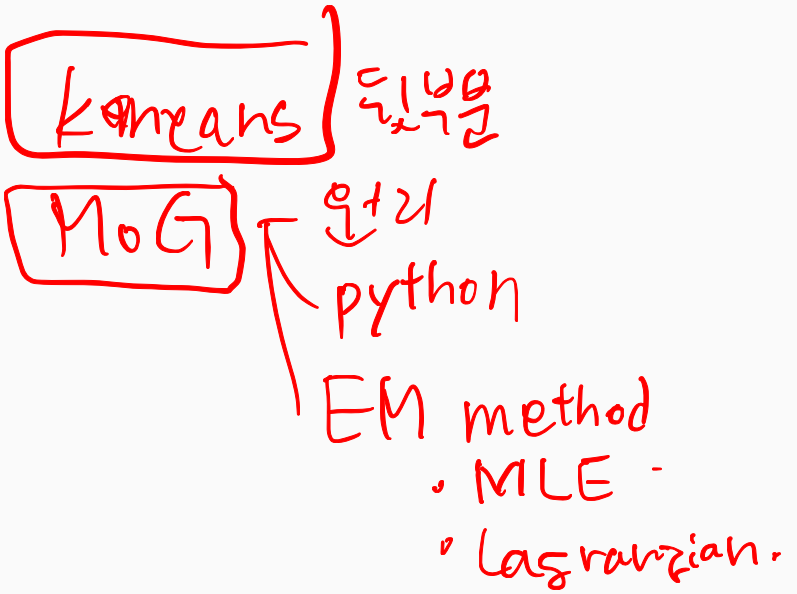
Lecture 07: Clustering II

[SCS4049-02] Machine Learning and Data Science

Seongsik Park (s.park@dgu.edu)

AI Department, Dongguk University



K-means algorithm: Advanced

K-means: centroid initialization method

If you happen to know approximately where the centroids should be (e.g., if you ran another clustering algorithm earlier), then you can set the initial hyperparameter containing the list of centroids.

Another solution is to run the algorithm multiple times with different random initializations and keep the best solution.

But, how exactly does it know which solution is the best? It uses a performance metric, called *inertia*, which is the mean squared distance between each instance and its closest centroid.

K-means: mini-batch k-means

Instead of using the full dataset at each iteration, the algorithm is capable of using mini-batches, moving the centroids just slightly at each iteration. This speeds up the algorithm typically by a factor of three or four and makes it possible to cluster huge datasets that do not fit in memory.

Although the mini-batch k-means algorithm is much faster than the regular k-means algorithm, its inertia is generally slightly worse, especially as the number of clusters increases.

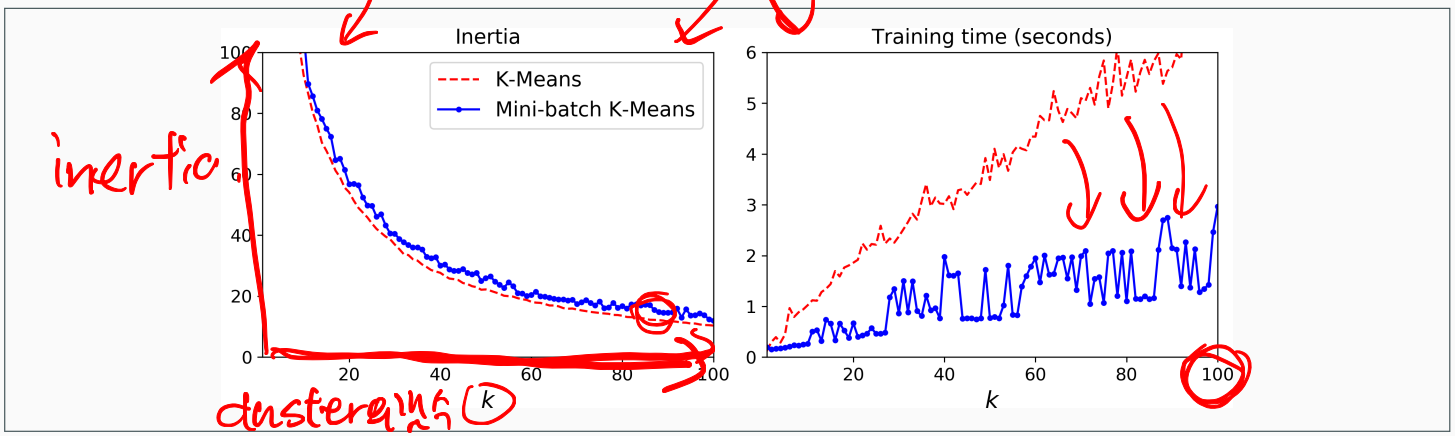


Figure 9-6. Mini-batch K-Means vs K-Means: worse inertia as k increases (left) but much faster (right)

K-means: finding the optimal number of clusters

Elbow rule: any lower value would be dramatic, while any higher value would not help much.

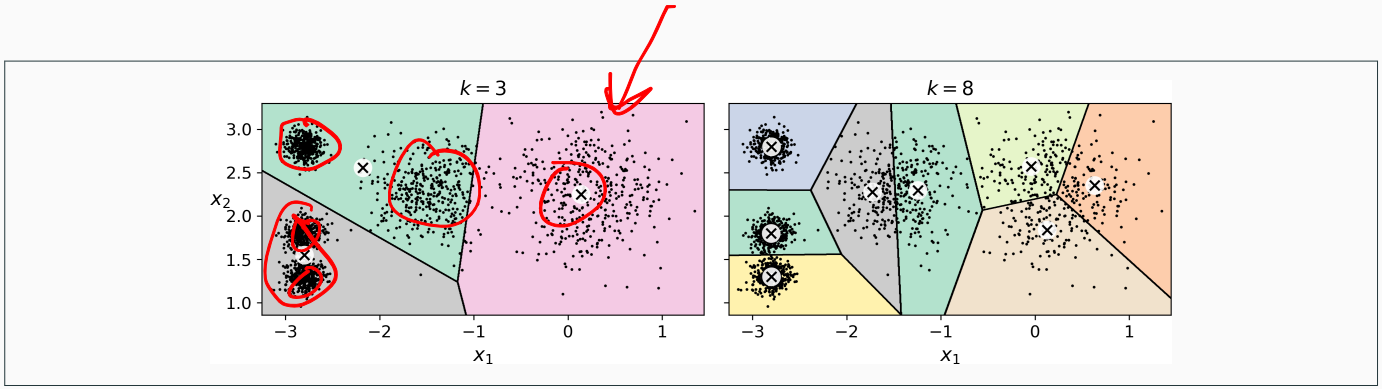


Figure 9-7. Bad choices for the number of clusters

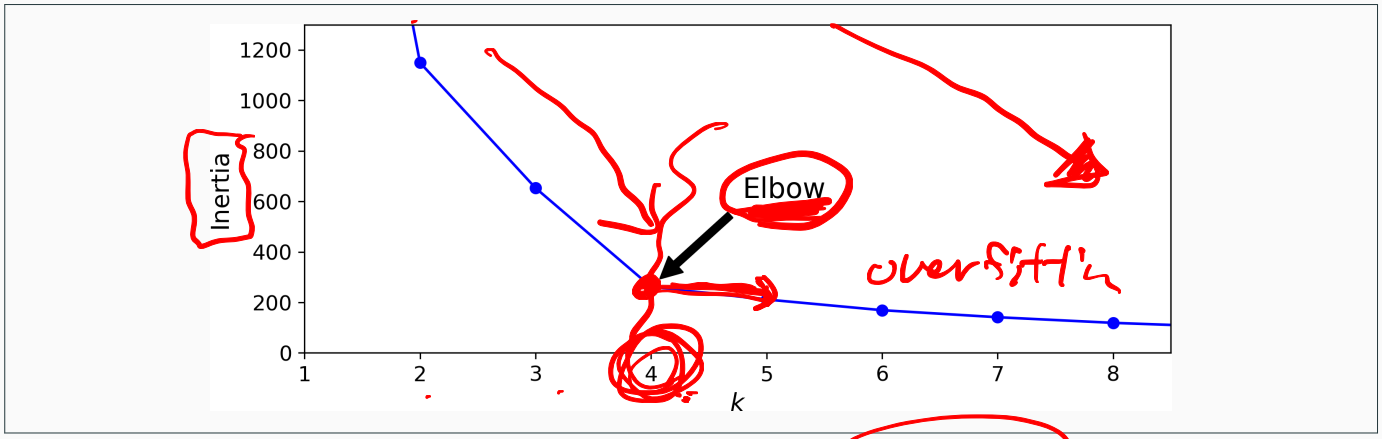
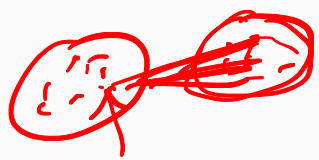


Figure 9-8. Selecting the number of clusters k using the "elbow rule"

K-means: finding the optimal number of clusters

A more precise approach (but also more computationally expensive) is to use the *silhouette score*, which is the mean *silhouette coefficient* over all the instances.

- An instance's silhouette coefficient is equal to $(b - a)/\max(a, b)$
- where a is the mean distance to the other instances in the same cluster (i.e., the mean intra-cluster distance)
- and b is the mean nearest-cluster distance (i.e., the mean distance to the instances of the next closest cluster, defined as the one that minimizes b, excluding the instance's own cluster).



The silhouette coefficient can vary between -1 and +1.

- A coefficient close to +1 means that the instance is well inside its own cluster and far from other clusters,
- while a coefficient close to 0 means that it is close to a cluster boundary,
- and finally a coefficient close to -1 means that the instance may have been assigned to the wrong cluster.

K-means: finding the optimal number of clusters

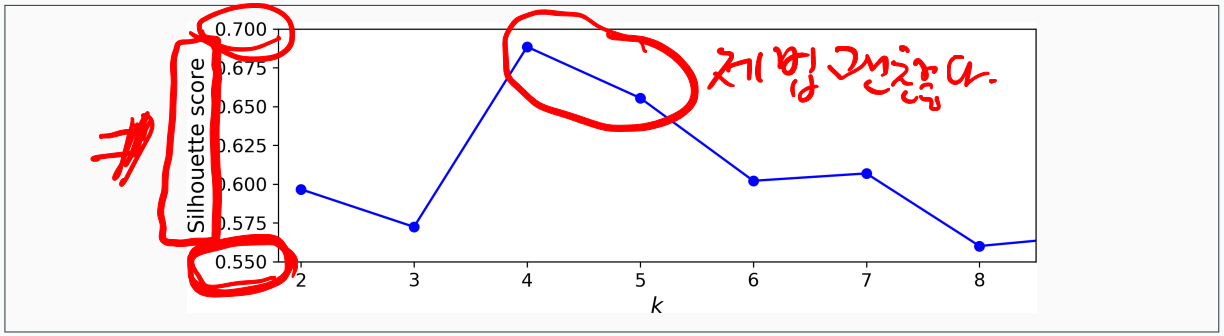


Figure 9-9. Selecting the number of clusters k using the silhouette score coefficient. This is called a *silhouette diagram* (see Figure 9-10):

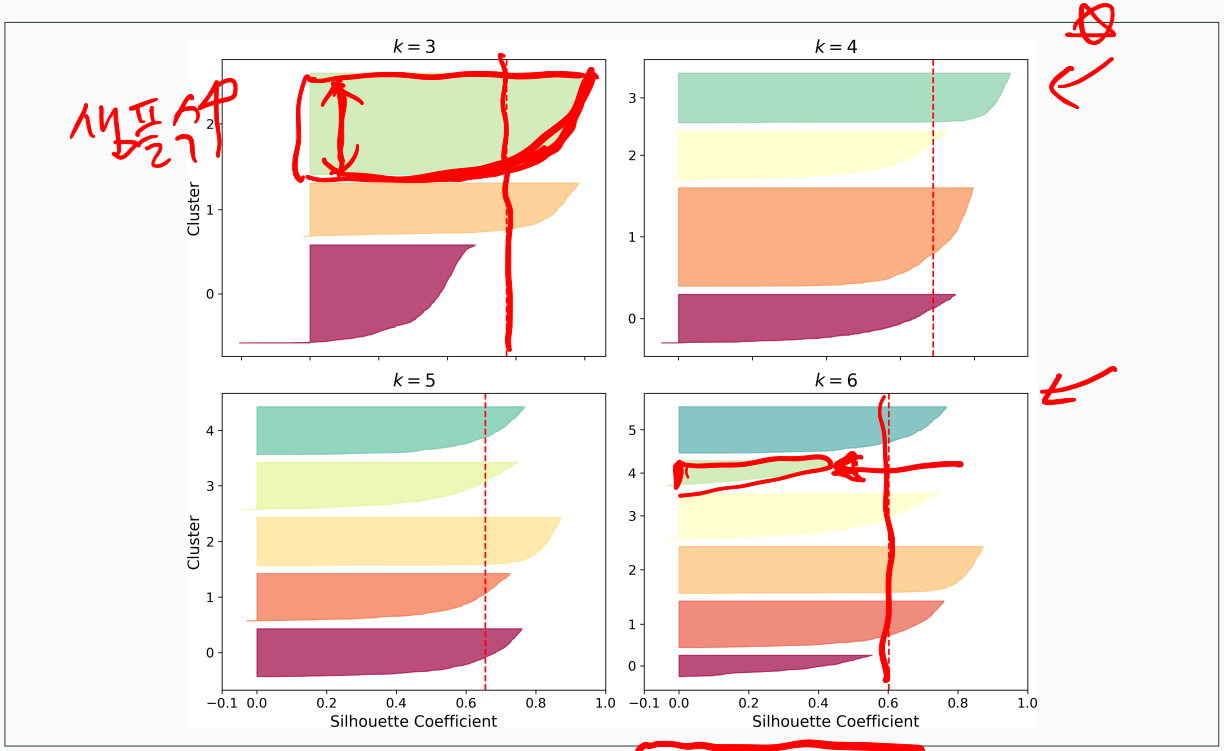


Figure 9-10. Silouhette analysis: comparing the silhouette diagrams for various values of k

K-means: limitations

- It is necessary to run the algorithm several times to avoid suboptimal solution
- You need to specify the number of clusters
- K-means does not behave very well when the clusters have varying sizes, different densities, or nonspherical shapes

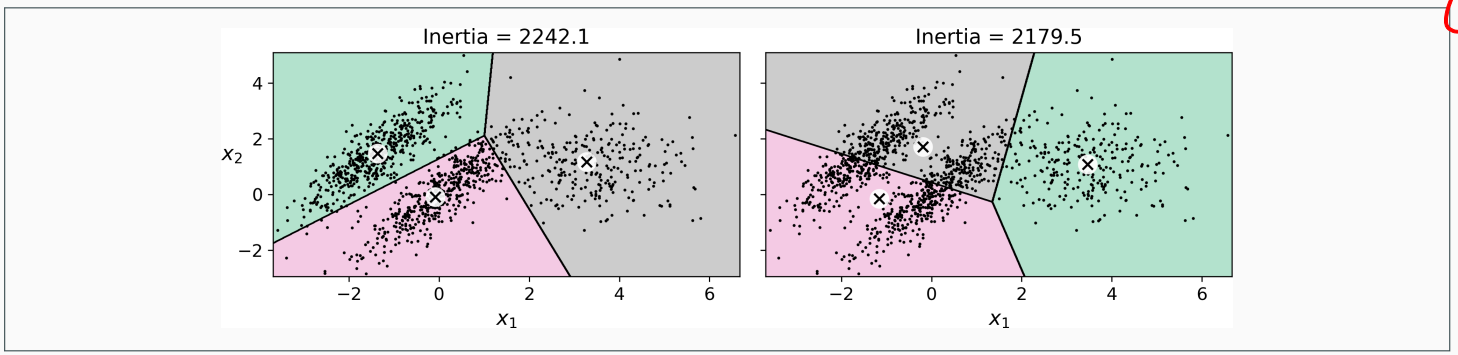


Figure 9-11. K-Means fails to cluster these ellipsoidal blobs properly

K-means, MoG.

Mixtures of Gaussians

MoG: 결론부터

1. **Initialize** the means $\boldsymbol{\mu}_k$, covariances $\boldsymbol{\Sigma}_k$ and mixing coefficients π_k.
2. **E-step** Evaluate the responsibilities using the current parameter values

$$\gamma(z_{nk}) = \frac{\pi_k \mathcal{N}(\mathbf{x}_n|\boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k)}{\sum_{j=1}^{K} \pi_j \mathcal{N}(\mathbf{x}_n|\boldsymbol{\mu}_j, \boldsymbol{\Sigma}_j)} \tag{1}$$

3. **M-step** Re-estimate the parameters using the current responsibilities

$$\boldsymbol{\mu}_k = \frac{1}{N_k} \sum_{n=1}^{N} \gamma(z_{nk}) \mathbf{x}_n \tag{2}$$

$$\boldsymbol{\Sigma}_k = \frac{1}{N_k} \sum_{n=1}^{N} \gamma(z_{nk}) (\mathbf{x}_n - \boldsymbol{\mu}_k)(\mathbf{x}_n - \boldsymbol{\mu}_k)^T \tag{3}$$

$$\pi_k = \frac{N_k}{N} \tag{4}$$

4. Evaluate the log likelihood

$$\log p(\mathbf{X}|\boldsymbol{\mu}, \boldsymbol{\Sigma}, \boldsymbol{\pi}) = \sum_{n=1}^{N} \log \left\{ \sum_{k=1}^{K} \pi_k \mathcal{N}(\mathbf{x}_n|\boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k) \right\} \tag{5}$$

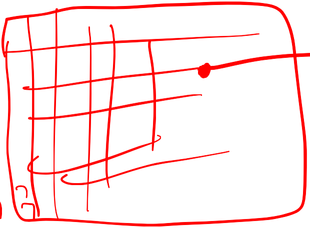
and check for convergence of either the parameters or the log likelihood. If the convergence criterion is not satisfied, return to step 2.

100번째

||

놓을 어떻게 표현해

= vector



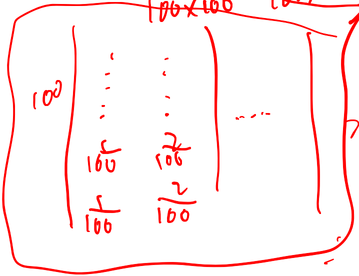
100번째

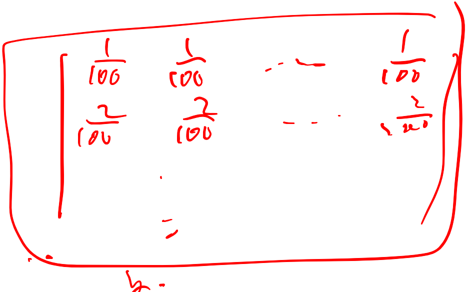
$[\frac{i}{100} \quad \frac{j}{100}]$ = 행렬

⇒ x를 어떻게 표현하지 = 벡터

= meshgrid(,)

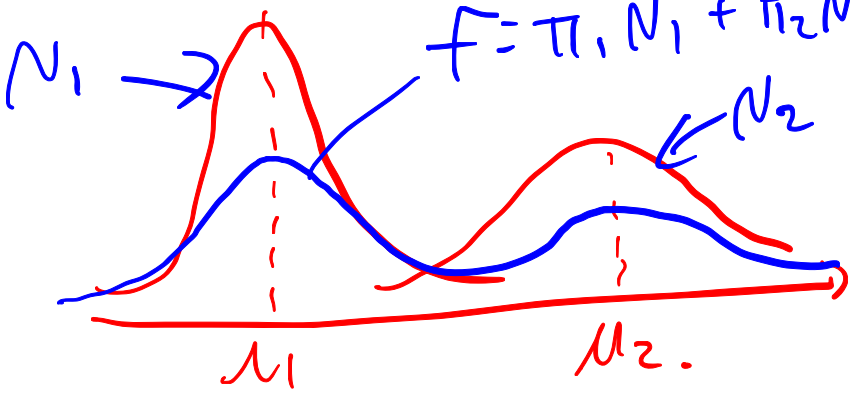
100×100   100×100





mixture distribution.

1-D Gaussian $N(\mu_1, \sigma_1^2)$, $N(\mu_2, \sigma_2^2)$.



$$N(x; \mu, \sigma^2) = \frac{1}{\sqrt{2\pi\sigma^2}} \exp\left\{ - \frac{(x-\mu)^2}{2\sigma^2} \right\}.$$

mixture distribution

$$f(x; \mu_1, \sigma_1^2, \mu_2, \sigma_2^2, \pi_1, \pi_2)$$

mixture coefficient

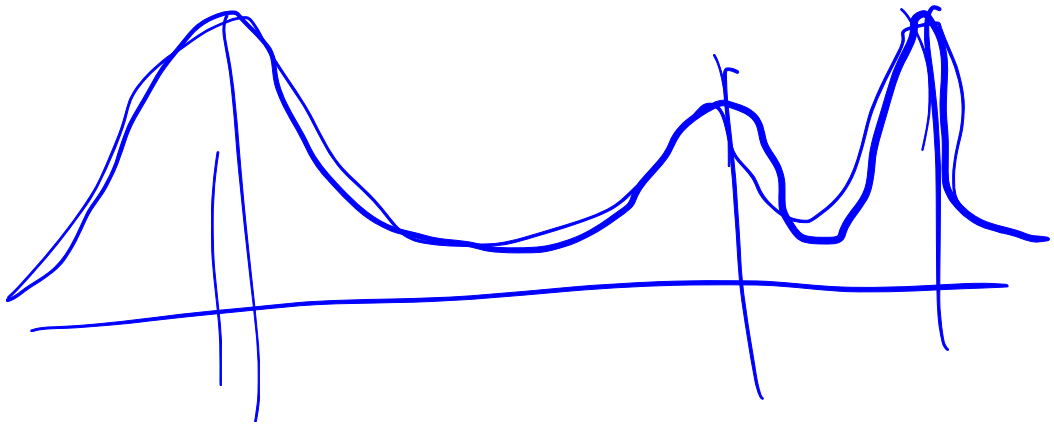
$$\pi_1 + \pi_2 = 1,$$

$$= \underbrace{\pi_1}_{0.5} N(x; \mu_1, \sigma_1^2) + \underbrace{\pi_2}_{0.5} N(x; \mu_2, \sigma_2^2)$$

" mixture distribution"

$N(\mu, \sigma^2)$ mode = 하나인 확률분포.

mixture, 여러개의 mode가 가능해짐. 2, 3, 4, ...



mode = 최대 $\sum_{k=1}^{2} N(x; \mu_k, \sigma_k^2)$.

MoG: model

Mixture of Gaussian distributions can be written as a linear superposition of Gaussians.

$$p(\mathbf{x}) = \sum_{k=1}^{K} \pi_k \mathcal{N}(\mathbf{x}|\boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k) \tag{6}$$

Let us introduce K-dimensional binary random variable $\mathbf{z}$ having a 1-of-K representation in which a particular element z_k is equal to 1 and all other elements are equal to 0. The values of z_k therefore satisfy $z_k \in \{0, 1\}$ and $\sum_k z_k = 1$, and we see that there are K possible states for the vector $\mathbf{z}$ according to which element is nonzero.

The marginal distribution over $\mathbf{z}$ is specified in terms of the mixing coefficients π_k, such that

$$p(z_k = 1) = \pi_k \tag{7}$$

where $0 \leq \pi_k \leq 1$ and $\sum_{k=1}^{K} \pi_k = 1$.

MoG: 1-of-K representation

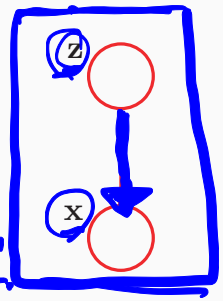


Figure 9.4 Graphical representation of a mixture model, in which the joint distribution is expressed in the form $p(\mathbf{x}, \mathbf{z}) = p(\mathbf{z})p(\mathbf{x}|\mathbf{z})$.

Because $\mathbf{z}_k$ uses a 1-of-K representation, we can also write this distribution in the form

$$p(\mathbf{z}) = \prod_{k=1}^{K} \pi_k^{z_k} \tag{8}$$

Similarly, the conditional distribution of $\mathbf{x}$ given a particular value for $\mathbf{z}$ is a Gaussian

$$p(\mathbf{x}|z_k = 1) = \mathcal{N}(\mathbf{x}|\boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k) \tag{9}$$

which can also be written in the form

$$p(\mathbf{x}|\mathbf{z}) = \prod_{k=1}^{K} \mathcal{N}(\mathbf{x}|\boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k)^{z_k} \tag{10}$$

MoG: model

The joint distribtion is given by $p(\mathbf{x}|\mathbf{z})p(\mathbf{z})$, and the marginal distribution of $\mathbf{x}$ is then obtained by summing the joint distribution over all possible states of $\mathbf{z}$ to give

$$p(\mathbf{x}) = \sum_{\mathbf{z}} p(\mathbf{x}|\mathbf{z})p(\mathbf{z}) = \sum_{k=1}^{K} \pi_k \mathcal{N}(\mathbf{x}|\boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k) \tag{11}$$

Another quantity that will play an important role is the condition probability of $\mathbf{z}$ given $\mathbf{x}$. We shall use $\gamma(z_k)$ to denote $p(z_k = 1|\mathbf{x})$, whose value can be found using Bayes' theorem

$$\gamma(z_k) \equiv p(z_k = 1|\mathbf{x}) = \frac{p(\mathbf{x}|z_k = 1)p(z_k = 1)}{\sum_{j=1}^{K} p(\mathbf{x}|z_j = 1)p(z_j = 1)} \tag{12}$$

$$= \frac{\pi_k \mathcal{N}(\mathbf{x}|\boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k)}{\sum_{j=1}^{K} \pi_k \mathcal{N}(\mathbf{x}|\boldsymbol{\mu}_j, \boldsymbol{\Sigma}_j)} \tag{13}$$

marginal

~~posterior~~

likelihood prior

$$p(x) = \sum_z p(x|z)p(z)$$

posterior.

$$p(z|x) = \frac{p(z,x)}{p(x)} = \frac{p(x|z)p(z)}{\sum p(x|z)p(z)}$$

직관적인 의미

x가 주어졌을 때, z의 확률

= k번째 cluster에 소속될 확률

x가 주어졌을 때

k=1	0.1
:	0.1
	0.3
5	0.5
	0.0

주어진 x의

$$\arg\max_k \gamma(z_k) = p(z|x)$$

$p(z_1|x) = \gamma(z_1) = 0.1$

$p(z_2|x) = \gamma(z_2) = 0.05$

$p(z_3|x) = \gamma(z_3) = 0.55$

$p(z_4|x) = \gamma(z_4) = 0.3$

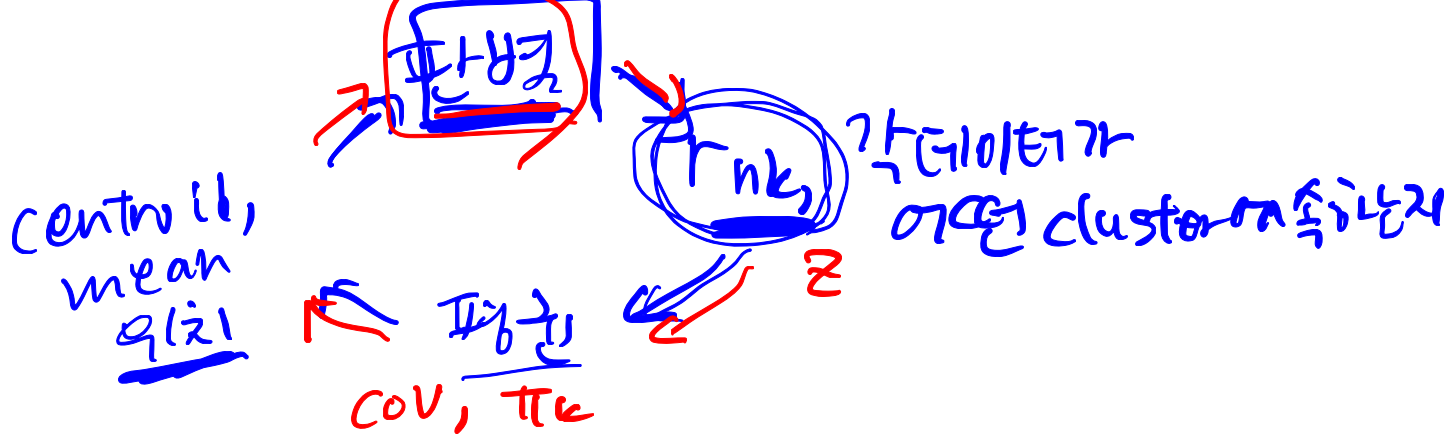
→ 주어진 x에서 각각의 cluster에 속할 확률

$$\arg\max_{k} \gamma(z_k)$$

$$p(z|x) = \frac{p(z,x)}{p(x)} = \frac{p(x|z)\,p(z)}{\sum_z p(x|z)\,p(z)}$$

$$p(x) = \sum_z p(z,x)$$

K-means.



$$\underset{k}{\arg\min} \; \|x_n - \mu_k\|^2$$

MoG.

$$\max. \; \gamma(z_k)$$

분산

MoG: model

We shall view π_k as the prior probability of $z_k = 1$, and the quantity $\gamma(z_k)$ as the corresponding posterior probability once we have observed $\mathbf{x}$. As we shall see later, $\gamma(z_k)$ can also be viewed as the *responsibility* that component k takes for 'explaining' the observation $\mathbf{x}$.

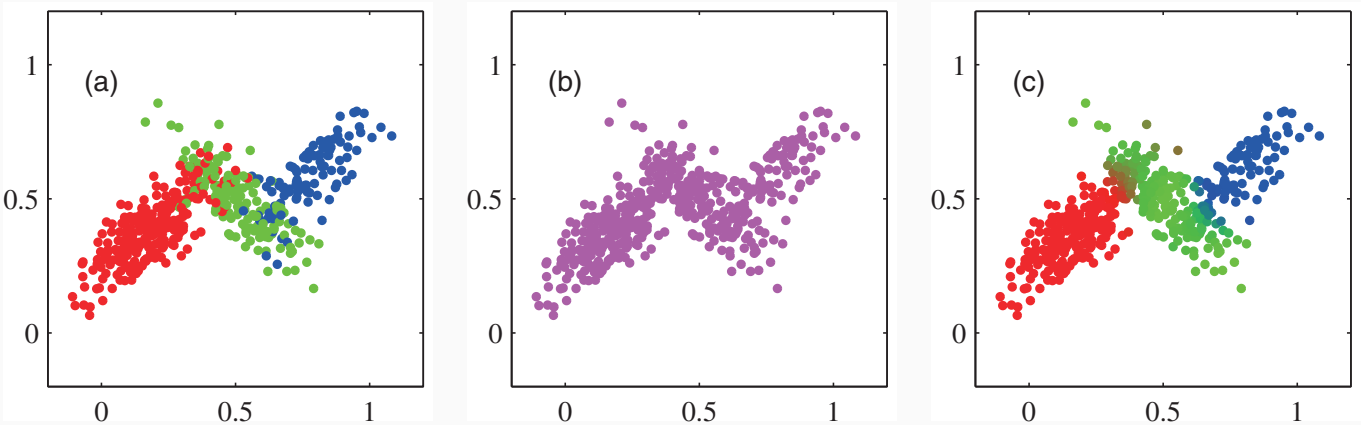


Figure 9.5 Example of 500 points drawn from the mixture of 3 Gaussians shown in Figure 2.23. (a) Samples from the joint distribution $p(\mathbf{z})p(\mathbf{x}|\mathbf{z})$ in which the three states of $\mathbf{z}$, corresponding to the three components of the mixture, are depicted in red, green, and blue, and (b) the corresponding samples from the marginal distribution $p(\mathbf{x})$, which is obtained by simply ignoring the values of $\mathbf{z}$ and just plotting the $\mathbf{x}$ values. The data set in (a) is said to be *complete*, whereas that in (b) is *incomplete*. (c) The same samples in which the colours represent the value of the responsibilities $\gamma(z_{nk})$ associated with data point $\mathbf{x}_n$, obtained by plotting the corresponding point using proportions of red, blue, and green ink given by $\gamma(z_{nk})$ for $k = 1, 2, 3$, respectively

MoG: example

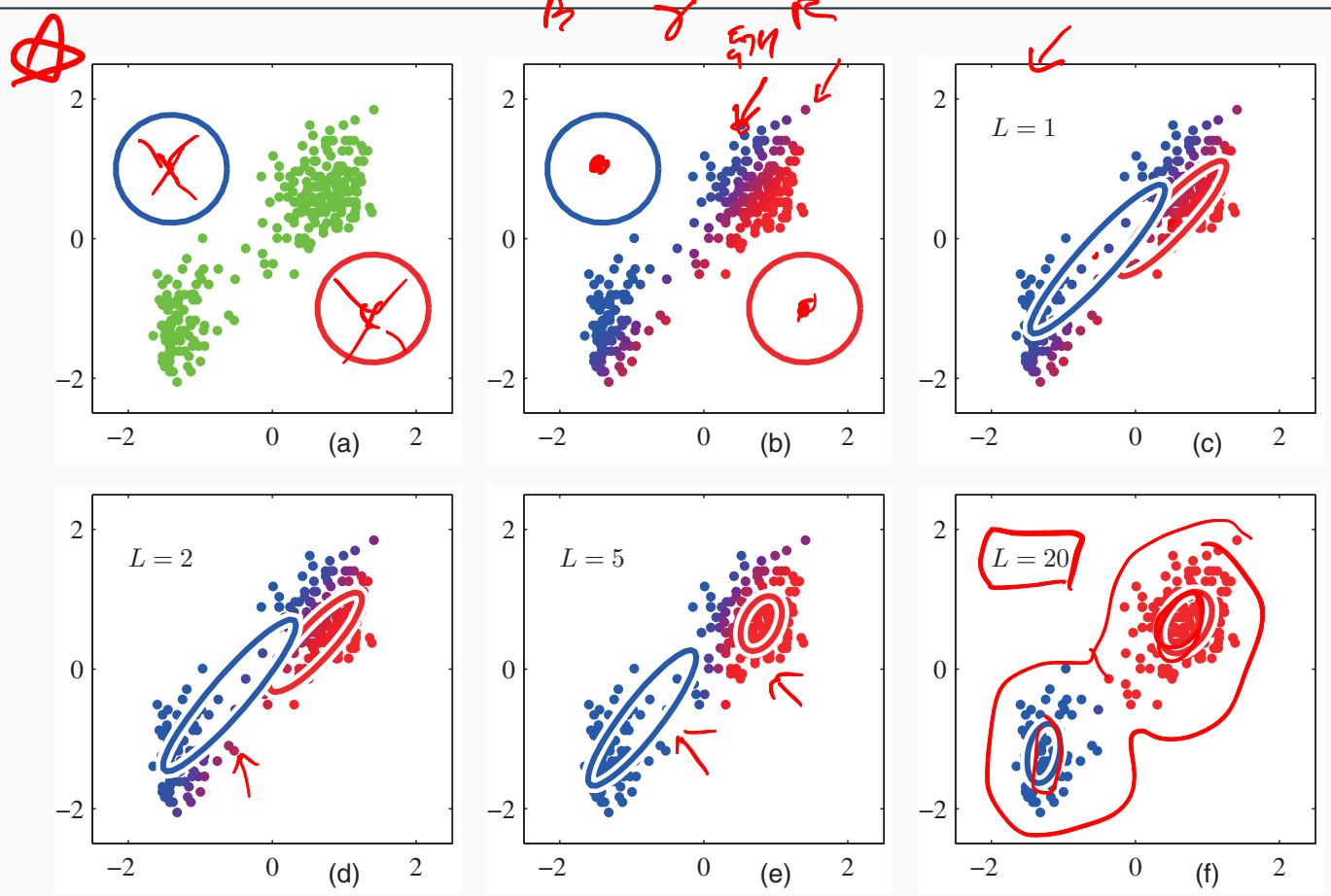


Figure 9.8 Illustration of the EM algorithm using the Old Faithful set as used for the illustration of the K-means algorithm in Figure 9.1. See the text for details.

MoG: python example

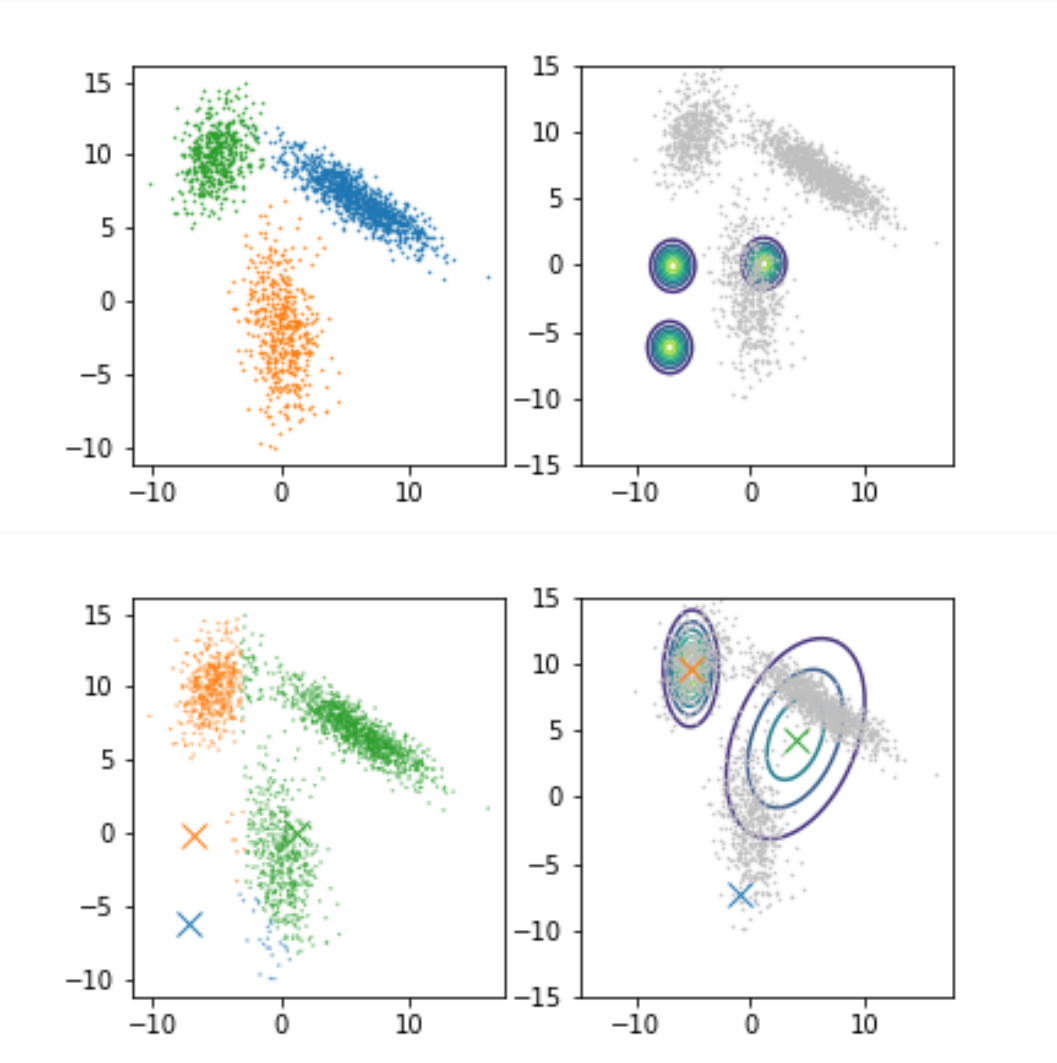


Figure 1: MoG python example: dataset and initialization (top) and the 1st iteration (bottom).

MoG: python example

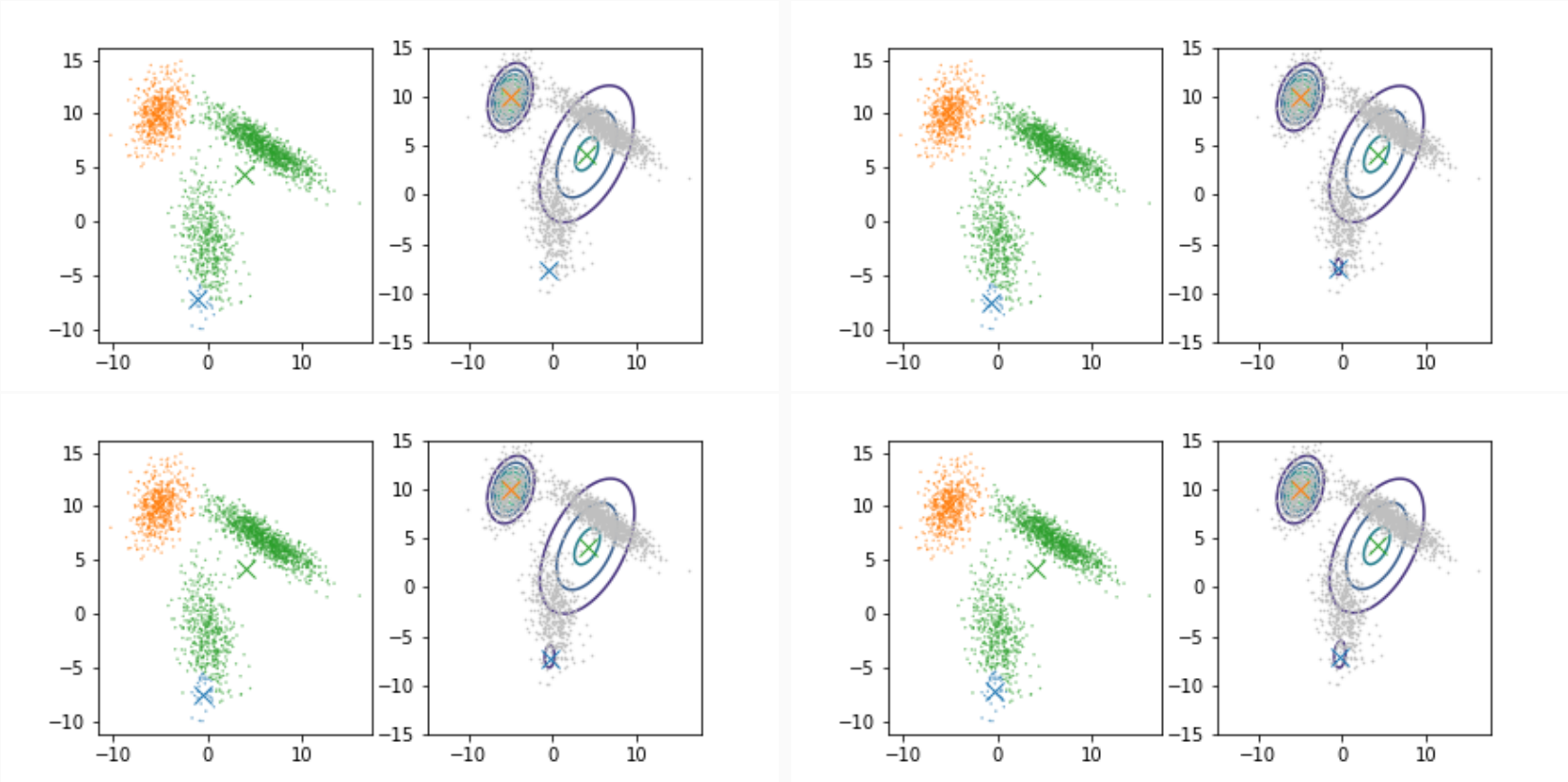


Figure 2: From the 2nd to 5th iterations.

MoG: python example

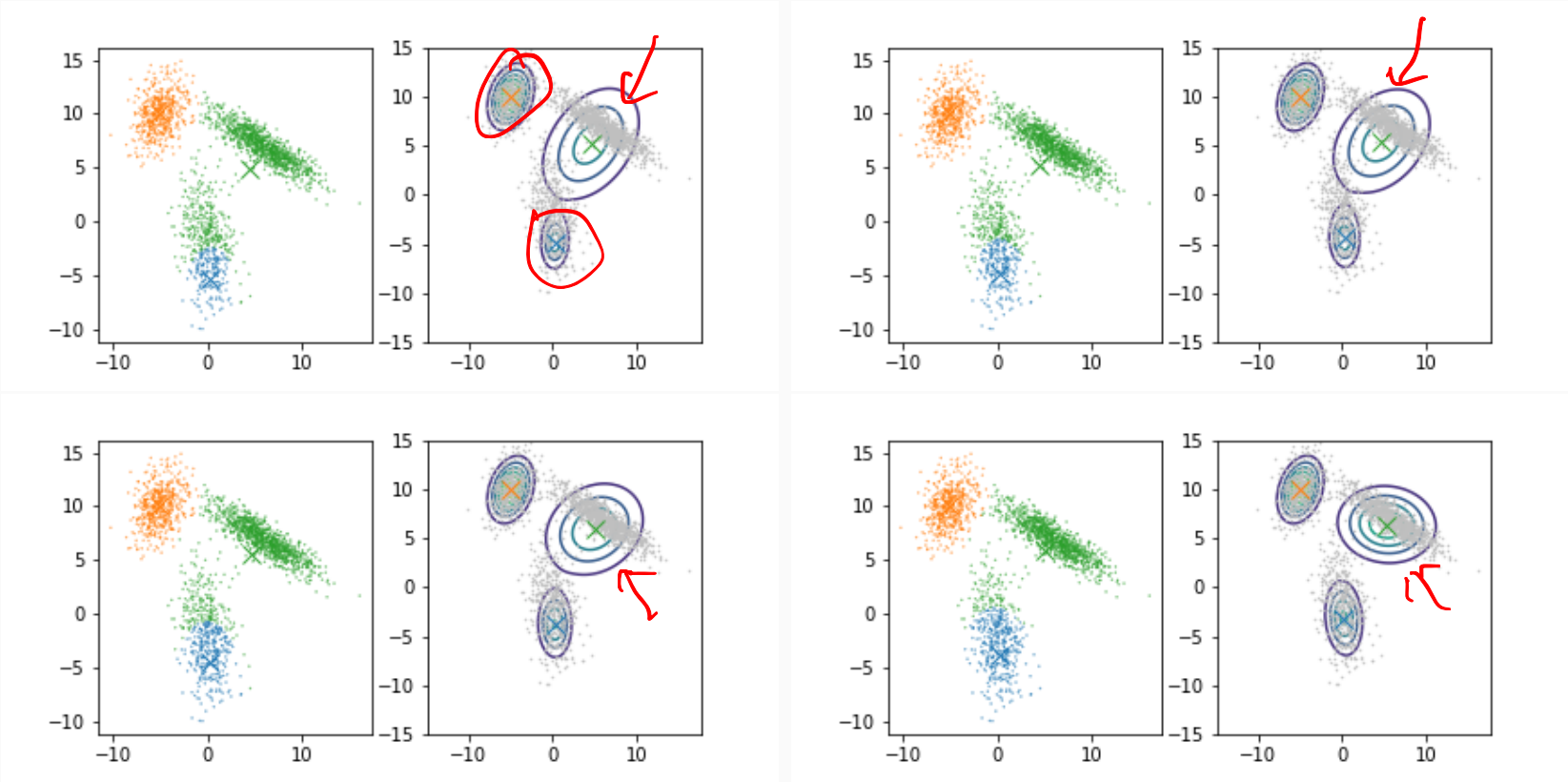


Figure 3: From the 12th to 15th iterations.

MoG: python example

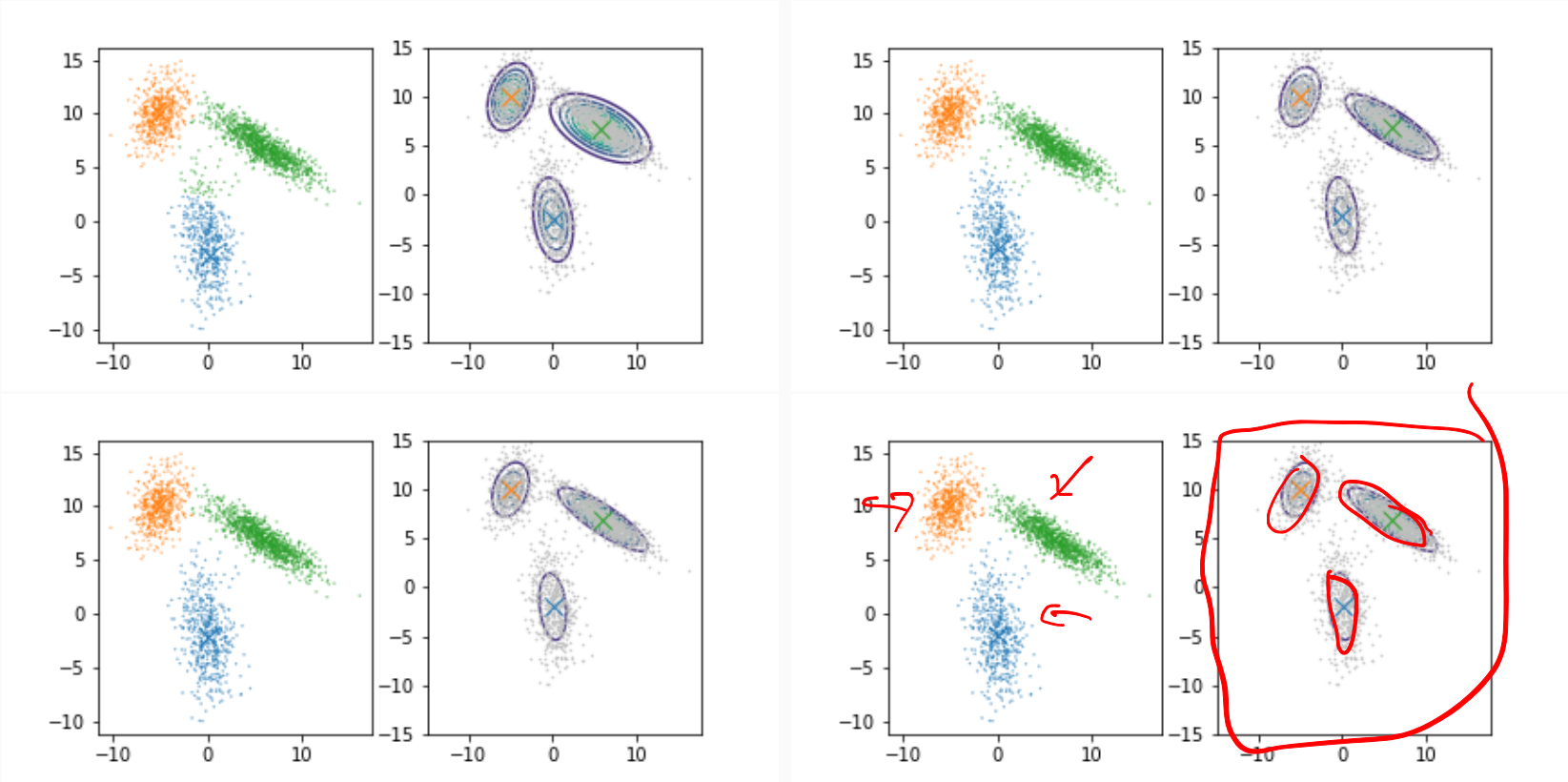


Figure 4: From the 16th to 19th iterations.

Mixtures of Gaussians: EM algorithm

- Maximum likelihood Estimate (MLE)
- Lagrangian multiplier.

MoG: maximum likelihood estimate

Maximum likelihood estimation, or MLE, is on flavor of parameter estimation in machine learning. In order to perform parameter estimation, we need:

- some data $\mathbf{x}$
- some hypothesized generating function of the data $f(\mathbf{x}, \theta)$
- a set of parameters from that function θ
- some evaluation of the goodness of our parameters (an objective function)

In MLE, the objective function (evaluation) we chose is the likelihood of the data given our model. To find the best θ then, we need to find the θ which maximizes our evaluation function (the likelihood). Therefore, in its general form the MLE is:

$$\hat{\theta}_{\text{MLE}} = \arg\max_{\theta} p(\mathbf{x}|\theta) \tag{14}$$

MoG: maximum likelihood estimate

Gaussian distribution을 따르는 i.i.d. 샘플 $\mathbf{x} = (x_1, x_2, ..., x_N)$로부터 평균 $\theta = \mu$를 MLE로 추정하면,

$$\mathcal{L} = p(\mathbf{x}|\theta) = \prod_{n=1}^{N} \mathcal{N}(x_n|\mu) \tag{15}$$

$$\log \mathcal{L} = \sum_{n=1}^{N} \log \mathcal{N}(x_n|\mu) \tag{16}$$

$$\frac{d}{d\mu} \log \mathcal{L} = -\text{const} \sum_{n=1}^{N} (x_n - \mu) \tag{17}$$

$$\frac{d}{d\mu} \log \mathcal{L} = 0 \iff \hat{\mu} = \frac{1}{N} \sum_{n=1}^{N} x_n \tag{18}$$

$\hat{\mu}_{MLE}$ = arithmetic mean

MoG: EM algorithm

An elegant and powerful method for finding maximum likelihood solutions for models with latent variables is called the *expectation-maximization* algorithm, or EM algorithm.

Let us begin by writing down the conditions that must be satisfied at a maximum of the likelihood function.

$$\log p(\mathbf{X}|\boldsymbol{\mu}, \boldsymbol{\Sigma}, \boldsymbol{\pi}) = \sum_{n=1}^{N} \log \left\{ \sum_{k=1}^{K} \pi_k \mathcal{N}(\mathbf{x}_n|\boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k) \right\} \tag{19}$$

Setting the derivatives of log likelihood with respect to the means $\boldsymbol{\mu}_k$ of the Gaussian components to zero,

$$0 = -\sum_{n=1}^{N} \underbrace{\frac{\pi_k \mathcal{N}(\mathbf{x}_n|\boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k)}{\sum_{j=1}^{K} \pi_j \mathcal{N}(\mathbf{x}_n|\boldsymbol{\mu}_j, \boldsymbol{\Sigma}_j)}}_{\gamma(z_{nk})} \boldsymbol{\Sigma}_k(\mathbf{x}_n - \boldsymbol{\mu}_k) \tag{20}$$

MoG: EM algorithm

Multiplying by $\boldsymbol{\Sigma}_k^{-1}$ and rearranging

$$0 = -\sum_{n=1}^{N} \underbrace{\frac{\pi_k \mathcal{N}(\mathbf{x}_n|\boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k)}{\sum_{j=1}^{K} \pi_j \mathcal{N}(\mathbf{x}_n|\boldsymbol{\mu}_j, \boldsymbol{\Sigma}_j)}}_{\gamma(z_{nk})} \boldsymbol{\Sigma}_k(\mathbf{x}_n - \boldsymbol{\mu}_k) \tag{21}$$

we obtain

$$0 = \sum_{n=1}^{N} \gamma(z_{nk})(\mathbf{x}_n - \boldsymbol{\mu}_k) \tag{22}$$

$$\boldsymbol{\mu}_k = \frac{1}{N_k} \sum_{n=1}^{N} \gamma(z_{nk}) \mathbf{x}_n \tag{23}$$

$$N_k = \sum_{n=1}^{N} \gamma(z_{nk}) \tag{24}$$

MoG: EM algorithm

If we set the derivative of log likelihood with respect to $\boldsymbol{\Sigma}_k$ to zero, and follow a similar line of reasoning, making use of the result for the maximum likelihood solution for the covariance matrix of a single Gaussian, we obtain

$$\boldsymbol{\Sigma}_k = \frac{1}{N_k} \sum_{n=1}^{N} \gamma(z_{nk})(\mathbf{x}_n - \boldsymbol{\mu}_k)(\mathbf{x}_n - \boldsymbol{\mu}_k)^T \tag{25}$$

each data point weighted by the corresponding posterior probability and with the denominator given by the effective number of points associated with the corresponding component

MoG: EM algorithm

Finally, we maximize log likelihood with respect to the mixing coefficients pi_k. Here we must take account of the constraint $\sum_k pi_k = 1$. This can be achieved using a Lagrange multiplier and maximizing the following quantity

$$\log p(\mathbf{X}|\boldsymbol{\pi}, \boldsymbol{\mu}, \boldsymbol{\Sigma}) + \lambda \left(\sum_{k=1}^{K} \pi_k - 1 \right) \tag{26}$$

which gives

$$0 = \sum_{n=1}^{N} \frac{\mathcal{N}(\mathbf{x}_n|\boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k)}{\sum_j \pi_j \mathcal{N}(\mathbf{x}_n|\boldsymbol{\mu}_j, \boldsymbol{\Sigma}_j)} + \lambda \tag{27}$$

$$0 = \sum_{k=1}^{K} \sum_{n=1}^{N} \gamma(z_{nk}) + \lambda \sum_{k=1}^{K} \pi_k \tag{28}$$

Hence

$$\pi_k = \frac{N_k}{N} \tag{29}$$

MoG: Lagrange multiplier

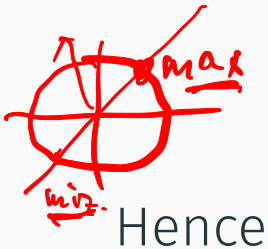
The method can be summarized as follows: in order to find the maximum or minimum of a function $f(x)$ subjected to the equality constraint $g(x) = 0$, form the Lagrangian function

$$\mathcal{L}(x, \lambda) = f(x) - \lambda g(x) \tag{30}$$

and find the stationary points of $\mathcal{L}$ considered as a function of x and the Lagrange multiplier λ. The solution corresponding to the original constrained optimization is always a saddle point of the Lagrangian function, which can be identified among the stationary points from the definiteness of the bordered Hessian matrix.

MoG: Lagrange multiplier

Minimize $f(x, y) = x + y$ subject to the constraint $x^2 + y^2 = 1$, i.e.,



$$g(x, y) = x^2 + y^2 - 1 = 0 \tag{31}$$

Hence,

$$\mathcal{L}(x, y, \lambda) = f(x, y) + \lambda g(x, y) = x + y + \lambda(x^2 + y^2 - 1) \tag{32}$$

Gradient

$$\nabla_{x,y,\lambda} \mathcal{L}(x, y, \lambda) = (1 + 2\lambda x, 1 + 2\lambda y, x^2 + y^2 - 1) \tag{33}$$

and therefore,

$$\nabla_{x,y,\lambda} \mathcal{L}(x, y, \lambda) \iff \left\{ \begin{array}{l} 1 + 2\lambda x = 0 \\ 1 + 2\lambda y = 0 \\ x^2 + y^2 - 1 = 0 \end{array} \right\} \tag{34}$$

MoG: Lagrange multiplier

$$\nabla_{x,y,\lambda}\mathcal{L}(x,y,\lambda) \iff \begin{cases} 1+2\lambda x = 0 \\ 1+2\lambda y = 0 \\ x^2+y^2-1=0 \end{cases} \tag{35}$$

This yields

$$x = y = -\frac{1}{2\lambda}, \quad \lambda \neq 0 \tag{36}$$

$$\frac{1}{4\lambda^2} + \frac{1}{4\lambda^2} - 1 = 0 \tag{37}$$

So,

$$\lambda = \pm\frac{1}{\sqrt{2}} \tag{38}$$

which implies that the stationary points of $\mathcal{L}$ are

$$\left(\frac{\sqrt{2}}{2}, \frac{\sqrt{2}}{2}, -\frac{\sqrt{2}}{2}\right), \quad \left(-\frac{\sqrt{2}}{2}, -\frac{\sqrt{2}}{2}, \frac{\sqrt{2}}{2}\right) \tag{39}$$

Appendix

Reference and further reading

- “Chap 9” of A. Geron, Hands-On Machine Learning with Scikit-Learn, Keras & TensorFlow
- “Chap 9” of C. Bishop, Pattern Recognition and Machine Learning
- D. Arthur, S. Vassilvitskii, k-means++: The advantages of careful seeding (2006)
- Z. Ghahramani, M. J. Beal, Variational Inference for Bayesian mixtures of Factor Analysers (2000)

HW #4

Due: 9월 22일 화요일, 23시 59분까지

- pdf로 업로드하세요.
- 손으로 작성한 파일을 스캔앱(Adobe scan, Office lens 등)을 써서 pdf로 저장해주세요.
- 컴퓨터로 작성(latex, word, ppt, 한글 등)한 파일도 가능합니다. pdf로 저장해주세요.
- 가독성이 떨어지는 파일도 불량처리 합니다.

1. Gaussian distribution $\mathcal{N}(x; \mu, \sigma)$를 따르는 N개의 i.i.d. 샘플 $\mathbf{x} = (x_1, x_2, ..., x_N)$에 대해 $\theta = (\mu, \sigma)$에 대한 MLE를 유도과정과 함께 구하세요. (20점)

2. python 코드에 내 말한 부분 추가해서 올릴게요.