

10th Meet

□ 오늘 공부할 것은

07

군집화

| | |
|--------------------------------------|-----|
| 01. K-평균 알고리즘 이해 | 431 |
| 사이킷런 KMeans 클래스 소개 | 432 |
| K-평균을 이용한 붓꽃 데이터 세트 군집화 | 433 |
| 군집화 알고리즘 테스트를 위한 데이터 생성 | 437 |
| 02. 군집 평가(Cluster Evaluation) | 441 |
| 실루엣 분석의 개요 | 442 |
| 붓꽃 데이터 세트를 이용한 군집 평가 | 443 |
| 군집별 평균 실루엣 계수의 시각화를 통한 군집 개수 최적화 방법 | 445 |
| 03. 평균 이동 | 449 |
| 평균 이동(Mean Shift)의 개요 | 449 |
| 04. GMM(Gaussian Mixture Model) | 455 |
| GMM(Gaussian Mixture Model) 소개 | 455 |
| GMM을 이용한 붓꽃 데이터 세트 군집화 | 457 |
| GMM과 K-평균의 비교 | 459 |
| 05. DBSCAN | 463 |
| DBSCAN 개요 | 463 |
| DBSCAN 적용하기 – 붓꽃 데이터 세트 | 467 |
| DBSCAN 적용하기 – make_circles() 데이터 세트 | 471 |
| 06. 군집화 실습 – 고객 세그먼테이션 | 474 |
| 고객 세그먼테이션의 정의와 기법 | 474 |
| 데이터 세트 로딩과 데이터 클렌징 | 475 |
| RFM 기반 데이터 가공 | 478 |
| RFM 기반 고객 세그먼테이션 | 481 |
| 07. 정리 | 487 |

Now, Whole My turn ???

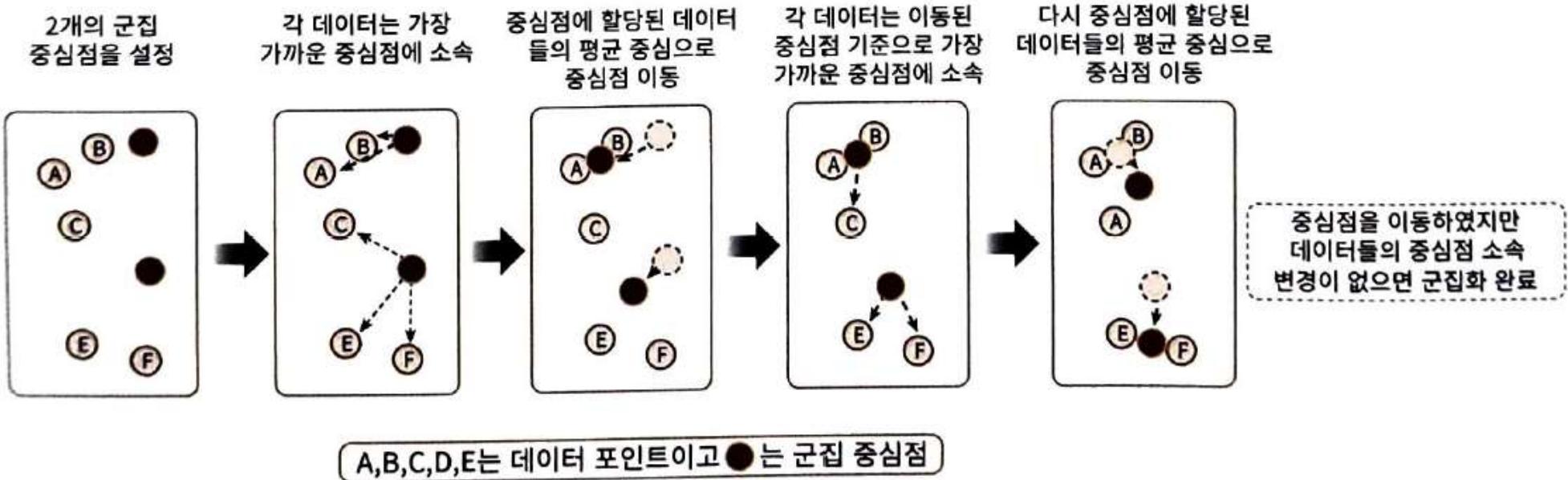
Chapter 07

군집화
(Clustering)

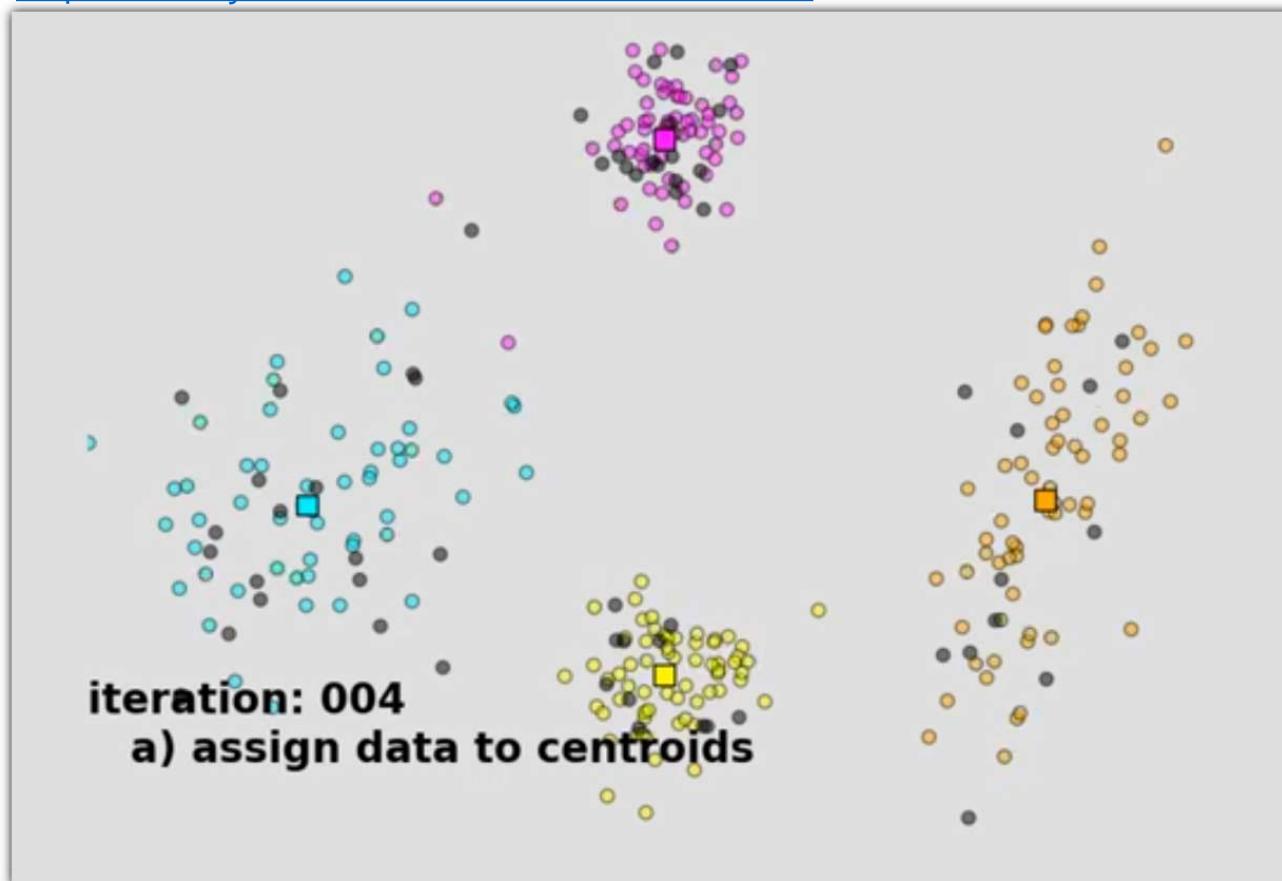
[07-01]

K-평균 알고리즘 이해
(K-means clustering algorithm)

군집 중심점 (centroid)



<https://www.youtube.com/watch?v=5I3Ei69l40s>



▷ K-means clustering algorithm 장단점

- (+) 일반적인 군집화에서 가장 많이 활용되는 알고리즘
- (+) 알고리즘이 쉽고 간결
- (-) 거리 기반 알고리즘으로 속성의 개수가 매우 많을 경우 clustering 정확도 낮음
(이를 위해 PCA를 통해 차원 감소를 적용해야 할 수도 있음)
- (-) 반복 횟수가 많을 경우 오랜 수행 시간 소요
- (-) 몇 개의 군집(cluster)을 선택해야 하는지 어려움

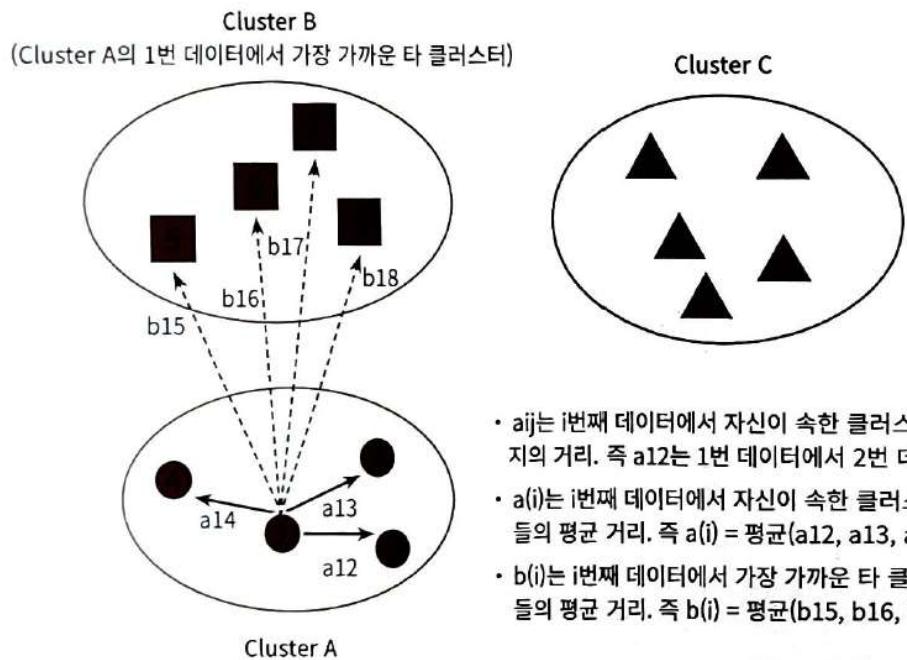
https://github.com/whatwant-school/python-ml/blob/main/10-week/10-week_01-K-means.ipynb

```
: from sklearn.cluster import KMeans
KMeans?
Init signature:
KMeans(
    n_clusters=8,
    *,
    init='k-means++',
    n_init='auto',
    max_iter=300,
    tol=0.0001,
    verbose=0,
    random_state=None,
    copy_x=True,
    algorithm='lloyd',
)
```

[07-02]
군집 평가
(Cluster Evaluation)

▷ 실루엣 분석 (Silhouette Analysis)

- 각 군집 간의 거리가 얼마나 효율적으로 분리되어 있는지 표현 = 같은 군집 데이터 가깝게, 다른 군집 데이터는 멀리
→ 실루엣 계수(Silhouette Coefficient)



- a_{ij} 는 i 번째 데이터에서 자신이 속한 클러스터내의 다른 데이터 포인트까지의 거리. 즉 a_{12} 는 1번 데이터에서 2번 데이터까지의 거리
- $a(i)$ 는 i 번째 데이터에서 자신이 속한 클러스터내의 다른 데이터 포인트들의 평균 거리. 즉 $a(i) = \text{평균}(a_{12}, a_{13}, a_{14})$
- $b(i)$ 는 i 번째 데이터에서 가장 가까운 타 클러스터내의 다른 데이터 포인트들의 평균 거리. 즉 $b(i) = \text{평균}(b_{15}, b_{16}, b_{17}, b_{18})$

$$s(i) = \frac{b(i) - a(i)}{\max(a(i), b(i))}$$

$a(i)$: 같은 군집 内 다른 데이터 포인트와의 거리 평균

$b(i)$: 가장 가까운 다른 군집과의 평균 거리

$$-1 \leq s(i) \leq 1$$

≈ 0

→ 근처의 군집과 더 멀리 떨어져 있음 (good)

→ 근처의 군집과 가까움

→ 아예 다른 군집에 데이터 포인트가 할당

▷ 군집 別 평균 실루엣 계수의 시각화를 통한 군집 개수 최적화 방법

https://scikit-learn.org/stable/auto_examples/cluster/plot_kmeans_silhouette_analysis.html

The screenshot shows a web browser displaying the scikit-learn documentation. The URL in the address bar is https://scikit-learn.org/stable/auto_examples/cluster/plot_kmeans_silhouette_analysis.html. The page title is "Selecting the number of clusters with silhouette analysis on KMeans clustering". The content discusses how silhouette analysis can be used to study the separation distance between resulting clusters. It explains that silhouette coefficients near +1 indicate the sample is far from neighboring clusters, while 0 indicates it's on or very close to the decision boundary, and negative values indicate it might have been assigned to the wrong cluster. An example is given where silhouette analysis is used to choose an optimal value for `n_clusters`. The page includes two plots: one showing silhouette coefficient values for different clusters and another showing the visualization of clustered data points.

scikit-learn 1.4.1

Prev Up Next

Please cite us if you use the software.

Selecting the number of clusters with silhouette analysis on KMeans clustering

Selecting the number of clusters with silhouette analysis on KMeans clustering

Silhouette analysis can be used to study the separation distance between the resulting clusters. The silhouette plot displays a measure of how close each point in one cluster is to points in the neighboring clusters and thus provides a way to assess parameters like number of clusters visually. This measure has a range of [-1, 1].

Silhouette coefficients (as these values are referred to as) near +1 indicate that the sample is far away from the neighboring clusters. A value of 0 indicates that the sample is on or very close to the decision boundary between two neighboring clusters and negative values indicate that those samples might have been assigned to the wrong cluster.

In this example the silhouette analysis is used to choose an optimal value for `n_clusters`. The silhouette plot shows that the `n_clusters` value of 3, 5 and 6 are a bad pick for the given data due to the presence of clusters with below average silhouette scores and also due to wide fluctuations in the size of the silhouette plots. Silhouette analysis is more ambivalent in deciding between 2 and 4.

Also from the thickness of the silhouette plot the cluster size can be visualized. The silhouette plot for cluster 0 when `n_clusters` is equal to 2, is bigger in size owing to the grouping of the 3 sub clusters into one big cluster. However when the `n_clusters` is equal to 4, all the plots are more or less of similar thickness and hence are of similar sizes as can be also verified from the labelled scatter plot on the right.

Silhouette analysis for KMeans clustering on sample data with `n_clusters` = 2

The silhouette plot for the various clusters.

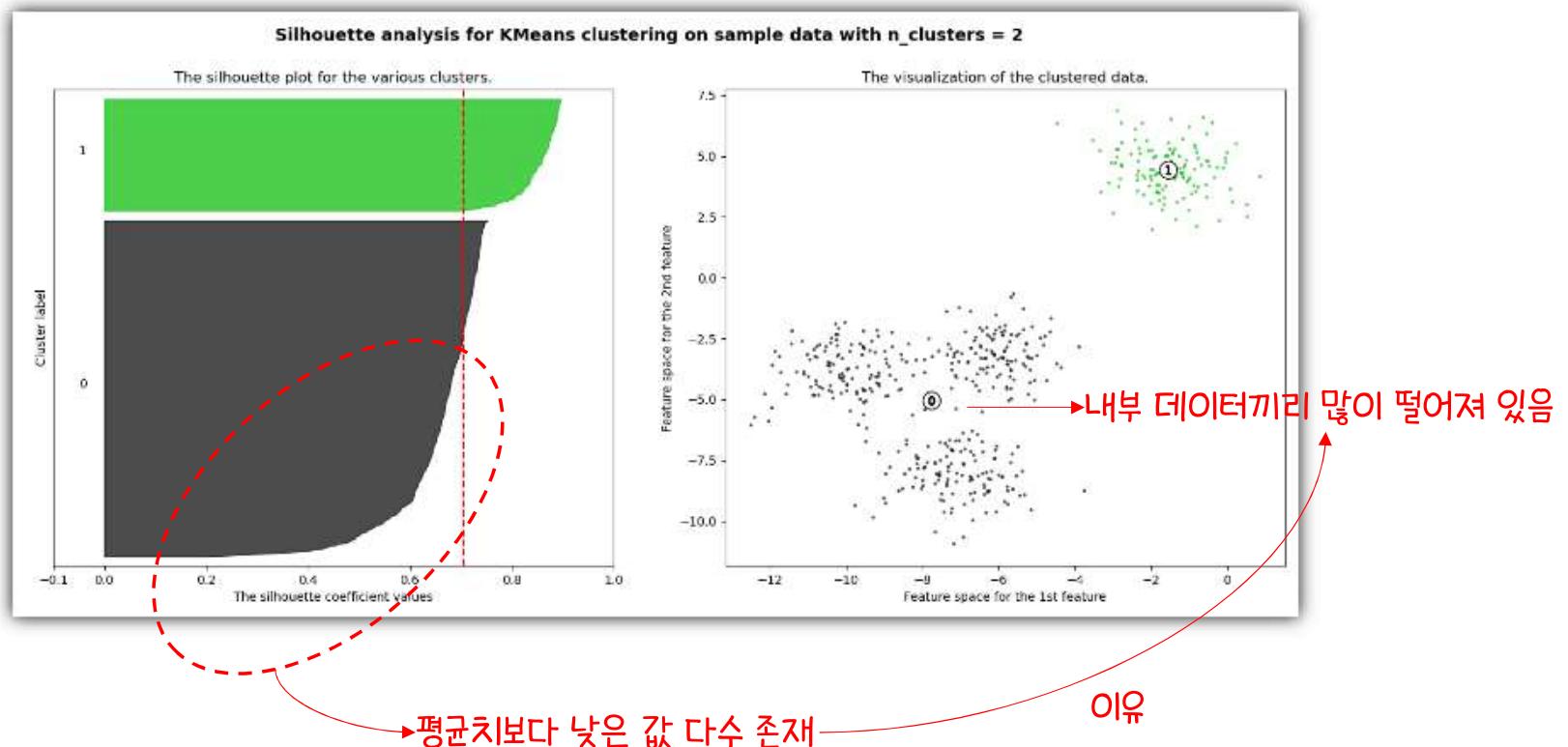
The visualization of the clustered data.

Toggle Menu

▷ 군집 別 평균 실루엣 계수의 시각화를 통한 군집 개수 최적화 방법

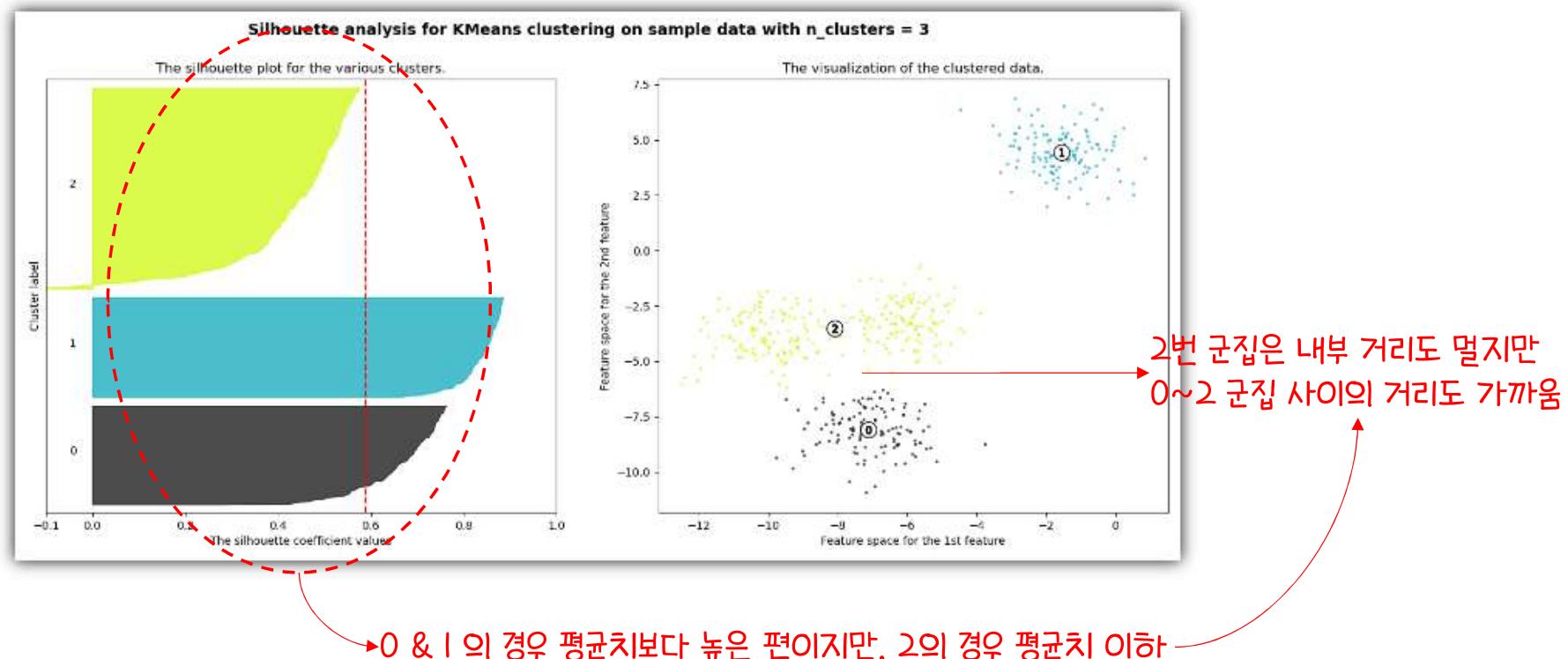
① 군집의 개수 = 2, 평균 실루엣 계수(silhouette_score) = 0.704

→ 매우 높은 값



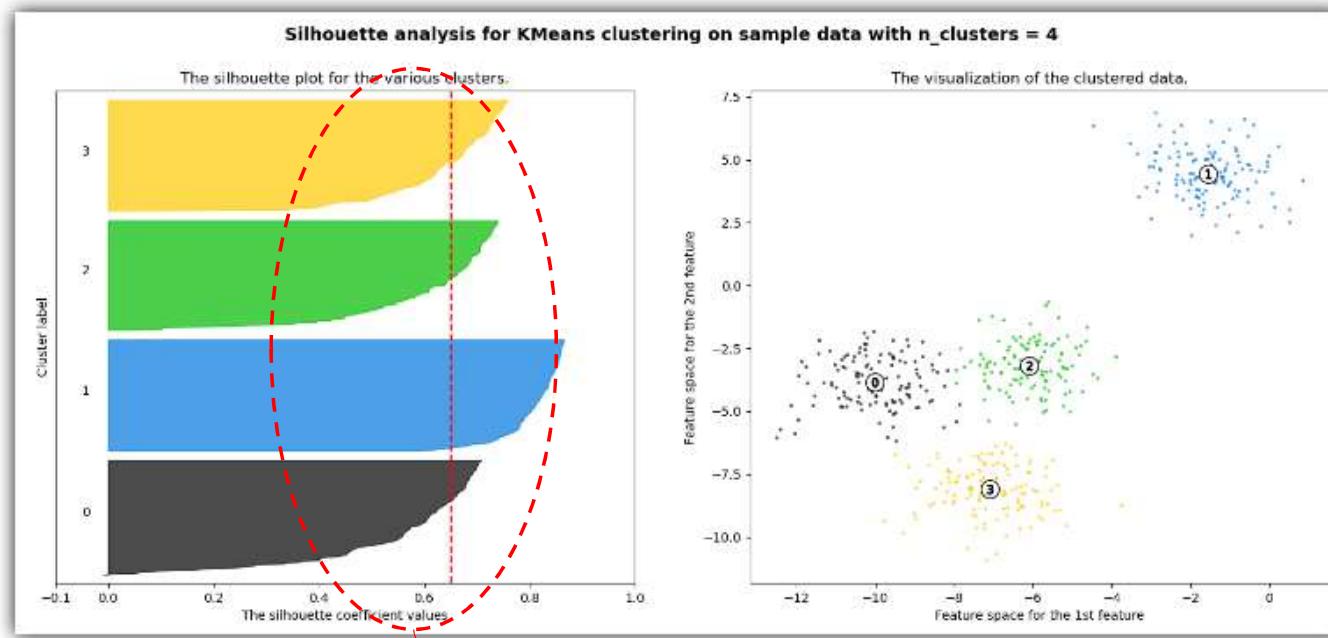
▷ 군집 別 평균 실루엣 계수의 시각화를 통한 군집 개수 최적화 방법

② 군집의 개수 = 3, 평균 실루엣 계수(silhouette_score) = 0.588



▷ 군집 別 평균 실루엣 계수의 시각화를 통한 군집 개수 최적화 방법

③ 군집의 개수 = 4, 평균 실루엣 계수(silhouette_score) = 0.65



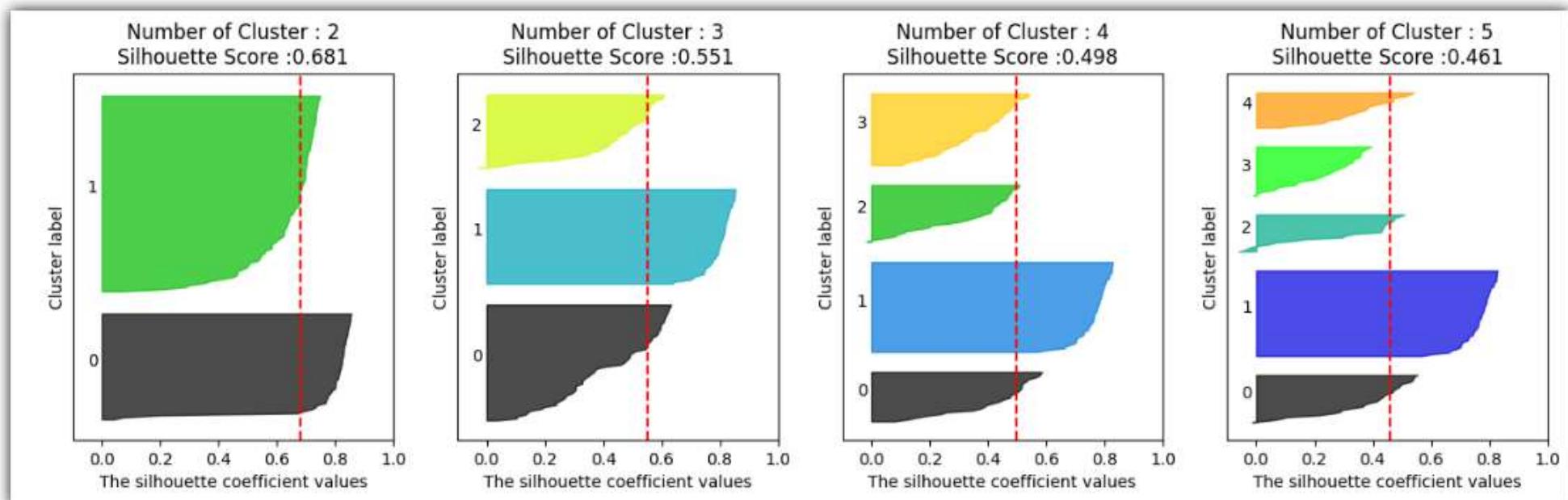
→ 비교적 균일

군집이 2개인 경우보다 평균 실루엣 계수 값이 작지만
4개인 경우가 가장 이상적인 군집화 개수로 판단 가능

▷ 좋은 군집화 조건

- ① 전체 실루엣 계수의 평균값(silhouette_score()값)은 0~1 사이의 값을 가지며, 1에 가까울수록 좋음
- ② 더불어 개별 군집의 평균값의 편차가 크지 않아야 함
= 개별 군집의 실루엣 계수 평균값이 전체 평균값에서 크게 벗어나지 않는 것
→ 만약, 전체 실루엣 계수 평균값은 높지만, 특정 군집에서 유난히 높고 다른 군집이 작아서 차이가 크면 좋은 군집화가 아님

https://github.com/whatwant-school/python-ml/blob/main/10-week/10-week_02-Evaluation.ipynb



[07-03]
평균 이동
(Mean Shift)

평균 이동 (Mean Shift)

중심을 데이터가 모여 있는 밀도 높은 곳으로 이동

K-평균 (K-mean)

중심에 소속된 데이터의 평균 거리 중심으로 이동

VS.

데이터의 분포도 이용

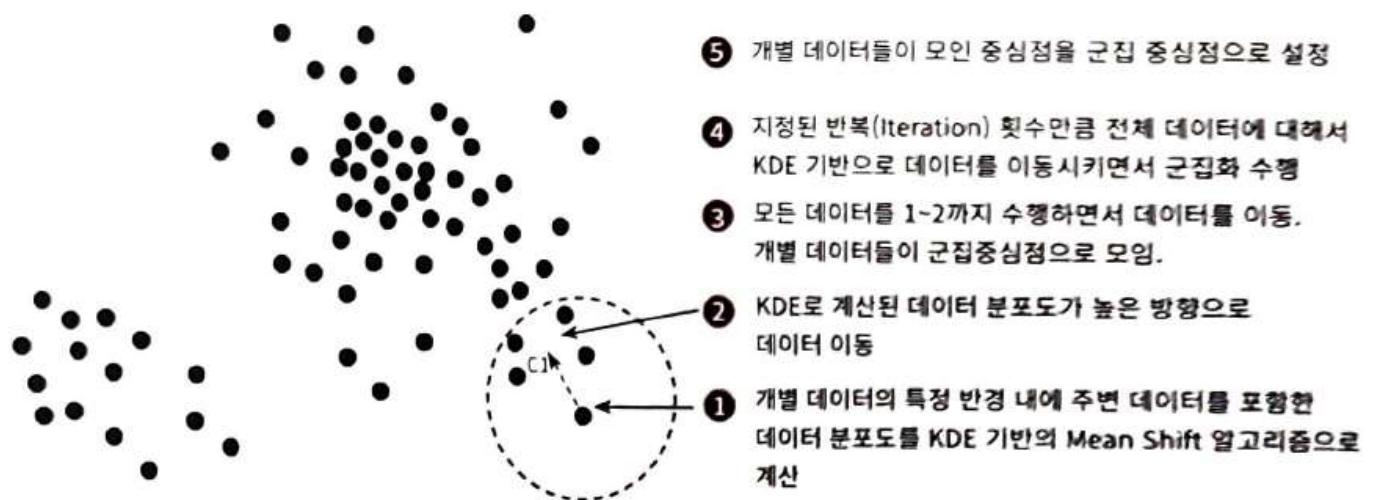
확률 밀도 함수 이용
(Probability Density Function)

KDE 이용
(Kernel Density Estimation)

특정 데이터를 반경 내의 데이터 분포 확률 밀도가 가장 높은 곳으로 이동

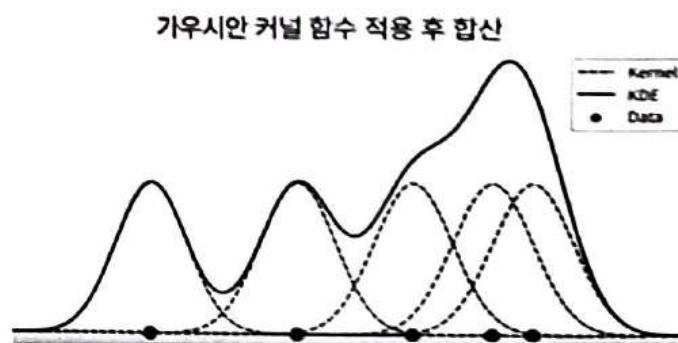
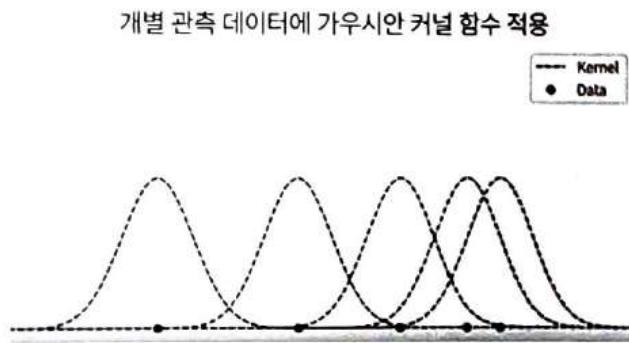
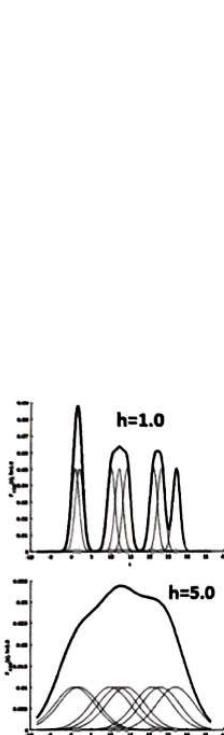
→ 주변 데이터와의 거리 값을 KDE 함수 값으로 입력한 뒤, 그 반환 값을 현재 위치에서 업데이트하면서 이동

→ 반복적으로 적용하면서 데이터의 군집 중심점 찾기



▷ KDE (Kernel Density Estimation)

- Kernel 함수를 통해 어떤 변수의 확률 밀도 함수(PDF, Probability Density Function)를 추정하는 대표적인 방법
- KDE는 개별 관측 데이터에 kernel 함수를 적용한 뒤, 모두 더한 후 개별 관측 데이터의 건수로 나눠 확률 밀도 함수 추정
 - . 대표적인 kernel 함수의 예 = 가우시안 분포 함수



$$KDE = \frac{1}{n} \sum_{i=1}^n K_h(x - x_i) = \frac{1}{nh} \sum_{i=1}^n K\left(\frac{x - x_i}{h}\right)$$

K : 커널 함수 (Kernel Function)
 x : 확률 변수 값 (Probability Variable)
 x_i : 관측값
 h : 대역폭 (bandwidth)

확률 밀도 추정 성능을 크게 좌우
작은 h 값(1.0)은 좁고 뾰족한 KDE,
큰 h 값(10)은 과도한 평활화(smoothing)

https://github.com/whatwant-school/python-ml/blob/main/10-week/10-week_03-MeanShift.ipynb

```
from sklearn.cluster import estimate_bandwidth

bandwidth = estimate_bandwidth(X)
print('bandwidth 값:', round(bandwidth, 3))
bandwidth 값: 1.877

import pandas as pd

clusterDF = pd.DataFrame(data=X, columns=['ftr1', 'ftr2'])
clusterDF['target'] = y

best_bandwidth = estimate_bandwidth(X)

meanshift = MeanShift(bandwidth=best_bandwidth)
cluster_labels = meanshift.fit_predict(X)
print('cluster labels 유형:', np.unique(cluster_labels))

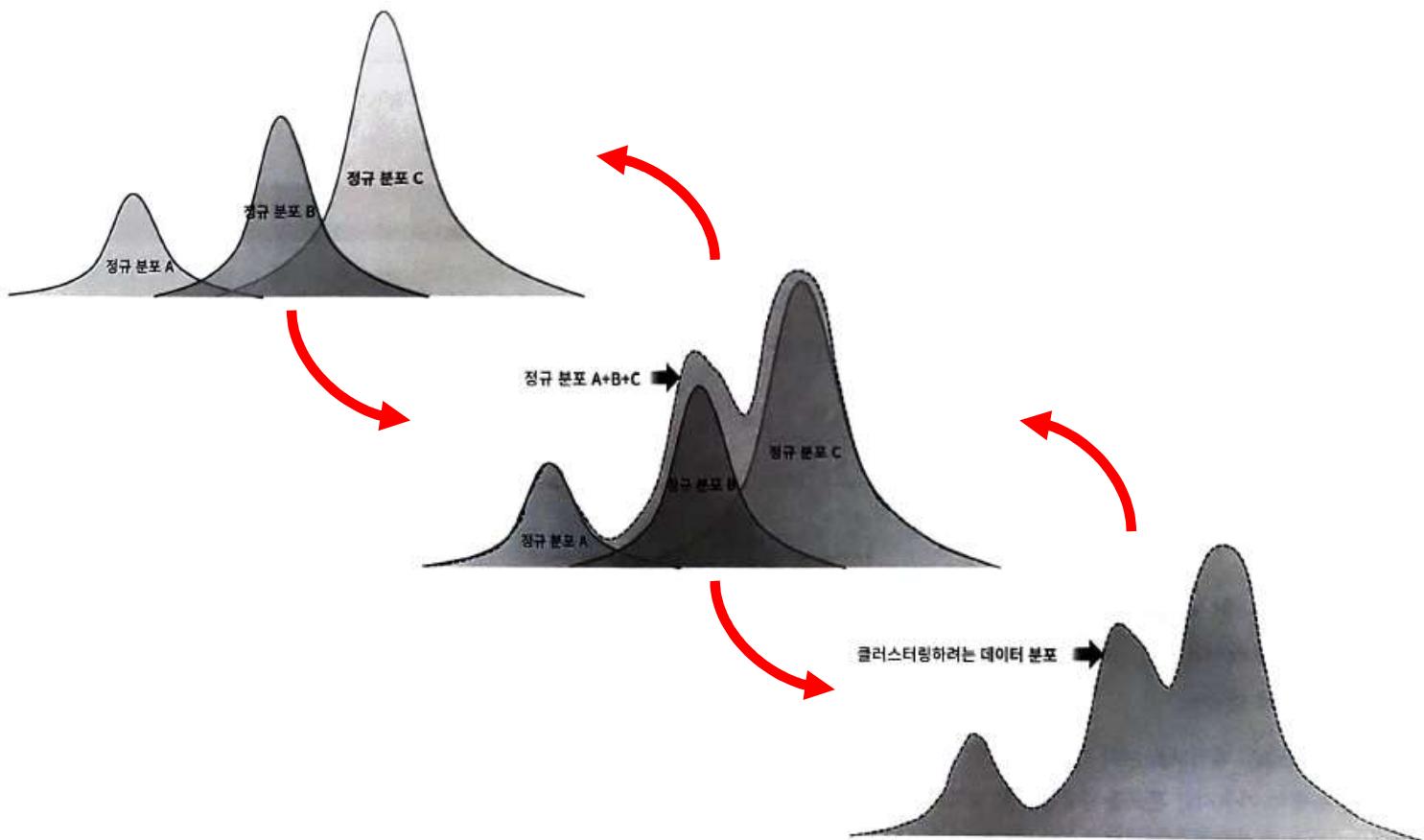
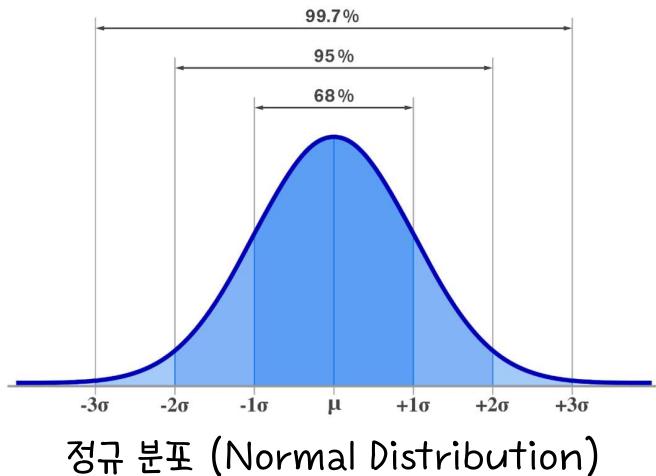
cluster labels 유형: [0 1 2]
```

[07-04]
GMM

(Gaussian Mixture Model)

▷ GMM (Gaussian Mixture Model)

- 군집화 하고자 하는 데이터가 여러 Gaussian Distribution을 가진 데이터 집합들이 섞여서 생성된 것이라는 가정
- 데이터 세트를 구성하는 여러 개의 정규 분포 곡선을 추출하고, 개별 데이터가 이 중 어느 정규 분포에 속하는지 결정
- 모수 추정



https://github.com/whatwant-school/python-ml/blob/main/10-week/10-week_04-GMM.ipynb

```
: print('### KMeans Clustering ###')
print(df.groupby('target')['kmeans_label'].value_counts())
print('\n### Gaussian Mixture Clustering ###')
print(df.groupby('target')['gmm_label'].value_counts())

### KMeans Clustering ###
target  kmeans_label
0        0            100
1        2            100
2        1            100
Name: count, dtype: int64

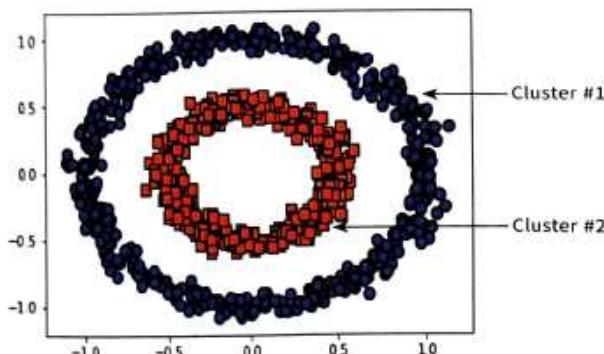
### Gaussian Mixture Clustering ###
target  gmm_label
0        0            100
1        2            100
2        1            100
Name: count, dtype: int64
```

[07-05]
DBSCAN

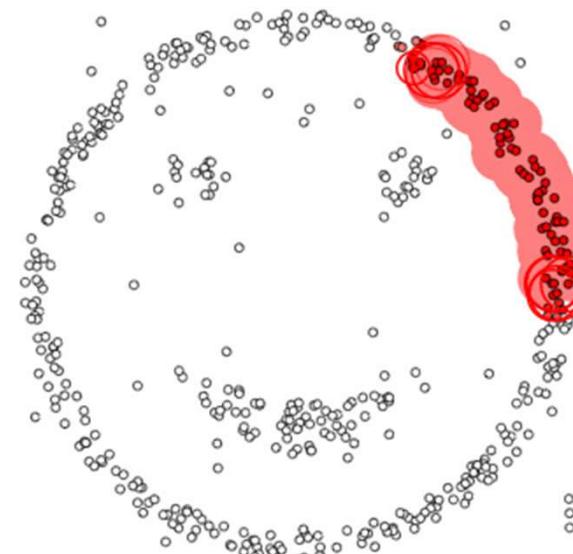
(Density Based Spatial Clustering
of Applications with Noise)

▷ DBSCAN (Density Based Spatial Clustering of Applications with Noise)

- 특정 공간 내에 데이터 밀도 차이 기반 알고리즘 → 복잡한 기하학적 분포도를 갖는 데이터 세트에 대해서도 잘 동작



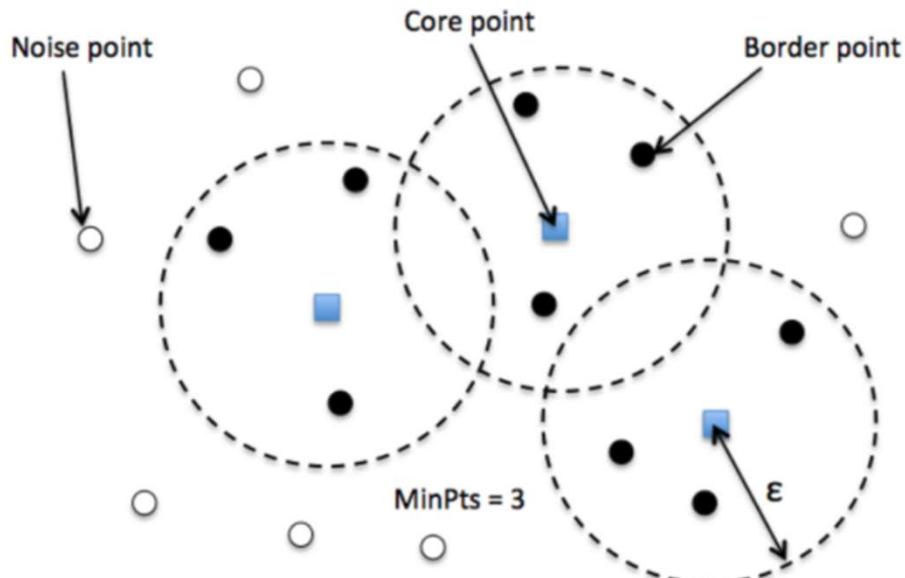
epsilon = 1.00
minPoints = 4



Restart



Pause



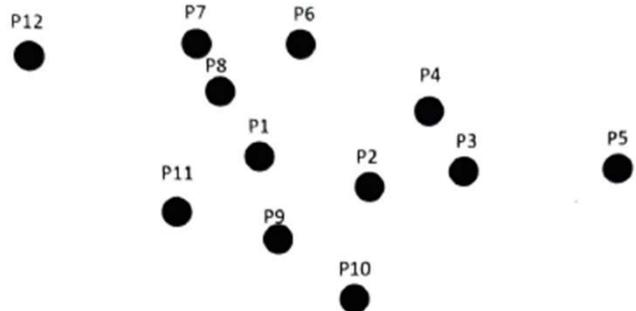
ε (epsilon) : 개별 데이터를 중심으로 입실론 반경을 가지는 원형의 영역

min points : 개별 데이터의 입실론 주변 영역에 포함되는 다른 데이터 개수

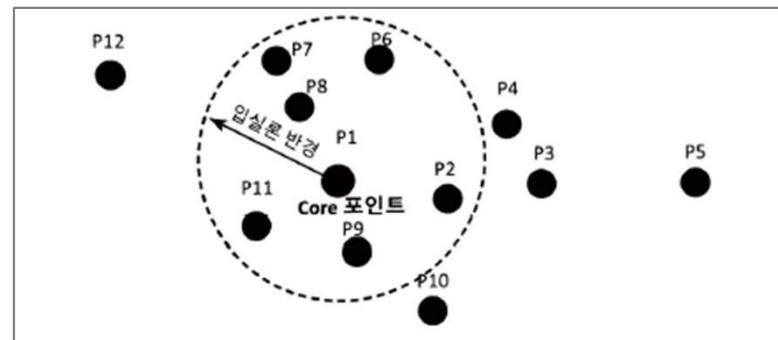
core point : 주변 영역에서 min points 이상의 다른 데이터를 갖고 있을 경우 해당 데이터

neighbor point : 주변 영역 내에 위치한 다른 데이터

noise point : min points 개수 이상의 neighbor point를 가지고 있지 않으며, core/neighbor point 모두 없는 데이터

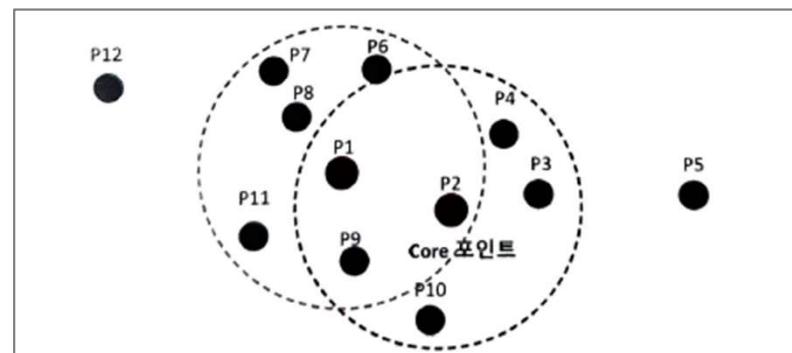


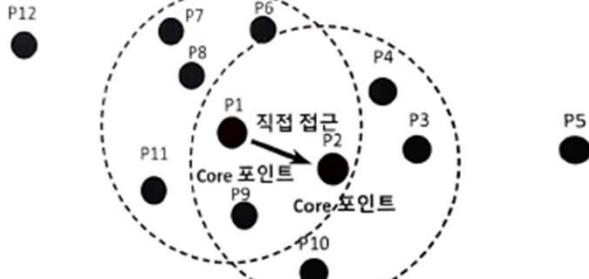
① min point = 6 (자기 자신의 데이터를 포함)



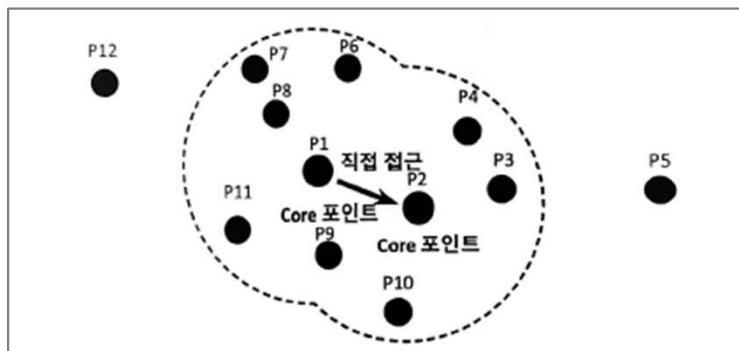
② P1 데이터를 기준으로 ε 반경 내에
포함된 데이터가 7개 $\rightarrow P1 = \text{core point}$

③ P2 데이터를 기준으로 ε 반경 내에
포함된 데이터가 6개 $\rightarrow P2 = \text{core point}$



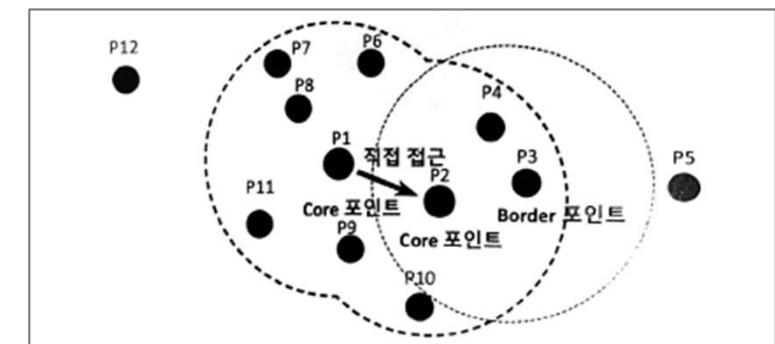


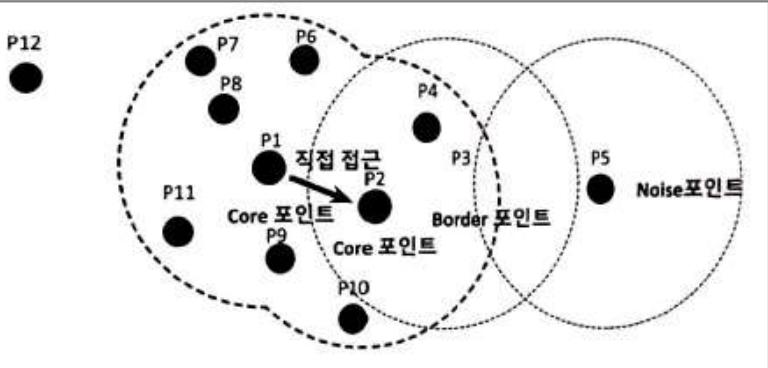
④ core point P1의 neighbor point P2 역시 core point
→ P1에서 P2로 연결해 직접 접근 가능



⑤ 특정 core point에서 직접 접근이 가능한 다른 core point를 서로 연결하면서 군집화 구성
→ 이렇게 점차 군집 영역 확장 = DBSCAN 군집화

⑥ P3 데이터의 경우 반경 내에 포함되는 neighbor point는 P2, P4 2개이므로 군집으로 구분할 수 있는 core point 아님
→ core point는 아니지만 neighbor point 중에 core point를 갖고 있는 경우 = border point





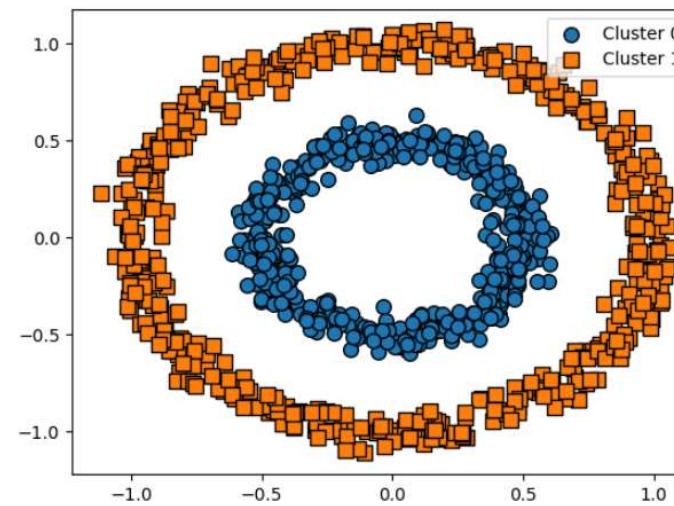
⑦ P5와 같이 반경 내에 min points를 갖고 있지도 않고 neighbor point 중에 core point를 갖고 있지도 않는 데이터
→ noise point

https://github.com/whatwant-school/python-ml/blob/main/10-week/10-week_05-DBSCAN.ipynb

```
from sklearn.cluster import DBSCAN

dbscan = DBSCAN(eps=0.2, min_samples=10, metric='euclidean')
dbscan_labels = dbscan.fit_predict(X)
df['dbscan_cluster'] = dbscan_labels

visualize_cluster_plot(dbscan, df, 'dbscan_cluster', iscenter=False)
```

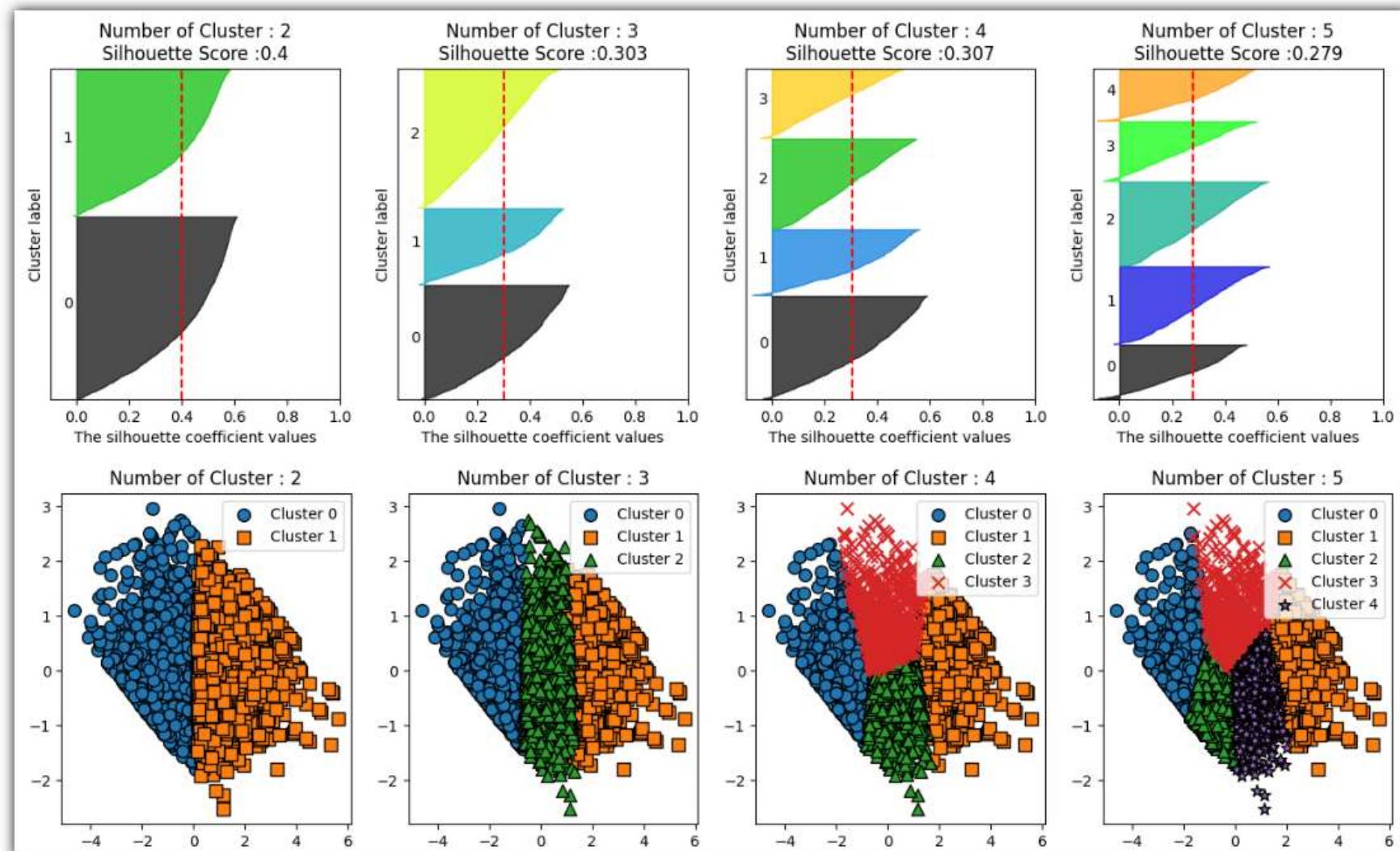


[07-06]
군집화 실습
- 고객 세그먼테이션
(Customer Segmentation)

<http://archive.ics.uci.edu/dataset/352/online+retail>

The screenshot shows the UC Irvine Machine Learning Repository dataset page for 'Online Retail'. The page has a header with the repository logo, navigation links for Datasets, Contribute Dataset, and About Us, a search bar, and a login button. The main content area features a blue header for the 'Online Retail' dataset, which was donated on 11/5/2015. Below this, a text summary states: 'This is a transnational data set which contains all the transactions occurring between 01/12/2010 and 09/12/2011 for a UK-based and registered non-store online retail.' To the right of the summary are four buttons: 'DOWNLOAD', 'IMPORT IN PYTHON', and 'CITE' (highlighted in yellow), along with metrics for '8 citations' and '145329 views'. The 'Dataset Characteristics' section includes 'Subject Area' (Business), 'Associated Tasks' (Classification, Clustering), 'Feature Type' (Integer, Real), '# Instances' (541909), and '# Features' (6). The 'Dataset Information' section contains an 'Additional Information' block with a detailed description of the dataset's purpose and characteristics. It also includes a 'Has Missing Values?' section indicating 'No'. The 'Introductory Paper' section lists a paper titled 'Data mining for the online retail industry: A case study of RFM model-based customer segmentation using data mining' by Daqing Chen, Sai Laing Sain, Kun Guo, published in 2012 in the Journal of Database Marketing and Customer Strategy Management, Vol. 19, No. 3. The 'Creators' section lists Daqing Chen from the School of Engineering, London South Bank University. The 'DOI' section provides the identifier 10.24432/C5BW33. The 'License' section notes that the dataset is licensed under a Creative Commons Attribution 4.0 International (CC BY 4.0) license, allowing sharing and adaptation.

https://github.com/whatwant-school/python-ml/blob/main/10-week/10-week_06-Retail.ipynb



You've really worked hard today

Next Week ~ ?

08**텍스트 분석**

| | |
|---|-----|
| NLP이나 텍스트 분석이냐? | 488 |
| 01. 텍스트 분석 이해 | 489 |
| 텍스트 분석 수행 프로세스 | 490 |
| 파이썬 기반의 NLP, 텍스트 분석 패키지 | 490 |
| 02. 텍스트 사전 준비 작업(텍스트 전처리) – 텍스트 정규화 | 491 |
| 클렌징 | 492 |
| 텍스트 토큰화 | 492 |
| 스톱 워드 제거 | 495 |
| Stemming과 Lemmatization | 496 |
| 03. Bag of Words – BOW | 498 |
| BOW 피처 벡터화 | 499 |
| 사이킷런의 Count 및 TF-IDF 벡터화 구현: CountVectorizer, TfidfVectorizer | 501 |
| BOW 벡터화를 위한 최소 행렬 | 503 |
| 최소 행렬 – COO 형식 | 504 |
| 최소 행렬 – CSR 형식 | 505 |
| 04. 텍스트 분류 실습 – 20 뉴스그룹 분류 | 509 |
| 텍스트 정규화 | 509 |
| 피처 벡터화 변환과 머신러닝 모델 학습/예측/평가 | 512 |
| 사이킷런 파이프라인(Pipeline) 사용 및 GridSearchCV와의 결합 | 516 |
| 05. 감성 분석 | 519 |
| 감성 분석 소개 | 519 |
| 지도학습 기반 감성 분석 실습 – IMDB 영화평 | 519 |
| 비지도학습 기반 감성 분석 소개 | 523 |
| SentiWordNet을 이용한 감성 분석 | 525 |
| VADER를 이용한 감성 분석 | 532 |

| | |
|---|-----|
| 06. 토픽 모델링(Topic Modeling) – 20 뉴스그룹 | 534 |
| 07. 문서 군집화 소개와 실습(Opinion Review 데이터 세트) | 538 |
| 문서 군집화 개념 | 538 |
| Opinion Review 데이터 세트를 이용한 문서 군집화 수행하기 | 538 |
| 군집별 핵심 단어 추출하기 | 546 |
| 08. 문서 유사도 | 550 |
| 문서 유사도 측정 방법 – 코사인 유사도 | 550 |
| 두 벡터 사이각 | 550 |
| Opinion Review 데이터 세트를 이용한 문서 유사도 측정 | 554 |
| 09. 한글 텍스트 처리 – 네이버 영화 평점 감성 분석 | 558 |
| 한글 NLP 처리의 어려움 | 558 |
| KoNLPy 소개 | 559 |
| 데이터 로딩 | 562 |
| 10. 텍스트 분석 실습 – 캐글 Mercari Price Suggestion Challenge | 566 |
| 데이터 전처리 | 567 |
| 피처 인코딩과 피처 벡터화 | 573 |
| 릿지 회귀 모델 구축 및 평가 | 578 |
| LightGBM 회귀 모델 구축과 앙상블을 이용한 최종 예측 평가 | 581 |
| 11. 정리 | 582 |

09**추천 시스템**

| | |
|------------------------|-----|
| 01. 추천 시스템의 개요와 배경 | 584 |
| 추천 시스템의 개요 | 584 |
| 온라인 스토어의 필수 요소, 추천 시스템 | 586 |
| 추천 시스템의 유형 | 587 |

Who ~ ?

See you Next Weekend ~ ?