_1 = \begin{bmatrix} 3 \\ 6 \end{bmatrix}; \quad \Sigma_1 = \begin{bmatrix} 1/2 & 0 \\ 0 & 2 \end{bmatrix};$$

$$\mu_2 = \begin{bmatrix} 3 \\ -2 \end{bmatrix}; \quad \Sigma_2 = \begin{bmatrix} 2 & 0 \\ 0 & 2 \end{bmatrix};$$

Draw and Visualize (qualitatively) the iso-contours

$$\Sigma_1^{-1} =$$

$$\Sigma_2^{-1} =$$



Assume; $P(w_1) = P(w_1) = 0.5$;

Get expression of DB:

Quadratic Decision Boundaries

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

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

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:

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

Special cases of equation:

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

Case 1:

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

Case 2:

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; Hyperbolic ??

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

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

In the above equation, the total number of parameters is: ??

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

Organize these parameters, and manipulate the equation to obtain:

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

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

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

(d^2-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/0, we get a hyper-spheres/planes.

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

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

Example of linearization:

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

To <u>Linearize</u>, let $x_3 = x_1^2$. Then:

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

$$\text{where, } X = [x_1, x_2, x_3]^T$$

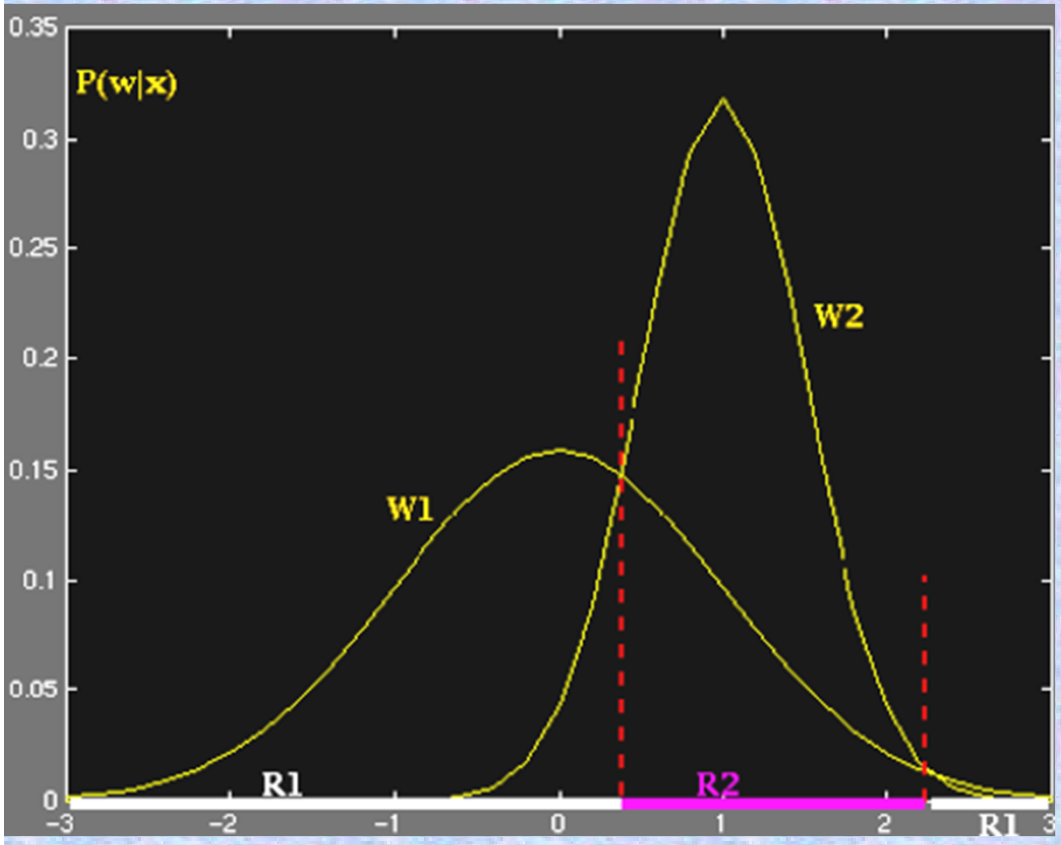
$$\text{and } \; W^T = [-3, 1, -1]$$

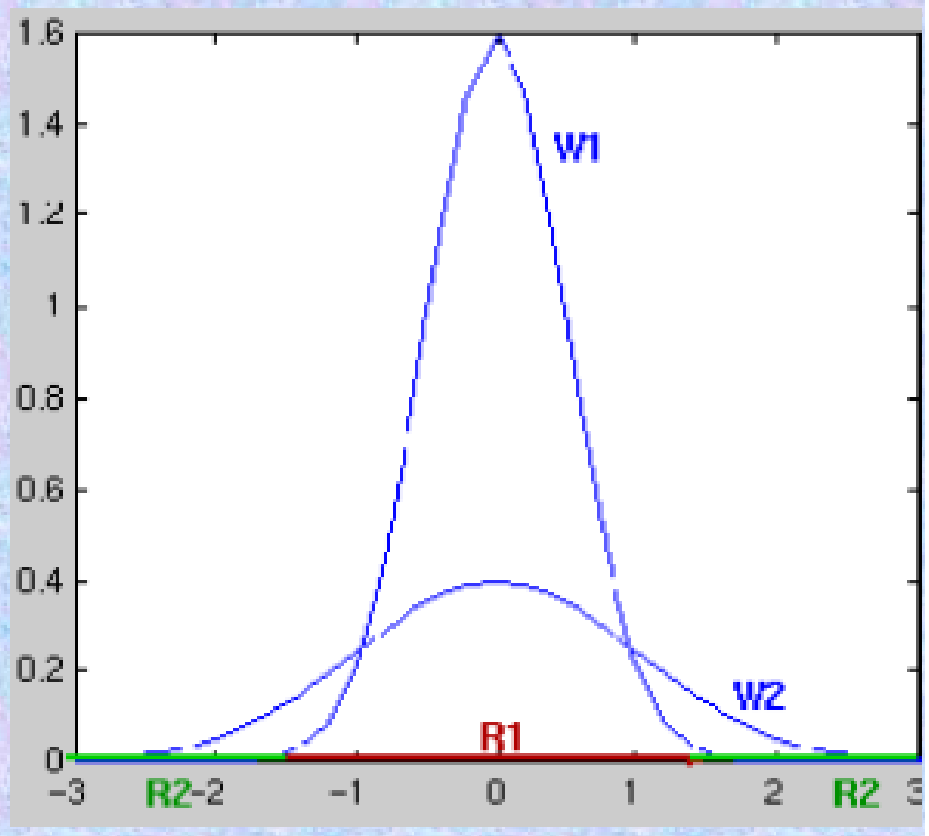
CASE – C. – Arbitrary Σ, all parameters are class dependent – contd..

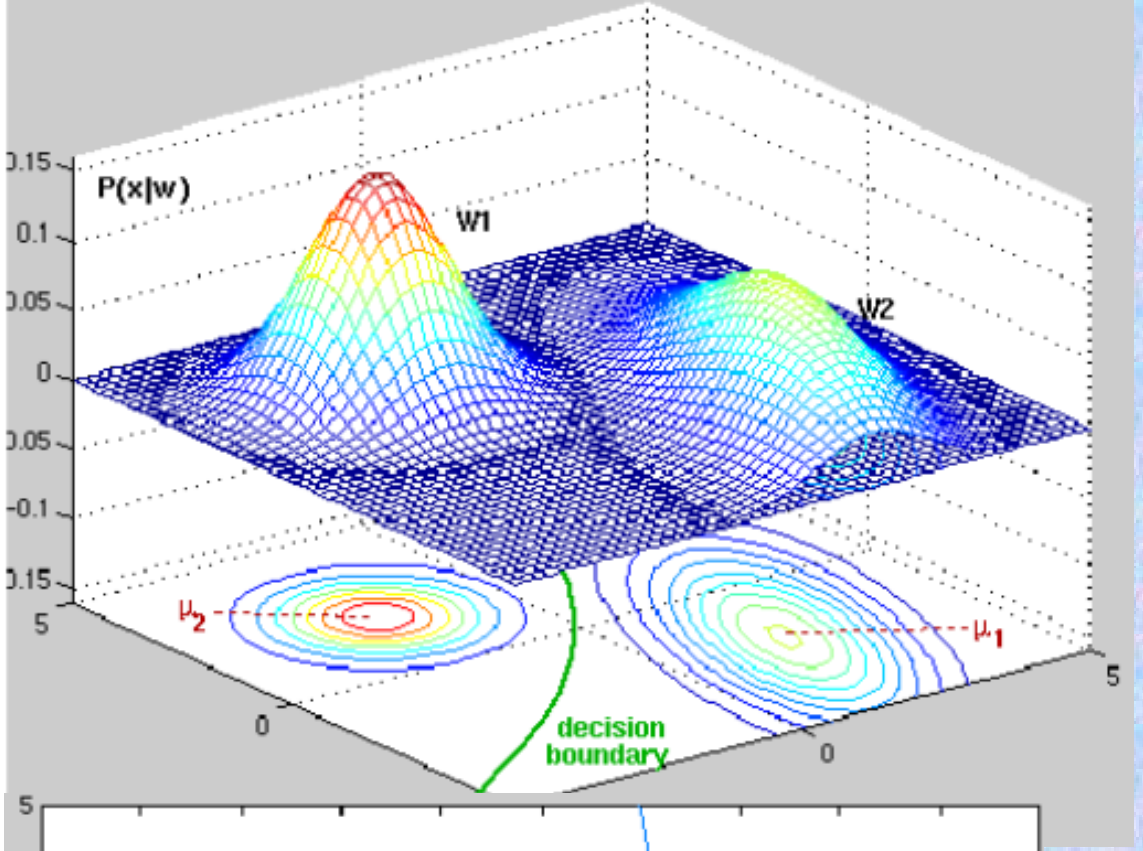
$$g_i(X) = \frac{-1}{2}[(X-\mu_i)^T \Sigma_i^{-1}(X-\mu_i)] - \frac{-1}{2}\ln|\Sigma_i| + \ln P(w_i)$$

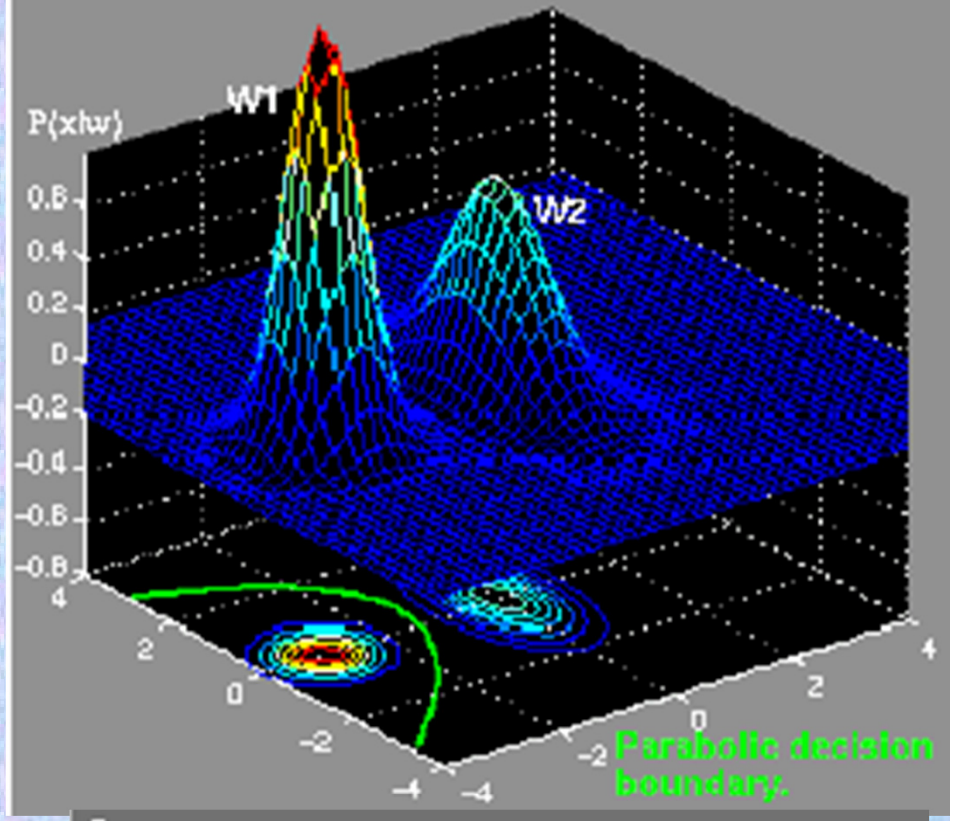
Thus, $g_i(X) = X^T W_i X + \omega_i^T X + \omega_{i0}$; *where* $\mathrm{W_i} = \frac{-1}{2}\Sigma_i^{-1}$;

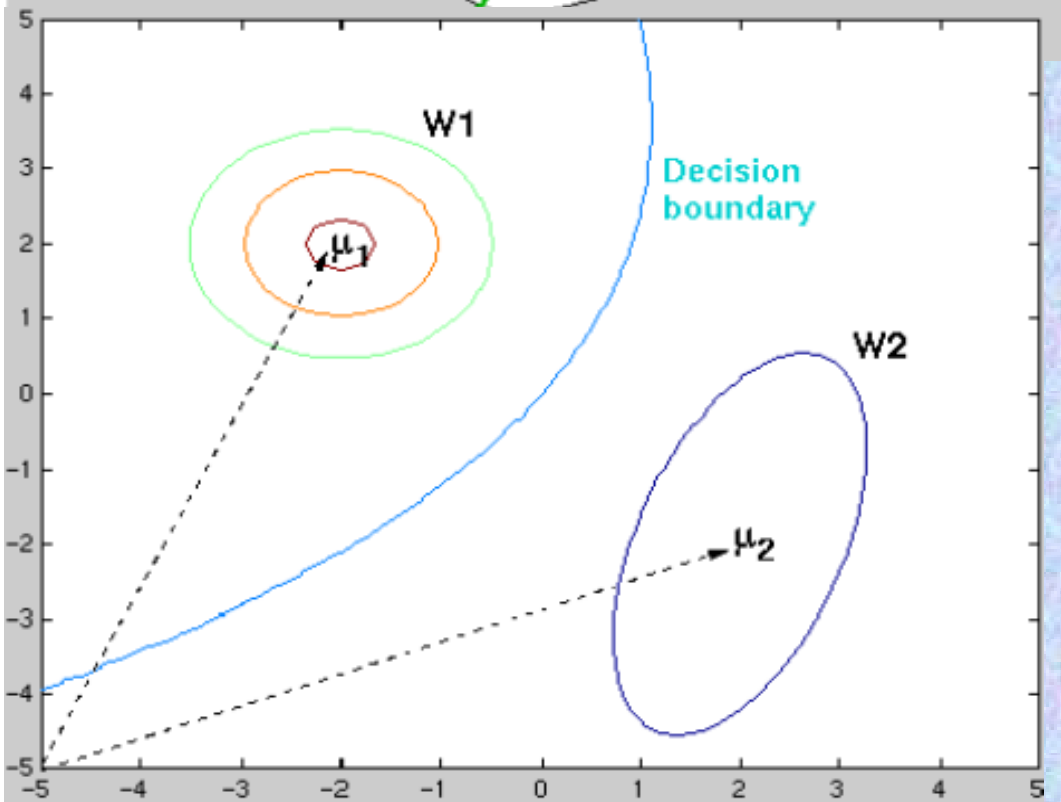
$$\omega_i = \Sigma_i^{-1}\mu_i \quad \text{and} \quad \omega_{i0} = -\frac{1}{2}\mu_i^T \Sigma_i^{-1}\mu_i - \frac{1}{2}\ln|\Sigma| + \ln P(w_i)$$

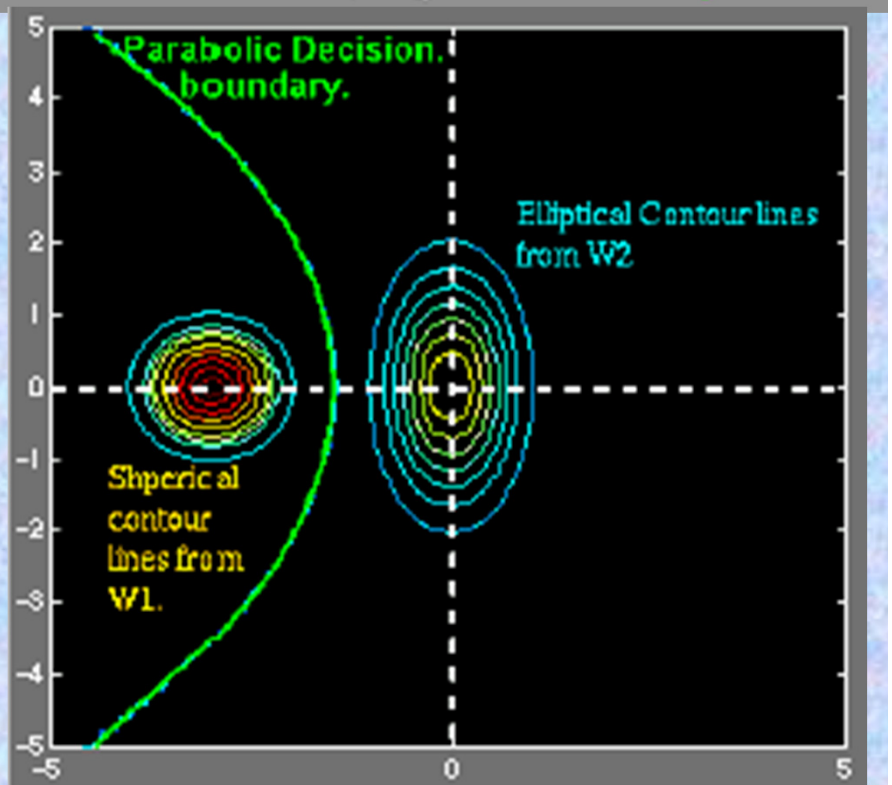





P(x|w)
W1
W2
μ_2
μ_1
decision
boundary


P(x|w)
W1
W2
Parabolic decision
boundary.

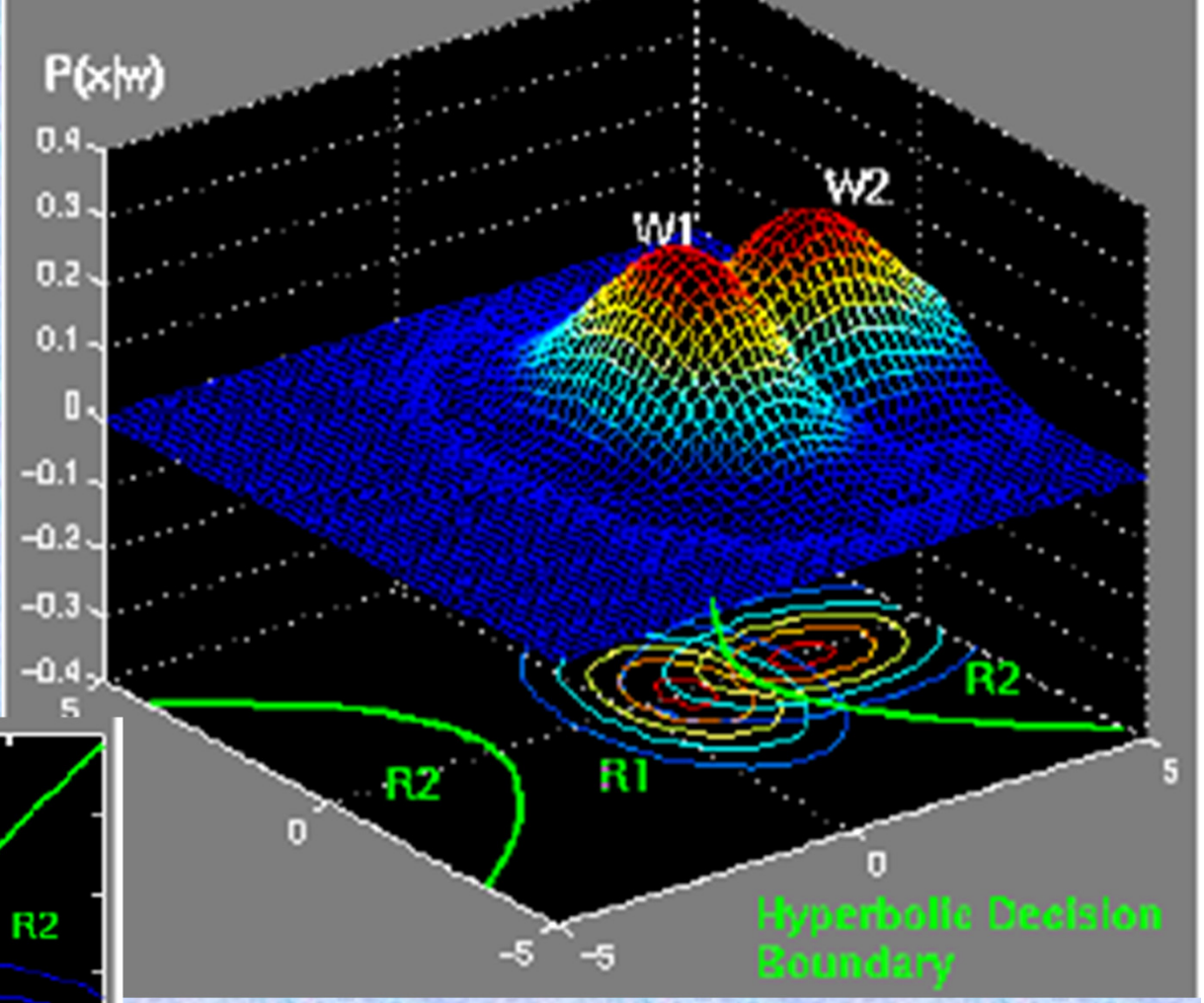

W1
Decision
boundary
μ_1
W2
μ_2

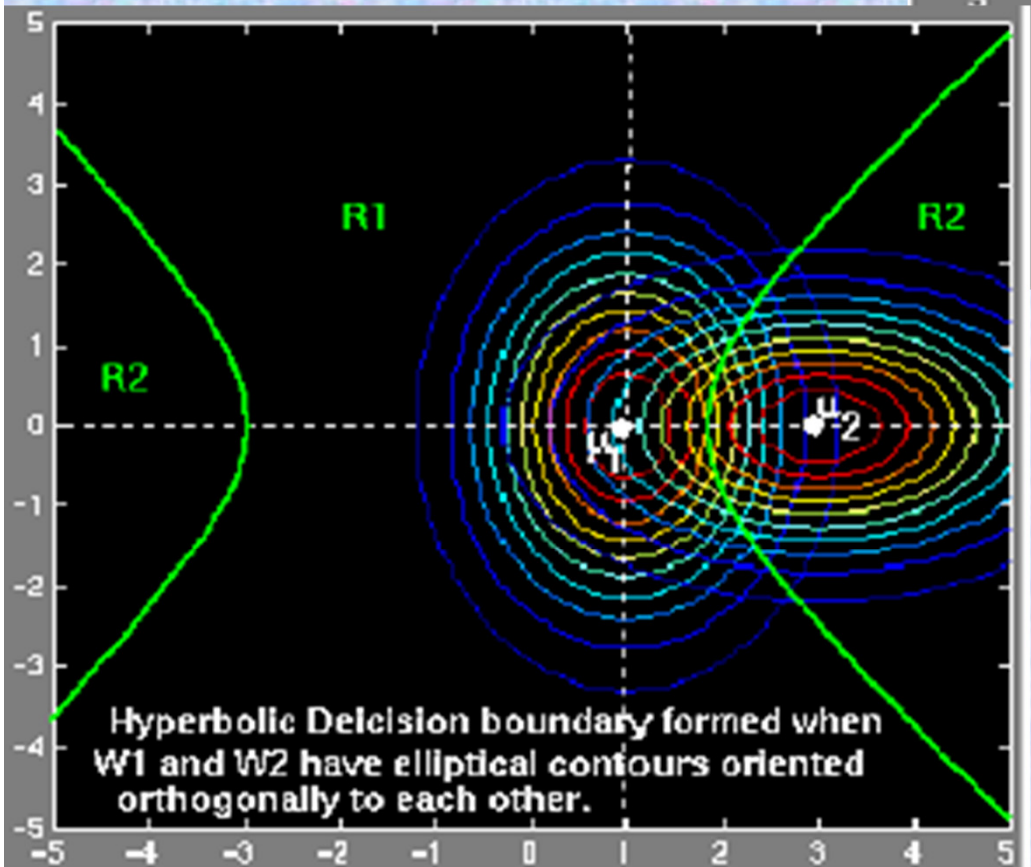

Parabolic Decision.
boundary.
Elliptical Contour lines
from W2
Shperic al
contour
lines from
W1.

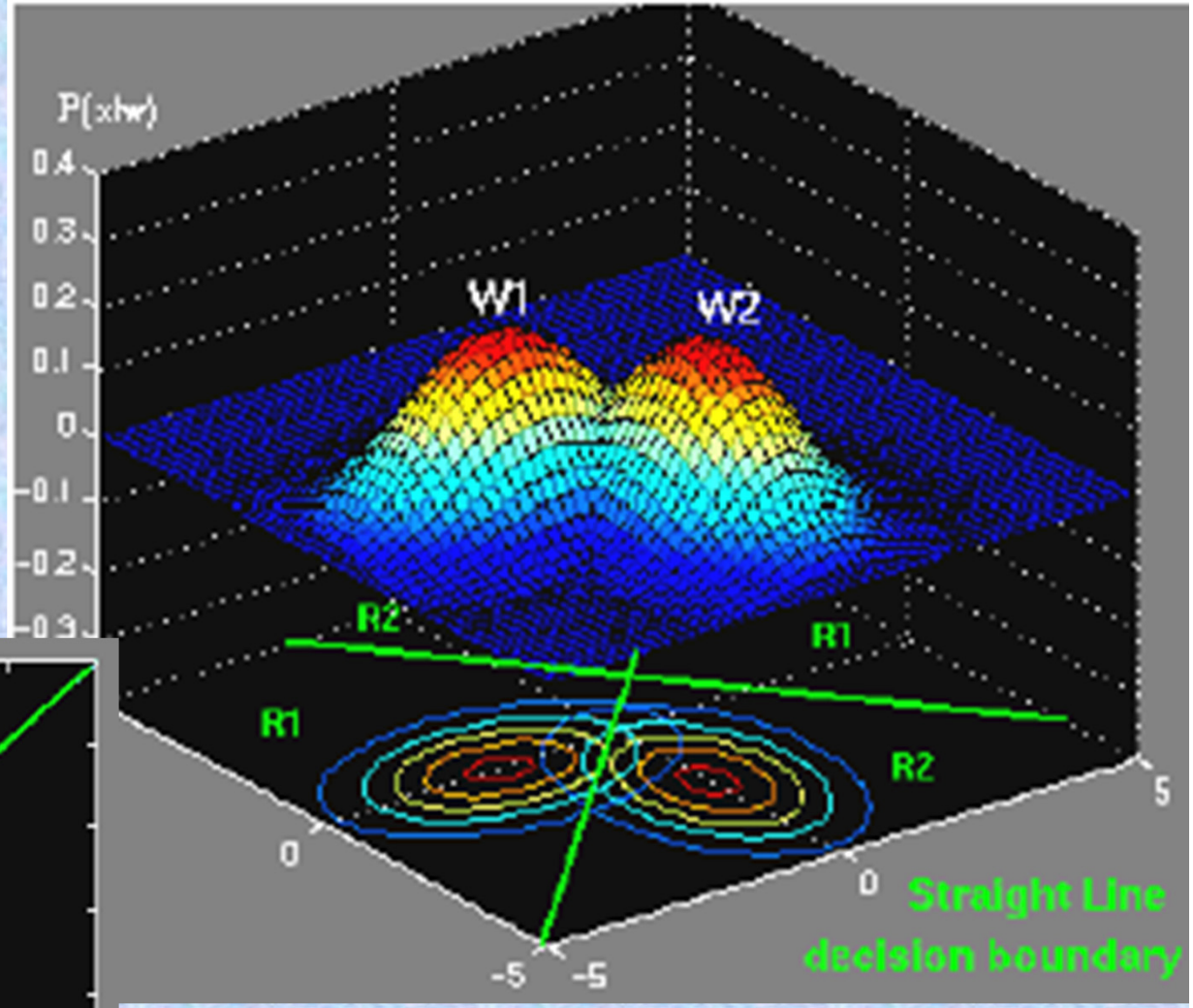
$$\sigma_1^x = \sigma_2^y; \sigma_1^y = \sigma_2^x;$$

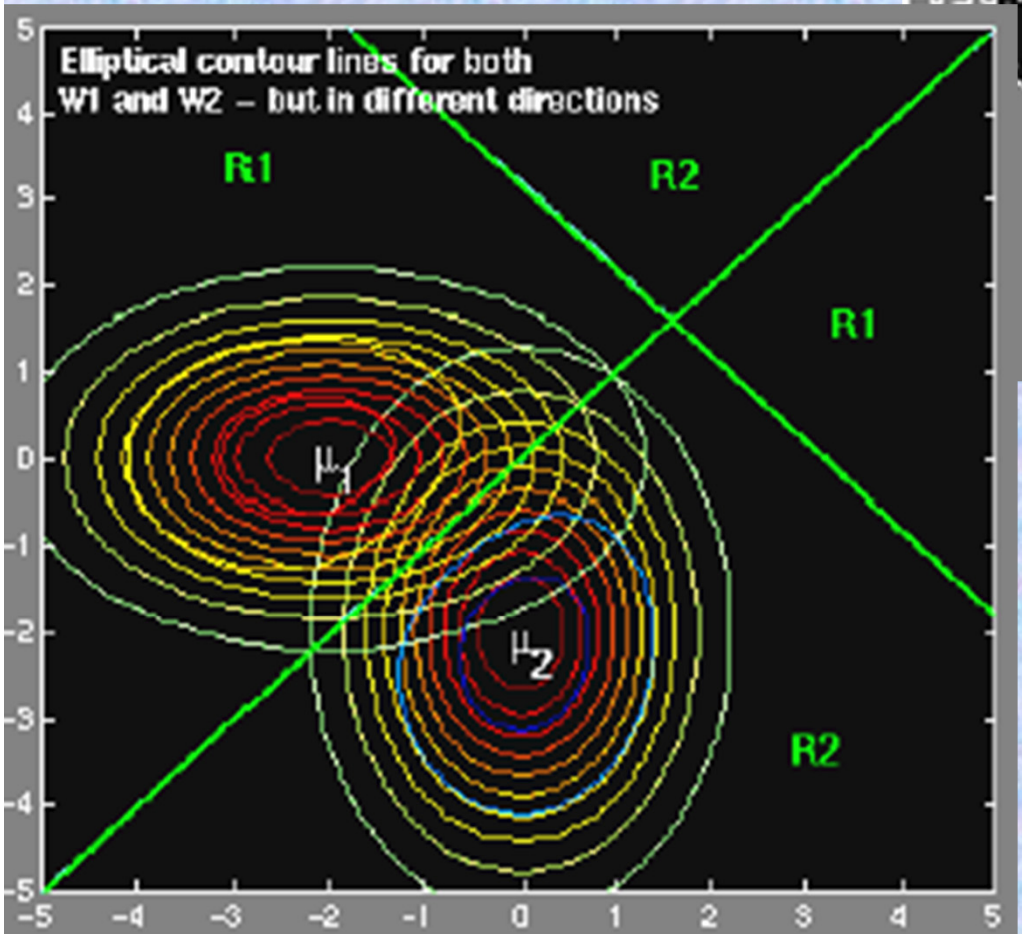
$$\rho_1 = \rho_2 = 0;$$

$$\mu_1^x < \mu_2^x; \mu_1^y = \mu_2^y;$$









$$\sigma_1^x = \sigma_2^y; \sigma_1^y = \sigma_2^x;$$

$$\rho_1 = \rho_2 = 0;$$

$$\mu_1^x = \mu_2^x \pm C; \mu_1^y = \mu_2^y \mp C;$$

Read about GMM,

and estimation using

MLE or EM methods.

Kullback-Leibler divergence

The directed Kullback-Leibler divergence between Exp(λ_0) ('true' distribution) and Exp(λ) ('approximating' distribution) is given by:

$$\Delta(\lambda_0||\lambda) = \log(\lambda_0) - \log(\lambda) + \frac{\lambda}{\lambda_0} - 1.$$

$$D_{\mathrm{KL}}(P\|Q) = \sum_i P(i) \log \frac{P(i)}{Q(i)}.$$

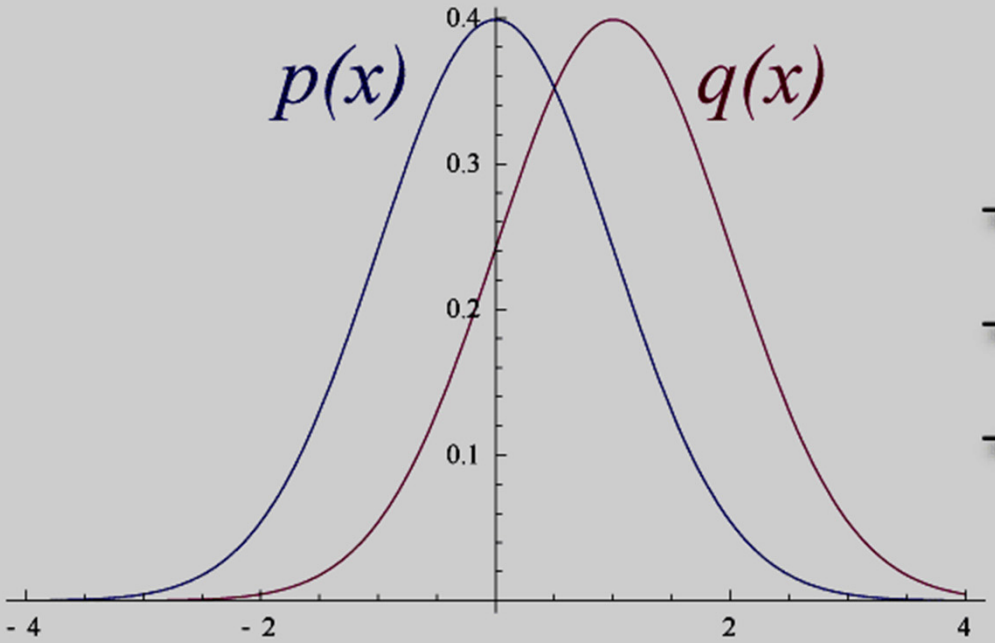
$$D_{KL}(p,q) = \sum_i p(i) \log \frac{p(i)}{q(i)} - \sum p(i) + \sum q(i)$$

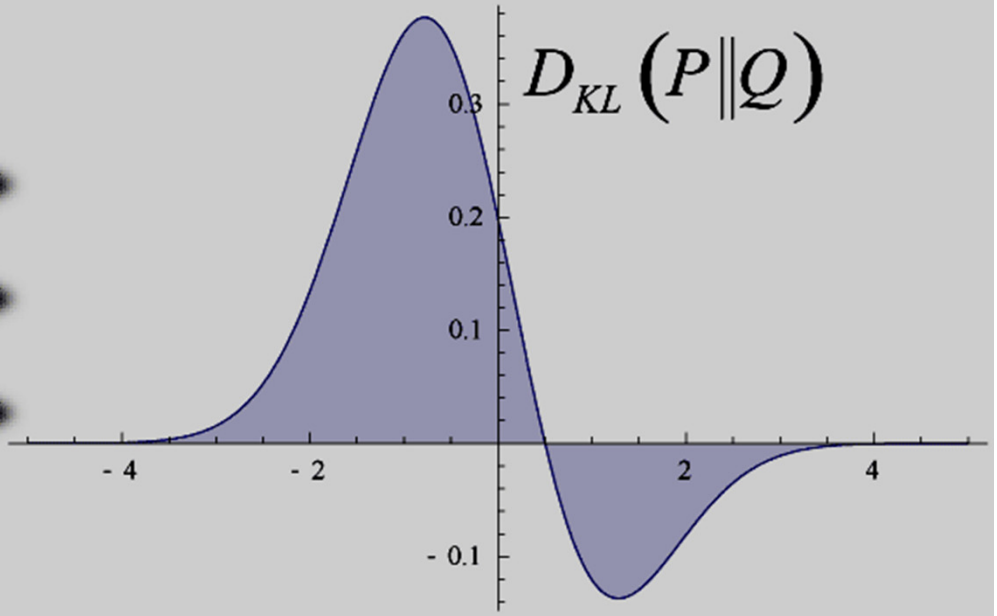
or

$$D_{KL}(p,q) = -\sum_i p(i) \log q(i) + \sum_i p(i) \log p(i)$$

$$= H(p,q) \quad - \quad \mathrm{H(p)}$$

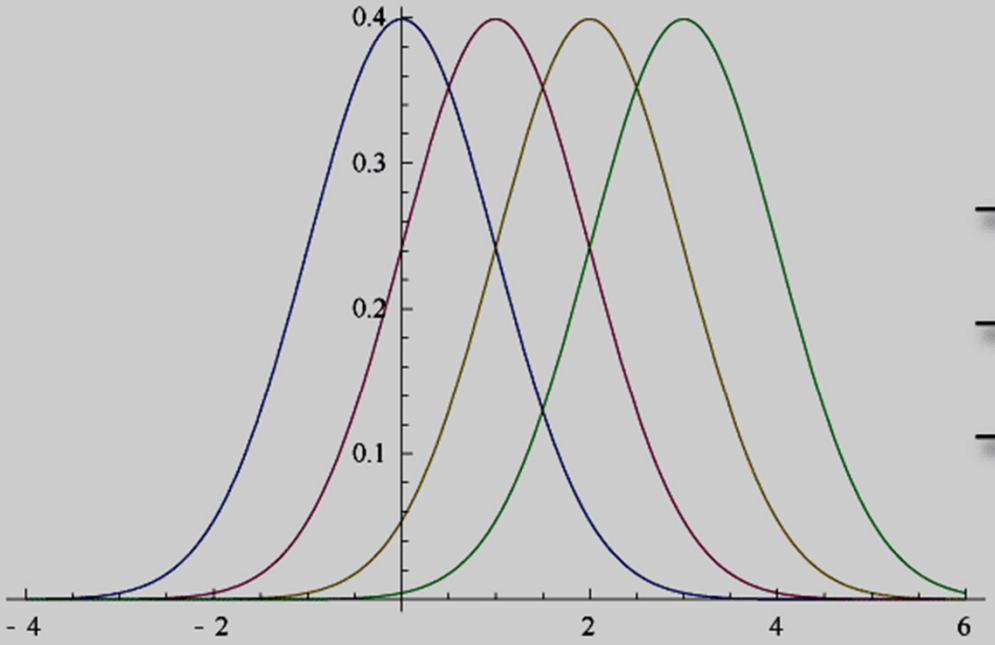
$$= \mathrm{cross\_entropy(P\,\&\,Q)} \quad - \quad \mathrm{entropy(p)}$$

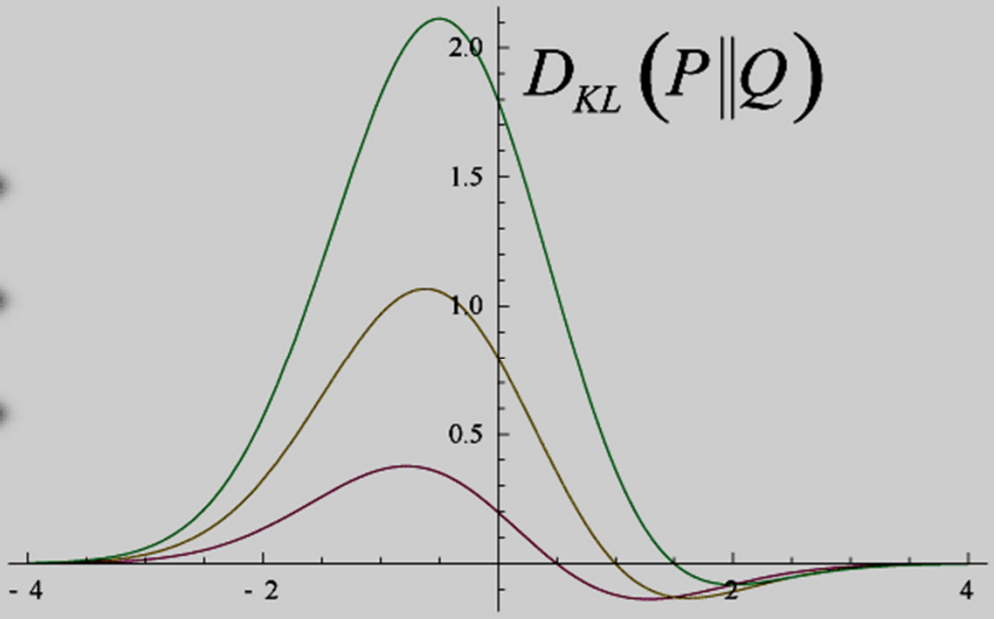




Original Gaussian PDF's

KL Area to be Integrated





- **Bregman divergence**

$$D_{BG}(p,q) = F(p) - F(q) - \langle \nabla F(q), p - q \rangle$$

- **Jensen–Shannon divergence: The Bregman distance associated with F for points (P, Q), is the difference between the value of F at point P and the value of the first-order Taylor expansion of F around point Q evaluated at point P. F is a continuously-differentiable real-valued and strictly convex function defined on a closed convex set.**

$$D_{JS}(p,q) = \frac{D_{KL}(P,M) + D(Q,M)}{2}; \text{ where } M = (P+Q)/2$$

- **Deviance information criterion**
- **Bayesian information criterion**
- **Quantum relative entropy**
- **Information gain in decision trees**
- **Solomon Kullback and Richard Leibler**
- **Information theory and measure theory**
- **Entropy power inequality**
- **Information gain ratio**
- **F-divergence**

Principal Component Analysis

- Eigen analysis, Karhunen-Loeve transform
- Eigenvectors: derived from Eigen decomposition of the scatter matrix
- A projection set that best explains the distribution of the representative features of an object of interest.
- PCA techniques choose a dimensionality-reducing linear projection that maximizes the scatter of all projected samples.

Principal Component Analysis Contd.

- **Let us consider a set of N sample images $\{x_1, x_2, \ldots\ldots, x_N\}$ taking values in n-dimensional image space.**
- **Each image belongs to one of c classes $\{X_1, X_2, \ldots\ldots, X_c\}$.**
- **Let us consider a linear transformation, mapping the original n-dimensional *image space* to m-dimensional *feature space*, where $m < n$.**
- **The new feature vectors $y_k \in R^m$ are defined by the linear transformation –**

$$y_k = W^T x_k \qquad k = 1, 2, \ldots\ldots, N$$

where, $W \in R^{nxm}$ is a matrix with orthogonal columns representing the basis in feature space.

Principal Component Analysis Contd..

- **Total scatter matrix S_T is defined as**

$$S_T = \sum_{k=1}^{N} (x_k - \mu)(x_k - \mu)^T$$

where, *N* is the number of samples , and $\mu \in R^n$ is the mean image of all samples .

$$\sigma_{ij} = E[(x_i - \mu_i)(x_j - \mu_j)]$$

- **The scatter of transformed feature vectors $\{y_1, y_2, \ldots, y_N\}$ is $W^T S_T W$.**

- **In PCA, W_{opt} is chosen to maximize the determinant of the total scatter matrix of projected samples, *i.e.*,**

$$W_{opt} = \arg\max_{W} \left| W^T S_T W \right|$$

where $\{w_i \mid i = 1,2,\ldots,m\}$ is the set of *n* dimensional eigenvectors of S_T corresponding to *m* largest eigenvalues (check proof).

Principal Component Analysis Contd.

- Eigenvectors are called eigen images/pictures and also basis images/facial basis for faces.

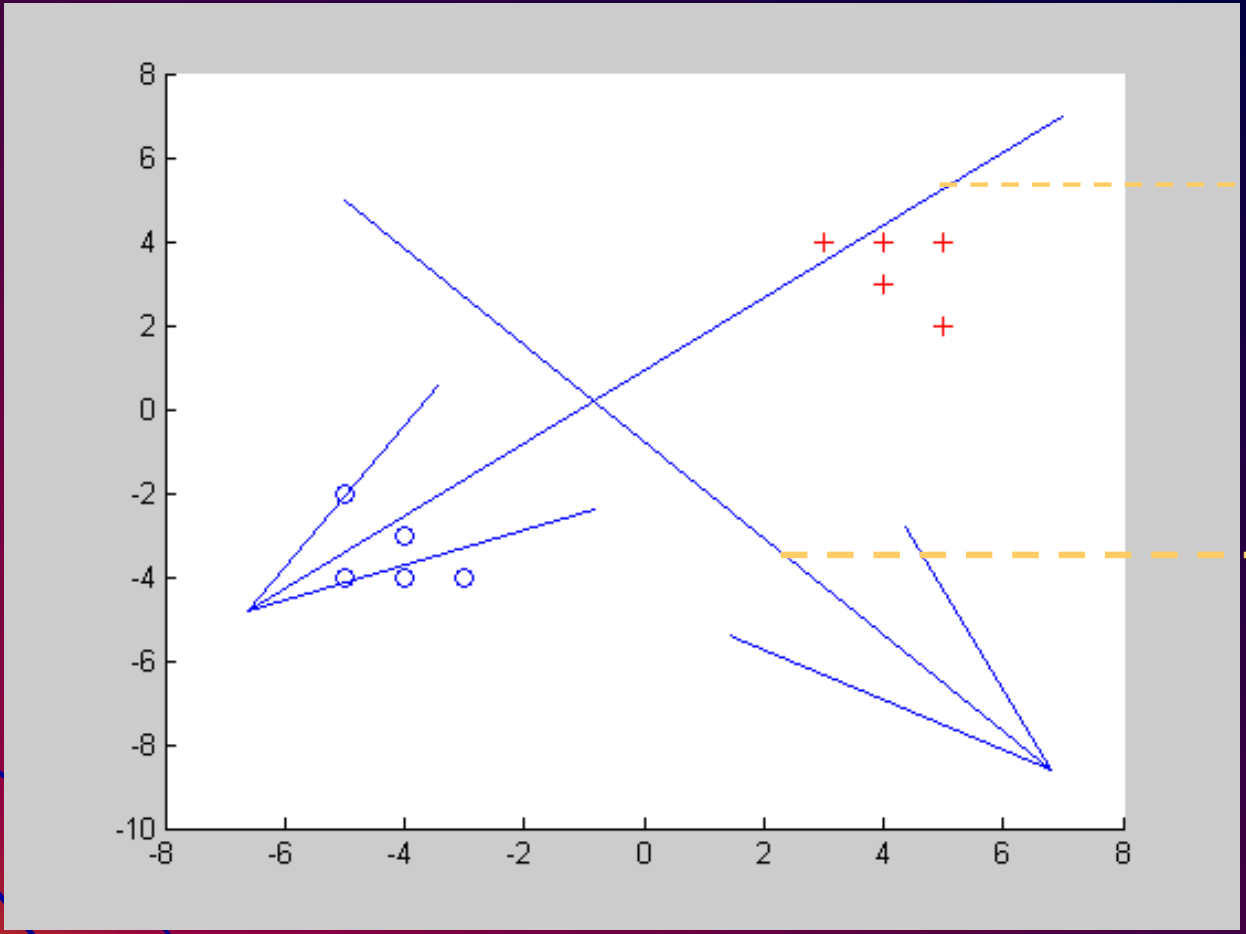
- Any data (say, face) can be reconstructed approximately as a weighted sum of a small collection of images that define a facial basis (eigen images) and a mean image of the face.

- Data form a scatter in the feature space through projection set (eigen vector set)

- Features (eigenvectors) are extracted from the training set without prior class information

➔Unsupervised learning

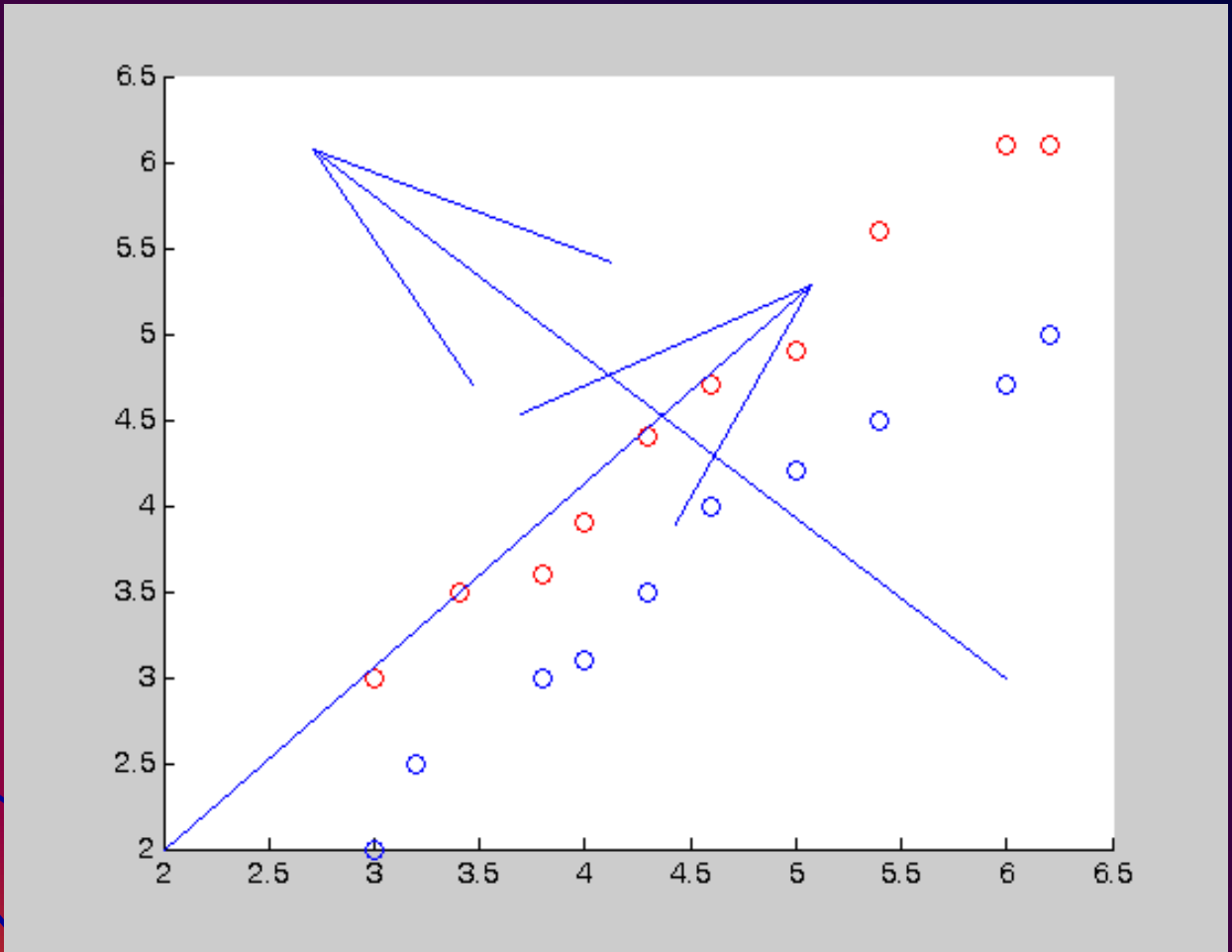
Demonstration of KL Transform



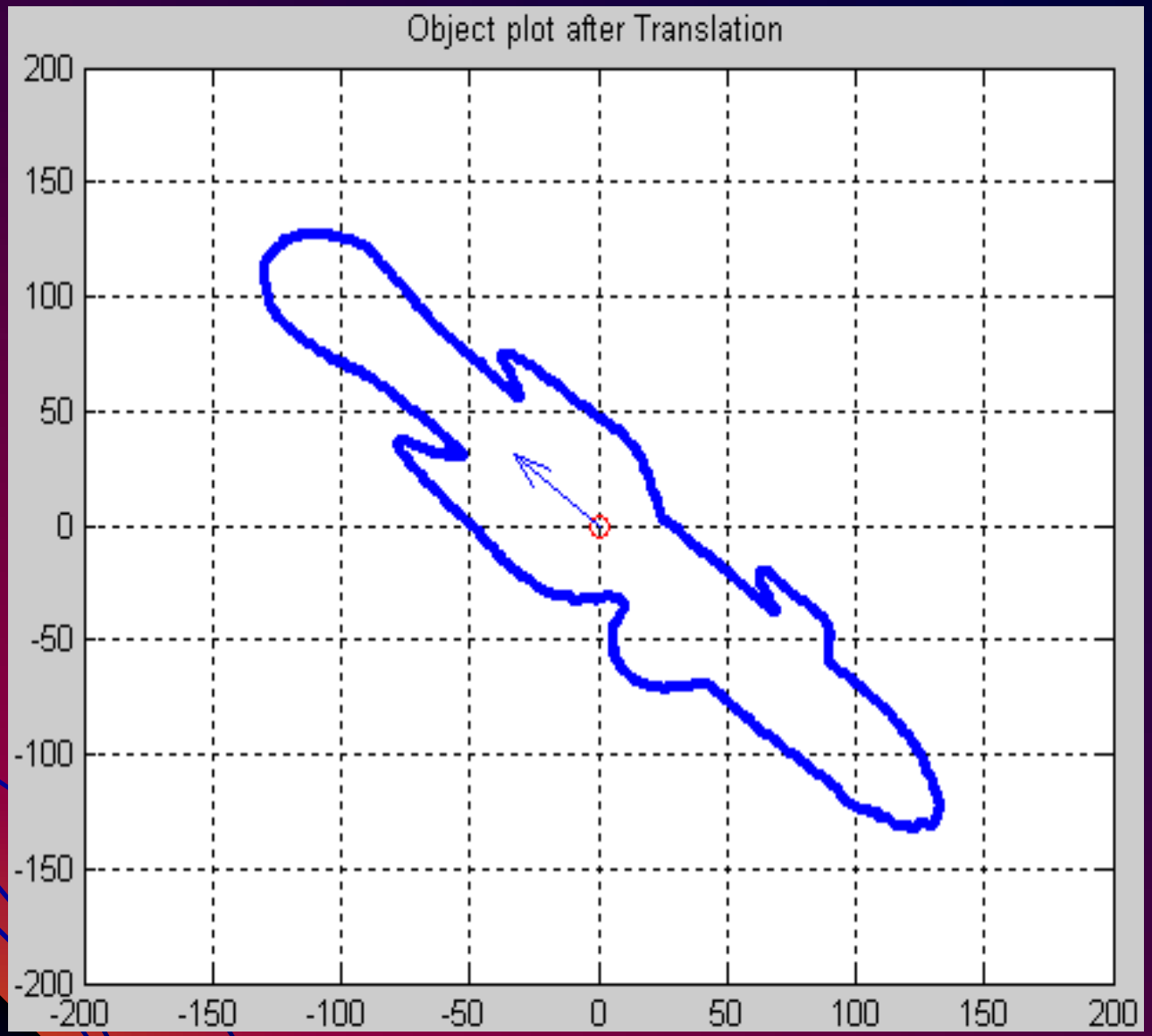
First eigen vector

Second eigen vector

Another One



Another Example



Source: SQUID Homepage

Principal components analysis (PCA) is a technique used to reduce multi-dimensional data sets to lower dimensions for analysis.

The applications include exploratory data analysis and generating predictive models. PCA involves the computation of the eigenvalue decomposition or Singular value decomposition of a data set, usually after mean centering the data for each attribute.

PCA is mathematically defined as an orthogonal linear transformation, that transforms the data to a new coordinate system such that the greatest variance by any projection of the data comes to lie on the first coordinate (called the first principal component), the second greatest variance on the second coordinate, and so on.

PCA can be used for dimensionality reduction in a data set by retaining those characteristics of the data set that contribute most to its variance, by keeping lower-order principal components and ignoring higher-order ones. Such low-order components often contain the "most important" aspects of the data. But this is not necessarily the case, depending on the application.

For a data matrix, X^T, with zero empirical mean (the empirical mean of the distribution has been subtracted from the data set), where each *column* is made up of results for a different subject, and each *row* the results from a different probe. This will mean that the PCA for our data matrix X will be given by:

$$Y = W^T X = \Sigma V^T,$$

where $W\Sigma V^T$ is the singular value decomposition (SVD) of X.

Goal of PCA:

Find some orthonormal matrix W^T, where $Y = W^T X$; such that

$COV(Y) \equiv (1/(n-1))YY^T$ is diagonalized.

The rows of W are the principal components of X, which are also the eigenvectors of COV(X).

Unlike other linear transforms (DCT, DFT, DWT etc.), PCA does not have a fixed set of basis vectors. Its basis vectors depend on the data set.

SVD – the theorem

Suppose M is an m-by-n matrix whose entries come from the field K, which is either the field of real numbers or the field of complex numbers. Then there exists a factorization of the form

$$M = U\Sigma V^*$$

where U is an m-by-m unitary matrix over K, the matrix Σ is m-by-n with nonnegative numbers on the diagonal and zeros off the diagonal, and V* denotes the conjugate transpose of V, an n-by-n unitary matrix over K. Such a factorization is called a *singular-value decomposition of M.*

The matrix V thus contains a set of orthonormal "input" or "analysing" basis vector directions for M.

The matrix U contains a set of orthonormal "output" basis vector directions for M. The matrix Σ contains the singular values, which can be thought of as scalar "gain controls" by which each corresponding input is multiplied to give a corresponding output.

A common convention is to order the values $\Sigma_{i,i}$ in non-increasing fashion. In this case, the diagonal matrix Σ is uniquely determined by M (though the matrices U and V are not).

For p = min(m,n) — U is m-by-p, Σ is p-by-p, and V is n-by-p.

The Karhunen-Loève transform is therefore equivalent to finding the singular value decomposition of the data matrix *X*, and then obtaining the reduced-space data matrix Y by projecting X down into the reduced space defined by only the first *L* singular vectors, W_L:

$$X = W\Sigma V^T; \quad Y = W_L^T X = \Sigma_L V_L^T$$

The matrix W of singular vectors of X is equivalently the matrix W of eigenvectors of the matrix of observed covariances $C = X X^T$ *(find out?) =:*

$$COV(X) = XX^T = W\Sigma\Sigma^T W^T = WDW^T$$

The eigenvectors with the largest eigenvalues correspond to the dimensions that have the strongest correlation in the data set. PCA is equivalent to empirical orthogonal functions (EOF).

PCA is a popular technique in pattern recognition. But it is not optimized for class separability. An alternative is the linear discriminant analysis, which does take this into account. PCA optimally minimizes reconstruction error under the L_2 norm.

PCA by COVARIANCE Method

We need to find a dxd orthonormal transformation matrix W^T, such that:

$$Y = W^T X$$

with the constraint that:
Cov(Y) is a diagonal matrix, and $W^{-1} = W^T$.

$$COV(Y) = E[YY^T] = E[(W^T X)(W^T X)^T]$$
$$= E[(W^T X)(X^T W)] = W^T E[XX^T]W$$
$$= W^T COV(X)W = W^T (WDW^T)W = D$$

$$WCOV(Y) = WW^T COV(X)W = COV(X)W$$

Can you derive from the above, that:

$$[\lambda_1 W_1, \lambda_2 W_2, ..., \lambda_d W_d] =$$
$$[COV(X)W_1, COV(X)W_2, ..., COV(X)W_d]$$

$$\mathbf{X} = \begin{bmatrix} X_1 \\ \vdots \\ X_n \end{bmatrix}$$

and *y* do not fully describe the distribution. A 2×2 covariance matrix is needed; the directions of the arrows correspond to the eigenvectors of this covariance matrix and their lengths to the square roots of the eigenvalues.

are random variables, each with finite variance, then the covariance matrix Σ is the matrix whose (*i*, *j*) entry is the covariance

$$\Sigma_{ij} = \operatorname{cov}(X_i, X_j) = \operatorname{E}\left[(X_i - \mu_i)(X_j - \mu_j)\right]$$

where

$$\mu_i = \operatorname{E}(X_i)$$

is the expected value of the *i*th entry in the vector **X**.[citation needed] In other words, we have

$$\Sigma = \begin{bmatrix} \operatorname{E}[(X_1 - \mu_1)(X_1 - \mu_1)] & \operatorname{E}[(X_1 - \mu_1)(X_2 - \mu_2)] & \cdots & \operatorname{E}[(X_1 - \mu_1)(X_n - \mu_n)] \\ \operatorname{E}[(X_2 - \mu_2)(X_1 - \mu_1)] & \operatorname{E}[(X_2 - \mu_2)(X_2 - \mu_2)] & \cdots & \operatorname{E}[(X_2 - \mu_2)(X_n - \mu_n)] \\ \vdots & \vdots & \ddots & \vdots \\ \operatorname{E}[(X_n - \mu_n)(X_1 - \mu_1)] & \operatorname{E}[(X_n - \mu_n)(X_2 - \mu_2)] & \cdots & \operatorname{E}[(X_n - \mu_n)(X_n - \mu_n)] \end{bmatrix}.$$

The inverse of this matrix, $\sum^{-1}$ is the **inverse covariance matrix**, also known as the **concentration matrix** or **precision matrix**;[1] see precision (statistics). The elements of the precision matrix have an interpretation in terms of partial correlations and partial variances.[citation needed]

Generalization of the variance

[e

The definition above is equivalent to the matrix equality

$$\Sigma = \operatorname{E}\left[(\mathbf{X} - \operatorname{E}[\mathbf{X}])\,(\mathbf{X} - \operatorname{E}[\mathbf{X}])^{\mathrm{T}}\right]$$

This form can be seen as a generalization of the scalar-valued variance to higher dimensions. Recall that for a scalar-valued random variable *X*

$$\sigma^2 = \operatorname{var}(X) = \operatorname{E}[(X - \operatorname{E}(X))^2] = \operatorname{E}[(X - \operatorname{E}(X)) \cdot (X - \operatorname{E}(X))].$$

Indeed, the entries on the diagonal of the covariance matrix $\sum$ are the variances of each element of the vector $\mathbf{X}$.

Conflicting nomenclatures and notations

[e

Nomenclatures differ. Some statisticians, following the probabilist William Feller, call this matrix the **variance** of the random vector X, because it is the natural generalization to higher dimensions of the 1-dimensional variance. Others call it the **covariance matrix**, because it is the matrix of covariances between the scalar components of the vector X. Thus

$$\operatorname{var}(\mathbf{X}) = \operatorname{cov}(\mathbf{X}) = \operatorname{E}\left[(\mathbf{X} - \operatorname{E}[\mathbf{X}])(\mathbf{X} - \operatorname{E}[\mathbf{X}])^{\mathrm{T}}\right].$$

However, the notation for the cross-covariance *between* two vectors is standard:

$$\operatorname{cov}(\mathbf{X}, \mathbf{Y}) = \operatorname{E}\left[(\mathbf{X} - \operatorname{E}[\mathbf{X}])(\mathbf{Y} - \operatorname{E}[\mathbf{Y}])^{\mathrm{T}}\right].$$

The var notation is found in William Feller's two-volume book *An Introduction to Probability Theory and Its Applications*,[2] but both forms are quite standard and there is no ambiguity between them.

The matrix Σ is also often called the variance-covariance matrix since the diagonal terms are in fact variances.

Properties [edit]

For $\Sigma = \mathrm{E}\left[(\mathbf{X} - \mathrm{E}[\mathbf{X}])\,(\mathbf{X} - \mathrm{E}[\mathbf{X}])^{\mathrm{T}}\right]$ and $\boldsymbol{\mu} = \mathrm{E}(\mathbf{X})$, where **X** is a random *p*-dimensional variable and **Y** a random *q*-dimensional variable, the following basic properties apply:[citation needed]

1. $\Sigma = \mathrm{E}(\mathbf{X}\mathbf{X}^{\mathrm{T}}) - \boldsymbol{\mu}\boldsymbol{\mu}^{\mathrm{T}}$
2. Σ is positive-semidefinite and symmetric.
3. $\operatorname{cov}(\mathbf{A}\mathbf{X} + \mathbf{a}) = \mathbf{A}\operatorname{cov}(\mathbf{X})\mathbf{A}^{\mathrm{T}}$
4. $\operatorname{cov}(\mathbf{X}, \mathbf{Y}) = \operatorname{cov}(\mathbf{Y}, \mathbf{X})^{\mathrm{T}}$
5. $\operatorname{cov}(\mathbf{X}_1 + \mathbf{X}_2, \mathbf{Y}) = \operatorname{cov}(\mathbf{X}_1, \mathbf{Y}) + \operatorname{cov}(\mathbf{X}_2, \mathbf{Y})$
6. If *p* = *q*, then $\operatorname{var}(\mathbf{X} + \mathbf{Y}) = \operatorname{var}(\mathbf{X}) + \operatorname{cov}(\mathbf{X}, \mathbf{Y}) + \operatorname{cov}(\mathbf{Y}, \mathbf{X}) + \operatorname{var}(\mathbf{Y})$
7. $\operatorname{cov}(\mathbf{A}\mathbf{X} + \mathbf{a}, \mathbf{B}^{\mathrm{T}}\mathbf{Y} + \mathbf{b}) = \mathbf{A}\operatorname{cov}(\mathbf{X}, \mathbf{Y})\mathbf{B}$
8. If $\mathbf{X}$ and $\mathbf{Y}$ are independent or uncorrelate, then $\operatorname{cov}(\mathbf{X}, \mathbf{Y}) = \mathbf{0}$

where $\mathbf{X}, \mathbf{X}_1$ and $\mathbf{X}_2$ are random *p*×1 vectors, $\mathbf{Y}$ is a random *q*×1 vector, $\mathbf{a}$ is a *q*×1 vector, $\mathbf{b}$ is a *p*×1 vector, and $\mathbf{A}$ and $\mathbf{B}$ are *q*×*p* matrices.

This covariance matrix is a useful tool in many different areas. From it a transformation matrix can be derived, called a whitening transformation, that allows one to completely decorrelate the data[citation needed] or, from a different point of view, to find an optimal basis for representing the data in a compact way[citation needed] (see Rayleigh quotient for a formal proof and additional properties of covariance matrices). This is called principal components analysis (PCA) and the Karhunen-Loève transform (KL-transform).

As a linear operator [edit]

Applied to one vector, the covariance matrix maps a linear combination, **c**, of the random variables, **X**, onto a vector of covariances with those variables: $\mathbf{c}^{\mathrm{T}}\Sigma = \operatorname{cov}(\mathbf{c}^{\mathrm{T}}\mathbf{X}, \mathbf{X})$. Treated as a bilinear form, it yields the covariance between the two linear combinations: $\mathbf{d}^{\mathrm{T}}\Sigma\mathbf{c} = \operatorname{cov}(\mathbf{d}^{\mathrm{T}}\mathbf{X}, \mathbf{c}^{\mathrm{T}}\mathbf{X})$. The variance of a linear combination is then $\mathbf{c}^{\mathrm{T}}\Sigma\mathbf{c}$, its covariance with itself.

Similarly, the (pseudo-)inverse covariance matrix provides an inner product, $\langle c - \mu | \Sigma^{+} | c - \mu \rangle$ which induces the Mahalanobis distance, a measure of the "unlikelihood" of *c*.[citation needed]

Which matrices are covariance matrices? [edit]

From the identity just above, let $\mathbf{b}$ be a $(p \times 1)$ real-valued vector, then

$$\operatorname{var}(\mathbf{b}^{\mathrm{T}}\mathbf{X}) = \mathbf{b}^{\mathrm{T}}\operatorname{var}(\mathbf{X})\mathbf{b},$$

which must always be nonnegative since it is the variance of a real-valued random variable. and the symmetry of the covariance matrix's definition it follows that only a positive-semidefinite matrix can be a covariance matrix.[citation needed] The answer to the converse question, whether *every* symmetric positive semi-definite matrix is a covariance matrix, is "yes." To see this, suppose **M** is a *p*×*p* positive-semidefinite matrix. From the finite-dimensional case of the spectral theorem, it follows that **M** has a nonnegative symmetric square root, that can be denoted by $\mathbf{M}^{1/2}$. Let $\mathbf{X}$ be any *p*×1 column vector-valued random variable whose covariance matrix is the *p*×*p* identity matrix. Then

$$\operatorname{var}(\mathbf{M}^{1/2}\mathbf{X}) = \mathbf{M}^{1/2}(\operatorname{var}(\mathbf{X}))\mathbf{M}^{1/2} = \mathbf{M}.$$

Example of PCA

Samples: $x_1 = \begin{bmatrix} -1 \\ 1 \\ 2 \end{bmatrix}; x_2 = \begin{bmatrix} -2 \\ 3 \\ 1 \end{bmatrix}; x_3 = \begin{bmatrix} 4 \\ 0 \\ 3 \end{bmatrix};$ $\quad X = \begin{bmatrix} -1 & -2 & 4 \\ 1 & 3 & 0 \\ 2 & 1 & 3 \end{bmatrix}$

3-D problem, with N = 3.

Each column is an observation (sample) and each row a variable (dimension),

Mean of the samples: $\mu_x = \begin{bmatrix} 1/3 \\ 4/3 \\ 2 \end{bmatrix};$ $\quad \tilde{x}_1 = \begin{bmatrix} -4/3 \\ -1/3 \\ 0 \end{bmatrix}; \tilde{x}_2 = \begin{bmatrix} -7/3 \\ 5/3 \\ -1 \end{bmatrix}; \tilde{x}_3 = \begin{bmatrix} 11/3 \\ -4/3 \\ 1 \end{bmatrix};$

Method – 1 (easiest)

$\tilde{X} = \begin{bmatrix} -4/3 & -7/3 & 11/3 \\ -1/3 & 5/3 & -4/3 \\ 0 & -1 & 1 \end{bmatrix};$ $\quad$ COVAR = $(\tilde{X}\,\tilde{X}^T)/2 = (1/2)\begin{bmatrix} 62/3 & -25/3 & 6 \\ -25/3 & 14/3 & -3 \\ 6 & -3 & 2 \end{bmatrix}$

Method – 2 (PCA defn.)

$$S_T = \left(\frac{1}{N-1}\right)\sum_{k=1}^{N}(x_k - \mu)(x_k - \mu)^T$$

$$\tilde{x}_1 = \begin{bmatrix} -4/3 \\ -1/3 \\ 0 \end{bmatrix}; \tilde{x}_2 = \begin{bmatrix} -7/3 \\ 5/3 \\ -1 \end{bmatrix}; \tilde{x}_3 = \begin{bmatrix} 11/3 \\ -4/3 \\ 1 \end{bmatrix};$$

C1 =

1.7778	0.4444	0
0.4444	0.1111	0
0	0	0

C2 =

5.4444	-3.8889	2.3333
-3.8889	2.7778	-1.6667
2.3333	-1.6667	1.0000

C3 =

13.4444	-4.8889	3.6667
-4.8889	1.7778	-1.3333
3.6667	-1.3333	1.0000

SigmaC =

20.6667	-8.3333	6.0000
-8.3333	4.6667	-3.0000
6.0000	-3.0000	2.0000

COVAR =
SigmaC/2 =

10.3333	-4.1667	3.0000
-4.1667	2.3333	-1.5000
3.0000	-1.5000	1.0000

Next do SVD, to get vectors.

For a face image with N samples and dimension d (=w*h, very large), we have:

The array X or Xavg of size d*N (N vertical samples stacked horizontally)

Thus XX^T will be of d*d, which will be very large. To perform eigen-analysis on such large dimension is time consuming and may be erroneous.

Thus often X^TX of dimension N*N is considered for eigen-analysis. Will it result in the same, after SVD? Lets check:

$$S = \tilde{X}\,\tilde{X}^{T} = (1/2)\begin{bmatrix} 62/3 & -25/3 & 6 \\ -25/3 & 14/3 & -3 \\ 6 & -3 & 2 \end{bmatrix} = \begin{matrix} 10.3333 & -4.1667 & 3.0000 \\ -4.1667 & 2.3333 & -1.5000 \\ 3.0000 & -1.5000 & 1.0000 \end{matrix}$$

$$S^{m} = \tilde{X}^{T}\,\tilde{X} = \begin{matrix} 0.9444 & 1.2778 & -2.2222 \\ 1.2778 & 4.6111 & -5.8889 \\ -2.2222 & -5.8889 & 8.1111 \end{matrix}$$

Lets do SVD of both:

$$S = X \tilde{X}^T =$$

10.3333	-4.1667	3.0000
-4.1667	2.3333	-1.5000
3.0000	-1.5000	1.0000

U =

-0.8846	-0.4554	-0.1010
0.3818	-0.8313	0.4041
-0.2680	0.3189	0.9091

S =

13.0404	0	0
0	0.6263	0
0	0	0.0000

V =

-0.8846	-0.4554	0.1010
0.3818	-0.8313	-0.4041
-0.2680	0.3189	-0.9091

$$S^m = \tilde{X}^T \tilde{X} =$$

0.9444	1.2778	-2.2222
1.2778	4.6111	-5.8889
-2.2222	-5.8889	8.1111

U =

-0.2060	0.7901	0.5774
-0.5812	-0.5735	0.5774
0.7872	-0.2166	0.5774

S =

13.0404	0	0
0	0.6263	0
0	0	0.0000

V =

-0.2060	0.7901	0.5774
-0.5812	-0.5735	0.5774
0.7872	-0.2166	0.5774

Samples:

Example, where d <> N:

$$x_1 = \begin{bmatrix} -3 \\ -3 \end{bmatrix}; x_2 = \begin{bmatrix} -2 \\ -2 \end{bmatrix}; x_3 = \begin{bmatrix} -1 \\ -1 \end{bmatrix}; x_4 = \begin{bmatrix} 4 \\ 4 \end{bmatrix}; x_5 = \begin{bmatrix} 5 \\ 5 \end{bmatrix}; x_6 = \begin{bmatrix} 6 \\ 7 \end{bmatrix};$$

2-D problem (d=2), with N = 6.

Each column is an observation (sample) and each row a variable (dimension),

X =

-3	**-2**	**-1**	**4**	**5**	**6**
-3	**-2**	**-1**	**4**	**5**	**7**

Mean of the samples:

$$\mu_x = \begin{bmatrix} 3/2 \\ 5/3 \end{bmatrix};$$

XM=

-4.5000	-3.5000	-2.5000	2.5000	3.5000	4.5000
-4.6667	-3.6667	-2.6667	2.3333	3.3333	5.3333

$XM^T * XM =$

42.0278	32.8611	23.6944	-22.1389	-31.3056	-45.1389
32.8611	25.6944	18.5278	-17.3056	-24.4722	-35.3056
23.6944	18.5278	13.3611	-12.4722	-17.6389	-25.4722
-22.1389	-17.3056	-12.4722	11.6944	16.5278	23.6944
-31.3056	-24.4722	-17.6389	16.5278	23.3611	33.5278
-45.1389	-35.3056	-25.4722	23.6944	33.5278	48.6944

$COVAR(X) = XM * XM^T$

=

77.5000	82.0000
82.0000	87.3333

$$\text{COVAR}(X) = XM * XM^T$$

$$= \begin{matrix} 77.5000 & 82.0000 \\ 82.0000 & 87.3333 \end{matrix}$$

U =

$$\begin{matrix} -0.6856 & -0.7280 \\ -0.7280 & 0.6856 \end{matrix}$$

S =

$$\begin{matrix} 164.5639 & 0 \\ 0 & 0.2694 \end{matrix}$$

V =

$$\begin{matrix} -0.6856 & -0.7280 \\ -0.7280 & 0.6856 \end{matrix}$$

$XM^T * XM =$

$$\begin{matrix} 42.0278 & 32.8611 & 23.6944 & -22.1389 & -31.3056 & -45.1389 \\ 32.8611 & 25.6944 & 18.5278 & -17.3056 & -24.4722 & -35.3056 \\ 23.6944 & 18.5278 & 13.3611 & -12.4722 & -17.6389 & -25.4722 \\ -22.1389 & -17.3056 & -12.4722 & 11.6944 & 16.5278 & 23.6944 \\ -31.3056 & -24.4722 & -17.6389 & 16.5278 & 23.3611 & 33.5278 \\ -45.1389 & -35.3056 & -25.4722 & 23.6944 & 33.5278 & 48.6944 \end{matrix}$$

U =

$$\begin{matrix} -0.5053 & -0.1469 & -0.7547 & 0.3882 & 0.0214 & 0.0486 \\ -0.3951 & -0.0654 & 0.3632 & 0.0984 & -0.4091 & 0.7284 \\ -0.2849 & 0.0162 & -0.0433 & -0.3456 & -0.7396 & -0.5002 \\ 0.2660 & 0.4241 & -0.5083 & -0.5306 & -0.1150 & 0.4429 \\ 0.3762 & 0.5057 & -0.0258 & 0.6601 & -0.4043 & -0.0539 \\ 0.5432 & -0.7337 & -0.1938 & 0.0541 & -0.3293 & 0.1332 \end{matrix}$$

S =

$$\begin{matrix} 164.5639 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0.2694 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0.0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0.0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0.0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0.0 \end{matrix}$$

V = U ??

Scatter Matrices and Separability criteria

Scatter matrices used to formulate criteria of class separability:

- **Within-class scatter Matrix**: It shows the scatter of samples around their respective class expected vectors.

$$S_W = \sum_{i=1}^{c} \sum_{x_k \in X_i} (x_k - \mu_i)(x_k - \mu_i)^T$$

- **Between-class scatter Matrix**: It is the scatter of the expected vectors around the mixture mean…..μ is the mixture mean..

$$S_B = \sum_{i=1}^{c} N_i (\mu_i - \mu)(\mu_i - \mu)^T$$

Scatter Matrices and Separability criteria

❖ **Mixture scatter matrix**: It is the covariance matrix of all samples regardless of their class assignments.

$$S_T = \sum_{k=1}^{N} (x_k - \mu)(x_k - \mu)^T = S_W + S_B$$

- The criteria formulation for class separability needs to convert these matrices into a number.
- This number should be larger when between-class scatter is larger or the within-class scatter is smaller.

Several Criteria are..

$$J_1 = tr(S_2^{-1} S_1)$$

$$J_2 = \ln \left| S_2^{-1} S_1 \right| = \ln \left| S_1 \right| - \ln \left| S_2 \right|$$

$$J_3 = tr(S_1) - \mu(trS_2 - c)$$

$$J_4 = \frac{trS_1}{trS_2}$$

Linear Discriminant Analysis

- Learning set is labeled – supervised learning

- Class specific method in the sense that it tries to 'shape' the scatter in order to make it more reliable for classification.

- Select W to maximize the ratio of the between-class scatter and the within-class scatter.

Between-class scatter matrix is defined by-

$$S_B = \sum_{i=1}^{c} N_i (\mu_i - \mu)(\mu_i - \mu)^T$$

μ_i is the mean of class X_i

N_i is the no. of samples in class X_i.

Within-class scatter matrix is:

$$S_W = \sum_{i=1}^{c} \sum_{x_k \in X_i} (x_k - \mu_i)(x_k - \mu_i)^T$$

Linear Discriminant Analysis

- If S_W is nonsingular, W_{opt} is chosen to satisfy

$$W_{opt} = \arg\max \frac{\left|W^T S_B W\right|}{\left|W^T S_W W\right|}$$

$$W_{opt} = [w_1, w_2, \ldots, w_m]$$

$\{w_i \mid i = 1,2,\ldots,m\}$ is the set of eigenvectors of S_B and S_W corresponding to m largest eigen values.i.e.

$$S_B w_i = \lambda_i S_W w_i$$

- There are at most $(c-1)$ non-zero eigen values. So upper bound of m is $(c-1)$.

Linear Discriminant Analysis

S_W is singular most of the time. It's rank is at most N-c

Solution – Use an alternative criterion.

- Project the samples to a lower dimensional space.
- Use PCA to reduce dimension of the feature space to N-c.
- Then apply standard FLD to reduce dimension to c-1.

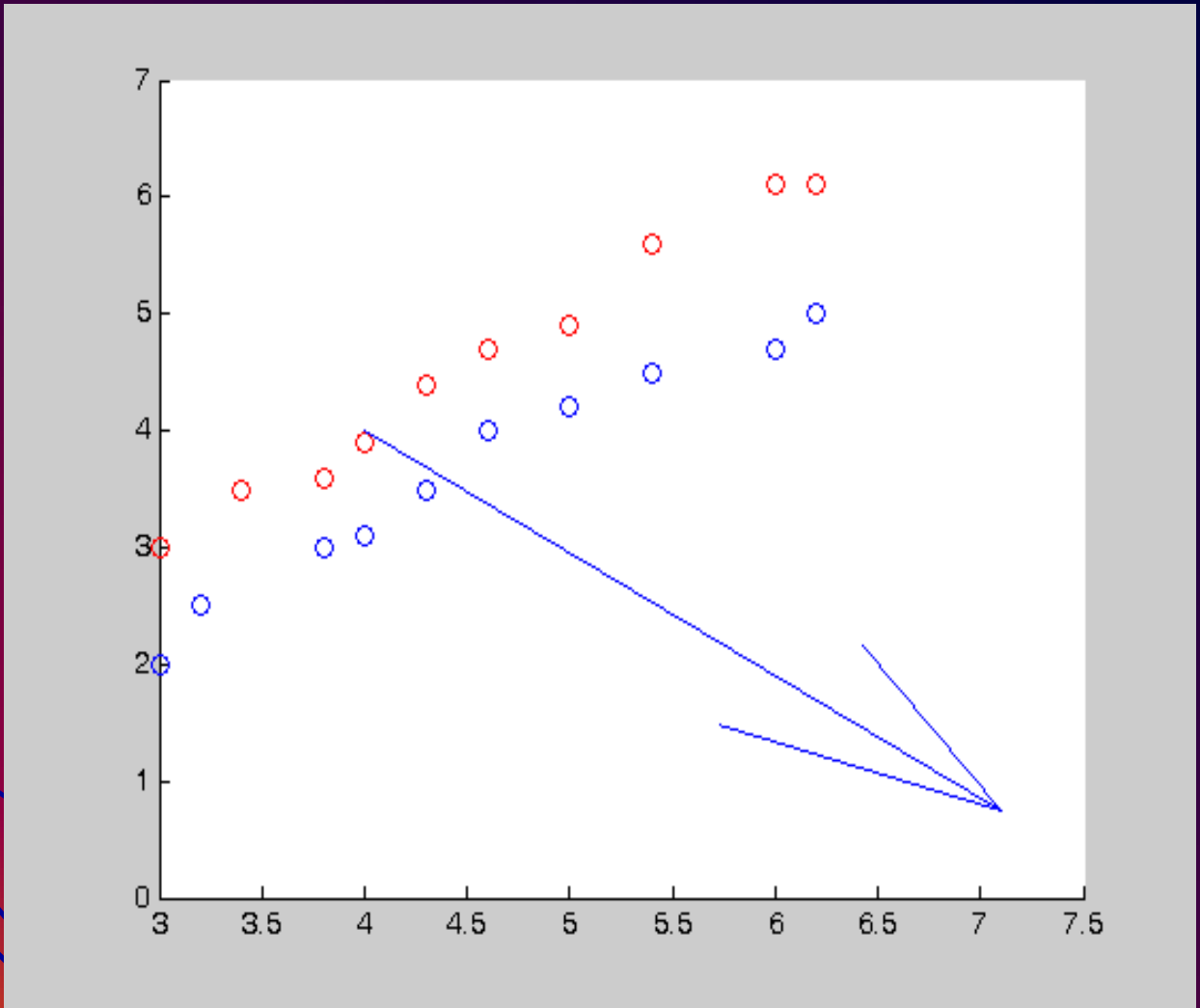
W_{opt} is given by

$$W_{opt} = W_{fld}^T W_{pca}^T$$

$$W_{pca} = \arg\max_{W} \left| W^T S_T W \right|$$

$$W_{fld} = \arg\max_{W} \frac{\left| W^T W_{pca}^T S_B W_{pca} W \right|}{\left| W^T W_{pca}^T S_W W_{pca} W \right|}$$

Demonstration for LDA



Hand workout EXAMPLE:

Data Points:

1	2	3	5	4	6	8	-2	-1	1	3	4	2	5
1	2	3	4	5	6	7	3	4	5	6	7	8	9

Class:

1	1	1	1	1	1	1	2	2	2	2	2	2	2

Lets try **PCA** first :

Overall data mean: 2.9286
5.0000

COVAR of the mean-subtracted data:

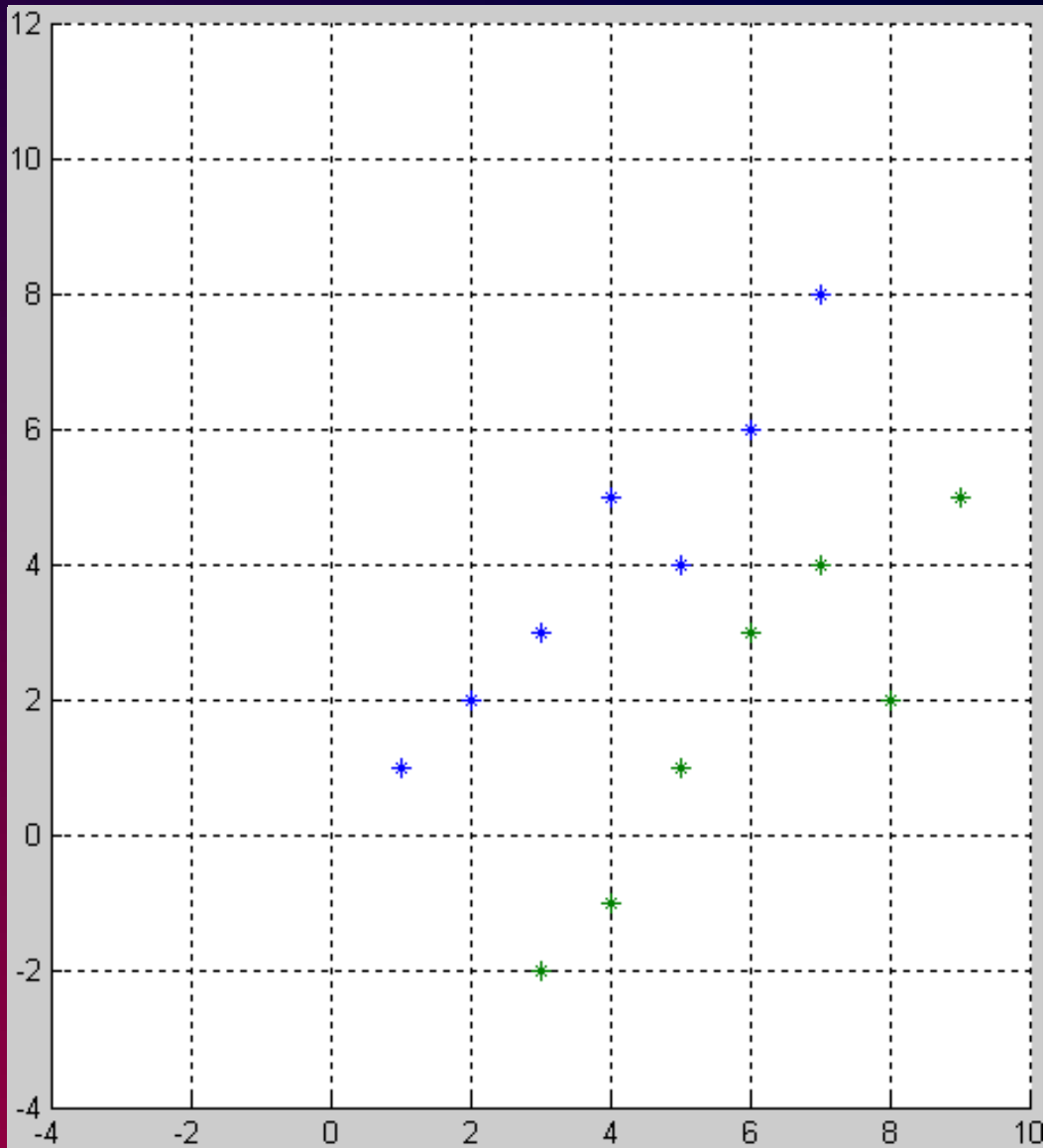
7.3022 3.3077
3.3077 5.3846

Eigenvalues after SVD of above:

9.7873 2.8996

Finally, the eigenvectors:

-0.7995 -0.6007
-0.6007 0.7995



Same EXAMPLE for LDA :

Data Points:	1	2	3	5	4	6	8	-2	-1	1	3	4	2	5
	1	2	3	4	5	6	7	3	4	5	6	7	8	9
Class:	1	1	1	1	1	1	1	2	2	2	2	2	2	2

S_w = **10.6122 8.5714**
8.5714 8.0000

S_b = **20.6429 -17.00**
-17.00 14.00

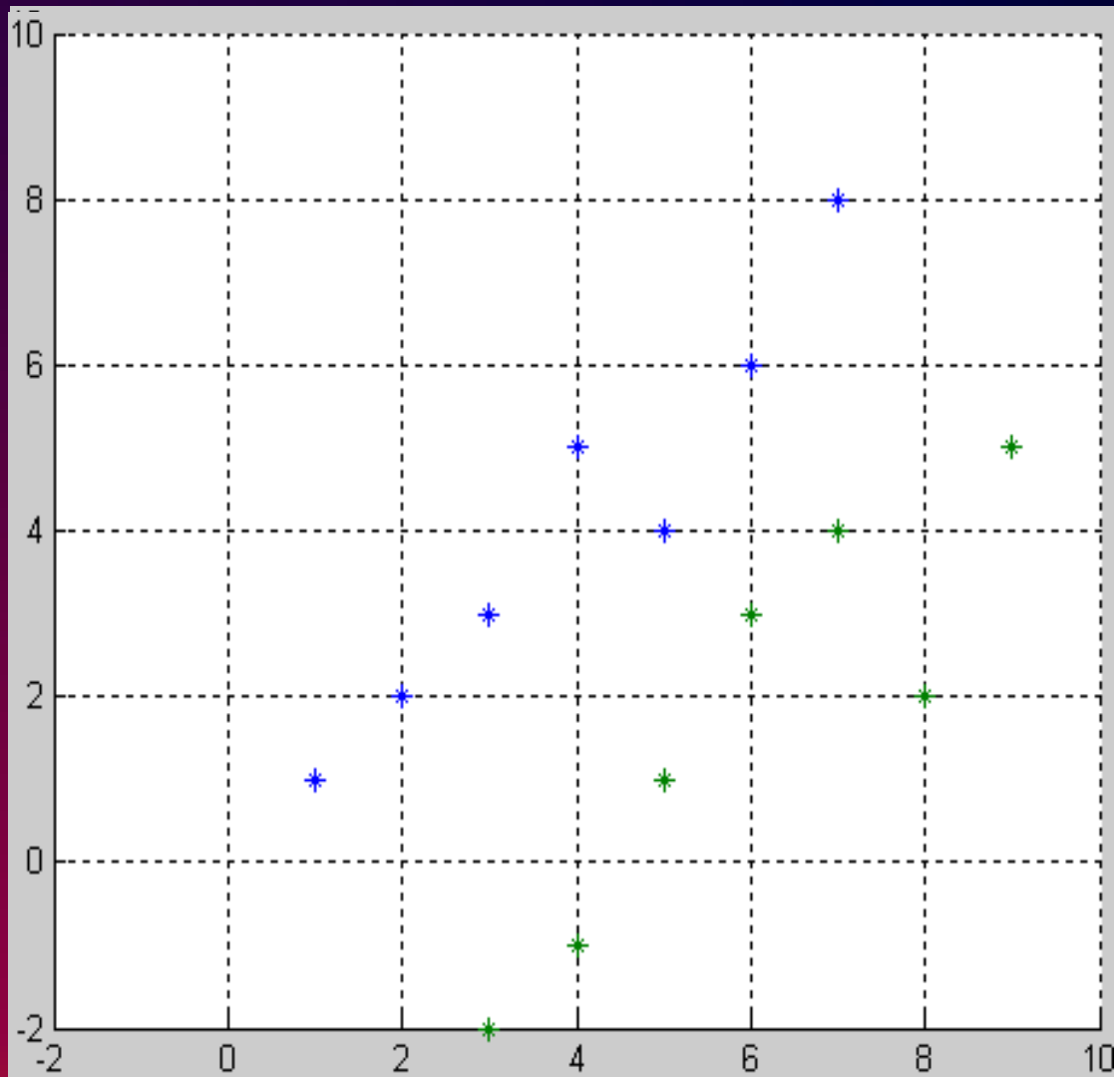
INV(S_w) . S_b = **27.20 -22.40**
-31.268 25.75

Perform Eigendecomposition on above:

Eigenvalues of $S_w^{-1} S_b$: **53.687**
0

Eigenvectors:

- 0.7719 0.6357
0.6357 0.7719

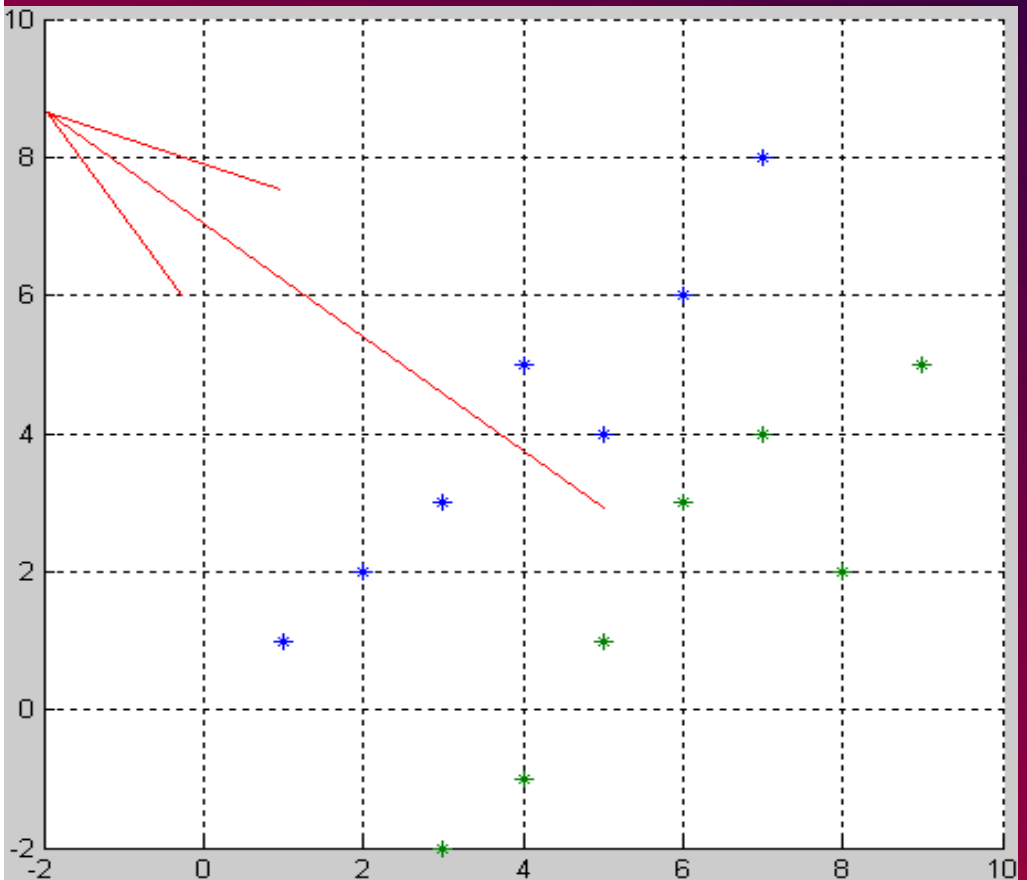


S_w = 10.6122 8.5714
8.5714 8.0000

S_b = 20.6429 - 17.00
- 17.00 14.00

Eigenvalues of $S_w^{-1} S_b$: 53.687
0

Eigenvectors: - 0.7719 0.6357
0.6357 0.7719

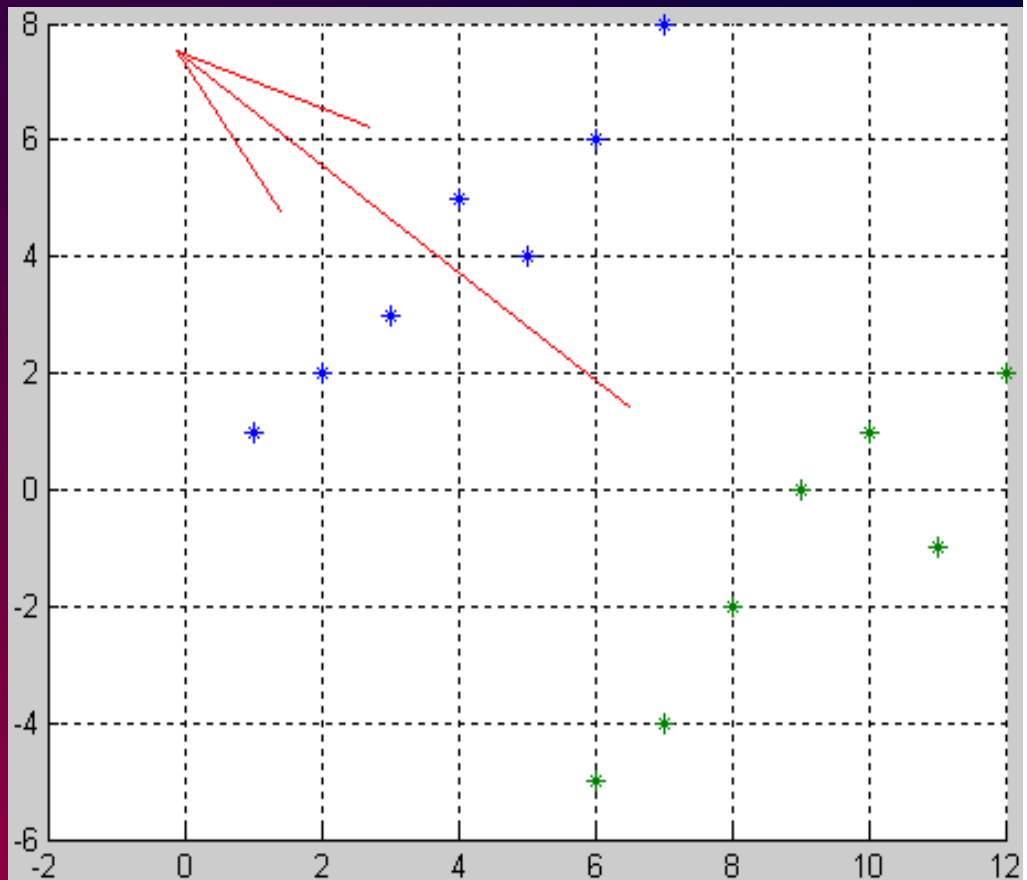


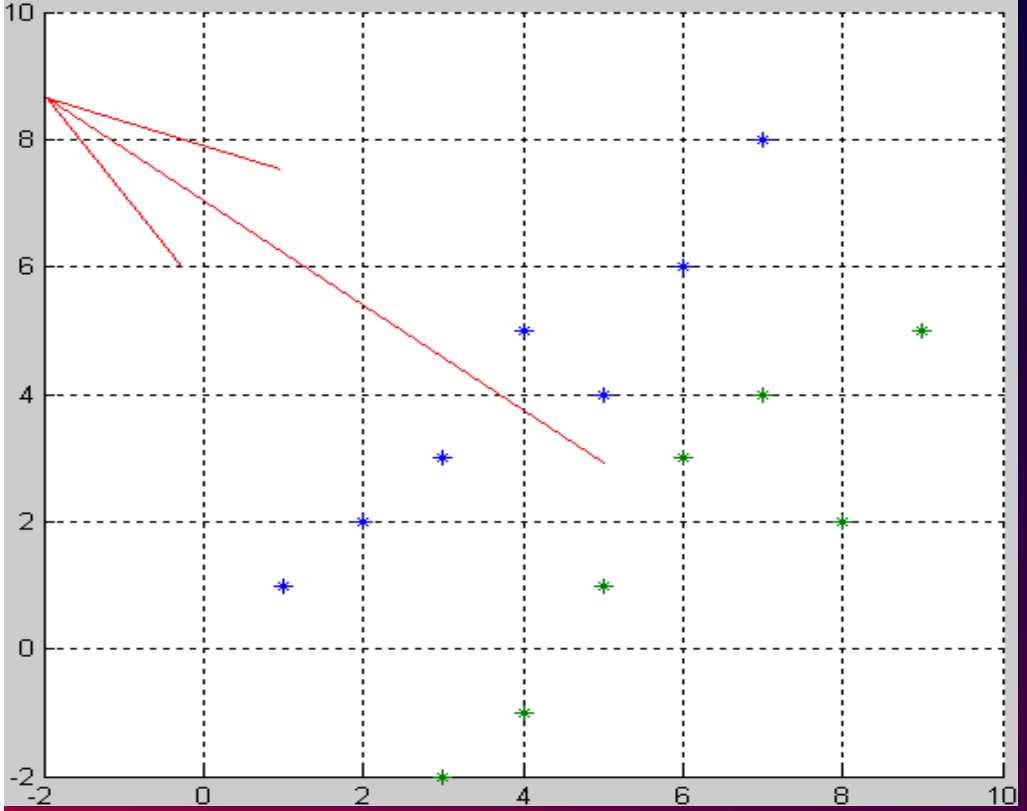
S_w = 10.6122 8.5714
8.5714 8.0000

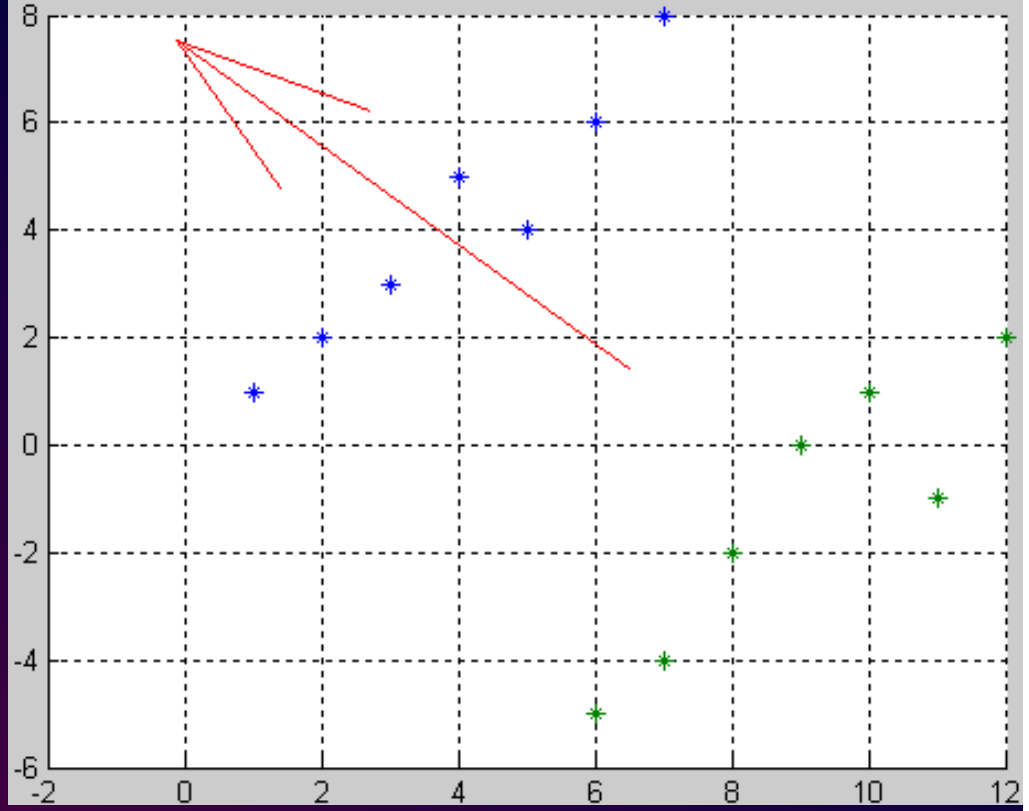
S_b = 203.143 - 95.00
- 95.00 87.50

Eigenvalues of $S_w^{-1} S_b$: 297.83
0.0

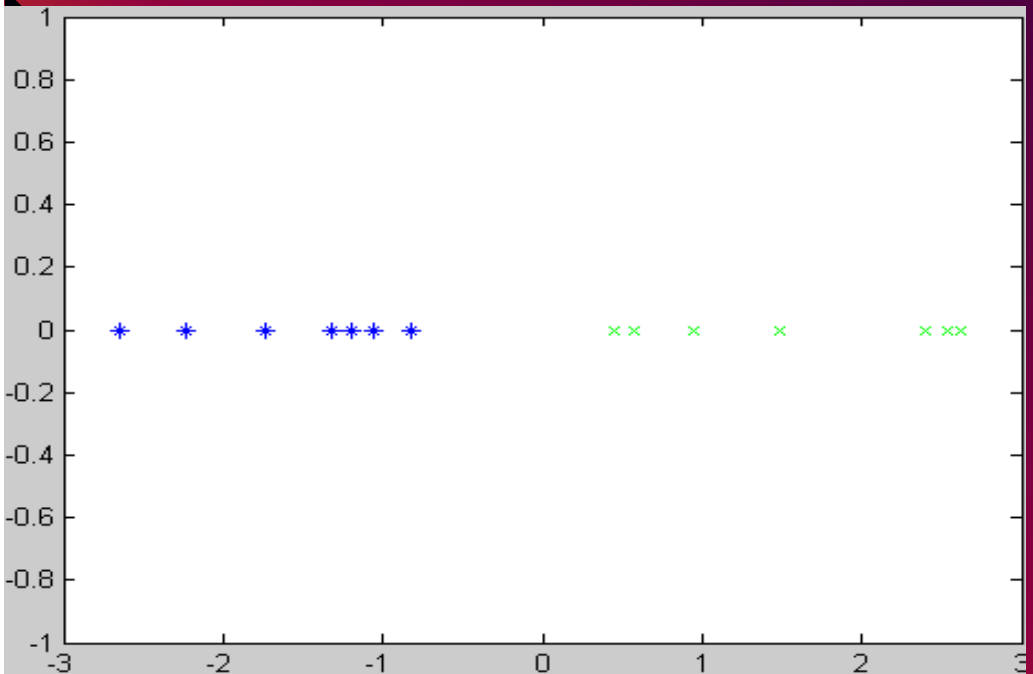
Eigenvectors: -0.7355 -0.6775
0.6775 0.7355

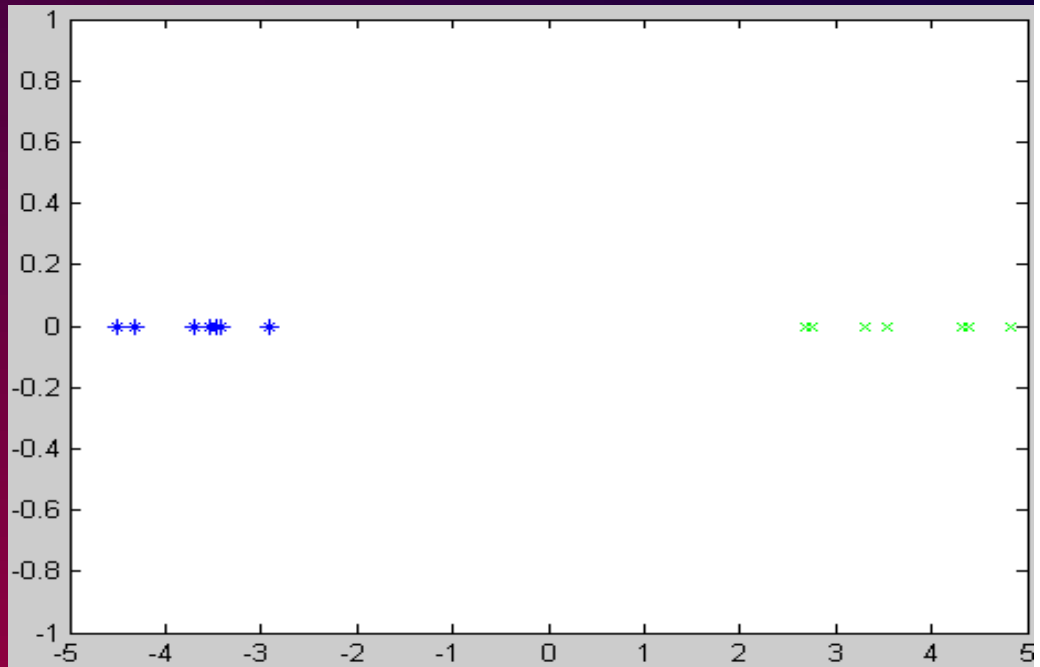






After linear projection, using LDA:





Same EXAMPLE for LDA, with C = 3:

Data Points:

1	2	3	5	4	6	8	-2	-1	1	3	4	2	5
1	2	3	4	5	6	7	3	4	5	6	7	8	9

Class:

1	1	1	2	2	3	3	1	1	1	2	2	3	3

S_w = **8.0764 - 2.125**
- 2.125 4.1667

S_b = **56.845 52.50**
52.50 50.00

INV(S_w) . S_b =

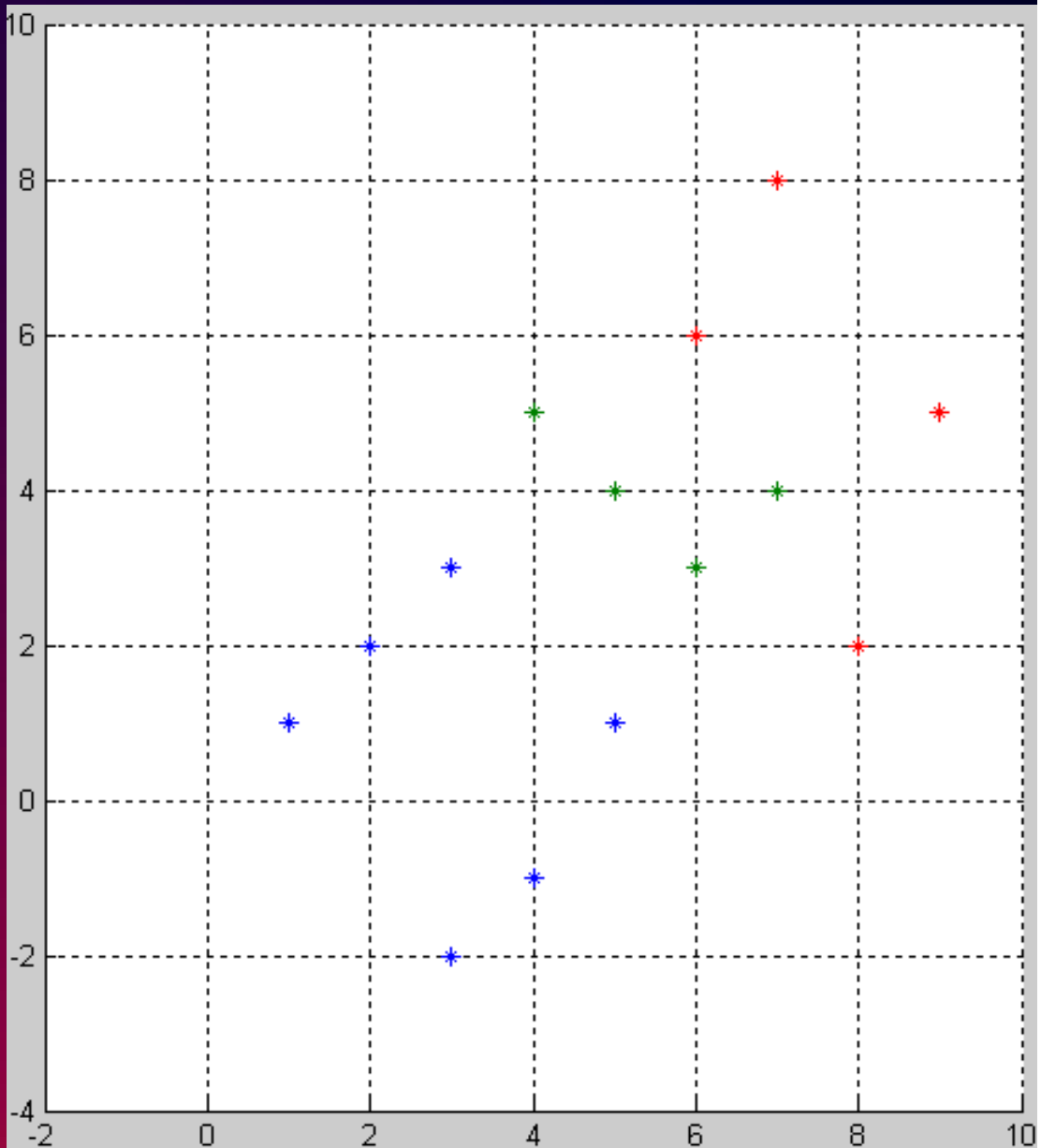
11.958 11.155
18.7 17.69

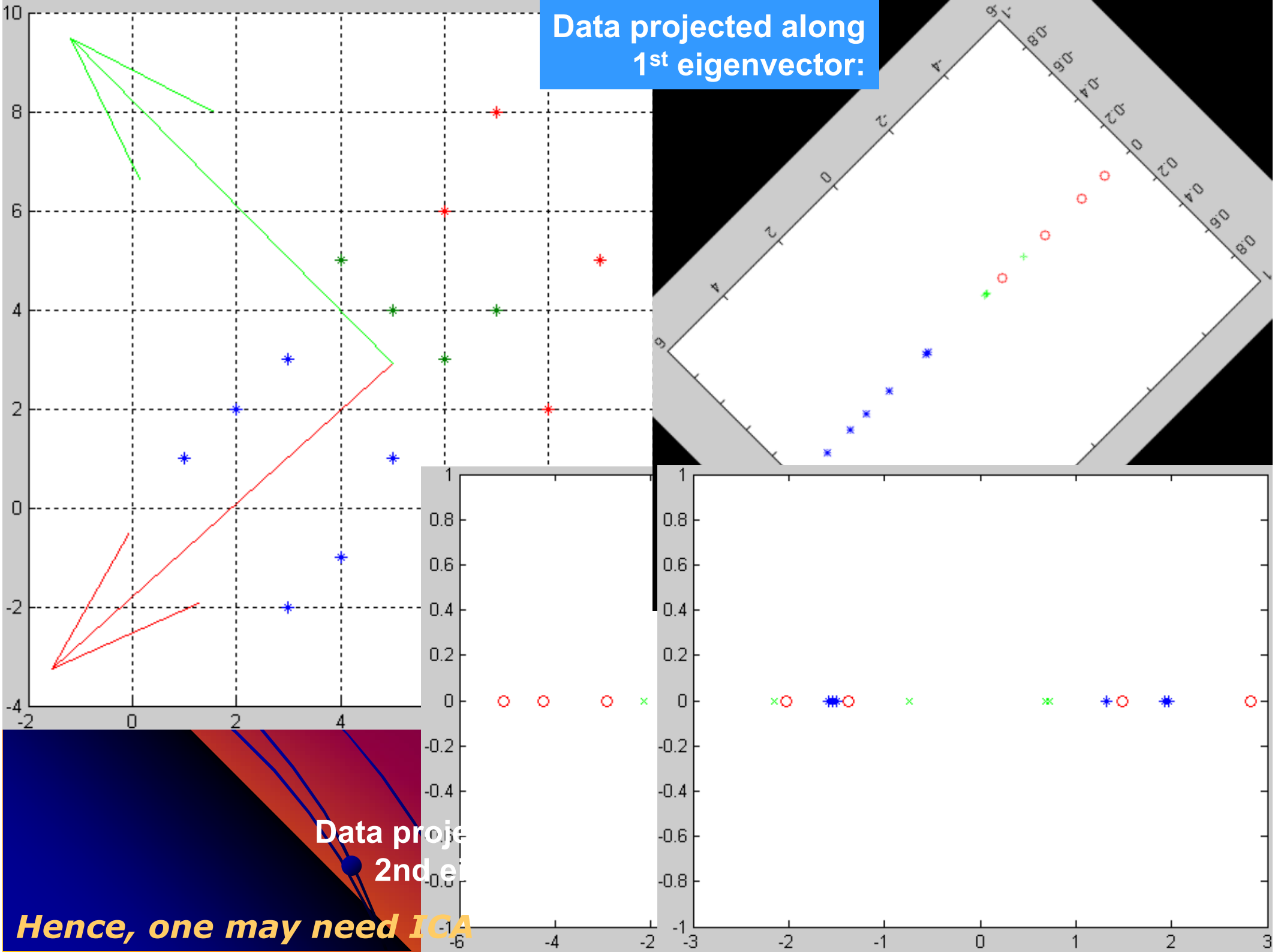
Perform Eigendecomposition on above:

Eigenvalues of $S_w^{-1} S_b$: **30.5**
0.097

Eigenvectors:

- 0.728 - 0.69
- 0.69 0.728




Data projected along
1st eigenvector:
Data proje
2nd e
Hence, one may need ICA

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

- **Neuro-fuzzy (soft computing) concepts**
- **Multi-classifier Combination – decision and feature fusion**
- **Reinforcement learning**
- **Learning from small data sets**
- **Generalization capabilities**
- **Evolutionary Computations**
- **Genetic algorithms**
- **Pervasive computing**
- **Neural dynamics**
- **Support Vector machines - kernel methods**
- **Modern ML methods – semi-supervised, transfer learning, domain adaptation….**

REFERENCES

- "Pattern Recognition: Statistical. Structural and Neural Approaches"; Robert J. Schallkoff; John Wiley and Sons; 1992+.

- Pattern Classification; Duda R.O., Hart P.E. & D. G. Stork:. John Wiley and Sons, Singapore (2001).

- Statistical pattern Recognition; S. Fukunaga; Academic Press, 2000.

- Bishop – PR