BDA x 이지스 퍼블리싱 머신러닝 스터디 **4**회차 과제

조 이름 : 8조

<1번 문제**>**

k-평균 군집화 모델의 평가지표인 Rand지수와 실루엣 계수에 대해 설명해보시오

Rand지수 :

sklearn.metrics.**adjusted_rand_score**(*labels_true*, *labels_pred*)

Rand지수는 확률에 따라 조정된다. Rand 지수는 예측된 군집과 실제 군집에서 같거나 다른 군집에 할당된 모든 표본 쌍과 계수쌍을 고려하여 두 군집 간의 유사성 측도를 계산한다.

RI score을 이용하여 ARI은 다음과 같은 식으로 나온다.
ARI = (RI - Expected_RI) / (max(RI) - Expected_RI)

Rand지수는군집 및 표본 수와 무관하게 랜덤 라벨링의 경우 0.0에 가까운 값을 가지며 군집이 동일한 경우 정확하게 1.0을 갖도록 한다. Rand지수는 불일치 군집화의 경우 -0.5 아래로 제한된다.

ARI는 대칭 측도이다.

adjusted_rand_score(a, b) == adjusted_rand_score(b, a)

실루엣 계수 :
실루엣 분석은 각 군집 간의 거리가 얼마나 효율적으로 분리돼 있는지를 나타낸다. 효율적으로 잘 분리됐다는 것은 다른 군집과의 거리는 떨어져 있고 동일 군집끼리의 데이터는 서로 가깝게 잘 뭉쳐 있다는 의미이다. 군집화가 잘 될수록 개별 군집은 비슷한 정도의 여유공간을 가지고 떨어져 있을 것이다.

실루엣 분석은 실루엣 계수(silhouette coefficient)를 기반으로 합니다. 실루엣 계수는 개별 데이터가가지는 군집화 지표이다. 개별 데이터가 가지는 실루엣 계수는 해당 데이터가 같은 군집 내의 데이터와 얼마나 가깝게 군집화돼 있고, 다른 군집에 있는 데이터와는 얼마나 멀리 분리돼 있는지를 나타내는 지표이다.

사이킷런은 이러한 실루엣 분석을 위해 다음과 같은 메서드를 제공한다.

• sklearn.metrics.silhouette_samples(X, labels, metric='euclidean', #kwds): 인자로 x leature 데이터 세트와 각피처 데이터 세트가 속한 군집 레이블 값인 labels 데이터를 입력해주면 각 데이터 포인트의 실루엣 계수를 계산해 반환한다.

• sklearn.metrics.silhouette_score(X, labels, metric='euclidean', sample size=None, **kwls): 인자로 X feature 데이터 세트와 각 피처 데이터 세트가 속한 군집 레이블 값인 labels 데이터를 입력해주면 전체 데이터의 실루엣계수 값을 평균해 반환한다.

즉, np.mean(silhouette_samples())이다. 일반적으로 이 값이 높을수록 군집화가 어느정도 잘 됐다고 판단할 수 있다. 하지만 무조건 이 값이 높다고 해서 군집화가 잘 됐다고 판단할 수는 없다.

좋은 군집화가 되려면 다음 기준 조건을 만족해야 한다.

1. 전체 실루엣 계수의 평균값, 즉 사이킷런의 silhouette_score( ) 값은 0~1 사이의 값을 가지며, 1에 가까울수록 좋다.

2. 하지만 전체 실루엣 계수의 평균값과 더불어 개별 군집의 평균값의 편차가 크지 않아야 합니다. 즉, 개별 군집의 실루엣 계수 평균값이 전체 실루엣 계수의 평균값에서 크게 벗어나지 않는 것이 중요하다. 만약 전체 실루엣 계수의 평균값은 높지만, 특정 군집의 실루엣 계수 평균값만 유난히 높고 다른 군집들의 실루엣 계수 평균값은 낮으면 좋은 군집화 조건이 아니다.

<2번 문제**>**

2. K-최근접 이웃 모델과 K-평균 군집화 모델의 차이를 서술하고 각 모델이 활용될 수 있는 사례를 적어보시오

K-최근접 이웃 모델 :

- KNN은 지도학습(supervised learning) 알고리즘으로, 분류(Classification)와 회귀(Regression)에 사용
- 주어진 데이터의 레이블을 예측하기 위해 이웃한 데이터 포인트들을 사용
- 주로 분류 문제에서 어떤 클래스에 속하는지 예측하거나, 회귀 문제에서 수치 값을 예측하는 데 사용

K-평균 군집화 모델 :

- K-평균은 비지도학습(unsupervised learning) 알고리즘으로, 데이터를 K개의 클러스터로 그룹화함
- 레이블이 주어지지 않은 데이터에서 유사한 패턴을 찾음
- 주로 패턴을 찾고 데이터를 그룹으로 분류하는 데 사용

K-최근접 이웃 모델의 활용 사례:

- 이미지 분류: 손글씨 숫자나 얼굴 인식과 같은 이미지 분류 문제에 사용될 수 있음
- 추천 시스템: 사용자의 행동 패턴에 기반하여 비슷한 사용자나 항목을 찾아서 추천하는 데 활용

K-평균 군집화 모델의 활용 사례 :

- 고객 세분화: 고객 데이터를 사용하여 비슷한 특성을 가진 그룹으로 세분화하여 마케팅 전략을 계획할 때 사용
- 이상치 탐지: 데이터셋에서 일반적인 패턴을 찾고, 그에 벗어난 데이터를 이상치로 간주할 때 활용

<3번 문제**>**

- 단일연결법
  - 두 군집 간 원소끼리의 거리를 모두 비교한 후 그중 최소거리를 군집 간 거리로 정의하는 방법
  - 군집 간의 연결이 뾰족한 형태를 가져 이상치에 민감할 수 있다
- 최장연결법
  - 두 군집 간 원소끼리의 거리를 모두 비교한 후 그중 최대 거리를 군집 간 거리로 정의하는 방법
- 평균연결법
  - 두 군집 간 원소끼리의 거리를 모두 비교한 후 그 평균 거리를 군집 간 거리로 정의하는 방법
  - 이상치에 비교적 강인하며, 균형잡힌 군집을 형성하는 경향이 있다
- 중심연결법
  - 각 군집의 중심을 구한 후 중심 사이의 거리를 군집의 거리로 정의하는 방법
  - 각 군집의 중심은 군집 내 원소의 무게 중심을 정의한다
  - 이상치에 민감할 수 있고, 군집이 불균형한 크기를 가질 때 영향을 받을 수 있다
- 와드연결법
  - 두 군집을 병합했을 때 군집 내 분산의 증가분을 두 군집 사이의 거리로 정의하는 방법
  - 군집 내 분산은 각 군집을 그 군집의 중심으로만 근사했을 때 발생하는 정보의 손실로 해석

<4번 문제>

```
import numpy as np
from matplotlib import pyplot as plt
from scipy.cluster.hierarchy import dendrogram
from sklearn.datasets import load_iris
from sklearn.cluster import AgglomerativeClustering
```

필요한 라이브러리를 가져오기

- 수치 연산을 위한 numpy
- 플로팅을 위한 matplotlib
- 계층적 클러스터링을 위한 scipy.cluster.hierarchy
- 아이리스 데이터셋 및 AgglomerativeClustering 모델을 위한 sklearn

```
X = load_iris().data
```

아이리스 데이터셋을 로드하고 특성 데이터(X)를 추출. 이는 각 아이리스 샘플에 대한 꽃받침 길이, 꽃받침 너비, 꽃잎 길이 및 꽃잎 너비의 측정값이다.

```
model = AgglomerativeClustering(distance_threshold=0,n_clusters=None)
model = model.fit(X)
```

- AgglomerativeClustering 모델의 인스턴스를 생성
- distance_threshold=0는 알고리즘이 모든 쌍의 거리가 지정된 임계값보다 크기 전까지 클러스터링을 진행해야 함을 나타낸다.
- n_clusters=None은 모든 클러스터가 형성될 때까지 알고리즘이 멈추지 않아야 함을 의미
- 모델을 아이리스 데이터셋에 맞춘다.

```
counts = np.zeros(model.children_.shape[0])
n_samples = len(model.labels_)
```

각 클러스터에 포함된 포인트 수를 저장할 배열(counts)을 초기화. n_samples는 전체 데이터 포인트 수이다.

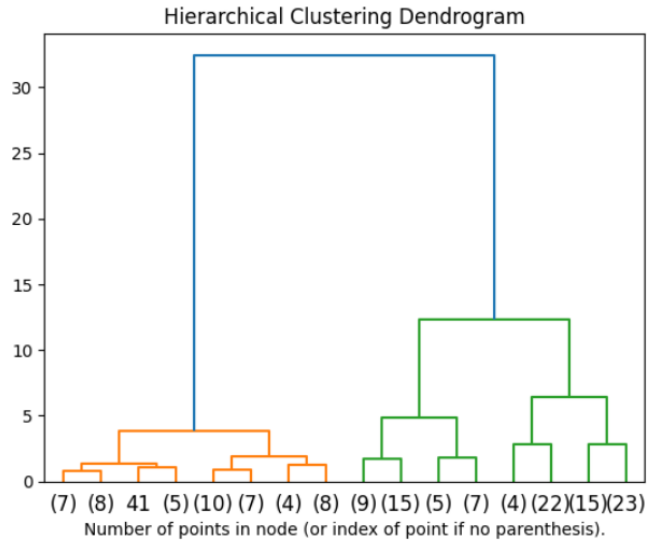
```
for i, merge in enumerate(model.children_):
    current_count = 0
    for child_idx in merge:
        if child_idx < n_samples:
            current_count += 1
        else:
            current_count += counts[child_idx - n_samples]
    counts[i]=current_count
```

계층적 클러스터링 도중 형성된 머지(클러스터)를 반복적으로 검사한다. 각 머지에 대해 포인트 수를 세기 위해 자식 인덱스가 n_samples보다 작은지(개별 데이터 포인트) 또는 이전에 형성된 클러스터인지 확인하고 적절한 카운트를 더한다.

```
linkage_matrix = np.column_stack([model.children_,model.distances_,counts]).astype(float)
```

자식 인덱스, 거리 및 카운트를 수평으로 쌓아서 링크 메트릭스를 생성

```
dendrogram(linkage_matrix,truncate_mode="level",p=3)
plt.title("Hierarchical Clustering Dendrogram")
plt.xlabel("Number of points in node (or index of point if no parenthesis).")
plt.show()
```



생성된 링크 메트릭스를 사용하여 덴드로그램 그리기.

- truncate_mode="level"은 덴드로그램을 특정 수준까지만 자르도록 지정
- p=3은 세 번째 레벨에서 자르도록 지정
- 플롯에 제목과 축 레이블을 추가하고 그림을 표시

<5번 문제**>**

PCA 모델과 릿지 회귀의 관련성은 주로 특잇값 분해(Singular Value Decomposition, SVD)와 회귀 모델의 해를 통한 차원 축소와 정규화의 측면에서 나타납니다.

1. **릿지 회귀의 유일해와 특잇값 분해 식:**
   - 릿지 회귀에서 회귀 계수 벡터 β는 다음과 같이 표현됩니다.
     $\beta = (X^T X + \alpha I)^{-1} X^T y$
     여기서, X는 독립 변수 행렬, α는 릿지 회귀의 정규화 파라미터, I는 단위 행렬입니다.
   - 위의 식은 특잇값 분해를 사용하여 다시 표현될 수 있습니다.
     $\beta = V \cdot D \cdot U^T \cdot y$
     여기서 U, D, V는 X의 특잇값 분해 행렬입니다.
2. **PCA와 차원 축소:**
   - PCA에서 데이터의 주성분을 찾기 위해 특잇값 분해가 사용됩니다. 데이터 행렬 X를 $U \cdot D \cdot V^T$로 분해하면, U의 열벡터는 주성분을 나타냅니다.
   - PCA에서 주성분은 데이터의 분산을 최대화하는 방향이므로, 특잇값 분해의 U 행렬이 중요합니다.
3. **최소제곱법 모델의 해와 PCA:**
   - 최소제곱법에서 회귀 계수 벡터 β는 다음과 같이 표현됩니다.
     $\beta = (X^T X)^{-1} X^T y$
   - 이는 특잇값 분해를 사용하여 다시 표현될 수 있습니다.
     $\beta = V \cdot D^{-1} \cdot U^T \cdot y$
   - 최소제곱법에서 해를 구하는 것은 데이터의 주성분을 찾는 것과 관련이 있습니다. 주성분을 찾아 차원을 줄이면서 데이터의 분산을 최대화하기 때문입니다.

요약하면, 릿지 회귀의 유일해와 특잇값 분해 식, 그리고 최소제곱법 모델의 해와 PCA는 모두 선형 대수와 통계의 관점에서 서로 연결되어 있습니다. 특잇값 분해는 차원 축소와 모델의 안정성을 향상시키는 데에 활용되며, 주성분은 데이터의 중요한 특성을 나타내는 데에 사용됩니다.

<6번 문제**>**

1. MDS (Multi-Dimensional Scaling):
  - 특징: MDS는 데이터 간의 거리를 보존하는 방식으로 차원을 축소합니다. 주로 유클리드 거리를 기반으로 하지만 다른 거리 메트릭을 사용할 수도 있습니다.
  - 동작 원리: 데이터 간의 거리 행렬을 계산한 후, 이 거리 행렬을 보존하면서 저차원으로 매핑합니다.
  - 적용 분야: MDS는 데이터의 전반적인 구조를 보존하면서 차원을 축소하므로 데이터 시각화나 유사성 분석에 사용됩니다.

2. Isomap (Isometric Mapping):
  - 특징: Isomap은 데이터 간의 지오메트릭 구조를 보존하려는 목적으로 개발되었습니다. 이웃 간의 지오메트릭 거리를 보존하여 저차원에 매핑합니다.
  - 동작 원리: 각 데이터 포인트의 근접 이웃을 찾아 거리 행렬을 만들고, 이 거리 행렬을 확장하여 지오메트릭 거리를 계산한 후, 저차원으로 매핑합니다.
  - 적용 분야:고차원 데이터의 지오메트릭 구조를 유지하면서 차원을 축소하므로, 자연 이미지, 생물학적 데이터 등에 적용됩니다.

3. LLE (Locally Linear Embedding):
  - 특징: LLE는 각 데이터 포인트를 그 근처 이웃들과 선형 관계로 표현하여 차원을 축소합니다. 이웃 간의 지역적인 선형 구조를 보존합니다.
  - 동작 원리: 각 데이터 포인트를 이웃들과의 선형 관계로 근사화하며, 이 선형 관계를 보존하는 식으로 저차원에 매핑합니다.
  - 적용 분야: 고차원 데이터에서 지역적인 구조를 유지하면서 차원을 축소하므로, 얼굴 인식, 음성 분류 등에 사용됩니다.

4. t-SNE (t-Distributed Stochastic Neighbor Embedding):
  - 특징: t-SNE는 고차원에서의 데이터 간의 유사성을 보존하면서 차원을 축소하는데 중점을 둡니다. 특히, 군집 간의 거리를 잘 보존합니다.
  - 동작 원리: 고차원 데이터 포인트 간의 유사성을 확률 분포로 표현하고, 저차원에서도 유사성을 보존하도록 합니다.
  - 적용 분야: 군집 간의 구조를 잘 보존하면서 시각적으로 효과적인 표현을 제공하므로, 시각적 데이터 분석이나 클러스터링에 주로 사용됩니다.