Notation Preliminaries

Unknown data vectors: $\underline{x}$

Prototypes:

$\underline{y}_m^{(k)}$ is the m-th prototype that belongs to class S_k

k = class index

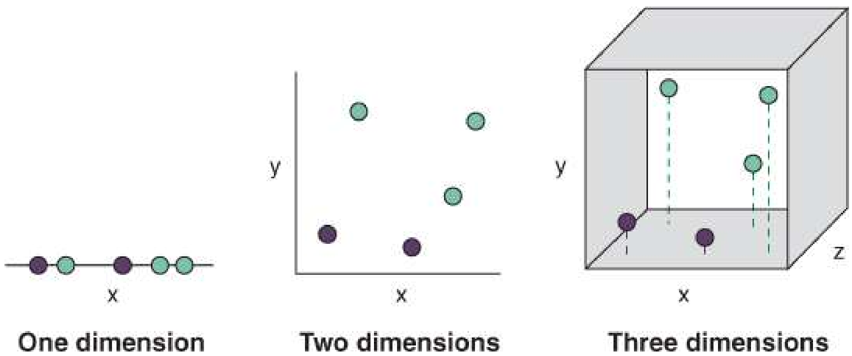
m_k = no. of prototypes in S_k

$\underline{y}_m^{(k)}$, m = 1,2, … , m_k defines all prototypes of class S_k

Prototype $\underline{y}_m^{(k)} = (y_{1m}^{(k)}, y_{2m}^{(k)}, y_{3m}^{(k)}, \ldots, y_{Nm}^{(k)})$

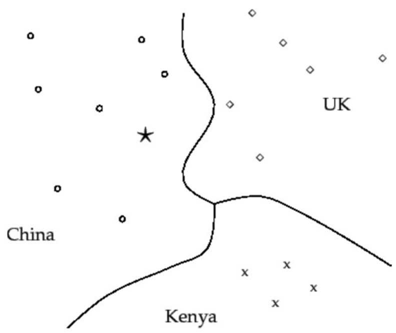
<u>Feature Space (Dimensionality N) (in general, input data)</u>

Unknown $\underline{x}=(x_1, x_2, x_3, \ldots, x_N)^T$



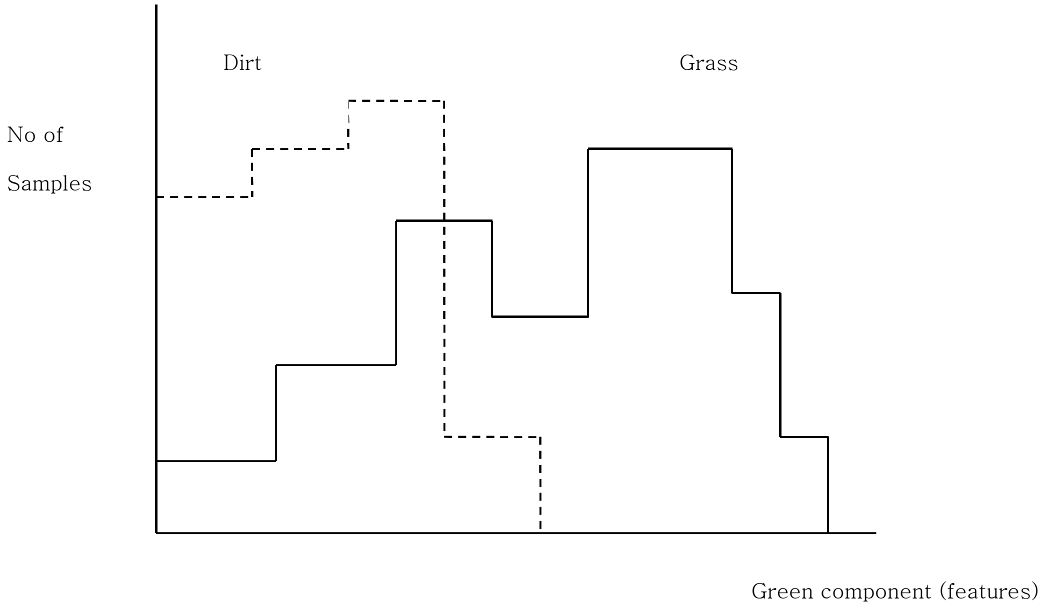
<u>Classification Space (Dimensionality K)</u>

S_k, k=1,2,3, … K defines all classes.



Introductory Example

- Classify dirt and grass regions from aerial pictures or images.



Decision Rule if $x<5.5$, $x \in S_1 = dirt$

If $x \geq 5.5$, $x \in S_2 = grass$

- Better than randomly assigning class
- Will have classification errors? YES

How to improve?

1. Incorporate statistics of samples
2. Add more features

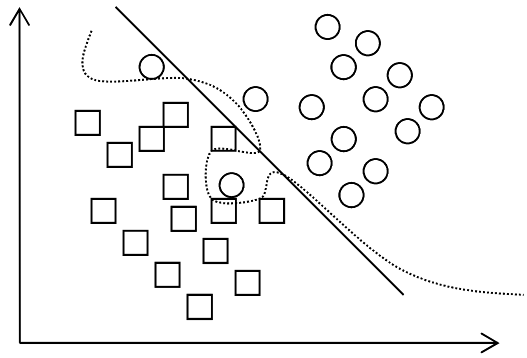
   Possible features

   i. Blue
   ii. Red
   iii. Frequency: to get a feature, define some average frequency magnitude over the image, $\bar{g}$.
   iv. Texture regularity

For 2 features

$N = 2$

Sample: $\underline{x} = \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} = \begin{bmatrix} \text{green component} \\ \text{texture regularity} \end{bmatrix}$



2D feature space

Feature Space	Decision Surface	Linear Case
1-D	Point (0-D)	
2-D	1-D (curve)	Line
3-D	2-D (surface)	Plane
N-D	(N-1)-D surface	Hyperplane (N-1)-D

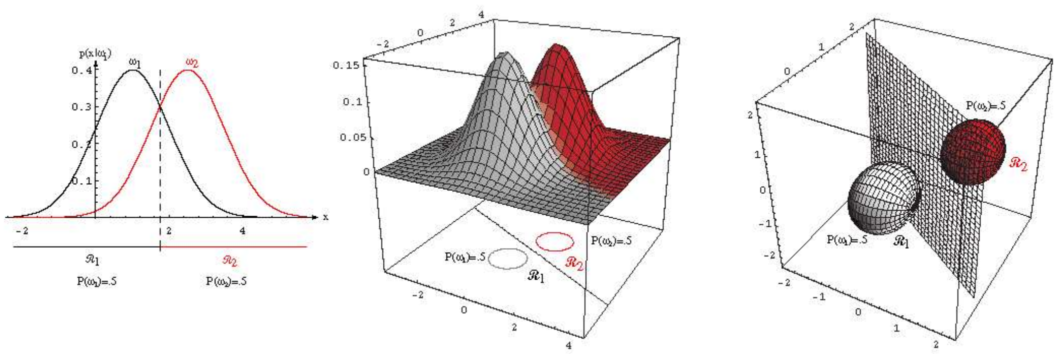


Figure 2.10: If the covariances of two distributions are equal and proportional to the identity matrix, then the distributions are spherical in d dimensions, and the boundary is a generalized hyperplane of $d-1$ dimensions, perpendicular to the line separating the means. In these 1-, 2-, and 3-dimensional examples, we indicate $p(\mathbf{x}|\omega_i)$ and the boundaries for the case $P(\omega_1) = P(\omega_2)$. In the 3-dimensional case, the grid plane separates $\mathcal{R}_1$ from $\mathcal{R}_2$.

Arbitrary 3D Decision Boundaries

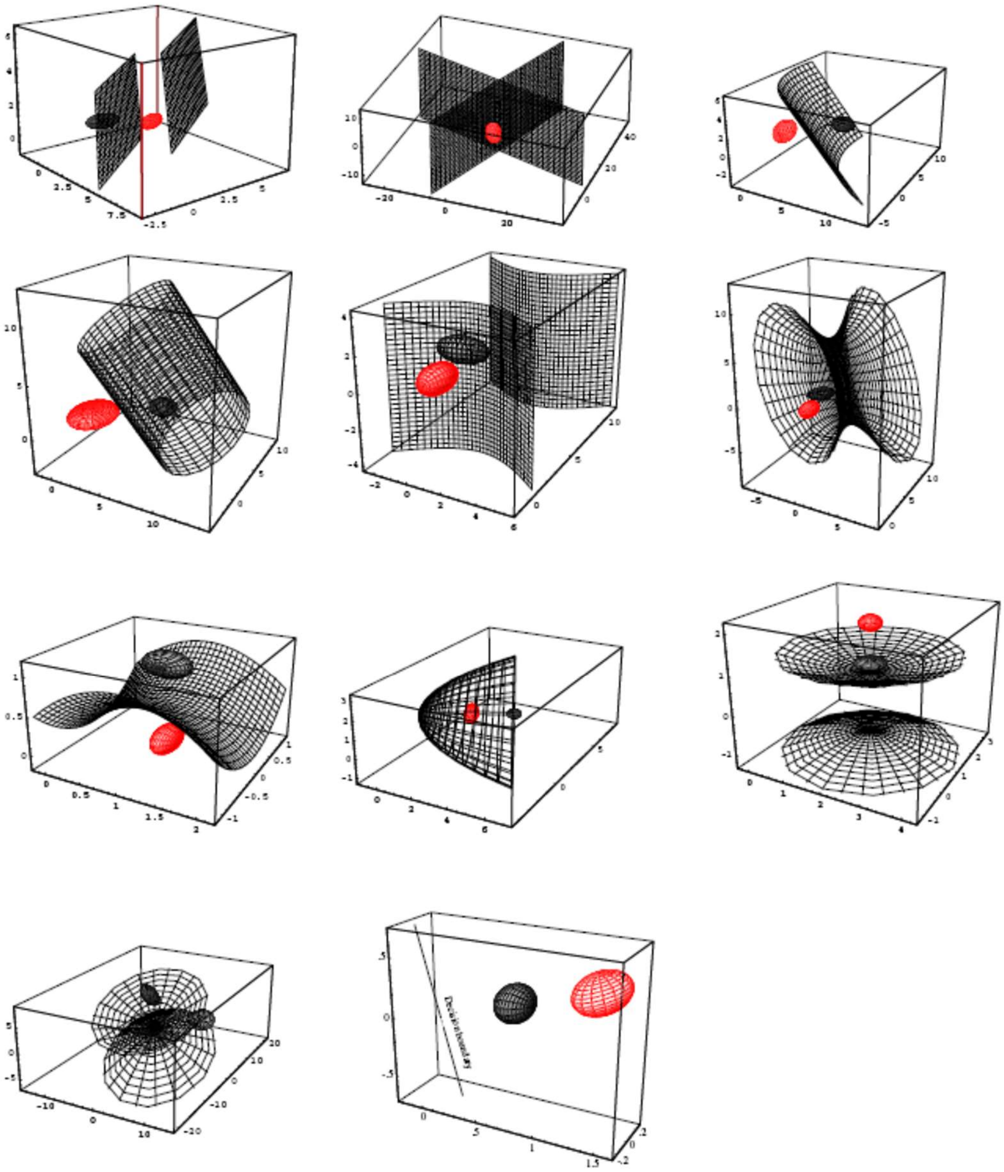


Figure 2.15: Arbitrary three-dimensional Gaussian distributions yield Bayes decision boundaries that are two-dimensional hyperquadrics. There are even degenerate cases in which the decision boundary is a line.

Inverse Crime

- Test performance on new data, not on the training set (no inverse crime)

Note

1. Important to select good features

2. Important to have a good classifier

3. Performance on new patterns (unknowns)
   A. How well does the system generalize?
   B. How representative is the training set of the actual unknown data?
   C. What is the inherent dimensionality of the problem?

4. Can selection of a decision boundary be automated? Training or Learning

5. To what degree are underlying statistics of the samples (features) known? How can they be used to improve the classifier's performance? -> Statistical classification

6. Can these algorithms be implemented in parallel hardware for fast execution? Artificial neural networks

7. What if you don't know what classes the prototypes belong to? Unsupervised classification and Unsupervised learning.

Data Normalization

Normalization is often required to have the desired meaning.

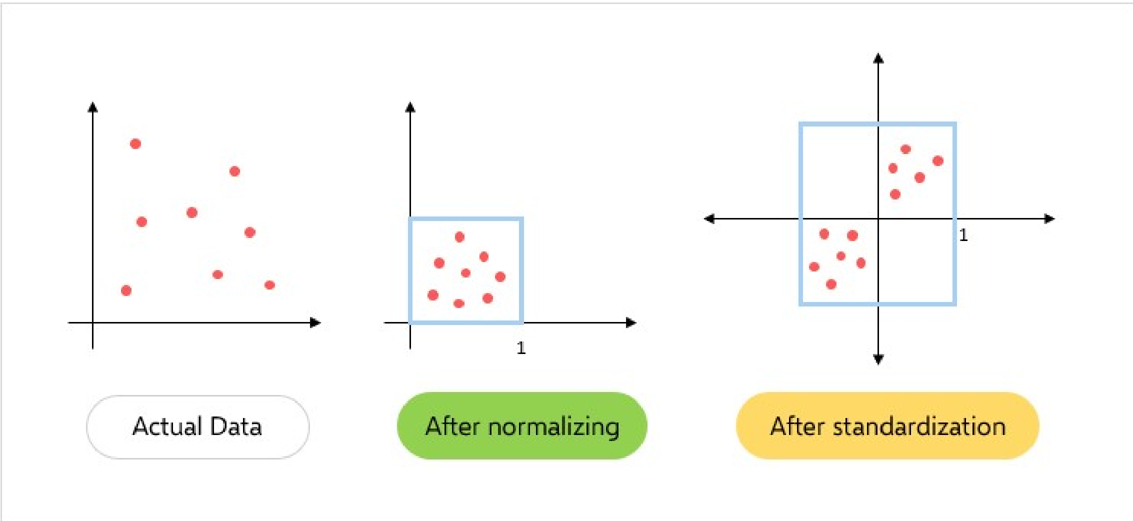
$$x_{std}^{(i)} = \frac{x^{(i)} - \mu_x}{\sigma_x}$$

$$x_{norm}^{(i)} = \frac{x^{(i)} - \mathbf{x}_{min}}{\mathbf{x}_{max} - \mathbf{x}_{min}}$$

standardization

min-max scaling (“normalization”)

	input	standardized	normalized
0	0	-1.46385	0.0
1	1	-0.87831	0.2
2	2	-0.29277	0.4
3	3	0.29277	0.6
4	4	0.87831	0.8
5	5	1.46385	1.0



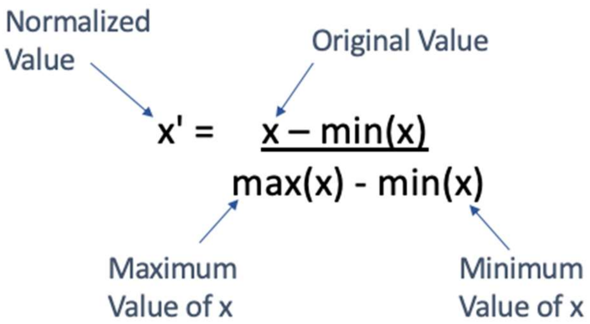
Example 1: Extremal Value Normalization

$$x_r^{'} = \frac{x_r}{a_r}$$

where $a_r = \max_{k,m}\{y_{r_m}^{(k)}\} - \min_{k,m}\{y_{r_m}^{(k)}\}$

This is extremal value normalization (sensitive to noise)

Example 2: Min-Max Normalization



Example 3: Standard Deviation Normalization

Less sensitive to noise

Let $a_r = \sigma_r$

$$\sigma_r^2 = \frac{\sum_{k=1}^{K}\sum_{m=1}^{M_k}[y_{r_m}^{(k)} - \overline{y}_r]^2}{\sum_{k=1}^{K} M_k}$$

where $\overline{y_r} = [\sum\sum y_{r_m}^{(k)}] / \sum_{k=1}^{K} M_k$

This is standard deviation normalization.