Lecture 8 - 2/22

Soft Clustering

Soft Clustering

every row has a single assignment

So far, clustering was done using **hard assignments** (1 point -> 1 cluster)

Sometimes this doesn't accurately represent the data: it seems reasonable to have overlapping clusters.

In this case, we can use **soft assignment** to assign points to every cluster with a certain probability.

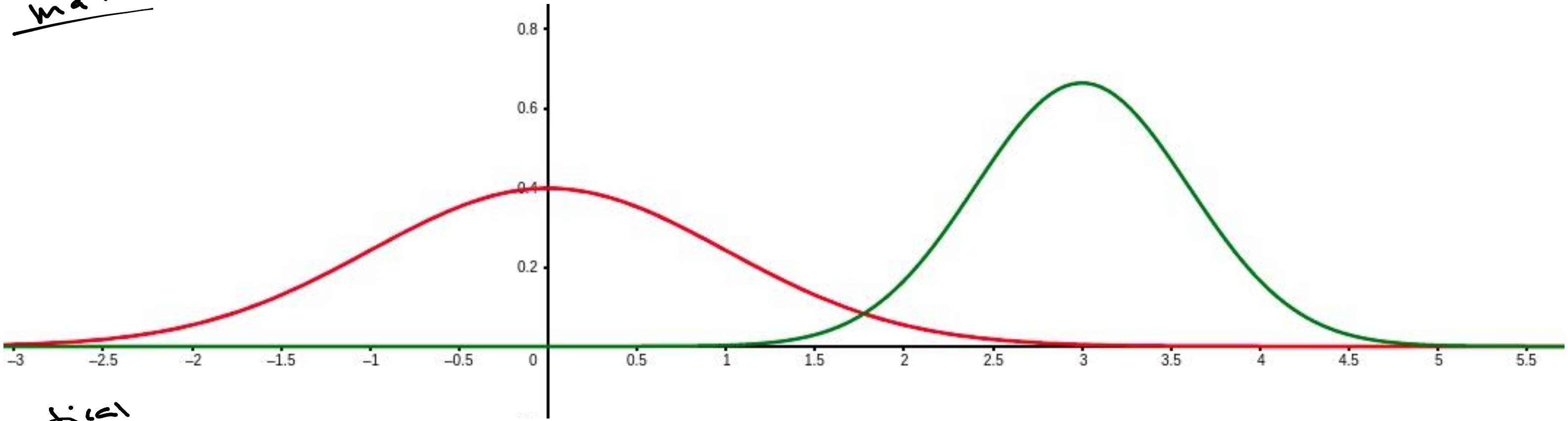
– the model we will use

→ two normal distributions

Soft Clustering - Example

Generate data using $\mathbf{N(\mu_1, \sigma_1)}$ and $\mathbf{N(\mu_2, \sigma_2)}$

math



practical

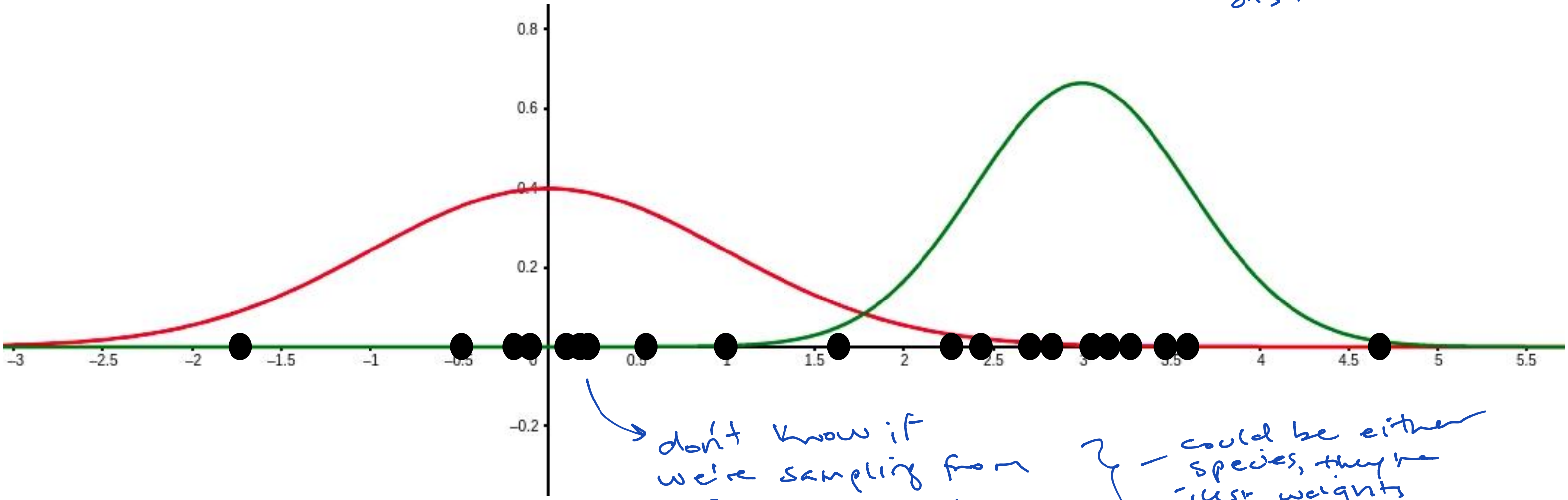
Or: we are given the weights of animals. Unknown to us these are weights from two different species. Can we determine the species (group / assignment) from the height?

↳ which animal is more likely?

Soft Clustering - Example

Generate data using $\mathbf{N(\mu_1, \sigma_1)}$ and $\mathbf{N(\mu_2, \sigma_2)}$

– once you know whether you're picking red, or green
↳ Follow that distribution



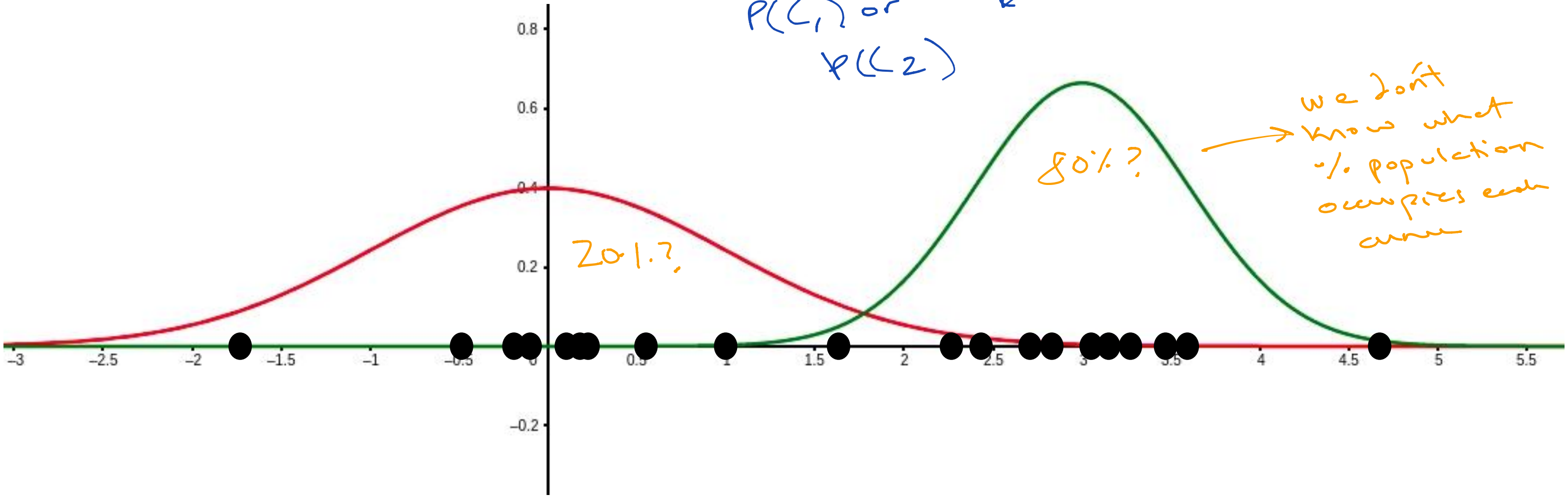
don't know if we're sampling from green or red curve – could be either species, they're just weights

Soft Clustering - Example

— d — this many times to generate dataset (but don't know upfront the prob. of picking from each curve)

Generate data using **N(μ_1, σ_1)** and **N(μ_2, σ_2)**

$P(C_1)$ or $P(C_2)$



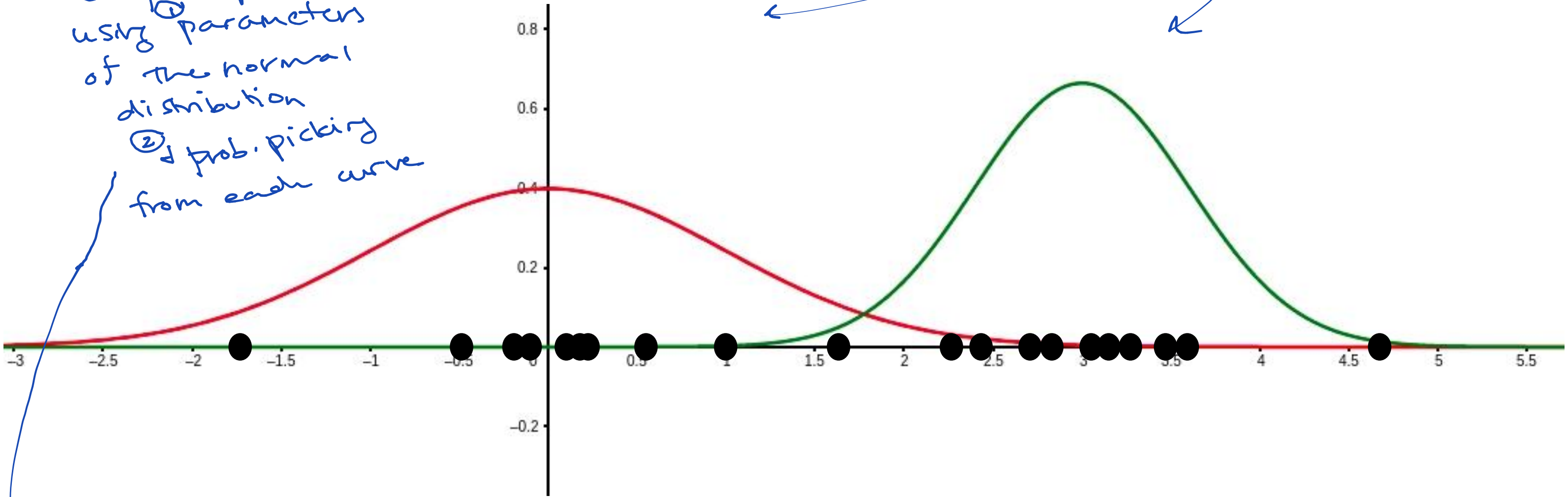
Any of these points could technically have been generated from either curve.

Soft Clustering - Example

Generate data using **N(μ$_1$, σ$_1$)** and **N(μ$_2$, σ$_2$)**

within the probability of each curve (calculate) we have this distribution of weights/points

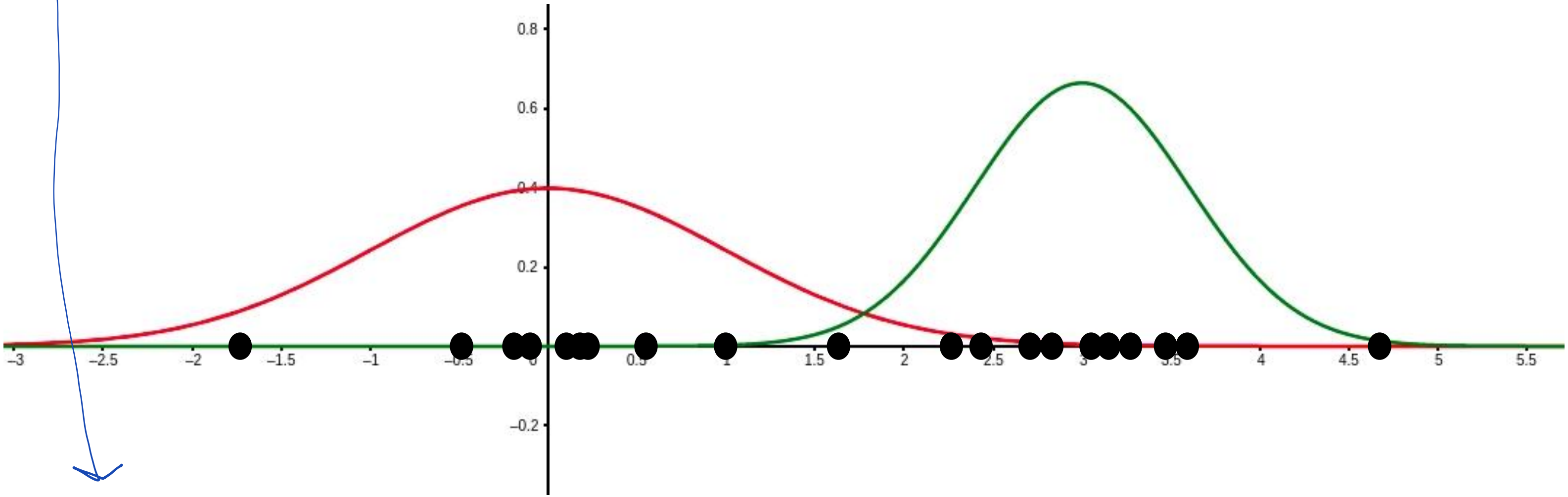
- compute prob. ① using parameters of the normal distribution ② prob. picking from each curve



For each point we can compute the probability of it being generated from either curve

Soft Clustering - Example

Generate data using $\mathbf{N(\mu_1, \sigma_1)}$ and $\mathbf{N(\mu_2, \sigma_2)}$



We can create soft assignments based on these probabilities.

Mixture Model

X comes from a mixture model with k mixture components if the probability distribution of X is:

#distributions/components/clusters

$$P(X=x) = \sum_{j=1}^{k} \underbrace{P(C_j)}_{} \overbrace{P(X=x|C_j)}^{}$$

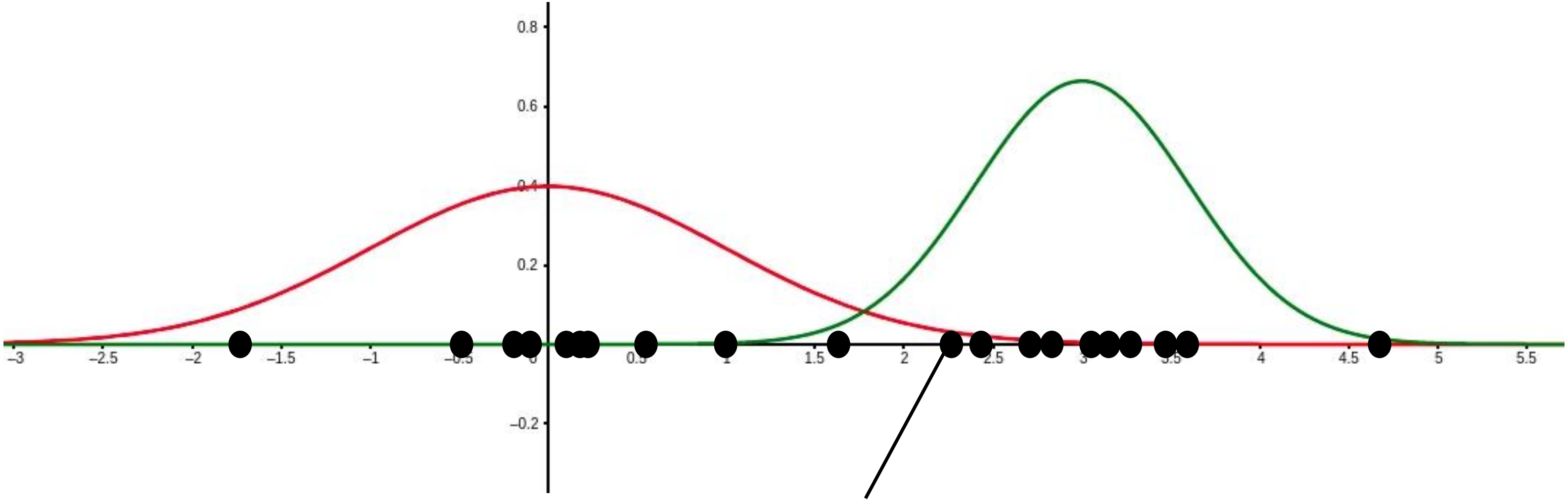
once you know you're in that curve

Mixture proportion
Represents the probability of belonging to C_j (a curve)

Probability of seeing x when sampling from C_j

weighted distribution

Example



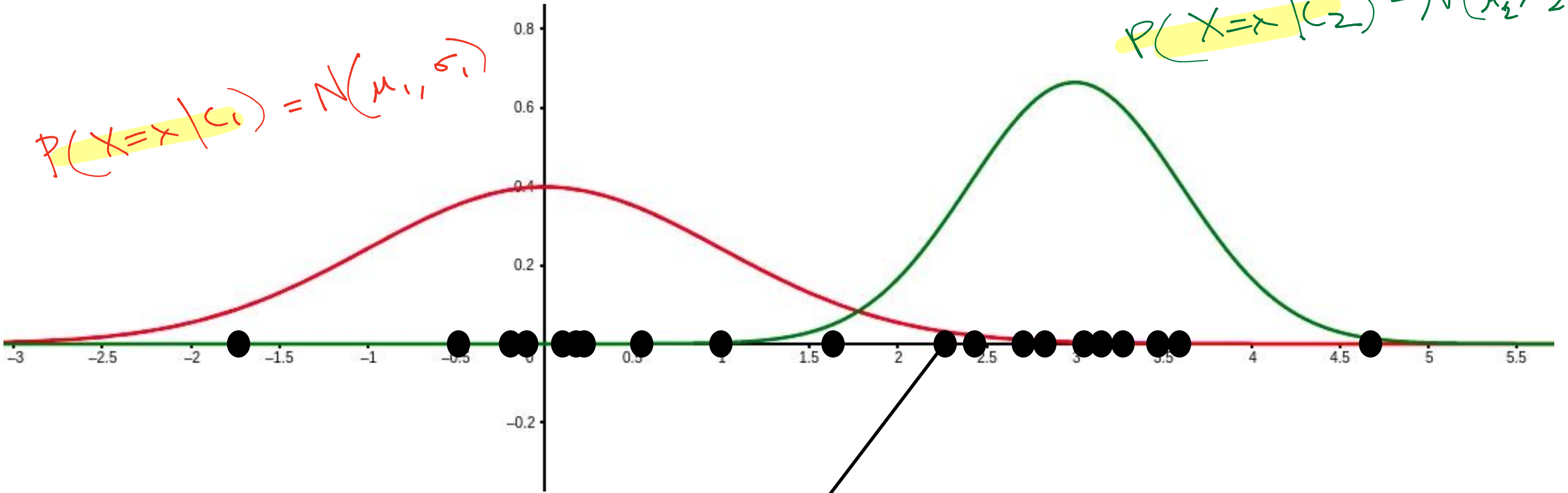
What is the probability distribution here?

Example

clearly a "mixture distribution" – so use the mixture model

$P(X=x|C_1) = N(\mu_1, \sigma_1)$

$P(X=x|C_2) = N(\mu_2, \sigma_2)$



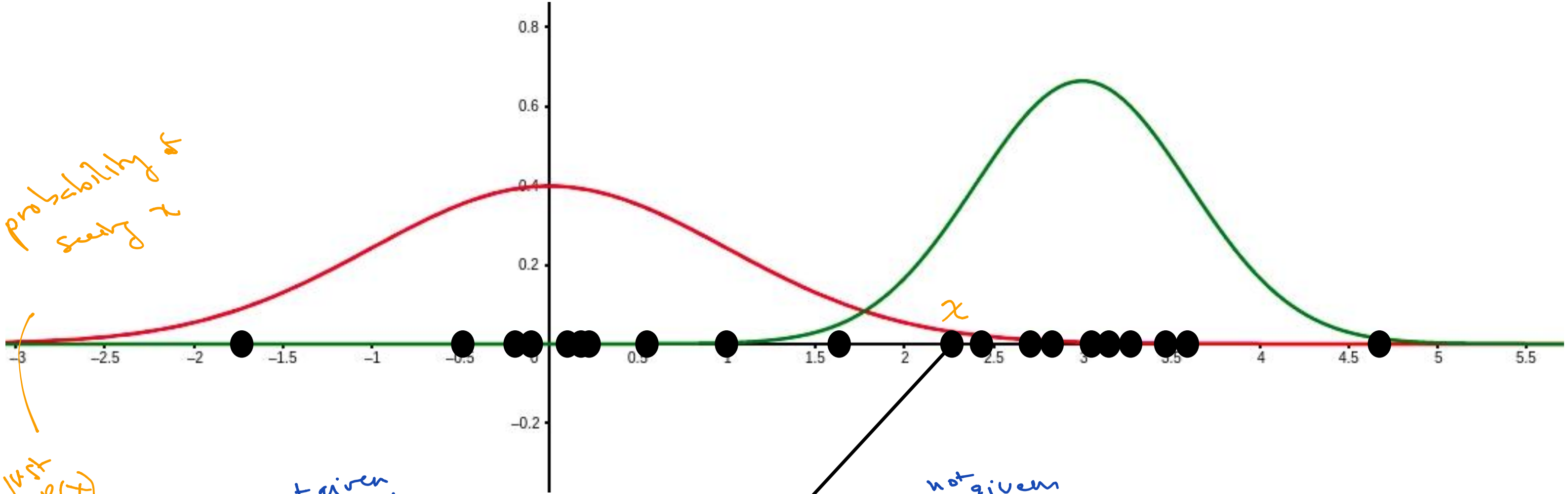
$$P(X=x) = P(C_1)P(X=x|C_1) + P(C_2)P(X=x|C_2)$$

prob. of C_1 given you're in C_1
* prob. of being in C_1 ($P(C_1)$)

prob. of C_2 given you're in C_2
* prob. of being in C_2 ($P(C_2)$)

Example

probability of seeing x



just P(x)

not given upfront

not given

$$\underbrace{P(X=x)}_{\text{any possible x that X can take on}} = \overbrace{P(C_1)}^{\text{not given upfront}}\frac{1}{\sigma_1\sqrt{2\pi}}e^{-\frac{1}{2}\left(\frac{x-\mu_1}{\sigma_1}\right)^2} + \overbrace{P(C_2)}^{\text{not given}}\frac{1}{\sigma_2\sqrt{2\pi}}e^{-\frac{1}{2}\left(\frac{x-\mu_2}{\sigma_2}\right)^2}$$

any possible x that X can take on

Gaussian Mixture Model

A Gaussian Mixture Model (GMM) is a mixture model where

$$P(X = x|C_i) \sim N(\mu, \sigma)$$

the probability within the cluster...

(once you know P(Cn))

follows a normal distribution

(parameterized by μ and σ)

GMM Clustering

goal in all of ML

Goal: Find the GMM that maximizes the probability of seeing the data we have.

Finding the GMM means finding the parameters that uniquely characterize it. What are these parameters?

ex) what are parameters of normal distr. that max. the prob. of seeing the given data?

P(C$_i$) & **μ_i** & **σ_i** for all **k** components.

what parameters uniquely characterize the data?

GMM Clustering

each has this distribution

n data points from GMM, sampled independently

(size of dataset)

Goal: maximization function

Find the argument θ that maximizes this

contains all parameters

$$\theta^* = \arg\max_{\theta} \prod_{i=1}^{n} \sum_{j=1}^{k} P(C_j) P(X_i \mid C_j)$$

product over all the datapoint of all their individual probabilities of the GMM

Joint probability distribution of our data

Assuming our data are independent

$\Theta = \{\mu_1 \ldots \mu_k, \sigma_1 \ldots \sigma_k, P(C_1) \ldots P(C_k)\}$

Θ^* is the arg max of Θ

GMM Clustering

from calculus → take derivative w/ respect to product
set = 0, solve
↳ BUT very complicated for a product

How do we find the critical points of this function?
↳ value of θ that is a local max/min

Notice: taking the log-transform does not change the critical points = trick :)
↳ transform into a space where the problem is easier to solve

Define:

☆ transform the space

$$l(\theta) = \log(L(\theta))$$

$$= \sum_{i=1}^{n} \log(\sum_{j=1}^{k} P(C_j)P(X_i \mid C_j))$$

↳ the product becomes a sum (much simpler)

GMM Clustering

For $\boldsymbol{\mu} = [\boldsymbol{\mu}_1, \ldots, \boldsymbol{\mu}_k]^T$ and $\boldsymbol{\Sigma} = [\boldsymbol{\Sigma}_1, \ldots, \boldsymbol{\Sigma}_k]^T$

We can solve

$$\frac{d}{d\Sigma} l(\theta) = 0 \qquad \frac{d}{d\mu} l(\theta) = 0$$

find best value for Σ

find best value for μ

As described by prev. slides

GMM Clustering

To get

The Logic

1) have an optimization function
trying to maximize

2) difficult to take max as is,
so LOG TRANSFORM

3) take derivative w/ respect to each parameter, set = 0, solve
get CRITICAL POINTS

→ the solutions

$$\hat{\mu}_j = \frac{\sum_{i=1}^{n} P(C_j|X_i) X_i}{\sum_{i=1}^{n} P(C_j|X_i)}$$

$$\hat{\Sigma}_j = \frac{\sum_{i=1}^{n} P(C_j|X_i)(X_i - \hat{\mu}_j)^T (X_i - \hat{\mu}_j)}{\sum_{i=1}^{n} P(C_j|X_i)}$$

how do you get this?

$$\hat{P}(C_j) = \frac{1}{n} \sum_{i=1}^{n} P(C_j|X_i)$$

^ represents estimated value

GMM Clustering

Do we have everything we need to solve this?

Still need $\mathbf{P(C_j \mid X_i)}$ (i.e. the probability that X_i was drawn from C_j)

GMM Clustering

Baye's Rule

$$P(C_j|X_i) = \frac{P(X_i|C_j)}{P(X_i)}P(C_j)$$

$$= \frac{P(X_i|C_j)P(C_j)}{\sum_{j=1}^{k} P(C_j)P(X_i|C_j)}$$

but... you need $P(C_j|X_i)$ to get this? → see eqns on prev. slide

why we need an algorithm to do this

can't just solve an equation... need an algorithm

Looks like a loop! Seems we need $P(C_j)$ to get $P(C_j \mid X_i)$ and $P(C_j \mid X_i)$ to get $P(C_j)$

(stuck)

Expectation Maximization Algorithm

1. Start with random $\boldsymbol{\theta}$
2. Compute $\mathbf{P(C_j \mid X_i)}$ for all $\mathbf{X_i}$ by using $\boldsymbol{\theta}$
3. Compute / Update $\boldsymbol{\theta}$ from $\mathbf{P(C_j \mid X_i)}$
4. Repeat 2 & 3 until convergence

can compute using random values (what we were trying to solve for before)

* values help each other by updating throughout the algorithm

why would we converge necessarily? (ask in OH) – not as simple as Lloyd's algorithm

Demo

wrote some code
that uses
GMM

(see the repo)

→ Clustering Aggregation