GDG AI Study- AI week3 task3

DBSCAN을 활용한 클러스터링 과제

담당자 컴퓨터소프트웨어공학과
20204026 정명훈

00 목차

01
비지도
학습

02
DBSCAN

03
과제 설명

04
과제1

05
과제2

06
마무리

01 비지도 학습

☑

비지도 학습 (Unsupervised Learning) 이란?

머신러닝의 종류는 3가지
비지도 학습, 지도 학습, 강화 학습

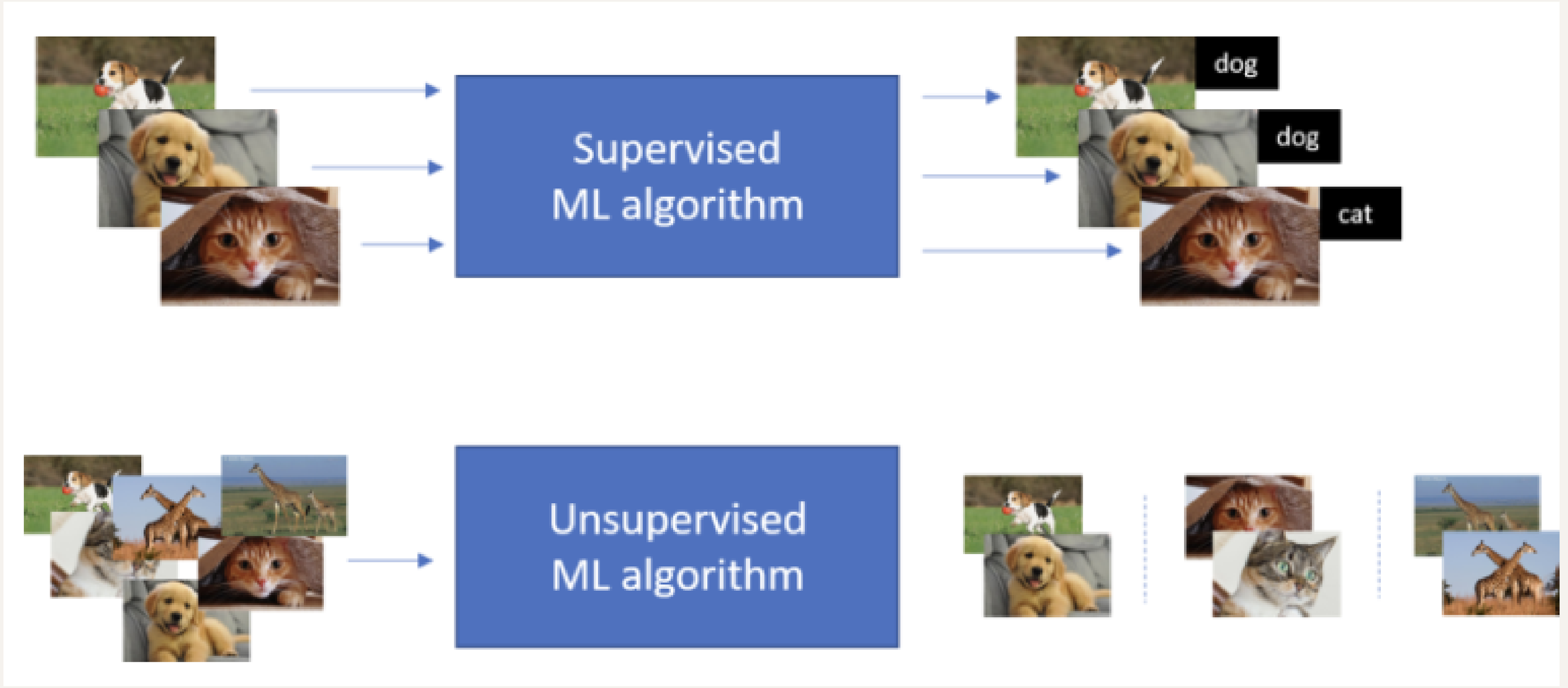
비지도 학습은 크게 **군집**과 **차원 축소** 두 가지

여러 문제를 학습함으로써
해당 데이터의 패턴, 특성 및 구조를 스스로 파악하여
이를 통해 데이터가 어떻게 구성되었는지
새로운 규칙성을 알아내는 과정

이 학습은 지도 학습 또는 강화 학습과는 달리
입력 값에 대한 목표치인 레이블이 없다.
즉, 정답을 맞추는 것이 목적이 아님.

지도 학습 vs 비지도 학습
지도 학습은 문제의 정답을 주고 학습하는 경우
비지도 학습은 정답이 주어져 있지 않지만 수많은 데이터 속에서 의미를 찾는 방법

01 비지도 학습



비지도 학습은

자가 학습 알고리즘을 사용하여 라벨이나 사전 학습 없이 학습하며 유사점, 차이점, 패턴을 기반으로 자체 규칙을 추론하고 정보를 구조화한다.

데이터에서 이전에 감지되지 않은 패턴을 식별하는 데 유용하며 데이터를 분류하는 데 유용한 특성을 식별하는 데 도움

☑

DBSCAN
밀도 기반 군집화 알고리즘

군집(Clustering)에도 많은 알고리즘들이 있다.
K-means 가 가장 대표적인 군집 알고리즘
DBSCAN, BIRCH, Mean Shift 등
여러 가지 모델이 존재

Multi Dimension의 데이터를 **밀도 기반**으로
서로 가까운 데이터 포인트를 함께 그룹화하는 알고리즘

DBSCAN은 밀도가 다양하거나 모양이 불규칙한
클러스터가 있는 데이터와 같이 모양이 잘 정의되지 않은
데이터를 처리할 때 유용하게 사용 가능

고차원의 데이터베이스들을 처리 할 수 있으며,
노이즈를 걸러낼 수 있다는 장점

03 과제 설명

☑

DBSCAN을 활용한 클러스터링 과제

과제 1: 초승달 데이터셋 클러스터링

목표: 초승달 데이터셋(Make Moons)에서 **DBSCAN을 사용해 군집을 생성하고 이상치를 탐지**

과제 2: 실제 데이터셋에서 이상치 탐지

목표: 와인 품질 데이터셋에서 DBSCAN을 사용하여 **이상치를 탐지**

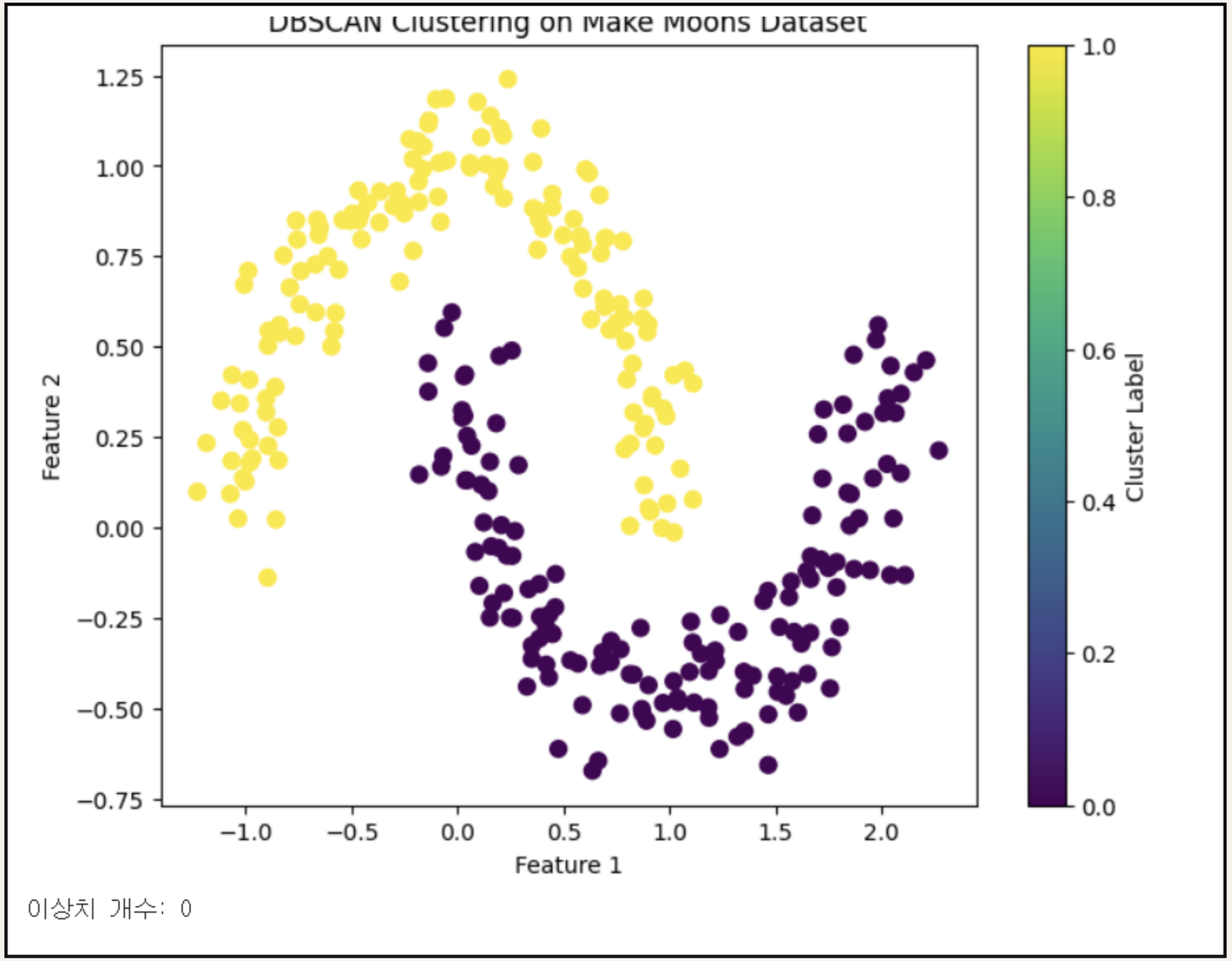
04 과제1
- 초승달 데이터셋에서 DBSCAN을 사용해 군집을 생성하고 이상치를 탐지

```
# 1. 데이터 생성 - make_moons 데이터셋으로 샘플 300개, 노이즈 0.1을 추가한 데이터셋 생성
X, _ = make_moons(n_samples=300, noise=0.1, random_state=42)

# 2. DBSCAN 모델 생성 및 학습
dbscan = DBSCAN(eps=0.2, min_samples=5) # DBSCAN 모델 초기화: eps(거리 임계값), min_samples(최소 샘플 개수) 설정
labels = dbscan.fit_predict(X) # DBSCAN 모델 학습 및 클러스터 예측

# 이상치 개수 확인 - 클러스터 레이블이 -1인 샘플의 개수
outliers = np.sum(labels == -1)

# 3. 결과 시각화
plt.figure(figsize=(8, 6))
plt.scatter(X[:, 0], X[:, 1], c=labels, cmap='viridis', s=50) # 데이터 산점도: 각 포인트를 클러스터 레이블에 따라 색깔로 구분
plt.colorbar(label='Cluster Label') # 색상 막대 추가해서 클러스터 레이블과 색상 매핑 표시
plt.title('DBSCAN Clustering on Make Moons Dataset')
plt.xlabel('Feature 1')
plt.ylabel('Feature 2')
plt.show()

print("이상치 개수:", outliers)
```

eps를 0.2 / min_samples를 5로 설정

쉽게 말하자면 Eps는 반지름/ Minpts는 최소 점 개수를 의미

04 과제1
- 초승달 데이터셋에서 DBSCAN을 사용해 군집을 생성하고 이상치를 탐지



☑

이상치 제로
두 개의 클러스터가
시각화 되었음

1 오른쪽에 있는 컬러 바는 클러스터 라벨을 나타냄
- 0 (짙은 보라색)과 1 (노란색)

2 해당 데이터셋에서 클러스터링이 성공적으로 수행되었음

3 이상치가 없는 이유는 eps와 min_samples 파라미터가 적절하게 설정되어 모든 점이 클러스터에 포함된 것으로 파악

05 과제2
- 와인 품질 데이터셋에서 DBSCAN을 사용하여 이상치를 탐지

```
# 1. 데이터 로드
data = load_wine()
X = data.data # 입력 데이터를 변수 X에 numpy 배열 형식으로 저장
feature_names = data.feature_names

# 2. 데이터 스케일링
scaler = StandardScaler() # StandardScaler 객체 생성 (평균 0, 분산 1로 표준화)
X_scaled = scaler.fit_transform(X) # 데이터를 스케일링하여 X_scaled에 저장

# 3. DBSCAN 모델 생성
eps = 3.0  # eps 값 -  DBSCAN에서 가까운 이웃을 정의하는 반경 (조정 가능)
min_samples = 3  # DBSCAN에서 하나의 클러스터로 인정할 최소 샘플 개수 (조정 가능)
dbscan = DBSCAN(eps=eps, min_samples=min_samples)

# 4. 클러스터링 수행
# 데이터를 클러스터링하고 클러스터 레이블을 예측 (-1은 이상치, 0 이상은 클러스터)
labels = dbscan.fit_predict(X_scaled)

# 5. 결과 분석
# labels에서 고유값(클러스터 ID)와 각 ID의 개수를 계산
clusters, counts = np.unique(labels, return_counts=True)

outliers = np.sum(labels == -1)  # 이상치 (-1로 표시)
total_samples = len(labels)
outlier_ratio = outliers / total_samples
```

eps를 3.0

min_samples를 3으로 설정

05 과제2
- 와인 품질 데이터셋에서 DBSCAN을 사용하여 이상치를 탐지

```
클러스터 개수: 3
이상치 개수: 7
클러스터별 샘플 수:
-1      7
 0    168
 1      3
Name: count, dtype: int64
```

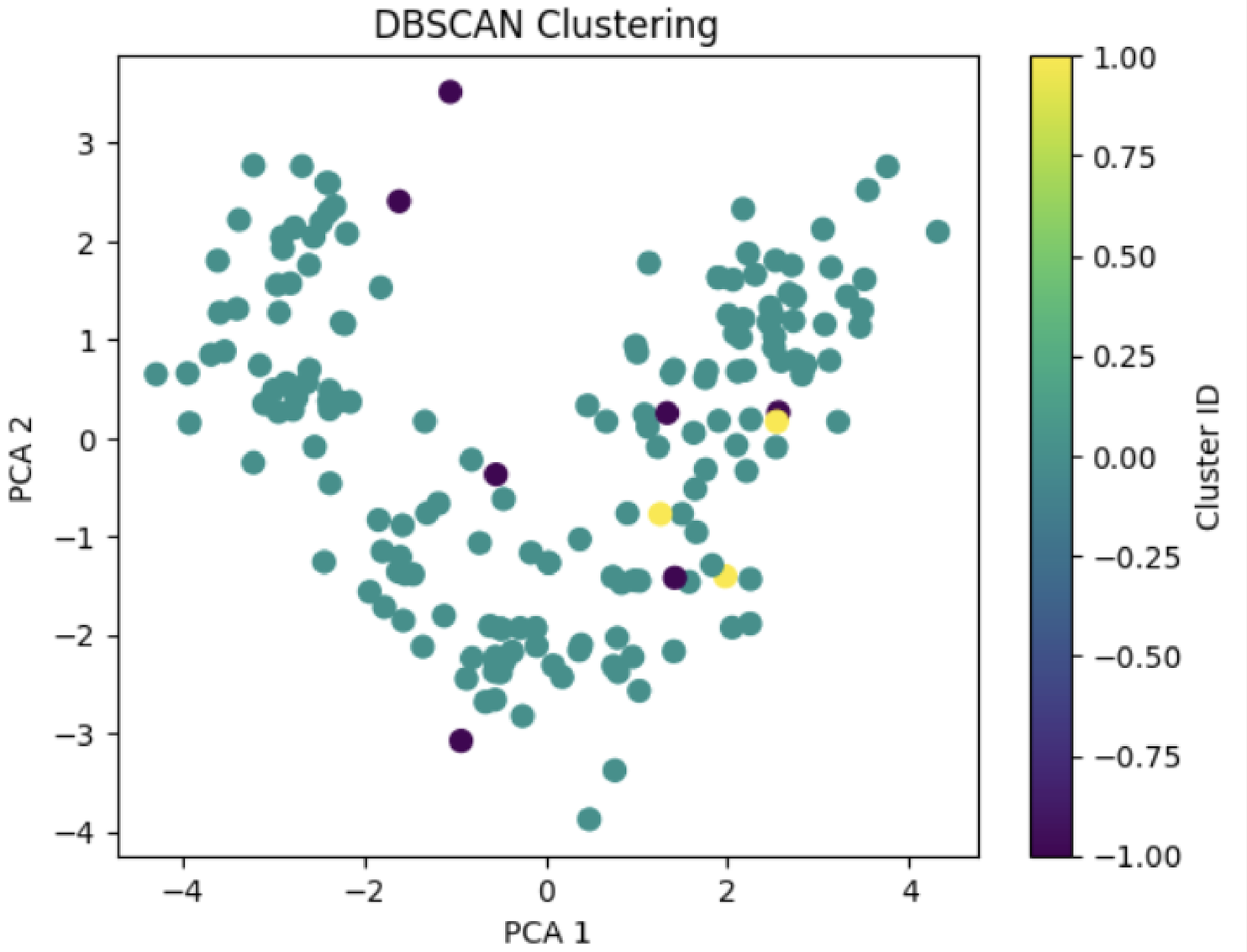
☑
이상치 7개와
두 개의 클러스터가
나타남

1 168개의 데이터가 클러스터 0에 속하고 있음.
→ 이는 **데이터의 대부분이 비교적 밀집된 영역에 존재**

2 3개의 데이터가 클러스터 1에 속하고 있음.
→ **이 클러스터가 정말 의미 있는지 검토가 필요**

3 의미 있는 소규모 클러스터이거나
단순히 소수의 가까운 데이터로 묶인 노이즈일 가능성

05 과제2
- 와인 품질 데이터셋에서 DBSCAN을 사용하여 이상치를 탐지



☑

PCA를 통해 차원 축소하고 시각화

eps 값을 조금 더 키우거나,
min_samples 값을 줄여보는 방법

```
클러스터 개수: 2
이상치 개수: 7
클러스터별 샘플 수:
-1      7
 0    168
 1      3
Name: count, dtype: int64
```

05 과제2
- 와인 품질 데이터셋에서 DBSCAN을 사용하여 이상치를 탐지

```
# 1. 데이터 로드
data = load_wine()
X = data.data # 입력 데이터를 변수 X에 numpy 배열 형식으로 저장
feature_names = data.feature_names

# 2. 데이터 스케일링
scaler = StandardScaler() # StandardScaler 객체 생성 (평균 0, 분산 1로 표준화)
X_scaled = scaler.fit_transform(X) # 데이터를 스케일링하여 X_scaled에 저장

# 3. DBSCAN 모델 생성
eps = 4.0  # eps 값 -  DBSCAN에서 가까운 이웃을 정의하는 반경 (조정 가능)
min_samples = 3  # DBSCAN에서 하나의 클러스터로 인정할 최소 샘플 개수 (조정 가능)
dbscan = DBSCAN(eps=eps, min_samples=min_samples)

# 4. 클러스터링 수행
# 데이터를 클러스터링하고 클러스터 레이블을 예측 (-1은 이상치, 0 이상은 클러스터)
labels = dbscan.fit_predict(X_scaled)

# 5. 결과 분석
# labels에서 고유값(클러스터 ID)와 각 ID의 개수를 계산
clusters, counts = np.unique(labels, return_counts=True)

outliers = np.sum(labels == -1)  # 이상치 (-1로 표시)
total_samples = len(labels)
outlier_ratio = outliers / total_samples
```

eps를 3.0 → 4.0으로 소폭 늘리고
min_samples를 3 그대로 설정

05 과제2
- 와인 품질 데이터셋에서 DBSCAN을 사용하여 이상치를 탐지

```
클러스터 개수: 2
이상치 개수: 0
클러스터별 샘플 수:
0    175
1      3
Name: count, dtype: int64
```

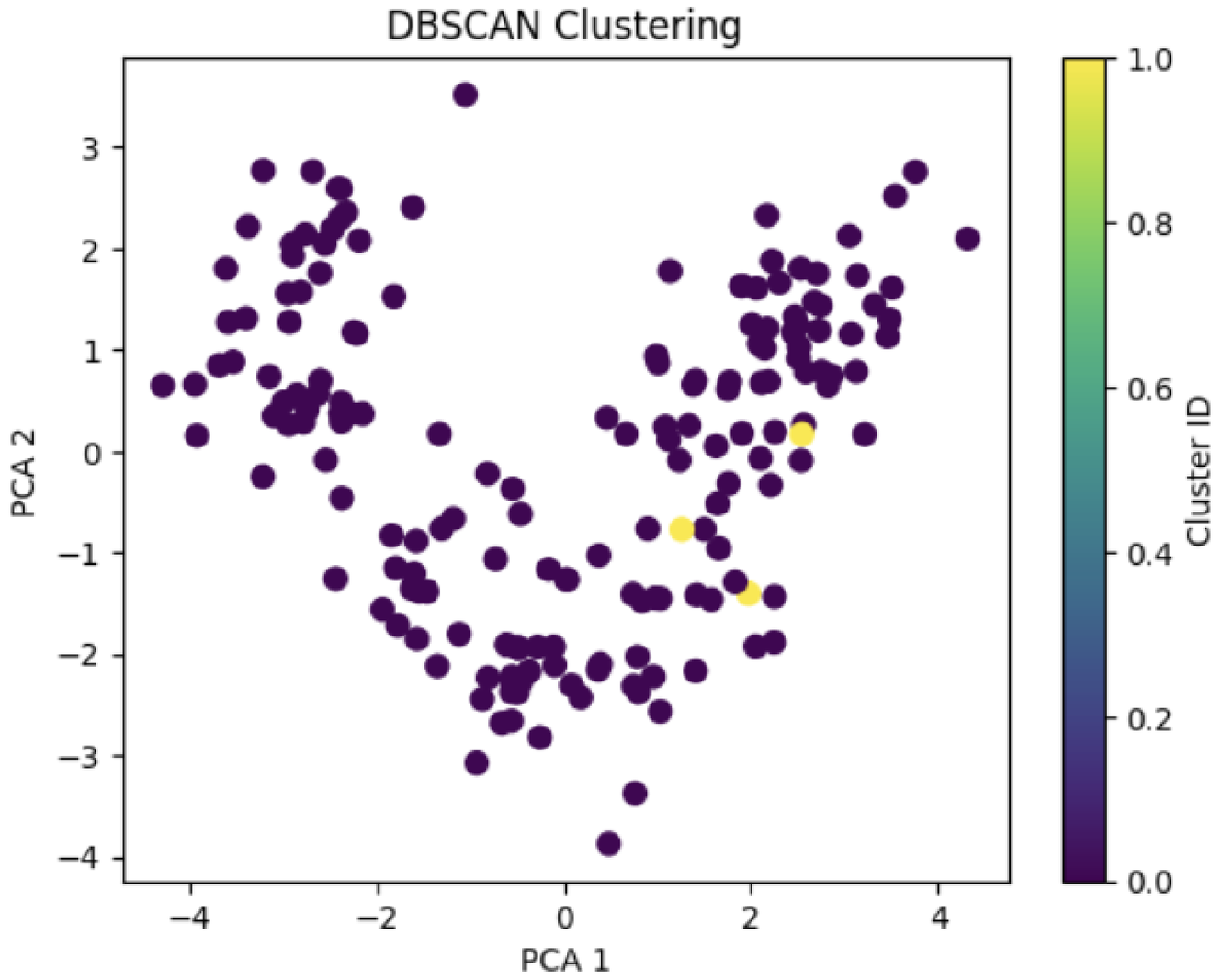
☑
이상치는 제로
클러스터는 여전히
두 개의 클러스터로
구분됨

1 2개의 클러스터를 감지

2 하나는 주 클러스터(0번), 나머지는 소규모 클러스터(1번)

3 클러스터 1에 속한 데이터가 **정말 의미 있는 패턴**인지,
혹은 **파라미터 설정(eps, min_samples)의 결과로**
발생한 작은 군집인지 추가적으로 분석

05 과제2
- 와인 품질 데이터셋에서 DBSCAN을 사용하여 이상치를 탐지



☑

PCA를 통해
차원 축소하고
시각화

클러스터1에 속한 데이터는 뭘까

```
클러스터 개수: 2
이상치 개수: 0
클러스터별 샘플 수:
0    175
1      3
Name: count, dtype: int64
```

05 과제2
- 와인 품질 데이터셋에서 DBSCAN을 사용하여 이상치를 탐지

```python
# 클러스터별 데이터 특성 확인
df = pd.DataFrame(X, columns=feature_names)
df['Cluster'] = labels

# 각 클러스터의 평균값 출력
cluster_means = df.groupby('Cluster').mean()
print(cluster_means)

# 클러스터별 샘플 크기
print(df['Cluster'].value_counts())
```

```
           alcohol  malic_acid       ash  alcalinity_of_ash   magnesium  ₩
Cluster
0        13.012000    2.355257  2.373371          19.540000   98.885714
1        12.336667    1.233333  1.966667          16.866667  149.666667

         total_phenols  flavanoids  nonflavanoid_phenols  proanthocyanins  ₩
Cluster
0             2.298743      2.0332              0.363429         1.569371
1             2.083333      1.8000              0.270000         2.846667

         color_intensity       hue  od280/od315_of_diluted_wines     proline
Cluster
0               5.094229  0.953863                      2.610686  745.954286
1               2.950000  1.166667                      2.670000  801.666667
Cluster
0    175
1      3
Name: count, dtype: int64
```

☑

클러스터별 데이터 특성 확인 / 각 클러스터의 평균값 출력

두 클러스터를 보기 좋게 비교해보면?

05 과제2
- 와인 품질 데이터셋에서 DBSCAN을 사용하여 이상치를 탐지

- ☑ 클러스터 0은 일반적인 와인 특성을 반영하며, 클러스터링 결과에서 대부분의 데이터를 차지
- ☑ 반면에 클러스터 1은 클러스터 0과 구별되는 특성을 가진 소수의 데이터를 나타냄
- ☑ 오른쪽에는 클러스터 간 큰 차이를 보이는 특성들 비교하는 도표
- ☑ 클러스터 1에 해당하는 와인은 고급 와인이나 특정 조건에서 생산된 와인일 가능성

/	클러스터 0	클러스터 1
malic_acid (말산 농도)	2.355257	1.233333
magnesium (마그네슘 농도)	98.885714	149.666667
proanthocyanins (프로안토시아닌 함량)	1.569371	2.846667
proline (프로린)	745.954286	801.666667
color_intensity	5.094229	2.950000
데이터 갯수	175	3

06 마무리

☑

과제를 진행하면서 깨달은 점 정리

https://blog.naver.com/march03190/222792748678

-> 비지도 학습 DBSCAN 알고리즘을
그림으로 잘 설명하고 있어서 참고하시면 좋을 것 같아서
링크 넣었습니다!

DBSCAN은 **데이터 포인트의 밀도를 기반으로**
클러스터를 형성하며,
비선형적이고 임의의 모양을 가진 클러스터를 잘 탐지

-> 이를 통해 복잡한 데이터 구조를 분석하는 데 효과적이라는 점을 깨달았습니다.

---

eps (반경)와 min_samples (최소 샘플 수) 설정에 따라
결과가 크게 달라진다.

-> 적절한 파라미터 값을 찾기 위해 데이터 분포를 시각화하거나, 여러 값으로 실험해보는 것이 중요하다는 점을 배웠습니다.

또한 단순히 시각화하는 것만으로 만족하는 것이 아닌
각 클러스터의 특성을 분석하며, 데이터의 의미를 도출하는 과정의 중요성과 도메인 지식의 필요성을 느꼈습니다.