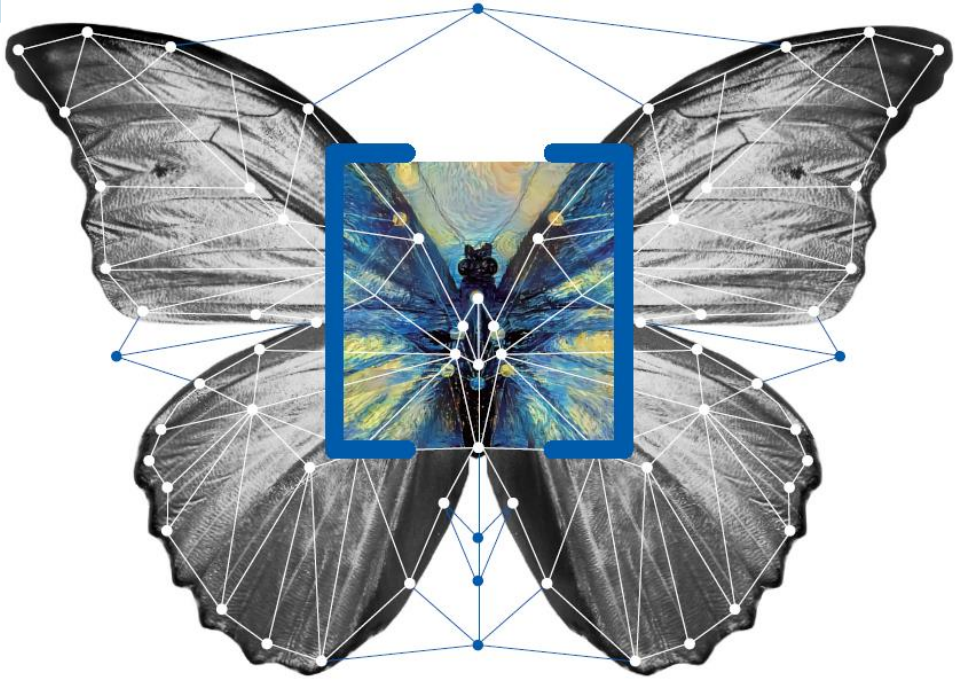
07_가우시안 혼합 모델(GMM)



MACHINE LEARNING 기계 학습

오일석 지음

6장. 비지도 학습

6.4 밀도 추정

- 6.4.1 커널 밀도 추정
- 6.4.2 가우시안 혼합
- 6.4.3 EM 알고리즘

- 밀도 추정 문제
  - 어떤 점 x에서 데이터가 발생할 확률, 즉 확률분포 $P(\mathbf{x})$를 구하는 문제
  - 예를 들어, 그림 6-8에서 $P(\mathbf{x}_1) > P(\mathbf{x}_2) > P(\mathbf{x}_3)$

따란색의 사전에 수집된 데이터로 학률밀도함수를 학습

오른쪽 그림을 직관적으로 보면 X_1 학률이 가장 높다

Why? X_1 주변에 데이터가 밀집해 있으니까

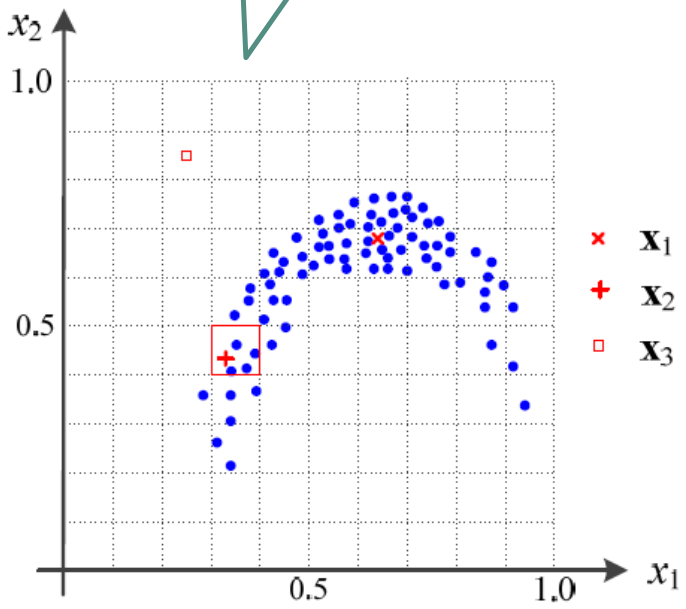


그림 6-8 밀도 추정 문제

6.4.1 커널 밀도 추정

히스토그램 방법

구간 (bin)

- 특징 공간을 칸의 집합으로 분할한 다음, 칸에 있는 샘플의 빈도를 세어 식 (6.7)로 추정
- 예, $P(\mathbf{x} = +) = \frac{4}{80} = 0.05$

특정 크기의 주변공간안에 데이터가 몇 개있는지의 비율 : 전체 80개 중에 + 주변에 4개 존재할 경우

$$P(\mathbf{x}) = \frac{bin(\mathbf{x})}{n} \qquad (6.7)$$

- 여러 문제점
  - 매끄럽지 못하고 계단 모양을 띠는 확률밀도함수가 됨
  - 칸의 크기와 위치에 민감함

어떻게 최적의 크기와 위치를 찾아낼 것인가?

6.4.1 커널 밀도 추정

커널 밀도 추정법

- 점 x에 [그림 6-9]가 예시하는 커널을 씌우고 커널 안에 있는 샘플의 가중 합을 이용함
- 대역폭 h의 크기가 중요

$$P_h(\mathbf{x}) = \frac{1}{n}\sum_{i=1}^{n} K_h(\mathbf{x}-\mathbf{x}_i) = \frac{1}{nh^d}\sum_{i=1}^{n} K\left(\frac{\mathbf{x}-\mathbf{x}_i}{h}\right) \quad (6.8)$$

여기서 $K_h(\mathbf{x}) = \frac{1}{h^d} K\left(\frac{\mathbf{x}}{h}\right)$

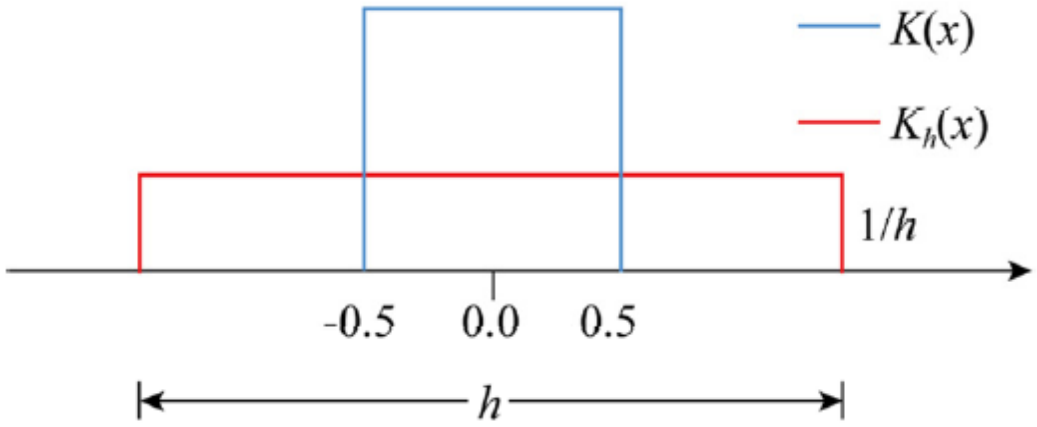


그림 6-9 표준커널함수 K와 크기 변환된 커널함수 K_h

6.4.1 커널 밀도 추정

히스토그램 방법과 커널 밀도 추정법의 비교

- 커널 밀도 추정법은 어떤 커널을 사용하는가에 따라 다양한 확률밀도함수를 추정할 수 있음.
- 오른쪽은 가우시안 함수를 커널로 사용하여 매끄러운 결과를 얻음.

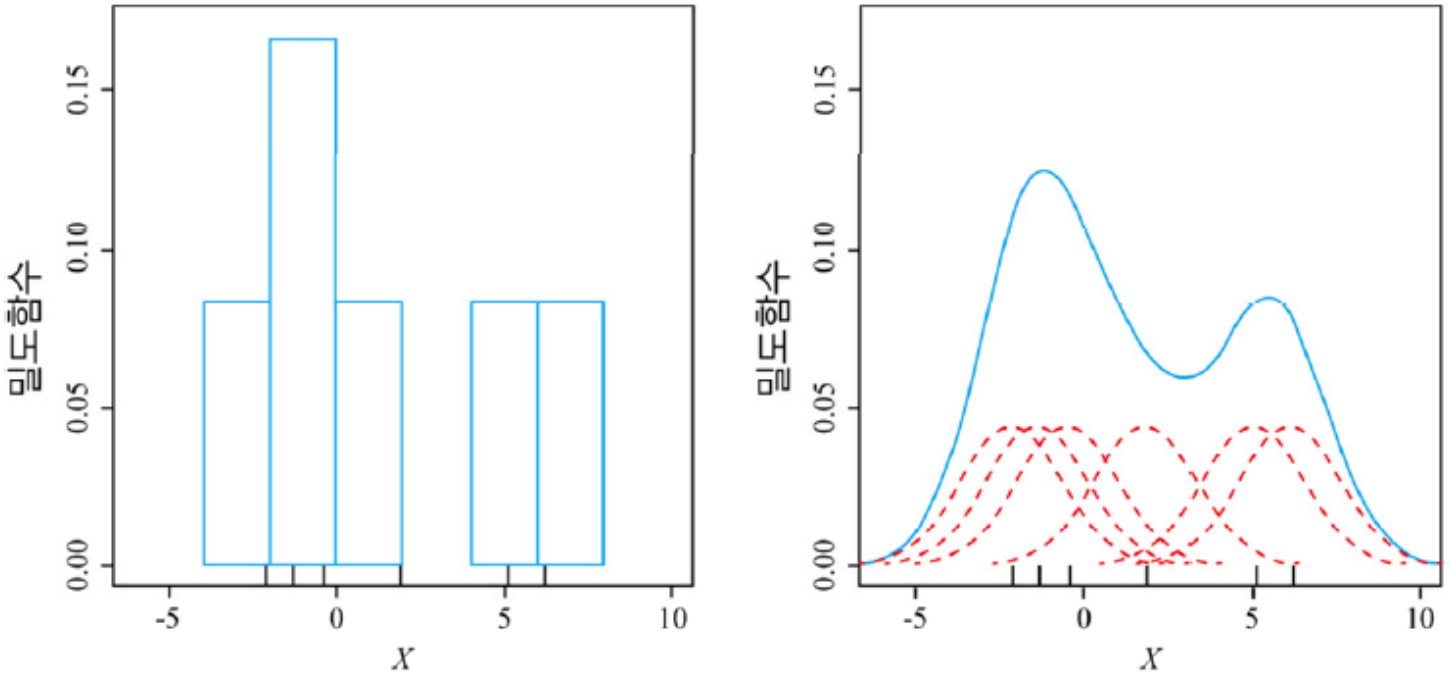


그림 6-10 히스토그램 방법(왼쪽)과 커널 밀도 추정법(오른쪽)의 비교

6.4.1 커널 밀도 추정

커널 밀도 추정법에서 대역폭 h의 중요성

- h가 너무 작으면(빨강) 뾰족뾰족한 모양, h가 너무 크면(녹색) 뭉개짐, 적절하게 설정해야 함(검정)

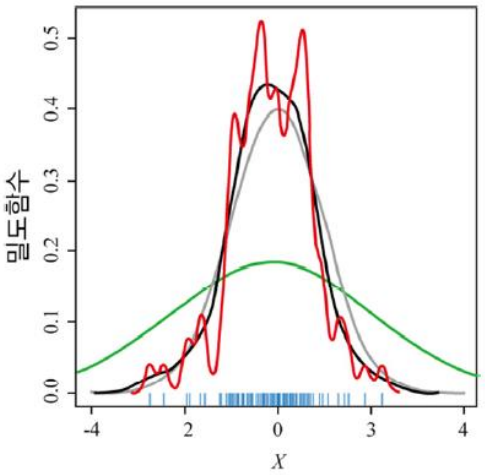


그림 6-11 대역폭이 확률밀도함수 추정에 미치는 영향

여전히 h 에 의해 변동성이 심함.

커널 밀도 추정 기법의 근본적 문제점

- 샘플을 모두 저장하고 있어야 하는 메모리 기반 방법(새로운 샘플이 주어질 때마다 식 (6.8)을 처음부터 다시 계산)
- 데이터 희소성(차원의 저주)

→ 데이터가 낮은 차원인 경우로 국한하여 활용

차원수가 늘수록 그만큼 더 많은 데이터가 학습에 필요

6.4.2 가우시안 혼합

- **가우시안을 이용한 방법**

  데이터의 희소성을 피할 수 있음. 대신 데이터가 가우시안 분포를 가진다는 가정이 필수.

  - 데이터가 가우시안 분포를 따른다고 가정하고 평균 벡터 μ와 공분산 행렬 Σ를 추정함

$$P(\mathbf{x}) = N(\mathbf{x}; \boldsymbol{\mu}, \boldsymbol{\Sigma}) = \frac{1}{\sqrt{|\boldsymbol{\Sigma}|}\sqrt{(2\pi)^d}} \exp\left(-\frac{1}{2}(\mathbf{x}-\boldsymbol{\mu})^{\mathrm{T}}\boldsymbol{\Sigma}^{-1}(\mathbf{x}-\boldsymbol{\mu})\right)$$

$$\text{이때 } \boldsymbol{\mu} = \frac{1}{n}\sum_{i=1,n}\mathbf{x}_i\,,\ \boldsymbol{\Sigma} = \frac{1}{n}\sum_{i=1,n}(\mathbf{x}_i-\boldsymbol{\mu})(\mathbf{x}_i-\boldsymbol{\mu})^{\mathrm{T}}$$

오로지 데이터에만 의존하지 않고 적정한 개수의 데이터만으로 평균과 분산만 구하고 나머지는 추정해버리기 때문

- **대부분 데이터가 하나의 가우시안으로 불충분([그림 6-12]의 오른쪽)**

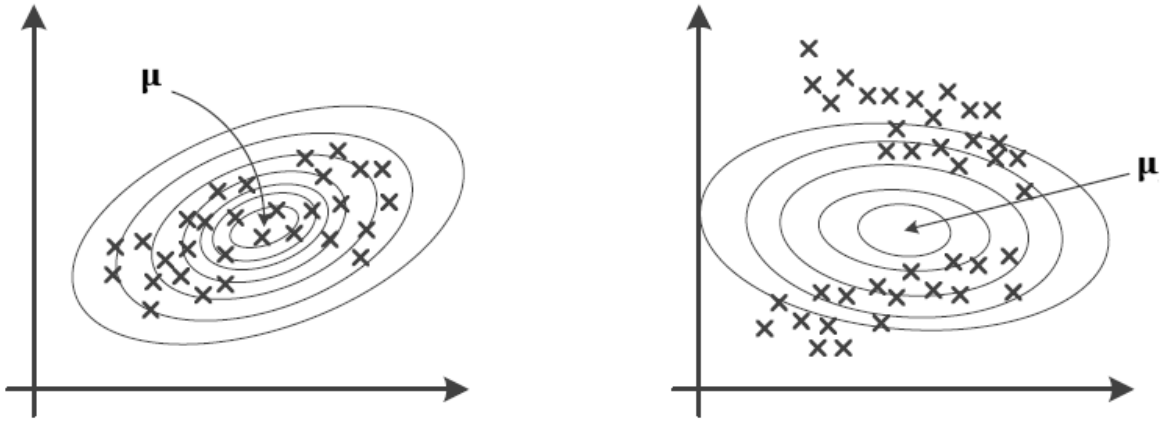


그림 6-12 하나의 가우시안으로 밀도 추정

6.4.2 가우시안 혼합

가우시안 혼합

- [그림 6-13]은 2개의 가우시안을 사용한 예

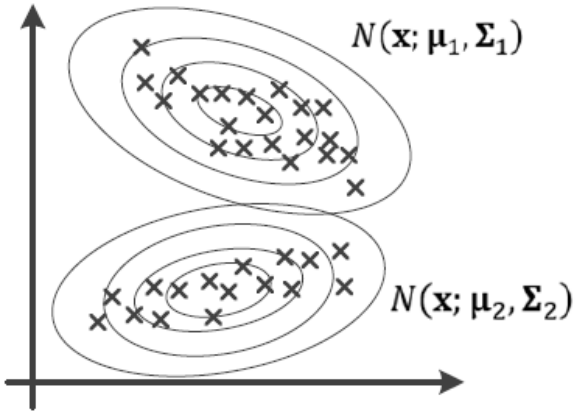


그림 6-13 가우시안 혼합으로 밀도 추정

전체 : 37개 데이터
윗쪽 가우시안 : 21개 데이터
아랫쪽 가우시안 : 16개 데이터

$$P(\mathbf{x}) = \frac{21}{37}N(\mathbf{x};\boldsymbol{\mu}_1,\boldsymbol{\Sigma}_1) + \frac{16}{37}N(\mathbf{x};\boldsymbol{\mu}_2,\boldsymbol{\Sigma}_2)$$

k개의 가우시안으로 일반화하면,

- 확률분포 $P(\mathbf{x})$는 k개 가우시안의 선형 결합으로 표현(식 (6.10))

모든 Π의 합은 1.
왜냐하면 여러 개의 가우시안이 혼합하여 하나의 확률밀도함수를 만드는 것이기 때문에

$$P(\mathbf{x}) = \sum_{j=1}^{k} \pi_j N(\mathbf{x};\boldsymbol{\mu}_j,\boldsymbol{\Sigma}_j) \qquad (6.10)$$

6.4.2 가우시안 혼합

- **주어진 데이터와 추정해야 할 매개변수를 정리하면,**

주어진 데이터: 훈련집합 $\mathbb{X} = \{\mathbf{x}_1, \mathbf{x}_2, \cdots, \mathbf{x}_n\}$, 가우시안의 개수 k

추정해야 할 매개변수집합: $\Theta = \{\boldsymbol{\pi} = (\pi_1, \pi_2, \cdots, \pi_k), (\boldsymbol{\mu}_1, \boldsymbol{\Sigma}_1), (\boldsymbol{\mu}_2, \boldsymbol{\Sigma}_2), \cdots, (\boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k)\}$

- **최대 우도를 이용한 최적화 문제로 공식화**

$$P(\mathbb{X}|\Theta) = \prod_{i=1}^{n} P(\mathbf{x}_i|\Theta) = \prod_{i=1}^{n} \left(\sum_{j=1}^{k} \pi_j N(\mathbf{x}_i; \boldsymbol{\mu}_j, \boldsymbol{\Sigma}_j) \right) \quad (6.11)$$

$$\log P(\mathbb{X}|\Theta) = \sum_{i=1}^{n} \log \left(\sum_{j=1}^{k} \pi_j N(\mathbf{x}_i; \boldsymbol{\mu}_j, \boldsymbol{\Sigma}_j) \right) \quad (6.12)$$

$$\hat{\Theta} = \underset{\Theta}{\operatorname{argmax}} \log P(\mathbb{X}|\Theta) \quad (6.13)$$

6.4.3 EM 알고리즘

EM 알고리즘을 이용한 식 (6.13)의 풀이

- Θ를 모르므로 난수로 설정하고 출발([그림 6-14]의 예시)

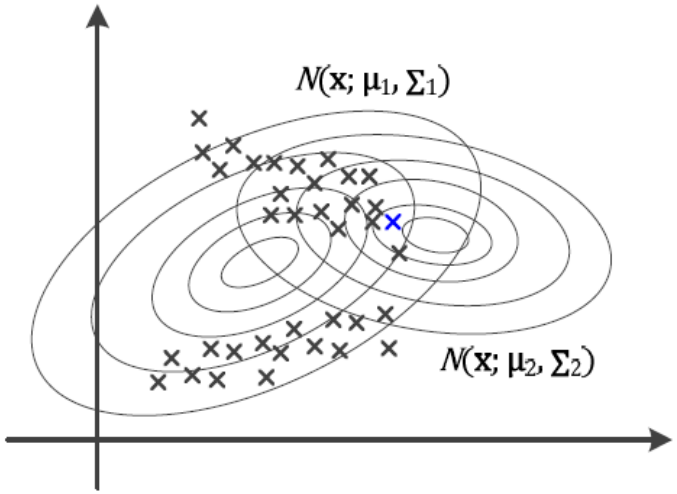


그림 6-14 샘플의 소속 확률을 어떻게 추정할 것인가

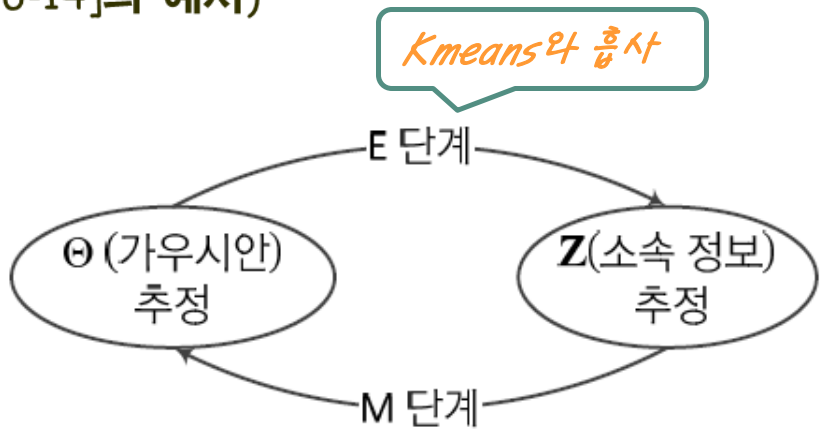


그림 6-15 가우시안 혼합을 위한 EM 알고리즘

- 가우시안으로 샘플의 소속 정보 개선(E단계) → 샘플의 소속 정보로 가우시안 개선(M단계) → 가우시안으로 샘플의 소속 정보 개선(E단계) → 샘플의 소속 정보로 가우시안 개선(M단계) → .... ([그림 6-15])

6.4.3 EM 알고리즘

가우시안 혼합을 위한 EM 알고리즘

알고리즘 6–4 가우시안 혼합을 위한 EM 알고리즘

입력: 훈련집합 $\mathbb{X} = \{\mathbf{x}_1, \mathbf{x}_2, \cdots, \mathbf{x}_n\}$, 가우시안의 개수 k

출력: 최적의 가우시안과 혼합 계수 $\Theta = \{\boldsymbol{\pi} = (\pi_1, \pi_2, \cdots, \pi_k), (\boldsymbol{\mu}_1, \boldsymbol{\Sigma}_1), (\boldsymbol{\mu}_2, \boldsymbol{\Sigma}_2), \cdots, (\boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k)\}$

```
1  Θ를 초기화한다.
2  while (!멈춤조건)
3     Θ를 이용하여 소속확률 행렬 Z를 추정한다.   // E단계
4     Z를 이용하여 Θ를 추정한다.                 // M단계
```

- 라인 3과 라인 4를 위한 수식
- z_{ji}는 $\mathbf{x}_i$가 j번째 가우시안에 속할 확률

$$z_{ji} = \frac{\pi_j N(\mathbf{x}_i; \boldsymbol{\mu}_j, \boldsymbol{\Sigma}_j)}{\sum_{q=1}^{k} \pi_q N(\mathbf{x}_i; \boldsymbol{\mu}_q, \boldsymbol{\Sigma}_q)} \tag{6.14}$$

$$\left.\begin{aligned} \boldsymbol{\mu}_j &= \frac{1}{n_j}\sum_{i=1}^{n} z_{ji}\mathbf{x}_i \\ \boldsymbol{\Sigma}_j &= \frac{1}{n_j}\sum_{i=1}^{n} z_{ji}(\mathbf{x}_i - \boldsymbol{\mu}_j)(\mathbf{x}_i - \boldsymbol{\mu}_j)^{\mathrm{T}} \\ \pi_j &= \frac{n_j}{n} \\ \text{이때 } n_j &= \sum_{i=1}^{n} z_{ji} \end{aligned}\right\} \tag{6.15}$$

sklearn.mixture.GaussianMixture

❖ Gaussian Mixture: **가우시안 혼합 분포의 매개 변수 추정**

(https://scikit-learn.org/stable/modules/generated/sklearn.mixture.GaussianMixture.html)

- **주요 파라미터**

- n_components: The number of mixture components, default=1

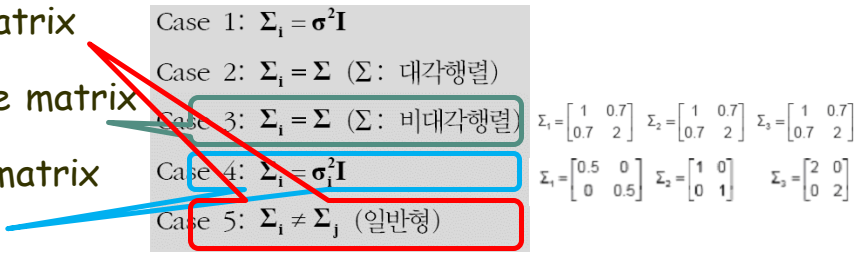
- covariance_type{'full', 'tied', 'diag', 'spherical'}: String describing the type of covariance parameters to use. , default='full'

'full': each component has its own general covariance matrix

'tied': all components share the same general covariance matrix

'diag': each component has its own diagonal covariance matrix

'spherical': each component has its own single variance



- tol: The convergence threshold. EM iterations will stop when the lower bound average gain is below this threshold, default=1e-3

- max_iterant: The number of EM iterations to perform, default=100

- init_params{'kmeans', 'random'}: The method used to initialize the weights, the means and the precisions, default='kmeans'

sklearn.mixture.GaussianMixture

fit(X) :Estimate model parameters with the EM algorithm.

Parameters:
X: *array-like of shape (n_samples, n_features)* 모델 학습을 위한 데이터
List of n_features-dimensional data points. Each row corresponds to a single data point.
Returns:
Self:*object* The fitted mixture.

predict(X): Predict the labels for the data samples in X using trained model.

Parameters:
X: *array-like of shape (n_samples, n_features)* 테스트를 위한 데이터
List of n_features-dimensional data points. Each row corresponds to a single data point.
Returns:
Labels: array, shape (n_samples,) Component labels.

score(X):Compute the per-sample average log-likelihood of the given data X

Parameters:
X*array-like of shape (n_samples, n_dimensions)* 테스트를 위한 데이터
List of n_features-dimensional data points. Each row corresponds to a single data point.
Returns:
log_likelihood:*float* Log-likelihood of X under the Gaussian mixture model.

fit_predict(X): Estimate model parameters using X and predict the labels for X.

Parameters:
X: *array-like of shape (n_samples, n_features)* 모델 학습을 위한 데이터
List of n_features-dimensional data points. Each row corresponds to a single data point.
Returns:
Labels: array, shape (n_samples,) Component labels.

sklearn.mixture.GaussianMixture

❖ 가우시안 혼합 분포의 추정

- Example

```
>>> import numpy as np
>>> from sklearn.mixture import GaussianMixture
>>> X = np.array([[1, 2], [1, 4], [1, 0], [10, 2], [10, 4], [10, 0]])
>>> gm = GaussianMixture(n_components=2, random_state=0).fit(X)
```

파이썬으로 GMM 연습

❖ 아래의 코드를 한번씩 사용해보기

gmmPy.py

plot_gmm_pdf.py

plot_gmm_covariances.py