

1/19/2012

START MOTIVATING classification & K-means

$$\begin{aligned}
 P(x|\text{class 1}) &= N(x; \mu_1, \sigma^2 I) \\
 P(x|\text{class 2}) &= N(x; \mu_2, \sigma^2 I)
 \end{aligned}
 \left. \vphantom{\begin{aligned} P(x|\text{class 1}) \\ P(x|\text{class 2}) \end{aligned}} \right\} \begin{array}{l} \text{circularly symmetric} \\ \text{same } \sigma \text{ value} \\ \text{different class} \\ \text{centres (means)} \end{array}$$

for a given x , we want to "classify" it by choosing the class conditional distribution with largest probability
 (because $P(\text{class 1}|x) > P(\text{class 2}|x) \Rightarrow P(x|\text{class 1}) > P(x|\text{class 2})$ if $P(\text{class 1}) = P(\text{class 2})$)
 choose class 1 if $P(x|\text{class 1}) > P(x|\text{class 2})$

$$\Rightarrow \frac{P(x|\text{class 1})}{P(x|\text{class 2})} > 1 \quad \text{"likelihood ratio test"}$$

$$e^{-\frac{1}{2} \frac{(x-\mu_1)^T (x-\mu_1)}{\sigma^2}} > e^{-\frac{1}{2} \frac{(x-\mu_2)^T (x-\mu_2)}{\sigma^2}}$$

True log of both sides

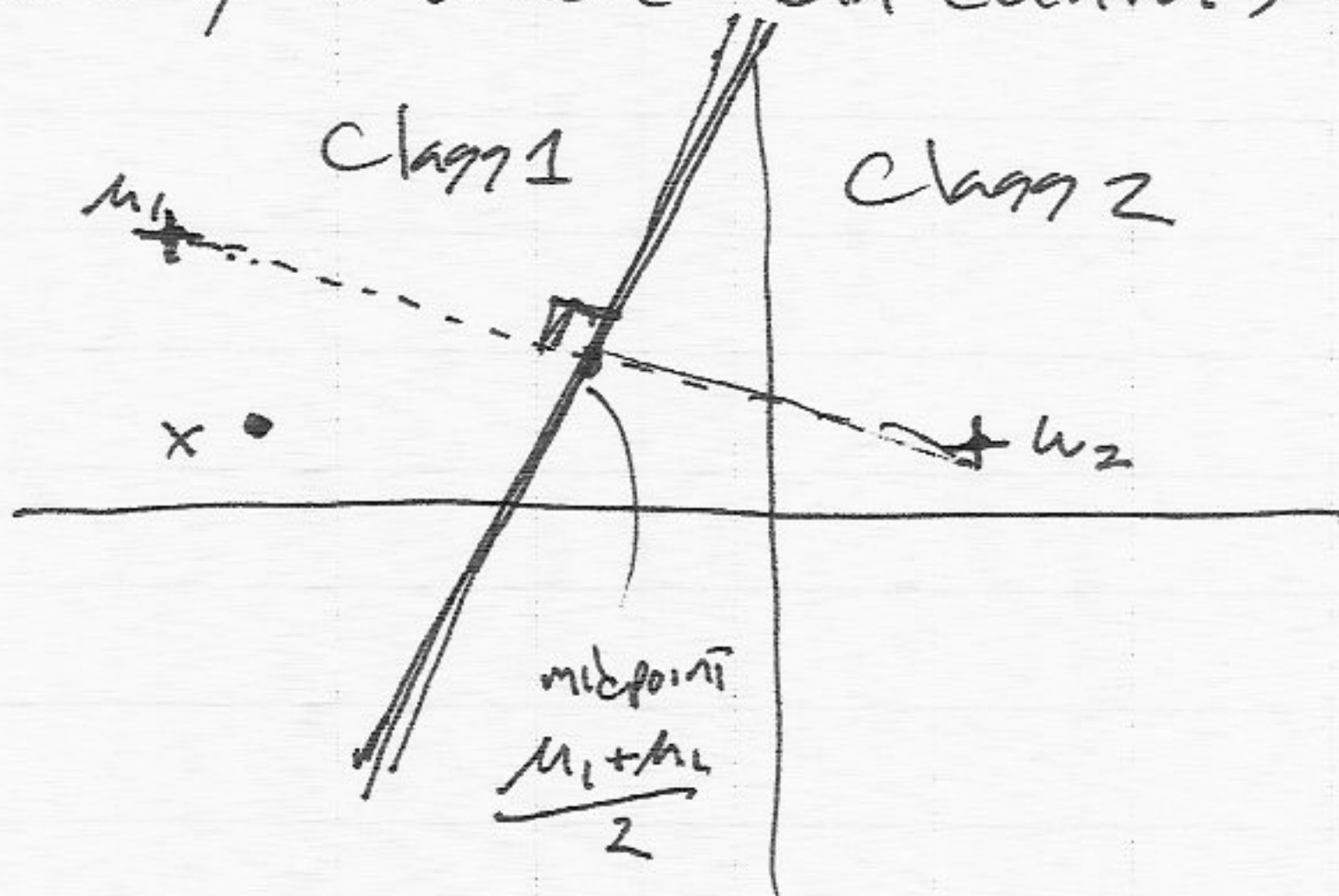
$$\frac{1}{2\sigma^2} (x-\mu_1)^T (x-\mu_1) > \frac{1}{2\sigma^2} (x-\mu_2)^T (x-\mu_2)$$

$$\|x-\mu_1\|^2 > \|x-\mu_2\|^2$$

~~True log of both sides~~

$$\Rightarrow \|x-\mu_1\| > \|x-\mu_2\|$$

so classify according to which mean (center) x is closest to!



Maximum likelihood estimation given i.i.d. samples $X = \{x_1, x_2, \dots, x_n\}$

$$L(X | \mu, \sigma^2) = \prod_{i=1}^n \frac{1}{(2\pi)^{k/2} |\sigma^2 I|^{1/2}} e^{-\frac{1}{2} (x_i - \mu)^T [\sigma^2 I]^{-1} (x_i - \mu)}$$

notation change. let $P = 1/\sigma^2 = \text{"Precision"}$

$$L(X | \mu, P) = \prod_{i=1}^n \frac{1}{(2\pi)^{k/2}} P^{1/2} e^{-\frac{1}{2} P (x_i - \mu)^T (x_i - \mu)}$$

as a function of X (~~that is~~ ^{for fixed params} μ & P), this is a joint probability, and integrates to 1

However, as a function of μ and P (for fixed samples X) it does not integrate to 1. This is why we call it a "likelihood" function.

Principle of maximum likelihood is that we would like to estimate value of μ and P ~~to~~ ^{that} maximize this likelihood. To do this, we take derivative of L ~~with respect to~~ ^{with respect to} the unknowns μ and P , ~~and set it to 0~~ ^{then}: set those equations to zero, and solve.

~~The~~ ^{The} math gets simpler if we work with the log of L .

$$\log L = \sum_{i=1}^n \log \frac{1}{(2\pi)^{k/2}} + \frac{1}{2} \log P - \frac{1}{2} P (x_i - \mu)^T (x_i - \mu)$$

$$= \text{const} + \frac{n}{2} \log P - \frac{1}{2} P \sum_{i=1}^n (x_i - \mu)^T (x_i - \mu)$$

$$\frac{\partial \log L}{\partial \mu} = 0$$

$$\frac{\partial \log L}{\partial P} = 0$$

Aside: matrix/vector derivatives.

We now want to take a derivative of a scalar with respect to a vector.

$$\frac{\partial f(u)}{\partial u} \begin{matrix} \text{scalar} \\ \text{column vector} \end{matrix} = \begin{bmatrix} \frac{\partial f(u)}{\partial u_1} \\ \frac{\partial f(u)}{\partial u_2} \\ \vdots \\ \frac{\partial f(u)}{\partial u_n} \end{bmatrix} \begin{matrix} \text{column vector} \\ \text{row vector} \end{matrix}$$

$$\frac{\partial f(u)}{\partial u^T} = \left[\frac{\partial f(u)}{\partial u_1} \quad \frac{\partial f(u)}{\partial u_2} \quad \dots \quad \frac{\partial f(u)}{\partial u_n} \right] \begin{matrix} \text{row vector} \end{matrix}$$

what about w.r.t a matrix?

$$\frac{\partial f(u)}{\partial u} \begin{matrix} \text{scalar} \\ \text{matrix} \end{matrix} = \begin{bmatrix} \frac{\partial f(u)}{\partial u_{11}} & \frac{\partial f(u)}{\partial u_{12}} & \dots & \frac{\partial f(u)}{\partial u_{1n}} \\ \frac{\partial f(u)}{\partial u_{21}} & & & \\ \vdots & & & \\ \frac{\partial f(u)}{\partial u_{ni}} & & & \end{bmatrix}$$

NOTE, some authors use $\left[\frac{\partial f(u)}{\partial u_{ji}} \right]$ which is the transpose of what we are computing

we will need:

$$f(u) = u^T x = u_1 x_1 + u_2 x_2 + \dots + u_n x_n$$

$$\frac{\partial f(u)}{\partial u} = \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{bmatrix} = x \quad \text{so} \quad \frac{\partial u^T x}{\partial u} = x$$

$$f(u) = u^T u = u_1^2 + u_2^2 + \dots + u_n^2$$

$$\frac{\partial f(u)}{\partial u} = \begin{bmatrix} 2u_1 \\ 2u_2 \\ \vdots \\ 2u_n \end{bmatrix} = 2u \quad \text{so} \quad \frac{\partial u^T u}{\partial u} = 2u$$

BACK TO MLE ESTIMATION

$$\frac{\partial \mathcal{L}(X|\mu, \rho)}{\partial \mu} = \frac{\partial}{\partial \mu} \text{const} + \frac{n}{2} \log \rho - \frac{1}{2} \rho \sum_{i=1}^n (x_i - \mu)^T (x_i - \mu)$$

$$= -\frac{1}{2} \rho \sum_{i=1}^n \frac{\partial}{\partial \mu} (x_i - \mu)^T (x_i - \mu)$$

$$\frac{\partial}{\partial \mu} x_i^T x_i - 2\mu^T x_i + \mu^T \mu$$

$$= -2x_i + 2\mu \quad [\text{from previous page}]$$

$$= -\frac{1}{2} \rho \sum_{i=1}^n (2\mu - 2x_i) = 0$$

$$\Rightarrow \rho \sum_{i=1}^n (x_i - \mu) = 0$$

$\rho = \frac{1}{\sigma^2}$ cannot be 0, therefore

~~Note that ρ dropped out here, so as long as~~

$$\sum_{i=1}^n (x_i - \mu) = 0$$

$$\Rightarrow \sum x_i = \sum \mu$$

$$\Rightarrow \sum x_i = n\mu$$

$$\Rightarrow \mu = \frac{\sum x_i}{n}$$

"centroid"
"mean" - why it is called k-means.

also, note that ρ dropped out, so as long as σ^2 is fixed and the same for both classes, we don't care what it is.

note, we aren't interested in changing/estimating $P = \frac{1}{\sigma^2}$ in this example. BUT if we were...

$$\frac{\partial \log L}{\partial p} = \frac{n}{2} \frac{1}{p} - \frac{1}{2} \sum_{i=1}^n (x_i - \mu)^T (x_i - \mu) = 0$$

Just a scalar

$$\frac{1}{p} = \sigma^2 = \frac{\sum_{i=1}^n (x_i - \mu)^T (x_i - \mu)}{n}$$

= sample variance

If we wanted to estimate full covariance matrix Σ from i.i.d. samples, we could go about it in same way (using M.L.E.), but the derivation is much harder (especially tricky because you need to constrain Σ to be symmetric, see Tom Minka's tutorial on Matrix Algebra useful for statistics & websites)

^{MLE} Estimator of full Gaussian parameters in K dimensions \rightarrow

$$\text{mean } \mu = \sum_{i=1}^n x_i / n$$

$$\text{covariance } \Sigma = \frac{\sum_{i=1}^n (x_i - \mu)(x_i - \mu)^T}{n}$$

note difference between this and what we derived above. This is a whole matrix.

This brings us to the k-means algorithm for clustering.

Given: vector samples x_1, x_2, \dots, x_n .

Desired: k cluster centers $\mu_1, \mu_2, \dots, \mu_k$

Algorithm

Initialize k cluster centers

Repeat

- assign each sample to the nearest cluster center
- Recompute cluster centers

UNTIL assignments do not change

Analysis

This algorithm is attempting to optimize the following objective function

$$\text{minimize SSE} = \sum_{i=1}^k \sum_{x_j \in \text{class } k} \|x_j - \mu_i\|^2$$

(provided we consistently break ties if there are equidistant centers. One strategy is to choose cluster with smallest index number)

Good news: It always converges. Each step/iteration of the algorithm improves the objective function (i.e. ~~does not~~ does not increase the SSE)

Bad news: It is only guaranteed to converge to a local optimum - ~~it~~ ^{may} NOT find global optimum

Simple proof of converge. SSE decreases at each iteration. There are only a finite set of assignments. Therefore the algorithm must eventually terminate (it can't keep decreasing forever).

Recall the objective function that k-means is optimizing

$$\text{minimize SSE} = \sum_{k=1}^K \sum_{x_n \in \text{class } k} \|x_n - \mu_k\|^2$$

Introduce a set of label variables Z_{nk} . There will be $N \times K$ of them. Define:

$$Z_{nk} = \begin{cases} 1 & \text{if sample point } n \text{ is from class } k \\ 0 & \text{otherwise} \end{cases}$$

This is a so-called "1 of K representation".

For a given sample point n , there are K variables Z_{nk} , and only one of them is 1 (the rest are zero).

Example

<p>class 1 + x_1 + x_3</p> <p>class 2 + x_2</p> <p>class 3 +</p>	<p>$Z_{nk} \rightarrow$</p> <p>N samples x_1</p> <p>x_2</p> <p>x_3</p>	<table border="1" style="border: none;"> <thead> <tr> <th></th> <th colspan="3" style="text-align: center;">K classes</th> </tr> <tr> <th></th> <th style="text-align: center;">1</th> <th style="text-align: center;">2</th> <th style="text-align: center;">3</th> </tr> </thead> <tbody> <tr> <td style="text-align: center;">1</td> <td style="text-align: center;">1</td> <td style="text-align: center;">0</td> <td style="text-align: center;">0</td> </tr> <tr> <td style="text-align: center;">2</td> <td style="text-align: center;">0</td> <td style="text-align: center;">1</td> <td style="text-align: center;">0</td> </tr> <tr> <td style="text-align: center;">3</td> <td style="text-align: center;">1</td> <td style="text-align: center;">0</td> <td style="text-align: center;">0</td> </tr> </tbody> </table>		K classes				1	2	3	1	1	0	0	2	0	1	0	3	1	0	0
	K classes																					
	1	2	3																			
1	1	0	0																			
2	0	1	0																			
3	1	0	0																			

with this notation we can rewrite the k-means objective function as

$$\text{minimize SSE} = \sum_{k=1}^K \sum_{n=1}^N Z_{nk} \|x_n - \mu_k\|^2$$

Note the simpler double summation, which removes the conditional $\sum_{x_n \in \text{class } k}$. The Z_{nk} representation still picks out the samples that belong to class k , but does so in a cleaner way.