혼자 공부하는 머신러닝+딥러닝

5·6장 객관식 문제집

트리 알고리즘 · 비지도 학습 — 5지선다 250문제 (정답·해설 포함)

강의 슬라이드 및 강의 스크립트 전체 분석 기반

활용 안내

본 문제집은 5장(트리 알고리즘)과 6장(비지도 학습)의 강의 슬라이드 본문과 강의 스크립트를 함께 분석하여 제작한 5지선다 객관식 250문항입니다. 각 문항 바로 아래에 정답과 해설을 함께 수록하였습니다.

구성 5장 134문항(결정 트리 60 · 교차 검증과 그리드 서치 38 · 트리의 앙상블 36), 6장 116문항(군집 알고리즘 38 · k-평균 40 · 주성분 분석 38).

5 장 트리 알고리즘

5-1 결정 트리

1. 결정 트리가 데이터를 분류하는 기하학적 방식을 가장 잘 설명한 것은?

① 하나의 직선(또는 평면)으로 공간을 둘로 가른다
② 특성 축에 수직인 경계들로 공간을 사각형 영역으로 잘게 나눈다
③ 원형 영역으로 데이터를 감싼다
④ 모든 샘플을 한 점으로 모은다
⑤ 거리 기반으로 가장 가까운 이웃을 찾는다

> **정답** ②
>
> **해설** 결정 트리는 매 분할에서 '한 특성 ≤ 임계값' 형태로 나누므로 경계가 특성 축에 수직입니다. 그 결과 입력 공간이 여러 개의 사각형(상자) 영역으로 쪼개지고, 영역마다 하나의 클래스가 할당됩니다. 직선 하나로 가르는 로지스틱 회귀와 본질적으로 다릅니다.

2. 결정 트리에서 max_depth(깊이)를 점점 늘릴 때 일반적으로 나타나는 현상은?

① 훈련·테스트 점수가 함께 계속 오른다
② 훈련 점수는 계속 오르지만 어느 지점부터 테스트 점수는 오히려 떨어진다
③ 훈련 점수만 떨어진다
④ 두 점수 모두 변하지 않는다
⑤ 테스트 점수만 계속 오른다

> **정답** ②
>
> **해설** 깊이를 늘릴수록 트리는 훈련 데이터를 더 잘게 맞춰 훈련 점수가 계속 오릅니다. 그러나 일정 깊이를 넘으면 훈련 데이터의 잡음까지 외워(과대적합) 새 데이터에 대한 테스트 점수는 떨어집니다. 깊이는 대표적인 복잡도 조절 하이퍼파라미터입니다.

3. 분류용 결정 트리에서 한 리프(말단) 노드에 도달한 샘플의 예측 클래스는 어떻게 정해지는가?

① 가장 먼저 도달한 샘플의 클래스
② 그 리프에 모인 훈련 샘플 중 가장 많은 클래스(다수결)
③ 항상 양성 클래스
④ 랜덤으로 하나 선택
⑤ 부모 노드의 클래스

> **정답** ②
>
> **해설** 리프 노드의 예측값은 그 리프에 도달한 훈련 샘플들 중 다수를 차지한 클래스입니다. 즉 트리는 입력을 조건들에 따라 한 리프로 보낸 뒤, 그 리프의 다수 클래스로 예측합니다.

4. 어떤 노드를 특성 A로 나누면 정보 이득이 0.12, 특성 B로 나누면 0.03이다. 결정 트리의 선택과 근거로 옳은 것은?

① B를 선택한다 — 정보 이득이 작아야 좋다
② A를 선택한다 — 정보 이득(불순도 감소)이 큰 분할을 택한다
③ 둘 다 사용해 평균낸다
④ 특성 이름이 빠른 A를 선택한다
⑤ 정보 이득과 무관하게 무작위로 고른다

> **정답** ②
>
> **해설** 결정 트리는 각 노드에서 가능한 분할 중 정보 이득(부모 불순도 − 자식 불순도의 가중평균)이 가장 큰 것을 고릅니다. A의 정보 이득이 더 크므로 A로 분할합니다.

5. 결정 트리로 3개 이상의 클래스를 분류(다중 분류)할 때 일대다(OvR) 같은 별도 변환이 필요한가?

① 필요하다 — 항상 이진 분류기 여러 개로 바꿔야 한다
② 필요 없다 — 각 리프에서 다수 클래스를 고르면 되므로 다중 분류를 자연스럽게 지원한다
③ 필요하다 — 클래스마다 트리를 따로 만들어야 한다
④ 다중 분류는 불가능하다
⑤ 표준화를 먼저 해야만 가능하다

> **정답** ②
>
> **해설** 결정 트리는 각 리프에서 다수 클래스를 예측값으로 내므로 클래스가 2개든 10개든 구조 변경 없이 다중 분류를 지원합니다. 로지스틱 회귀처럼 일대다로 쪼갤 필요가 없습니다.

6. 판다스의 info() 메서드가 한 번에 보여주는 두 가지 정보는?

① 평균과 표준편차
② 최솟값과 최댓값
③ 누락값 여부와 각 열의 데이터 타입
④ 사분위수와 중앙값
⑤ 상관계수와 공분산

> **정답** ③
>
> **해설** info() 메서드는 각 컬럼의 Non-Null Count(누락값 여부)와 Dtype(데이터 타입)을 동시에 보여줍니다. 데이터를 처음 받았을 때 반사적으로 호출하는 메서드입니다.

7. 결정 트리 하나만 사용할 때의 대표적 약점으로, 여러 트리를 묶는 앙상블의 동기가 되는 성질은?

① 학습이 너무 느리다
② 훈련 데이터가 조금만 바뀌어도 트리 구조가 크게 달라지는 높은 분산(불안정성)
③ 표준화가 반드시 필요하다
④ 해석이 전혀 불가능하다
⑤ 특성을 한 번씩만 쓸 수 있다

> **정답** ②
>
> **해설** 단일 결정 트리는 데이터의 작은 변화에도 분할 지점이 크게 흔들리는 '높은 분산'을 보입니다. 이를 완화하려고 여러 트리를 만들어 결합하는 것이 랜덤 포레스트 같은 앙상블입니다.

8. 와인 데이터의 네 개 열이 모두 가진 데이터 타입 float64는 무엇을 의미하는가?

① 64자리 문자열
② 64비트 정수
③ 64비트 부동소수점(실수)
④ 64개의 범주
⑤ 64차원 벡터

> **정답** ③
>
> **해설** float64는 64비트 부동소수점, 즉 실수 타입입니다. 알코올 도수 9.4, 당도 1.9처럼 소수점이 있는 값입니다. 만약 object 타입이면 보통 문자열이라 숫자로 변환해야 합니다.

9. describe() 메서드 출력에서 알 수 있는, 표준화가 필요하다고 판단한 근거는?

① 누락값이 많아서
② 세 특성(알코올, 당도, pH)의 값 범위(스케일)가 서로 크게 다르기 때문
③ 데이터 타입이 문자열이어서
④ 샘플 수가 부족해서
⑤ 타깃이 불균형해서

> **정답** ②
>
> **해설** describe()에서 alcohol 평균 약 10.5, sugar 약 5.4, pH 약 3.2로 세 특성의 값 범위가 크게

달랐습니다. 거리·가중치 기반 알고리즘에서는 큰 값을 가진 특성이 더 큰 영향을 주므로 StandardScaler로 표준화가 필요합니다.

10. train_test_split에서 test_size=0.2로 지정했을 때의 의미는?

① 전체의 2%를 테스트로 사용
② 전체의 20%를 테스트 세트로 떼어낸다
③ 전체의 80%를 테스트로 사용
④ 테스트 세트를 2개로 나눈다
⑤ 20개 샘플만 테스트로 쓴다

정답 ②

해설 test_size=0.2는 전체 데이터의 20%를 테스트 세트로 떼어내라는 의미로, 8 대 2 비율로 나뉩니다. 8 대 2가 머신러닝에서 가장 흔히 쓰이는 비율입니다.

11. random_state=42를 지정하는 주된 목적은?

① 모델 성능을 높이기 위해
② 매번 같은 분할/결과가 재현되도록 시드를 고정하기 위해
③ 42개의 폴드를 만들기 위해
④ 학습 속도를 높이기 위해
⑤ 42% 데이터를 사용하기 위해

정답 ②

해설 random_state=42는 매번 같은 결과가 나오도록 하는 시드 값입니다. 42라는 숫자 자체는 별 의미 없는 관습이며, 고정하지 않으면 코드를 다시 돌릴 때마다 결과가 미세하게 달라져 디버깅이 어려워집니다.

12. 결정 트리에서 하나의 특성이 트리 안의 여러 노드에서 반복적으로 분할 기준으로 쓰일 수 있는가?

① 아니다 — 각 특성은 트리 전체에서 한 번만 쓸 수 있다
② 그렇다 — 같은 특성을 서로 다른 임계값으로 여러 노드에서 반복 사용할 수 있다
③ 아니다 — 첫 노드에서만 사용된다
④ 특성은 분할에 쓰이지 않는다
⑤ 리프에서만 사용된다

정답 ②

해설 결정 트리는 한 특성을 깊이마다 다른 임계값으로 여러 번 사용할 수 있습니다. 예컨대 '당

도 ≤ 4'로 나눈 뒤 자식 노드에서 다시 '당도 ≤ 7'로 나눌 수 있어, 한 특성에 대한 비선형 구간 분할이 가능합니다.

13. StandardScaler를 사용할 때 fit()은 어느 데이터에만 적용해야 하는가?

① 테스트 세트
② 훈련 세트
③ 훈련 세트와 테스트 세트 모두
④ 검증 세트
⑤ 전체 데이터

정답 ②

해설 fit은 반드시 훈련 세트로만 합니다. 테스트 세트로 fit하면 테스트 데이터의 평균·표준편차 정보가 모델로 새어 들어가는 데이터 누수(data leakage)가 발생해 평가가 부풀려집니다.

14. 머신러닝 전처리의 황금률로 슬라이드가 강조한 원칙은?

① fit과 transform 모두 양쪽에
② fit은 훈련에만, transform은 양쪽에
③ fit은 테스트에만, transform은 훈련에만
④ fit과 transform 모두 테스트에만
⑤ 전처리는 학습 후에 한다

정답 ②

해설 fit은 훈련 세트로만 하고, 거기서 학습한 평균·표준편차를 transform으로 훈련 세트와 테스트 세트 모두에 적용합니다. 'fit은 훈련에만, transform은 양쪽에'가 핵심 원칙입니다.

15. 테스트 세트로 전처리를 fit하면 발생하는 문제를 가리키는 용어는?

① 과대적합(overfitting)
② 과소적합(underfitting)
③ 데이터 누수(data leakage)
④ 차원의 저주
⑤ 클래스 불균형

정답 ③

해설 테스트 데이터의 정보가 모델 학습에 미리 알려지는 것을 데이터 누수(data leakage)라고 합니다. 평가가 부풀려져 나오며, 캐글 대회에서 비공개 테스트셋 점수가 폭락하는 원인이 됩니다.

16. 로지스틱 회귀로 와인을 분류한 결과 훈련 점수 0.7808, 테스트 점수 0.7777이 나왔다. 이 진단으로 가장 적절한 것은?

① 전형적인 과대적합
② 두 점수가 모두 낮은 과소적합
③ 완벽한 일반화
④ 데이터 누수 발생
⑤ 이상치가 많음

> **정답** ②
>
> **해설** 두 점수가 거의 비슷하므로 과대적합은 아니지만, 둘 다 78%가 안 되는 낮은 값이므로 과소적합입니다. 모델이 너무 단순해서 데이터를 충분히 학습하지 못한 상태입니다.

17. 과소적합이 진단되었을 때 취할 수 있는 옵션이 아닌 것은?

① 모델을 더 복잡하게 만든다
② 특성을 추가하거나 특성 공학을 한다
③ 다른 알고리즘을 시도한다
④ 더 강한 규제를 추가해 모델을 더 단순하게 만든다
⑤ 비선형 모델로 바꾼다

> **정답** ④
>
> **해설** 과소적합은 모델이 너무 단순한 상태이므로 모델을 더 복잡하게, 특성 추가, 다른 알고리즘 시도가 해법입니다. 규제를 강화해 모델을 더 단순하게 만드는 것은 오히려 과소적합을 악화시킵니다.

18. 로지스틱 회귀가 학습한 계수와 절편(coef_, intercept_)을 본부장에게 설명하기 어려운 이유와 가장 관련 깊은 개념은?

① 데이터 누수
② 설명 가능한 AI(XAI)
③ 차원 축소
④ 교차 검증
⑤ 부트스트랩

> **정답** ②
>
> **해설** 성능과 별개로 의사결정자에게 설명할 수 없는 모델은 현장에서 채택되기 어렵습니다. 이것이 설명 가능한 AI(Explainable AI, XAI)가 중요한 이유이며, 의료·금융·법률·채용 같은 고위험 분야에서 특히 중요합니다.

19. 로지스틱 회귀의 '계수'와 결정 트리의 '분기 규칙' 중 비전문가에게 판단 근거를 설명하기 더 쉬운 쪽과 그 이유로 옳은 것은?

① 로지스틱 회귀 — 계수가 직관적이라서
② 결정 트리 — '당도 ≤ X이면 화이트' 같은 if-then 규칙으로 표현돼 사람이 따라가기 쉽다
③ 둘 다 동일하게 어렵다
④ 결정 트리 — 정확도가 더 높아서
⑤ 로지스틱 회귀 — 그래프가 예뻐서

> **정답** ②
>
> **해설** 결정 트리는 루트에서 리프까지의 경로가 '특성 임계값 비교'의 연속이라 if-then 규칙으로 그대로 읽힙니다. 반면 로지스틱 회귀의 계수·절편은 여러 특성의 가중합이라 비전문가에게 설명하기 어렵습니다. 이것이 트리의 해석 가능성(설명 가능성) 장점입니다.

20. 결정 트리와 로지스틱 회귀의 분류 공간 처리 방식의 본질적 차이는?

① 둘 다 직선 하나로 자른다
② 로지스틱 회귀는 직선 하나, 결정 트리는 공간을 여러 칸으로 쪼갠다
③ 로지스틱 회귀는 칸, 결정 트리는 직선 하나
④ 둘 다 공간을 칸으로 쪼갠다
⑤ 둘 다 곡선으로 자른다

> **정답** ②
>
> **해설** 로지스틱 회귀는 직선 하나로 공간을 자르는 게 한계이고, 결정 트리는 가로세로 직선들로 공간을 작은 칸(직사각형)들로 쪼갭니다. 칸을 늘리면 어떤 모양이든 근사할 수 있어 표현력이 큽니다.

21. 한 클래스가 여러 군데 흩어져 있어 직선 한 개로 분류할 수 없는 상황을 가리키는 용어는?

① 과대적합
② 선형 분리 불가능(linearly inseparable)
③ 데이터 누수
④ 차원의 저주
⑤ 클래스 불균형

> **정답** ②
>
> **해설** 한 클래스가 여러 영역에 흩어져 있어 직선 한 개로는 표현할 수 없는 문제를 '선형 분리 불가능(linearly inseparable)'이라고 합니다. 로지스틱 회귀가 와인에서 78%밖에 못 맞히는 이유

입니다.

22. 사이킷런에서 결정 트리 분류 모델을 만드는 클래스는?

① LogisticRegression
② DecisionTreeClassifier
③ KMeans
④ RandomForestClassifier
⑤ StandardScaler

정답 ②

해설 from sklearn.tree import DecisionTreeClassifier로 가져와 사용합니다. fit으로 학습, score로 평가하는 패턴은 로지스틱 회귀와 동일하며, 이것이 사이킷런 일관된 API의 힘입니다.

23. 깊이 제한 없는 결정 트리의 결과가 훈련 점수 0.9969, 테스트 점수 0.8592였다. 이 진단은?

① 과소적합
② 전형적인 과대적합
③ 완벽한 일반화
④ 데이터 누수
⑤ 선형 분리 불가능

정답 ②

해설 훈련 점수는 거의 완벽한데 테스트 점수는 14% 가까이 낮아 격차가 약 0.14입니다. 훈련 데이터를 외워버린 전형적인 과대적합 신호입니다.

24. 결정 트리에서 가장 위에 있는, 모든 트리가 시작하는 노드의 이름은?

① 리프 노드
② 루트 노드
③ 자식 노드
④ 중심 노드
⑤ 내부 노드

정답 ②

해설 가장 위에 있는 노드를 루트 노드(root node, 뿌리 노드)라고 합니다. 모든 결정 트리는 루트 노드에서 시작해 아래로 자라납니다.

25. 결정 트리에서 더 이상 분할되지 않는 가장 아래 노드의 이름은?

① 루트 노드
② 리프 노드
③ 부모 노드
④ 중심 노드
⑤ 분기 노드

> **정답** ②
>
> **해설** 가장 아래에 있어 더 이상 분할되지 않는 노드를 리프 노드(leaf node, 잎 노드)라고 합니다. 새 샘플은 루트에서 시작해 트리를 따라 내려가다 리프 노드에 도착하며, 그 리프가 직관 슬라이드의 '칸' 하나에 해당합니다.

26. plot_tree 함수에서 max_depth=1을 지정하는 것의 의미로 옳은 것은?

① 모델 트리의 최대 깊이를 1로 제한해 학습한다
② 이미 학습된 트리를 깊이 1까지만 화면에 그리는 시각화 옵션이다
③ 트리를 1개만 만든다
④ 특성을 1개만 사용한다
⑤ 1번 반복 학습한다

> **정답** ②
>
> **해설** plot_tree의 max_depth=1은 시각화 옵션으로, 이미 깊이 제한 없이 학습된 트리의 윗부분 한 단계까지만 화면에 그립니다. 모델 트리 자체의 깊이를 제한하는 DecisionTreeClassifier의 max_depth와 혼동하면 안 됩니다.

27. plot_tree에서 filled=True 옵션의 효과는?

① 트리를 흑백으로 그린다
② 노드의 다수 클래스에 따라 색을 칠하고, 순도가 높을수록 색이 진해진다
③ 특성 이름을 표시한다
④ 트리를 가득 채워 그린다
⑤ 리프 노드만 표시한다

> **정답** ②
>
> **해설** filled=True는 노드의 다수 클래스에 따라 색을 칠합니다. 이진 분류에서 한쪽은 주황 계열, 다른 쪽은 파랑 계열이며, 한 클래스가 압도적이면 짙은 색, 두 클래스가 비슷하면 흐릿한 색이 됩니다.

28. plot_tree로 그린 어떤 노드에 value=[10, 90]으로 표시됐다. 이 노드에 도달한 샘플을 트리는 어느 클래스로 예측하는가?

① 첫 번째 클래스(10인 쪽)
② 두 번째 클래스(90으로 다수인 쪽)
③ 예측할 수 없다
④ 두 클래스를 반반 예측
⑤ value와 예측은 무관하다

> **정답** ②
>
> **해설** value는 그 노드에 도달한 샘플의 클래스별 개수입니다. 예측은 다수 클래스를 따르므로 90인 두 번째 클래스로 예측합니다. 또한 이 비율(90/100=0.9)이 그 클래스에 대한 예측 확률이 됩니다.

29. 어떤 분할을 했더니 두 자식 노드의 불순도 가중평균이 부모 노드의 불순도와 같거나 더 컸다. 이 분할에 대한 결정 트리의 처리는?

① 반드시 그 분할을 채택한다
② 정보 이득이 0 이하라 도움이 되지 않으므로 그 분할은 선택되지 않는다
③ 불순도를 0으로 강제 조정한다
④ 샘플을 삭제한다
⑤ 트리를 처음부터 다시 만든다

> **정답** ②
>
> **해설** 정보 이득 = 부모 불순도 − 자식 불순도 가중평균입니다. 이 값이 0 이하이면 분할로 얻는 순도 향상이 없으므로, 트리는 정보 이득이 양수인(불순도를 줄이는) 다른 분할을 선택합니다.

30. 결정 트리 노드 안에 표시되는 'samples' 값이 의미하는 것은?

① 분할 기준값
② 그 노드의 지니 불순도
③ 그 노드에 들어 있는 전체 샘플 수
④ 클래스별 샘플 분포
⑤ 트리의 깊이

> **정답** ③
>
> **해설** samples는 그 노드에 들어 있는 전체 샘플 수입니다. 루트 노드의 samples는 훈련 세트 크기와 같은 5197로 나옵니다.

31. 노드 안의 value=[1258, 3939]에서 두 숫자가 의미하는 것은?

① 분할 기준의 최솟값과 최댓값

② 음성 클래스(레드)와 양성 클래스(화이트)의 샘플 수

③ 왼쪽과 오른쪽 자식의 깊이

④ 지니와 엔트로피 값

⑤ 훈련과 테스트 샘플 수

> **정답** ②
>
> **해설** value는 클래스별 샘플 분포로, 앞쪽 1258은 음성 클래스(레드 와인), 뒤쪽 3939는 양성 클래스(화이트 와인)의 수입니다. 합치면 samples(5197)와 일치합니다.

32. 부모 노드의 모든 샘플이 두 자식 노드로 나뉘는 방식에 대한 설명으로 옳은 것은?

① 일부 샘플은 어느 자식에도 속하지 않는다

② 한 샘플이 여러 노드에 동시에 속할 수 있다

③ 모든 샘플이 빠짐없이 정확히 한 자식 노드에 속한다

④ 샘플이 무작위로 복제되어 양쪽에 들어간다

⑤ 절반씩 균등하게 나뉜다

> **정답** ③
>
> **해설** 부모 노드의 모든 샘플은 두 자식으로 빠짐없이 나뉘며, 한 샘플은 항상 정확히 한 노드에만 속합니다. 왼쪽 2922 + 오른쪽 2275 = 부모 5197로 일치합니다.

33. 지니 불순도(gini)에서 DecisionTreeClassifier의 criterion 매개변수 기본값은?

① 'entropy'

② 'gini'

③ 'log_loss'

④ 'mse'

⑤ 'random'

> **정답** ②
>
> **해설** criterion 매개변수의 기본값은 'gini'(지니 불순도)입니다. criterion은 '기준'이라는 뜻으로, 노드를 분할할 때 어떤 기준을 쓸지 정합니다.

34. 지니 불순도의 정의 공식으로 옳은 것은?

① 음성 비율 + 양성 비율

② 1 - (음성 비율² + 양성 비율²)

③ 음성 비율² + 양성 비율²

④ (음성 비율 - 양성 비율)²

⑤ 음성 비율 × 양성 비율

> **정답** ②
>
> **해설** 지니 불순도 = 1 - (음성 클래스 비율² + 양성 클래스 비율²)입니다. 한 클래스만 있으면 0(순수), 두 클래스가 반반이면 이진 분류에서 최댓값 0.5가 됩니다.

35. 이진 분류에서 지니 불순도가 0이 되는 경우는?

① 두 클래스가 50:50으로 섞인 노드

② 한 클래스만 있는 순수 노드

③ 샘플이 100개인 노드

④ 두 클래스가 76:24인 노드

⑤ 어떤 경우에도 0이 될 수 없다

> **정답** ②
>
> **해설** 노드에 한 클래스만 있어 완전히 순수하면 지니 불순도가 0이 됩니다. 예: 100개가 모두 화이트면 $1^2 + 0^2 = 1$, $1 - 1 = 0$. 이런 노드를 순수 노드(pure node)라고 합니다.

36. 이진 분류에서 지니 불순도가 최댓값 0.5가 되는 경우는?

① 한 클래스만 있을 때

② 두 클래스가 정확히 반반(50:50) 섞여 있을 때

③ 샘플이 한 개일 때

④ 클래스가 3개일 때

⑤ 불순도는 1이 최댓값이다

> **정답** ②
>
> **해설** 두 클래스가 50:50이면 $0.5^2 + 0.5^2 = 0.5$, $1 - 0.5 = 0.5$로 이진 분류에서 지니 불순도의 최댓값입니다. 두 클래스가 정확히 반반일 때 가장 불순합니다.

37. 루트 노드(음성 1258, 양성 3939, 총 5197)의 지니 불순도 계산 결과는?

① 0

② 0.367

③ 0.5

④ 0.798

⑤ 1.0

정답 ②

해설 1 - ((1258/5197)² + (3939/5197)²) = 1 - (0.0586 + 0.5746) ≈ 0.367입니다. 0.5에 가까운 편이므로 두 클래스가 어느 정도 섞여 있습니다.

38. 정보 이득(information gain)의 정의로 옳은 것은?

① 부모 노드의 불순도 그 자체
② 부모 노드의 불순도와 자식 노드들 불순도(가중평균)의 차이
③ 두 자식 노드 불순도의 합
④ 리프 노드의 불순도
⑤ 전체 트리 깊이

정답 ②

해설 정보 이득은 분할 후 불순도가 얼마나 줄었는지를 나타내며, 부모 불순도에서 자식 노드들의 샘플 수 가중평균 불순도를 뺀 값입니다. 결정 트리는 이 값이 최대가 되는 분할을 고릅니다.

39. 정보 이득 계산에서 자식 노드들의 불순도를 단순 평균이 아니라 '샘플 수 가중 평균'으로 계산하는 이유는?

① 계산이 더 빨라서
② 자식 노드의 크기가 다를 수 있어 큰 자식의 불순도가 더 큰 영향을 줘야 공평하므로
③ 불순도를 0~1로 만들기 위해
④ 지니와 엔트로피를 통일하기 위해
⑤ 음수를 방지하기 위해

정답 ②

해설 자식 노드 크기가 다를 수 있어, 샘플이 많은 자식의 불순도가 더 큰 영향을 줘야 공평하므로 샘플 비율로 가중합니다. 예: 4000개짜리 자식이 1000개짜리보다 더 큰 가중치를 가집니다.

40. 결정 트리 알고리즘을 한 문장으로 요약한 것으로 옳은 것은?

① 불순도 기준을 사용해 정보 이득이 최대가 되도록 노드를 분할한다
② 거리를 최소화하도록 중심을 이동한다
③ 분산이 큰 방향을 찾아 투영한다
④ 오차를 역전파해 가중치를 갱신한다
⑤ 데이터를 무작위로 분할한다

정답 ①

해설 결정 트리는 '불순도 기준을 사용해 정보 이득이 최대가 되도록 노드를 분할'합니다. 노드

를 순수하게 나눌수록 정보 이득이 커집니다.

41. **criterion='entropy'로 지정했을 때 사용하는 불순도는?**

① 지니 불순도
② 엔트로피 불순도
③ 평균 제곱 오차
④ 로지스틱 손실
⑤ 교차 엔트로피 정확도

정답 ②

해설 criterion='entropy'를 지정하면 엔트로피 불순도를 사용합니다. 엔트로피는 정보이론(클로드 섀넌, 1948)에서 온 개념으로, 노드 안 클래스의 불확실성을 측정합니다.

42. **지니 불순도와 엔트로피 불순도의 관계에 대한 설명으로 옳은 것은?**

① 둘은 완전히 다른 것을 측정한다
② 둘 다 노드의 불순도를 측정하며 실용적으로 성능 차이가 거의 없다
③ 엔트로피가 항상 지니보다 정확하다
④ 지니는 회귀, 엔트로피는 분류 전용이다
⑤ 둘 중 하나만 사이킷런에서 지원한다

정답 ②

해설 지니와 엔트로피는 측정 단위가 다를 뿐 둘 다 노드 불순도를 잽니다. 실용적으로 1~2% 정도 차이가 날까 말까이며, 지니가 로그 계산이 없어 살짝 빨라서 사이킷런 기본값입니다.

43. **결정 트리가 매 분할 단계에서 그 시점에 가장 좋아 보이는 분할을 선택하는 방식을 가리키는 용어는?**

① 탐욕적(greedy) 알고리즘
② 동적 계획법
③ 전역 최적화
④ 역전파
⑤ 부트스트랩

정답 ①

해설 결정 트리는 탐욕적(greedy) 알고리즘으로, 매 단계 가장 큰 정보 이득을 주는 분할을 고르며 트리 전체 결과는 고려하지 않습니다. 따라서 전역 최적이 아닐 수 있고, 이 약점을 앙상블이 보완합니다.

44. 과대적합된 결정 트리를 해결하는 가장 간단하고 효과적인 방법으로 슬라이드가 든 것은?

① 특성 추가
② 가지치기(pruning), 즉 트리 깊이 제한
③ 표준화
④ 데이터 증강
⑤ 학습률 감소

> **정답** ②
> **해설** 가지치기(pruning)는 트리가 자라는 깊이를 제한하는 방법으로, 과대적합을 막는 가장 간단하고 효과적인 방법입니다. DecisionTreeClassifier의 max_depth 매개변수로 제어합니다.

45. max_depth=3으로 가지치기한 결과 훈련 0.8455, 테스트 0.8415가 나왔다. 깊이 제한 없던 모델(훈련 0.9969, 테스트 0.8592)과 비교한 핵심 변화는?

① 테스트 점수가 크게 올랐다
② 훈련-테스트 격차가 0.14에서 0.004로 줄어 일반화 성능이 좋아졌다
③ 두 점수 모두 크게 올랐다
④ 과대적합이 더 심해졌다
⑤ 훈련 점수만 올랐다

> **정답** ②
> **해설** 훈련 점수는 떨어졌지만 훈련-테스트 격차가 0.14에서 0.004로 거의 사라졌습니다. 약간의 테스트 점수 손실로 안정성(일반화 성능)을 크게 얻은 좋은 거래입니다.

46. 편향-분산 트레이드오프에서 모델이 너무 단순해 패턴을 충분히 못 잡는 정도를 가리키는 것은?

① 분산(variance)
② 편향(bias)
③ 정보 이득
④ 불순도
⑤ 학습률

> **정답** ②
> **해설** 편향(bias)은 모델이 너무 단순해서 데이터 패턴을 충분히 잡지 못하는 정도입니다. 편향이 크면 과소적합이 발생합니다(예: 로지스틱 회귀 78%).

47. 편향-분산 트레이드오프에서 모델이 훈련 데이터의 작은 변화에도 민감하게 반응하는 정도는?

① 편향(bias)
② 분산(variance)
③ 불순도
④ 정보 이득
⑤ 이너셔

> **정답** ②
>
> **해설** 분산(variance)은 모델이 훈련 데이터의 작은 변화에 민감하게 반응하는 정도입니다. 분산이 크면 과대적합이 발생합니다(예: 깊이 제한 없는 트리).

48. 편향과 분산의 관계에 대한 설명으로 옳은 것은?

① 둘 다 동시에 0으로 만들 수 있다
② 서로 반대 방향으로 움직여 동시에 0으로 만들 수 없고 균형점을 찾아야 한다
③ 둘은 항상 같이 커진다
④ 둘은 서로 무관하다
⑤ 분산만 줄이면 항상 최적이다

> **정답** ②
>
> **해설** 모델을 단순하게 하면 편향↑·분산↓, 복잡하게 하면 편향↓·분산↑로 둘은 반대로 움직입니다. 동시에 0으로 만들 수 없어 적절한 균형을 찾는 게 머신러닝의 핵심 게임입니다.

49. 결정 트리가 표준화 전처리가 필요 없는 근본적인 이유는?

① 거리를 계산하지 않기 때문에
② 분할 기준이 값의 절대 크기가 아니라 값의 순서에만 의존하기 때문에
③ 트리가 항상 깊게 자라기 때문에
④ 지니 불순도가 0~1이기 때문에
⑤ 샘플 수가 많기 때문에

> **정답** ②
>
> **해설** 결정 트리는 '특성값이 임계값보다 작은가?'라는 부등식으로 분할하므로 데이터의 절대 크기가 아니라 순서에만 의존합니다. 표준화는 순서를 보존하는 단조 변환이라 결과가 동일합니다.

50. 결정 트리가 단조 변환(monotonic transform)에 대해 불변(invariant)이라는 것의 의

미는?

① 트리 모양이 절대 바뀌지 않는다는 뜻
② 순서를 보존하는 변환(표준화 등)을 해도 트리의 결과가 같다는 뜻
③ 트리가 학습되지 않는다는 뜻
④ 불순도가 항상 일정하다는 뜻
⑤ 특성 개수가 고정된다는 뜻

> **정답** ②
>
> **해설** 단조 변환은 순서를 보존하는 변환이며, 표준화도 그 한 종류입니다. 트리는 순서에만 의존하므로 단조 변환을 해도 같은 결정을 내려 '단조 변환에 불변'이라고 합니다.

51. 전처리 관점에서 알고리즘 종류별 표준화 필요성을 한 줄로 요약하면?

① 모든 알고리즘이 표준화 필수
② 거리·가중치 기반은 표준화 필수, 트리 기반은 불필요
③ 트리 기반은 필수, 거리 기반은 불필요
④ 표준화는 항상 불필요
⑤ 회귀만 표준화가 필요

> **정답** ②
>
> **해설** 거리 기반(KNN)과 가중치 기반(로지스틱 회귀, 신경망)은 절대값에 민감해 표준화가 필수이고, 트리 기반은 순서에만 의존해 불필요합니다. 이 원리는 랜덤 포레스트, XGBoost 등 트리 계열 전반에 적용됩니다.

52. 결정 트리를 표준화하지 않은 원본 데이터로 학습할 때의 장점은?

① 성능이 항상 더 높아진다
② 분할 기준이 실제 특성값(예: sugar 1.625)으로 표시돼 해석이 쉬워진다
③ 학습이 불가능해진다
④ 과대적합이 사라진다
⑤ 특성 개수가 줄어든다

> **정답** ②
>
> **해설** 표준화 안 한 원본으로 학습하면 분할 기준이 'sugar -0.239' 같은 표준점수가 아니라 'sugar 1.625' 같은 실제 당도 값으로 표시되어, 비전공자에게 자연스럽게 설명할 수 있습니다.

53. 결정 트리의 feature_importances_ 속성에 대한 설명으로 옳은 것은?

① 각 특성의 평균값을 저장한다

② 각 특성이 분류에 기여한 정도를 비율로 나타내며 모두 더하면 1이 된다

③ 각 특성의 표준편차를 저장한다

④ 특성 개수를 저장한다

⑤ 음수 값을 가질 수 있다

> **정답** ②
>
> **해설** feature_importances_는 각 특성이 트리 전체에서 분류에 얼마나 기여했는지를 0~1 비율로 나타내며, 모두 더하면 정확히 1이 됩니다. 와인에서 [0.123, 0.869, 0.008]로 당도가 압도적이었습니다.

54. 와인 데이터에서 feature_importances_가 [0.123, 0.869, 0.008]이었다. 가장 중요한 특성은?

① 알코올 도수

② 당도(sugar)

③ pH

④ 세 특성이 동일

⑤ 판단할 수 없음

> **정답** ②
>
> **해설** 입력 순서가 alcohol, sugar, pH이므로 0.869인 두 번째 특성, 즉 당도(sugar)가 가장 중요합니다. 트리에서 루트와 깊이 1 노드가 모두 당도로 분할했던 것과 일치합니다.

55. 두 특성의 feature_importances_가 각각 0.87, 0.01로 나왔다. 가장 합리적인 해석과 활용은?

① 두 특성 모두 비슷하게 중요하다

② 첫 특성이 분류에 결정적이고 둘째는 거의 기여하지 않아, 둘째 특성 제거(차원 축소)를 고려할 수 있다

③ 둘째 특성이 더 중요하다

④ 중요도 합이 0.88이므로 모델이 잘못됐다

⑤ 특성 중요도는 해석에 쓸 수 없다

> **정답** ②
>
> **해설** 중요도가 클수록 그 특성으로 분할했을 때 불순도가 많이 줄었다는 뜻입니다. 0.87 대 0.01이면 첫 특성이 사실상 분류를 좌우하고 둘째는 거의 무의미하므로, 특성 선택·간소화에 활용할 수 있습니다.

56. **특성 중요도의 실무 활용처로 슬라이드가 든 것은?**

① 데이터 누수 방지
② 특성 선택(차원 축소)과 도메인 인사이트 제공
③ 학습률 조정
④ 테스트 세트 확대
⑤ 표준화 자동화

> **정답** ②
>
> **해설** 특성 중요도는 중요한 특성만 골라 모델을 가볍게 하는 특성 선택과, '와인 구분에는 당도가 결정적'이라는 식의 도메인 인사이트 제공에 활용됩니다.

57. **와인 정확도 사다리에서 '무조건 화이트라고 답하는' 베이스라인의 정확도가 76%인 이유는?**

① 로지스틱 회귀가 76%여서
② 데이터셋에 화이트 와인이 약 76%로 압도적으로 많기 때문
③ 동전 던지기 확률이 76%여서
④ 테스트 세트가 76%여서
⑤ 베이즈 오차가 76%여서

> **정답** ②
>
> **해설** 데이터셋에 화이트 76%, 레드 24%로 클래스 불균형이 있어, 아무 모델 없이 모두 화이트라고 답해도 76%는 맞습니다. 어떤 모델이든 이 76%는 넘어야 의미가 있습니다.

58. **데이터 자체에 내재된, 어떤 모델도 넘을 수 없는 이론적 정확도 한계를 가리키는 용어는?**

① 과대적합 한계
② 베이즈 오차(Bayes error)
③ 편향 한계
④ 분산 한계
⑤ 이너셔

> **정답** ②
>
> **해설** 베이즈 오차(Bayes error)는 데이터에 내재된 본질적 한계로, 같은 특성값이라도 두 클래스가 모두 가능한 경우 때문에 발생합니다. 3개 특성 와인 데이터의 이론적 상한은 약 90~95%로 추정됩니다.

59. 결정 트리가 어떤 샘플에 대해 내놓는 '클래스 예측 확률(predict_proba)'은 어떻게 계산되는가?

① 전체 훈련 데이터의 클래스 비율
② 그 샘플이 도달한 리프 노드 안에서 각 클래스가 차지하는 비율
③ 항상 0 또는 1
④ 특성 개수의 역수
⑤ 트리 깊이에 비례

> **정답** ②
> **해설** 결정 트리의 예측 확률은 그 샘플이 도달한 리프 노드의 클래스별 샘플 비율입니다. 리프 value가 [10, 90]이면 두 번째 클래스 확률이 90/100=0.9입니다.

60. 두 클래스 비율이 76:24인 불균형 데이터에서 '정확도(accuracy)'만으로 모델을 평가할 때 주의할 점은?

① 정확도가 높으면 항상 좋은 모델이다
② 무조건 다수 클래스로만 찍어도 76%가 나오므로, 정확도가 높다고 좋은 모델이라 단정할 수 없다
③ 정확도는 불균형과 무관하다
④ 정확도는 항상 50%가 된다
⑤ 불균형이면 정확도를 계산할 수 없다

> **정답** ②
> **해설** 불균형 데이터에서는 다수 클래스로만 예측해도 그 비율(76%)만큼 정확도가 나옵니다. 따라서 베이스라인과 비교하거나 정밀도·재현율·F1 같은 지표를 함께 봐야 모델의 실제 가치를 판단할 수 있습니다.

5-2 교차 검증과 그리드 서치

61. 테스트 세트를 사용하지 않고 모델을 평가하기 위해 훈련 세트에서 다시 떼어낸 데이터 세트를 무엇이라 하는가?

① 검증 세트(validation set)
② 부트스트랩 샘플
③ OOB 샘플
④ 폴드
⑤ 센트로이드

> **정답** ①
>
> **해설** 검증 세트(validation set, 또는 개발 세트/dev set)는 하이퍼파라미터 튜닝 과정에서 테스트 세트를 쓰지 않기 위해 훈련 세트에서 다시 떼어낸 데이터입니다.

62. 검증 세트를 만들 때 train_test_split을 두 번 호출하는 이유는?

① 테스트 세트를 두 개로 나누려고

② 전체를 훈련/테스트로 나눈 뒤, 다시 훈련 세트를 sub_input(학습용)과 val_input(검증용)으로 나누려고

③ 데이터를 두 배로 늘리려고

④ 랜덤 시드를 두 개 만들려고

⑤ 특성을 둘로 나누려고

> **정답** ②
>
> **해설** 첫 번째 호출로 전체를 훈련/테스트(8:2)로 나누고, 두 번째 호출로 훈련 세트를 다시 sub_input(실제 학습용)과 val_input(검증용)으로 나눕니다. test_input은 손대지 않습니다.

63. 교차 검증을 사용하면 데이터에서 검증 세트를 미리 한 덩어리 떼어 두지 않아도 되는 이유는?

① 검증이 필요 없어서

② 훈련 세트를 여러 폴드로 나눠 번갈아 가며 검증에 사용하므로 데이터를 고정으로 떼어 둘 필요가 없기 때문

③ 테스트 세트를 검증에 쓰기 때문

④ 모델이 스스로 검증하기 때문

⑤ 데이터가 무한하기 때문

> **정답** ②
>
> **해설** 교차 검증은 훈련 세트를 k개 폴드로 나눠 한 폴드를 검증, 나머지를 훈련에 쓰는 일을 k번 돌립니다. 모든 데이터가 한 번씩 검증에 쓰이므로 별도의 고정 검증 세트가 필요 없고, 데이터가 적을 때 특히 유리합니다.

64. 5-폴드 교차 검증 점수가 [0.84, 0.86, 0.85, 0.83, 0.87]로 나왔다. 보통 보고하는 '교차 검증 점수'는?

① 가장 높은 0.87

② 다섯 점수의 평균(약 0.85)

③ 가장 낮은 0.83

④ 마지막 폴드 점수 0.87

⑤ 다섯 점수의 합 4.25

> **정답** ②
>
> **해설** 교차 검증 점수는 각 폴드 검증 점수의 평균으로 보고합니다. 평균을 쓰면 한 번의 우연한 분할에 휘둘리지 않고 모델 성능을 더 안정적으로 추정할 수 있으며, 점수들의 표준편차로 변동성도 함께 볼 수 있습니다.

65. 검증 세트를 한 번만 떼어 평가하는 방식의 약점은?

① 계산이 너무 느리다

② 검증 세트를 어떻게 나누느냐(운)에 따라 점수가 달라져 불안정하다

③ 테스트 세트가 오염된다

④ 과소적합이 발생한다

⑤ 특성이 줄어든다

> **정답** ②
>
> **해설** 검증 세트 한 번 평가는 분할 운에 따라 점수가 달라져 불안정합니다. 쉬운 샘플만 들어가면 부풀려지고 어려운 샘플이 들어가면 낮게 나옵니다. 이를 해결하는 것이 교차 검증입니다.

66. 교차 검증(cross-validation)의 핵심 정의로 옳은 것은?

① 테스트 세트로 여러 번 평가한다

② 검증 세트를 떼어 평가하는 과정을 여러 번 반복하고 그 점수들을 평균한다

③ 데이터를 무작위로 복제한다

④ 학습률을 점진적으로 낮춘다

⑤ 특성을 압축한다

> **정답** ②
>
> **해설** 교차 검증은 검증 세트를 한 번만 떼는 게 아니라 여러 번 떼서 평가를 여러 번 하고 점수를 평균내는 방법입니다. 훨씬 안정적인 평가가 가능합니다.

67. 5-폴드 교차 검증을 한 번 수행하면 모델을 몇 번 학습시키는가?

① 1번

② 2번

③ 5번

④ 10번

⑤ 25번

정답 ③

해설 5-폴드 교차 검증은 데이터를 5등분해 한 폴드씩 검증으로 쓰며 5번의 평가를 하므로, 모델을 5번 처음부터 새로 학습시킵니다. 각 라운드는 독립적입니다.

68. 교차 검증이 한 번 검증보다 좋은 두 가지 이유로 옳은 것은?

① 속도와 메모리

② 안정성(우연 상쇄)과 데이터 활용도(모든 데이터가 학습·검증에 사용됨)

③ 정확도와 정밀도

④ 편향과 분산

⑤ 압축과 복원

정답 ②

해설 교차 검증은 점수를 평균내 우연을 상쇄하는 안정성과, 모든 데이터가 한 번씩 학습과 검증에 쓰이는 데이터 활용도 향상이라는 두 가지 이점이 있습니다.

69. 실무에서 교차 검증의 폴드 수로 가장 흔히 쓰이는 값은?

① 2-폴드

② 3-폴드

③ 5-폴드 또는 10-폴드

④ 100-폴드

⑤ 샘플 수만큼

정답 ③

해설 실무에서는 보통 5-폴드나 10-폴드가 표준입니다. 폴드가 많을수록 안정적이지만 학습 횟수가 늘어 시간이 더 걸리므로, 5나 10이 시간과 안정성의 균형점입니다.

70. 교차 검증은 무엇을 위해 사용하며, 최종 성능 평가는 무엇으로 하는가?

① 최종 평가용이며 검증 세트로 한다

② 매개변수 튜닝용이며 최종 평가는 별도 테스트 세트로 한다

③ 학습용이며 평가는 훈련 세트로 한다

④ 압축용이며 평가는 PCA로 한다

⑤ 교차 검증으로 최종 평가도 한다

정답 ②

해설 교차 검증은 매개변수 튜닝에 쓰는 것이고, 최종 성능 평가는 여전히 별도의 테스트 세트로 합니다. 데이터를 훈련/테스트로 나누고, 훈련 안에서 교차 검증으로 매개변수를 정한 뒤, 마

지막에 테스트로 평가합니다.

71. 사이킷런에서 교차 검증을 수행하는 함수는?

① cross_validate()

② train_test_split()

③ GridSearchCV()

④ StandardScaler()

⑤ KMeans()

정답 ①

해설 from sklearn.model_selection import cross_validate로 가져와 cross_validate(모델, 입력, 타깃) 형태로 호출합니다. 내부적으로 데이터를 폴드로 나눠 평가를 반복합니다.

72. cross_validate에 sub_input/val_input이 아니라 train_input 전체를 넣는 이유는?

① sub_input이 너무 작아서

② cross_validate가 내부적으로 알아서 폴드를 나누고 검증을 반복하므로 직접 나눌 필요가 없어서

③ 검증 세트가 필요 없어서

④ 테스트 세트를 쓰려고

⑤ train_input이 더 정확해서

정답 ②

해설 cross_validate는 내부적으로 데이터를 폴드로 나누고 한 폴드씩 검증으로 사용하며 평가를 반복하므로, 우리가 직접 sub와 val로 나눌 필요가 없습니다. train_input 전체를 넘기면 됩니다.

73. cross_validate가 반환하는 딕셔너리의 세 가지 키는?

① mean, std, var

② fit_time, score_time, test_score

③ loss, accuracy, f1

④ coef, intercept, classes

⑤ min, max, mean

정답 ②

해설 cross_validate는 fit_time(학습 시간), score_time(검증 시간), test_score(각 폴드의 검증 점수)를 키로 가진 딕셔너리를 반환합니다. 각 키마다 5개(기본 5-폴드)의 숫자가 들어 있습니다.

74. cross_validate 결과의 test_score 키가 의미하는 것은?

① 최종 테스트 세트의 점수
② 각 폴드의 검증 점수
③ 훈련 점수
④ 학습 시간
⑤ 테스트 세트 크기

> **정답** ②
>
> **해설** cross_validate에서 test_score는 '각 폴드의 검증 점수'를 의미합니다. 흔히 말하는 별도 '테스트 세트의 점수'와는 다른, 사이킷런 명명의 다소 어색한 부분입니다.

75. cross_validate 결과에서 최종 교차 검증 점수를 구하는 방법은?

① test_score의 최댓값
② test_score 5개 값을 평균(np.mean)
③ test_score의 합
④ fit_time의 평균
⑤ score_time의 최솟값

> **정답** ②
>
> **해설** np.mean(scores['test_score'])로 5개 폴드의 검증 점수를 평균낸 값이 최종 교차 검증 점수입니다. 와인 결정 트리에서 약 0.8553이 나왔습니다.

76. cross_validate가 분류 모델에 대해 기본적으로 사용하는 폴드 분할기는?

① KFold
② StratifiedKFold
③ GroupKFold
④ TimeSeriesSplit
⑤ ShuffleSplit

> **정답** ②
>
> **해설** cross_validate는 회귀 모델에는 KFold를, 분류 모델에는 타깃 클래스 비율을 골고루 유지하는 StratifiedKFold를 기본으로 사용합니다.

77. StratifiedKFold가 KFold와 다른 핵심 특징은?

① 데이터를 무작위로 섞지 않는다
② 각 폴드의 클래스 비율이 전체 데이터의 클래스 비율과 같아지도록 분할한다

③ 폴드 수가 항상 10개다

④ 회귀 전용이다

⑤ 더 빠르다

> **정답** ②
>
> **해설** StratifiedKFold('층화')는 각 폴드 안의 클래스 비율이 전체와 같아지도록 분할합니다. 와인의 화이트 76%, 레드 24% 비율이 각 폴드에서도 유지되어, 불균형 데이터의 평가 불안정을 막습니다.

78. StratifiedKFold(n_splits=10, shuffle=True, random_state=42)에서 shuffle=True의 역할은?

① 폴드 수를 늘린다

② 분할 전에 데이터를 섞는다

③ 클래스 비율을 무시한다

④ 테스트 세트를 섞는다

⑤ 특성을 섞는다

> **정답** ②
>
> **해설** shuffle=True는 분할 전에 데이터를 섞으라는 의미입니다. 데이터가 어떤 순서로 정렬돼 있을지 모르므로 무조건 섞어서 분할하는 것이 안전하며, n_splits=10은 10-폴드, random_state=42는 섞기 시드입니다.

79. 머신러닝 모델이 데이터로부터 스스로 학습하는 값을 무엇이라 하는가?

① 하이퍼파라미터

② 모델 파라미터

③ 검증 점수

④ 이너셔

⑤ 폴드

> **정답** ②
>
> **해설** 모델 파라미터는 머신러닝 모델이 학습 과정에서 스스로 찾는 값(예: 로지스틱 회귀의 계수와 절편)입니다.

80. 모델이 학습할 수 없어서 사용자가 직접 지정해야 하는 값을 무엇이라 하는가?

① 모델 파라미터

② 하이퍼파라미터

③ 타깃

④ 특성

⑤ 레이블

> **정답** ②
>
> **해설** 하이퍼파라미터는 모델이 학습할 수 없어 사용자가 지정해야 하는 값입니다. 예: max_depth, min_samples_split, n_estimators 등. 이를 조정하는 것이 하이퍼파라미터 튜닝입니다.

81. 그리드 서치(Grid Search)를 사용하는 핵심 이유는?

① 데이터를 압축하려고

② 여러 매개변수의 최적값이 서로 영향을 주므로 여러 매개변수를 동시에 바꿔가며 최적 조합을 자동 탐색하려고

③ 테스트 세트를 늘리려고

④ 학습률을 낮추려고

⑤ 표준화를 자동화하려고

> **정답** ②
>
> **해설** max_depth의 최적값은 min_samples_split 값이 바뀌면 함께 달라지므로, 여러 매개변수를 동시에 바꿔가며 최적 조합을 찾아야 합니다. 그리드 서치는 이를 자동화합니다.

82. 사이킷런에서 하이퍼파라미터 탐색과 교차 검증을 한 번에 수행하는 클래스는?

① cross_validate

② GridSearchCV

③ StandardScaler

④ DecisionTreeClassifier

⑤ train_test_split

> **정답** ②
>
> **해설** GridSearchCV 클래스는 매개변수 후보들의 모든 조합에 대해 자동으로 교차 검증을 수행해 가장 좋은 조합을 찾아줍니다. 이름의 CV가 교차 검증(Cross Validation)을 의미합니다.

83. GridSearchCV에서 n_jobs=-1로 지정하는 것의 의미는?

① 작업을 하지 않는다

② 모든 CPU 코어를 사용해 병렬 처리한다

③ 1개 코어만 사용한다

④ 음수 인덱스를 쓴다

⑤ 학습을 중단한다

> **정답** ②
>
> **해설** n_jobs=-1은 컴퓨터의 모든 CPU 코어를 활용해 병렬 처리하라는 의미입니다. 매개변수 조합이 많아도 의외로 빠르게 끝납니다.

84. GridSearchCV로 1,350개 조합을 5-폴드로 탐색하면 총 모델 학습 횟수는?

① 1,350번

② 5번

③ 6,750번

④ 270번

⑤ 1,355번

> **정답** ③
>
> **해설** 매개변수 한 조합당 5-폴드면 5번 학습하므로, 1,350 × 5 = 6,750번 학습합니다. 5-폴드 교차 검증이 그리드 서치 시간 견적의 기준이 되는 이유입니다.

85. GridSearchCV 학습 후 최적 매개변수 조합이 저장되는 속성은?

① best_score_

② best_params_

③ best_estimator_

④ cv_results_

⑤ n_iter_

> **정답** ②
>
> **해설** gs.best_params_에 최적 매개변수 조합이 저장됩니다(예: {'max_depth': 14, ...}). 최적 모델은 best_estimator_, 상세 결과는 cv_results_에 저장됩니다.

86. 랜덤 서치(Random Search)가 필요한 상황으로 옳은 것은?

① 데이터가 작을 때만

② 매개변수 값의 범위/간격을 미리 정하기 어렵거나 조합이 너무 많아 그리드 서치가 오래 걸릴 때

③ 타깃이 없을 때

④ 특성이 1개일 때

⑤ 표준화가 안 됐을 때

정답 ②

해설 랜덤 서치는 ① 매개변수 값의 범위/간격을 미리 정하기 어려운 연속값일 때, ② 매개변수 조합이 너무 많아 그리드 서치 시간이 오래 걸릴 때 유용합니다.

87. 랜덤 서치에 매개변수로 전달하는 것은?

① 매개변수 값의 목록(리스트)

② 매개변수를 샘플링할 확률 분포 객체

③ 최적값 하나

④ 교차 검증 점수

⑤ 테스트 세트

정답 ②

해설 랜덤 서치에는 값의 목록이 아니라 매개변수를 샘플링할 수 있는 확률 분포 객체(scipy.stats의 uniform, randint 등)를 전달합니다. 그 분포에서 무작위로 값을 뽑아 시도합니다.

88. scipy.stats의 randint와 uniform의 차이로 옳은 것은?

① randint는 실수, uniform은 정수를 추출

② randint는 정숫값, uniform은 실숫값을 균등 분포에서 추출

③ 둘 다 정규 분포에서 추출

④ randint는 분류용, uniform은 회귀용

⑤ 둘 다 같은 값을 추출

정답 ②

해설 둘 다 주어진 범위에서 고르게(균등 분포) 값을 추출하지만, randint는 정숫값을, uniform은 실숫값을 추출합니다. max_depth 같은 정수 매개변수에는 randint, learning_rate 같은 실수에는 uniform을 씁니다.

89. RandomizedSearchCV에서 n_iter=100의 의미는?

① 100개 폴드를 만든다

② 확률 분포에서 100번 무작위로 매개변수를 뽑아 시도한다

③ 100번 반복 학습한다

④ 100개 특성을 쓴다

⑤ 100% 데이터를 쓴다

정답 ②

해설 n_iter=100은 '100번 무작위로 매개변수를 뽑아 시도하라'는 의미입니다. 그리드 서치의

1,350번보다 훨씬 적은 시도로도, 차원이 높을수록 랜덤 서치가 더 효율적인 경향이 있습니다.

90. uniform(0.0001, 0.001)로 분포 객체를 만들 때 두 인자의 의미는?

① 시작값과 종료값

② 시작값과 폭(width)

③ 평균과 표준편차

④ 최솟값과 개수

⑤ 정수와 실수

정답 ②

해설 uniform의 두 인자는 시작값과 폭(width)입니다. uniform(0.0001, 0.001)은 0.0001부터 0.0001+0.001=0.0011까지의 균등 분포가 됩니다. 시작값과 종료값이 아님에 주의해야 합니다.

91. 랜덤 서치가 그리드 서치보다 효율적일 수 있는 이유에 대한 설명으로 옳은 것은?

① 항상 더 많은 조합을 시도하기 때문

② 그리드는 중요하지 않은 매개변수에도 자원을 균등 배분하지만, 랜덤은 중요한 매개변수에 다양한 값이 시도되는 효과가 있어서

③ 랜덤이 항상 전역 최적을 보장하기 때문

④ 교차 검증을 안 하기 때문

⑤ 데이터를 압축하기 때문

정답 ②

해설 그리드 서치는 모든 매개변수에 같은 양의 자원을 배분해 안 중요한 매개변수에도 시간을 낭비합니다. 랜덤 서치는 우연히 중요한 매개변수에 다양한 값이 시도되어, 같은 횟수로 더 좋은 결과를 낼 수 있습니다.

92. min_samples_leaf 매개변수가 의미하는 것은?

① 루트 노드의 최소 샘플 수

② 리프 노드가 가져야 하는 최소 샘플 수

③ 트리의 최대 깊이

④ 특성의 최소 개수

⑤ 폴드의 최소 개수

정답 ②

해설 min_samples_leaf는 리프 노드의 최소 샘플 수입니다. 분할 결과 자식 노드에 이 값보다 적은 샘플이 들어가면 그 분할이 거부됩니다. min_samples_split과 함께 트리 단순화에 기여합니

다.

93. Train-Validation-Test split에서 테스트 세트는 언제 사용해야 하는가?

① 매개변수 튜닝 과정에서 자주
② 학습 과정에서
③ 모든 매개변수 결정이 끝난 후 최종 성능 평가에서 딱 한 번
④ 검증 점수가 낮을 때마다
⑤ 교차 검증의 매 폴드마다

정답 ③

해설 테스트 세트는 모든 학습과 매개변수 결정이 끝난 뒤 최종 성능을 판단하는 마지막 시험으로 딱 한 번만 사용합니다. 튜닝 과정에 끼어들면 데이터 누수가 발생합니다.

94. 랜덤 서치 후 검증 점수(0.8695)와 테스트 점수(0.86)가 거의 비슷한 것이 의미하는 바는?

① 과대적합이 심하다
② 매개변수 튜닝 과정에 큰 데이터 누수가 없었다
③ 모델이 너무 단순하다
④ 테스트 세트가 잘못됐다
⑤ 교차 검증이 실패했다

정답 ②

해설 검증 점수와 테스트 점수가 비슷하면 튜닝 과정에 큰 데이터 누수가 없었고, 검증 세트에 과대적합하지 않았다는 뜻입니다. 그 매개변수가 실제 일반화 성능을 잘 대표합니다.

95. cross_validate에서 return_train_score=True 옵션의 효과는?

① 테스트 세트 점수를 반환한다
② 각 폴드의 훈련 점수도 함께 반환한다
③ 학습 시간을 줄인다
④ 폴드 수를 늘린다
⑤ 표준화를 자동 적용한다

정답 ②

해설 return_train_score=True를 주면 train_score도 반환되어, 훈련 점수와 검증 점수의 격차로 과대적합/과소적합을 진단할 수 있습니다. 5-3 앙상블에서 자주 사용합니다.

96. GridSearchCV와 RandomizedSearchCV의 사용 가이드라인으로 옳은 것은?

① 둘은 완전히 동일하다

② 매개변수가 적고 이산적이면 그리드 서치, 많거나 연속적이면 랜덤 서치

③ 그리드는 회귀, 랜덤은 분류 전용

④ 랜덤은 항상 그리드보다 정확하다

⑤ 둘 다 교차 검증을 하지 않는다

> **정답** ②
>
> **해설** 일반적으로 매개변수가 적고 이산적이면 그리드 서치(빈틈없는 격자 탐색), 많거나 연속적이면 랜덤 서치(확률 분포 샘플링)를 사용합니다.

97. 다음 중 교차 검증을 수행하지 않는 함수/클래스는?

① cross_validate()

② GridSearchCV

③ RandomizedSearchCV

④ train_test_split

⑤ StratifiedKFold를 cv로 받은 cross_validate

> **정답** ④
>
> **해설** train_test_split은 단순히 데이터를 훈련/테스트로 한 번 나누는 함수로 교차 검증을 하지 않습니다. cross_validate, GridSearchCV, RandomizedSearchCV는 모두 교차 검증을 수행합니다.

98. 교차 검증(또는 검증) 점수는 높은데 테스트 점수가 그보다 많이 낮다면, 가장 의심되는 상황은?

① 데이터가 너무 많다

② 검증 점수에 맞춰 하이퍼파라미터를 과하게 튜닝해 검증 데이터에 과대적합되었을 가능성

③ 모델이 과소적합되었다

④ 테스트 세트가 너무 크다

⑤ 교차 검증을 하면 안 됐다

> **정답** ②
>
> **해설** 검증 점수만 보고 하이퍼파라미터를 지나치게 맞추면 검증 데이터에 과대적합되어, 진짜 새 데이터인 테스트에서 성능이 떨어집니다. 그래서 테스트 세트는 모든 튜닝이 끝난 뒤 단 한 번만 사용해야 합니다.

5-3 트리의 앙상블

99. 정형 데이터(structured data)에서 가장 강력한 성능을 내는 머신러닝 알고리즘 계열은?

① 합성곱 신경망(CNN)
② 결정 트리 기반 앙상블 학습
③ 순환 신경망(RNN)
④ 트랜스포머
⑤ k-최근접 이웃(KNN)

> **정답** ②
>
> **해설** 정형 데이터(표 형태)에는 결정 트리 기반 앙상블이 가장 강력합니다. 캐글·데이콘 대회 정형 데이터 우승 솔루션의 대부분이 XGBoost, LightGBM 같은 트리 앙상블입니다. 비정형 데이터에는 신경망이 강합니다.

100. 이미지·텍스트·음성 같은 비정형 데이터에 강한 알고리즘 계열은?

① 결정 트리
② 랜덤 포레스트
③ 신경망(CNN, Transformer 등)
④ 선형 회귀
⑤ k-평균

> **정답** ③
>
> **해설** 비정형 데이터에는 신경망이 강합니다. 이미지에는 CNN, 텍스트에는 Transformer, 음성에는 RNN/Conformer 등을 사용하며, 7장 이후 본격적으로 배웁니다.

101. 앙상블 학습(ensemble learning)의 정의로 옳은 것은?

① 하나의 강력한 모델을 깊게 학습하는 것
② 더 좋은 예측을 위해 여러 개의 모델을 함께 훈련·결합하는 것
③ 데이터를 여러 폴드로 나누는 것
④ 특성을 압축하는 것
⑤ 타깃 없이 군집하는 것

> **정답** ②
>
> **해설** 앙상블 학습은 더 좋은 예측 결과를 만들기 위해 여러 개의 모델을 훈련해 결합하는 머신러닝 알고리즘입니다. 핵심은 다양성으로, 서로 다른 약점을 가진 모델들이 서로 보완합니다.

102. **약하게 정확한 분류기 여러 개를 다수결로 합치면 정확도가 1에 수렴한다는 18세기 통계학 정리는?**

① 베이즈 정리
② 콩도르세(Condorcet)의 배심원 정리
③ 중심극한정리
④ 큰 수의 법칙
⑤ 섀넌의 정보 정리

> **정답** ②
>
> **해설** 콩도르세의 배심원 정리(1785)는 51% 정확한 분류기를 N개 모아 다수결하면 N이 커질수록 정확도가 100%에 수렴한다는 것입니다. 단, 오차가 서로 독립적일 때만 성립하므로 '다양성'이 핵심입니다.

103. **앙상블이 효과를 내기 위한 가장 중요한 조건은?**

① 모든 모델이 동일할 것
② 모델들이 서로 다양하고 오차가 독립적일 것
③ 모델 수가 정확히 100개일 것
④ 모든 모델이 100% 정확할 것
⑤ 데이터가 표준화될 것

> **정답** ②
>
> **해설** 앙상블이 작동하려면 모델들이 서로 달라야(다양성) 하고, 오차가 서로 독립적(비상관)이어야 합니다. 똑같은 모델 여러 개는 한 명한테 여러 번 묻는 것과 같아 의미가 없습니다.

104. **랜덤 포레스트(Random Forest)에 대한 설명으로 옳은 것은?**

① 하나의 깊은 결정 트리를 만든다
② 무작위로 만든 여러 결정 트리(숲)의 예측을 결합하는 대표적 앙상블 알고리즘이다
③ 신경망의 일종이다
④ 차원 축소 알고리즘이다
⑤ 거리 기반 알고리즘이다

> **정답** ②
>
> **해설** 랜덤 포레스트는 결정 트리를 랜덤하게 만들어 트리의 숲을 만들고, 각 트리의 예측을 결합해 최종 예측을 도출하는 대표적 앙상블 알고리즘입니다. 안정적인 성능으로 베이스라인 모델로 자주 쓰입니다.

105. **사이킷런에서 랜덤 포레스트는 기본적으로 몇 개의 결정 트리를 만드는가?**

① 10개
② 50개
③ 100개
④ 256개
⑤ 1000개

> **정답** ③
>
> **해설** 사이킷런 RandomForestClassifier의 기본값 n_estimators는 100으로, 기본적으로 100그루의 결정 트리를 만듭니다.

106. **부트스트랩 샘플(bootstrap sample)의 정의로 옳은 것은?**

① 중복 없이 절반만 뽑는 샘플
② 데이터셋에서 중복을 허용해 훈련 세트와 같은 크기로 뽑는 샘플
③ 테스트 세트
④ 검증 세트
⑤ 표준화된 샘플

> **정답** ②
>
> **해설** 부트스트랩 샘플은 데이터셋에서 중복을 허용하며 샘플링하는 방식으로, 기본적으로 훈련 세트와 같은 크기로 만듭니다. 같은 샘플이 여러 번 뽑힐 수 있습니다.

107. **부트스트랩 샘플링에서 통계적으로 한 번도 뽑히지 않는 샘플의 비율은 약 몇 %인가?**

① 약 10%
② 약 37%
③ 약 50%
④ 약 63%
⑤ 약 90%

> **정답** ②
>
> **해설** 부트스트랩 샘플링에서 약 63%의 샘플이 적어도 한 번 뽑히고, 약 37%는 한 번도 뽑히지 않습니다. 이 37%를 OOB(Out Of Bag) 샘플이라고 합니다.

108. **RandomForestClassifier가 각 노드 분할에서 기본적으로 고려하는 특성 개수는?**

① 전체 특성
② 전체 특성의 제곱근(sqrt)만큼

③ 전체 특성의 절반

④ 특성 1개

⑤ 전체 특성의 제곱

> **정답** ②
>
> **해설** 분류 모델 RandomForestClassifier는 기본적으로 전체 특성 개수의 제곱근만큼을 무작위로 선택합니다. 특성이 16개면 4개를 사용합니다. (회귀 모델 RandomForestRegressor는 전체 특성을 사용합니다.)

109. 랜덤 포레스트가 100그루 트리의 다양성을 확보하는 두 가지 메커니즘은?

① 깊이 제한과 가지치기

② 부트스트랩 샘플(데이터 무작위성)과 특성 무작위 선택

③ 표준화와 정규화

④ 교차 검증과 그리드 서치

⑤ 학습률과 손실 함수

> **정답** ②
>
> **해설** 랜덤 포레스트는 ① 각 트리를 다른 부트스트랩 샘플로 학습(데이터 무작위성)하고 ② 각 노드에서 일부 특성만 무작위 선택(특성 무작위성)하여 트리들의 다양성을 확보합니다.

110. 랜덤 포레스트가 분류에서 최종 예측을 만드는 방식은?

① 트리들의 예측값을 모두 더한다

② 각 트리의 클래스별 확률을 평균해 가장 높은 확률의 클래스를 예측한다(사실상 다수결)

③ 가장 깊은 트리의 예측만 사용한다

④ 분산이 큰 방향으로 투영한다

⑤ 거리를 최소화한다

> **정답** ②
>
> **해설** 분류에서는 각 트리가 클래스별 확률을 출력하고, 이를 평균낸 뒤 가장 높은 확률의 클래스를 최종 예측으로 합니다(사실상 다수결). 회귀에서는 각 트리의 예측값을 단순 평균합니다.

111. 랜덤 포레스트가 단일 결정 트리보다 일반화 성능이 좋은 이유는?

① 트리를 더 깊게 만들기 때문

② 여러 트리가 각자 다른 패턴(노이즈 포함)을 학습하고 평균내면 우연한 노이즈는 상쇄되고 진짜 패턴만 남기 때문

③ 표준화를 하기 때문

④ 특성을 압축하기 때문

⑤ 학습률을 낮추기 때문

> **정답** ②
>
> **해설** 100그루가 각자 다른 부트스트랩 샘플로 서로 다른 패턴을 외우는데, 그 다양한 외움을 평균내면 우연한 노이즈는 상쇄되고 진짜 패턴만 남아 일반화 성능이 좋아집니다. 이것이 앙상블의 마법입니다.

112. OOB(Out Of Bag) 점수의 핵심 아이디어는?

① 테스트 세트로 평가한다

② 각 트리의 부트스트랩에 포함되지 않은 OOB 샘플을 그 트리의 검증 세트로 활용해 일반화 성능을 추정한다

③ 전체 데이터로 학습 점수를 낸다

④ 교차 검증을 5번 반복한다

⑤ 분산을 계산한다

> **정답** ②
>
> **해설** OOB 샘플은 그 트리 입장에서 '처음 보는 데이터'이므로 검증 세트 역할을 할 수 있습니다. OOB 점수는 별도 검증 세트나 교차 검증 없이도 자체적으로 일반화 성능을 추정하는 우아한 방법입니다.

113. OOB 점수를 사용하려면 어떻게 설정해야 하는가?

① oob_score=True로 지정

② n_jobs=-1로 지정

③ 기본값으로 자동 활성화

④ cross_validate를 호출

⑤ random_state를 None으로 설정

> **정답** ①
>
> **해설** OOB 점수는 기본값 False라서 oob_score=True로 명시적으로 켜야 합니다. 추가 계산 시간이 걸리기 때문입니다. 학습 후 rf.oob_score_ 속성으로 확인합니다.

114. OOB 점수의 장점으로 옳은 것은?

① 테스트 세트가 필요하다

② 한 번의 학습으로 평가까지 끝나 시간을 절약하고, 모든 훈련 데이터를 학습에 쓰면서도 평가가 가능하다

③ 항상 교차 검증보다 정확하다

④ 트리 개수를 줄여준다

⑤ 표준화를 자동화한다

> **정답** ②
>
> **해설** OOB 점수는 ① 한 번의 학습으로 평가까지 되어 5-폴드 교차 검증(5번 학습)보다 시간을 절약하고, ② 모든 훈련 데이터를 학습에 쓰면서도 평가가 가능합니다. 통계적으로 교차 검증과 비슷한 신뢰도를 가집니다.

115. 엑스트라 트리(Extra Trees)가 랜덤 포레스트와 다른 핵심 차이점은?

① 트리를 1그루만 만든다

② 부트스트랩 샘플을 사용하지 않고, 분할 기준값도 무작위로 정한다

③ 신경망을 사용한다

④ 표준화를 자동으로 한다

⑤ 교차 검증을 내장한다

> **정답** ②
>
> **해설** 엑스트라 트리(Extremely Randomized Trees)는 부트스트랩 샘플을 사용하지 않고(모든 트리가 전체 훈련 세트 사용), 분할 기준값도 무작위로 정합니다. 이 '극단적 무작위화'로 다양성을 만듭니다.

116. 엑스트라 트리가 랜덤 포레스트보다 학습이 빠른 이유는?

① 트리를 적게 만들어서

② 분할 기준값을 무작위로 정해 모든 분할 후보를 평가하지 않으므로

③ 표준화를 생략해서

④ 교차 검증을 안 해서

⑤ 특성을 압축해서

> **정답** ②
>
> **해설** 엑스트라 트리는 일부 특성을 고른 뒤 분할 기준값을 무작위로 정하고 최선 여부를 따지지 않습니다. 모든 분할 후보를 평가하지 않으므로 랜덤 포레스트보다 학습이 빠릅니다.

117. 랜덤 포레스트와 엑스트라 트리가 결정 트리보다 당도(특성)에 대한 의존성이 낮게 나오는 이유는?

① 당도를 일부러 제거해서

② 매 노드에서 특성을 일부만 무작위 선택하므로 다른 특성도 분할에 기여할 기회를 얻기

때문

③ 표준화를 했기 때문

④ 트리가 얕기 때문

⑤ 학습률이 낮기 때문

> **정답** ②
>
> **해설** 일반 결정 트리는 매 노드에서 모든 특성을 보고 가장 강한 당도를 계속 선택합니다. 랜덤 포레스트/엑스트라 트리는 특성을 일부만 무작위로 봐서, 당도가 후보에 없을 때 다른 특성이 쓰여 의존성이 분산됩니다.

118. 배깅(Bagging) 계열에 속하는 앙상블 알고리즘은?

① 그레이디언트 부스팅

② 랜덤 포레스트와 엑스트라 트리

③ XGBoost

④ LightGBM

⑤ 히스토그램 기반 그레이디언트 부스팅

> **정답** ②
>
> **해설** 랜덤 포레스트와 엑스트라 트리가 배깅 계열입니다. 배깅은 여러 트리를 독립적·병렬로 만들어 평균/다수결로 결합하며, 분산 감소에 강합니다.

119. 부스팅(Boosting)과 배깅(Bagging)의 핵심 차이는?

① 부스팅은 병렬, 배깅은 순차

② 배깅은 트리를 독립·병렬로 만들고, 부스팅은 트리를 순차적으로 만들어 이전 트리의 오차를 보완한다

③ 둘 다 동일하다

④ 부스팅은 회귀, 배깅은 분류 전용

⑤ 배깅은 신경망 기반이다

> **정답** ②
>
> **해설** 배깅은 트리들을 독립적·병렬로 만들어 다수결/평균으로 합칩니다. 부스팅은 트리들을 순차적으로 만들며 각 트리가 이전 트리들의 오차를 보완합니다. 배깅은 분산, 부스팅은 편향 감소에 강합니다.

120. 그레이디언트 부스팅(Gradient Boosting)에 대한 설명으로 옳은 것은?

① 깊은 트리를 독립적으로 만든다

② 깊이가 얕은 결정 트리를 사용해 이전 트리의 오차를 보완하며 경사 하강법으로 앙상블에 추가한다

③ 부트스트랩 샘플을 사용한다

④ 차원 축소 알고리즘이다

⑤ 거리를 최소화한다

> **정답** ②
>
> **해설** 그레이디언트 부스팅은 깊이가 얕은 결정 트리(약한 학습기)를 사용해 이전 트리의 오차를 보완하는 방식으로 앙상블하며, 경사 하강법으로 트리를 추가합니다. 분류는 로지스틱 손실, 회귀는 평균 제곱 오차를 사용합니다.

121. 그레이디언트 부스팅의 트리 깊이(max_depth) 기본값은 보통 얼마이며 그 이유는?

① 없음(무제한), 강한 학습기를 위해

② 3 정도로 얕게, 약한 학습기를 여러 개 합치는 부스팅 철학 때문

③ 100, 깊을수록 좋아서

④ 1, 가장 단순해서

⑤ 10, 중간이 좋아서

> **정답** ②
>
> **해설** 그레이디언트 부스팅은 max_depth=3 정도로 얕은 트리를 사용합니다. 각 트리가 약한 학습기 역할을 하고 여러 약한 학습기를 합쳐 강한 모델을 만드는 것이 부스팅의 철학입니다.

122. 그레이디언트 부스팅의 learning_rate 매개변수가 조절하는 것은?

① 트리의 깊이

② 각 트리의 영향력(낮을수록 천천히, 높을수록 공격적으로 학습)

③ 폴드의 개수

④ 특성의 개수

⑤ 부트스트랩 비율

> **정답** ②
>
> **해설** learning_rate는 각 트리의 영향력을 조절합니다. 낮으면 천천히 학습해 더 많은 트리가 필요하고, 높으면 빠르지만 불안정합니다. 기본값은 0.1이며 n_estimators와 함께 튜닝합니다.

123. 그레이디언트 부스팅이 '과대적합에 강하다'고 평가되는 이유와 가장 관련 깊은 특징은?

① 부트스트랩 샘플 사용

② 깊이가 얕은 트리를 사용해 트리 개수를 늘려도 훈련 데이터를 쉽게 외우지 못함

③ 특성 무작위 선택

④ 표준화 자동 적용

⑤ 교차 검증 내장

> **정답** ②
>
> **해설** 그레이디언트 부스팅은 얕은 트리를 사용해 훈련 데이터를 외울 능력 자체가 적습니다. 그래서 n_estimators를 500으로 늘려도 훈련 점수가 0.946으로 1보다 한참 낮아, 트리를 늘려도 과대적합이 심하지 않습니다.

124. 히스토그램 기반 그레이디언트 부스팅의 '히스토그램 기반'이 의미하는 핵심 기법은?

① 데이터를 그래프로 그린다

② 입력 특성을 256개 구간(bin)으로 나누어 분할 후보를 단순화해 매우 빠르게 탐색한다

③ 트리를 256그루 만든다

④ 256번 반복 학습한다

⑤ 256개 특성을 사용한다

> **정답** ②
>
> **해설** 히스토그램 기반은 각 특성의 값을 256개 구간으로 미리 나눠 분할 후보를 256개로 줄여 탐색을 매우 빠르게 합니다. 256은 8비트 정수로 표현 가능한 최대 값이라 메모리·계산에 효율적입니다.

125. 히스토그램 기반 그레이디언트 부스팅이 누락값(결측값)을 자동 처리하는 방식은?

① 누락값을 모두 0으로 채운다

② 256개 구간 중 하나를 누락값 전용으로 예약해 전처리 없이 처리한다

③ 누락값이 있는 행을 삭제한다

④ 평균으로 채운다

⑤ 누락값을 처리하지 못한다

> **정답** ②
>
> **해설** 256개 구간 중 하나를 누락값 전용으로 예약해 두므로, 누락값이 있는 데이터를 그대로 넣어도 알고리즘이 알아서 처리합니다. 결측값 imputation 단계가 생략되어 실무에서 매우 편리합니다.

126. 사이킷런에서 히스토그램 기반 그레이디언트 부스팅 분류 클래스는?

① GradientBoostingClassifier

② HistGradientBoostingClassifier

③ RandomForestClassifier

④ ExtraTreesClassifier

⑤ XGBClassifier

> **정답** ②
>
> **해설** HistGradientBoostingClassifier가 사이킷런의 히스토그램 기반 그레이디언트 부스팅 분류 클래스입니다. 큰 데이터에서 압도적으로 빠르고 결측값을 자동 처리해 정형 데이터의 사실상 표준입니다.

127. 순열 중요도(permutation importance)의 작동 원리는?

① 트리의 분할 빈도를 센다

② 특성 값을 무작위로 섞어 성능이 얼마나 떨어지는지로 중요도를 측정한다

③ 분산을 계산한다

④ 거리를 측정한다

⑤ 계수의 절댓값을 본다

> **정답** ②
>
> **해설** 순열 중요도는 어떤 특성 값을 무작위로 섞어, 섞은 뒤 성능이 크게 떨어지면 중요한 특성, 거의 안 변하면 안 중요한 특성으로 판단합니다. 어떤 모델에든, 어떤 데이터에든 적용 가능합니다.

128. permutation_importance를 가져오는 사이킷런 모듈은?

① sklearn.ensemble

② sklearn.inspection

③ sklearn.tree

④ sklearn.cluster

⑤ sklearn.decomposition

> **정답** ②
>
> **해설** from sklearn.inspection import permutation_importance로 가져옵니다. inspection은 모델을 '들여다본다'는 의미입니다. n_repeats 매개변수로 섞는 횟수를 지정합니다.

129. 기존 feature_importances_ 대비 순열 중요도의 장점은?

① 트리 모델에서만 작동한다

② 트리뿐 아니라 어떤 모델에든, 그리고 훈련/검증/테스트 어떤 데이터에든 적용할 수 있다

③ 계산이 더 빠르다

④ 음수 값을 준다

⑤ 표준화가 필요 없다

> **정답** ②
>
> **해설** feature_importances_는 트리 모델에서만, 훈련 데이터에 대해서만 작동합니다. 순열 중요도는 신경망·SVM 등 어떤 모델에든, 훈련/검증/테스트 어떤 데이터에든 적용 가능합니다.

130. XGBoost에 대한 설명으로 옳은 것은?

① 사이킷런 내장 알고리즘이다

② Extreme Gradient Boosting의 약자로 캐글을 평정한 외부 라이브러리이며 히스토그램 기반 그레이디언트 부스팅의 변종이다

③ 차원 축소 알고리즘이다

④ 신경망 라이브러리다

⑤ 군집 알고리즘이다

> **정답** ②
>
> **해설** XGBoost(Extreme Gradient Boosting)는 2014년경 캐글을 평정한 외부 라이브러리로, 히스토그램 기반 그레이디언트 부스팅의 변종입니다. 사이킷런 호환 인터페이스를 제공해 거의 같은 방식으로 사용합니다.

131. 그레이디언트 부스팅 계열인 XGBoost·LightGBM이 캐글 등 정형 데이터 대회에서 널리 쓰이는 핵심 이유는?

① 이미지·음성에 특화돼서

② 정형 데이터에서 매우 높은 예측 성능과 빠른 학습 속도를 제공하기 때문

③ 딥러닝이라서

④ 표준화가 필요 없어서만

⑤ 무료라서

> **정답** ②
>
> **해설** XGBoost·LightGBM은 그레이디언트 부스팅을 고도로 최적화한 라이브러리로, 표 형태(정형) 데이터에서 정확도와 학습 속도 모두 뛰어나 실무·대회에서 표준처럼 쓰입니다.

132. 그레이디언트 부스팅에서 learning_rate(학습률)를 지나치게 크게 설정하면 나타날 수 있는 문제는?

① 학습이 전혀 진행되지 않는다

② 각 트리가 오차를 과하게 보정해 학습이 불안정해지거나 과대적합되기 쉽다

③ 트리가 만들어지지 않는다

④ 항상 성능이 좋아진다

⑤ 표준화가 필요해진다

> **정답** ②
>
> **해설** 그레이디언트 부스팅은 이전 트리의 오차를 조금씩 줄여 가는데, learning_rate가 너무 크면 한 번에 과하게 보정해 진동·과대적합이 생길 수 있습니다. 보통 작은 학습률(예: 0.1)에 트리 수를 늘려 안정적으로 학습합니다.

133. 매개변수 튜닝 도구(cross_validate, GridSearchCV 등)와 앙상블 모델의 관계로 옳은 것은?

① 앙상블에는 사용할 수 없다

② 어떤 사이킷런 추정기에든 적용 가능하며, XGBoost·LightGBM도 사이킷런 호환이라 그대로 쓸 수 있다

③ 앙상블 전용 도구가 따로 있다

④ 교차 검증은 단일 모델에만 쓴다

⑤ 그리드 서치는 신경망에만 쓴다

> **정답** ②
>
> **해설** 5-2의 cross_validate, GridSearchCV, RandomizedSearchCV는 어떤 사이킷런 추정기에든 사용할 수 있습니다. XGBoost와 LightGBM도 사이킷런 호환 인터페이스라 그대로 적용 가능합니다.

134. 다음 중 부트스트랩 샘플을 기본적으로 사용하는 앙상블 알고리즘은?

① 엑스트라 트리

② 랜덤 포레스트

③ 그레이디언트 부스팅

④ 히스토그램 기반 그레이디언트 부스팅

⑤ XGBoost

> **정답** ②
>
> **해설** 랜덤 포레스트만 기본적으로 부트스트랩 샘플을 사용합니다. 엑스트라 트리는 분할 무작위로, 부스팅 계열은 순차적 학습으로 다양성을 만듭니다.

6 장 비지도 학습

6-1 군집 알고리즘

135. 1~5장에서 다룬 학습 방식들의 공통점은?

① 모두 비지도 학습이었다
② 모두 정답(타깃)이 있는 지도 학습이었다
③ 모두 군집이었다
④ 모두 차원 축소였다
⑤ 모두 신경망이었다

> **정답** ②
>
> **해설** 1~5장은 도미/빙어, 무게 예측, 와인 분류 등 모두 정답(타깃)을 알려주고 학습하는 지도 학습(supervised learning)이었습니다. 6장부터는 정답이 없는 비지도 학습을 배웁니다.

136. 비지도 학습(unsupervised learning)의 정의로 옳은 것은?

① 정답을 맞히는 학습
② 타깃(정답)이 없는 데이터에서 스스로 구조나 패턴을 발견하는 학습
③ 교차 검증으로 평가하는 학습
④ 여러 모델을 결합하는 학습
⑤ 깊은 트리를 만드는 학습

> **정답** ②
>
> **해설** 비지도 학습은 타깃이 없을 때 사용하는 머신러닝으로, 정답 없이 데이터 자체의 구조나 패턴을 발견합니다. supervised의 super-는 '위에서 감독한다'는 뜻이고 un-이 붙어 '감독 없는 학습'입니다.

137. 비지도 학습이 실무에서 매우 중요한 이유로 옳은 것은?

① 계산이 빨라서
② 현실 데이터의 대부분은 정답(레이블)이 없기 때문
③ 정확도가 항상 100%여서
④ 표준화가 필요 없어서
⑤ 메모리를 적게 써서

> **정답** ②
>
> **해설** 유튜브 영상, 인스타그램 사진, 쇼핑몰 고객 기록 등 세상 데이터의 대부분은 레이블이 없

습니다. 정답 없이 데이터 구조를 파악하는 비지도 학습이 점점 중요해지는 이유입니다.

138. 비슷한 데이터끼리 그룹으로 묶는 비지도 학습 작업을 무엇이라 하는가?

① 분류(classification)
② 회귀(regression)
③ 군집(clustering)
④ 차원 축소
⑤ 정규화

정답 ③

해설 군집(clustering)은 비슷한 데이터끼리 그룹으로 묶는 작업입니다. 넷플릭스의 취향별 사용자 묶기, 쇼핑몰의 고객 그룹화 등이 군집의 예입니다.

139. 100×100 흑백 이미지 한 장을 머신러닝 모델의 입력 특성 벡터로 쓰려면 보통 어떻게 변환하는가?

① 100개 값으로 줄인다
② 2차원 (100, 100)을 1차원 10,000개 값으로 펼친다(reshape)
③ 1개 평균값으로 만든다
④ 3차원으로 늘린다
⑤ 변환 없이 그대로 쓸 수 없다

정답 ②

해설 대부분의 모델은 샘플당 1차원 특성 벡터를 입력으로 받습니다. 그래서 (100,100) 이미지를 100×100=10,000개 픽셀값의 1차원 벡터로 펼쳐(reshape), 한 이미지를 10,000차원 샘플 하나로 다룹니다.

140. 과일 데이터가 저장된 npy 파일 포맷에 대한 설명으로 옳은 것은?

① 텍스트 기반 CSV 포맷
② 넘파이 배열을 그대로 저장하는 전용 바이너리 포맷
③ 이미지 JPEG 포맷
④ 엑셀 포맷
⑤ 압축 ZIP 포맷

정답 ②

해설 npy는 넘파이가 자기 배열을 파일로 저장할 때 쓰는 전용 바이너리 포맷입니다. CSV처럼 텍스트가 아니라 배열 모양과 숫자를 그대로 담아, 불러오면 reshape 정보까지 살아납니다.

np.load()로 읽습니다.

141. print(fruits.shape)의 출력 (300, 100, 100)에서 세 숫자의 의미는?

① (높이, 너비, 샘플 수)

② (샘플 개수, 이미지 높이, 이미지 너비)

③ (채널, 높이, 너비)

④ (클래스, 샘플, 픽셀)

⑤ (행, 열, 깊이)

정답 ②

해설 (300, 100, 100)은 (샘플 개수, 이미지 높이, 이미지 너비)입니다. 샘플 300장, 각 이미지는 100×100 픽셀(한 장당 1만 픽셀)입니다.

142. 맷플롯립에서 넘파이 배열을 이미지로 그리는 함수는?

① plt.plot()

② plt.imshow()

③ plt.bar()

④ plt.hist()

⑤ plt.scatter()

정답 ②

해설 plt.imshow()(image-show)는 넘파이 배열을 이미지로 그려 줍니다. 흑백 이미지는 cmap='gray'로 컬러맵을 지정합니다.

143. 과일 데이터의 흑백 이미지가 반전되어 있는(배경이 검고 사과가 흰) 이유는?

① 촬영 오류 때문

② 컴퓨터가 255에 가까운 높은 값에 집중하므로, 정보가 있는 사과를 높은 값(밝게), 의미 없는 배경을 낮은 값(0)으로 만들기 위해

③ 흑백 사진은 원래 반전되어 있어서

④ 압축 때문에

⑤ 사람 눈에 좋게 하려고

정답 ②

해설 알고리즘이 곱셈·덧셈을 할 때 픽셀값이 0이면 출력이 0이 되어 정보가 없고, 값이 높으면 출력에 크게 기여합니다. 그래서 정보가 있는 사과를 높은 값(밝게), 의미 없는 배경을 0(검게)으로 만들기 위해 반전시켰습니다.

144. 픽셀값이 0인 경우 알고리즘 출력에서 발생하는 문제는?

① 출력이 무한대가 된다
② 어떤 가중치를 곱해도 0이 되어 사실상 정보를 주지 못한다
③ 오버플로가 발생한다
④ 음수가 된다
⑤ 출력이 1이 된다

> **정답** ②
>
> **해설** 픽셀값이 0이면 어떤 가중치를 곱해도 0이 되어 그 픽셀은 출력에 아무 정보도 주지 못하는 '죽은 픽셀'이 됩니다. 반면 값이 높으면 곱한 결과가 커져 출력에 확실히 기여합니다.

145. cmap='gray_r'에서 '_r'이 의미하는 것은?

① red(빨강)
② reversed(반전)
③ raw(원본)
④ ratio(비율)
⑤ random(무작위)

> **정답** ②
>
> **해설** _r은 reversed(반전)입니다. gray는 값이 작으면 검게/크면 희게 칠하는데, gray_r은 반대로 칠합니다. 이미 반전된 데이터(사과=높은 값)에 gray_r을 적용하면 사람 눈에 익숙한 흰 배경+짙은 사과로 보입니다.

146. cmap='gray_r'로 표시할 때 실제 데이터 값에는 어떤 변화가 있는가?

① 데이터가 반전되어 저장된다
② 데이터는 전혀 바뀌지 않고 화면 표시 색만 뒤집힌다
③ 데이터가 0~1로 정규화된다
④ 데이터가 삭제된다
⑤ 데이터가 두 배가 된다

> **정답** ②
>
> **해설** cmap은 화면에 표시하는 색만 바꿀 뿐 fruits 배열 안의 숫자(데이터)는 전혀 바뀌지 않습니다. 실제 계산은 여전히 원래 값(사과=높은 값)으로 합니다. cmap은 화장과 같습니다.

147. 맷플롯립에서 여러 그래프를 격자처럼 배열하는 함수는?

① plt.imshow()

② plt.subplots()

③ plt.legend()

④ plt.show()

⑤ plt.bar()

> **정답** ②
>
> **해설** plt.subplots()는 여러 그래프를 격자처럼 쌓을 수 있게 합니다. subplots(1, 2)는 1행 2열(가로 2개), subplots(2, 1)은 2행 1열(세로 2개)을 만듭니다.

148. 넘파이 배열 arr의 형태가 (300, 100, 100)일 때 arr.mean(axis=(1,2))의 결과 형태는?

① (100, 100)

② (300,) — 각 이미지(샘플)마다 픽셀 전체의 평균 한 값

③ (300, 100)

④ (1,)

⑤ (100,)

> **정답** ②
>
> **해설** axis=(1,2)로 평균을 내면 1·2번 축(100×100 픽셀)이 사라지고 0번 축(샘플 300)만 남아 (300,)이 됩니다. 즉 이미지마다 '전체 픽셀 평균 밝기' 한 값을 얻습니다. 평균을 낸 축은 사라진다는 규칙이 핵심입니다.

149. fruits[0:100].reshape(-1, 100*100)에서 reshape 후 apple 배열의 크기는?

① (100, 100)

② (100, 10000)

③ (10000, 100)

④ (300, 10000)

⑤ (100, 100, 100)

> **정답** ②
>
> **해설** 사과 100장(0~99)을 잘라 100×100 이미지를 1만으로 펼치므로 (100, 10000)이 됩니다. 행 하나가 사진 한 장, 열 1만 개가 픽셀입니다.

150. reshape(-1, 100*100)에서 -1의 의미는?

① 배열을 뒤집는다

② 그 자리 숫자를 전체 원소 개수에 맞춰 자동으로 계산한다

③ 1을 빼라는 뜻

④ 음수 인덱스

⑤ 차원을 1 줄인다

> **정답** ②
>
> **해설** -1은 'reshape가 그 자리 숫자를 알아서 계산하라'는 뜻입니다. 전체 원소가 100만(100장 ×1만)이고 두 번째를 1만으로 정하면 첫 번째는 자동으로 100이 됩니다.

151. apple.mean(axis=1)을 호출하면 결과 배열의 길이는?

① 10000 (픽셀별 평균)

② 100 (사진 한 장마다 평균)

③ 300

④ 1

⑤ 200

> **정답** ②
>
> **해설** (100, 10000)에서 axis=1(가로/열 방향)로 평균내면 1만 픽셀이 뭉개지고 (100,)이 남습니다. 즉 사진 100장 각각의 평균 밝기로, 길이 100짜리 배열입니다.

152. 형태가 (100, 10000)인 배열에서 axis=0으로 평균(mean(axis=0))을 내면 결과의 길이는?

① 100

② 10000 — 같은 위치 픽셀을 100개 샘플에 대해 평균낸 '픽셀별 평균'

③ 1

④ 10000×100

⑤ 0

> **정답** ②
>
> **해설** axis=0은 행(샘플) 방향을 따라 합치므로 0번 축(100)이 사라지고 10000(픽셀)만 남습니다. 결과는 각 픽셀 위치의 샘플 평균, 즉 '픽셀별 평균'입니다. 반대로 axis=1이면 샘플마다 한 값이 되어 길이 100이 됩니다.

153. 히스토그램에서 바나나가 사과·파인애플과 픽셀 평균값만으로 잘 구분되는 이유는?

① 바나나가 가장 밝아서

② 바나나는 길쭉해서 사진에서 차지하는 영역이 작아 0(배경)이 많이 섞여 평균값이 작기 때문

③ 바나나가 노란색이어서

④ 바나나 사진이 더 많아서

⑤ 바나나가 동그래서

> **정답** ②
>
> **해설** 바나나는 길쭉해서 사진에서 차지하는 영역이 작고, 화면 대부분이 검은 배경(값 0)이라 전체 평균이 작게 나옵니다. 그래서 픽셀 평균값만으로도 사과·파인애플과 확실히 구분됩니다.

154. 사과와 파인애플이 픽셀 평균값(샘플별 평균)만으로 구분이 잘 안 되는 이유는?

① 둘 다 노란색이어서

② 둘 다 대체로 동그랗고 사진에서 차지하는 크기가 비슷해 전체 밝기 평균이 엇비슷하기 때문

③ 데이터가 부족해서

④ 표준화를 안 해서

⑤ 둘이 같은 과일이어서

> **정답** ②
>
> **해설** 사과와 파인애플은 둘 다 대체로 동그랗고 사진에서 차지하는 크기가 비슷해, 사진 한 장의 전체 평균값이 엇비슷합니다. 평균은 디테일을 뭉개므로 더 세밀하게 봐야 합니다.

155. 이미지 그룹에서 '샘플별 평균(한 이미지의 전체 픽셀 평균)' 대신 '픽셀별 평균(같은 위치 픽셀을 모든 이미지에 대해 평균)'을 보면 알 수 있는 것은?

① 샘플 개수

② 각 픽셀 위치에서 그 그룹이 평균적으로 밝은지/어두운지 → 그룹의 '평균적인 모양'을 이미지로 복원할 수 있다

③ 정답 레이블

④ 표준편차만

⑤ 아무 정보도 없다

> **정답** ②
>
> **해설** 픽셀별 평균은 같은 좌표 픽셀을 모든 샘플에 대해 평균낸 값이라, 다시 100×100으로 되돌려 그리면 그 그룹의 '평균 형상(평균 얼굴)'이 됩니다. 군집 중심 이미지가 이 평균 형상과 닮는 이유이기도 합니다.

156. np.mean(apple, axis=0)의 결과 길이와 의미는?

① 100, 사진별 평균

② 10000, 픽셀 위치별 평균(평균적인 사과 모습)

③ 300, 전체 평균

④ 1, 단일 값

⑤ 200, 두 과일 평균

> **정답** ②
>
> **해설** axis=0(세로/샘플 방향)으로 평균내면 100장이 뭉개지고 (10000,)이 남습니다. 즉 픽셀 1만 개 각각의 위치별 평균으로, 100장 사과가 공유하는 '평균적인 사과의 모습'입니다.

157. 픽셀별 평균값을 다시 100×100으로 reshape해 imshow로 그리면 무엇을 볼 수 있는가?

① 무작위 노이즈

② 그 과일들의 평균적인 형태(대표 얼굴) 이미지

③ 히스토그램

④ 산점도

⑤ 트리 구조

> **정답** ②
>
> **해설** 픽셀별 평균을 100×100 이미지로 되돌리면 그 과일의 '대표 얼굴'(평균적인 형태)이 보입니다. 이 apple_mean 같은 평균 이미지가 6-2의 클러스터 중심과 똑같이 생기게 됩니다.

158. 6-1처럼 '정답 레이블을 미리 알고' 그룹별 평균을 구하는 방식이 실제 비지도(군집) 상황에서는 통하지 않는 근본 이유는?

① 계산이 느려서

② 실제 비지도 문제에는 어떤 샘플이 어느 그룹인지 알려 주는 정답이 아예 없기 때문

③ 평균을 못 구해서

④ 이미지가 흑백이라서

⑤ 표준화가 안 돼서

> **정답** ②
>
> **해설** 비지도 학습에는 정답 레이블이 없으므로 '사과만 모아 평균'을 낼 수 없습니다. 그래서 정답 없이도 비슷한 샘플을 스스로 묶고 중심을 찾는 k-평균 같은 군집 알고리즘이 필요합니다.

159. 다음 중 비지도 학습 과제가 아닌 것은?

① 비슷한 데이터를 묶는 군집

② 특성 수를 줄이는 차원 축소

③ 정답 레이블로 메일을 정상/스팸으로 분류하는 일

④ 이상치(특이 데이터) 탐지

⑤ 연관 규칙 찾기

> **정답** ③
>
> **해설** 스팸 분류는 정답(정상/스팸) 레이블을 보고 학습하는 지도 학습입니다. 군집·차원 축소·이상치 탐지·연관 규칙은 정답 없이 데이터 구조만 활용하는 비지도 학습입니다.

160. 군집(clustering)과 분류(classification)의 가장 큰 차이는?

① 군집이 항상 더 정확하다

② 분류는 미리 정해진 정답 레이블로 학습·예측하지만, 군집은 정답 없이 비슷한 것끼리 묶는다

③ 분류는 비지도, 군집은 지도 학습이다

④ 둘은 완전히 같은 작업이다

⑤ 군집은 정답이 반드시 필요하다

> **정답** ②
>
> **해설** 분류는 각 샘플의 정답 클래스를 알고 그 경계를 학습하는 지도 학습이고, 군집은 정답이 없는 상태에서 데이터의 유사성만으로 그룹을 형성하는 비지도 학습입니다.

161. 군집 알고리즘이 만든 '클러스터'에 처음부터 의미 있는 이름(예: 사과·바나나)이 자동으로 붙는가?

① 그렇다 — 알고리즘이 이름을 지어 준다

② 아니다 — 알고리즘은 그룹 번호(0,1,2…)만 부여하고, 그 의미는 사람이 사후에 해석한다

③ 정답 데이터에서 가져온다

④ 항상 알파벳으로 붙는다

⑤ 이름이 없으면 군집이 안 된다

> **정답** ②
>
> **해설** 군집 결과는 단지 '몇 번 그룹'이라는 번호일 뿐, 그것이 사과인지 바나나인지는 알고리즘이 모릅니다. 각 클러스터의 의미는 사람이 결과를 보고 해석해 붙입니다.

162. 흑백 이미지에서 픽셀값 0과 255는 각각 무엇을 의미하는가?(일반적 관례)

① 0=흰색, 255=검정

② 0=검정(어두움), 255=흰색(가장 밝음)

③ 둘 다 회색

④ 0=투명, 255=불투명

⑤ 0=빨강, 255=파랑

> **정답** ②
>
> **해설** 8비트 흑백 이미지에서 픽셀값은 0~255이며 0이 가장 어두운 검정, 255가 가장 밝은 흰색입니다. 이 값의 크기가 곧 밝기이고, 머신러닝에서는 그대로 특성값으로 쓰입니다.

163. (100, 100) 흑백 이미지 한 장을 1차원으로 펼치면(reshape) 그 길이는?

① 100

② 200

③ 10000 — 100×100, 원소(픽셀) 총 개수는 그대로 보존된다

④ 1000

⑤ 256

> **정답** ③
>
> **해설** reshape는 모양만 바꿀 뿐 원소 개수를 보존합니다. (100,100) 이미지는 픽셀이 100×100=10,000개이므로 1차원으로 펼치면 길이 10,000인 벡터가 됩니다.

164. 군집을 수행하기 전에 이미지 묶음을 (샘플 수, 픽셀 수) 형태의 2차원 배열로 펼치는 이유는?

① 보기 좋아서

② KMeans 등 사이킷런 모델이 (샘플 수, 특성 수) 형태의 2차원 입력을 받기 때문

③ 이미지를 압축하려고

④ 색을 바꾸려고

⑤ 정답을 만들려고

> **정답** ②
>
> **해설** 사이킷런 모델은 한 샘플을 1차원 특성 벡터로 받아 전체를 (샘플 수, 특성 수) 2차원 배열로 다룹니다. 그래서 (300,100,100) 이미지를 (300, 10000)으로 펼쳐 입력합니다.

165. 군집 분석의 실제 활용 예로 가장 적절한 것은?

① 주가를 정확히 예측하기

② 구매 패턴이 비슷한 고객을 그룹으로 나눠 마케팅하기(고객 세분화)

③ 이미지에 정답 라벨 붙이기

④ 회귀선 그리기

⑤ 오차를 역전파하기

> **정답** ②
>
> **해설** 고객 세분화, 문서 주제별 묶음, 유사 이미지 그룹화처럼 '정답은 없지만 비슷한 것끼리 묶고 싶은' 문제에 군집이 널리 쓰입니다.

166. 한 그룹의 이미지들을 픽셀 단위로 평균낸 '평균 이미지'가 개별 이미지보다 흐릿하게(부드럽게) 보이는 이유는?

① 해상도가 낮아서
② 이미지마다 위치·모양이 조금씩 달라 평균을 내면 그 차이가 상쇄·뭉개지기 때문
③ 흑백이라서
④ 픽셀이 손상돼서
⑤ 평균은 항상 0이라서

> **정답** ②
>
> **해설** 샘플마다 대상의 위치·크기·형태가 미세하게 달라, 같은 좌표 픽셀을 평균내면 일치하지 않는 부분이 흐려집니다. 그 결과 그룹의 '전형적인 형태'가 부드럽게 드러납니다.

167. 히스토그램으로 두 그룹의 '샘플별 평균값' 분포가 거의 겹쳐 보인다면 알 수 있는 사실은?

① 두 그룹이 완전히 같다
② 그 단일 통계값(평균)만으로는 두 그룹을 잘 구분하기 어렵다
③ 평균이 잘못 계산됐다
④ 그룹이 하나뿐이다
⑤ 정답이 필요 없다

> **정답** ②
>
> **해설** 두 분포가 겹친다는 것은 평균값으로는 두 그룹이 잘 분리되지 않는다는 뜻입니다. 이럴 때는 다른 특성이나 더 정교한 표현(예: 픽셀별 패턴)이 필요합니다.

168. 비지도 학습에서 모델 성능을 '정확도(accuracy)'로 평가하기 어려운 근본적인 이유는?

① 계산이 느려서
② 비교할 정답 레이블이 없기 때문
③ 데이터가 너무 많아서
④ 정확도 공식이 없어서
⑤ 표준화를 안 해서

> **정답** ②
>
> **해설** 정확도는 예측을 정답과 비교해야 계산됩니다. 비지도 학습에는 정답이 없으므로 정확도로 평가할 수 없고, 이너셔·실루엣 계수 같은 다른 방식으로 품질을 가늠합니다.

169. 픽셀 10,000개처럼 매우 고차원인 데이터를 그대로 다룰 때 생길 수 있는 문제는?

① 문제가 전혀 없다

② 계산량·메모리 부담이 크고 시각화·분석이 어려워, 차원 축소의 필요성이 생긴다

③ 정확도가 항상 100%가 된다

④ 샘플이 자동으로 줄어든다

⑤ 정답이 생긴다

> **정답** ②
>
> **해설** 차원이 커지면 계산 비용과 메모리 요구가 늘고 2~3차원으로 보기도 어렵습니다. 그래서 중요한 정보를 유지하며 차원을 줄이는 PCA 같은 기법을 함께 씁니다.

170. 맷플롯립 imshow로 흑백 이미지를 그릴 때 cmap을 지정하지 않으면 색이 이상하게 보일 수 있는 이유는?

① 이미지가 손상돼서

② 기본 컬러맵이 흑백이 아니라 색상 맵이어서 — cmap='gray'를 지정해야 흑백으로 표시된다

③ 해상도가 낮아서

④ 픽셀값이 음수라서

⑤ imshow는 흑백을 지원하지 않아서

> **정답** ②
>
> **해설** imshow의 기본 컬러맵은 흑백이 아니어서 픽셀값이 알록달록하게 매핑됩니다. 흑백으로 보려면 cmap='gray'(또는 반전인 'gray_r')를 지정해야 합니다.

171. 형태가 (300, 100, 100)인 과일 배열에서 fruits[0]의 형태(shape)는?

① (300, 100, 100)

② (100, 100) — 첫 번째 이미지 한 장

③ (300, 100)

④ (1, 300)

⑤ (10000,)

> **정답** ②

> **해설** 맨 앞 축(샘플)에서 인덱스 0을 고르면 그 축이 사라지고 한 장의 이미지 (100, 100)이 남습니다. 인덱싱이 어떻게 차원을 줄이는지 이해하는 것이 중요합니다.

172. 비지도 학습에 '정답이 없다'는 점이 단점이 아니라 장점이 되는 경우는?

① 언제나 단점이다

② 사람이 일일이 정답(레이블)을 달지 않아도 되므로, 레이블링 비용 없이 대량의 데이터를 활용할 수 있다

③ 정확도가 높아져서

④ 표준화가 필요 없어서

⑤ 계산이 빨라져서

> **정답** ②
>
> **해설** 현실의 데이터 대부분은 정답이 없고, 사람이 일일이 라벨을 다는 데는 큰 비용이 듭니다. 비지도 학습은 라벨 없이도 데이터의 구조를 찾아내므로 이런 대량 데이터를 활용할 수 있습니다.

6-2 k-평균

173. 정답(타깃)을 몰라도 비슷한 샘플들이 모이는 평균값(중심)을 자동으로 찾아주는 군집 알고리즘은?

① 결정 트리

② k-평균(k-means)

③ 로지스틱 회귀

④ PCA

⑤ 랜덤 포레스트

> **정답** ②
>
> **해설** k-평균(k-means)은 정답을 몰라도 비슷한 샘플들이 모이는 중심(평균값)을 알고리즘이 스스로 찾아줍니다. '클러스터 k개로 나눠줘'라고만 하면 알아서 중심을 찾아냅니다.

174. k-평균이 찾아낸, 각 클러스터의 한가운데에 위치하는 평균값을 무엇이라 하는가?

① 루트 노드

② 클러스터 중심(센트로이드)

③ 주성분

④ 이너셔

⑤ 폴드

> **정답** ②
>
> **해설** 클러스터 중심(cluster center) 또는 센트로이드(centroid)는 각 클러스터에 속한 샘플들의 평균 위치로, 클러스터의 한가운데에 있습니다. 두 단어는 같은 뜻입니다.

175. k-평균 알고리즘의 1단계 작업은?

① 가장 가까운 중심에 샘플을 배정한다
② 무작위로 k개의 클러스터 중심을 정한다
③ 샘플들의 평균으로 중심을 옮긴다
④ 이너셔를 계산한다
⑤ 데이터를 표준화한다

> **정답** ②
>
> **해설** 1단계는 무작위로 k개의 클러스터 중심을 정하는 것입니다. 처음엔 진짜 아무 데나 찍어도 되며, 엉터리여도 괜찮습니다.

176. k-평균 알고리즘의 작동 순서로 옳은 것은?

① 중심 이동 → 무작위 초기화 → 샘플 배정
② 무작위 초기화 → 가장 가까운 중심에 샘플 배정 → 샘플 평균으로 중심 이동 → 변화 없을 때까지 반복
③ 샘플 배정 → 중심 삭제 → 종료
④ 표준화 → 분할 → 평가
⑤ 투영 → 복원 → 시각화

> **정답** ②
>
> **해설** k-평균은 ① 무작위로 k개 중심 초기화 ② 각 샘플을 가장 가까운 중심에 배정 ③ 클러스터에 속한 샘플 평균으로 중심 이동 ④ 중심에 변화가 없을 때까지 2~3을 반복하는 순서로 작동합니다.

177. k-평균 알고리즘에서 'k'가 의미하는 것은?

① 반복 횟수
② 사용자가 정해 주는 클러스터 개수
③ 특성 개수
④ 샘플 개수
⑤ 이너셔 값

> **정답** ②

해설 k는 우리가 정해 주는 클러스터(무리) 개수입니다. 과일이 세 종류면 k=3으로 지정해 중심 세 개를 찾게 합니다.

178. k-평균 학습 과정에서 정답(타깃)이 사용되는가?

① 예, 매 단계 정답을 참조한다
② 아니오, 거리 계산과 평균만 반복할 뿐 정답은 전혀 사용하지 않는다
③ 예, 마지막 단계에만 사용한다
④ 예, 중심 초기화에 사용한다
⑤ 예, 1단계에만 사용한다

정답 ②

해설 k-평균은 거리 계산과 평균이라는 두 가지만 반복하며, 어디에도 '이건 사과다'라는 정답이 들어가지 않습니다. 그런데도 비슷한 것끼리 저절로 모이는 것이 비지도 학습의 힘입니다.

179. 사이킷런에서 k-평균 군집을 수행하는 클래스와 모듈은?

① sklearn.tree의 KMeans
② sklearn.cluster의 KMeans
③ sklearn.linear_model의 KMeans
④ sklearn.decomposition의 KMeans
⑤ sklearn.ensemble의 KMeans

정답 ②

해설 from sklearn.cluster import KMeans로 가져옵니다. cluster는 '군집' 모듈입니다. KMeans(n_clusters=3, random_state=42)로 객체를 만듭니다.

180. KMeans(n_clusters=3)에서 n_clusters=3의 의미는?

① 3번 반복한다
② 데이터를 3개의 클러스터로 나눈다
③ 3개 특성을 쓴다
④ 3개 샘플만 쓴다
⑤ 3-폴드 교차 검증

정답 ②

해설 n_clusters=3은 알고리즘의 k 값으로, 데이터를 3개의 클러스터로 나누라는 의미입니다. 과일이 세 종류라 3으로 지정했습니다.

181. **km.fit(fruits_2d)가 지도 학습의 fit과 다른 결정적 차이는?**

① fit이 두 개 필요하다

② 타깃(y)을 주지 않고 데이터만 준다

③ 교차 검증을 내장한다

④ 표준화를 자동으로 한다

⑤ 차이가 없다

> **정답** ②
>
> **해설** 지도 학습은 fit(X, y)처럼 정답 y를 함께 주지만, 비지도 학습인 k-평균은 fit(fruits_2d)처럼 정답 없이 데이터만 줍니다. '알아서 묶어 봐'가 비지도 학습의 fit입니다.

182. **k-평균 학습 후 각 샘플의 클러스터 배정 결과가 저장되는 속성은?**

① km.cluster_centers_

② km.labels_

③ km.inertia_

④ km.n_iter_

⑤ km.components_

> **정답** ②
>
> **해설** km.labels_에 300개 샘플 각각이 0, 1, 2번 클러스터 중 어디에 배정됐는지가 저장됩니다.

183. **km.labels_의 0, 1, 2라는 숫자에 대한 올바른 이해는?**

① 0번은 항상 사과를 의미한다

② 단지 클러스터에 붙인 이름표 번호일 뿐, 정해진 의미는 없다

③ 정확도 순위다

④ 샘플 개수다

⑤ 거리 값이다

> **정답** ②
>
> **해설** 0, 1, 2는 그냥 클러스터에 붙인 이름표 번호일 뿐, '0번은 사과'처럼 정해진 의미가 없습니다. 어느 번호가 무슨 과일인지는 나중에 그림으로 확인해야 합니다.

184. **np.unique(km.labels_, return_counts=True)가 반환하는 것은?**

① 평균과 표준편차

② 고유한 레이블 값과 각 레이블의 개수

③ 최솟값과 최댓값

④ 중심 좌표

⑤ 이너셔와 반복 횟수

> **정답** ②
>
> **해설** np.unique는 고유한 값을 찾아주고, return_counts=True를 주면 각 값의 개수도 반환합니다. 결과 (array([0,1,2]), array([91, 98, 111]))는 클러스터별 샘플 수가 91, 98, 111(합 300)임을 뜻합니다.

185. k-평균 결과가 100, 100, 100이 아니라 91, 98, 111로 나온 것이 의미하는 바는?

① 코드가 틀렸다

② 정답을 안 보고 모양만으로 묶었으므로 일부 샘플이 다른 무리로 새어 들어가 완벽하진 않다

③ 데이터가 손상됐다

④ 표준화가 필요하다

⑤ 클러스터가 3개가 아니다

> **정답** ②
>
> **해설** 원래 각 100장씩이지만 91, 98, 111로 나온 것은 정답을 보지 않고 모양만으로 묶어 몇 장씩 엉뚱한 무리로 새어 들어갔다는 뜻입니다. 완벽하진 않아도 꽤 비슷하게 나눴습니다.

186. fruits[km.labels_==0]처럼 True/False 배열로 원소를 골라내는 기법은?

① 슬라이싱

② 불리언 인덱싱(boolean indexing)

③ 팬시 인덱싱

④ 브로드캐스팅

⑤ 리셰이핑

> **정답** ②
>
> **해설** km.labels_==0은 0번 클러스터인 자리만 True인 배열을 만들고, 넘파이 배열에 이를 인덱스로 넣으면 True 위치 원소만 골라냅니다. 이를 불리언 인덱싱이라 하며 for문 없이 조건에 맞는 것만 추출합니다.

187. k-평균이 최종적으로 찾은 클러스터 중심이 저장되는 속성은?

① km.labels_

② km.cluster_centers_

③ km.inertia_

④ km.n_iter_

⑤ km.feature_importances_

> **정답** ②
>
> **해설** km.cluster_centers_에 세 클러스터의 중심이 저장됩니다. fruits_2d 기준이라 한 줄이 1만 짜리 1차원이므로, 이미지로 보려면 reshape(-1, 100, 100)으로 펼칩니다.

188. k-평균이 찾은 클러스터 중심 이미지가 6-1의 apple_mean(평균 얼굴)과 똑같이 생긴 이유는?

① 우연의 일치

② 클러스터 중심이 곧 그 클러스터에 모인 샘플들의 평균이기 때문

③ 같은 코드여서

④ 표준화 때문

⑤ 데이터가 같아서

> **정답** ②
>
> **해설** 클러스터 중심이 곧 그 클러스터에 모인 사진들의 평균이므로, 6-1에서 손으로 구한 평균 얼굴과 똑같이 생깁니다. 6-1은 정답을 보고, 6-2는 정답 없이 그 평균을 찾았다는 차이만 있습니다.

189. km.transform(샘플) 메서드가 반환하는 것은?

① 클러스터 레이블

② 샘플과 각 클러스터 중심 사이의 거리

③ 예측 확률

④ 이너셔

⑤ 특성 중요도

> **정답** ②
>
> **해설** transform은 어떤 샘플이 각 클러스터 중심으로부터 얼마나 떨어져 있는지 거리를 계산합니다. 출력 [[5267.7, 8837.4, 3393.8]]에서 가장 작은 세 번째 값(3393.8)이 2번 중심에 가장 가깝다는 뜻입니다.

190. fruits_2d[100:101]처럼 [100]이 아니라 [100:101]로 슬라이싱하는 이유는?

① 100번과 101번 두 개를 쓰려고

② 사이킷런이 2차원 배열을 요구하므로 (1, 10000) 모양을 유지하기 위해

③ 속도를 높이려고

④ 100번이 없어서

⑤ 관습일 뿐 의미 없다

> **정답** ②
>
> **해설** 사이킷런은 입력으로 항상 2차원 배열을 원합니다. [100]이라고 쓰면 1차원이 되어 모양이 안 맞고, [100:101]로 슬라이싱하면 (1, 10000) 2차원이 유지됩니다.

191. km.predict(샘플) 메서드의 역할은?

① 거리를 모두 출력한다

② 샘플을 넣으면 가장 가까운 클러스터 중심을 찾아 그 클러스터 번호를 예측한다

③ 이너셔를 계산한다

④ 중심을 이동시킨다

⑤ 데이터를 복원한다

> **정답** ②
>
> **해설** predict는 샘플을 넣으면 가장 가까운 클러스터 중심을 찾아 그 클러스터 번호를 예측값으로 반환합니다. 거리를 직접 비교할 필요 없이 가장 가까운 것을 골라줍니다.

192. km.n_iter_ 속성이 저장하는 값은?

① 클러스터 개수

② 알고리즘이 중심이 멈출 때까지 반복한 횟수

③ 샘플 개수

④ 특성 개수

⑤ 이너셔 값

> **정답** ②
>
> **해설** n_iter_는 알고리즘이 '중심에 변화가 없을 때까지' 반복한 횟수입니다. 과일 데이터에서는 단 3번 만에 수렴했습니다. 보통 몇 번이면 수렴해 효율적입니다.

193. k-평균 알고리즘의 대표적인 단점은?

① 속도가 너무 느리다

② 클러스터 개수 k를 사전에 지정해야 한다

③ 정답이 필요하다

④ 표준화가 필수다

⑤ 특성이 많아야 한다

> **정답** ②

해설 k-평균의 단점 중 하나는 클러스터 개수 k를 사전에 지정해야 한다는 점입니다. 현실 데이터는 몇 개 그룹으로 나뉘는지 모를 때가 대부분이라 적절한 k를 찾는 방법이 필요합니다.

194. 적절한 클러스터 개수 k를 찾는 대표적인 방법은?

① 교차 검증
② 엘보우(elbow) 방법
③ 그리드 서치
④ 부트스트랩
⑤ 표준화

정답 ②

해설 엘보우(elbow, 팔꿈치) 방법이 적절한 k를 찾는 대표적 방법입니다. k를 늘려가며 이너셔를 그래프로 그려 기울기가 팔꿈치처럼 꺾이는 지점을 찾습니다.

195. 이너셔(inertia)의 정의로 옳은 것은?

① 클러스터 개수
② 클러스터 중심과 그 클러스터에 속한 샘플 사이 거리의 제곱 합
③ 샘플 개수
④ 특성 중요도
⑤ 설명된 분산

정답 ②

해설 이너셔는 클러스터 중심과 그 클러스터에 속한 샘플들 사이 거리의 제곱을 모두 더한 값입니다. 샘플들이 얼마나 옹기종기 가깝게 모여 있는지를 나타냅니다.

196. 이너셔 값과 군집 상태의 관계로 옳은 것은?

① 이너셔가 크면 잘 뭉쳐 있다
② 이너셔가 작을수록 샘플들이 중심 가까이 빽빽하게 모여 잘 뭉쳐 있다
③ 이너셔는 클수록 좋다
④ 이너셔는 군집과 무관하다
⑤ 이너셔는 항상 0이다

정답 ②

해설 이너셔가 작다 = 샘플들이 중심 가까이 빽빽하게 모여 잘 뭉쳐 있다, 크다 = 멀리 흩어져 엉성하게 모여 있다는 뜻입니다.

197. 엘보우 방법에서 '가장 낮은 이너셔'를 단순히 찾으면 안 되는 이유는?

① 계산이 느려서
② 클러스터 개수 k를 늘리면 이너셔는 무조건 줄어들어 무한정 k를 키우게 되기 때문
③ 이너셔가 음수가 되어서
④ 정답이 필요해서
⑤ 이너셔가 항상 일정해서

> **정답** ②
> **해설** k를 늘리면 이너셔는 무조건 줄어들어 극단적으로 샘플마다 클러스터를 주면 0이 됩니다. 그래서 '가장 낮은 값'이 아니라 '감소가 꺾이는 지점(팔꿈치)'을 찾아야 합니다.

198. 교차 검증을 군집(k-평균)에 사용할 수 없는 이유는?

① 계산이 너무 느려서
② 교차 검증은 정답이 있는 지도 학습에서 성능을 평가하는 방법이라 정답이 없는 군집에는 맞지 않으므로
③ 폴드를 만들 수 없어서
④ 이너셔가 없어서
⑤ 사실 사용할 수 있다

> **정답** ②
> **해설** 교차 검증은 정답이 있는 지도 학습에서 모델 성능을 평가할 때 쓰는 방법입니다. 정답이 없는 군집에는 맞지 않으므로, k는 엘보우 방법으로 찾습니다.

199. k-평균 알고리즘이 반복을 멈추는(수렴하는) 조건은?

① 미리 정한 1회만 수행
② 클러스터 중심이 더 이상(거의) 이동하지 않을 때
③ 이너셔가 음수가 될 때
④ 샘플을 다 쓸 때
⑤ 정답을 맞힐 때

> **정답** ②
> **해설** k-평균은 '할당 → 중심 갱신'을 반복하다가 중심의 위치가 더 이상 변하지 않으면(수렴) 멈춥니다. 이때까지 걸린 반복 횟수가 n_iter_에 저장됩니다.

200. k-평균은 초기 중심을 무작위로 잡는다. 이 때문에 생길 수 있는 일은?

① 항상 같은 결과만 나온다

② 초기값에 따라 최종 군집 결과가 조금씩 달라질 수 있고, 최선이 아닌 해(지역 최솟값)에 빠질 수 있다

③ 절대 수렴하지 않는다

④ 정답이 필요해진다

⑤ 클러스터가 1개로 합쳐진다

> **정답** ②
>
> **해설** k-평균은 초기 중심 위치에 영향을 받아, 실행마다 결과가 달라지거나 전역 최적이 아닌 지역 최솟값에 수렴할 수 있습니다. 그래서 여러 번 실행해 가장 좋은 결과를 고릅니다.

201. k-평균을 여러 번 실행해 그중 가장 좋은 군집을 고를 때 사용하는 기준은?

① 가장 큰 이너셔

② 이너셔가 가장 작은(가장 잘 뭉친) 결과

③ 샘플이 가장 많은 결과

④ 반복이 가장 적은 결과

⑤ 무작위로 하나

> **정답** ②
>
> **해설** 이너셔가 작을수록 샘플들이 중심 가까이 잘 모인 좋은 군집입니다. 사이킷런 KMeans도 여러 번 초기화해(n_init) 이너셔가 가장 낮은 결과를 최종으로 채택합니다.

202. 사이킷런 KMeans가 더 나은 초기 중심을 똑똑하게 고르기 위해 기본으로 사용하는 초기화 방법은?

① 완전 무작위

② k-means++ — 서로 멀리 떨어진 지점들을 초기 중심으로 선택

③ 항상 원점

④ 첫 k개 샘플

⑤ 정답 위치

> **정답** ②
>
> **해설** 사이킷런 KMeans의 기본 초기화는 k-means++로, 초기 중심들을 서로 떨어지게 잡아 지역 최솟값에 빠질 위험을 줄이고 수렴을 빠르게 합니다.

203. k-평균은 거리를 기반으로 동작한다. 특성마다 값의 범위(스케일)가 크게 다르면 생기는 문제는?

① 문제가 없다

② 값이 큰 특성이 거리를 지배해 군집이 왜곡될 수 있어, 보통 표준화가 필요하다

③ 항상 정확해진다

④ 샘플이 줄어든다

⑤ k가 자동으로 정해진다

> **정답** ②
>
> **해설** 거리 계산에서는 범위가 큰 특성이 거리값을 좌우해 군집이 그 특성에 치우칩니다. 그래서 k-평균은 표준화에 민감하며, 결정 트리와 달리 보통 표준화가 필요합니다.

204. k-평균이 샘플과 클러스터 중심 사이의 거리를 잴 때 일반적으로 사용하는 거리는?

① 맨해튼 거리

② 유클리드 거리(직선 거리)

③ 코사인 유사도

④ 해밍 거리

⑤ 거리를 쓰지 않는다

> **정답** ②
>
> **해설** 표준적인 k-평균은 유클리드 거리(좌표 차이의 제곱합의 제곱근)를 사용해 가장 가까운 중심을 찾고, 중심을 그 샘플들의 평균으로 갱신합니다.

205. k-평균이 잘 찾아내지 못하는 클러스터 모양은?

① 둥글고 크기가 비슷한 클러스터

② 길쭉하거나 휘어진(비구형) 클러스터 — k-평균은 둥근 모양·비슷한 크기를 가정한다

③ 점이 하나인 클러스터

④ 정사각형 클러스터

⑤ 모든 모양을 똑같이 잘 찾는다

> **정답** ②
>
> **해설** k-평균은 중심까지의 거리로 묶으므로 둥글고 크기가 비슷한 클러스터를 가정합니다. 초승달처럼 길쭉하거나 밀도가 다른 군집은 잘 못 나누며, 이런 경우 DBSCAN 등 다른 기법이 낫습니다.

206. k-평균에서 한 샘플은 몇 개의 클러스터에 속하는가?

① 여러 개에 동시에 속한다

② 정확히 하나 — 가장 가까운 중심 하나에 배정되는 하드(hard) 군집이다

③ 0개

④ k개 모두

⑤ 절반씩 나뉜다

> **정답** ②
>
> **해설** k-평균은 각 샘플을 가장 가까운 중심 하나에만 배정하는 하드 군집입니다. 한 샘플이 여러 그룹에 확률적으로 걸치는 소프트 군집(예: 가우시안 혼합)과 다릅니다.

207. k-평균의 '중심 갱신 단계'에서 새 클러스터 중심은 어떻게 계산되는가?

① 무작위로 다시 뽑는다

② 그 클러스터에 배정된 샘플들의 평균(좌표 평균)으로 옮긴다

③ 가장 먼 샘플로 옮긴다

④ 변하지 않는다

⑤ 정답 위치로 옮긴다

> **정답** ②
>
> **해설** k-평균에서 중심(centroid)은 그 클러스터에 속한 샘플들의 평균입니다. 그래서 'k-평균(means)'이라는 이름이 붙었고, 할당이 바뀌면 평균도 다시 계산해 중심을 옮깁니다.

208. transform()으로 얻은 '각 클러스터 중심까지의 거리' 중 가장 작은 값에 해당하는 클러스터는 그 샘플에 대해 무엇을 뜻하는가?

① 가장 먼 클러스터

② predict가 그 샘플을 배정할 클러스터(가장 가까운 중심)

③ 무시되는 클러스터

④ 이너셔가 0인 클러스터

⑤ 정답 클러스터

> **정답** ②
>
> **해설** transform은 샘플에서 모든 중심까지의 거리를 반환합니다. 그중 거리가 가장 짧은 클러스터가 곧 predict가 그 샘플을 배정하는(가장 가까운) 클러스터입니다.

209. k-평균에서 클러스터 개수 k를 실제 그룹 수보다 너무 작게 잡으면 나타나는 문제는?

① 군집이 더 정확해진다

② 서로 다른 그룹이 한 클러스터에 뭉뚱그려져 구분되지 않는다

③ 샘플이 사라진다

④ 이너셔가 0이 된다

⑤ 수렴하지 않는다

> **정답** ②
>
> **해설** k가 너무 작으면 본래 다른 그룹들이 하나의 클러스터로 합쳐져 구분이 사라집니다. 반대로 너무 크면 한 그룹이 여러 조각으로 쪼개집니다.

210. k-평균에서 k를 데이터의 실제 그룹 수보다 너무 크게 잡으면?

① 모든 그룹이 합쳐진다
② 하나의 자연스러운 그룹이 여러 클러스터로 쪼개진다
③ 이너셔가 커진다
④ 정답이 필요해진다
⑤ 수렴이 불가능하다

> **정답** ②
>
> **해설** k가 과하면 본래 하나인 그룹이 여러 클러스터로 분할되어 해석이 어려워집니다. 그래서 엘보우 방법 등으로 적절한 k의 균형점을 찾습니다.

211. 정답을 모를 때 군집 품질을 평가하는 지표로, 같은 클러스터 내 응집도와 다른 클러스터와의 분리도를 함께 보는 것은?

① 정확도
② 실루엣 계수(silhouette score)
③ 정밀도
④ 재현율
⑤ F1 점수

> **정답** ②
>
> **해설** 실루엣 계수는 한 샘플이 자기 클러스터에는 가깝고 이웃 클러스터와는 먼 정도를 -1~1로 나타냅니다. 정답이 없어도 군집의 품질을 가늠할 수 있어 이너셔와 함께 자주 쓰입니다.

212. k-평균을 같은 데이터에 random_state를 고정해 실행하면?

① 매번 다른 결과가 나온다
② 초기화가 재현되어 매번 같은 군집 결과를 얻는다
③ 수렴하지 않는다
④ 정답이 필요해진다
⑤ 이너셔가 0이 된다

> **정답** ②
>
> **해설** k-평균의 무작위 초기화는 random_state로 고정할 수 있습니다. 같은 값을 주면 초기 중심

이 동일하게 재현되어 결과가 일정해지므로, 실험 재현성을 확보할 수 있습니다.

6-3 주성분 분석

213. 머신러닝에서 '차원(dimension)'이 일반적으로 가리키는 것은?

① 샘플의 개수
② 특성(feature)의 개수
③ 클러스터의 개수
④ 정답 레이블의 개수
⑤ 반복 횟수

정답 ②

해설 머신러닝에서 차원은 보통 특성의 개수를 가리킵니다. 과일 사진을 10,000개 픽셀로 펼치면 10,000개 특성, 즉 10,000차원 데이터가 됩니다.

214. 차원 축소(dimensionality reduction)의 목적으로 가장 적절한 것은?

① 샘플 개수를 줄이는 것
② 데이터를 가장 잘 나타내는 일부 특성을 골라 특성 개수를 줄이면서 중요한 정보는 최대한 보존하는 것
③ 정답을 추가하는 것
④ 클러스터를 만드는 것
⑤ 표준화를 수행하는 것

정답 ②

해설 차원 축소는 데이터를 잘 표현하는 일부 특성을 추려 특성 개수(차원)를 줄이되 중요한 정보는 최대한 보존하는 비지도 학습입니다. 저장 공간과 계산 시간을 줄이고 성능도 개선할 수 있습니다.

215. 주성분 분석(PCA)이 찾는 '주성분(principal component)'의 정의로 옳은 것은?

① 샘플이 가장 적은 방향
② 데이터에 있는 분산이 가장 큰 방향
③ 정답과 가장 가까운 방향
④ 거리가 가장 짧은 방향
⑤ 이너셔가 가장 큰 방향

정답 ②

해설 주성분은 데이터의 분산이 가장 큰 방향을 가리키는 벡터입니다. 분산이 크다 = 데이터가 가장 넓게 퍼져 있다 = 그 방향이 데이터를 가장 잘 설명한다는 의미입니다.

216. 주성분의 개수와 원본 특성의 관계로 옳은 것은?

① 주성분은 항상 1개다
② 원본 특성 개수보다 많을 수 없다
③ 주성분은 샘플 개수와 같다
④ 주성분은 항상 2개다
⑤ 주성분은 특성과 무관하다

정답 ②

해설 주성분은 원본 특성의 개수만큼 찾을 수 있지만 그보다 많을 수는 없습니다. 보통 분산이 큰 순서대로 일부만 골라 차원을 줄입니다.

217. 사이킷런에서 PCA 클래스를 임포트하는 올바른 구문은?

① from sklearn.cluster import PCA
② from sklearn.decomposition import PCA
③ from sklearn.tree import PCA
④ from sklearn.ensemble import PCA
⑤ from sklearn.linear_model import PCA

정답 ②

해설 PCA는 분해(decomposition)를 담당하는 sklearn.decomposition 모듈에 있습니다. KMeans가 sklearn.cluster에 있던 것과는 위치가 다릅니다.

218. PCA의 n_components 매개변수에 정수 50을 지정했다면 그 의미는?

① 분산의 50%를 보존한다
② 주성분을 50개 찾는다
③ 샘플을 50개 사용한다
④ 50번 반복한다
⑤ 클러스터를 50개 만든다

정답 ②

해설 n_components에 정수를 주면 그 개수만큼 주성분을 찾습니다. 50을 주면 분산이 큰 순서로 주성분 50개를 찾아 10,000차원을 50차원으로 줄입니다.

219. n_components=50으로 학습한 PCA 객체 pca의 components_ 배열 크기가 (50,

10000)인 이유는?

① 샘플이 50개, 특성이 10000개라서
② 주성분 50개 각각이 원본 10000차원 공간의 벡터이기 때문
③ 클러스터가 50개라서
④ 50번 반복했기 때문
⑤ 정답이 10000개라서

> **정답** ②
>
> **해설** components_의 첫 차원(50)은 주성분 개수, 두 번째 차원(10000)은 원본 특성 개수입니다. 즉 주성분 하나하나가 원본 10,000차원 공간에서의 방향 벡터입니다.

220. PCA에서 transform() 메서드가 수행하는 일은?

① 데이터를 원본으로 복원한다
② 원본 데이터를 주성분에 투영해 차원이 줄어든 데이터로 변환한다
③ 정답을 예측한다
④ 클러스터 중심을 계산한다
⑤ 분산을 0으로 만든다

> **정답** ②
>
> **해설** transform은 원본 데이터를 찾아낸 주성분 위에 투영(projection)해 차원이 축소된 데이터로 바꿉니다. (300,10000) 데이터를 주성분 50개로 변환하면 (300,50)이 됩니다.

221. (300, 10000) 크기 데이터를 n_components=50인 PCA로 transform하면 결과 크기는?

① (300, 10000)
② (300, 50)
③ (50, 10000)
④ (50, 300)
⑤ (300, 300)

> **정답** ②
>
> **해설** 샘플 수 300은 그대로 유지되고 특성만 10000에서 50으로 줄어 (300, 50)이 됩니다. PCA는 샘플이 아니라 특성(차원)을 줄입니다.

222. PCA의 inverse_transform() 메서드의 역할은?

① 차원을 더 줄인다

② 축소된 데이터를 다시 원본 차원으로 복원(재구성)한다

③ 정답을 만든다

④ 분산을 계산한다

⑤ 클러스터를 나눈다

> **정답** ②
>
> **해설** inverse_transform은 주성분으로 축소했던 데이터를 다시 원본 차원으로 되돌립니다. 50개 주성분으로도 92% 분산을 보존하므로 복원된 이미지가 원본과 거의 비슷합니다.

223. explained_variance_ratio_ 속성이 나타내는 값은?

① 각 샘플까지의 거리

② 각 주성분이 원본 데이터의 분산을 설명하는 비율

③ 클러스터 개수

④ 정답률

⑤ 반복 횟수

> **정답** ②
>
> **해설** explained_variance_ratio_는 각 주성분이 원본 데이터의 분산을 얼마나 설명하는지 나타내는 비율입니다. 이 값들을 모두 더하면 보존된 전체 분산 비율을 알 수 있습니다.

224. 50개 주성분의 explained_variance_ratio_를 모두 더한 값이 0.92라는 것의 의미는?

① 정확도가 92%다

② 50개 주성분만으로 원본 데이터 분산의 92%를 보존하고 있다

③ 92개 클러스터가 있다

④ 92번 반복했다

⑤ 92개 샘플을 썼다

> **정답** ②
>
> **해설** 합이 0.92라는 것은 10,000개 특성을 단 50개 주성분으로 줄였는데도 원본 정보(분산)의 92%를 유지하고 있다는 뜻입니다. 매우 효율적인 압축입니다.

225. PCA의 n_components에 정수 대신 0.5처럼 0~1 사이 실수를 지정하면?

① 주성분을 0.5개 찾는다

② 설명된 분산의 50%에 도달할 때까지 필요한 주성분 개수를 자동으로 찾는다

③ 샘플의 50%만 쓴다

④ 오류가 발생한다

⑤ 분산을 절반으로 줄인다

> **정답** ②
>
> **해설** 0~1 사이 실수를 주면 그 비율만큼의 분산을 보존하는 데 필요한 주성분 개수를 PCA가 자동으로 정합니다. 0.5를 주면 분산 50%를 설명하는 최소 주성분 개수(예: 2개)를 찾습니다.

226. PCA로 차원을 줄인 데이터를 로지스틱 회귀에 넣었을 때 나타난 효과로 옳은 것은?

① 정확도가 크게 떨어졌다

② 원본과 비슷한 높은 정확도를 유지하면서 훈련 속도가 크게 빨라졌다

③ 정답이 필요 없어졌다

④ 표준화가 필수가 됐다

⑤ 클러스터가 생겼다

> **정답** ②
>
> **해설** 10,000개 특성을 50개로 줄였는데도 거의 100%에 가까운 정확도를 유지하면서 훈련 시간은 약 20배 이상 빨라졌습니다. 차원 축소가 성능 저하 없이 속도를 높여준 사례입니다.

227. 고차원 데이터를 2차원 산점도로 시각화하려면 PCA의 n_components를 어떻게 설정하나?

① n_components=10000

② n_components=2

③ n_components=0

④ n_components=100

⑤ 설정할 수 없다

> **정답** ②
>
> **해설** n_components=2로 주성분 2개만 뽑으면 데이터를 2차원 평면에 점으로 찍어 산점도로 시각화할 수 있습니다. 사람이 눈으로 군집 구조를 확인하기 좋습니다.

228. (1000, 100) 크기 데이터에 n_components=10인 PCA를 적용해 transform한 결과로 옳은 것은?

① (10, 100)이 된다

② (1000, 10)이 되어 샘플 1000개는 유지되고 특성만 10개로 줄어든다

③ (100, 10)이 된다

④ (1000, 100) 그대로다

⑤ (10, 1000)이 된다

정답 ②

해설 PCA는 특성(차원)을 줄이는 기법이므로 샘플 수 1000은 그대로 유지되고 특성만 100에서 10으로 줄어 (1000, 10)이 됩니다. 샘플 개수를 바꾸는 군집과 혼동하면 안 됩니다.

229. 여러 주성분 중 '첫 번째 주성분(제1주성분)'의 특징으로 옳은 것은?

① 설명된 분산이 가장 작다
② 설명된 분산이 가장 크다
③ 항상 0이다
④ 샘플 수와 같다
⑤ 마지막에 계산된다

정답 ②

해설 PCA는 분산이 큰 순서대로 주성분을 찾으므로 첫 번째 주성분이 설명된 분산이 가장 큽니다. 즉 데이터를 가장 잘 설명하는 가장 중요한 방향입니다.

230. PCA에서 두 번째 주성분은 첫 번째 주성분과 어떤 관계인가?

① 완전히 같다
② 서로 직교(수직)하며, 첫 주성분이 설명하지 못한 분산 중 가장 큰 방향이다
③ 항상 반대 방향이다
④ 무작위 방향이다
⑤ 첫 주성분의 절반이다

정답 ②

해설 PCA의 주성분들은 서로 직교합니다. 두 번째 주성분은 첫 번째와 수직이면서, 첫 번째가 담지 못한 남은 분산을 가장 많이 설명하는 방향으로 정해집니다.

231. PCA를 적용하기 전에 보통 데이터를 '평균이 0이 되도록' 중심화(centering)하는 이유는?

① 계산을 느리게 하려고
② 분산(퍼짐)의 방향을 올바로 찾기 위해 원점을 데이터 중심에 맞추기 때문
③ 샘플을 줄이려고
④ 정답을 만들려고
⑤ 색을 바꾸려고

정답 ②

해설 PCA는 데이터가 퍼진 방향(분산)을 찾는데, 중심이 원점에서 벗어나 있으면 방향이 왜곡

됩니다. 그래서 평균을 빼 원점을 데이터 중심으로 옮긴 뒤 주성분을 계산합니다.

232. 사이킷런 PCA는 fit할 때 데이터의 평균을 빼는 중심화를 자동으로 해 주는가?

① 아니다 — 사용자가 직접 빼야 한다
② 그렇다 — 내부에서 자동으로 평균을 빼고 주성분을 계산한다
③ 평균을 더한다
④ 중심화하지 않는다
⑤ 표준편차로 나누기만 한다

정답 ②

해설 사이킷런 PCA는 fit 과정에서 각 특성의 평균을 자동으로 빼(centering) 줍니다. 다만 스케일(표준편차)까지 맞추진 않으므로, 단위가 크게 다른 특성은 별도 표준화를 고려합니다.

233. 사용하는 주성분의 개수를 늘릴수록 explained_variance_ratio_의 합은 어떻게 변하는가?

① 줄어든다
② 점점 커지며(단조 증가), 원본 특성 수만큼 모두 쓰면 1.0(100%)에 도달한다
③ 항상 0.5다
④ 변하지 않는다
⑤ 음수가 된다

정답 ②

해설 주성분을 더할수록 설명되는 분산이 누적되어 합이 증가합니다. 가능한 주성분을 모두 사용하면 원본 분산을 전부 설명해 합이 1.0이 됩니다.

234. 주성분을 분산이 큰 순서로 정렬했을 때, 뒤쪽(분산이 작은) 주성분들이 주로 담고 있는 것은?

① 가장 중요한 정보
② 분산이 작은 방향 — 흔히 잡음(noise)이나 사소한 변화
③ 정답 레이블
④ 샘플 개수
⑤ 클러스터 중심

정답 ②

해설 분산이 작은 방향은 데이터의 미세한 변동이나 잡음에 해당하는 경우가 많습니다. 그래서 이런 주성분을 버리면 차원이 줄면서 잡음 제거(노이즈 감소) 효과도 얻을 수 있습니다.

235. PCA로 차원을 줄이면 정보가 손실되는가?

① 전혀 손실되지 않는다

② 그렇다 — 버린 주성분이 설명하던 분산만큼 정보가 손실되며, 보존 비율로 손실 정도를 가늠한다

③ 정보가 오히려 늘어난다

④ 샘플만 손실된다

⑤ 정답이 손실된다

> **정답** ②
>
> **해설** 주성분 일부를 버리면 그 방향의 분산만큼 정보가 사라집니다. explained_variance_ratio_의 합(예: 0.92)이 곧 보존된 정보 비율이며, 나머지(0.08)가 손실분입니다.

236. 스크리 그래프(scree plot)는 무엇을 시각화한 것인가?

① 샘플 간 거리

② 각 주성분이 설명하는 분산(비율)을 순서대로 나타낸 그래프 — 적절한 주성분 수 결정에 사용

③ 정답률 변화

④ 클러스터 중심 위치

⑤ 학습 곡선

> **정답** ②
>
> **해설** 스크리 그래프는 주성분 번호에 따른 설명된 분산을 그린 그림입니다. 곡선이 완만해지는(꺾이는) 지점을 보고 몇 개의 주성분을 남길지 판단합니다.

237. n_components=2로 줄인 데이터를 산점도로 그렸더니 같은 클래스끼리 가깝게 모여 보였다. 이로부터 알 수 있는 것은?

① 원본에서는 절대 구분되지 않는다

② 원본 고차원 공간에서도 그 그룹들이 어느 정도 잘 구분된다는 신호다

③ PCA가 실패했다

④ 샘플이 줄었다는 뜻이다

⑤ 정답이 필요하다

> **정답** ②
>
> **해설** 2개 주성분만으로도 그룹이 분리돼 보인다면, 분산이 큰 핵심 방향에서 이미 그룹 차이가 드러난다는 의미입니다. 즉 원본 공간에서도 그 그룹들이 비교적 잘 구분됨을 시사합니다.

238. (샘플 1000, 특성 100) 데이터에 PCA를 적용할 때 찾을 수 있는 주성분의 최대 개수는?(샘플이 충분할 때)

① 1000개

② 100개 — 보통 특성 수를 넘을 수 없다

③ 10000개

④ 1개

⑤ 무한대

> **정답** ②
>
> **해설** 주성분 개수는 특성 수(와 샘플 수) 중 작은 값을 넘을 수 없습니다. 특성이 100개이고 샘플이 충분(1000개)하므로 최대 100개의 주성분을 찾을 수 있습니다.

239. PCA의 주성분(components_의 각 행)은 무엇으로 이루어진 벡터인가?

① 샘플 하나의 값

② 원본 각 특성에 대한 가중치들의 조합 — 즉 원본 특성들의 선형 결합 방향

③ 정답 레이블

④ 클러스터 번호

⑤ 거리 값

> **정답** ②
>
> **해설** 각 주성분은 원본 특성마다 곱해지는 가중치(계수)들의 벡터입니다. 이 가중치로 원본 특성들을 선형 결합한 방향이 바로 그 주성분이며, 데이터를 이 방향에 투영해 차원을 줄입니다.

240. PCA로 변환(축소)해 얻은 새 특성들은 원본 특성과 같은 것인가?

① 같다 — 원본 특성 일부를 그대로 쓴 것이다

② 아니다 — 원본 특성들을 조합해 만든 '새로운 축(주성분)' 위의 값이다

③ 정답 레이블이다

④ 샘플 번호다

⑤ 항상 0과 1이다

> **정답** ②
>
> **해설** PCA가 만든 특성은 기존 특성을 골라낸 것이 아니라, 여러 특성을 가중 결합해 만든 새 좌표축(주성분) 상의 값입니다. 그래서 원래의 '알코올·당도' 같은 직접적 의미와는 다릅니다.

241. 고차원 데이터를 PCA로 줄인 뒤 분류기에 넣으면 학습이 빨라지는 근본 이유는?

① 샘플이 줄어서

② 특성(차원) 수가 줄어 모델이 처리할 계산량이 감소하기 때문

③ 정답이 늘어서

④ 표준화가 돼서

⑤ 트리가 생겨서

> **정답** ②
>
> **해설** 모델의 계산량은 특성 수에 크게 좌우됩니다. PCA로 1만 차원을 수십 차원으로 줄이면 계산이 가벼워져, 정확도를 거의 유지하면서도 학습·예측 속도가 크게 빨라집니다.

242. PCA가 차원을 줄이면서도 분류 성능을 거의 유지할 수 있는 이유는?

① 정답을 사용해서

② 분산이 큰(정보가 많은) 방향을 우선 남기므로 분류에 중요한 정보가 대부분 보존되기 때문

③ 샘플을 늘려서

④ 트리를 써서

⑤ 운이 좋아서

> **정답** ②
>
> **해설** PCA는 분산이 큰 방향부터 남깁니다. 데이터의 핵심 변동(클래스 구분에 기여하는 정보)이 대체로 이 방향에 담겨 있어, 차원을 줄여도 성능이 잘 유지됩니다.

243. explained_variance_ratio_ 배열에서 첫 번째 값이 두 번째 값보다 항상 크거나 같은 이유는?

① 우연이다

② PCA가 설명하는 분산이 큰 순서대로 주성분을 정렬하기 때문

③ 값을 무작위로 배열해서

④ 정답 순서라서

⑤ 샘플 순서라서

> **정답** ②
>
> **해설** PCA는 분산을 가장 많이 설명하는 주성분을 1번으로, 그다음을 2번으로... 내림차순 정렬합니다. 따라서 explained_variance_ratio_도 앞쪽이 더 크거나 같은 값으로 나옵니다.

244. 데이터 분산을 99% 보존하려는 경우와 80% 보존하려는 경우 중, 일반적으로 더 많은 주성분이 필요한 쪽은?

① 80%를 보존하는 쪽

② 99%를 보존하는 쪽 — 더 많은 분산을 담으려면 주성분이 더 필요하다

③ 둘은 항상 같다

④ 주성분이 필요 없다

⑤ 보존율과 무관하다

> **정답** ②
>
> **해설** 더 많은 분산을 보존할수록 그만큼 더 많은 주성분이 필요합니다. 99% 보존은 80% 보존보다 일반적으로 더 많은 주성분을 요구하며, 그만큼 차원 축소 효과는 작아집니다.

245. PCA로 압축한 뒤 inverse_transform으로 복원한 데이터가 원본과 '완전히 똑같지는 않은' 이유는?

① 코드 오류 때문

② 버린 주성분이 담고 있던 정보가 사라져 완벽 복원이 불가능하기 때문(손실 압축)

③ 샘플이 늘어서

④ 정답이 없어서

⑤ 복원은 항상 완벽하다

> **정답** ②
>
> **해설** 차원 축소 과정에서 일부 주성분을 버리므로 그 정보는 되돌릴 수 없습니다. 따라서 복원 이미지는 원본과 비슷하지만 완전히 같지는 않은 손실 압축의 결과입니다.

246. PCA는 지도 학습인가, 비지도 학습인가?

① 지도 학습 — 정답을 사용한다

② 비지도 학습 — 정답 레이블 없이 데이터의 분산 구조만으로 주성분을 찾는다

③ 강화 학습이다

④ 둘 다 아니다

⑤ 정답이 반드시 필요하다

> **정답** ②
>
> **해설** PCA는 타깃(정답)을 쓰지 않고 입력 데이터의 분산만으로 주성분을 찾는 비지도 학습입니다. 군집과 마찬가지로 정답 없이 데이터 구조를 분석합니다.

247. 차원 축소가 데이터 시각화에 특히 유용한 이유는?

① 색을 입혀 줘서

② 사람이 볼 수 있는 2~3차원으로 줄여 고차원 데이터의 구조를 그림으로 확인할 수 있기 때문

③ 정확도를 높여서

④ 샘플을 늘려서

⑤ 정답을 만들어 줘서

> **정답** ②
>
> **해설** 사람은 2~3차원만 직관적으로 볼 수 있습니다. PCA로 고차원 데이터를 2~3개 주성분으로 줄이면 산점도로 그려 군집·분포·이상치 같은 구조를 눈으로 파악할 수 있습니다.

248. PCA가 특성 간 상관관계가 높은(중복된 정보가 많은) 데이터에서 특히 효과적인 이유는?

① 상관이 높으면 PCA가 작동하지 않는다

② 상관된 특성들의 공통 변동을 소수의 주성분으로 압축할 수 있어, 적은 차원으로 많은 정보를 담기 때문

③ 정답이 많아서

④ 샘플이 많아서

⑤ 표준화 때문

> **정답** ②
>
> **해설** 여러 특성이 비슷하게 움직이면(높은 상관) 그 정보는 사실상 중복입니다. PCA는 이 공통 변동을 하나의 주성분으로 모아, 적은 수의 주성분만으로 데이터를 잘 표현할 수 있습니다.

249. PCA를 사용하기에 적절하지 않은 경우의 예로 가장 가까운 것은?

① 특성이 수천 개로 매우 많을 때

② 특성이 이미 2~3개로 매우 적어 더 줄일 필요가 없을 때

③ 특성 간 상관이 높을 때

④ 시각화를 하고 싶을 때

⑤ 계산을 빠르게 하고 싶을 때

> **정답** ②
>
> **해설** 차원 축소는 특성이 많아 계산·시각화가 부담스러울 때 유용합니다. 이미 특성이 2~3개로 적다면 줄일 이유가 거의 없고, 오히려 정보 손실만 생길 수 있습니다.

250. 특성 선택(feature selection)과 PCA(특성 추출)의 차이로 옳은 것은?

① 둘은 완전히 같다

② 특성 선택은 기존 특성 중 일부를 '그대로 고르는' 것이고, PCA는 기존 특성들을 '조합해 새 특성'을 만드는 것이다

③ 특성 선택이 항상 더 좋다

④ PCA는 특성을 그대로 고른다

⑤ 특성 선택은 비지도가 아니다

정답 ②

해설 특성 선택은 원래 특성 중 유용한 것을 추려 남기므로 의미가 보존됩니다. PCA는 여러 특성을 결합해 새 축(주성분)을 만드는 특성 추출이라, 차원은 더 줄지만 개별 축의 직접적 의미는 약해집니다.