실루엣 계수 (Silhouette Coefficient)

- 군집화(Clustering) 알고리즘의 성능을 평가하는 지표
- 각 데이터 포인트가 해당 군집 내에 얼마나 잘 군집화되어 있는지, 그리고 다른 군집과는 얼마나 잘 분리되어 있는지를 측정
- 값의 범위는 -1에서 1까지이며, 1에 가까울수록 군집화가 잘 되었다고 판단
  - **1에 가까울수록**: 해당 데이터 포인트가 자신의 군집에 잘 속해 있고, 다른 군집과는 멀리 떨어져 있음을 의미합니다.
  - **0에 가까울수록**: 해당 데이터 포인트가 군집 경계에 위치하여, 두 군집 사이에 모호하게 걸쳐 있음을 의미합니다.
  - **-1에 가까울수록**: 해당 데이터 포인트가 잘못된 군집에 할당되었음을 의미합니다.
- 실루엣 계수 수식 : $s = (b - a) / max(a, b)$
  - a: 해당 데이터 포인트와 동일한 군집 내 다른 데이터 포인트들 간의 평균 거리 (응집도)
  - b: 해당 데이터 포인트와 가장 가까운 다른 군집 내 데이터 포인트들 간의 평균 거리 (분리도)

적용 가능한 상황

- 군집화 알고리즘의 최적 군집 개수(K)를 결정할 때 (가장 높은 실루엣 계수를 보이는 K를 선택).
- 서로 다른 군집화 알고리즘의 성능을 비교할 때.
- 군집화 결과의 품질을 정량적으로 평가하고 싶을 때.

주의사항

- 실루엣 계수는 유클리드 거리 기반의 군집화 알고리즘(예: K-Means)에 주로 사용됩니다. 거리 측정 방식에 따라 결과가 달라질 수 있습니다.
- 데이터셋의 크기가 매우 클 경우, 모든 데이터 포인트에 대한 거리를 계산하는 데 시간이 오래 걸릴 수 있습니다.
- 볼록한 형태(convex shape)의 군집에는 잘 작동하지만, 복잡한 형태의 군집에는 적합하지 않을 수 있습니다.

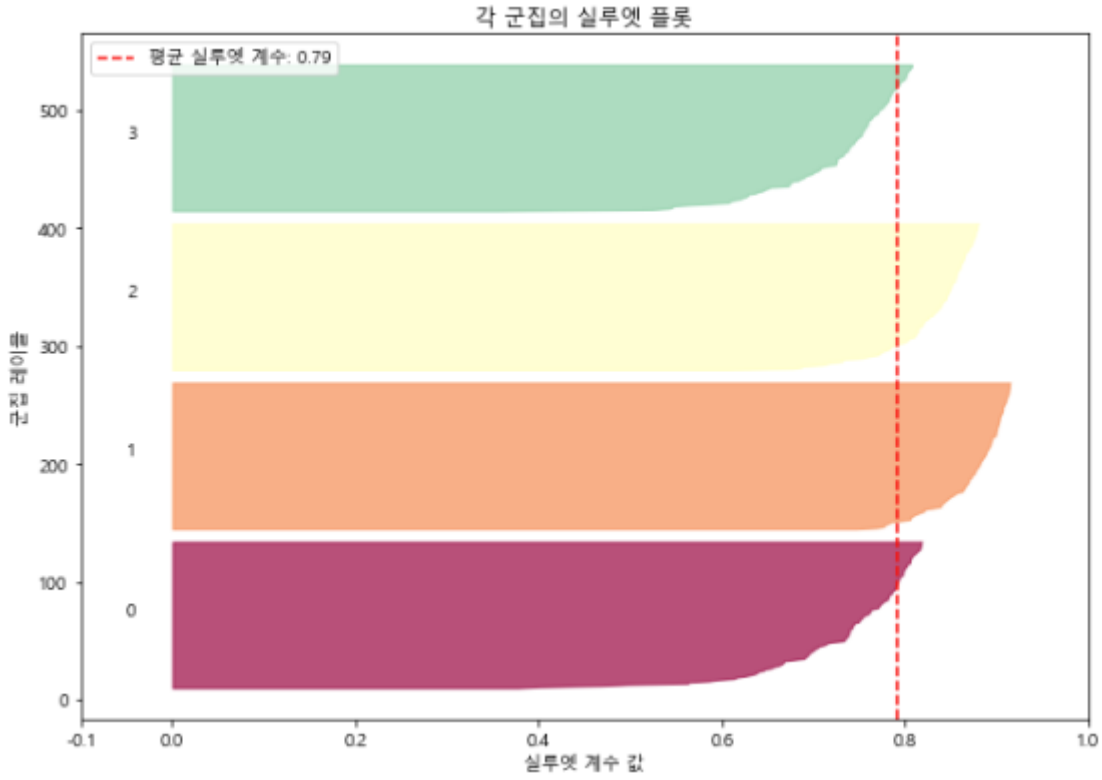
파라미터 종류 (silhouette_score, silhouette_samples)

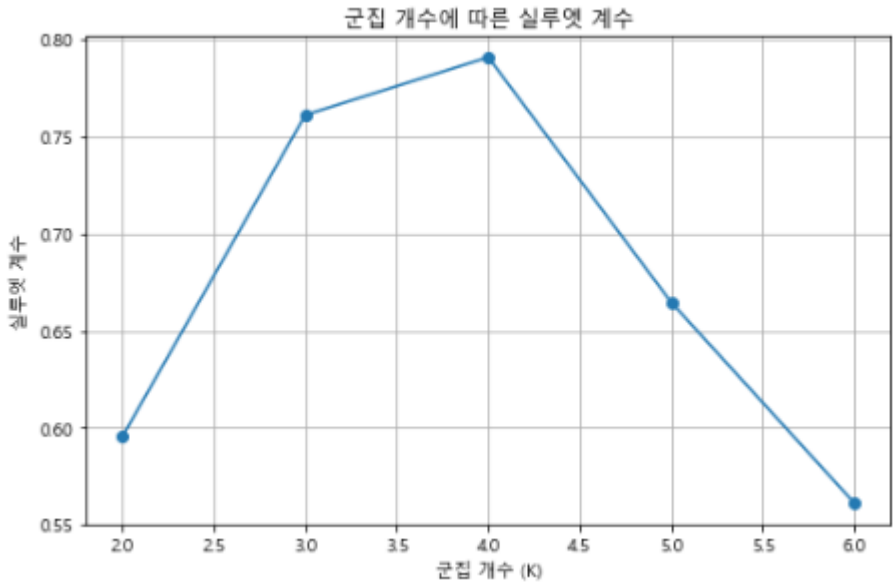
- `silhouette_score(X, labels, metric='euclidean', sample_size=None, random_state=None, **kwds)`
  - `X`: 데이터 포인트 (n_samples, n_features) 형태의 배열.
  - `labels`: 각 데이터 포인트에 할당된 군집 레이블.
  - `metric`: (선택 사항) 거리를 계산하는 데 사용할 메트릭. 기본값은 'euclidean'입니다.
  - `sample_size`: (선택 사항) 계산 속도를 높이기 위해 사용할 샘플의 크기. None이면 모든 샘플을 사용합니다.
  - `random_state`: (선택 사항) `sample_size`가 None이 아닐 때 샘플링을 위한 난수 시드.
  - 반환 값: 전체 데이터셋의 평균 실루엣 계수.
- `silhouette_samples(X, labels, metric='euclidean', **kwds)`
  - `X`: 데이터 포인트 (n_samples, n_features) 형태의 배열.
  - `labels`: 각 데이터 포인트에 할당된 군집 레이블.
  - `metric`: (선택 사항) 거리를 계산하는 데 사용할 메트릭. 기본값은 'euclidean'입니다.
  - 반환 값: 각 샘플에 대한 실루엣 계수 값의 배열.

코드 예시

```
import numpy as np
import matplotlib.pyplot as plt
from sklearn.datasets import make_blobs
from sklearn.cluster import KMeans
from sklearn.metrics import silhouette_score, silhouette_samples

# 1. 예시 데이터 생성
X, y = make_blobs(n_samples=500, n_features=2, centers=4, cluster_std=1.0,
random_state=42)

# 2. K-Means 군집화 수행 (예: K=4)
kmeans = KMeans(n_clusters=4, random_state=42, n_init=10)
cluster_labels = kmeans.fit_predict(X)

# 3. 실루엣 계수 계산
silhouette_avg = silhouette_score(X, cluster_labels)
print(f"전체 데이터셋의 실루엣 계수: {silhouette_avg:.4f}") # 0.7911

# 4. 각 샘플의 실루엣 계수 계산 및 시각화
sample_silhouette_values = silhouette_samples(X, cluster_labels)

plt.figure(figsize=(10, 7))
y_lower = 10
for i in range(4): # 4개의 군집
    # 해당 군집의 실루엣 값들을 정렬
    ith_cluster_silhouette_values = sample_silhouette_values[cluster_labels == i]
    ith_cluster_silhouette_values.sort()

    size_cluster_i = ith_cluster_silhouette_values.shape[0]
    y_upper = y_lower + size_cluster_i

    color = plt.cm.Spectral(float(i) / 4)
    plt.fill_betweenx(np.arange(y_lower, y_upper),
                      0, ith_cluster_silhouette_values,
                      facecolor=color, edgecolor=color, alpha=0.7)

    plt.text(-0.05, y_lower + 0.5 * size_cluster_i, str(i))
    y_lower = y_upper + 10 # 다음 막대를 위한 공간

plt.title("각 군집의 실루엣 플롯")
plt.xlabel("실루엣 계수 값")
plt.ylabel("군집 레이블")
plt.axvline(x=silhouette_avg, color="red", linestyle="--", label=f'평균 실루엣 계
수: {silhouette_avg:.2f}')
plt.xticks([-0.1, 0, 0.2, 0.4, 0.6, 0.8, 1])
plt.legend()
plt.show()

# 5. 최적의 K 찾기
range_n_clusters = [2, 3, 4, 5, 6]
```

```
silhouette_scores = []

for n_clusters in range_n_clusters:
    kmeans = KMeans(n_clusters=n_clusters, random_state=42, n_init=10)
    cluster_labels = kmeans.fit_predict(X)
    silhouette_avg = silhouette_score(X, cluster_labels)
    silhouette_scores.append(silhouette_avg)
    print(f"군집 개수 = {n_clusters}, 실루엣 계수 = {silhouette_avg:.4f}")
'''
군집 개수 = 2, 실루엣 계수 = 0.5955
군집 개수 = 3, 실루엣 계수 = 0.7613
군집 개수 = 4, 실루엣 계수 = 0.7911
군집 개수 = 5, 실루엣 계수 = 0.6647
군집 개수 = 6, 실루엣 계수 = 0.5614
'''

plt.figure(figsize=(8, 5))
plt.plot(range_n_clusters, silhouette_scores, marker='o')
plt.title("군집 개수에 따른 실루엣 계수")
plt.xlabel("군집 개수 (K)")
plt.ylabel("실루엣 계수")
plt.grid(True)
plt.show()
```





결과 해석 방법

- **전체 실루엣 계수**: 값이 1에 가까울수록 군집화가 잘 되었다고 판단합니다. 일반적으로 0.5 이상이면 합리적인 군집화로 간주됩니다.
- **실루엣 플롯**: 각 군집의 실루엣 플롯을 통해 개별 군집의 품질을 시각적으로 확인할 수 있습니다.
  - 플롯의 너비가 넓고, 평균 실루엣 계수(빨간 점선)보다 길게 뻗어 있는 군집은 잘 분리된 군집입니다.
  - 플롯의 너비가 좁거나, 평균 실루엣 계수보다 짧은 군집은 군집화가 잘 되지 않았거나, 다른 군집과 겹치는 부분이 많을 수 있습니다.
  - 음수 실루엣 계수를 가지는 데이터 포인트는 잘못된 군집에 할당되었을 가능성이 높습니다.
- **군집 개수에 따른 실루엣 계수**: 여러 군집 개수에 대해 실루엣 계수를 계산하고 그래프로 그렸을 때, 가장 높은 실루엣 계수를 보이는 군집 개수를 최적의 K로 선택할 수 있습니다.

장단점 및 대안

- **장점**:
  - 군집화 결과의 품질을 정량적으로 평가할 수 있습니다.
  - 최적의 군집 개수를 결정하는 데 유용한 가이드라인을 제공합니다.
  - 시각화를 통해 각 군집의 품질을 개별적으로 파악할 수 있습니다.
- **단점**:
  - 계산 비용이 높을 수 있습니다 (특히 대규모 데이터셋).
  - 볼록한 형태가 아닌 복잡한 형태의 군집에는 적합하지 않을 수 있습니다.
  - 데이터의 특성(밀도, 분포 등)에 따라 실루엣 계수만으로 최적의 군집을 판단하기 어려울 수 있습니다.
- **대안**:
  - **엘보우 방법 (Elbow Method)**: 군집 내 제곱합(SSE)을 이용하여 최적의 K를 찾는 방법입니다. 실루엣 계수와 함께 사용하여 최적 K를 결정하는 데 도움을 받을 수 있습니다.
  - **Gap Statistic**: 군집화 결과와 무작위 분포를 비교하여 최적의 K를 찾는 방법입니다.
  - **Dunn Index, Davies-Bouldin Index**: 다른 군집 유효성 지표들도 존재하며, 데이터 특성에 따라 적절한 지표를 선택하여 사용합니다.