Chapter 2 PDF Estimation

Parametric Estimation [Maximum Likelihood Estimation
Bayesian Estimation

Non-Parametric Estimation [KNN Estimation
Parzen Window Estimation

Expectation-Maximization Algorithm (EM)

by Mateo Daniel Rodriguez Romero & Zhengji Li

Probability Lingo

$P(A)$ = probability of event a
$P(A|B)$ = probability of event a given event b occured
$P(A, B|C)$ = probability of event a and b given event c occured
$E\{f(y)\} = \int f(x)p(y)dy$

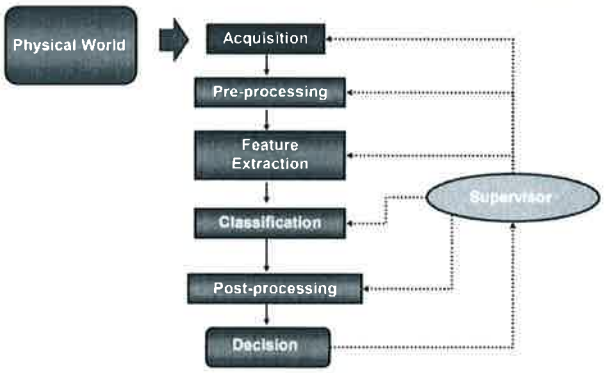
$$P(X, Y|Z) = P(Y|X, Z)P(X|Z) \text{ or } P(X|Y, Z)P(Y|Z)$$

The probability that X and Y happen if we know that Z happens is the same as the probability that X happens when we know that Z happens and that then Y happens when we know that X and Z happen.

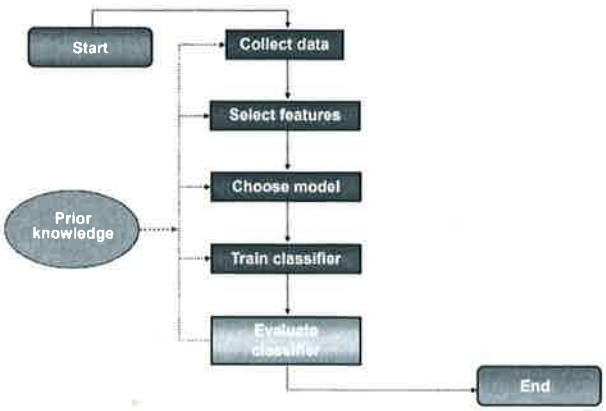
$$p(x|\theta) = p(x|m, \sum) = \frac{1}{\sqrt{(2\pi)^n|\sum|}} exp[-\frac{1}{2\sum}(x-m)^t(x-m)]$$

When the covariance matrix has only the variance in the diagonal means there are no co-variance between features, i.e. they are i.i.d.

ML Procedures.



Preprocessing = filtering / histogram manipulation / Segmentation - Enhancing the quality of the data Transforms can be part of pre-processing or feat. extraction



likelihood function $\prod_{i=1}^{n} p(x_i|\theta)$

Parametric Estimation - *we assume we know the model (gaussian, poisson, ...)*

$$x - N(\mu, \sigma^2) \mid X = \{x1, x2, x3...x_N\} \mid \theta = [\mu, \sigma^2] \mid p(x|\theta) = \frac{1}{\sqrt{2\pi\sigma^2}} e^{\frac{-(x-\mu)^2}{2\sigma^2}}$$

We consider that our $p(X|\theta) = p(x_1, x_2, ...|\theta)$ and that it follows a normal distribution with variance equal to 1 and we want to discover our mean. So graphically we would have an infinite ammount of normal curves for all the infinite values of μ. We get the following equation for the likelyhood function:

$$\prod_{k=1}^{N} p(x_k|\theta) = (\frac{1}{\sqrt{2\pi}})^N e^{(-\frac{1}{2}[(x_k^2 - 2x_k\mu + \mu^2) + (x_{k+1}^2 - ...)])} \quad (1)$$

$$\approx (\frac{1}{\sqrt{2\pi}})^N e^{(-\frac{(\mu-\mu_0)^2}{\alpha})} \quad (2)$$

- – Estimation goodness - The perfect estimation has no bias and no variance
  The bias is the shift of the estimated mean
  For the variance we use:
  * Cramer-Rao bound
  Compute the sharpness of the estimation, this is done with the Fisher information, and then it's compared to the variance of the estimator:

$$I(\theta)_{i,j} = E\{\frac{\partial}{\partial\theta_i}[ln(p(X|\theta))]\frac{\partial}{\partial\theta_j}[ln(p(X|\theta))]\} \quad (3)$$

$$var(\hat{\theta}) \geq \frac{1}{I(\theta)} \quad (4)$$

  * Assymptotic properties

  Consistant estimator = asymptotically efficient $\frac{var(\hat{\theta})}{I^{-1}(\theta)}$ & unbiased $E\{\hat{\theta}\} = \theta$

μ_0 is the mean of all training samples.

- **Maximum Likelihood** - maximize the likelihood function

$\hat{\theta} = \arg\max_{\theta} p(X|\theta) \Rightarrow$ Log-likelihood $\hat{\theta} = \arg\max \ln p(X|\theta)$

mono-dim Example here. for distribution

variance σ known

with observations $\Rightarrow \mu$?

1. Calculate the likelihood function:

$$p(X|\theta) = \prod_{i=1}^{N} p(x_i|\theta) \tag{5}$$

$\ln \prod_{i=1}^{N} \frac{1}{\sigma\sqrt{2\pi}} \exp\left(-\frac{(x_i-\mu)^2}{2\sigma^2}\right)$ (prove back)

$$\propto \ln\left(\prod_{i=1}^{N} p(x_i|\theta)\right) = \sum_{i=1}^{N} \ln(p(x_i|\theta)) \tag{6}$$

$$= \sum_{i=1}^{N} \ln((2\pi)^{-1/2}\sigma^{-1}) - \frac{1}{2}\frac{(x_i-\mu)^2}{\sigma^2} \tag{7}$$

$$L(\theta) = \sum_{i=1}^{N} -\frac{1}{2}\ln(2\pi) - \ln(\sigma) - \frac{1}{2}\frac{(x_i-\mu)^2}{\sigma^2} \tag{8}$$

2. Maximize by derivating by μ and equalising to zero:

$$\frac{\partial L(\theta)}{\partial \mu} = \sum_{i=1}^{N} -\frac{1}{2}\frac{2(x_i-\mu)(-1)}{\sigma^2} = 0 \tag{9}$$

$$\sum_{i=1}^{N}(x_i-\mu) = 0 \tag{10}$$

$$\sum_{i=1}^{N} x_i - \mu N = 0 \tag{11}$$

$$\mu = \frac{\sum_{i=1}^{N} x_i}{N} \tag{12}$$

3. Maximize by derivating by σ and equalising to zero:

$$\frac{\partial L(\theta)}{\partial \sigma} = \sum_{i=1}^{N}\left(-\frac{1}{\sigma} - \frac{1}{2}(x_i-\mu)^2(-2)(\sigma^{-3})\right) = 0 \tag{13}$$

$$\frac{1}{\sigma}\sum_{i=1}^{N}\left(\frac{(x_i-\mu)^2}{\sigma^2} - 1\right) = 0 \tag{14}$$

$$\sum_{i=1}^{N}\frac{(x_i-\mu)^2}{\sigma^2} = N \tag{15}$$

$$\frac{1}{\sigma^2}\sum_{i=1}^{N}(x_i-\mu)^2 = N \tag{16}$$

mean vector

$$\sigma^2 = \frac{1}{N}\sum_{i=1}^{N}(x_i-\mu)^2 \tag{17}$$

covariance matrix

For multivariate gauss. it's replacing μ with $\hat{m}$ and σ^2 for $\sum$ and for the derivative of σ it's $(x_i-\hat{m})(x_i-\hat{m})^t$

Tricky point $\mu \rightarrow$ mean of the observations (estimation)

$\sigma \rightarrow$ variance of the observations (estimation)

In ML Estimation θ fixed
Bayesian θ to be random variable (may have prior density for distribution of θ)

$p(x|X) = \int_\theta p(x,\theta|X)\,d\theta = \int_\theta p(x|\theta,X)\,p(\theta|X)\,d\theta$ (samples $\rightarrow X$)
$= \int_\theta p(x|\theta)\,p(\theta|X)\,d\theta$

$=p(X|Y,Z)\cdot P(Y|Z) \Rightarrow P(X,Y|Z)$

- Bayesian estimation - Weighted average of all possible models

Given observation we have our PRIOR DENSITY $p(\theta)$ with a $max(p(\theta)) = M_1$
Running the X_N samples we get our POSTERIOR DENSITY $p(\theta|X)$ with a $max(p(\theta|X)) = M_2$

Using Bayesian Formula

wrong: when θ is given, distribution of p_x is fixed, so X could be removed

We want to find the estimation of the real distribution: $\hat{p}(x|X) = \int_\theta p(x,\theta|X)d\theta = \int_\theta p(x|\theta,X)p(\theta|X)d\theta$ (X doesn't add any value on the first prob)

~~X is independent from the continuous X,~~ (subset)

So $\int_\theta p(x|\theta)p(\theta|X)d\theta$ which is basically an infinte weighted average of the probability of x given theta weighted by the probability that a given distribution from the samples (?) has theta

$p(\theta|X) = \frac{p(X|\theta)p(\theta)}{p(X)}$

Bayes formula: $p(\theta|X) = \frac{p(X|\theta)p(\theta)}{p(X)} = \frac{p(X|\theta)p(\theta)}{\int p(X|\theta)p(\theta)d\theta}$

$p(X) = \int_\theta p(X|\theta)p(\theta)d\theta$

$p(X|\theta)$ likelihood function

From here it's considered that $\theta = [\mu, \sigma^2]$ where only μ is unknown so we basically replace θ for μ and also that $p(\mu) - N(\mu_0, \sigma_0^2)$ prior knowledge for μ (Independence Assumption)

simplified version of caculation

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

$$p(\mu|X) = \frac{p(X|\mu)p(\mu)}{p(X)} = \frac{(\prod_{i=1}^{N} p(x_i|\mu))p(\mu)}{p(X)} \tag{18}$$

$$= \frac{(\prod_{i=1}^{N} \frac{1}{\sqrt{2\pi\sigma^2}} e^{(\frac{-1}{2\sigma^2}(x_i-\mu)^2)})(\frac{1}{\sqrt{2\pi\sigma_0^2}} e^{(\frac{-1}{2\sigma_0^2}(x-\mu_0)^2))})}{p(X)} \tag{19}$$

$p(x_i|\mu) \sim N(\mu, \sigma^2)$
$p(\mu) \sim N(\mu_0, \sigma_0^2)$

$\Downarrow$ obtain

$p(\mu|X) = \alpha' \exp\{-\frac{1}{2}[(\frac{N}{\sigma^2} + \frac{1}{\sigma_0^2})\mu^2 - 2(\frac{1}{\sigma^2}\sum_{i=1}^{N} x_i + \frac{\mu_0}{\sigma_0^2})\mu]\}$

minus $\frac{1}{2}$

$= \frac{1}{p(X)} (\frac{1}{\sqrt{2\pi\sigma^2}})^N \frac{1}{\sqrt{2\pi\sigma_0^2}} e^{(\frac{-1}{2\sigma^2}\sum_{i=1}^{N}(x_i-\mu)^2 - \frac{1}{2\sigma_0^2}(\mu-\mu_0)^2)}$ ... (20)

However, Remember the answer back (next page)

$e^{(-\frac{1}{2\sigma^2}\sum_{i=1}^{N}(x_i-\mu)^2 - \frac{1}{2\sigma_0^2}(\mu-\mu_0)^2)} \rightarrow -\frac{1}{2}(\frac{1}{\sigma^2}\sum_{i=1}^{N}(x_i-\mu)^2 + \frac{1}{\sigma_0^2}(\mu-\mu_0)^2)$ (21)

$= -\frac{1}{2}(\frac{\sum_{i=1}^{N} x_i^2}{\sigma^2} - \frac{\sum_{i=1}^{N} 2x_i\mu}{\sigma^2} + \frac{N\mu^2}{\sigma^2} + (\frac{\mu^2}{\sigma_0^2} - \frac{2\mu\mu_0}{\sigma_0^2} + \frac{\mu_0^2}{\sigma_0^2}))$ (22)

$-\frac{1}{2}(\mu^2(\frac{N}{\sigma^2} + \frac{1}{\sigma_0^2}) - 2\mu(\frac{\sum_{i=1}^{N} x_i}{\sigma^2} + \frac{\mu_0}{\sigma_0^2}) + \frac{\sum_{i=1}^{N} x_i^2}{\sigma^2} + \frac{\mu_0^2}{\sigma_0^2})$ (23)

Since $p(\mu|X)$ is a normal distribution $N - (\mu_N, \sigma_N^2)$, we can realte its exponential parts:

$$\frac{1}{\sqrt{2\pi\sigma_N^2}} e^{(\frac{-1}{2\sigma_N^2}(\mu-\mu_N)^2)} = \frac{1}{\sqrt{2\pi\sigma_N^2}} e^{(-\frac{1}{2}(\frac{\mu^2}{\sigma_N^2} - \frac{2\mu\mu_N}{\sigma_N^2} + \frac{\mu_N^2}{\sigma_N^2}))} \tag{24}$$

where:

$$\mu^2\left(\frac{N}{\sigma^2}+\frac{1}{\sigma_0^2}\right)=\frac{\mu^2}{\sigma_N^2}\rightarrow\sigma_N^2=\frac{\sigma_0^2\sigma^2}{N\sigma_0^2+\sigma^2} \tag{25}$$

and

$$-2\mu\left(\frac{\sum_{i=1}^{N}x_i}{\sigma^2}+\frac{\mu_0}{\sigma_0^2}\right)=-\frac{2\mu\mu_N}{\sigma_N^2} \tag{26}$$

$$\mu_N=\frac{\sigma_N^2\sum x_i}{\sigma^2}+\frac{\sigma_N^2\mu_0}{\sigma_0^2}=\frac{\sigma_0^2\sigma^2}{N\sigma_0^2+\sigma^2}\frac{\sum x_i}{\sigma^2}+\frac{\sigma_0^2\sigma^2}{N\sigma_0^2+\sigma^2}\frac{\mu_0}{\sigma_0^2} \tag{27}$$

$$=\frac{\sigma_0^2\sum x_i}{N\sigma_0^2+\sigma^2}+\frac{\sigma^2\mu_0}{N\sigma_0^2+\sigma^2}\;\Big|\;\frac{\sum x_i}{N}=\overline{X} \tag{28}$$

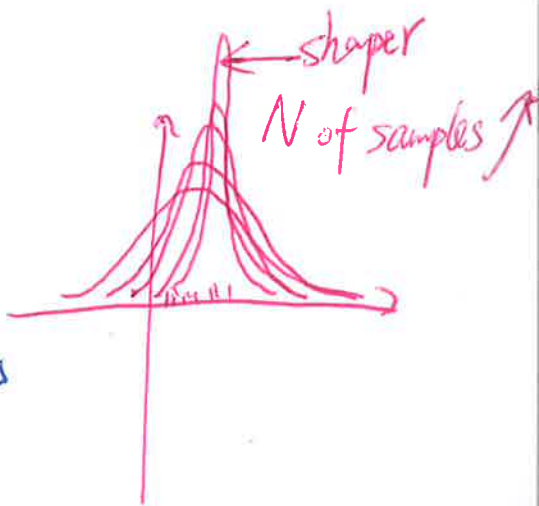
$$=\frac{\sigma_0^2N\overline{X}}{N\sigma_0^2+\sigma^2}+\frac{\sigma^2\mu_0}{N\sigma_0^2+\sigma^2} \tag{29}$$

From here, we go back to the start: $\hat{\mathrm{p}}(x|X)=\int_\theta p(x,\theta|X)d\theta=\int_\theta[p(x|\theta)-N(\mu,\sigma^2)][p(\theta|X)-N(\mu_N,\sigma_N^2)]d\theta$ so we get that $\hat{\mathrm{p}}(x|X)-N(\mu_N,\sigma_N^2+\sigma^2)$

ML vs. Bayessian

ML is a consistent estimator, given high number of training samples and it's fairly simple to implement. Bayessian estimation is usefull if we have prior information, which would be discarded in ML. (normally used with low number of samples)

Answer $\begin{cases}\mu_N=\left(\frac{N\sigma_0^2}{N\sigma_0^2+\sigma^2}\right)\overline{X}+\left(\frac{\sigma^2}{N\sigma_0^2+\sigma^2}\right)\mu_0\\ \sigma_N^2=\frac{\sigma_0^2\sigma^2}{N\sigma_0^2+\sigma^2}\end{cases}$ properties



$\lim_{N\to\infty}\mu_N=\overline{X}\quad\sigma_N^2=0$ means distribution obey the distribution of samples

$\lim_{N\to0}\mu_N=\mu_0\quad\sigma_N=\sigma_0$ when there is no posterior knowledge use the prior knowledge.

ML > Bayesian in practical
Lower computional complexity
Easier interpretability

Non-Parametric Estimation

We have no knowledge of the pdf.

Considering x^* is a random sample that belongs to the region R, and R has an infintesimally small volume V:

$$P_R = \int_R p(x^*)dx^* \rightarrow p(x^*)V \tag{30}$$

The estimation simply counts the number of samples from the training set that are inside this volume, so we can write this as:

$$p(x^*)V = \frac{K}{N} \rightarrow p(x^*) = \frac{K}{NV} \tag{31}$$

Where N is the total number of samples and K the number of samples inside the volume

- **K-Nearest Neighbour**

  Fixes K to a constant, and expands the volume until a number of K samples are inside

- **Parzen windows**

  Fixes V to a constant, and counts the number of K samples inside.

  For this we need a window function (can have different shapes):

$$\gamma(x) = \{1; 0\} \rightarrow \gamma[\frac{x_k - x^*}{h}] \tag{32}$$

$$p(x^*) = \frac{\sum_{k=1}^{N} \gamma[\frac{x_k - x^*}{h}]}{Nh^n} \tag{33}$$

Where $V = h^n$ since it has sides length h in n dimensions

Developing the equation of the Bias, i.e. $E\{\hat{p}(x)\} = p(x)$:

$$\frac{1}{NV(h)} \sum_{i=1}^{N} E\{\gamma[\frac{x - x_i}{h}]\} \tag{34}$$

$$\frac{N}{NV(h)} E\{\gamma[\frac{x - y}{h}]\} = \frac{1}{V(h)} \int \gamma[\frac{x - y}{h}]p(y)dy = \frac{1}{v(h)} p(x) * \gamma[\frac{x}{h}] \tag{35}$$

Since the expectation of the window is the same regardless of the sample, we use y and remove the sum.

$\hat{p}(x) = \frac{1}{N}\sum_{k=1}^{N}\frac{1}{V(h)}\left(\frac{x-x_k}{h}\right)$ $\quad \gamma(x) \geq 0 \;\forall x \in R^n \int_{R^n} \gamma(x)\,d(x) = 1$

For the variance it's used an uper bound, and it's unbiased when

$$var(\hat{x}) \leq \frac{\cdots}{Nh^n} \tag{36}$$

$$\lim_{N\to+\infty} h_N = 0 \tag{37}$$

$$\lim_{N\to+\infty} Nh_N^n = +\infty \;\rightarrow var(\hat{x}) \leq 0 \tag{38}$$

This means that the window acts as a filter, the smaller h the closer it is to an impulse and the lower the bias, thus why h^n should tend to 0

Ex: $h_N = \frac{1}{\sqrt[2n]{N}}$

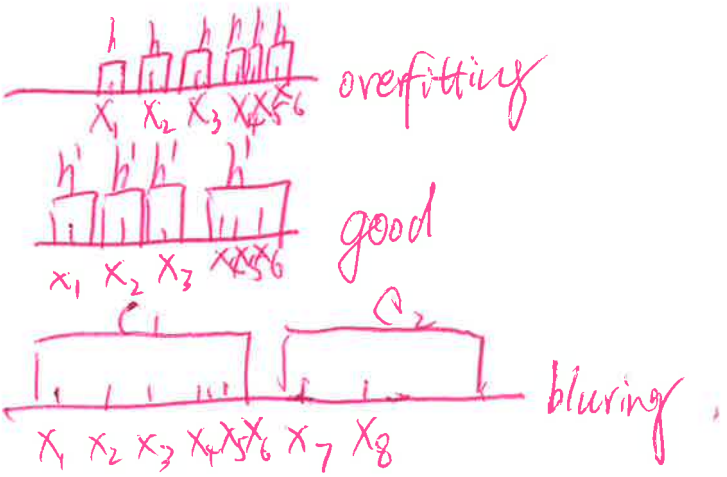
K and h increase = underfitting/oversmoothing/bluring

K and h decrease = overfitting

approximation

windowsize ↑ bluring performance.

window size ↓ overfitting



Estimation with incomplete data

Considering

$$p(x) = \sum_{i=1}^{M} P_i p(x|m_i, \sum_i) \tag{39}$$

as a mixture of gaussian func. where P_i is the prior probability, m_i the mean vector and $\sum_i$ the covariance matrix.

Objective is to estimate the distribution of each gaussian. We consider $Z = (X, Y)$ where X are the samples and Y is the missing information

- **Expectation maximization** - more elaborate technique than ML

  Expectation step (k iteration index):

$$E\{ln(p(X, Y|\theta))|X, \theta^k\} \tag{40}$$

$$= \int ln[p(X, Y|\theta)]p(Y|X, \theta^k)dY = Q(\theta, \theta^k) \tag{41}$$

  Read as assuiming X samples we have and theta at iteration k

  This is "expectation" step and now the maximization step:

$$\theta^{k+1} = arg\max_{\theta} Q(\theta, \theta^k) \tag{42}$$

  Ex: To estimate θ we should compute the maximum likelihood on Z and to "get" Y we average ($E\{\}$) on what we have (X)

  Considering $\theta = [m, \sum]$; $m = [\mu_1, \mu_2]$ and $\sum = [[\sigma_1^2, 0][0, \sigma_2^2]]$

$$Z = \{[0, 2], [1, 0], [2, 2], [x_{4,1}, 4]\} \mid x_{4,1} = Y \tag{43}$$

$$Q(\theta, \theta^0) = \int ln(p(x_1, x_2, x_3, x_4|\theta)p(x_{4,1}|X, \theta^0)d_{x_{4,1}} \tag{44}$$

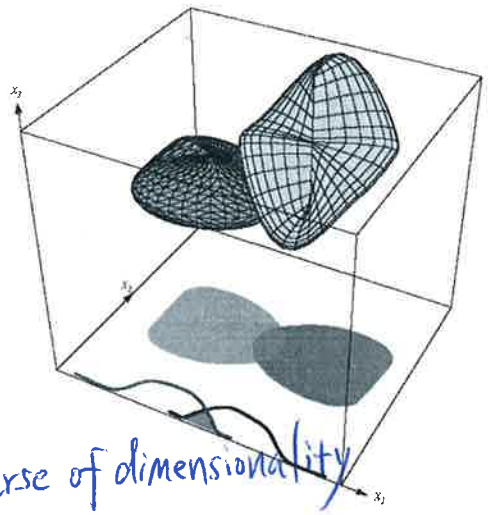
$$\int ln[\prod_{i=1}^{3} p(x_i|\theta)p(x_4|\theta)]p(x_{4,1}|\theta^0)d_{x_{4,1}} \tag{45}$$

$$\sum_{i=1}^{3}[\int ln(p(x_i|\theta))p(x_{4,1}|\theta^0)d_{x_{4,1}}] + \int ln(p(x_4|\theta))p(x_{4,1}|\theta^0)d_{x_{4,1}} \tag{46}$$

$$\sum_{i=1}^{3} ln(p(x_i|\theta)) + \int ln(p(x_{4,1}, x_{4,2}|\theta))p(x_{4,1}|\theta^0) \tag{47}$$

Feature reduction Chapter3

Adding features can help separate data:



There is a non-zero Bayes error in the 1-D x_1 space or the 2-D x_1, x_2 space. However, the Bayes error vanishes in the 3-D x_1, x_2, x_3 space because of non-overlapping densities.

=> also called curse of dimensionality

Hughes effect: adding more features to separate data becomes a dissadvantage after a specific point Why? <- classification accuracy affected
computational complexity

- Adding features changes the distribution type of data, i.e. no longer gaussian
- Overfitting

So we reduce features to not get this problems and reduce computational complexity

The reason behind this is that using more dimensions requeires more training data, since in each dimension increases volume. Consider a circle inside a square:

- 1D the circle and the square are a line of lenght = 2R = L
- 2D the circle area is now $A_c = \pi R^2$ and the quare is L^2 which gives a difference of $(1 - \frac{\pi}{4})$
- 3D the sphere inside the cube gives a differenc of

by Mateo Daniel Rodriguez Romero & Zhengji Li

separability measure $J(\cdot)$
search strategy in the solution space
3 Typical Separability measures (in Chinese 散度)
- Divergence
- Bhattacharyya distance
- Jeffres-Matusita distance (JM)

Feature selection

Consider $F = \{X1, X2, X3...Xn\}$ with n Features and f_k and $f_{\underline{k}}$ being k selected elements and $\underline{k}$ eliminated ones thus $k + \underline{k} = n$

We want to pick the features that SEPARATE the best possible the classes so we want F^*:

$$F^* = argmax\{J(F')\} \quad \text{opportune function} \tag{48}$$

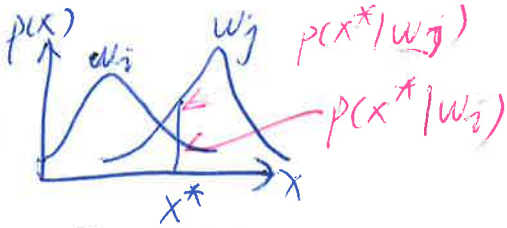
F' is the subset of F with the selected features

Types of $J()$s

- Divergence:

We use the likelihood ratio: $\frac{p(x|w_1)}{p(x|w_2)}$ $L_{ij}(X) = \frac{p(x|w_i)}{p(x|w_j)}$ class

overlap degree



Multi
① Dave $\sum_{i=1}^{c} \sum_{j=1}^{c} P_i P_j D_{ij}$
② worst case

If the distributions are overlaped the ratio will be =1, else if it's very big or small it means there is low overlap

But this is no simetric (1/2 ≠ 2/1), so we use 2 ratios:

integration

$D_{ij}(F') = E\{L'_{ij}(x)\} + E\{L'_{ji}(x)\} = E\{\ln p(x|w_i) - \ln p(x|w_j)\} + E\{\ln(x|w_j) - \ln(x|w_i)\}$

L' means ln

$$D_{1,2} = E\{ln\frac{p(x|w_1)}{p(x|w_2)}|w_1\} + E\{ln\frac{p(x|w_2)}{p(x|w_1)}|w_2\} \tag{49}$$

use basic

$$\int_x \{p(x|w_i) - p(x|w_j)\} ln\frac{p(x|w_1)}{p(x|w_2)} p(x|w_1)dx + \int ln\frac{p(x|w_2)}{p(x|w_1)} p(x|w_2)dx \tag{50}$$

$$ln\frac{p(x|w_i)}{p(x|w_j)} dx \int ln\frac{p(x|w_1)}{p(x|w_2)}[p(x|w_1) - p(x|w_2)]dx \tag{51}$$

Properties: ① $w_i = w_j$ $D_{ij} = 0$ ③ $D_{ij} = D_{ji}$
② $w_i \neq w_j$ $D_{ij} > 0$ ④ #(k) ↑, D ↑
⑤ Independent

- Bhattacharyya:

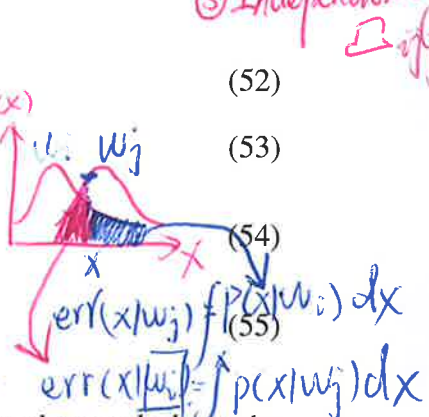
Minimize the bayess error using the area of overlap between two distributions with a crossing point in T:

$$p(err|w_1)P(w_1) + p(err|w_2)P(w_2) \tag{52}$$

$$p(err|w_2) = \int_T^{+\infty} p(x|w_1)dx \tag{53}$$

$$p(err|w_1) = \int_{-\infty}^{T} p(x|w_2)dx \tag{54}$$

$$\int_{-\infty}^{+\infty} min\{p(x|w_1)P(w_1), p(x|w_2)P(w_2)\}dx \tag{55}$$



To solve this we would need the real densities and probabilities, and we only have the estimates, and solving the integral is complex so we use the upper bound ε_u i.e. the Chernoff bound. The smallest the value the better separability

Find an upper bound $\min\{a,b\} \Rightarrow$ inequality $\min\{a,b\} \leq a^s b^{1-s}, 0 \leq s \leq 1$
Chernoff bound

$$min\{a,b\} \leq a^s b^{1-s} \;\; ;0 \leq s \leq 1 \tag{56}$$

$$\int [p(x|w_1)P(w_1)]^s [p(x|w_2)P(w_2)]^{1-s} dx \tag{57}$$

$$= P(w_1)^s P(w_2)^{1-s} \int p(x|w_1)^s p(x|w_2)^{1-s} dx \tag{58}$$

$$= P(w_1)^s P(w_2)^{1-s} e^{-\mu_{1,2}(s)} \tag{59}$$

Where for a gaussian distribution the chernoff distance $\mu_{1,2}(s) =$

$$\frac{s(1-s)}{2}(m_1-m_2)^t\{s\sum_1+(1-s)\sum_j\}^{-1}(m_1-m_2)+\frac{1}{2}ln(\frac{|s\sum_1+(1-s)\sum_j|}{|\sum_1|^s|\sum_2|^{1-s}}) \tag{60}$$

A especific case is where $s = 0.5$ where we get the battacharyya bound, and for gaussians the equation above with 0.5 is called battacharyya distance $B_{1,2}$ and it as the following properties:

$$B_{1,2} = \mu_{1,2}(0.5) \tag{61}$$

$$w_1 = w_2 \rightarrow B_{1,2} = 0 \tag{62}$$

$$w_1 \neq w_2 \rightarrow B_{1,2} > 0 \tag{63}$$

$$B_{1,2} = B_{2,1} \tag{64}$$

And it has no saturating behaviour just like divergence

- Jeffries-Matusita: Basically find one that saturates, so it's the MSE of the desities:

$$JM^2_{1,2} = \int (\sqrt{p(x|w_1)} - \sqrt{p(x|w_2)})^2 dx \tag{65}$$

$$\int p(x|w_1)dx - 2\int \sqrt{p(x|w_1)p(x|w_2)}dx + \int p(x|w_2)dx \tag{66}$$

$$1 + 1 - 2\int \sqrt{p(x|w_1)p(x|w_2)}dx \tag{67}$$

This is written as a function of the Battacharyya distance (because):

$$\int \sqrt{p(x|w_1)p(x|w_2)}dx = e^{-B_{1,2}} \tag{68}$$

$$JM_{1,2} = \sqrt{2(1-e^{-B_{1,2}})} \tag{69}$$

JM saturates at $\sqrt{2}$

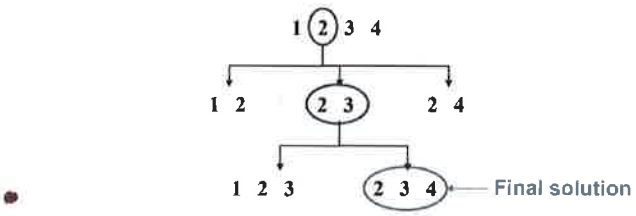
Search Strategies

Used to find the best subset of features that optimize the crierion

- SFS - Sequential forward selection

  1. Pick the best mean distance in a 1D/1 feautre plain, meaning along each feature calculate the distance between each class and calculate the mean. Then compare all the means from all the features as pick the highest one.
  2. Pick the best mean distance using the previously stored one, now in 2D/2 features, meaning the selected feature is now the one of the axis and the other is the other feature you're testing
  3. keep repeating

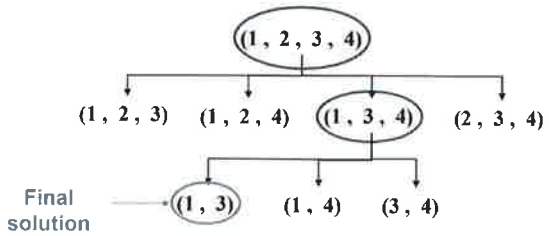
  - Example: m=3 ; n=4



- SBS

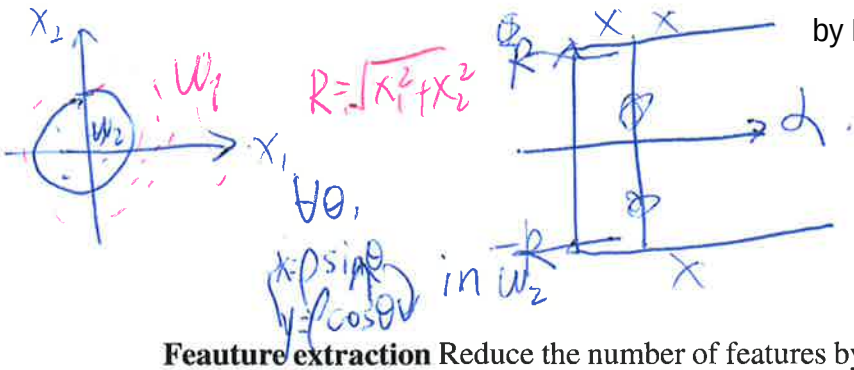
  Pick a group of features and progressively remove one till you find the best one

  - Example: m=2 ; n=4



Consider n features and m subsets of features, the best search depends on these numbers

SFS is better if you want to go up to $m = n/2$, else SBS is better, since it's faster



Feauture extraction Reduce the number of features by using transformations, not eliminating any possible relevant data

- PCA - Principal Component Analysis

  Lingo: best represents the data in least square sense

$$det(\sum -\lambda I) = 0 \rightarrow find\ eigenvalues\ aka\ char.equation \tag{70}$$

  特征值

  Best represent the data in less features. It is unsupervised ~~i~~.e. no labels are needed

  Apply a transformation to the data:

  ① Reconstruct Samples
  ② Minimize the error of the distance between the reconstructed data with real data

$$x = \sum_{i=1}^{n} y_i \phi_i \tag{71}$$

  x_i ← reconstructed
  ϕ_i ← bases for the subspace
  y_i ← x_i 的 Φy_i

$$x' = \sum_{i=1}^{m} y_i \Phi_i \ | m < n \tag{72}$$

$$error(J_m) = \sum_{i=1}^{N} ||x_i - x'||^2 \tag{73}$$

  x_i is represented by reconstruction

  "I will project x along each phi, and each projection will give me each value y_i. These values together will provide me the new coordinates for x"

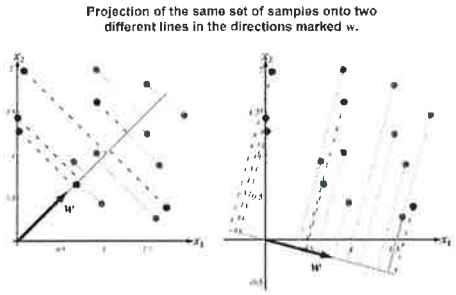
  What are the phi ks so that this is minimized? we pick the eigenvectors, i.e. the directions where the variance is the highest, i.e. where the entropy is highest

  1. Calculate the baricenter (mean of all data) 重心(均值)
  2. Use this to move the data to the center of the space by $x' = x - m$ (Data Shifting)
  3. Calculate the covariance matrix (协方差矩阵)
  4. ☆ Calculate the eigen values and with each of them calculate their eigen vector (特征值/特征向量)
  5. Calculate $y = \Phi^t x$ ex: (变换)
     $\Phi = [\phi_1, \phi_2]$ $x = [x_1, x_2]$
     $\phi_1 = [n_1, n_2]$ $\phi_2 = [n_3, n_4]$
     $y = [y_1, y_2]$ $y_1 = x_1 n_1 + x_1 n_2$ $y_2 = x_2 n_3 + x_2 n_4$
     In the case of reducing to one dimension we would only use ϕ_1 and thus the output will only be y_1

- LDA - Linear Discriminant Analysis

  Best separates the data in less features

  Goal: find a projection to a line w where the points of the classes are separated, where $y = W^t x$



To determine how separated the classes are we use the **Fisher's linear discriminant** and we want to MAXIMIZE IT WITH RESPECT TO w:

$$\frac{||m_1' - m_2'||^2}{\sigma_1^{2'} + \sigma_2^{2'}} \rightarrow J = \frac{||m_1' - m_2'||^2}{S_1' + S_2'} \tag{74}$$

$$m' = W^t m \tag{75}$$

$$W^t S_b W = W^t (m_1 - m_2)(m_1 - m_2)^t W \tag{76}$$

$$(W^t m_1 - W^t m_2)(W^t (m_1 - m_2)) = (m_1' - m_2')^2 \tag{77}$$

$$\text{where } W^t m_1 = m_1' \ \& \ W^t m_2 = m_2' \tag{78}$$

Where $S_w = S_1 + S_2$ is the scatter matrix within class and $S_b = (m_1 - m_2)(m_1 - m_2)^t$ between classes.

The same done in eq 53-55 can be done for S_w:

$$W^t S_w W = W^t (S_1 + S_2) W = W^t S_1 W + W^t S_2 W \tag{79}$$

$$ex\ for\ S_1 \rightarrow W^t \Big(\sum_{x\ \epsilon\ class_1} (x - m_1)(x - m_1)^t \Big) W \tag{80}$$

$$= \sum_{x\ \epsilon\ class_1} (W^t x - W^t m_1)(W^t (x - m_1)) \tag{81}$$

$$= \sum_{x\ \epsilon\ class_1} (W^t x - W^t m_1)^2 = \sum_{x\ \epsilon\ class_1} (y - m_1')^2 = S_1' \tag{82}$$

and we can write $J(w)$:

$$J(w) = \frac{w^t S_b w}{w^t S_w w} \tag{83}$$

And maximized (i.e. the w direction that maximizes the separability) is
$w = S_w^{-1}(m_1 - m_2)$

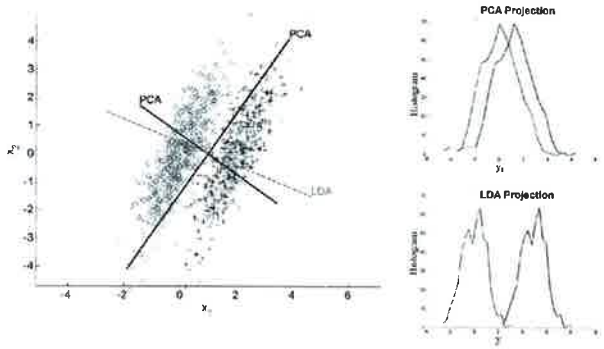
*For more classes/dimension it's $|J(w)| \rightarrow$ maximized are the eigen vectors of $w = S_w^{-1}S_B$

$$y = W^t x \mid W = [w_1, w_2, w_3, ...w_{c-1}] \tag{84}$$

$$S_b = n_1 S_{b1} + n_2 S_{b2} + ...n_c S_{bc} \mid S_w = S_1 + S_2 + S_3 + ...S_c \tag{85}$$

Where n's are the number of samples per class, so S_b is a weighted avg.

$J(W)$ is the same formula only with W being the vector of w's and to calculate it as an escalar we use the determinant of the parts of the numerator and the denominator



Multiclass Case

C classes $\Rightarrow$ C-1 discriminant functions

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

$$S_B = \sum_{i=1}^{c} [\#(X_i)](m_i - m)(m_i - m)^t$$

Global mean vector

$$m = \frac{1}{N} \sum_{\forall x} x$$

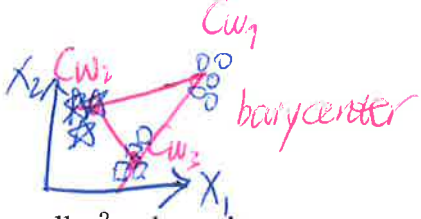
Chapter 4 Supervised Classification

Intro. design of a supervised classifier depends on: ① available set of features ② ~ training samples ③ classification model ④ cost function to optimize

Bayesian Classification
Minimum Risk Theory
Discriminant Functions
Decision Trees
Accuracy Evaluation

by Mateo Daniel Rodriguez Romero & Zhengji Li

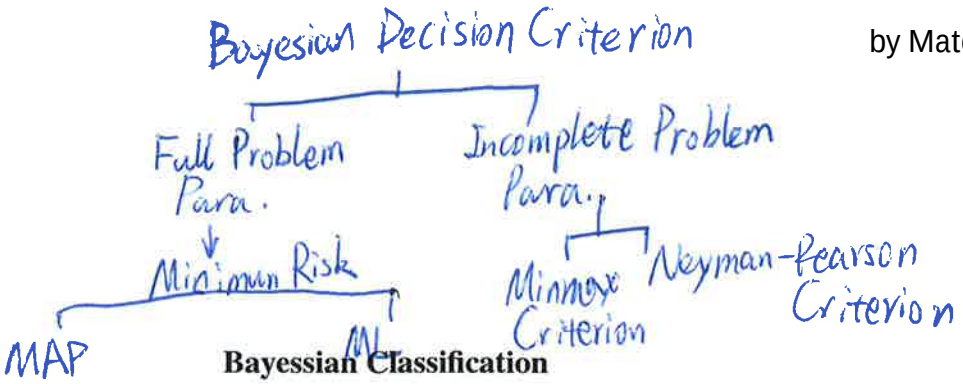
Supervised Classification



- Parametric
  - MDM - minimal distance to means Classes should have a small σ^2 or have the same shape
    1. Calculate the mean of each class (aka baricenter aka class prototype)
    2. Given a new point calculate the distance to each of the class prototypes
    3. Assign it to the closest one
  - Box

    1. Calculate the baricenter of each class
    2. Calculate the σ^2 of the data in the direction of each axis
    3. Create boxes with size of $\alpha\sigma^2$
    4. Given a new point assign it in the box it falls

    *The boxes don't necessarilly have equal length and witdh
    *If a new point is outside of all boxes it can be assigned to the closest one or left unclassified
    *Boxes can't overlap
  - Bayessian Use all available data for classification
    SEE NEXT PAGE
- Non Parametric
  - Bayessian
  - DT
  - Neural Nets
  - SVM



Bayessian Classification

- Full problem parametrization
  - Minimum risk (explained after)
    - MAP Criterion - Maximum a posteriori aka Minimum error crit.
      - It's called both minimum and maximum because they can be seen both ways, in the case of the maximum a posteriori probability the equation comes from:

$$\hat{w} = arg\max_{w_i}\{P(w_i|x)\} \tag{86}$$

rewrite the posterior probability ($P(w_i|x)$) as the prior ($P(w_1)$)

$$P(w_1|x) = \frac{P(w_1)p(x|w_1)}{p(x)} \tag{87}$$

$$\hat{w} = arg\max_{w_i}\{p(x|w_i)P(w_i)\} \tag{88}$$

$$\rightarrow p(x|w_i)P(w_i) \geq p(x|w_j)P(w_j) \tag{89}$$

1. Now with the other definition, the minimum error criterion starts by using the total probability theorem:

$$P_{err} = P(x \rightarrow w_2|w_1)P(w_1) + P(x \rightarrow w_1|w_2)P(w_2) \tag{90}$$

$$= \int_{T}^{+\infty} p(x|w1)P(w1)dx + \int_{-\infty}^{T} p(x|w_2)P(w_2)dx \tag{91}$$

$$= \int_{T}^{+\infty} P(w_1|x)p(x)dx + \int_{-\infty}^{T} P(w_2|x)p(x)dx \tag{92}$$

in eq 58-59 we change from posterior to prior

2. Defining the probability of error:

$$P_{err} = E\{P(err|x)\} = \int_{-\infty}^{+\infty} P(err|x)p(x)dx \tag{93}$$

$$= \int_{-\infty}^{+T} P(err|x)p(x)dx + \int_{T}^{+\infty} P(err|x)p(x)dx \tag{94}$$

Equalizing this equation to the one in step 1 we reach to the conlusion that:

3. The probability of error can only be given by $P(w_1|x)$ or $P(w_2|x)$, so since for a given point one will be higher than the other we minimize it:

$$P(err|x) = min\{P(w_1|x), P(w_2|x)\} \tag{95}$$

4. final equation:

$$P_{errmin} = \int_{-\infty}^{+\infty} min\{P(w_1|x), P(w_2|x)\}p(x)dx \tag{96}$$

* ML Criterion - Maximum likelyhood criterion
  1. Usign the same definition as MAP:

$$\hat{w} = arg\max_{w_i}\{P(w_i|x)\} = arg\max_{w_i}\{p(x|w_i)\} \tag{97}$$

$$\rightarrow p(x|w_i) \geq p(x|w_j) \tag{98}$$

but the difference is that we assume both distributions have the same prior probability $P(w_i)$

Minimum risk theory:
Depending on the application there can be a related cost to assigning incorrectly a new point of data. With this we can integrate this information:

$$A = \{\alpha_1, \alpha_2, ..., \alpha_R\} \tag{99}$$

*Elements in the diagonal are correctly assigned and the data in the matrix should

no-call call fireman

$\Lambda = \begin{bmatrix} 0 & 1000\text{euros} \\ 10^8 & 0 \end{bmatrix}$

nofire nofire

It is thus possible to define a $R \times C$ cost matrix Λ:

$$\Lambda = \begin{bmatrix} \lambda(\alpha_1|\omega_1) & \lambda(\alpha_1|\omega_2) & \cdots & \lambda(\alpha_1|\omega_C) \\ \lambda(\alpha_2|\omega_1) & \lambda(\alpha_2|\omega_2) & & \lambda(\alpha_2|\omega_C) \\ \vdots & & \ddots & \vdots \\ \lambda(\alpha_R|\omega_1) & \lambda(\alpha_R|\omega_2) & \cdots & \lambda(\alpha_R|\omega_C) \end{bmatrix}$$

← row 1 take action 1 given different situations

$\lambda_{ij} = \lambda(\alpha_i|\omega_j)$ ($\lambda_{ij} \geq 0$) is the cost (loss) incurred for taking action α_i given the class is ω_j.

be "provided by an expert in the domain"

1. Calculate the conditional risk/expectation:

given observation $\Rightarrow_c$ taking action

$$R(\alpha_i|x) = \sum_{j=1} \lambda(\alpha_i|w_j) P(w_j|x) \tag{100}$$

(101)

2. Compare which one is lower (ex with two options):

$$R(\alpha_1|x) \leq R(\alpha_2|x) \tag{102}$$
$$\lambda_{1,1}P(w_1|x) + \lambda_{1,2}P(w_2|x) \leq \lambda_{2,1}P(w_1|x) + \lambda_{2,2}P(w_2|x) \tag{103}$$

$$\lambda_{1,1}p(x|w_1)P(w_1) + \lambda_{1,2}p(x|w_2)P(w_2) \leq \lambda_{2,1}p(x|w_1)P(w_1) + \lambda_{2,2}p(x|w_2)P(w_2) \tag{104}$$

$$\frac{p(x|w_1)}{p(x|w_2)} P(w_1)(\lambda_{1,1} - \lambda_{2,1}) \leq P(w_2)(\lambda_{2,2} - \lambda_{1,2}) \tag{105}$$

$$\frac{p(x|w_1)}{p(x|w_2)} > || < \frac{P(w_2}{P(w_1)} \frac{(\lambda_{2,2} - \lambda_{1,2})}{(\lambda_{1,1} - \lambda_{2,1})} \tag{106}$$

If $>$ assign it to w_1, if $<$ assign it w_2.
*If $\lambda_{1,2}$ and $\lambda_{2,1}$ are equal we get the MAP formula

$\lambda_{12} = \lambda_{21}$ MinRisk = MAP

Minimum risk criterion
if $\lambda_{22} = \lambda_{11} = 0$ (no punishment for correct d)

d_1 $\Lambda(x) = \frac{P(x|w_1)}{P(x|w_2)} > \frac{P(w_2)}{P(w_1)} \frac{\lambda_{12}}{\lambda_{21}}$

Discriminant Functions

A classifier assigns a feature vector x to a class that has a higher discriminant function value, the discriminant functions ($g_i(x)$) we've seen:

- Minimize conditional risk: $g_i(x) = -R(\alpha_i|x)$
- Minimize error: $g_i(x) = P(w_i|x) \rightarrow ln(p(x|w_i)P(w_i))$
  - From here, we'll consider **MAP** as our discriminant function, and throw a $ln()$ cause it's gaussian

remember

$$p(\mathbf{x}|\omega_i) = \frac{1}{(2\pi)^{n/2}|\Sigma_i|^{1/2}} \exp\left[-\frac{1}{2}(\mathbf{x}-\mathbf{m_i})^t \Sigma_i^{-1}(\mathbf{x}-\mathbf{m_i})\right] = N(\mathbf{m_i}, \Sigma_i)$$

with $i = 1, \ldots, C$

- In this case, the discriminant function is written as

$$g_i(\mathbf{x}) = \ln p(\mathbf{x}\,|\,\omega_i) + \ln P(\omega_i)$$

$$\Downarrow$$

$$g_i(\mathbf{x}) = -\frac{1}{2}(\mathbf{x}-\mathbf{m_i})^t \Sigma_i^{-1}(\mathbf{x}-\mathbf{m_i}) - \frac{n}{2}\ln 2\pi - \frac{1}{2}\ln|\Sigma_i| + \ln P(\omega_i)$$

*If $g_1(x) = g_2(x)$ it means that the point is on the decision boundary and thus must be assigned randomly

1. $\sum = \sigma^2 I$ (aka isolevels take a spherical shape)
   With this we know:

$$\sum^{-1} = \frac{1}{\sigma^2} I \tag{107}$$

$$|\sum| = \sigma^{2n} \tag{108}$$

So simplifying the og equation:

$$g_i(x) = -\frac{1}{2}(x - m_i)^t \frac{I}{\sigma^2}(x - m_i) + ln(P(w_i)) \tag{109}$$

$$= \frac{-1}{2\sigma^2}(x^t x - x^t m_i - m_i^t x - m_i^t m_i) + ln(P(w_i)) \;\mid\; x^t m_i = m_i^t x \tag{110}$$

$$= \frac{m_i^t x}{\sigma^2} - \frac{m_i^t m_i}{2\sigma^2} + ln(P(w_i)) \;\mid\; x^t x \rightarrow ignored \tag{111}$$

$$= w_i^t x + bias_i \tag{112}$$

where $w_i = \frac{m_i}{\sigma^2}$ and $bias_i = \frac{-m_i^t m_i}{2\sigma^2} + ln(P(w_i))$

Now to get the equation to the decision boudary we calculate $g_i(x) = g_j(x)$ and

- In this case, they can be written as

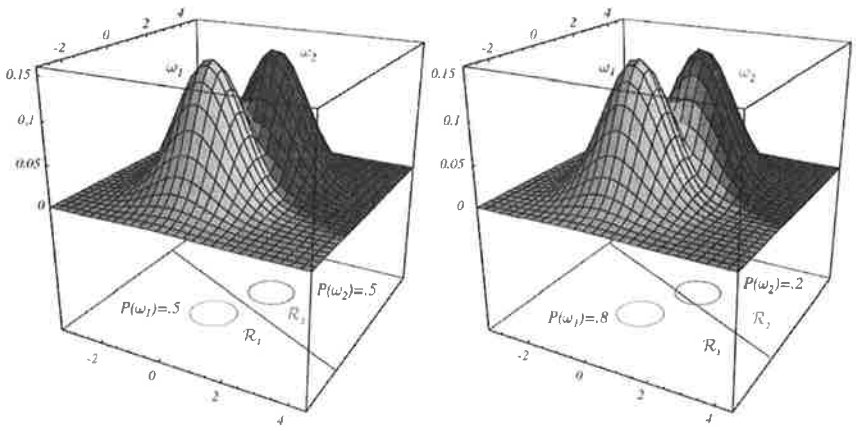
$$\mathbf{w}^t(\mathbf{x}-\mathbf{x}_0)=0$$

where

$$\begin{cases}\mathbf{w}=\mathbf{m}_i-\mathbf{m}_j \\ \mathbf{x}_0=\frac{1}{2}(\mathbf{m}_i+\mathbf{m}_j)-\frac{\sigma^2}{\|\mathbf{m}_i-\mathbf{m}_j\|^2}\ln\frac{P(\omega_i)}{P(\omega_j)}(\mathbf{m}_i-\mathbf{m}_j)\end{cases}$$

Where $\frac{1}{2}(m_i + m_j)$ is the midpoint between the means and the rest is a bias "correction term mostly dependant on the prior probabilities" that gives more area to the biggest prob.

Examples with 2-D distributions



2. $\sum_i = \sum$ (aka all distributions don't have a standard shape?)
Same demonstrations as before but this time

- Decision boundaries are

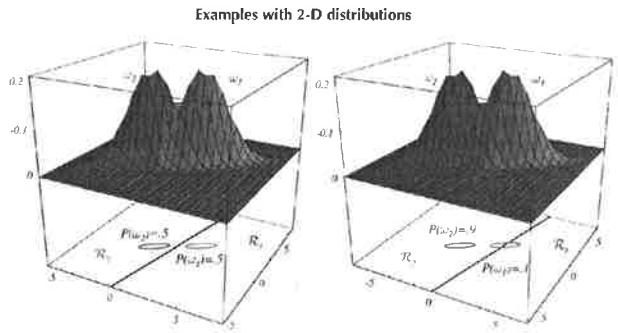
$$\mathbf{w}^t(\mathbf{x}-\mathbf{x}_0)=0$$

where

$$\begin{cases}\mathbf{w}=\Sigma^{-1}(\mathbf{m}_i-\mathbf{m}_j)\\ \mathbf{x}_0=\frac{1}{2}(\mathbf{m}_i+\mathbf{m}_j)-\frac{\ln[P(\omega_i)/P(\omega_j)](\mathbf{m}_i-\mathbf{m}_j)}{(\mathbf{m}_i-\mathbf{m}_j)^t\Sigma^{-1}(\mathbf{m}_i-\mathbf{m}_j)}\end{cases}$$

- Hyperplane passes through $\mathbf{x}_0$ but is not necessarily orthogonal to the line between the means.

w is now "rotated" by the covariance matrix, to take into account the shapes of the distributions. (Meaning the decision boundary line is no longer perpendicular to w, now it can be tilted)



3. $\sum_i$ is arbitrary

- Discriminant functions are

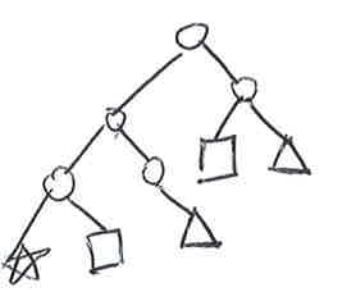
$$g_i(\mathbf{x})=\mathbf{x}^t\mathbf{W}_i\mathbf{x}+\mathbf{w}_i^t\mathbf{x}+w_{i0}$$ ☜ Quadratic discriminant

where

$$\begin{cases}\mathbf{W}_i=-\frac{1}{2}\Sigma_i^{-1}\\ \mathbf{w}_i=\Sigma_i^{-1}\mathbf{m}_i\\ w_{i0}=-\frac{1}{2}\mathbf{m}_i^t\Sigma_i^{-1}\mathbf{m}_i-\frac{1}{2}\ln|\Sigma_i|+\ln P(\omega_i)\end{cases}$$

- Decision boundaries are hyperquadrics.

Basically means the decision boudary no longer needs to be a line, it can take any shape



we prefer rules that making decision tree

looks like right

Decision trees

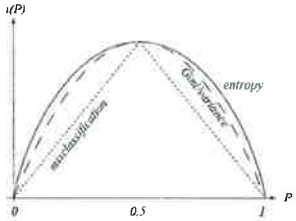
Used to classify non numerical data. We create a tree of questions that lead to the final class

⇐⇒ general framework ⇐⇒ {numbers of splits, test selection, impurity, stop, pruning}

- CART - Classification and regression trees
  Prefered to use a branching facto of 2, each time the data is split it's basically going into a smaller subset of data

  Impurity A pure partition is when a question splits the data into perfectly pure subsets, i.e. they only contain one type of class

  - Entropy impurity: $i(N) = -\sum_j P(\omega_j) log_2 \left(P(\omega_j)\right)$ (Fraction of patterns at node N belonging to ω_j)
  - Variance impurity: $i(N) = P(\omega_1)P(\omega_2)$
  - Gini impurity: $i(N) = \sum_{i \neq j} P(\omega_i)P(\omega_j) = 1 - \sum_j P^2(\omega_j)$
  - Misclassification impurity: $i(N) = 1 - \max P(\omega_j)$



We use one of these impurities to calculate the drop in impurity:

$$\Delta i(N) = i(N) - P_L i(N_L) - (1 - P_L) i(N_R) \tag{113}$$

L and R being left and right node. This has a problem that the splits are local, i.e. they don't guarantee that the next one will give good results

- Twoing Criterion What happens when I have more than two classes before a split??
  1. We create superclasses, in the case of 3 classes we create:

     $$C1 = [w_1] \;\; C2 = [w_2, w_3] C1 = [w_2] \;\; C2 = [w_1, w_3] C1 = [w_3] \;\; C2 = [w_1, w_2] \tag{114}$$

  2. Then calculate the impurity of each case, for each superclass
  3. pick the best one
- When to stop splitting? Splitting too much leads to overfitting to the training data Solutions:
  - Stopped splitting
    - Validation dataset, with accuracy measure aka cross-validation
    - Impurity threshold - meaning stop splitting if Δi goes lower than a specified value

- Complexity-accuracy tradeoff criterion

$$\alpha(tree\ size) + \sum_{j=1}^{N} i(j) \tag{115}$$

  Evaluate using the number of branches (ex) and the sum of the impurity of each end node at the current depth level. Bigger tree = overfit, unless there still is high impurity
- Hypothesis testing - Check if the new split is "statistically significant" i.e. is the imp. drop close to 0?

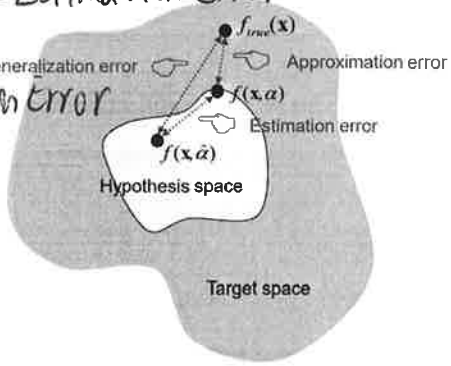
* Prunning
  Let the tree reach overfit / impurity = 0 and remove branches that have a low impurity drop by "merging" the classes into one / merging the decision areas into one
  It gets more computationaly demanding if the training set is big

- Generalization error

  When we train a linear model, there will be an error between the aproximated model and the ground truth. Possibly caused by the limited dataset size, but even with an infinite amount of data we won't get the perfect model, and this is caused because it's linear, and it can't fit the real model perfectly. Here the hypotheis is given by the linear limitation of the model, if the model would be non linear this area would increase.

$\mathcal{E}(f(x,\hat{a}), f(x,a))$ <= Estimation Error (may be solved)

$\mathcal{E}(f(x,a), f_{true}(x))$ <= Approximation Error (can't be solved)



Estimation error caused by data limitations and **Approximation error** is caused by the model linear limitation

Gen. error:

$$E\{R(\alpha|x,y)\} = \int_{(x,y)} R(\alpha|x,y)p(x,y)dxdy \tag{116}$$

"The error is the average (E) of the cost (R) for any pair of points (x, y) belonging to the input and output spaces"
Where alpha in the case of a linear model is $\alpha = [w, b]$ equal to the weight and the bias $y' = wx + b$. And R is the loss function

The problem is that $p(x, y)$ is the "true joint pdf" which we don't have. So we estimate using the **empirical risk** i.e. the error computed on the training samples:

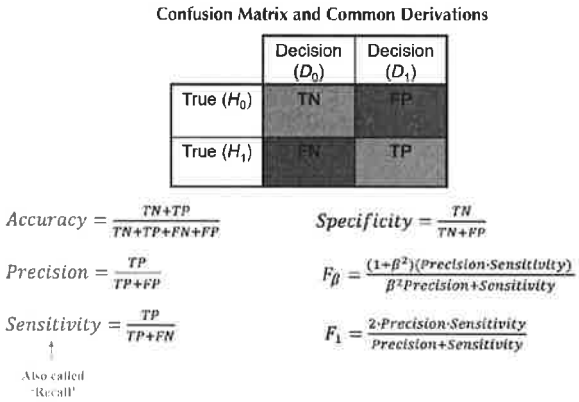
$$E\{R(\alpha|x,y)\} \approx \frac{1}{N}\sum_{i=1}^{N} L(y_i, f(x_i,\alpha)) \tag{117}$$

ROC (needed)

Accuracy Estimation

- Exhaustive cross-validation
  - Leave one out:
    Remove one sample from training data, and use it as test. Then start from scratch and pick another sample as test, and keep repeating
- Non-exhaustive
  - Hold out
    Split dataset in training and test (commonly by half)
  - K-fold
    Make K clusters and apply same logic of leave one out
  - Monte Carlo
    Repeat hold out various times
- Nested

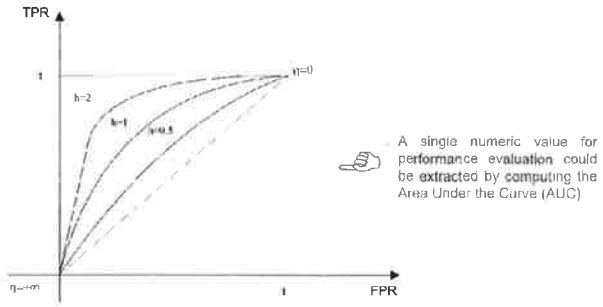
Confusion matrix



Where the accuracy is the "overall accuracy" and the specificity is the "accuracy"

ROC curve Receiver Operating Characteristics

TPR - true positive rate $\int_T^{+\infty} p(x|w_2)dx$
FPR - false positive rate $\int_T^{+\infty} p(x|w_1)dx$
If the classes are not overlapped the curve is squared from 00 to 10 to 11
If the classes overlap completely it's TPR = FPR
Partial overlap are the cases in between



Comparing Classifiers

If I have two classifiers and want to know which one works better, how can I do this? And if the difference is really small, can I tell if it's just noice? We consider H_0 as the hypotesis that there is a difference between classif. and the oppostie for H_1

- T Test

$$T = \frac{\frac{\Delta a}{k}}{\sqrt{\frac{Var(\Delta)}{k}}} \tag{118}$$

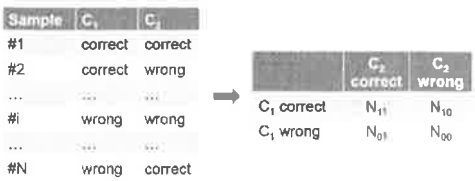
Where $\frac{\Delta a}{k}$ is the average difference between the accuracy of the two classifiers given k folds And the $\sqrt{\frac{Var(\Delta)}{k}}$ the avg of the std deviations??

Then if T is lower than a given Tcritical value the difference is not statistically significant

$$T \leq T_c \rightarrow reject\ H_0 \mid T > T_c \rightarrow accept\ H_0 \tag{119}$$

- McNemar's Test

Compare on each sample the two classifiers and checks if they agree and disagree in the same way using a contingency table:



x^2 distribution

It has it's own formula and a similar threshold to T test:

$$Z^2 = \frac{(N_10 - N_01)^2}{N_10 + N_01} \tag{120}$$