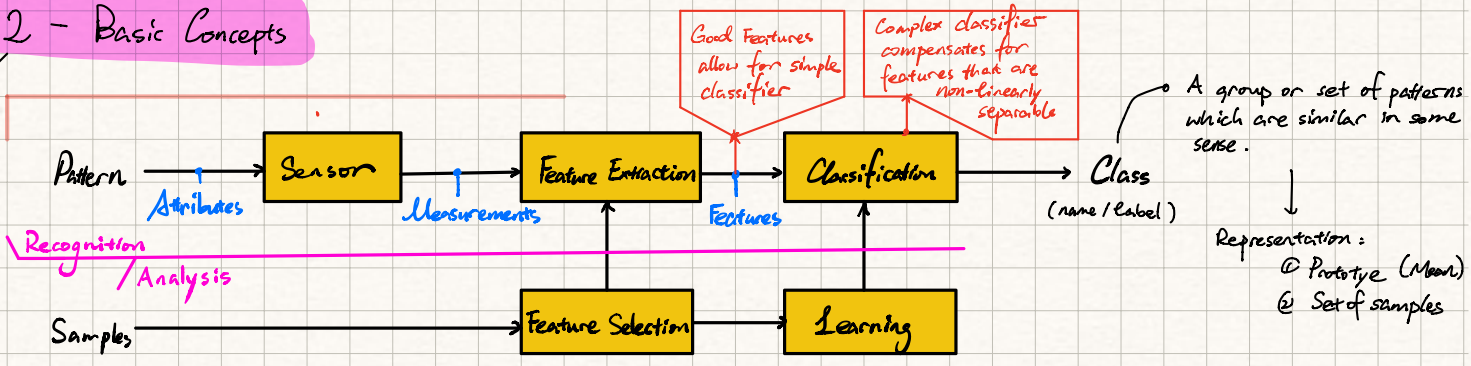
§ 2 - Basic Concepts



× Classification Types (Three)

① **Known probability model** for each class. → Approach: **Statistical decision theory** ⇒ find optimal classifier by **minimizing prob. of error.**

② **Known class labels** for the samples. → Approach: ① **Learn empirical probability model** based on samples
② **Derive classifiers directly** from **distribution** of samples in feature space.

③ **No known label info** for the sample → Need to determine # of classes & class definition.

↳ Refer as **Clustering** problem.

⇓

Approach: look for **naturally occuring** orders, groupings of clusters in data.

× Summary:

Two Patterns are **Similar** <= (if) sharing **common properties**

⇓

(vector space rep):

Closeness in feature space

↑

measured by **distance metric** $d(\vec{y}_1, \vec{y}_2)$.

§3 - Pattern Representation Random Vectors

- In Statistical Pattern Rec,

Pattern ⇐ **Random Vector** : Vector of **random variables**

× Random Variables

- a single scalar that's random (x)
- Discrete : random integer
- Continuous : random real #.
- Characterized by:

CDF : $F_x(\tau) = Pr(x < \tau)$

PDF : $p(x)$, where:

$$F_x(\tau) = Pr(x < \tau) = \int_{-\infty}^{\tau} p(x)\,dx$$

with

$$\int_{-\infty}^{\infty} p(x)\,dx = 1$$

Expectation :

$$E[f(x)] = \int_{-\infty}^{\infty} p(x)\,f(x)\,dx$$

Choice of $f(x)$:

- Mean : $\mu_x = E[x]$
- Variance : $\sigma_x^2 = E[x^2] - E[x]^2 = E[(x - E[x])^2]$

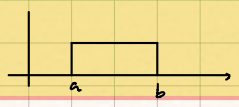
× Common Distribution Models

Gaussian:

$$p(x) = \frac{1}{\sqrt{2\pi}\sigma_x} e^{-\frac{1}{2}\left(\frac{x-\mu_x}{\sigma_x}\right)^2}$$

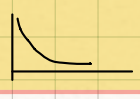
Uniform:

$$p(x) = \begin{cases} 0 & x < a \\ \frac{1}{b-a} & a \le x \le b \\ 0 & b < x \end{cases}$$



Exponential:

$$p(x) = \begin{cases} 0 & x < 0 \\ \lambda e^{-\lambda x} & x \ge 0 \end{cases}$$



× Joint Statistics

× relationship of 2 random variables x & y

Joint Probability Distribution $p(x,y)$:

$$Pr(x < \alpha, y < \beta) = \int_{-\infty}^{\alpha}\int_{-\infty}^{\beta} p(x,y)\,dx\,dy$$

Marginal Distribution $p(x)$:

$$p(x) = \int_{-\infty}^{\infty} p(x,y)\,dy$$

Expectation :

$$E[f(x,y)] = \int_{-\infty}^{\infty}\int_{-\infty}^{\infty} p(x,y)\,f(x,y)\,dx\,dy$$

$$E[x+y] = E[x] + E[y]$$

Correlation : (variance: $E[(x-\mu_x)^2]$)

$$E[(x-\mu_x)(y-\mu_y)]$$

Normalized by std. deviation.

Correlation Coefficient:

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

measures ability to predict 'y' as linear func of 'x'.

- $\rho_{x,y} = 0$ ⇒ No predictability (uncorrelated)
- $\rho_{x,y} = \pm 1$ ⇒ Perfect predictability.
  (x & y deterministically correlated)

Independance

if $p(x,y) = p(x)p(y)$

⇒ knowing x tells nothing about y.

Uncorrelated

if $E[xy] = E[x]E[y]$

Independence ⇒ Uncorrelated

Uncorrelated ⇏ Independence

Conditional Statistics. → assessing unknown (identity of pattern) conditioned on known (measurement)

$p(x|y)$: PDF for x conditioned on y (measurement)

$p(x|A)$: PDF for x given event A (class).

└ distribution of measurements given a class

[Likelihood Distribution]

$p(A|x)$: [Posterior Distribution]

└ distribution of class given measurements.

Bayes Rule

$$p(x|y) = \frac{p(x,y)}{p(y)} = \frac{p(y|x)\,p(x)}{p(y)}$$

($p(x,y)$: Joint; $p(x|y)$: conditional prob.; $p(y)$: marginal)

$$\Rightarrow \quad p(x|A) = \frac{p(A|x)\,p(x)}{p(A)}$$

($p(x|A)$: Likelihood; $p(A|x)$: Posterior (Hard to get); $p(x)$: marginal; $p(A)$: prior/marginal of A)

$$p(A|x) = \frac{p(x|A)\,p(A)}{p(x)}$$

Random Vectors

$\vec{x} = [x_1, \ldots, x_n]^T$

PDF: (Joint density function) of random variables in vector:

$$Pr(x_1 < \tau_1, \ldots, x_2 < \tau_n) = \int_{-\infty}^{\tau_1} \cdots \int_{-\infty}^{\tau_2} p(\vec{x})\, d\vec{x}$$

Marginal Distribution of a subset:

$$P(x_1, \ldots x_{\ell-1}, x_{\ell+1}, x_n) = \int_{-\infty}^{\infty} p(\vec{x})\, dx_\ell$$

Expectation:

$$E[f(\vec{x})] = \int_{-\infty}^{\infty} \cdots \int_{-\infty}^{\infty} p(\vec{x}) f(\vec{x})\, dx$$

($p(x_1 \cdots x_n)$ (Joint prob.))

Mean : $(\text{vector})^n$

$$\vec{\mu}_x = E[\vec{x}]$$

Covariance : $(\text{matrix})^{n \times n}$ → correlation between x_i & x_j.

$$\Sigma_x = E[(\vec{x} - \mu_x)(\vec{x} - \mu_x)^T]$$

- Diagonal(Σ_x) = variance of random variables (+ve)
- $\Sigma_x = \Sigma_x^T$ (Diag Symmetry)

Independent ($\vec{x}$ & $\vec{y}$)

if $p(\vec{x}, \vec{y}) = p(\vec{x})\,p(\vec{y})$

Bayes Rule:

$$p(\vec{x}|A) = \frac{p(A|\vec{x})\,p(\vec{x})}{p(A)}$$

Uncorrelated

if $E[\vec{x}\vec{y}^T] = E[\vec{x}]\,E[\vec{y}^T]$

Sample Statistics

- In reality, we don't know prob-density or mean or covariance of a random vector.
  we infer actual observations: $\{\vec{x}_1, \vec{x}_2 \ldots \vec{x}_n\}$

  ↓

  Sample Statistics

More samples $n \uparrow \Rightarrow$ Better Distribution

Sample Mean:

$$\vec{m}_x = \frac{1}{N}\sum_{i=1}^{N}\vec{x}_i$$

Sample Covariance:

$$\Sigma = E[\vec{x}\vec{x}^T] - E[\vec{x}]E[\vec{x}]^T$$

$$S_x = \frac{1}{N}\sum_{i=1}^{N}(\vec{x}_i - \vec{m}_x)(\vec{x}_i - \vec{m}_x)^T = \frac{1}{N}\sum_{i=1}^{N}(x_i\vec{x}_i^T) - \vec{m}_x\vec{m}_x^T$$

Multivariable Gaussian Distribution

$$p(\vec{x}) = \frac{1}{(2\pi)^{n/2}|\Sigma|^{\frac{1}{2}}} e^{\left\{-\frac{1}{2}(\vec{x}-\vec{\mu})^T\Sigma^{-1}(\vec{x}-\vec{\mu})\right\}}$$

- $p(\vec{x})$: scalar
- $|\Sigma|$: determinants of covariance (= magnitude of matrix)

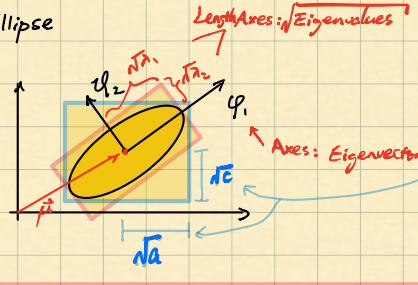
$\mathbb{R}^1$	$\Sigma = \sigma^2$, $\lvert\Sigma\rvert = \sigma^2$	$p(x) = \frac{1}{(2\pi)^{\frac{1}{2}}\sigma} e^{\left\{-\frac{1}{2}[(x-\mu)^2/\sigma^2]\right\}}$
$\mathbb{R}^n$, Diagonal Only (Uncorrelated)	$\Sigma = \mathrm{diag}(\frac{1}{\sigma^2})$, $\lvert\Sigma\rvert = \prod_{i=1}^{n}\sigma_{ii}^2$	$p(\vec{x}) = \prod_{i=1}^{n}\frac{1}{(2\pi)^{\frac{1}{2}}\sigma_{ii}} e^{\left\{-\frac{1}{2}[(x_i-\mu_{ii})^2/\sigma_{ii}^2]\right\}}$
$\mathbb{R}^2$ ★	$\Sigma = \begin{bmatrix} a & b \\ b & c \end{bmatrix}$	$p(\vec{x}) = \frac{1}{(2\pi)^{n/2}\lvert\Sigma\rvert^{\frac{1}{2}}} e^{\left\{-\frac{1}{2}[(\vec{x}-\vec{\mu})^T\Sigma^{-1}(\vec{x}-\vec{\mu})]\right\}}$

- a, c: Bounding box
- b: orientation
  - $b<0$: cw (correlated)
  - $b>0$: ccw (correlated)
  - $b=0$: uncorrelated

Equiprobability Contour:

$$p(\vec{x}) = (\vec{x}-\vec{\mu})^T\Sigma^{-1}(\vec{x}-\vec{\mu}) = C$$

$\mathbb{R}^2$: Ellipse



"$(A-\lambda I)\vec{v} = 0$"

Eigenvalue & Eigenvectors

④ $\det(S-\lambda I) = 0$

$|\Sigma - \lambda I| = 0$

⇓

λ_1, λ_2

⇓

② $S\vec{v} = \lambda_i\vec{v}$, $\forall i = \{1,2\}$

⇓

φ_1, φ_2

§ 4/5 - Classification

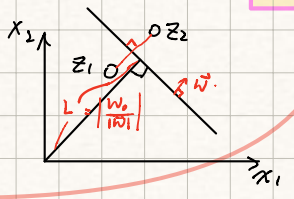
- Patterns → $\vec{x} = \begin{bmatrix} x_1 \\ \vdots \\ x_n \end{bmatrix}$ vectors in feature space $\mathbb{R}^n$
- Classification: • Compare { two patterns ⇒ how similar their attributes are
  pattern $\vec{x}$ & prototype $\vec{z}$ ⇒ how similar it is with a class

① MED (Minimum Euclidean Distance)

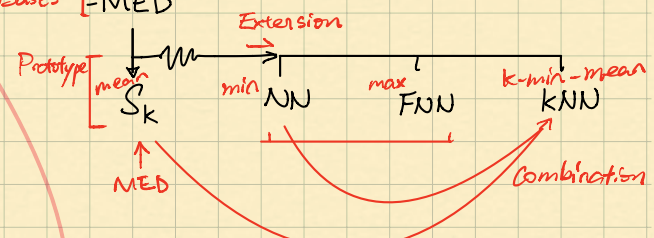
$\vec{x} \in C_k$, iff $d_E(\vec{x}, \vec{z}_k) < d_E(\vec{x}, \vec{z}_l)$, $\forall l \neq k$ ($\vec{z}$: prototype)

where $d_E(\vec{x}, \vec{z}_k) = \|\vec{x} - \vec{z}_k\|_2 = [(\vec{x}-\vec{z}_k)^T(\vec{x}-\vec{z}_k)]^{\frac{1}{2}}$

- Criteria: $-\vec{z}_1^T\vec{x} + \frac{1}{2}\vec{z}_1^T\vec{z}_1 < -\vec{z}_2^T\vec{x} + \frac{1}{2}\vec{z}_2^T\vec{z}_2$
- Discriminant func: $g_k(\vec{x}) = -\vec{z}_k^T\vec{x} + \frac{1}{2}\vec{z}_k^T\vec{z}_k$
- MED: $\vec{x} \in C_k$, iff $g_k(\vec{x}) < g_l(\vec{x})$, $\forall l \neq k$
- Decision Boundary: $g(\vec{x}) = \langle \vec{w}, \vec{x} \rangle + \vec{w}_0 = 0$.



Bases [-MED



Prototype Selection:

- Sample Mean: $z_k(x) = \frac{1}{N_k}\sum_{i=1}^{N_k} \vec{x}_i$
  - Less sensitive to Noise & Outliers ☺
  - Poor @ handling long, thin, dendrill-like clusters ☹ (non-linearly separable clusters)
- Nearest Neighbour: $z_k(x) = \vec{x}_k$ (NN)
  s.t. $d_E(\vec{x}, \vec{x}_k) = \min_i d_E(\vec{x}, \vec{x}_i)$ $\forall \vec{x}_i \in C_k$.
  - Better @ (non-linearly separable) ☺
  - More sensitive to noise & outliers ☹
  - Computationally complex
  - Hard to store, Privacy/Proprietary
  - { ① Compute dist between $\vec{x}$ with all labelled samples $\vec{x}_i$ for class 'k'
    ② choose min of ① as $\vec{x}_k$ for that class 'k'
- FNN (Furthest Neighbor): $z_k(x) = \vec{x}_k$ Choose max dist in the class 'k' as prototype '$\vec{x}_k$' for class 'k'.
  s.t. $d_E(\vec{x}, \vec{x}_k) = \max_i d_E(\vec{x}, \vec{x}_i)$ $\forall x_i \in C_k$ ⇒ More Compact Clusters ☺ ☹

② kNN (K-Nearest Neighbor)

- Less sensitive to noise & outliers. ☺
- Better at long, thin, dendrill, non-linearly separable clusters.
- ☹ — Computationally Complex
- For given $\vec{x}$ to classify,
  compute $d_E(\vec{x}, \vec{x}_k)$, where $\vec{x}_k$ is the mean of 'k' nearest points in that class.

(§ Part 2)

Weighted Euclidean Distance Metric (WED)

$$d_{W_0}(\vec{x}, \vec{z}) = \left[(\vec{x}-\vec{z})^T W^T W (\vec{x}-\vec{z})\right]^{\frac{1}{2}}$$

where: $W = \begin{bmatrix} w_{11} & \cdots & w_{1n} \\ \vdots & \ddots & \vdots \\ w_{n1} & \cdots & w_{nn} \end{bmatrix}$ ⇓ Scaling & rotation

Assume W_0 = diag. ⇒ (W_0 no rotation)

$d_{W_0}(\vec{x}, \vec{z}) = d_E(\vec{x}', \vec{z}')$ Transformed Space!!

, where $\vec{x}' = W_0\vec{x}$, $\vec{z}' = W_0\vec{z}$

$W_0 = \text{diag}([w_0, \ldots w_n]) = \begin{bmatrix} w_0 & \cdots & 0 \\ \vdots & \ddots & \vdots \\ 0 & \cdots & w_n \end{bmatrix}$

Linear space ⇒ Quadratic Space

Dataset — 'W' Transform. (Linear-space) → ED (in 'W' space.) (best for.)

① Correlated → WED → Uncorrelated
② diff-variance → (normalized in 1 std. deviation 5). → unit variance (Hypersphere)

GED when MED: $W = \Lambda^{-\frac{1}{2}} \Phi$

Orthonormal Covariance Transforms. ⇒ ① Uncorrelate Features. (Rotate Coordinate Axes using 'Eigen.')

find transform A,

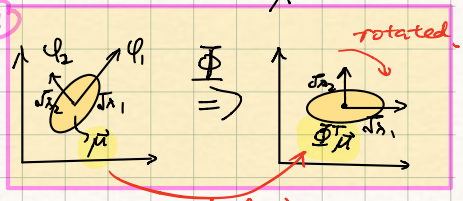
s.t. $A \Sigma A^T = \Lambda$ (that diagonalize 'Σ')

Λ: diag-covariance matrix.

Whose:

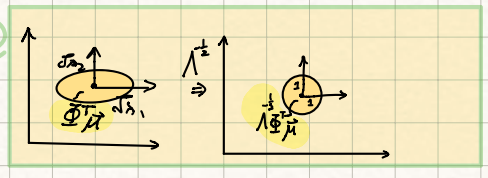
$$A = \begin{bmatrix} \vec{\Phi}_1^T \\ \vdots \\ \vec{\Phi}_n^T \end{bmatrix} = \Phi^T \qquad \Lambda = \begin{bmatrix} \lambda_1 & \cdots & 0 \\ 0 & \ddots & \vdots \\ 0 & \cdots & \lambda_n \end{bmatrix} = \text{diag}[\lambda_1, \ldots, \lambda_n]$$

⇓ to get diagonal covariance matrix 'Λ'

① 

Whitening Transformation ② ⇒ Unify Variance (Normalize)

$$\Lambda^{-\frac{1}{2}} = \begin{bmatrix} \lambda_1^{-\frac{1}{2}} & \cdots & 0 \\ 0 & \ddots & \vdots \\ 0 & \cdots & \lambda_n^{-\frac{1}{2}} \end{bmatrix}$$

② 

Generalized Euclidean Distance Metrics (GED)

$W = \Lambda^{-\frac{1}{2}} \Phi^T$, $W \in \mathbb{R}^{m \times n}$ in MED

GED

$$d_G(\vec{x}, \vec{z}) = \left[(\vec{x} - \vec{z})^T (S^{-1}) (\vec{x} - \vec{z}) \right]^{\frac{1}{2}}$$

where $S = \Phi \Lambda^{-1} \Phi^T$

Minimum Intra-Class Distance Classifier (MICD)

$$\vec{x} \in A \text{ iff } (\vec{x} - \vec{m}_A)^T S_A^{-1} (\vec{x} - \vec{m}_A) < (\vec{x} - \vec{m}_B)^T S_B^{-1} (\vec{x} - \vec{m}_B)$$

MICD Metric : GED metric — intra distance measurement ⇒ compare. (normalize space into std. deviation)

Decision Boundaries: Non-linear (Quadratic Surface)

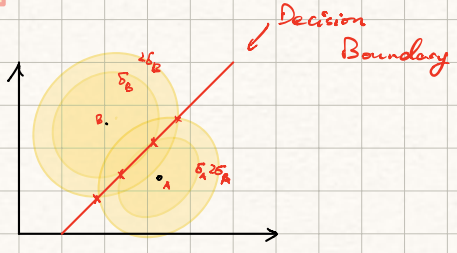
- Lies on intersection of equidistance contours around classes.

$$(\vec{x} - \vec{m}_A)^T S_A^{-1} (\vec{x} - \vec{m}_A) = C$$

- Quadratic Surface in Hyperspace:

$$\vec{x}^T Q_0 \vec{x} + Q_1 \vec{x} + Q_2 = 0$$

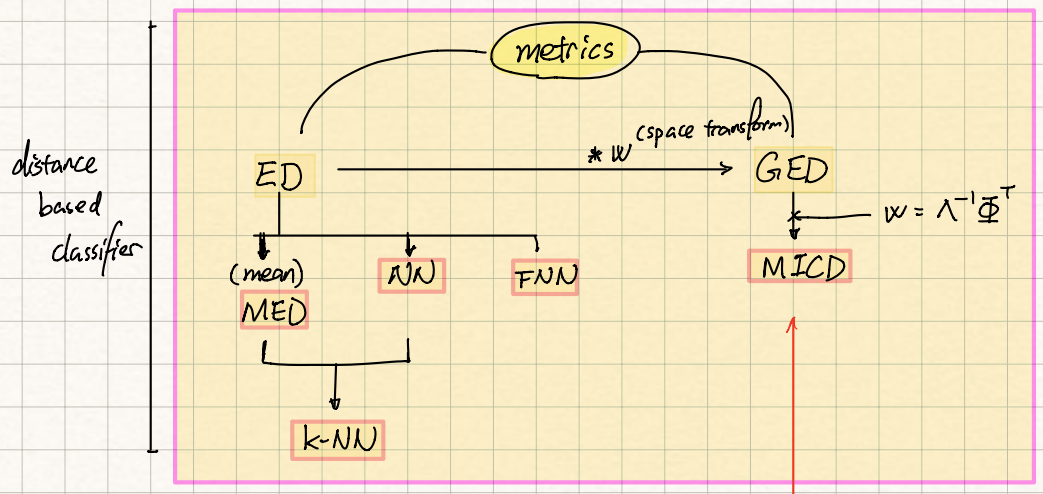
where
$$Q_0 = S_A^{-1} - S_B^{-1}$$
$$Q_1 = 2 [\vec{m}_B^T S_B^{-1} - \vec{m}_A S_A^{-1}]$$
$$Q_2 = \vec{m}_A^T S_A^{-1} \vec{m}_A - \vec{m}_B^T S_B^{-1} \vec{m}_B.$$



- Issue: in favor of the class with largest variance, regardless of x !

- Pro: Lower sensitivity to noise & outliers
  Great handling class distribution & Gaussian models. (Gaussian-like)
- Con: Poor @ handling more complex class distribution.

↳ (Does not take into account of probabilistic info of classes)



account for variability
by incorporating statistical params of the class (mean & variance)

↓

well for Gaussian-like statistics.
but poor for non-Gaussian.

↓ Soln

probabilistic classifiers.

§ 6.7 - Probabilistic Classification

Idea: Given known "class conditional probability density distribution", we can create powerful similarity measures that tell us the **likelihood / probability** of each class given an observed pattern.

Basis: Optimal in the minimum "**probability of error**" sense.

× Bayesian Classifier

- Let L_{ij} : cost of deciding on class 'c_j' when the true class is 'c_i'
- Total Risk with deciding $\vec{x}$ belongs to C_j:

The expected cost: $r_j(\vec{x}) = \sum_{i=1}^{K} L_{ij} P(c_i|\vec{x})$ (# of classes)

$r_j(\vec{x})$: Total Risk; $P(c_i|\vec{x})$: Probability I'm wrong (Posterior distribution of class 'c_j' given pattern '$\vec{x}$')

- $\mathbb{R}^2$ case:

Total Cost (Bayes' rule):

$$r_1(\vec{x}) = L_{11} P(c_1|\vec{x}) + L_{21} P(c_2|\vec{x}) = \frac{L_{11} P(\vec{x}|c_1) P(c_1) + L_{21} P(\vec{x}|c_2) P(c_2)}{P(\vec{x})}$$

$$r_2(\vec{x}) = L_{12} P(c_1|\vec{x}) + L_{22} P(c_2|\vec{x}) = \frac{L_{12} P(\vec{x}|c_1) P(c_1) + L_{22} P(\vec{x}|c_2) P(c_2)}{P(\vec{x})}$$

× K-class Bayesian Classifier:

$$\vec{x} \in C_i \quad \text{if} \quad r_i(\vec{x}) < r_j(\vec{x}) \quad \forall j \neq i$$

- $\mathbb{R}^2$ case: $(L_{12} - L_{11}) P(\vec{x}|c_1) P(c_1) \underset{2}{\overset{1}{\gtrless}} (L_{21} - L_{22}) P(\vec{x}|c_2) P(c_2)$.

× Cost Function

- "zero-one" loss (most common):

$$L_{ij} = \begin{cases} 0 & i = j \\ 1 & i \neq j \end{cases}$$

⇒ Total Risk Func:

$$r_j(\vec{x}) = \sum_{\substack{i=1 \\ i \neq j}}^{K} P(c_i|\vec{x}) = P(\varepsilon|\vec{x})$$ (cond. prob.)

⇒ Meaning: All errors have equal costs

⇒ $\min\{r_j(\vec{x})\}$ (total risk) $\equiv \min\{\text{prob. of error}\}$

× Types: { MAP (Maximum A Posteriori), ML (Maximum Likelihood) } ⇒ probability classifiers.

MAP ⇒ Choose most probable class

ML ⇒ choose the class that makes observed pattern $\vec{x}$ most probable.

× MAP Classifier

$$P(A|\vec{x}) \underset{B}{\overset{A}{\gtrless}} P(B|\vec{x})$$ ⇒ All patterns with higher probability for 'A' than for 'B' will be classified as 'A'

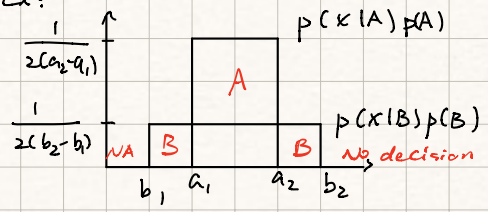
$\therefore \quad \frac{P(\vec{x}|A)}{P(\vec{x}|B)} \underset{B}{\overset{A}{\gtrless}} \frac{P(B)}{P(A)}$ ✓ Bayes.

when dealing with PDF with exponential dependence (Gama / Gauss)

$\therefore \quad l(x) \underset{B}{\overset{A}{\gtrless}} \theta \quad \Rightarrow \quad \log(l(\vec{x})) \underset{B}{\overset{A}{\gtrless}} \log\theta$ (log-likelihood.)

likelihood Ratio ↑ threshold ↑ — reduce computation 'e'

× Ex:



o Often, P(A) & P(B) priors are unknown.
hence impossible to use the postesior $p(A|\vec{x})$ & $p(B|\vec{x})$.

× ML Classifies ⇒ [Special case of MAP when P(A) = P(B)]

$p(\vec{x}|A) \underset{B}{\overset{A}{\gtrless}} p(\vec{x}|B) \quad \Rightarrow \quad \frac{p(\vec{x}|A)}{p(\vec{x}|B)} \underset{B}{\overset{A}{\gtrless}} 1$

(§ 7 Part II)

× MAP for Normal Distribution

× Gaussian: $\begin{cases} p(x|A) = \mathcal{N}(\mu_A, \sigma_A^2) = \frac{1}{\sqrt{2\pi}\sigma_A} e^{\left\{-\frac{1}{2}\left(\frac{x-\mu_A}{\sigma_A}\right)^2\right\}} \\ p(x|B) = \mathcal{N}(\mu_B, \sigma_B^2) \end{cases}$

MAP: $\frac{\mathcal{N}(\mu_A, \sigma_A^2)}{\mathcal{N}(\mu_B, \sigma_B^2)} \underset{B}{\overset{A}{\gtrless}} \frac{P(B)}{P(A)}$

$\Rightarrow \quad \frac{\exp\left\{-\frac{1}{2}\left(\frac{x-\mu_A}{\sigma_A}\right)^2\right\}}{\exp\left\{-\frac{1}{2}\left(\frac{x-\mu_B}{\sigma_B}\right)^2\right\}} \underset{B}{\overset{A}{\gtrless}} \frac{P(B)\sigma_A}{P(A)\sigma_B}$

log-likelihood ⇒ $\left\{-\frac{1}{2}\left(\frac{x-\mu_A}{\sigma_A}\right)^2\right\} - \left\{-\frac{1}{2}\left(\frac{x-\mu_B}{\sigma_B}\right)^2\right\} \underset{B}{\overset{A}{\gtrless}} \ln[\sigma_A P(B)] - \ln[\sigma_B P(A)]$

⇓

MAP (Norm) $\left[\left(\frac{x-\mu_B}{\sigma_B}\right)^2\right] - \left[\left(\frac{x-\mu_A}{\sigma_A}\right)^2\right] \underset{B}{\overset{A}{\gtrless}} 2\left\{\ln[\sigma_A P(B)] - \ln[\sigma_B P(A)]\right\}$.

'=' ↑ solve for decision boundary

× Case: $\sigma_A = \sigma_B$, $P(A) = P(B) = \frac{1}{2}$.

$x = \frac{(\mu_B^2\sigma_A^2 - \mu_A^2\sigma_A^2)}{2(\mu_B\sigma_A^2 - \mu_A\sigma_A^2)} = \frac{(\mu_B^2 - \mu_A^2)}{2(\mu_B - \mu_A)} = \frac{(\mu_B + \mu_A)}{2}$

→ For equal likely, equal variance classes, MAP reduce → (threshold midway btwn the means)

→ $\mathbb{R}^n$ case: $p(\vec{x}|A) = \mathcal{N}(\vec{\mu}_A, \Sigma_A^2)$

$p(\vec{x}|B) = \mathcal{N}(\vec{\mu}_B, \Sigma_B^2)$

$$\Rightarrow \frac{\exp\left[-\frac{1}{2}(\vec{x}-\vec{\mu}_A)^T \Sigma_A^{-1}(\vec{x}-\vec{\mu}_A)\right]}{\exp\left[-\frac{1}{2}(\vec{x}-\vec{\mu}_B)^T \Sigma_B^{-1}(\vec{x}-\vec{\mu}_B)\right]} \underset{B}{\overset{A}{\gtrless}} \frac{|\Sigma_A|^{\frac{1}{2}} P(B)}{|\Sigma_B|^{\frac{1}{2}} P(A)}$$

↓ Log

MAP

$$(\vec{x}-\vec{\mu}_B)^T \Sigma_B^{-1}(\vec{x}-\vec{\mu}_B) - (\vec{x}-\vec{\mu}_A)^T \Sigma_A^{-1}(\vec{x}-\vec{\mu}_A) \underset{B}{\overset{A}{\lessgtr}} 2\ln\left[\frac{P(B)}{P(A)}\right] + \ln\left[\frac{|\Sigma_A|}{|\Sigma_B|}\right]$$

Bias decision in favor based on prior probabilities

Bias on smaller volume $|\Sigma|$

same as MICD RH.

$$d^2_{MICD}(\vec{x}, \mu_B, \Sigma_B) - d^2_{MICD}(\vec{x}, \mu_A, \Sigma_A) \underset{B}{\overset{A}{\lessgtr}} 2\ln\left[\frac{P(B)}{P(A)}\right] + \ln\left[\frac{|\Sigma_A|}{|\Sigma_B|}\right]$$

Decision Bndry: $\vec{x}^T Q_0 \vec{x} + Q_1 \vec{x} + Q_2 + 2Q_3 + Q_4 = 0$

$$Q_0 = S_A^{-1} - S_B^{-1}$$
$$Q_1 = 2\left[\vec{m}_B^T S_B^{-1} - \vec{m}_A^T S_A^{-1}\right]$$
$$Q_2 = \vec{m}_A^T S_A^{-1} \vec{m}_A - \vec{m}_B^T S_B^{-1} \vec{m}_B$$
$$Q_3 = \ln\left[\frac{P(B)}{P(A)}\right]$$
$$Q_4 = \ln\left[\frac{|S_A|}{|S_B|}\right]$$

→ **Relationships**

when normal distribution

MAP: $(\vec{x}-\vec{\mu}_B)^T \Sigma_B^{-1}(\vec{x}-\vec{\mu}_B) - (\vec{x}-\vec{\mu}_A)^T \Sigma_A^{-1}(\vec{x}-\vec{\mu}_A) \underset{B}{\overset{A}{\lessgtr}} 2\ln\left[\frac{P(B)}{P(A)}\right] + \ln\left[\frac{|\Sigma_A|}{|\Sigma_B|}\right]$

ML: $(\vec{x}-\vec{\mu}_B)^T \Sigma_B^{-1}(\vec{x}-\vec{\mu}_B) - (\vec{x}-\vec{\mu}_A)^T \Sigma_A^{-1}(\vec{x}-\vec{\mu}_A) \underset{B}{\overset{A}{\lessgtr}} \ln\left[\frac{|\Sigma_A|}{|\Sigma_B|}\right]$

MICD: $(\vec{x}-\vec{\mu}_B)^T \Sigma_B^{-1}(\vec{x}-\vec{\mu}_B) - (\vec{x}-\vec{\mu}_A)^T \Sigma_A^{-1}(\vec{x}-\vec{\mu}_A) \underset{B}{\overset{A}{\lessgtr}} 0$

MED: $(\vec{x}-\vec{\mu}_B)^T (\vec{x}-\vec{\mu}_B) - (\vec{x}-\vec{\mu}_A)^T (\vec{x}-\vec{\mu}_A) \underset{B}{\overset{A}{\lessgtr}} 0$

MAP = MICD, iff
- normal distribution $\mathcal{N}(\vec{\mu}, \Sigma)$
- equal prior prob. $(P(A) = P(B))$
- equal volume $(|\Sigma_A| = |\Sigma_B|)$

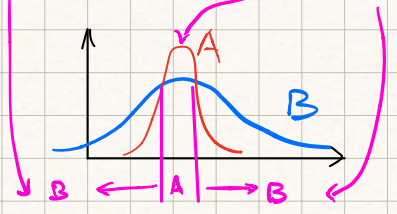
$\ln(1) = 0$

$$2\ln\left[\frac{P(B)}{P(A)}\right] + \ln\left[\frac{|\Sigma_A|}{|\Sigma_B|}\right]$$

MICD in favour of class with largest variance regardless of x.

MED in favor of class with lowest variance close to mean
highest variance beyond a certain point in both dim.

Fix this issue by taking account of the probability.



Performance of Bayes Classifier

→ Can compute Theoretical Limit without data !!

Probability of Error: $P(\varepsilon|\vec{x}) = \min [P(A|\vec{x}), P(B|\vec{x})]$

We choose maximum posterior as our class.
∴ error would be the minimum posterior.

Expected probability of error:

$$P(\varepsilon) = \int P(\varepsilon|\vec{x})\, p(\vec{x})\, d\vec{x}$$

$$= \int \min [P(\vec{x}|A)P(A),\ P(\vec{x}|B)P(B)]\, dx.$$

Hard

define decision Regions R_A & R_B:

$R_A = \vec{x}$ s.t. $P(A|\vec{x}) > P(B|\vec{x})$

$R_B = \vec{x}$ s.t. $P(B|\vec{x}) > P(A|\vec{x})$

$$\Downarrow$$

$$P(\varepsilon) = \int_{R_A} P(\vec{x}|B)\, P(B)\, dx + \int_{R_B} P(\vec{x}|A)\, P(A)\, dx$$

Ex: univariate Normal, equal variance, equally likely, two class

→ (1D)

- $n=1$, $P(A) = P(B) = 0.5$, $\sigma_A = \sigma_B = \sigma$, $\mu_A < \mu_B$
- Likelihood:

$$\begin{cases} p(x|A) = N(\mu_A, \sigma_A^2) \\ p(x|B) = N(\mu_B, \sigma_B^2) \end{cases}$$

- find $p(\varepsilon)$

- Recall this case: MAP $\xrightarrow[\text{Boundary}]{\text{reduce}}$ $x = \frac{(\mu_B+\mu_A)}{2}$
- $\mu_A < \mu_B$, $\Rightarrow$ $R_A = x$ s.t. $x < \frac{(\mu_B+\mu_A)}{2}$
  $R_B = x$ s.t. $x > \frac{(\mu_B+\mu_A)}{2}$

$$\therefore P(\varepsilon) = \int_{R_A} \cancel{P(B)}^{\frac{1}{2}} P(\vec{x}|B)\, d\vec{x} + \int_{R_B} \cancel{P(A)}^{\frac{1}{2}} P(\vec{x}|A)\, d\vec{x}$$

$$= \frac{1}{2}\int_{-\infty}^{\frac{(\mu_B+\mu_A)}{2}} N(\mu_B, \sigma^2)\, dx + \frac{1}{2}\int_{\frac{(\mu_B+\mu_A)}{2}}^{\infty} N(\mu_A, \sigma^2)\, dx$$

∵ symmetric class. $\Rightarrow P(\varepsilon|A) = P(\varepsilon|B)$

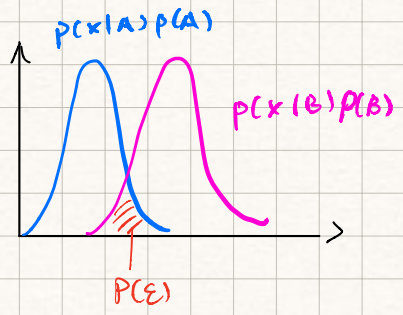
$$\therefore P(\varepsilon) = \int_{\frac{(\mu_B+\mu_A)}{2}}^{\infty} \frac{1}{\sqrt{2\pi}\sigma_A} \exp\left\{-\frac{1}{2}\left(\frac{x-\mu_A}{\sigma_A}\right)^2\right\} dx.$$

$\Rightarrow$ Let $y = \frac{x-\mu_A}{\sigma}$, $dx = \sigma\, dy$ ↓ Normalize to $N(0,1)$ $\Rightarrow$ for lookup table. (Q func)

$$P(\varepsilon) = \int_{\frac{(\mu_B-\mu_A)}{2\sigma}}^{\infty} \frac{1}{\sqrt{2\pi}} \exp\left[-\frac{1}{2}y^2\right] dx$$

↳ $Q(\alpha) = \int_{\alpha}^{\infty} \frac{1}{\sqrt{2\pi}} \exp\left[-\frac{1}{2}y^2\right] dy$

$$\therefore P(\varepsilon) = Q\left(\frac{\mu_B-\mu_A}{2\sigma}\right) \text{ (lookup table)}$$

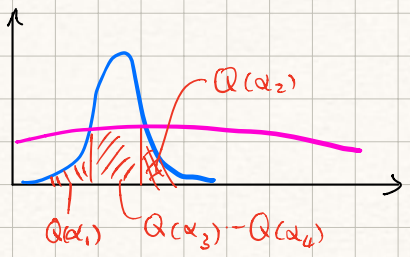


Observation:

- as dist. btwn means ↑ shaded area ($P(\varepsilon)$) monotonically ↓
- At $\alpha = 0$, $\mu_A = \mu_B = 0$ & $P(\varepsilon) = \frac{1}{2}$
- $\lim_{\alpha \to \infty} P(\varepsilon) = 0$.

- For case $P(A) \neq P(B)$ / $\sigma_A \neq \sigma_B$,

(1D) Decision boundary change AND additional boundry is introduced.



$$P(\varepsilon) = P(A)Q(\alpha_1) + P(B)[Q(\alpha_3) - Q(\alpha_4)] + P(A)Q(\alpha_2)$$

- n-D case (Multivariate, $n > 1$)

$$\text{For } \begin{cases} p(\vec{x}|A) = \mathcal{N}(\vec{\mu}_A, \Sigma) \\ p(\vec{x}|B) = \mathcal{N}(\vec{\mu}_B, \Sigma) \\ p(A) = p(B) \end{cases}$$

$$\hookrightarrow \boxed{P(\varepsilon) = Q\left(d_M(\vec{\mu}_A, \vec{\mu}_B)/2\right)}$$

where $d_M(\vec{\mu}_A, \vec{\mu}_B)$ is Mahalanobis Dist. btwn classes.

$$d_M(\vec{\mu}_A, \vec{\mu}_B) = \left[(\vec{\mu}_A - \vec{\mu}_B)^T \Sigma^{-1} (\vec{\mu}_A - \vec{\mu}_B)\right]^{\frac{1}{2}}$$

↳ GED

Explanation:

- $\forall$ cases where $\Sigma_A = \Sigma_B$, $p(A) = p(B)$,

the decision boundary btwn classes is ALWAYS straight line in hyperspace.

- slope based on Σ (since orthonormal whitening ($\Lambda^{-1}\Phi^T$) is identical)
- intersects with the midpoint of the line segment btwn ($\vec{\mu}_A$ & $\vec{\mu}_B$)

- The probability of err. ≡ the area under $p(\vec{x}|A)p(A)$ on the class B side of the DBND.
  + the area under $p(\vec{x}|B)p(A)$ on the class A side of the DBND

Decision Boundary

- non-Gaussian PDF:

Ex1 Suppose two classes have density functions & a priori probabilities:

$$p(x|C_1) = \begin{cases} ce^{-2x}, & 0 \leq x \leq 1 \\ 0, & \text{else} \end{cases}$$

$$p(x|C_2) = \begin{cases} ce^{-2(1-x)}, & 0 \leq x \leq 1 \\ 0, & \text{else} \end{cases}$$

where $c = \frac{2}{1-e^{-2}}$ ≡ normalizing the PDF.

$$P(C_1) = P(C_2) = \frac{1}{2}$$

∴ Expected Prob of error:

$$P(\varepsilon) = \int \min\left[P(\vec{x}|C_1), P(\vec{x}|C_2)P(C_2)\right] d\vec{x}$$

$$= \int_{R_{C_1}} P(\vec{x}|C_2)P(C_2)\,d\vec{x} + \int_{R_{C_2}} P(\vec{x}|C_1)P(C_1)\,d\vec{x}$$

$$= \int_0^{0.5} 0.5\,P(\vec{x}|C_2)\,dx + \int_{0.5}^{1.0} 0.5\,P(\vec{x}|C_1)\,d\vec{x}$$

∵ Symmetry

$$= \int_{0.5}^{1.0} ce^{-\lambda x}\,d\vec{x}$$

$P(\varepsilon|C_1) = P(\varepsilon|C_2)$

$$= \frac{c}{\lambda}\left[e^{-\lambda/2} - e^{-\lambda}\right]$$

$$\therefore P(\varepsilon|x) = \begin{cases} P(C_2|\vec{x}) & 0 \le x \le \frac{1}{2} \\ P(C_1|x) & \frac{1}{2} \le x \le 1 \end{cases} \Rightarrow \begin{cases} \frac{P(x|C_2)P(C_2)}{P(x)} \\ \frac{P(x|C_1)P(C_1)}{P(x)} \end{cases} \Rightarrow \begin{cases} \frac{e^{-\lambda x}\,0.5}{e^{-\lambda x} + e^{-\lambda(1-x)}} \\ \frac{e^{-\lambda(1-x)}\,0.5}{e^{-\lambda x} + e^{-\lambda(1-x)}} \end{cases}$$

× Error Bounds

- × In practice, the exact $P(\varepsilon)$ is only easy to compute for simple cases as shown before
- × We may determine Bounds on $P(\varepsilon)$ instead of exact $P(\varepsilon)$
  - Easier to compute
  - Leads to estimates of classifier performance.

× Bhattacharyya Bound

× Derivation:

Approximate $\min[a,b] \le \sqrt{(a,b)}$

$$\Downarrow$$

$$P(\varepsilon) = \int \min\left[P(\vec{x}|A)P(A), P(\vec{x}|B)P(B)\right] d\vec{x}$$

$$\Downarrow$$

$$P(\varepsilon) \le \sqrt{P(A)P(B)} \int \sqrt{P(\vec{x}|A)P(\vec{x}|B)}\,d\vec{x}$$

∵ $P(A) + P(B) = 1$

Let define Bhattacharyya Coef as

$$\rho = \int \sqrt{P(\vec{x}|A)P(\vec{x}|B)}\,d\vec{x}$$

∴ Upper Bound (Bhattacharyya Bound) of $P(\varepsilon)$: $P(\varepsilon) \le \frac{1}{2}\rho$ ☆

max possible error:

when $P(A) = P(B) = \frac{1}{2}$

$$\Rightarrow \sqrt{P(A)P(B)} = \frac{1}{2}$$

§ 8/9 - Estimation & Learning

Motivation

Recall: ① Bayesian Classifier achieves Minimum Probability of Error
↳ Better than MICD classifier for situations,
where the class conditional PDFs '$p(\vec{x}|C_i)$' are known !
↑ But normally unknown

Idea: Learn Empirical PDFs ⇒ to apply Bayesian Classifier.

Remark: ① Bayesian Classification is optimal
② Empirical PDFs result in sub-optimal classifier.
③ Performance to the theoretical limit (minimum) P(E) depends on accuracy of estimated PDF.

Categories: ① Parametric Estimation: { function form of PDF (v), parameters of PDF (x) } ⇒ { ① Maximum Likelihood Estimation, ② Bayesian Estimation }
② Non-parametric Estimation. (Direct Estimation) { function form of PDF (x), parameters of PDF (x) } ⇒ { ① Histogram Estimation, ② Parzen Window Estimation, ③ k-NN Estimation }

① Parametric Learning

{ function form of PDF (v), parameters of PDF (x) }

Ex Gaussian: $N(\vec{\mu}_A, \Sigma_A)$ (v)
$\vec{\mu}_A = ?$ $\Sigma_A = ?$ (x)

Two Approaches { ① Maximum Likelihood Estimation — { Params fixed (v), Params unknown quantity (x) } ⇒ Goal: find estimate values s.t. max probability that given samples ∈ the resulting PDF: $p(\vec{x}|\theta)$
② Bayesian (Maximum Posterior) Estimation
↳ { Params as random variables
↳ known prior distribution (v) }

⇓
Goal: obtain a posterior distribution $p(\theta|\vec{x})$ which indicates the estimate value based on the given samples.

Maximum Likelihood Estimation

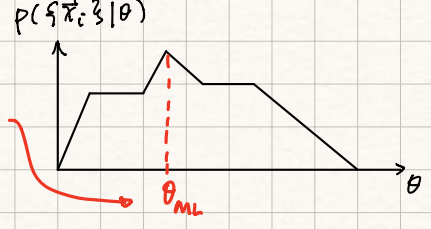
Given: ① set of samples: $\{\vec{x}_i\}$
② PDF form: $p(\vec{x})$ Ex. Gaussian.

Goal: obtain parameters θ ⇒ PDF

Ex: Given ② PDF: $p(\vec{x}|A) = N(\vec{\mu}_A, \Sigma_A)$
⇒ Goal: $\theta = (\vec{\mu}_A, \Sigma_A)$

Formulation:

$$\hat{\theta}_{ML} = \arg\max_{\theta} [p(\{\vec{x}_i\}|\theta)]$$



Verbal: Given $\vec{x}_i$ (observation),
the maximum estimate of parameter θ
is chosen to be that value which maximizes the PDF: $p(\vec{x}_i|\theta)$

Derivation: Assume I.I.D. condition. (sample are independent of each other)

$$p(\{\vec{x}_i\}|\theta) = \underbrace{p(\vec{x}_1, \vec{x}_2, \cdots, \vec{x}_N|\theta)}_{\text{Joint Distribution}} = \underbrace{\prod_{i=1}^{N} p(\vec{x}_i|\theta)}_{\text{product of individual sample prob.}}$$

Likelihood prob.

- To max $\{p(\{\vec{x}_i\}|\theta)\}$, take derivative & set to 0

$$\left.\frac{\partial}{\partial\theta} p(\{\vec{x}_i\}|\theta)\right|_{\theta=\hat{\theta}_{ML}} = 0$$

- Log Likelihood:

$$l(\theta) = \log\left\{\prod_{i=1}^{N} p(\{\vec{x}_i\}|\theta)\right\} = \sum_{i=1}^{N} \log p(\vec{x}_i|\theta)$$

∴ Maximum Likelihood Condition:

$$\frac{\partial}{\partial\theta} l(\theta)\Big|_{\theta=\hat{\theta}_{ML}} = 0$$

Estimation Bias

Definition:

An estimate $\hat{\vec{\theta}}$ is unbiased if its expected value is equal to the true value: $E[\hat{\vec{\theta}}] = \vec{\theta}$

Ex ML estimate of the mean is Unbiased:

$$E[\hat{\vec{\mu}}_{ML}] = E\left[\frac{1}{N}\sum_{i=1}^{N}\vec{x}_i\right]$$

$$= \frac{1}{N}\sum_{i=1}^{N} E[\vec{x}_i]$$

$\because \vec{\mu} = E[\vec{x}]$

$$\therefore = \frac{1}{N}\sum_{i=1}^{N}\vec{\mu}$$

$$= \vec{\mu}$$

$$\therefore E[\hat{\vec{\mu}}_{ML}] = \vec{\mu} \Rightarrow \text{Unbiased}$$ ☺

Ex ML estimate of covariance Matrix is Biased:

$$E[\hat{\Sigma}_{ML}] = E\left[\frac{1}{N}\sum_{i=1}^{N}(\vec{x}_i - \hat{\vec{\mu}}_{ML})(\vec{x}_i - \hat{\vec{\mu}}_{ML})^T\right]$$

$$= \frac{1}{N}\sum_{i=1}^{N} E\left[(\vec{x}_i - \hat{\vec{\mu}}_{ML})(\vec{x}_i - \hat{\vec{\mu}}_{ML})^T\right]$$

$\hat{\vec{\mu}}_{ML} = \frac{1}{N}\sum_{i=1}^{N}\vec{x}_i$

$$= \frac{1}{N}\sum_{i=1}^{N} E\left[((\vec{x}_i - \vec{\mu}) - (\hat{\vec{\mu}}_{ML} - \vec{\mu}))((\vec{x}_i - \vec{\mu}) - (\hat{\vec{\mu}}_{ML} - \vec{\mu}))^T\right]$$

$$= E\left[\frac{1}{N}\sum_{i=1}^{N}(\vec{x}_i - \vec{\mu})(\vec{x}_i - \vec{\mu})^T\right] - E\left[(\hat{\vec{\mu}}_{ML} - \vec{\mu})(\hat{\vec{\mu}}_{ML} - \vec{\mu})^T\right] - E\left[(\hat{\vec{\mu}}_{ML} - \vec{\mu})(\hat{\vec{\mu}}_{ML} - \vec{\mu})^T\right]$$

$$= \Sigma - E\left[\left(\frac{1}{N}\sum_{i=1}^{N}\vec{x}_i - \vec{\mu}\right)\left(\frac{1}{N}\sum_{i=1}^{N}\vec{x}_i - \vec{\mu}\right)^T\right]$$

$$= \Sigma - E\left[\left(\frac{1}{N}\sum_{i=1}^{N}(\vec{x}_i - \vec{\mu})\right)\left(\frac{1}{N}\sum_{i=1}^{N}(\vec{x}_i - \vec{\mu})\right)^T\right]$$

$$= \Sigma - \frac{1}{N^2}E\left[\sum_{i=1}^{N}\sum_{j=1}^{N}(\vec{x}_i - \vec{\mu})(\vec{x}_j - \vec{\mu})^T\right]$$

$\because$ IID $\Rightarrow E[(\vec{x}_i - \vec{\mu})(\vec{x}_j - \vec{\mu})^T] = 0 \quad \forall i \neq j.$

$$\therefore = \Sigma - \frac{1}{N^2}\sum_{i=1}^{N} E[(x_i - \vec{\mu})(\vec{x}_i - \vec{\mu})^T]$$ ($E[\cdots] = \Sigma$)

$$\therefore E[\hat{\Sigma}_{ML}] = \Sigma - \frac{1}{N^2}N\Sigma = \frac{N-1}{N}\Sigma \Rightarrow \text{Biased}$$ ☹

↳ As $N \to \infty$, Bias negligible.

↳ or ($\times \frac{N}{N-1}$) for ML Estimate: $E\left[\frac{N}{N-1}\hat{\Sigma}_{ML}\right] = \Sigma$.

Bayesian Estimation

→ Idea: • Instead of MLE { treating params fixed; finding param ⇒ maximise samples come from estimated PDF

• We do BE { treating params random var. with an assumed a priori distribution; use the observed samples ⇒ obtain posterior-distribution which indicates params.

→ Procedure: • Let $p(\theta)$ be a priori distribution
$\{\vec{x}_i\}$ be the set of samples.

• Posteriori Distribution: $p(\vec{\theta}|\{\vec{x}_i\}) = \frac{p(\{\vec{x}_i\}|\vec{\theta})\,p(\vec{\theta})}{p(\{\vec{x}_i\})}$ ← scale factor ⇐ from the requirement of PDF: $\int p(\vec{\theta}|\{\vec{x}_i\})\,d\vec{\theta} = 1$.

→ EX/ Suppose: { PDF : $p(x) = \mathcal{N}(\mu, \sigma^2)$ (✓); μ : (✗); σ^2 : (✓) $= \sigma_0$

↳ Bayesian Estimate of μ?

① Assume a priori PDF for param $\theta = \mu$.

↳ $p(\mu) = \mathcal{N}(\mu|\mu_0, \sigma_0^2)$ (μ_0: initial guess; σ_0^2: uncertainty of the guess)

② Given Samples, compute $p(\mu|\{x_i\})$

$$p(\mu|\{x_i\}) = \alpha\, p(\{x_i\}|\mu)\, p(\mu)$$

• Assume samples IID.

$$= \alpha \prod_{i=1}^{N} p(x_i|\mu)\, p(\mu)$$

where $\alpha = \frac{1}{p(\{x_i\})}$ as scale factor independent of μ.

• Sub:

$$= \alpha \prod_{i=1}^{N} \frac{1}{\sqrt{2\pi}\sigma} \exp\left\{-\frac{1}{2}\left(\frac{x_i-\mu}{\sigma}\right)^2\right\} \cdot \frac{1}{\sqrt{2\pi}\sigma_0} \exp\left\{-\frac{1}{2}\left(\frac{\mu-\mu_0}{\sigma_0}\right)^2\right\}$$

$$= \alpha' \exp\left\{-\frac{1}{2}\left[\sum_{i=1}^{N} \frac{x_i^2 - 2x_i\mu + \mu^2}{\sigma^2} + \frac{\mu^2 - 2\mu\mu_0 + \mu_0^2}{\sigma_0^2}\right]\right\}$$

$$= \alpha'' \exp\left\{-\frac{1}{2}\left[\frac{N}{\sigma^2} + \frac{1}{\sigma_0^2}\right]\mu^2 - 2\left[\frac{N m_N}{\sigma^2} + \frac{\mu_0}{\sigma_0^2}\right]\mu\right\}$$

where m_N is sample mean !! ; quadratic in μ. ⇒ Gaussian Form !!

• In short:

∴ posteriori Density:

$$p(\mu|\{x_i\}) = \mathcal{N}(\mu|\mu_N, \sigma_N^2)$$

where $\mu_N = \frac{N\sigma_0^2}{N\sigma_0^2+\sigma^2} m_N + \frac{\sigma^2}{N\sigma_0^2+\sigma^2}\mu_0$

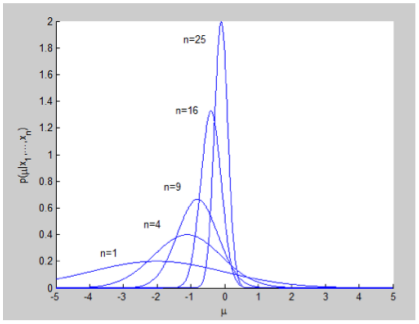
∴ Peak Density @ μ_N, with variance of σ_N^2

∴ Bayesian estimate of μ is $\hat{\mu}_B = \mu_N$.

Observations:

• Bayesian estimate can be interpreted as weighted average of initial guess μ_0 & sample mean m_N.

- If $\sigma_0 = 0$, ⇒ sure of initial guess that we ignore the samples.
  $\sigma_0 > 0$, ⇒ some uncertainty, sample mean m_N has greater dominance
  If $\sigma_0 >> \sigma$, ⇒ initial uncertainty is relatively large, samples weighted heavily.
- As $N \to \infty$, $\sigma_N^2 \to 0$ & $\mu_N \to m_N$
  ↳ # of samples ↗, the density narrows & peaks @ true mean during BL !



As measures arrive, the PDF of the estimate narrows, implying that the estimation error is decreasing.

(Slide 9) Non-parametric

Non-parametric Learning

- Given N samples $\{\vec{x}_i\}$ with labels $\{y_c\}$.
- estimate distribution directly

→ 3 Main categories:
- @ Histogram Estimation → Grp labeled samples into discrete regions to approx. $p(\vec{x})$
- @ Parzen window Estimation → Approx. $p(\vec{x})$ in continuous manner based on the local distribution of each sample, thus controlling resolution along x-axis explicitly, with resolution along PDF axis data dependent.
- @ kNN Estimation.
  ↳ Approx. $p(\vec{x})$ in continuous manner based on contribution of nearest neighbour samples, ⇒ thus controlling res along PDF axis explicitly. with resolution along x-axis data dependent.

Histogram Estimation.

- simplest approach.
- construct normalized histogram.

Derivation / Proof:

- Consider some interval $R = [a, b]$
- If $p(x)$ is const. over this region,
  ⇒ $$Pr(x \in R) = P_R = \int_a^b p(x)\,dx = p(a) \cdot (b-a)$$
- Suppose N samples $\{x_1, \dots x_N\}$ from PDF $p(x)$
- ⇒ # of samples M that fall within region R must obey a Binomial Distribution:
  $$p(M) = \begin{bmatrix} N \\ M \end{bmatrix} P_R^M (1 - P_R)^{N-M}$$

o Log Form:

$$\log p(M) = \log\left[\binom{N}{M}\right] + M \log[p_{IR}] + (N-M)\log[1-p_{IR}]$$

- Derivative = 0:

$$\frac{\partial \log p(M)}{\partial p_{IR}} = \frac{M}{p_{IR}} - \frac{N-M}{1-p_{IR}} = 0$$

$$\Rightarrow (1-p_{IR})M = p_{IR}(N-M)$$

- ML estimate of p_{IR} as:

$$\hat{p}_{IR} = \frac{M}{N}$$

- Recall:

$$p_{IR} = p(a)\cdot(b-a)$$

- ML estimate:

$$\hat{p}_{IR}(x) = \hat{p}(x)\cdot(b-a)$$

$$\Downarrow$$

$$\hat{p}(x) = \frac{M}{N|R|}$$

$|R|$ = size of region R

Steps:

- Given a set of bins R_i:

① Count # of samples M_i that falls into each bin 'i'
② Count # totall of samples 'N'
③ For a particular pattern x,

$$\hat{p}(x) = \frac{M_i}{N|R_i|} \quad \forall\, x \in R_i$$

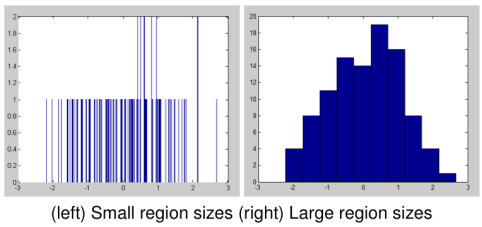
Trade off:

- For good res along x, $\Rightarrow$ small-sized regions $\Rightarrow$ small M_i $\Rightarrow$ poor PDF $p(x)$ resolution.
- For good res along $p(x)$, $\Rightarrow$ large M_i $\Rightarrow$ large-sized region $\Rightarrow$ poor 'x' resolution

Disadvantages:

- Estimated PDF is ~~is~~ continuous.
- Shifting origin changes the shape of the estimated PDF.
- $p(x)$ const. over each region.

Example: 100 samples from a Gaussian distribution



(left) Small region sizes (right) Large region sizes

Parzen Window Estimation

Advantages: ① No need to predefine region sizes
② Estimated PDF is ALWAYS continuous
③ Origin-Independent

Idea:

- Every sample x_i locally influences the estimated PDF in the vicinity of x_i.
- ↳ if observed x_i, PDF can't be too small
  if see lots $\{x_i\}$, PDF in that region large
- ↳ estimated PDF = sum of the contributions.

$$\hat{p}(x) \propto \sum_i \Phi(x - x_i)$$

~ Linear combination of window function (local distributions)

window function. ⇒ controls how observed sample x_i influences the PDF.

Window Function Φ

① Must be normalized:

$$\int_{-\infty}^{\infty} \Phi(x)\, dx = 1$$

② Stretch / Compress the window function ⇒ change the locality of influence of a sample.

$$\Phi\left(\frac{x - x_i}{h}\right)$$

h: scaling factor

③ Keep things normalized:

$$\int_{-\infty}^{\infty} \frac{1}{h} \Phi\left(\frac{x - x_i}{h}\right) dx = 1$$

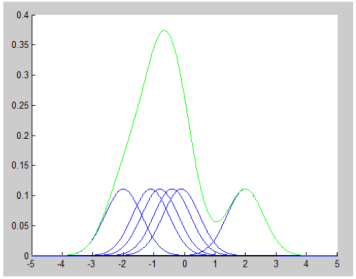
Parzen Window Est.

⇒ Given 'N' samples $\{x_1, \ldots, x_N\}$,

$$\hat{p}(x) = \frac{1}{N} \sum_i \frac{1}{h} \Phi\left(\frac{x - x_i}{h}\right)$$

- Common window func: rect, triangular, gaussian, & exponential.

Example: 6 samples, Gaussian window function



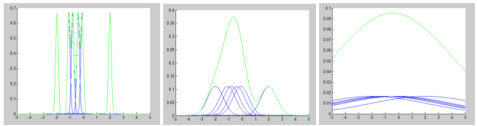
Disadvantages:

① Need to choose a window function & scale factor h.

Rule of Thumb:
- one option, try $h = K/\sqrt{N}$ for some const. K.

② For small N, const K is important
- if too small, the estimate is noisy with sharp peaks at the samples
- if too large, the estimate is smeared with low resolution.

Example: 6 samples, Gaussian window function, different K



(left) small K, (middle) medium K, (right) large K

K-NN Est.

Recall:

{ Histogram : region size
  Parzen Window : window function width } ⇒ explicitly ctrl res along x-axis's,
res along PDF is data independent.

⇓

kNN

- Determine the size of region required at each point to enclose this many samples.

⇓

explicitly ctrls the res along PDF axis.

⇓

x-axis's res is now data dependent.

Procedure:

- create an interval $[x-\alpha, x+\alpha]$, centered around 'x'
- Increase α till it contains a suitable # of observations M
- Compute estimate of $p(x)$ as:

$$\hat{p}(x) = \frac{M}{N|R(x)|} = \frac{M}{N \cdot 2\alpha}$$

where $R(x)$ is the smallest region,
centered around x, which encloses M sample points.

Remark:

- Freqly, we set $M = \sqrt{N}$, ⇒ no free params
  to sample density
  - high ⇒ $|R(x)|$ ↓ small ⇒ high res where it's needed
  - low ⇒ $|R(x)|$ ↑ large ⇒ low res where probly acceptable in sparse regions.

Pro: ① Avoid setting $p(x)$ identically to zero for regions have no samples.
⇒ results in a more realistic non-zero probability.

Con: ① Estimated PDF is highly "peaked" & non-normalized

§ 10/11 - Discriminant Function

Motivations:

Recall

- Let $g(\vec{x})$ be discriminant function

  s.t. two classes classified by: $g(\vec{x}) \gtrless 0$ (A / B)

Idea:

Alternative Approach:

- Assume a particular form for the discriminant functions (ex: hyperplane)
- Use given samples to directly estimate the params of the discriminant functions.
- Given discriminant functions, decision rules & surfaces are defined

Goal: learn discriminant functions directly from the samples.

Linear Discriminant

$g(\vec{x}) = \vec{w}^T \vec{x} + w_0$ ⇒ hyperplane.

↳ $g(\vec{x}) = w_1 x_1 + w_2 x_2 + \cdots + w_n x_n + w_0$

$= \sum_{i=1}^{n} w_i x_i + w_0$

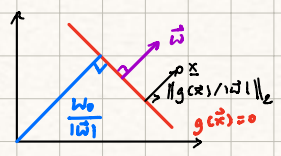
Ex:

Two-class case $\vec{x} = (x_1, x_2)$

⇒ $g(x_1, x_2) \Rightarrow x_2 = -\frac{w_1}{w_2} x_1 - \frac{w_0}{w_2}$

Property:

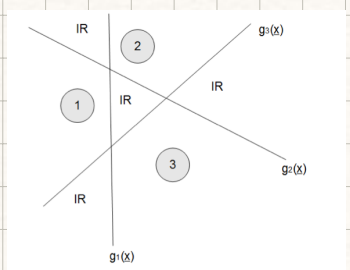
- $\vec{w} \perp$ hyperplane
- $\frac{\vec{w}}{|\vec{w}|}$: unit normal of hyperplane.



Multi-class with Linear Discriminant

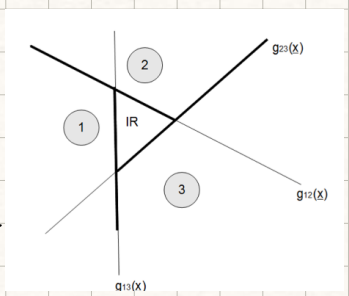
3 Strategies:

① one vs. all $g_i(\vec{x})$ $\forall i$ ⇒ $g_i(\vec{x}) > 0$ if $\vec{x} \in C_i$; $g_i(\vec{x}) < 0$ else.



② one vs. one (in pairs)

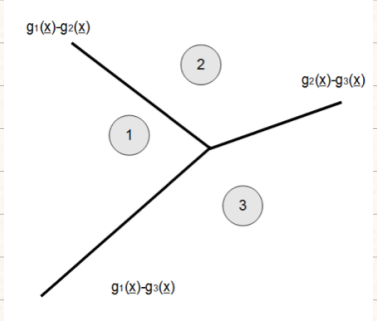
$g_{ij}(\vec{x}) > 0$ $\forall j \neq i$ if $\vec{x} \in C_i$



③ Each class has discriminant func.

$g_i(\vec{x}) > g_j(\vec{x})$ $\forall j \neq i$ if $\vec{x} \in C_i$

↓



← only this one avoids producing indeterminant regions

(3)

× Generalized:

$$\vec{x} \in C_i \quad \text{if} \quad g_i(\vec{x}) > g_j(\vec{x}) \quad \forall j \neq i$$

$$\text{with } g_i(\vec{x}) = \vec{w}_i^T \vec{x} + w_{i0}, \quad i = 1 \cdots K.$$

⇓

× Decision Boundary btwn C_i & C_j : $g_i(\vec{x}) = g_j(\vec{x})$:

$$g_i(\vec{x}) - g_j(\vec{x}) = (\vec{w}_i - \vec{w}_j)^T \vec{x} + w_{j0} - w_{i0} = 0$$

↑ a hyperplane with normal vector: $(w_i - w_j)$!

× Learning Discriminant

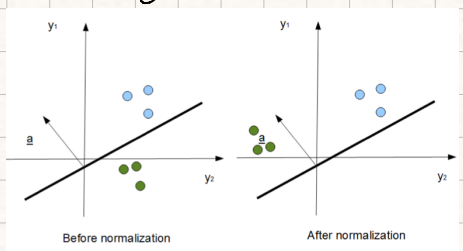
- Assume. classes are linearly separable in the original feature space ($\vec{x}$)

$$\Rightarrow \quad \mathbf{X} = \begin{bmatrix} x_1 \\ \vdots \\ x_n \\ 1 \end{bmatrix} \quad \& \quad \vec{a} = \begin{bmatrix} w_1 \\ \vdots \\ w_n \\ w_0 \end{bmatrix}$$

- In $\mathbf{X}$ space, the decision surface is a hyperplane which contains the origin & has normal vector: $\frac{\vec{a}}{|\vec{a}|}$

- Give labels $\{\mathbf{X}_1, \cdots \mathbf{X}_N\}$,
  find $\vec{a}$ s.t. $\vec{a}^T \mathbf{X}_i > 0 \quad \forall \mathbf{X}_i \in C_1$
  $\vec{a}^T \mathbf{X}_i < 0 \quad \forall \mathbf{X}_i \in C_2$

- Simplification:
  perform normalization by replacing all $\mathbf{X}_i$ by $-\mathbf{X}_i \quad \forall \mathbf{X}_i \in C_2$.
  ⇒ finally $\vec{a}^T \mathbf{x}_i > 0 \quad \forall i$.



× Gradient Descent $\quad \vec{a}_{k+1} = \vec{a}_k - \rho_k \nabla J(\vec{a}_k)$ (J: criterion func.)

- Perceptron: convergance based on sum of dist. of misclassified samples to DB.
- MSE : converges based on sum of squared error.

Non-sequential: update based on all sample @ the same time (offline)

Sequential: based on one sample at time. (online)

Perceptron approach

- **Step 1:** Set an initial guess for the weight vector ($\underline{a}_0$) and let $k = 0$
- **Step 2:** Based on $\underline{a}_k$, construct the classifier and determine the set of misclassified samples $Y(\underline{a})$. If there are no misclassified samples, stop here since we have arrived at the solution. Otherwise, continue to Step 3.
- **Step 3:** Compute a scalar multiple of the sum of misclassified samples $\rho_k \sum_{y \in Y(\underline{a})} (\underline{y})$
- **Step 4:** Determine $\underline{a}_{k+1}$ as

$$\underline{a}_{k+1} = \underline{a}_k + \rho_k \sum_{y \in Y(\underline{a})} (\underline{y}) \tag{30}$$

- **Step 5:** Go to Step 2.

Variations: Fixed-increment: $\rho_k = 1$

Variable-increment: $\rho_k \propto \frac{1}{k}$ → avoid over-shooting

Single sample correction: treat samples sequentially, change weight vector with each misclassification.

Sequential Perceptron approach

- **Step 1:** Set an initial guess for the weight vector ($\underline{a}_0$) and let $k = 0$
- **Step 2:** Based on $\underline{a}_k$, construct the classifier and determine the set of misclassified samples $Y(\underline{a})$. If there are no misclassified samples, stop here since we have arrived at the solution. Otherwise, continue to Step 3.
- **Step 3:** Compute a scalar multiple of the k^{th} misclassified sample $\rho_k \underline{y}^k$
- **Step 4:** Determine $\underline{a}_{k+1}$ as

$$\underline{a}_{k+1} = \underline{a}_k + \rho_k \underline{y}^k \tag{31}$$

- **Step 5:** Go to Step 2.

Minimum squared error approach

- One issue with the perceptron approach is that if the classes are not linearly separable, the learning procedure will never stop since there will always be misclassified samples!
- One way around this is to terminate after a fixed number of iterations, but the resulting weight vector may or may not be appropriate for classification.
- Solution: What if we use a different criterion that will converge even if there are misclassified samples?
- The minimum squared error criterion provides a good compromise in performance for both separable and non-separable problems.

Minimum squared error approach

- Instead of solving a set of inequalities:

$$\underline{a}^T \underline{y}_i > 0, \; i = 1, \ldots, N \tag{43}$$

- we can obtain a solution vector for a set of equations:

$$\underline{a}^T \underline{y}_i = b_i, \; i = 1, \ldots, N \tag{44}$$

- Let the error vector $\underline{e}$ be defined as:

$$\underline{e} = \begin{bmatrix} \underline{y}_1^T \\ \vdots \\ \underline{y}_i^T \\ \vdots \\ \underline{y}_N^T \end{bmatrix} \begin{bmatrix} a_1 \\ \vdots \\ a_i \\ \vdots \\ a_n \end{bmatrix} - \begin{bmatrix} b_1 \\ \vdots \\ b_i \\ \vdots \\ b_N \end{bmatrix} = Y\underline{a} - \underline{b} \tag{45}$$

Minimum squared error approach

- Instead of finding a solution $\underline{a}$ that gives no misclassifications, which could be impossible if it is not a linearly separable problem, we want to find a solution $\underline{a}$ that minimizes $|\underline{e}|^2$.
- This gives us the following sum of squared error criterion function:

$$\nabla J_s(\underline{a}) = |\underline{e}|^2 = |Y\underline{a} - \underline{b}|^2 = \sum_{i=1}^{N} (\underline{a}^T \underline{y}_i - b_i)^2 \quad (46)$$

Minimum squared error approach

- The gradient of $J_s(\underline{a})$ can be written as:

$$\nabla J_s(\underline{a}) = Y^T(Y\underline{a}_k - \underline{b}) \quad (47)$$

- This gives us the weight update formula as:

$$\underline{a}_{k+1} = \underline{a}_k + \rho_k \nabla J_p(\underline{a}) \quad (48)$$

$$\underline{a}_{k+1} = \underline{a}_k - \rho_k Y^T(Y\underline{a}_k - \underline{b}) \quad (49)$$

Minimum squared error approach

- **Step 1:** Set an initial guess for the weight vector ($\underline{a}_0$) and let $k = 0$
- **Step 2:** Determine $\underline{a}_{k+1}$ as

$$\underline{a}_{k+1} = \underline{a}_k - \rho_k Y^T(Y\underline{a}_k - \underline{b}) \quad (50)$$

- **Step 3:** If convergence reached, stop. Otherwise, go to Step 2.

Sequential variant of minimum squared error approach

- Sequential variant of minimum squared error approach:
- **Step 1:** Set an initial guess for the weight vector ($\underline{a}_0$) and let $k = 0$
- **Step 2:** Determine $\underline{a}_{k+1}$ as

$$\underline{a}_{k+1} = \underline{a}_k - \rho_k(b_k - \underline{a}_k^T \underline{y}^k)\underline{y}^k \quad (51)$$

- **Step 3:** If convergence reached, stop. Otherwise, go to Step 2.

Minimum squared error approach: parameter setting

- How do we set up parameters (i.e., ρ_k, b) for the MSE approach?
- Typically, ρ_k decreases with k (e.g., ρ_k/k) to obtain convergence
- In terms of b, useful settings include:
  - Setting $\underline{b}$ as a vector of ones
  - Setting the first N_1 of the N components to N/N_1 and the rest to N/N_2, where N_1 and N_2 are the number of samples in each class (e.g., if there are 10 samples in class 1 and 3 samples in class 2, then the first 10 components of $\underline{b}$ are set to 13/10 and the rest are set to 13/3.

§ 11 - Clustering

* Definitions.

- A set of samples that are similar ⟶ Diff similarity criterions ⟶ diff clustering results
- A region in feature space containing a high density of samples ⟶ peaks in sample density func. are associated with clusters

⇓

Clusters def ⇒ influences the strategies that we use to perform clustering.

* Intuitive Approaches.

① Mixture Density Strategy

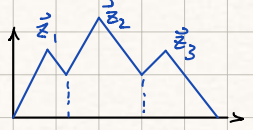
- Def: high sample density region as cluster
- Give a set of $\mathcal{N}$ samples
  - ↳ can estimate combined PDF ($p(\vec{x})$) using density estimation (ex: Parzen Window)
  - ↳ $p(\vec{x})$ refer to as a mixture density, since it's a mixture of the K class densities:

$$p(\vec{x}) = \sum_{i=1}^{K} p(c_i)\, p(\vec{x} \mid c_i)$$

- Intuition:
  - if K classes reasonably distinct & nearly equally likely
    - ⇒ $p(\vec{x})$ should have 'K' peaks / local maxima, one per class.
    - ⇒ Define cluster prototype @ each local maxima:

      $\vec{z}_i = \vec{x}$ s.t. $\vec{x}$ is the i^{th} local max of $p(\vec{x})$.

    - ⇒ Based on the prototypes, we can determine our clusters using dist. metrics (ex: Euclidean Dist.) or cluster boundaries based on local minima.

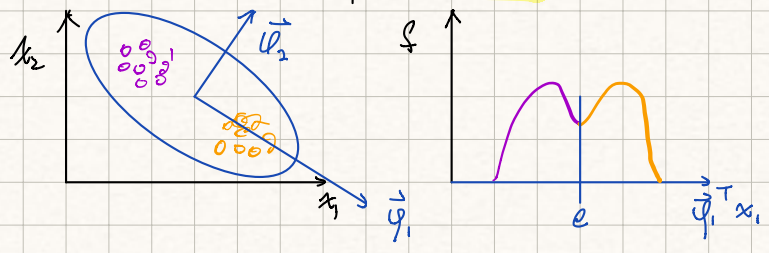


- Cons:
  - In practice, too few samples will result in large # of spurious local maxima
  - Individual sample densities of a particular class may be skewed (ex. Gamma) and so a simple Euclidean distance metric may not define the cluster boundaries well.
  - Peaks of clusters may be too close (or even overlap) to distinguish properly

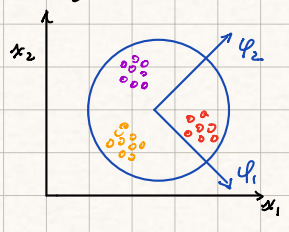
② PCA : Principal Component Analysis.

- Assume we have two fairly compact and distinct clusters.
  - The combined covariance matrix describes the shape of the total sample distributions.
  - If we take the maximum eigenvalue eigenvector $\vec{\Phi}_1$ as the maximum separation direction, we may do the following.
    - Project samples onto $\vec{\Phi}_1$
    - Construct histogram
    - Choose threshold θ @ minimum of distribution, and specify clusters based on the following rule:

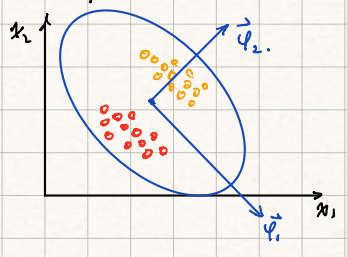
$$\vec{x} \in C_1 \text{ if } \vec{\Phi}_1^T \vec{x} \in \theta.$$



  - Cons:
    - While this clustering scheme is simple & intuitive, it cannot handle more complicated but typical problems such as:
      - Having $K > 2$ clusters: ✗

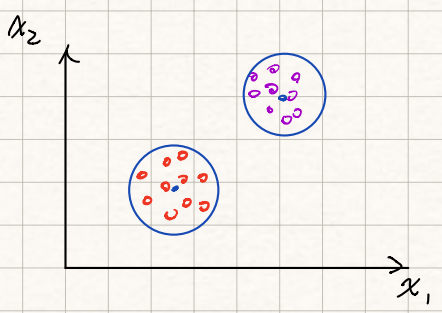


      - Non-spherical distributions ✗



③ Euclidean Dist. Threshold Strategy

- Let's use the definition of clustering based on similarity with each other
- Based on this notion, what we can do is assign a pattern to a cluster if the pattern to a cluster if the dist. between the cluster & the prototype is less than some threshold T.
  - Step 1: Let $\vec{z}_1 = \vec{x}_1$, 1st sample
  - Step 2: For each sample, assign it to closest prototype for which $|\vec{x}_i - \vec{z}_j| < T$, If no such prototype exists, Let $\vec{x}_i$ be a new prototype.
  - Step 3: Repeat step2 till all samples are assigned to clusters.



- Cons:
  - T threshold arbitrary.
    - T small ⇒ many clusters will be found.
    - T large ⇒ few clusters will be found.
  - Sensitivity to the order in which samples are considered (affects specific prototypes)
  - As such, most of these intuitive approaches seem rather arbitrary!
  - To help remedy such issues, we want to develop formal criteria for clustering that are less likely to impose some artificial structure on data, and more sensitive to any natural structural.

× Criteria for Clustering

↳ serve quantitative measure of the clusters.

↳ A given partitioning of the sample set is optimum, when criterion function is maximized/minimized.

↳ commonly used criterion functions:
- Sum of squared errors
- Scatter volume.

× Sum of Squared Error Criteria.

→ favour convergence towards the minimum trace of S_W.

$J_e \rightarrow \min \operatorname{tr}(S_W)$

- For K clusters, cluster C_i has N_i samples.

$$\Rightarrow \boxed{J_i = \sum_{\vec{x} \in C_i} |\vec{x} - \vec{z}_i|^2}$$

↳ The prototype $\vec{z}_i$ which minimize the single class error is just the class mean $\vec{m}_i$

⇒ Total:

$$\boxed{J_e = \sum_{i=1}^{K} J_i = \sum_{i=1}^{K} \sum_{\vec{x} \in C_i} |\vec{x} - \vec{m}_i|^2}$$

- A clustering / partitioning that minimizes this total error J_e is called Minimum Variance Partition.

× Scatter Volume Criteria

→ favour minimum scatter volume of S_W

$J_v \rightarrow \min |S_W|$.

- For class C_i, scatter matrix: $S_i = \sum_{\vec{x} \in C_i} (\vec{x} - \vec{m}_i)(\vec{x} - \vec{m}_i)^T$
- S_i is just N_i times the covariance matrix of class i.
- Summing over K classes, we have a measure of total within class scatter:

$$S_W = \sum_{i=1}^{K} S_i$$

- Note: trace of S_w ≡ sum of squared error criterion J_e

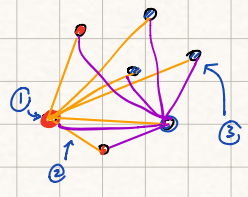
$$tr(S_w) = \sum_{i=1}^{K} tr(S_i) = \sum_{i=1}^{k} \sum_{\vec{x} \in C_i} |\vec{x} - \vec{m}_i|^2 = J_e$$

- Scatter Volume: (the determinant of S_w)

$$J_v = |S_w|$$

→ × Minimum Variance Algo ⇒ minimize sum of sqr error criterion ~ min $tr(S_w)$

→ × K-means Algo



Step 1: Choose prototypes $\{\vec{z}_1, \dots \vec{z}_k\}$ arbitrarily.
Step 2: Euclidean Dist. btwn N samples to each cluster
Step 3: Assign N samples to the K clusters based on min Euclidean Dist.

$$\vec{x} \in C_i \text{ iff } |\vec{x} - \vec{z}_i| < |\vec{x} - \vec{z}_j|, \quad j \neq i$$

Step 4: Compute new cluster prototypes as the cluster means:

$$\vec{z}_{i,new} = \frac{1}{N_i} \sum_{i=1}^{N_i} \vec{x}_i, \quad \forall x \in C_i$$

Step 5: If any clusters prototypes change, go back to Step 2.

Cons:

① Sensitivity to initialization. ⇒ can be reduced by running random initializations multiple times & choosing the best results.
② Assumes # of clusters is known ⇒ can be reduced by starting with large # of clusters and then consolidating them.
③ Sensitive to geometric properties of clusters.

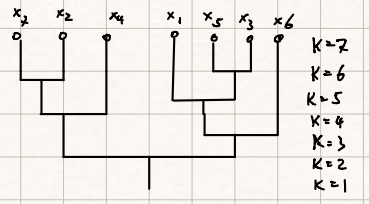
Variants:

- Complex: ISODATA: (Iterative Self Organizing Data Analysis Technique A)
  - if # of samples in any cluster is less than some threshold, the cluster may be eliminated.
  - if max variance feature for the cluster is larger than some threshold and there are sufficient samples in the cluster, split the cluster.
  - if the pairwise dist. btwn clusters is below some threshold, the clusters are combined.
- ISODATA is quite flexible, but needs # of thresholds to be defined.
- ISODATA works best when used in an interactive, empirical env.

Hierarchical Clustering: Simple Example.

- N samples, treat each as a cluster prototype.
- We now find the two most similar clusters & merge them, $\Rightarrow (N-1)$ clusters.
- Continue merging til some criterion is satisfied.
- scheme:

Dendrogram



Measures of similarity for Hierarchical Clustering

- Popular similarity measures based on Euclidean distance.
  (Each measure gives diff. clustering results)

↳ Minimum distance (nearest neighbor)

$$d_{min}(C_i, C_j) = \min_{\vec{x} \in C_i,\ \vec{x}' \in C_j} |\vec{x} - \vec{x}'|$$

- Nearest Neighbor measure well-suited for string-like clusters but highly sensitive to noise and outliers.

↳ Maximum dist (furthest neighbor)

$$d_{max}(C_i, C_j) = \max_{\vec{x} \in C_i,\ \vec{x}' \in C_j} |\vec{x} - \vec{x}'|$$

- tend to discourage growth of elongated clusters.
  since two sub-clusters are only merged if the least similar pair in the resulting cluster is sufficiently similar.
- This makes it less sensitive to outliers & noise
  $\Rightarrow$ well suited for compact, spherical clusters.

↳ Average dist:

$$d_{avg}(C_i, C_j) = \frac{1}{N_i N_j} \sum_{\vec{x} \in C_i} \sum_{\vec{x}' \in C_j} |\vec{x} - \vec{x}'|$$

- less sensitive to noise & outliers.

↳ Dist. btwn means

$$d_{mean}(C_i, C_j) = |\vec{m}_i - \vec{m}_j|$$

- Index-mean measure $\Rightarrow$ less sensitive to noise & outliers
- Most efficient to implement.

Termination Criterion

- Maximum # of levels
- # of clusters
- Error threshold