14 Unsupervised Learning

- Input data available $D = \{x_n\}$, but target values are not available. UNSUPERVISED LEARNING is LEARNING WITHOUT A TEACHER, there is no supervisor.

In unsupervised learning I can CHARACTERIZE THE STRUCTURE OF THE INPUT SPACE and find properties of our input data that help to understand what will be the output.

Note: SEMI-SUPERVISED LEARNING: you have target values just for a part of the input.

GOAL: understand where the data comes from. We can assume a parametric model.

If we assume that those data comes from Gaussian distribution we can easily estimate the parameters (mean and covariance).

GAUSSIAN MIXTURE MODEL

Mixed combination of **K** Gaussian distributions. We have K Gaussian distribution and we sum all of them each multiplied by a factor that is the importance-weight.

$$P(x) = \sum_{k=1}^{K} \underbrace{\pi_k \mathcal{N}(x; \mu_k, \Sigma_k)}_{\text{Avg weighted}}$$

Each instance x generated by:

1. Choosing one of the K Gaussians with uniform probability
2. Generating an instance at random according to that Gaussian.

Let's see a very simple algorithm called Kmeans.

$\boxed{\text{GOAL of K-MEANS}}$: computing the means of the Gaussians distribution.

INPUT: $D = \{x_n\}$, value K   OUTPUT: $\mu_1, \dots, \mu_K$

This is the task that is also called CLUSTERING, we want generate K groups of samples and portion the dataset in K partition and in each partition we have similar points.

ITERATIVE ALG. with two step.

1. take the first K training element as SINGLE ELEMENT CLUSTERS (randomly chosen centr.)

2. Assign each of the remaining N−K training samples with THE CLOSEST CENTROID. AFTER EACH ASSIGNMENT, recompute the centroid of the new cluster.

We can repeat the second step until we get a specific convergence situation $\longrightarrow$ THE CENTROID DOES NOT CHANGE.
IT IS GUARANTED THAT THE ALGORITHM CONVERGES.

The termination condition will eventually occur. Unfortunately this method has several drawback.

(−) TELL K in advance, you may be not sure which is the correct number of clusters (maybe you cannot visualize your data)

(−) THE ALGORITHM IS based on Distance, in complex dataset the units in the different dimensions can be difficult to compare.

There are some solutions. K-MEANS does not consider the covariance.

Let's remodel a little bit the GMM by introducing another set of variable $\vec{z} = (z_1, \dots, z_K)^T$

$$z_k = \begin{cases} 1 & \text{sample } x \text{ generate by Gaussian } k. \\ 0 & \text{otherwise} \end{cases}$$

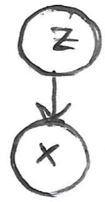
We have a 1-out-of K encoding (only one component is 1 all the others are 0).

$$P(z_k = 1) = \pi_k \qquad P(\vec{z}) = \prod_{k=1}^{K} \pi_k^{z_k}$$

For a given value of $\vec{z}$:

$$P(\vec{x} \mid z_k = 1) = \mathcal{N}(\vec{x}; \mu_k, \Sigma_k)$$

We want to model the fact that z affect x, we can draw:



Thus

$$P(\vec{x} \mid \vec{z}) = \prod_{k=1}^{K} \mathcal{N}(\vec{x}; \mu_k, \Sigma_k)$$

CHAIN RULE: $\boxed{P(\vec{x}, \vec{z}) = P(\vec{x} \mid \vec{z}) \, p(\vec{z})}$

Now:

$$\boxed{P(\vec{x}) = \sum_{z} P(\vec{z}) P(\vec{x} \mid \vec{z}) = \sum_{k=1}^{K} \pi_k \mathcal{N}(\vec{x}; \mu_k, \Sigma_k)}$$

GMM distribution $P(\vec{x})$ can be seen as the marginalization of a distribution $P(\vec{x}, \vec{z})$ over variables $\vec{z}$. This is important because it puts EVIDENCE on the precence of some variables that affect our distribution but they are <u>not observable</u>.

Z: LATENT VARIABLES $\Rightarrow$ not observable, but affect the distribution.

This a general approach

We have to define the posterior probability:

$$\gamma(z_k) = P(z_k = 1 \mid \vec{x}) = \frac{P(z_k = 1) P(\vec{x} \mid z_k = 1)}{P(\vec{x})}$$

$$= \frac{\pi_k \mathcal{N}(\vec{x}; \mu_k, \Sigma_k)}{\sum_{J=1}^{K} \pi_J \mathcal{N}(\vec{x}; \mu_J, \Sigma_J)}$$

π_k: prior probability of z_k, so some gaussian generate the data, before we get any data

γ_k: posterior probability, this data has been generated, after observing the data.

Given a dataset and GMM we want to estimate μ_k, Σ_k and π_k (generalization of k-means). We solve this problem by considering the maximum likelihood:

$$\boxed{\underset{\vec{\pi}, \vec{\mu}, \Sigma}{\operatorname{argmax}} \ln P(X \mid \vec{\pi}, \vec{\mu}, \Sigma)}$$

This problem is not simple and we need to use an iterative method, based on this observation:

When we reach a local maximum, derivative of the log-likelihood is zero you have these 3 eq:

γ depends on $\mu, \Sigma, \pi \ldots$ we cannot compute nothing here

This is not the sol. ↓ but gives the solution algorithm) intuition.

$$\begin{cases} \mu_k = \dfrac{1}{N_k} \displaystyle\sum_{n=1}^{N} \gamma(z_{nk}) \vec{x}_n \\ \Sigma_k = \dfrac{1}{N_k} \displaystyle\sum_{n=1}^{N} \gamma(z_{nk}) (\vec{x}_n - \vec{\mu}_k)(\vec{x}_n - \vec{\mu}_k)^T \\ \pi_k = \dfrac{N_k}{N}, \quad \text{with } N_k = \displaystyle\sum_{n=1}^{N} \gamma(z_{nk}) \end{cases}$$

IF WE HAVE γ WE CAN COMPUTE THE PARAMETERS of the model and VICEVERSA

We do an iterative process:

- start with an initialization of parameters $\pi_k^{(0)}, \mu_k^{(0)}, \Sigma_k^{(0)}$
- Repeat until termination condition $t = 0, \ldots, T$
- E STEP: given $\pi_k, \mu_k, \Sigma_k \longrightarrow$ compute $\gamma(z_{nk})$

• M STEP: given $\gamma(z_{nk}) \longmapsto$ compute π_k, μ_k, Σ_k 5

THIS IS CALLED EXPECTATION MAXIMIZATION ALGORITHM

- Converges to local maximum likelihood
- Provides estimates of the latent variables z_{nk}
- Extended version of k-means
- Can be generalized to other distributions

The initialization is the most critical part!

EXAM QUESTION: difference between EM and k-means.

General EM Problem:

- Observed data $\vec{X} = \{x_1, \dots, x_N\}$
- Unobserved latent variables $\vec{Z} = \{z_1, \dots, z_N\}$
- Parametrized prob. distribution $P(\vec{Y}|\vec{\theta})$
  - → $\vec{Y} = \{y_1, \dots, y_N\}$ where $y_n = x_n \cup z_n$
  - → $\vec{\theta}$ are the parameters.

DETERMINE: θ^* that (locally) maximizes $E[\ln P(\vec{Y}|\vec{\theta})]$

Unsupervised learning is very useful, since we have a lot of data, but what missing is LABELED data. Unsupervised learning can be also useful for supervised learning, HELPING IN UNDERSTANDING what are the information that you are processing. HAVING AN UNSUPERVISED PHASE IMPROVES A LOT!

NOTE: In general when I have an unsupervised dataset I cannot compute accuracy, because I don't have ground truth.

K-means no good performances with images! we are using distance function in the space of pixels of images.

Note! training a network only with images and no labels; to do so we can allow the input and the output to have the same dimension (same size)

We put the images both at input and output, and you can train the network. (an example was binary encoding function). IN THE INTERMIADIATE LAYER I LEARN HOW TO ENCODE THE INPUT. This kind of networks are called AUTO ENCODER NETWORK.