EM algorithm

LAB

2020-10-23

LAB 2

- Deadline: November, 4
- To: adv.statistics.2020@gmail.com.
- Subject: LAB 2
- Report file name: LAB2_LastName1_LastName2

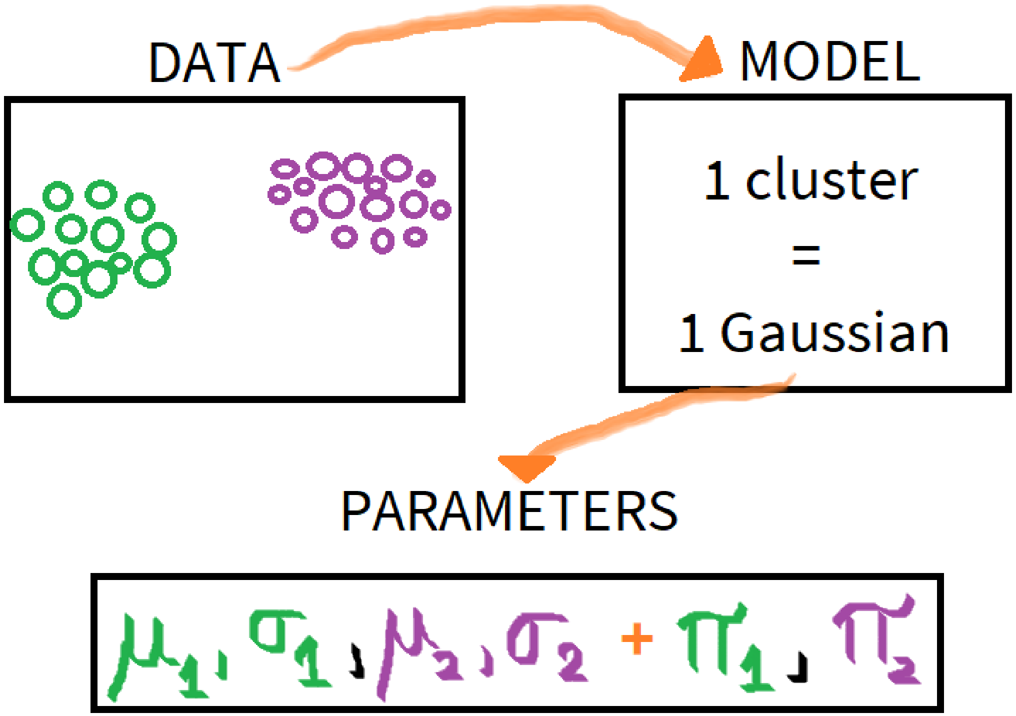
For this lab:

- You'll solve the 4 exercises on these slides

- (As usual) You should submit a report based on these exercises

Guidelines: We expect a self-contained report with answers, figures and results. You can use RMarkdown, it is not mandatory. Additionally, you will send the code (.R) or the file that builds the report (.Rmd), we only want to look at it if there is a mistake on the report. If you are not confortable writing equations in LaTeX you can add a hand-written appendix with the exercises that need a derivation with formulas.

Model-based clustering - GMM



We need to estimate these parameters -> EM

EM algorithm

It's an statistical algorithm to estimate parameters.

Particular example: GMM parameter estimation

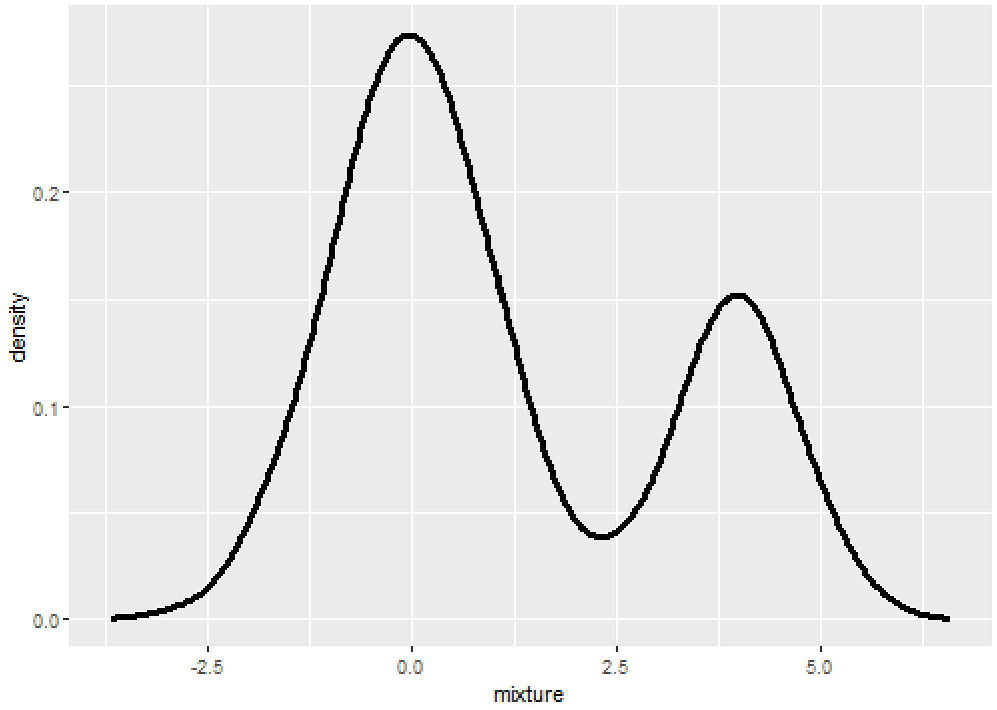
Characteristics:

- It is iterative
- Random initialization
- Consists of two steps (E-step, M-step)
- E-step: compute probabilities of each point to belong to each distribution
- M-step: Re-compute the parameters based on the new probabilities

GMM: mixture of Gaussians

math: $f(x) = \sum_{k=1}^{K} \pi_k f_k(x),\ f_k \sim \mathcal{N}(\mu_k, \sigma_k^2)$

```
distr <- sample(c(0, 1), 10000, c(0.3, 0.7), replace = TRUE)
mixture <- distr*rnorm(10000)+(1-distr)*rnorm(10000, 4, 0.75)
ggplot() +
  geom_line(aes(x=mixture), color="black", size=1.2, stat="density")
```



Which parameters do we need to estimate?

K normal distributions:

- μ_k: mean (position)
- σ_k^2: variance (dispersion)
- the proportion π_k for each distribution

All parameters: $\theta = (\pi_k, \mu_k, \sigma_k^2)_{k=1}^K$

We will also compute all p_{ik} and we will keep them in a matrix P

Algorithm

Random initialize θ_0

while (not convergence)

E-step: Compute p_{ik}

$$p_{ik}^{(t+1)} = P(Z = k | X_i = x_i) = \frac{\pi_k^{(t)} f_{k,t}(x_i)}{\sum_{l=1}^{K} \pi_l^{(t)} f_{l,t}(x_i)}$$

M-step: Estimate parameters

$$\hat{\pi}_k^{(t+1)} = \frac{1}{n} \sum_{i=1}^{n} p_{ik}^{(t+1)}$$

$$\hat{\mu}_k^{(t+1)} = \frac{1}{n\hat{\pi}_k^{(t+1)}} \sum_{i=1}^{n} x_i p_{ik}^{(t+1)}$$

$$\hat{\sigma}_k^{(t+1)} = \sqrt{\frac{1}{n\hat{\pi}_k^{(t+1)}} \sum_{i=1}^{n} p_{ik}^{(t+1)} (x_i - \hat{\mu}_k^{(t+1)})^2}$$

EX 1: Code your own EM algorithm.

You can test it with these data

```
distr <- sample(c(0, 1), 1000, c(0.3, 0.7), replace = TRUE)
mixture <- distr*rnorm(1000)+(1-distr)*rnorm(1000, 4, 0.75)
mixture_matrix <- matrix(mixture)
```

You should obtain something similar to:

```
result = EM(mixture, 2, 100)
result$means
```

```
## [1]  3.98810447 -0.06418016
```

```
result$sigmas
```

```
## [1] 0.7223961 0.9640072
```

```
result$prop
```

```
## [1] 0.3143361 0.6856639
```

Applications

- CLUSTERING:

To divide in groups your data

assign one observation to a distribution

distribution = cluster

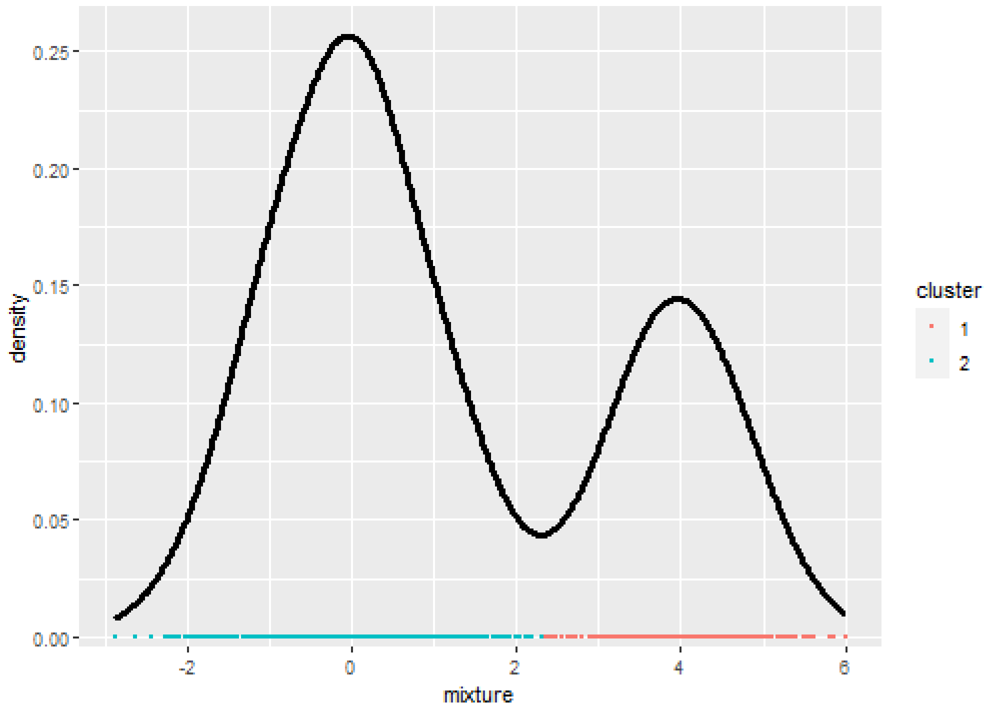
Matrix of probabilities: $(P)_{ij} = p_{ij}$

Assign observation i^* to the distribution with biggest p_{i^*j}

- NEW DATA GENERATION

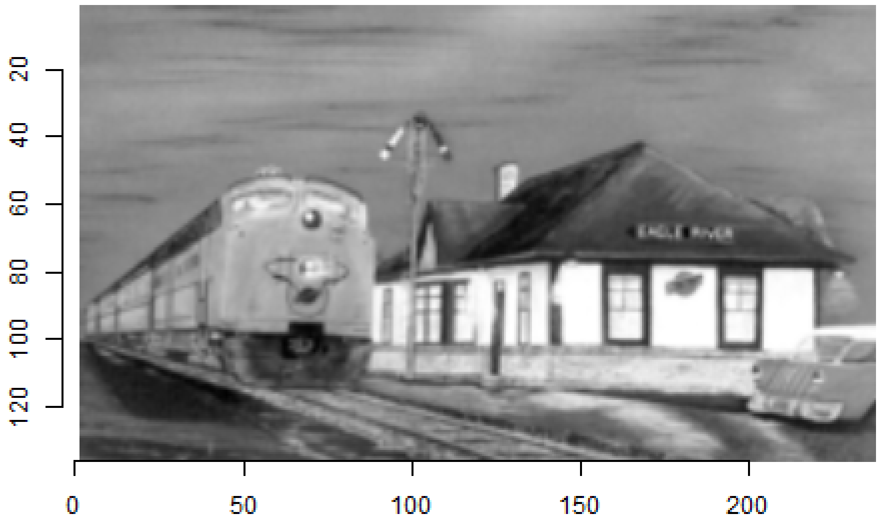
Concrete simple example

```
cluster = as.factor(apply(result$prob, 1, which.max))
ggplot() +
  geom_line(aes(x=mixture), color="black", size=1.2, stat="density")-
  geom_point(aes(x = mixture, color = cluster), y = 0, size = 0.6)
```



EX 2: Use your own EM algorithm to segment a grayscale picture

```
library(imager)
train = load.image(here::here("fig","train.jpg"))
train = resize_halfXY(resize_halfXY(train)) # resize
plot(train)
```



Hint: as.data.frame() and as.cimg( ) will be useful

What happens if we want to cluster an RGB image?

When the data $x_i \in \mathbb{R}^p$ -> Multivariate EM

Parameters:

- cluster proportion π -> cluster proportion π
- mean μ -> mean vector μ
- variance σ^2 -> covariance matrix Σ

Expected log-likelihood:

$$\sum_{i=1}^{n} \sum_{k=1}^{K} p_{ij} [log(\pi_k) - \frac{p}{2} log(2\pi) - \frac{1}{2} log|\Sigma_k| - \frac{(x_i - \mu_k)^T \Sigma_k^{-1} (x_i - \mu_k)}{2}]$$

EX 3: Multivariate EM

Derive the μ and Σ estimators when the data is multidimensional

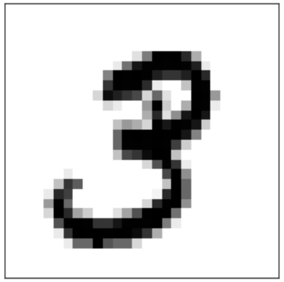
Tips:

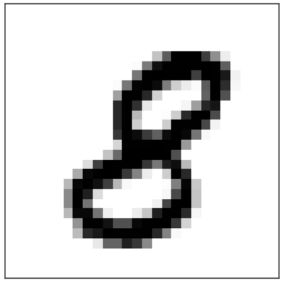
- Use the following properties of the trace:
  - tr(a)=a (a scalar)
  - tr(ABC) = tr(CAB)
- Check out the matrix cookbook to use matrix and vector derivatives

You don't have to code it

EX 4: www.thispersondoesnotexist.com

Use the EM algorithm to generate your own thisdigitdoesnotexist.com based on the MNIST dataset





Pipeline:

- Load dataset: https://pjreddie.com/media/files/mnist_train.csv

(It can take long to load, first column is the label of the digit)

- keep only some samples (5000 for example)
- apply PCA (to reduce the num dimensions, 30 for example)
- fit a GMM with the EM algorithm (EMcluster::emcluster, e.g. K=10)
- sample new data
- go back to original dimension and print the new images
- BONUS (not graded): create a simple site that shows the fake digits