1. Data Preprocessing (1)

빅데이터와 금융자료 분석

김아현

데이터 정제 : 결측값 처리

- 결측값 처리
  - 결측값의 유형
    - 완전 무작위 결측(MCAR, Missing Completely At Random): 결측값이 다른 변수들과 독립적일 때 발생한다.
      - 설문 조사에서 응답자가 페이지를 넘기는 것을 잊어서 발생한 결측치.
    - 무작위 결측(MAR, Missing At Random): 결측값이 다른 관찰된 변수들에 의존적일 경우에 발생한다.
      - 남성이 건강에 관한 질문에 답하기를 꺼려하는 경향이 있는 경우
    - 비무작위 결측(NMAR, Not Missing At Random): 결측값이 다른 관찰된 변수들뿐 아니라 결측값을 포함한 변수의 값에도 의존적일 경우에 발생한다.
      - 소득이 높은 사람들이 소득에 관한 질문을 회피하는 경우

데이터 정제 : 결측값 처리

- 결측값 처리
  - 결측값 처리의 중요성
    - 결측값을 처리하기 위한 다양한 방법이 있는데, 어떻게 처리하는지에 따라 분석의 정확도와 모델의 품질에 직접적인 영향을 미침.
    - 데이터의 특성, 결측치의 양과 패턴, 분석의 목적 등을 고려하여 적절한 결측값 처리 방법을 선택해야 함.

  - 결측값 처리 방법
    - 결측을 포함한 행 데이터나 변수를 제외
      - 간단하고 쉽게 적용 가능
      - 유용한 정보를 잃을 수 있고, 데이터가 편향될 위험이 있음.

    - 결측값인 채 처리하기
      - 결측값이 있어도 이를 무시하고 처리 가능한 알고리즘들이 있음.
      - 일부 알고리즘은 학습하는 동안 결측값을 처리하는 과정까지 포함되어 있음.

데이터 정제 : 결측값 처리

- 결측값 처리
  - 결측값 처리 방법
    - 대푯값으로 채우기
      - 결측값이 랜덤하게 발생한다면 자주 나올만한 값으로 채운다는 아이디어에 기반함.
      - 데이터의 손실을 줄이고, 구현이 간단하고 빠름.
      - 데이터의 분산을 감소시키고, 실제 분포를 왜곡할 수 있음.
      - 변수들 간의 상관관계를 고려하지 않음.

      - 수치형 변수
        - 평균, 중앙값 등
        - 다른 범주형 특성 변수로 그룹을 만든 뒤 해당 그룹 별 평균, 중앙값 등을 대입

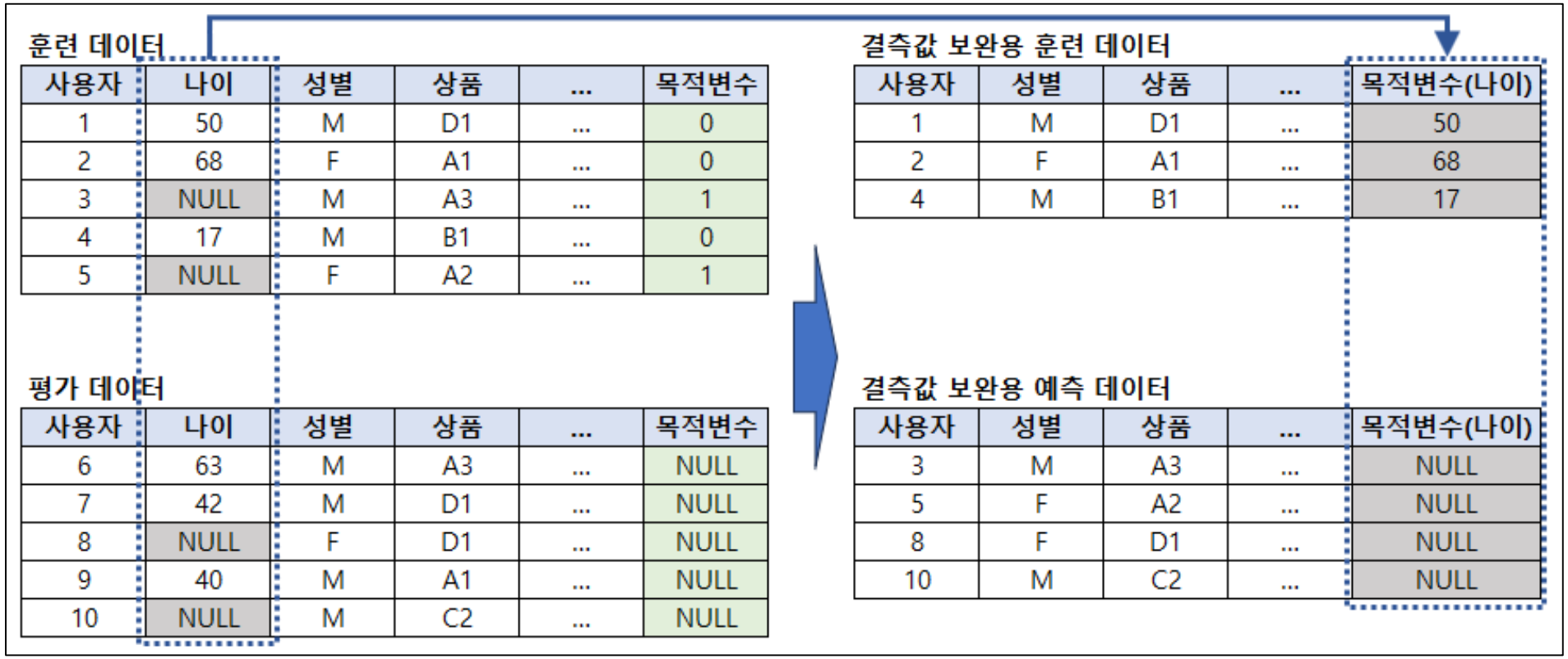
      - 범주형 변수
        - 해당 특성의 범주값 중 최빈값
        - 결측값 자체를 하나의 새로운 범주로 취급

데이터 정제 : 결측값 처리

- 결측값 처리
  - 결측값 처리 방법
    - 예측 모델을 이용하여 결측값 예측하기 – Regression 활용
      - 결측값을 가진 변수가 다른 변수와 관련이 있는 경우에 고려될 수 있음.
      - 결측값을 가진 변수를 목적변수로 간주, 그 외의 변수를 특성변수로 삼는 모델을 구축한 뒤, 이 모델에서 예측을 수행한 값으로 결측값을 채움.
      - 데이터의 패턴을 유지하며, 결측값을 보다 정교하게 추정할 수 있음.
      - 모델 구축이 복잡하고 상대적으로 시간이 많이 소요됨.

데이터 정제 : 결측값 처리

- 결측값 처리
  - 결측값 처리 방법
    - 예측 모델을 이용하여 결측값 예측하기 – Regression 활용



데이터 정제 : 결측값 처리

- 결측값 처리
  - 결측값 처리 방법
    - 예측 모델을 이용하여 결측값 예측하기 - KNN(K-Nearest Neighbors) 활용
      - 결측값을 포함한 데이터와 가장 가까운 K개의 이웃 데이터를 찾고, 이웃 데이터들의 정상값들을 평균한 값으로 대체함.
      - 데이터들의 로컬 구조를 반영할 수 있음.
      - 계산 비용이 높으며, 이상값에 취약함.

데이터 정제 : 결측값 처리

- 결측값 처리
  - 결측값 처리 방법
    - 예측 모델을 이용하여 결측값 예측하기 - KNN(K-Nearest Neighbors) 활용

ID	X1	X2	X3	X4	X5
1	80	30	7	14	27
2	44	NULL	10	0	29
3	NULL	85	25	5	88
4	50	70	74	9	49
5	29	54	49	20	NULL

데이터 정제 : 결측값 처리

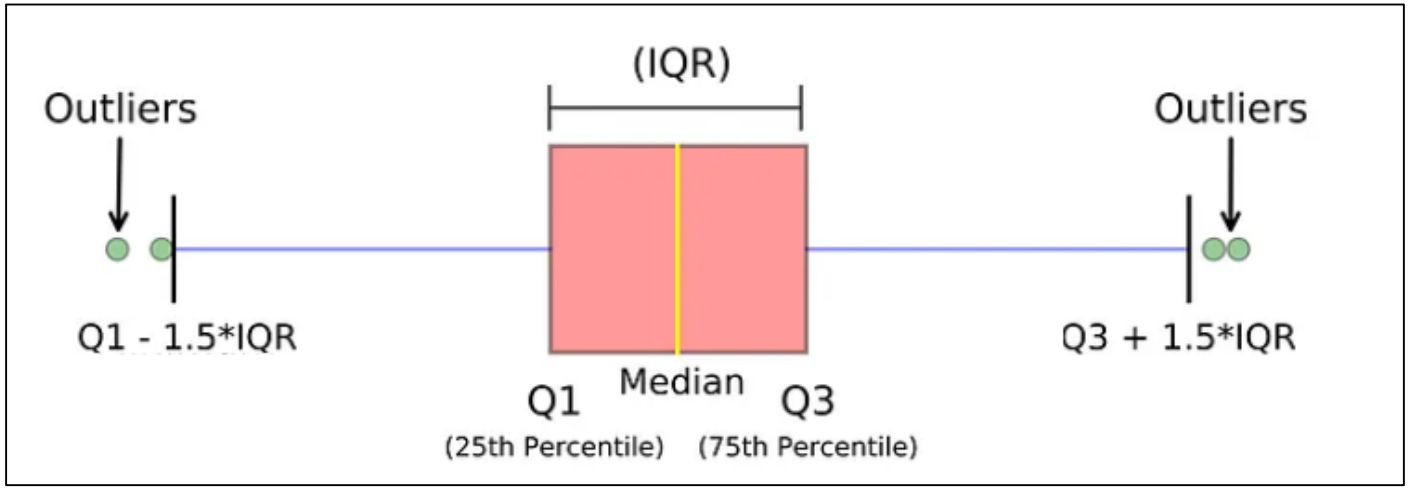
- 결측값 처리
  - 결측값 처리 방법
    - 결측값으로 새로운 특징 만들기
      - 결측 여부에 대한 더미 변수.
      - 각 행 별로 결측값이 있는 특성변수의 수를 카운팅
      - 여러 개의 특성 변수에서 관찰되는 결측값들의 조합을 조사하여 몇 개의 패턴으로 분류할 수 있다면 어느 패턴인지를 하나의 특성변수로 생성.
    - 시계열 데이터에서 자주 적용되는 결측값 처리
      - 직전 또는 직후 값으로 채우기
      - 선형 보간법을 적용하여 채우기

데이터 정제 : 이상값 처리

- 이상값 처리
  - 이상값
    - 어느 특성 변수에 대해 대부분의 관찰 값이 포함되는 범위에서 많이 벗어난, 아주 작거나 큰 값을 의미함.
    - 이상값이 포함된 데이터는 분석 결과가 왜곡될 수 있어, 미리 진단 및 처리해 주어야 함.

데이터 정제 : 이상값 처리

- 이상값 처리
  - 이상값의 식별
    - 통계적 방법
      - Z 스코어
        - 어느 관측값의 평균으로부터의 편차가 해당 변수의 표준편차 대비 몇 배인지를 나타냄.
        - Z 스코어의 절대값이 3 등의 임계값을 넘으면 잠재적인 이상값으로 진단함.
      - 박스플롯 이용
        - Q1 – 1.5×IQR 보다 작거나, Q3 + 1.5×IQR 보다 큰 값들을 이상값으로 진단함.

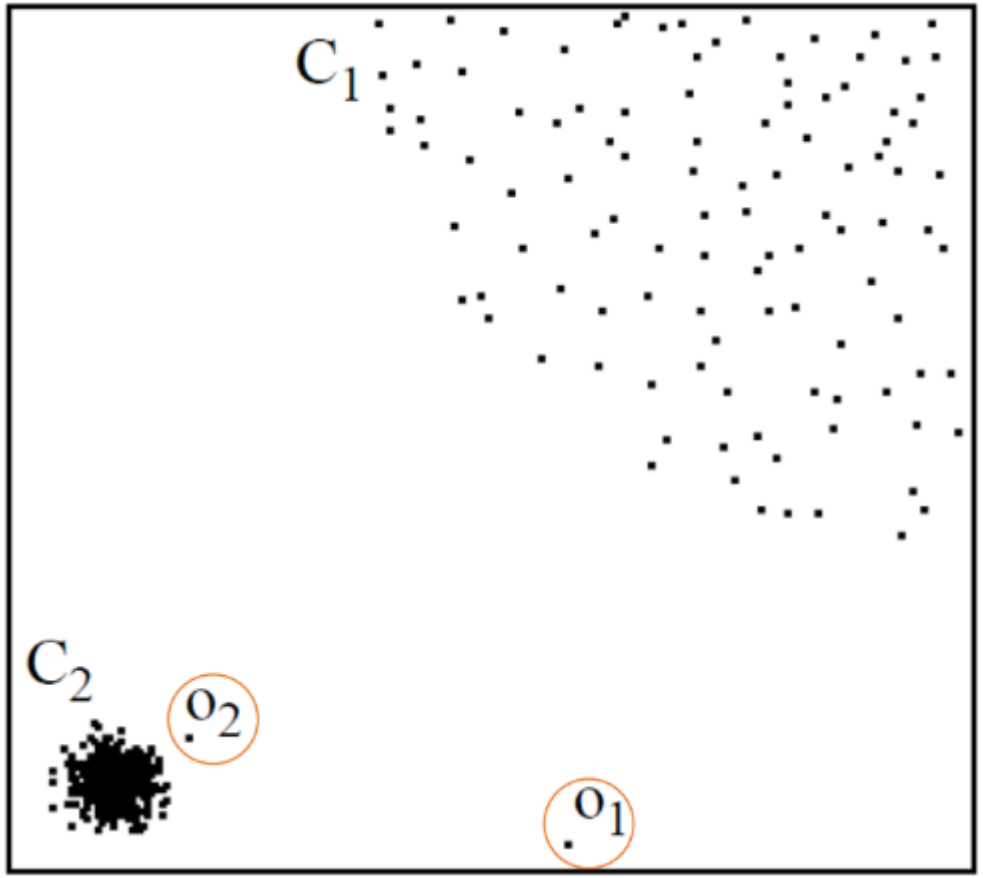


데이터 정제 : 이상값 처리

- 이상값 처리
  - 이상값의 식별
    - 머신러닝 및 딥러닝 기법
      - LOF (Local Outlier Factor)
      - Isolation Forest
      - OC-SVM (One Class Support Vector Machine)
      - Clustering
      - AutoEncoder

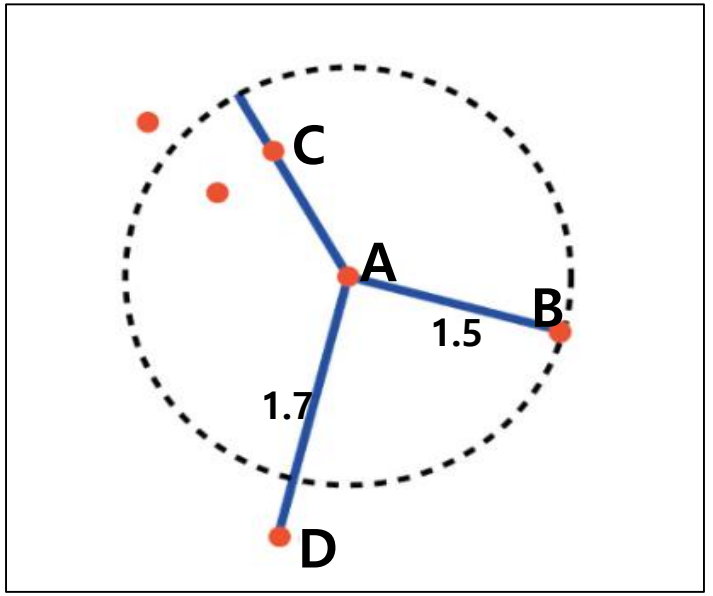
데이터 정제 : 이상값 처리

- LOF (Local Outlier Factor)
  - LOF 개요
    - 밀도 기반의 이상값 탐지기법
    - 주어진 데이터가 이상값이라면 해당 데이터의 밀도가 주변 이웃 데이터들의 밀도 보다 작을 것이라는 아이디어에서 착안
    - 모든 데이터를 고려하지 않고, 해당 데이터의 주변 데이터들의 밀도만 고려한다는 것이 핵심.



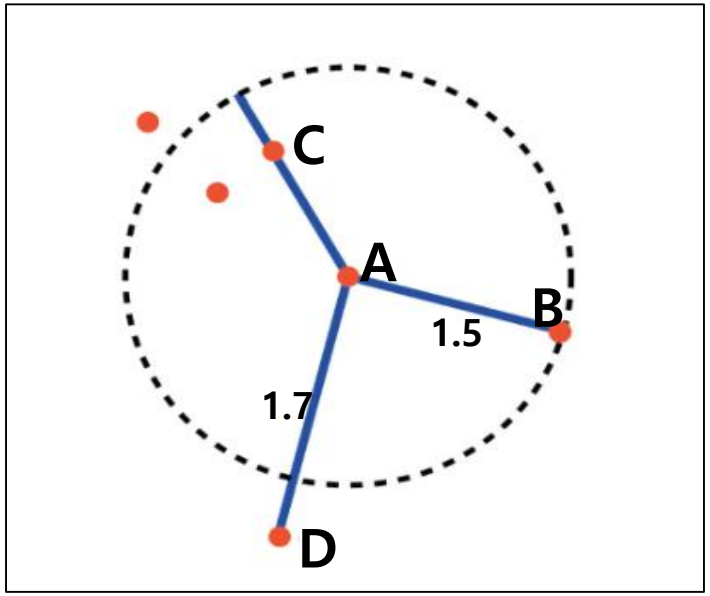
데이터 정제 : 이상값 처리

- LOF (Local Outlier Factor)
  - LOF 훈련 알고리즘
    - 하이퍼파라미터 k
      - 어느 관찰값을 기준으로 그 주변 관찰치를 몇 개까지 포함할 것인지를 결정
    - $k - distance(A)$
      - 특정 관찰값 A를 기준으로 k nearest neighbor까지의 거리
      - 예. $3 - distance(A) = 1.5$
    - $N_k(A)$
      - $k - distance(A)$ 안에 들어오는 데이터의 수
      - Tie가 존재하는 경우에는 k보다 커짐
      - 예. $N_3(A) = 3$



데이터 정제 : 이상값 처리

- LOF (Local Outlier Factor)
  - LOF 훈련 알고리즘
    - Reachability-Distance
      - $RD(A,B) = \max(k - distance(B), dist(A,B))$
      - A와 B가 가까이 있는 경우에도,
        B의 $k - distance$만큼의 거리를 배정함
      - 그렇지 않으면 A와 B의 실제 거리를 배정함.
      - 예. $RD(C,A) = 1.5,\ RD(D,A) = 1.7$

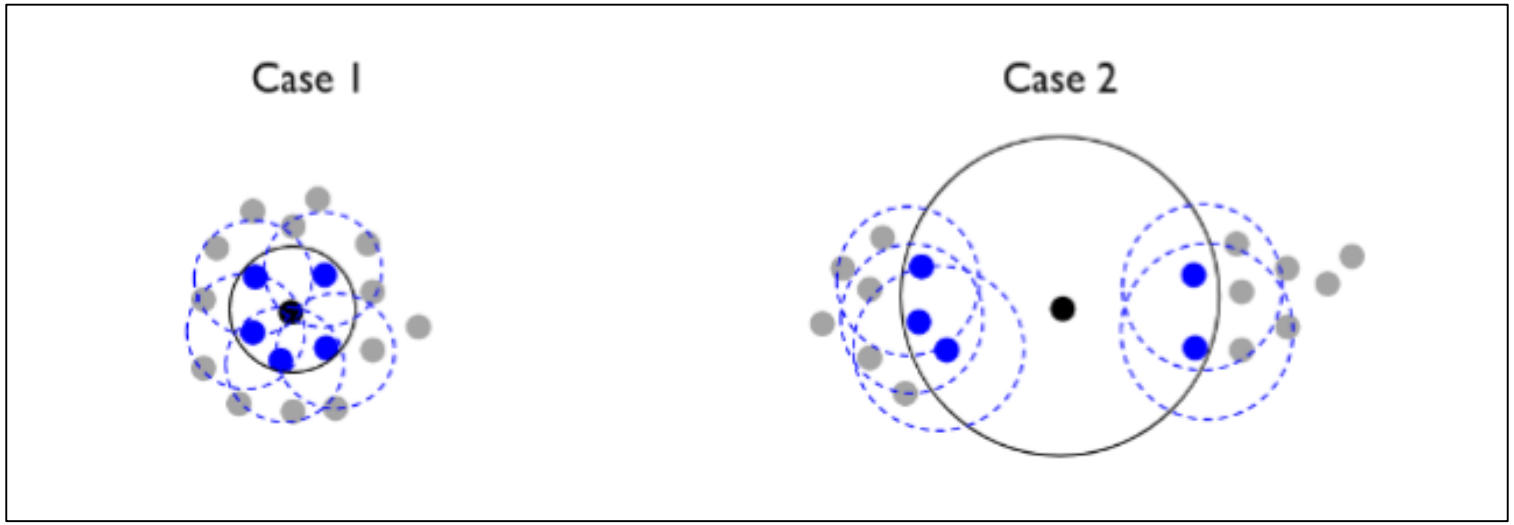


데이터 정제 : 이상값 처리

- LOF (Local Outlier Factor)
  - LOF 훈련 알고리즘
    - Local Reachability Density (LRD)

$$LRD_k\ (A) = \frac{1}{\sum_{B \in N_k(A)} RD(A,B)\ / N_k(A)}$$

      - A가 A와 가까운 B들 사이에서 얼마나 떨어져 있는지를 측정함
      - A가 주변 데이터의 공간으로부터 떨어져 있을수록 LRD는 작아짐.
      - 예. Case1의 LRD > Case2의 LRD



데이터 정제 : 이상값 처리

- LOF (Local Outlier Factor)
  - LOF 훈련 알고리즘
    - Local Outlier Factor (LOF)
      - 주변 데이터의 LRD와 A의 LRD 비율의 평균

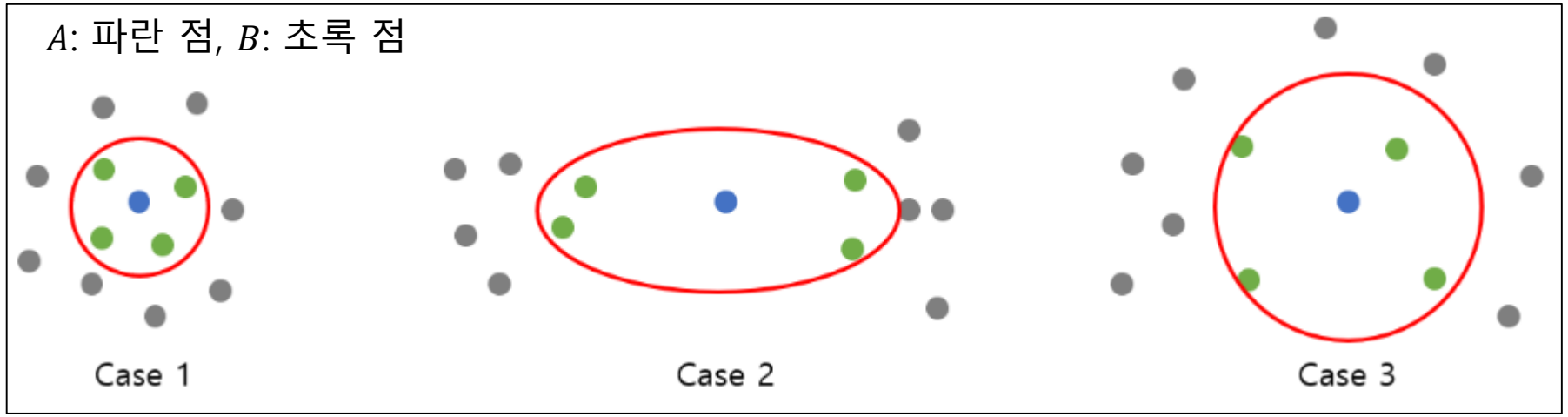
$$LOF_k(A) = \frac{\sum_{B \in N_k(A)} \frac{LRD_k(B)}{LRD_k(A)}}{N_k(A)}$$

      - LOF가 크면 주변 데이터들이 뭉쳐 있는 정도보다 훨씬 많이 떨어져 있음을 의미하므로, LOF 값이 클수록 Outlier 정도가 심한 것으로 해석할 수 있음.

      - 어떤 임계값 c를 정한 뒤, 이를 기준으로 그보다 크면 Outlier로 진단함.
        - LOF < c : higher density than neighbors (Inlier)
        - LOF > c : lower density than neighbors (Outlier)

데이터 정제 : 이상값 처리

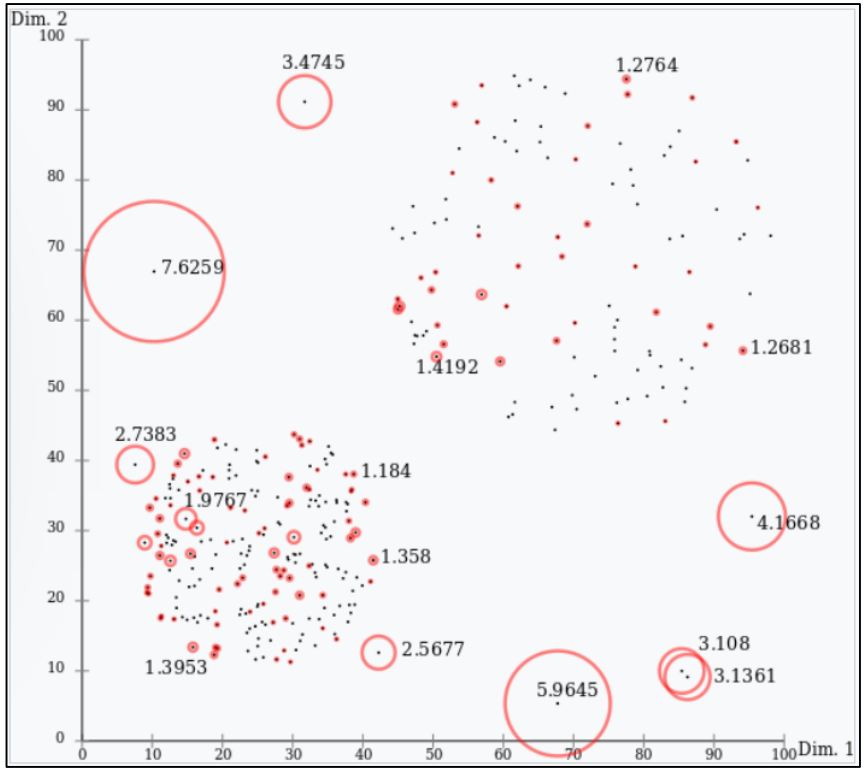
- LOF (Local Outlier Factor)
  - LOF 훈련 알고리즘
    - Local Outlier Factor (LOF)



Case	$LRD_k(A)$	$LRD_k(B)$	$LOF_k(A)$
Case 1	Large	Large	Small
Case 2	Small	Large	Large
Case 3	Small	Small	Small

데이터 정제 : 이상값 처리

- LOF (Local Outlier Factor)
  - LOF 훈련 알고리즘
    - Local Outlier Factor (LOF)
      - 분석 사례

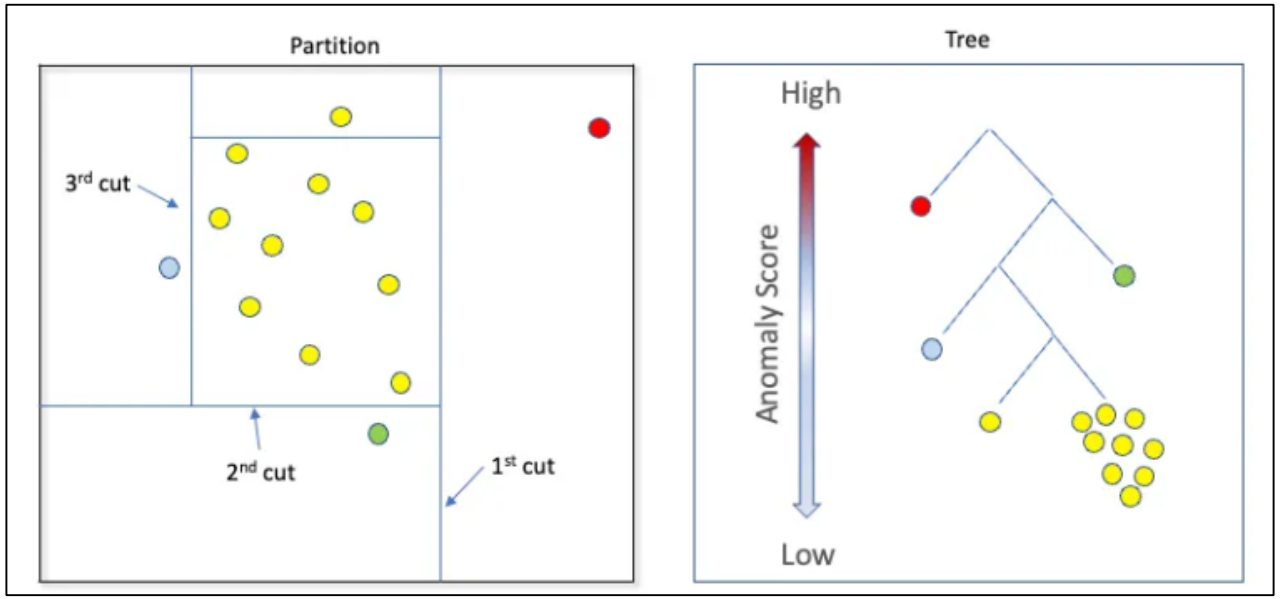


데이터 정제 : 이상값 처리

- LOF (Local Outlier Factor)
  - LOF 특징
    - 주변 데이터들의 상대적인 밀도를 이용하는 방식
      - 밀집된 군집 주변의 데이터는 조금만 떨어져도 이상값으로 진단 가능함.
    - 이상값 판단 기준을 결정하기 어려움.
    - 하이퍼파라미터 k에 민감하게 성능이 변함.
    - 데이터의 크기가 큰 경우에 계산시간이 오래 걸림.

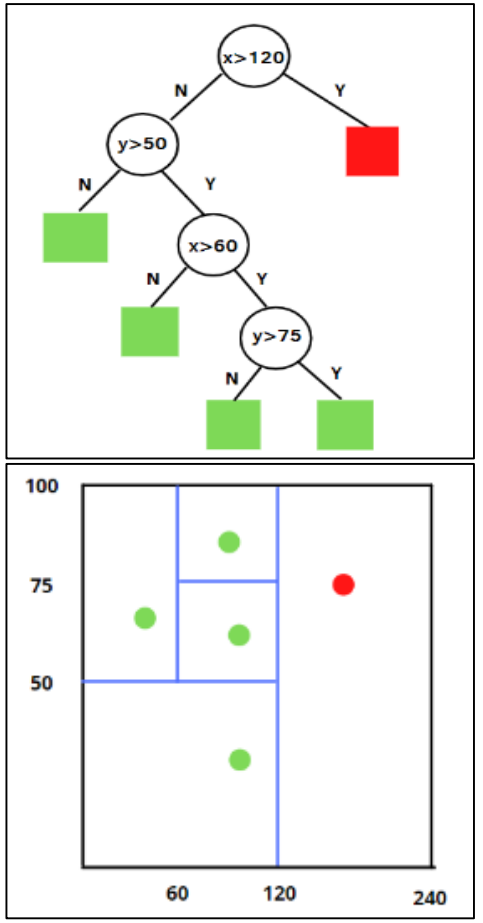
데이터 정제 : 이상값 처리

- Isolation Forest
  - Isolation Forest 개요
    - Isolation Forest 란
      - 비지도학습법으로 데이터에서 이상값를 탐지하기 위한 방법론
      - 여러 트리를 결합한 앙상블 구조로, 이상값은 정상 데이터에 비해 트리에서 고립이 더 잘될 것이라는 아이디어에 착안하여 개발됨.
      - 각 데이터에 대하여 이상값 점수를 계산하고, 점수가 높은 경우 이상값으로 판단함.



데이터 정제 : 이상값 처리

- Isolation Forest
  - Isolation Forest 훈련 알고리즘
    - iTree를 훈련
      - Sub-sampling : 비복원 추출로 데이터 중 일부를 샘플링
      - Splitting rule
        - 랜덤하게 하나의 feature Q를 선택
        - Q의 범위(min~max) 중 uniform하게 split point를 선택
      - 아래 조건을 만족할 때까지 반복적으로 분기하여 iTree를 생성함.
        - 사전에 설정한 최대 깊이에 도달
        - 분리하였을 때 자식노드에 하나의 데이터만 남은 경우
        - 분리하였을 때 자식노드에 데이터 값이 모두 동일한 경우
    - 위 과정을 반복하여 여러 개의 iTree 도출



데이터 정제 : 이상값 처리

- Isolation Forest
  - Anomaly Score (이상값 스코어) 산출
    - 새로운 관찰치가 주어지면, 각 iTree 별 path length(분리 횟수)를 구하고, 이를 통합한 평균 path length 를 기반으로 Anomaly Score $s(x,n)$ 를 계산하여 이상값 여부를 판별함.

$$s(x,n) = 2^{-\frac{E[h(x)]}{c(n)}}$$

    - $h(x)$: 해당 관찰치의 path length
    - $E[h(x)]$: 모든 iTree에서 해당 관찰치에 대한 평균 path length
    - $c(n)$: $h(x)$ 를 normalize 하기 위한 값으로, iTree의 평균길이.

$$c(n) = 2H_{n-1} - \frac{2(n-1)}{n}, H_n = 1 + \frac{1}{2} + \cdots + \frac{1}{n}$$

    - $E[h(x)]$ 에 따른 $s(x,n)$ 의 값
      - 관찰치 x 의 최대경로길이 : $E[h(x)] \to (n-1), s$ 는 0에 가까워짐
      - 관찰치 x 가 전체 경로길이의 평균과 유사: $E[h(x)] \to c(n),\ s \to 0.5$
      - 관찰치 x 가 이상값 : $E[h(x)] \to 0,\ s \to 1$

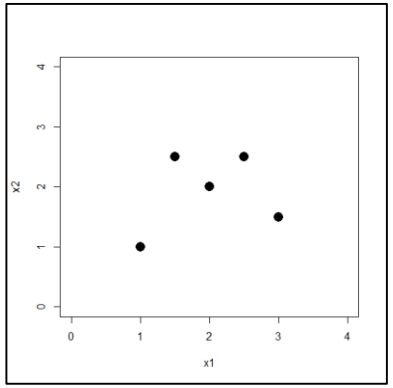
데이터 정제 : 이상값 처리

- Isolation Forest
  - 예제

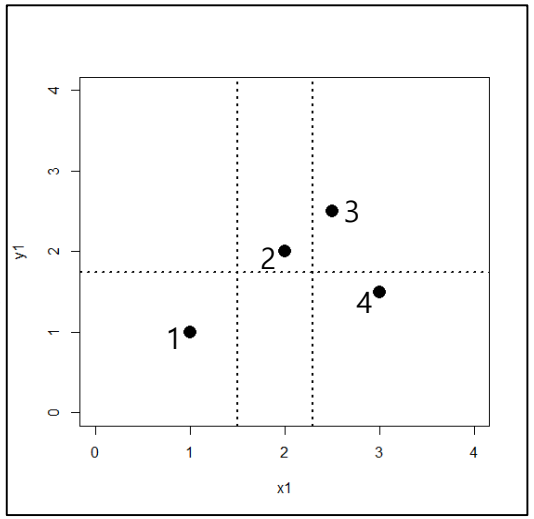
하이퍼 파라미터 설정
- iTree 개수 :2
- Sample 수 : 4
- 최대깊이 :제한없음

원본 데이터

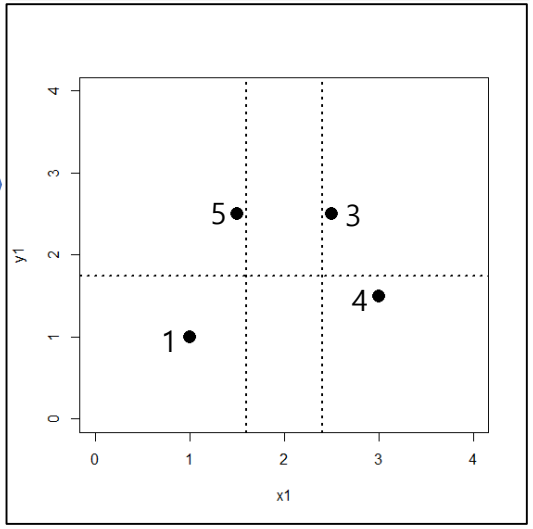
Id	x1	x2
1	1	1
2	2	2
3	2.5	2.5
4	3	1.5
5	1.5	2.5



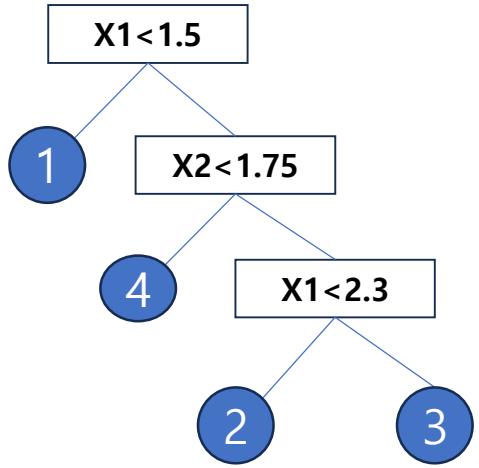
Sub-Sampling



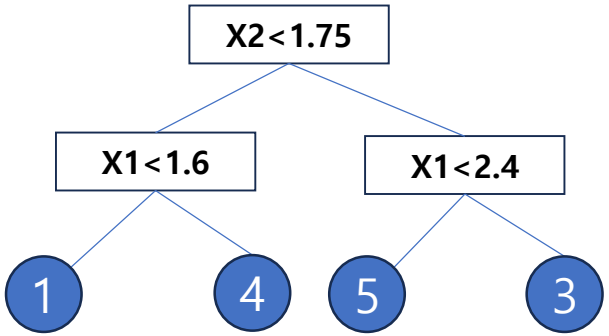
Sub-Sampling



<iTree 1>

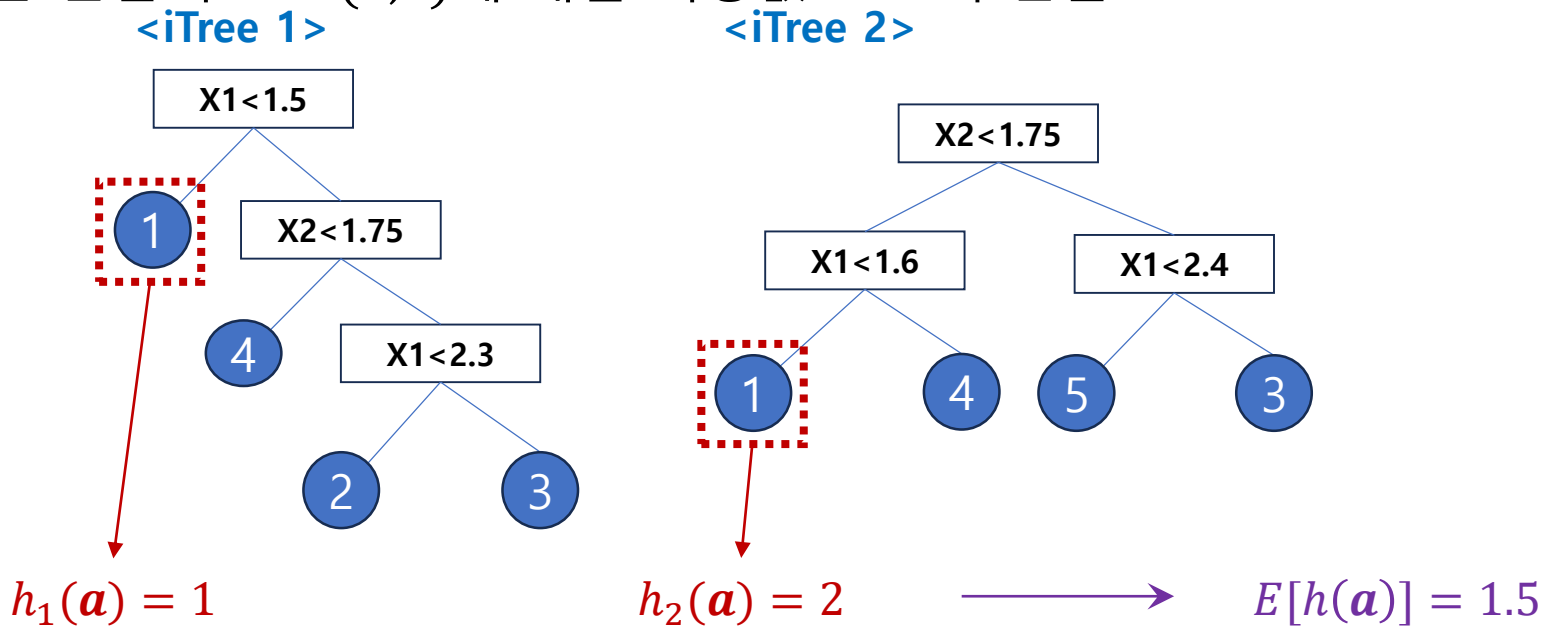


<iTree 2>



데이터 정제 : 이상값 처리

- Isolation Forest
  - 예제
    - 새로운 관찰치 $\boldsymbol{a} = (1,1)$에 대한 이상값 스코어 산출



- $n = 5$이므로, $c(5) = 2 \cdot H_4 - \frac{8}{5} = 2 \cdot \frac{25}{12} - \frac{8}{5} = 2.57.$
- $s(\boldsymbol{a}, 5) = 2^{-(\frac{1.5}{2.57})} = 2^{-0.58} = 0.67$

데이터 정제 : 이상값 처리

- Isolation Forest
  - Isolation Forest 특징
    - 다른 이상탐지 알고리즘에 비해 속도가 빠른 편임.
    - 샘플링을 통해 swamping과 masking 문제 해결 가능함.
    - 데이터 분포에 대한 가정이 필요없음.
    - 고차원의 데이터에서도 잘 작동함.
    - 샘플 수와 iTree 개수에 따라 결과가 불안정한 편임.

특성 공학

- 베이스라인 모델과 특성
  - 트리 계열
    - 수치의 크기(범위) 자체에는 큰 의미가 없고 크고 작은 관계에만 영향을 받음.
    - 결측값이 있어도 처리 가능한 경우가 많음.
    - 트리의 분기 과정에서 특성변수 간의 상호작용이 반영됨.
  - 신경망 계열
    - 수치값의 범위에 영향을 받음.
    - 결측값이 있으면 이를 채워야 처리 가능한 경우가 많음.
    - 이전 층의 출력을 결합하여 계산하는 연산으로 특성변수 간 상호작용이 반영됨.

특성 공학

- 특성 공학
  - 특성변수의 변환
    - 수치형 변수 변환
      - 표준화 $x' = \frac{x-\bar{x}}{s}$ (해당 변수의 평균 : $\bar{x}$, 표준편차 : s)
        - 변환된 변수의 평균이 0, 표준편차가 1이 되도록 함.
        - 더미 변수는 표준화를 실시하지 않아도 됨.
        - 훈련 자료와 평가 자료로 구분된 경우
          - 훈련 자료의 평균, 표준편차를 이용하여 평가 자료에 적용할 수 있음.
          - 훈련 자료와 평가 자료를 통합한 자료로 평균, 표준편차를 구한 뒤, 통합된 자료에 한꺼번에 적용할 수 있음.
          - 훈련 자료와 평가 자료가 각각 서로 다른 평균과 표준편차를 이용하여 변환되지 않도록 해야 함.

특성 공학

- 특성 공학
  - 특성변수의 변환
    - 수치형 변수 변환
      - 최소-최대 스케일링 : $x' = \frac{x - x_{min}}{x_{max} - x_{min}}$
        - 변수가 취하는 범위를 특정 구간(보통 0~1)으로 변환함.
        - 변환 뒤 평균이 0이 되지 않으며, 이상값에 민감함.
        - 한정된 범위 이내의 값만 가지도록 정의된 변수에 적절한 변환 방식

특성 공학

- 특성 공학
  - 특성변수의 변환
    - 수치형 변수 변환
      - 비선형 변환
        - 변수의 분포 형태를 바꿀 수 있어, 치우침이 없는 분포를 만들고자 할 때 고려됨.
        - 로그 변환 : $\log(x)$, $\log(x+1)$, $\text{sign(x)} \cdot \log(|x|)$
        - Box-Cox 변환 : $x' = \begin{cases} \frac{x^{\lambda}-1}{\lambda} & , \ if \ \lambda \neq 0 \\ \log(x) & , \ if \ \lambda = 0 \end{cases}$

특성 공학

- 특성 공학
  - 특성변수의 변환
    - 수치형 변수 변환
      - 클리핑 (Clipping)
        - 상한과 하한을 설정한 뒤, 해당 범위를 벗어나면 상한값과 하한값으로 치환함.
        - 이상값의 처리에 자주 사용됨.
      - 구간분할 (binning)
        - 적절하게 정의된 구간을 이용해 수치형 변수를 (순서가 있는) 범주형 변수로 변환.
        - 순서대로 다시 수치화하거나, 범주형 변수로 두고 원-핫 인코딩을 적용할 수 있음.
      - 순위변환
        - 수치의 크기나 간격정보를 버리고 대소관계만을 얻어내는 방법
        - 순위를 행의 수로 나눠 0~1 범위로 바꾸기도 함.
      - RankGauss
        - 순위로 변환한 뒤 순서를 유지한 채 정규분포가 되도록 변환하는 방법.
        - 신경망 등에서 일반 표준화 변환보다 성능이 좋은 것으로 알려짐.

특성 공학

- 특성 공학
  - 특성변수의 변환
    - 범주형 변수 변환
      - 원-핫 인코딩 (One-hot Encoding)
        - 범주형 변수의 각 레벨에 대해 해당 레벨인지 여부를 나타내는 0과 1 두 값을 갖는 더미 변수를 각각 생성함.
        - 특성의 수가 범주형 변수의 레벨 개수에 따라 증가한다는 결점이 있음.
          - 지나치게 많은 더미 변수가 생성되면, 시간이나 메모리가 급증하거나 모델 성능에 악영향을 줄 수 있으므로 범주형 변수의 레벨을 줄여서 처리하는 것이 권장됨.

특성 공학

- 특성 공학
  - 특성변수의 변환
    - 범주형 변수 변환
      - 레이블 인코딩 (Label Encoding)
        - 각 레벨을 단순히 정수로 변환함.
        - 순서형이 아닌 경우 인덱스 수치는 본질적인 의미가 없으며, 트리 모델에 기반을 둔 방법이 아닌 경우는 적절하지 않음.
        - 트리 모델에 기반을 둔 방법에서 범주형 변수를 변환하는 기본적인 방법.

특성 공학

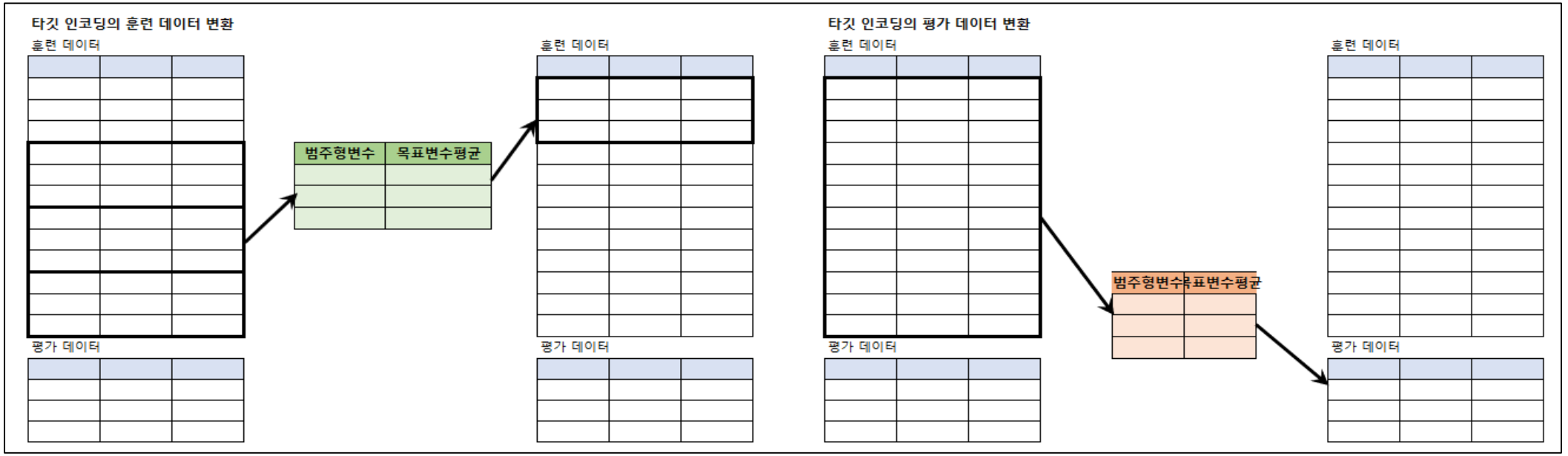
- 특성 공학
  - 특성변수의 변환
    - 범주형 변수 변환
      - 빈도 인코딩 (frequency encoding)
        - 각 레벨의 출현 빈도로 범주형 변수를 대체하는 방법
        - 레이블 인코딩의 변경으로, 각 레벨의 출현 빈도와 목적 변수 간에 관련성이 있을 때 유효함.
        - 동률 발생에 주의해야 함.

특성 공학

- 특성 공학
  - 특성변수의 변환
    - 범주형 변수 변환
      - 타깃 인코딩
        - 목표변수를 이용하여 범주형 변수를 수치화 : 범주형 변수의 각 레벨 그룹에서 목표변수의 대표값을 학습데이터로 집계하고 그 값으로 치환함.
          - 회귀 : 목표변수의 평균, 중앙값 등
          - 이진분류 : 양성의 출현 비율
          - 다중분류 : 클래스 수 만큼의 이진분류가 있다고 가정하고, 클래스 수 만큼의 타깃 인코딩 특성을 만듬.
        - 목표변수의 데이터 정보 누출의 우려가 있음에 주의해야 함.

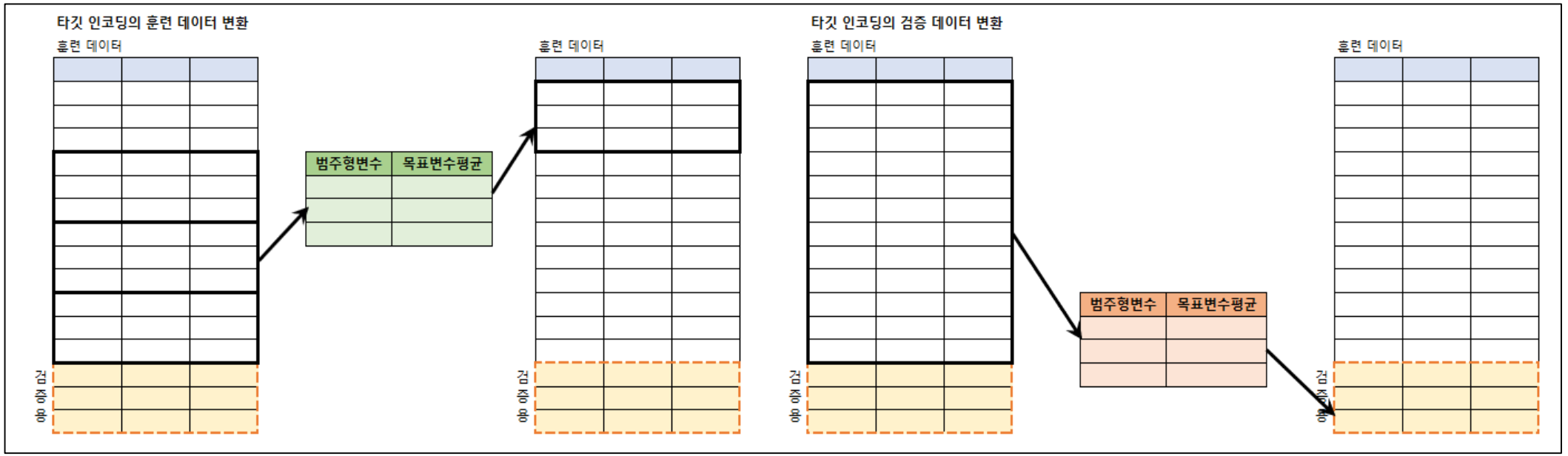
특성 공학

- 특성 공학
  - 특성변수의 변환
    - 범주형 변수 변환
      - 타깃 인코딩
        - 타깃 인코딩 기법
          - 훈련 데이터는 타깃 인코딩용 폴드로 분할한 뒤, 아웃-오브-폴드 방식으로 목표변수의 평균을 계산하여 적용함.
          - 평가 데이터는 훈련 데이터 전체의 목표변수의 평균을 계산하여 적용함.



특성 공학

- 특성 공학
  - 특성변수의 변환
    - 범주형 변수 변환
      - 타깃 인코딩
        - 교차검증 시행 시
          - 교차검증을 실시하는 경우 검증 데이터를 제외한 학습 데이터를 타깃 인코딩 용 폴드로 분할. 이 작업을 교차검증 폴드 수 만큼 반복
          - 검증 데이터는 훈련 데이터 전체의 목표변수의 평균을 계산하여 적용함.

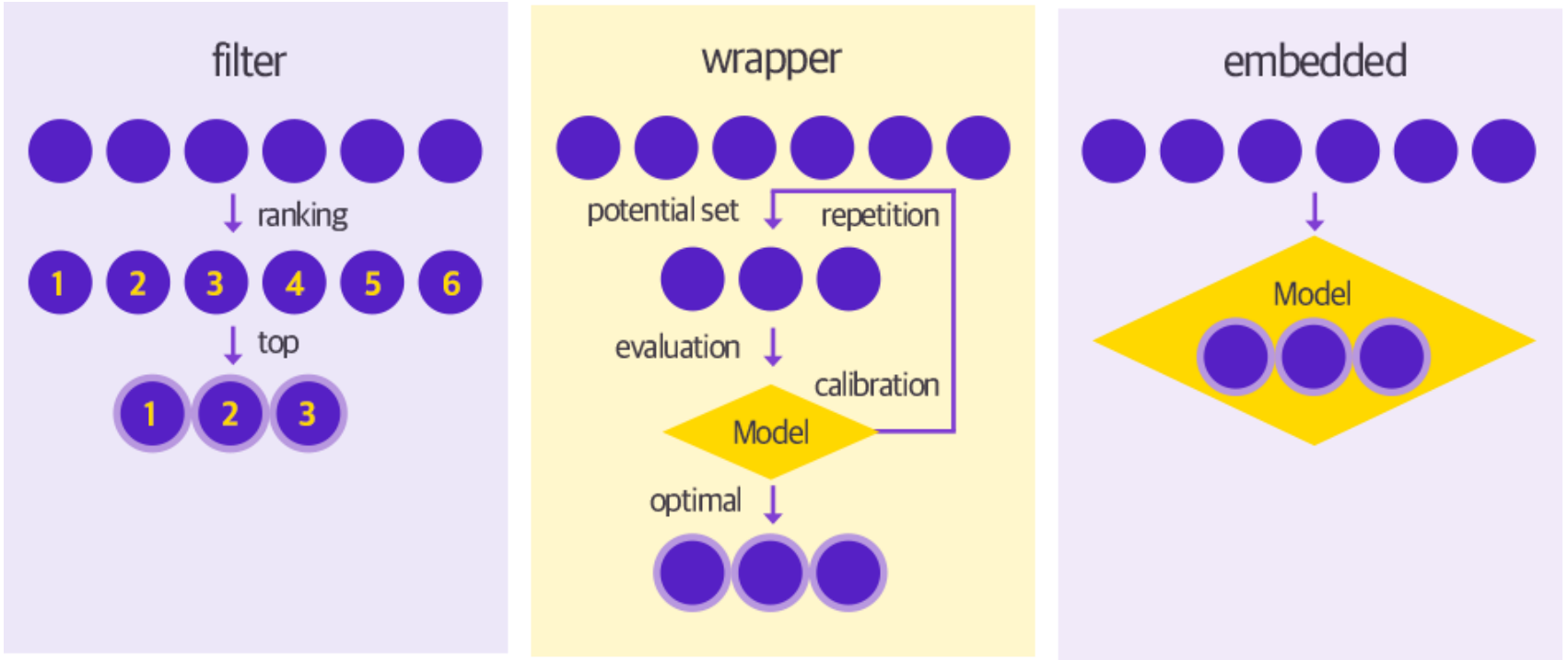


특성 공학

- 특성 공학
  - 차원 축소
    - 차원 축소의 필요성
      - 모델의 해석력 향상
      - 모델 훈련시간의 단축
      - 차원의 저주 방지
      - 과적합(overfitting)에 의한 일반화 오차를 줄여 성능 향상
    - 특성 선택 VS 특성 추출
      - 특성 선택 (feature selection)
        - 훈련자료에 주어진 특성변수들 가운데 불필요한 특성변수는 제거하고, 예측 성능이 좋은 특성변수의 조합을 찾는 과정.
        - filtering, wrapper, embedded 방식이 있음.
      - 특성 추출 (feature extraction)
        - 훈련 자료에 포함된 특성변수들을 결합해 더 유용한 특성변수들을 생성하는 과정.
        - PCA, SVD 등 차원축소법이 주로 활용됨.

특성 공학

- 특성 공학
  - 특성 선택



특성 공학

- 특성 공학
  - 특성 선택
    - Filter 방식 : 각 특성변수를 독립적인 평가함수로 평가함.
      - 각 특성변수 X_i와 목표변수(Y)와의 연관성을 측정한 뒤, 목표변수를 잘 설명할 수 있는 특성 변수만을 선택하는 방식.
      - X_i 와 Y의 1:1 관계로만 연관성을 판단.
      - 연관성 파악을 위해 t-test, chi-square test, regression F test등의 지표가 활용됨.

특성 공학

- 특성 공학
  - 특성 선택
    - Wrapper 방식 : 학습 알고리즘을 이용.
      - 다양한 특성변수의 조합에 대해 목표변수를 예측하기 위한 알고리즘을 훈련하고, cross-validation 등의 방법으로 훈련된 모델의 예측력을 평가함. 그 결과를 비교하여 최적화된 특성변수의 조합을 찾는 방법.
      - 특성변수의 조합이 바뀔 때 마다 모델을 학습함.
      - 특성변수에 중복된 정보가 많은 경우 이를 효과적으로 제거함.
      - 대표적인 방법으로는 순차탐색법인 forward selection, backward selection, stepwise selection 등이 있음.
      - 캐글 등에서 인기있는 알고리즘인 Boruta의 경우 랜덤 포레스트의 wrapper 방식에 해당함.

특성 공학

- 특성 공학
  - 특성 선택
    - Filter 와 Wrapper의 장단점 비교

	장점	단점
Filter	계산비용이 적고, 속도가 빠름.	특성변수 간의 상호작용을 고려하지 못함.
Wrapper	특성변수 간의 상호작용을 고려함. 주어진 학습 알고리즘에 대해 항상 최적의 특성변수 조합을 찾음.	모델을 학습해야 하므로, 계산비용이 크고, 속도가 느림. 과적합(overfitting)의 가능성이 있음.

특성 공학

- 특성 공학
  - 특성 선택
    - Boruta 알고리즘
      - 개요
        - 원 데이터의 특성변수들과 그 permutation 버전인 shadow 변수들을 이용하여 랜덤포레스트 모델을 학습함. 중요한 특성변수라면, shadow 변수보다 특성중요도가 높을 것이라는 아이디어에 기반함.
      - 알고리즘
        1. 전체 반복 수 m을 정하고, 아래 ① ~ ④의 과정을 m회 반복한다.
           ① 원 특성변수에 대한 shadow 특성변수를 만든다.
           ② 랜덤포레스트 모델을 학습하고, 전체 변수(원변수 + shadow변수)의 특성중요도의 평균과 표준편차를 구한다. 이를 이용하여 각 특성 별로 z-score (= 특성중요도의 평균 / 표준편차)를 구한다.
           ③ shadow 변수들의 z-score중 최대값 $z_{max}^{(m)}$을 구한다.
           ④ 원 변수 중 $z_{max}^{(m)}$보다 큰 z-score를 가지는 변수에 hit 표시를 한다.

특성 공학

- 특성 공학
  - 특성 선택
    - Boruta 알고리즘
      - 알고리즘 (계속)

2. ① ~ ④의 과정을 m회 반복한 뒤, 각 특성변수 별 hit의 총합 (X_i)을 구하고, 그 결과에 다음의 통계적 가설검정을 수행한다.
   - $H_0: p_i = 0.5 \quad VS \quad H_1: p_i \neq 0.5$
   - 귀무가설이 사실일 때, X_i는 $Binomial[m, 0.5]$ 분포를 가짐.
   - 유의수준 α로 검정
     - X_i 가 오른꼬리확률 $\alpha/2$를 가지는 임계치보다 크면 해당 특성변수 X_i는 confirm.
     - X_i 가 왼꼬리확률 $\alpha/2$를 가지는 임계치보다 작으면 해당 특성변수 X_i는 reject
     - 위 두 경우가 아닌 경우는 tentative 라벨을 달아줌.
3. 알고리즘 종료. confirm 라벨이 달린 변수들만 최종적으로 선택함. 보수적으로 선택한다면 tentative 라벨이 달린 변수들까지 포함할 수 있음.

특성 공학

- 특성 공학
  - 특성 선택
    - Boruta 알고리즘
      - 특징
        - 분류와 회귀 문제에 모두 적용 가능함.
        - 특성 변수간의 다중공선성에 덜 민감한 편임.
        - 목표 변수와 복잡한 관계가 있는 특성 변수들에 대해서도 잘 작동되는 편임.

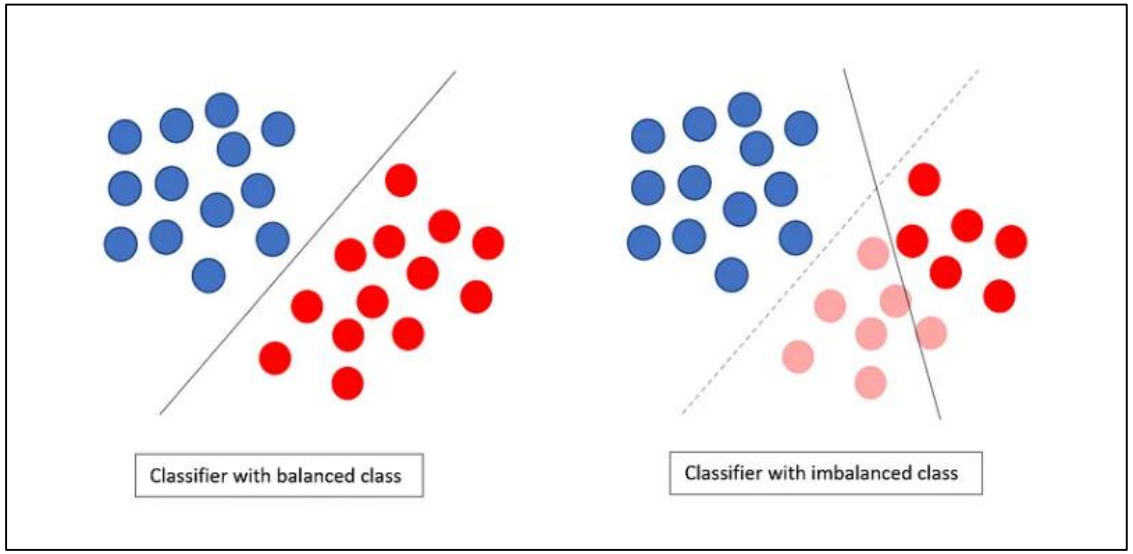
      - 단점
        - 계산시간이 많이 소요됨.
        - 하이퍼파라미터의 튜닝 결과에 민감함.
        - 데이터의 품질에 영향을 많이 받음.

특성 공학

- 특성 공학
  - 특성 선택
    - Embedded 방식 : 학습 알고리즘 자체에 feature selection을 포함하는 경우
      - Wrapper 방식은 모든 특성변수 조합에 대한 학습을 마친 결과를 비교하는데 비해, Embedded 방식은 학습 과정에서 최적화된 변수를 선택한다는 점에서 차이가 있음.
      - 대표적인 방법으로는 특성변수에 규제를 가하는 방식인 Lasso를 활용하는 방식이나, 트리 모델의 특성 중요도를 활용하는 방식이 있음.

클래스 불균형의 처리

- 데이터 불균형 문제
  - 클래스 불균형 문제(class imbalance problem)
    - 학습 데이터의 클래스 변수가 균일하게 분포하지 않고 하나의 값에 치우쳐서 편향된 모델을 학습하는 문제



    - 대부분의 입력을 다수(majority) 클래스로 분류하게 되어, 소수(minor) 클래스에 대한 예측 성능이 저하되는 경우가 많은데, 일반적으로 소수(minor) 클래스가 분석자의 관심범주라 문제가 됨.

클래스 불균형의 처리

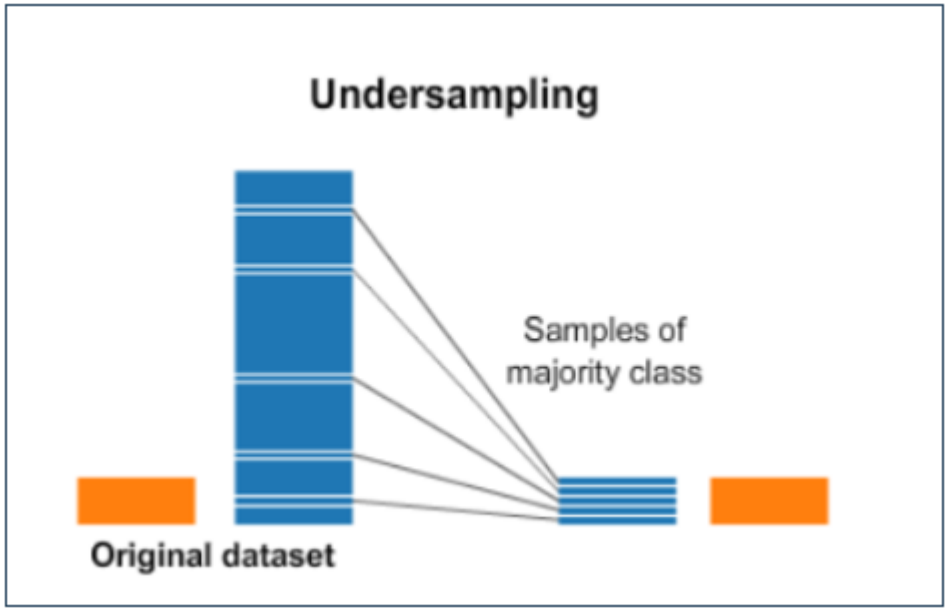
- 데이터 불균형 문제
  - 데이터 샘플링을 이용한 불균형 데이터 문제의 해결
    - Under Sampling
      - Random Under Sampling
      - CNN (Condensed Nearest Neighbor)
      - Tomek's Link
      - OSS (One-Sided Selection) → CNN + Tomek's Link
    - Over Sampling
      - Random Over Sampling
      - SMOTE (Synthetic Minority Over-Sampling Technique)
      - ADASYN (Adaptive Synthetic Sampling)

클래스 불균형의 처리

- 데이터 불균형 문제
  - Under Sampling

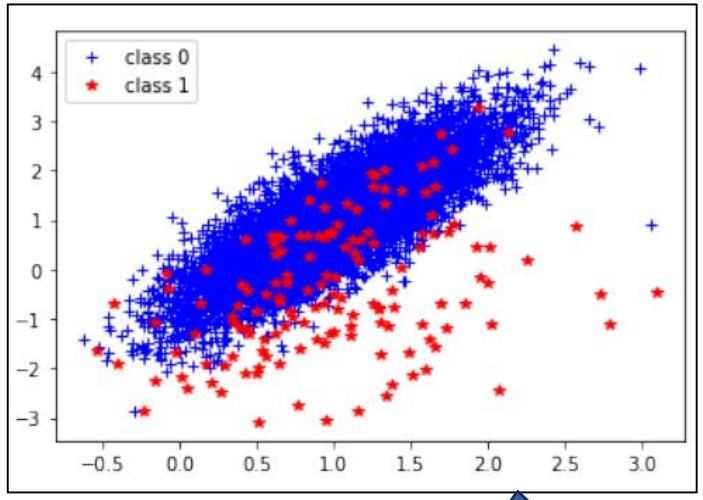
장점	단점
훈련 데이터의 수가 감소하여 계산 시간이 줄어든다.	제거된 데이터에 대한 정보 손실이 발생한다.

    - Random Under Sampling
      - major 클래스에서 무작위로 선택하여 minor 클래스의 샘플 개수와 균형을 맞춤.(예. 2:1, 1:1 등)

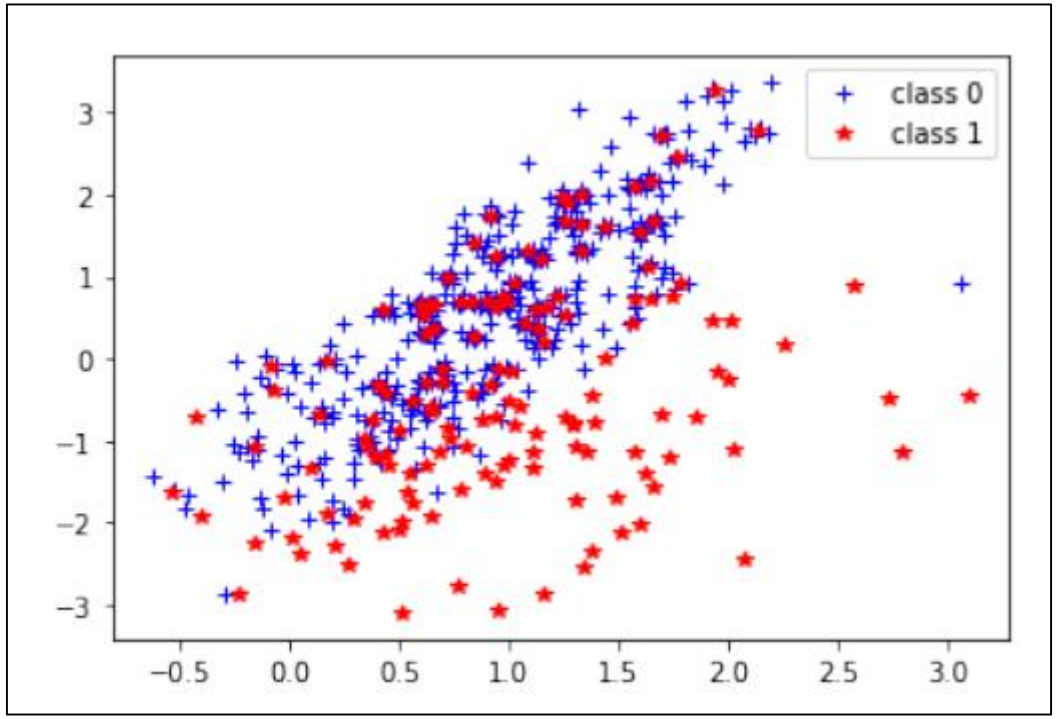


클래스 불균형의 처리

- 데이터 불균형 문제
  - Under Sampling
    - CNN (Condensed Nearest Neighbor)
      - minor 클래스는 모두 표본집합 S에 포함됨. 또 major 클래스 데이터 중 하나를 골라 그와 가장 가까운 데이터(1-Nearest Neighbor)가 minor 클래스에 속하는 경우에만 S에 포함시킴. 이 과정을 S에 더 이상 추가되는 자료가 없을 때까지 반복

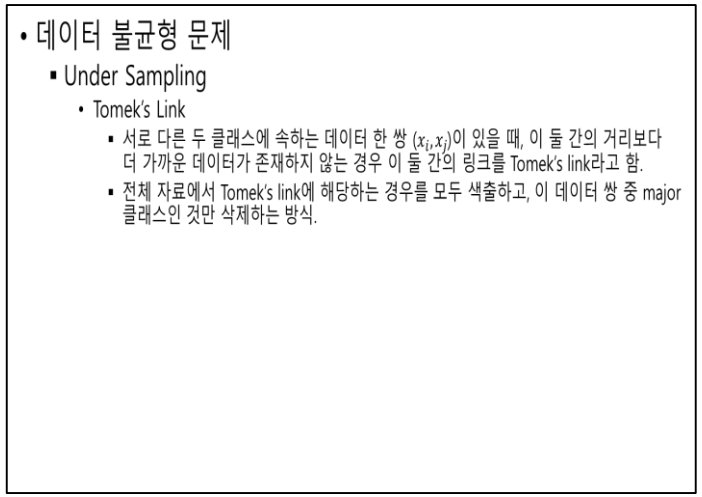


CNN

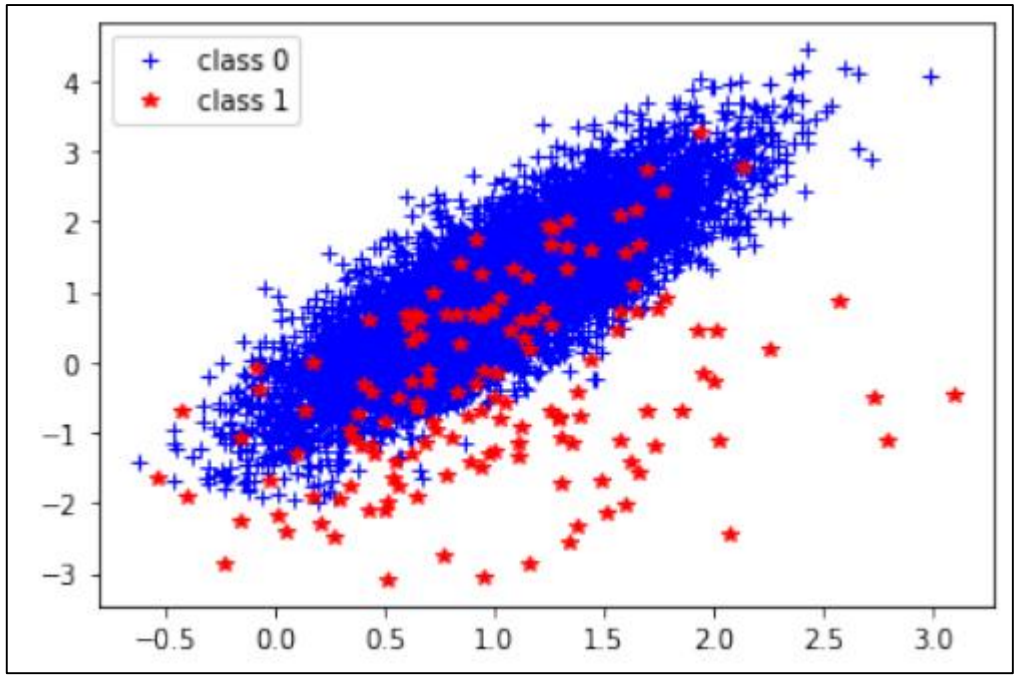


클래스 불균형의 처리

- 데이터 불균형 문제
  - Under Sampling
    - Tomek's Link
      - 서로 다른 두 클래스에 속하는 데이터 한 쌍 (x_i, x_j)이 있을 때, 이 둘 간의 거리보다 더 가까운 데이터가 존재하지 않는 경우 이 둘 간의 링크를 Tomek's link라고 함.
      - 전체 자료에서 Tomek's link에 해당하는 경우를 모두 색출하고, 이 데이터 쌍 중 major 클래스인 것만 삭제하는 방식.



Tomek's Link

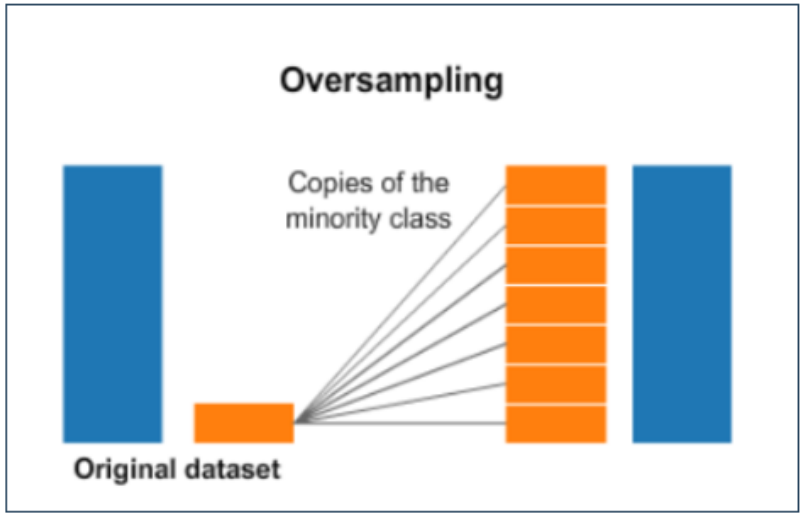


클래스 불균형의 처리

- 데이터 불균형 문제
  - Over Sampling

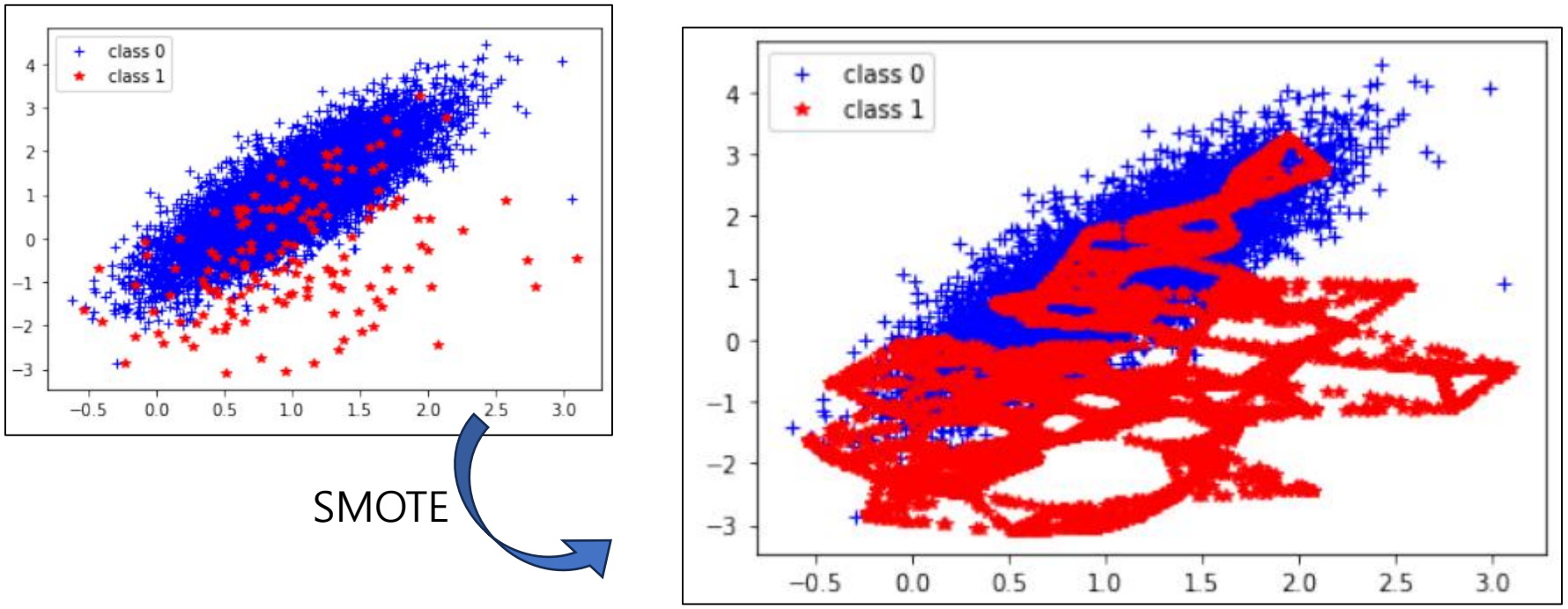
장점	단점
훈련 데이터에 대한 정보 손실이 발생하지 않는다. Under Sampling을 적용한 경우보다 분류 정확도가 높은 편이다.	데이터 증가로 인해 계산시간이 증가한다. 이상치나 노이즈의 영향이 커진다. 과적합 문제가 발생한다.

    - Random Over Sampling
      - minor 클래스에서 무작위 복원 추출하여 major 클래스의 샘플 개수와 균형을 맞춤.
        (예. 2:1, 1:1 등)



클래스 불균형의 처리

- 데이터 불균형 문제
  - Over Sampling
    - SMOTE (Synthetic Minority Over-Sampling Technique)
      - 어느 minor 클래스 데이터에 대해 가장 가까운 k개의 minor neighborhood를 식별하고 그 중 하나를 랜덤하게 선택함. 선택된 두 minor 클래스 자료 벡터에 uniform 난수를 곱하여 가상의 minor 데이터를 생성함.



클래스 불균형의 처리

- 데이터 불균형 문제
  - Over Sampling
    - ADASYN (Adaptive Synthetic Sampling)
      - SMOTE의 개선된 버전.
      - 각 minor 클래스 자료 주변에 SMOTE 방식으로 생성되는 가상의 minor 데이터의 수는 주변에 major 클래스의 비중에 따라 달라지도록 한 것.