

6G 통신 및 신경망 검증 ML 시스템 1주차

이도영



목차

- ML, 연구용 ML, 프로덕션용 ML 이란
- 연구용, 프로덕션용 ML 차이
- 반복 프로세스
- 머신러닝 문제 구조화하기
- 손실함수

ML, 연구용 ML, 프로덕션용 ML

- ML: 데이터로부터 규칙(모델)을 자동으로 학습하여,
보지 못한 입력에 대해 예측이나 판단을 수행하는 기술
- 연구용 ML: 데이터를 통해 새로운 모델이나 방법이 효과적인지
검증하는데 초점을 둔 머신러닝
- 프로덕션용 ML: 실제 서비스 환경에서 발생하는 데이터를 바탕으로
예측을 안정적으로 제공하는 머신러닝

연구용, 프로덕션용 ML 차이

	연구용 ML	프로덕션용 ML
요구사항	벤치마크 데이터셋에서 최적의 모델 성능 달성하기	이해관계자마다 다름
계산 우선순위	빠른 훈련, 높은 Throughput	빠른 추론, 낮은 Latency
데이터	정적	동적
공정성	중요하지 않은 경우가 많음	반드시 고려해야 함
해석 가능성	중요하지 않은 경우가 많음	반드시 고려해야 함

연구용, 프로덕션용 ML 차이 - 요구사항

	연구용 ML	프로덕션용 ML
요구사항	벤치마크 데이터셋에서 최적의 모델 성능 달성하기	이해관계자마다 다름

□ 연구용: 연구의 목표는 모델 성능

벤치마크 데이터셋에서 결과를 달성하는 모델 개발

□ 프로덕션용: 많은 관계자들의 각각 요구사항이 다를 수 있음(서로 충돌 가능성)

모든 요구사항을 충족하는 모델 설계, 개발, 선택 어려움 존재

따라서, 연구에서 성공적인 ML이더라도 프로덕션용 ML로 사용되지 않을 수 있음



연구용, 프로덕션용 ML 차이 – 계산 우선순위

	연구용 ML	프로덕션용 ML
계산 우선순위	빠른 훈련, 높은 Throughput	빠른 추론, 낮은 Latency

□ 일반적으로 | 검증 데이터 | < | 훈련 데이터 | 를 따름

=> 모델 개발 단계 병목사항: 훈련

=> 모델 개발 단계 이후 병목사항: 예측

연구용, 프로덕션용 ML 차이 – 계산 우선순위(cont')

	연구용 ML	프로덕션용 ML
계산 우선순위	빠른 훈련, 높은 Throughput	빠른 추론, 낮은 Latency

□ 연구용: 특정 기간 내 처리된 쿼리 수

Ex) 1초에 처리할 수 있는 샘플 수 집중

=> Throughput을 늘리기 위해 Latency를 늘리는 경우도 존재

연구용, 프로덕션용 ML 차이 – 계산 우선순위(cont')

	연구용 ML	프로덕션용 ML
계산 우선순위	빠른 훈련, 높은 Throughput	빠른 추론, 낮은 Latency

□ 프로덕션용: 쿼리를 수신하고 결과를 반환하는데 걸리는 시간

Ex) 페이지 로드가 3초 이상 걸리는 경우 절반 이상 사용자가 페이지를 나감

=> Latency를 줄이기 위해 Query개수를 줄임

연구용, 프로덕션용 ML 차이 – 계산 우선순위(cont')

	연구용 ML	프로덕션용 ML
계산 우선순위	빠른 훈련, 높은 Throughput	빠른 추론, 낮은 Latency

□ 최신 분산 시스템 양상: 대부분의 쿼리 배치처리

=> Latency와 Throughput은 비례

□ Latency 수치: 백분위수 사용

=> 중앙값, 이상치 발견 편리

연구용, 프로덕션용 ML 차이 - 데이터

	연구용 ML	프로덕션용 ML
데이터	정적	동적

❑ 연구용: 데이터 형식이 잘 지정되어 있음

새로운 아키텍처, 기술 벤치마킹에 사용 가능

데이터 셋의 장단점 알기 쉬움, 모델 오픈소스 등 배포되어있음

❑ 프로덕션용: 잡음이 많고 끊임없이 변동하는 데이터

레이블이 있는 경우 레이블이 희소하고 불균형하거나 올바르지 않음

[레이블: Ground Truth(정답 여부)]



연구용, 프로덕션용 ML 차이 - 공정성

	연구용 ML	프로덕션용 ML
공정성	중요하지 않은 경우가 많음	반드시 고려해야 함

❑ 연구용: 모델이 사용되지 않기 때문에 공정성을 나중으로 미루는 경우가 많음

❑ 프로덕션용: 과거 시점 데이터를 분석하고 찾아낸 패턴을 기반으로

데이터의 추론 진행

=> Bias 가능성

연구용, 프로덕션용 ML 차이 – 해석 가능성

	연구용 ML	프로덕션용 ML
해석 가능성	중요하지 않은 경우가 많음	반드시 고려해야 함

□ 연구용: 모델 성능을 높이는 단일 목표

=> 중요하지 않은 경우가 대부분

□ 프로덕션용: 모델 신뢰 및 잠재 편향을 감지할 수 있도록 결정이 내려진 이유 이해,
개발자가 모델을 디버깅하고 개선할 수 있어야함

반복 프로세스

□ ML이 한단계씩 진행될 때 앞단계의 가정을 깨뜨리기 때문에 반복구조를 가짐

1단계(프로젝트 범위

2단계(데이터 엔지니어링

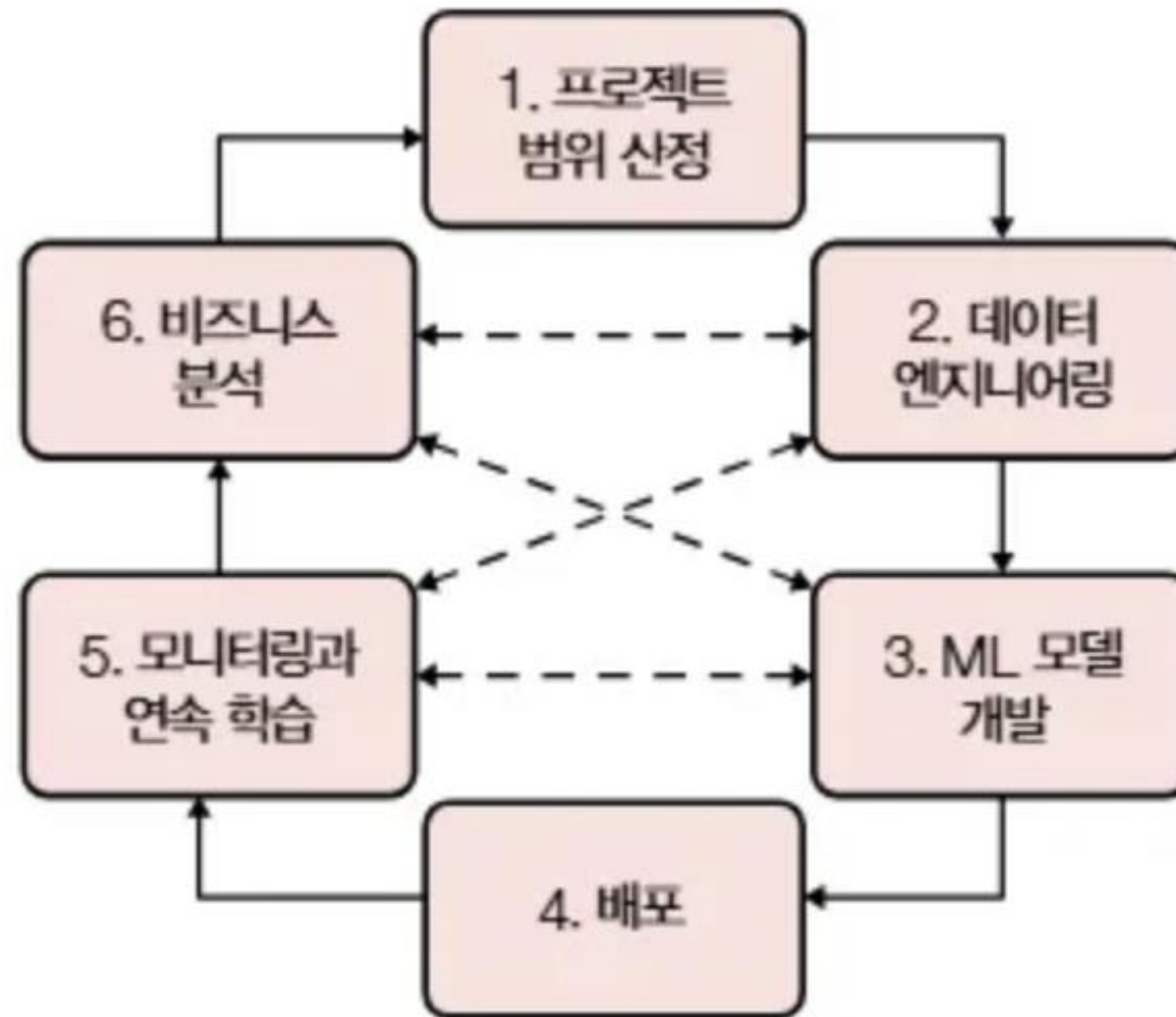
3단계(ML 모델 개발

4단계(배포): 개발한

추정 후 할당

식

초기 모델 개발



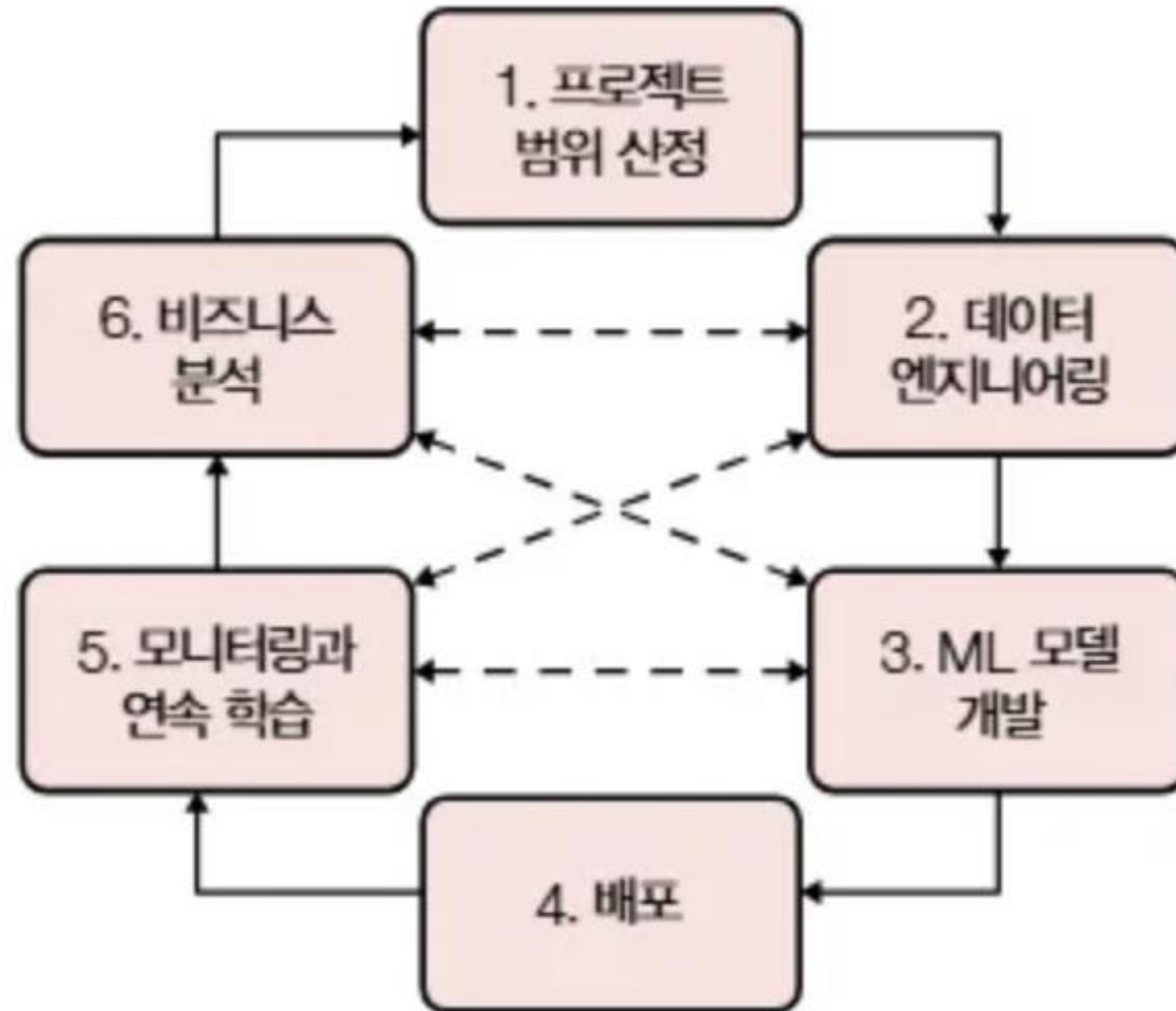
반복 프로세스 (cont')

□ ML이 한단계씩 진행될 때 앞단계의 가정을 깨뜨리기 때문에 반복구조를 가짐

5단계(모니터링과 연속 학습): 지속적으로 성능 저하를 모니터링하고 변화하는
환경, 요구사항에 적응하도록 유지 관리

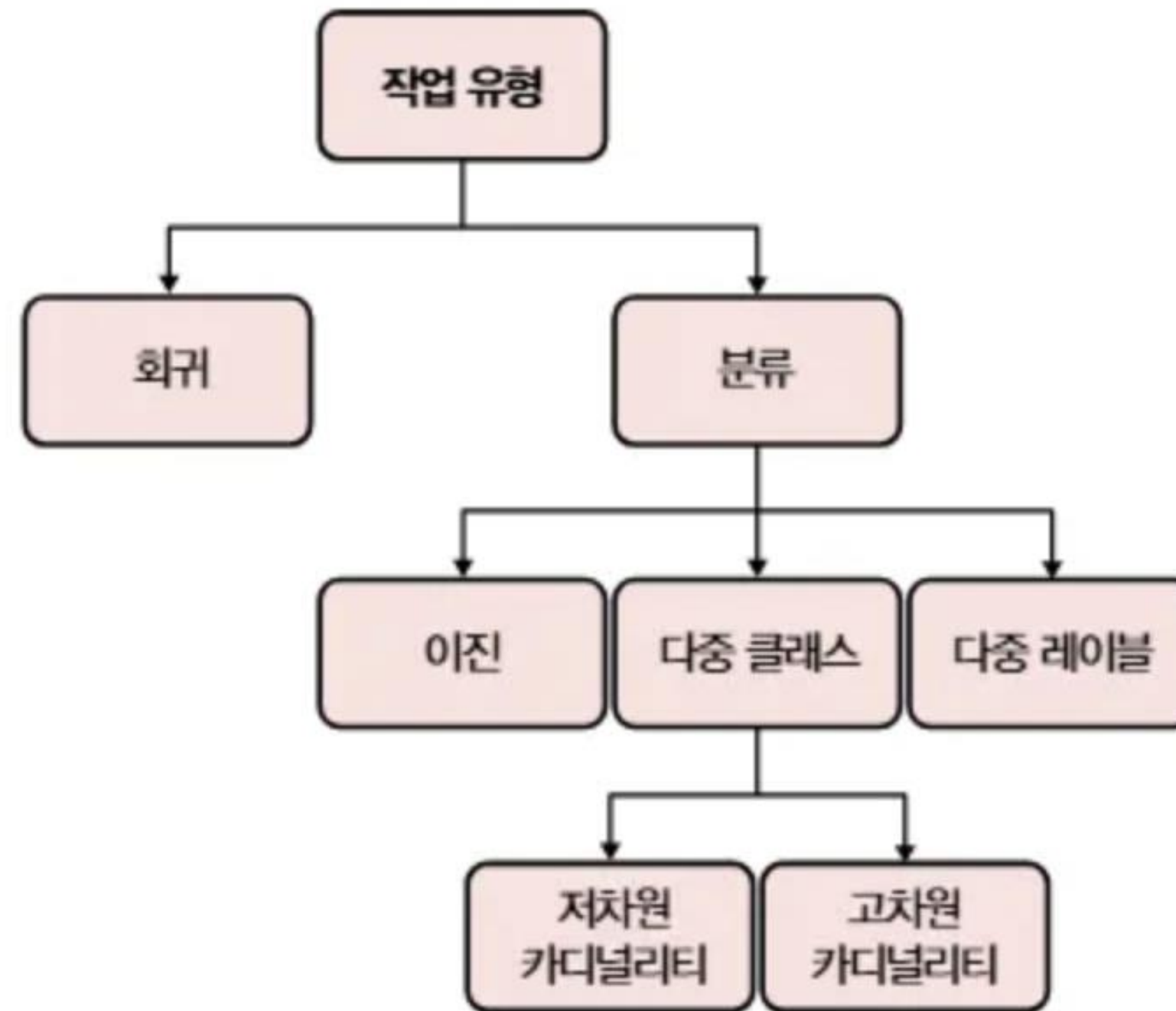
6단계(비즈니스 분석): 모델 성능을 비즈니스 목표 관점에서 평가하고 분석해
비즈니스 인사이트 추출

반복 프로세스 (cont')



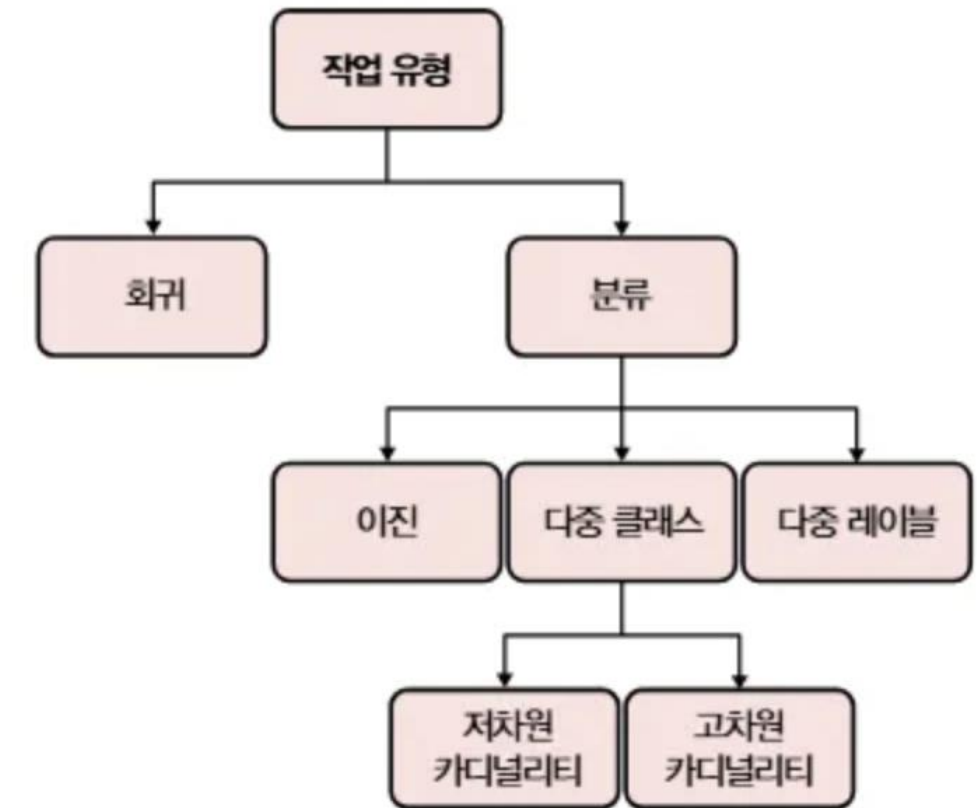
머신러닝 문제 구조화하기

□ 일반적인 ML 작업 유형 -> 분류, 회귀



머신러닝 문제 구조화하기 (cont')

- 분류: 입력을 여러 범주로 분류(~인것, ~아닌것)
- 회귀: 연속 값 출력
- 이진 분류: 분류 가능한 클래스 두 개
- 다중 클래스 분류: 분류 가능한 클래스가 세 개 이상
- 다중 레이블 분류: 데이터가 여러 클래스에 동시 속함

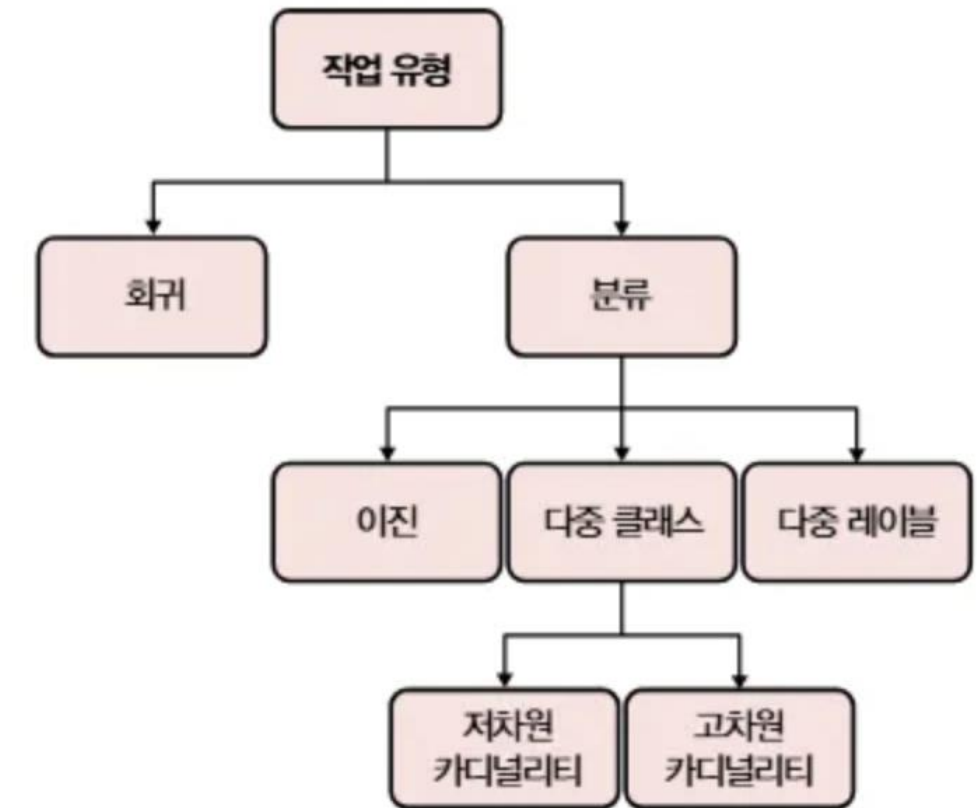


머신러닝 문제 구조화하기 – 다중 클래스 분류(cont')

□ 다중 클래스 분류: 분류 가능한 클래스가 세 개 이상

□ 클래스가 많은 경우: 고차원 카디널리티

Ex) 질병 진단 문제의 질병 수



이런 경우 계층적 분류 유용

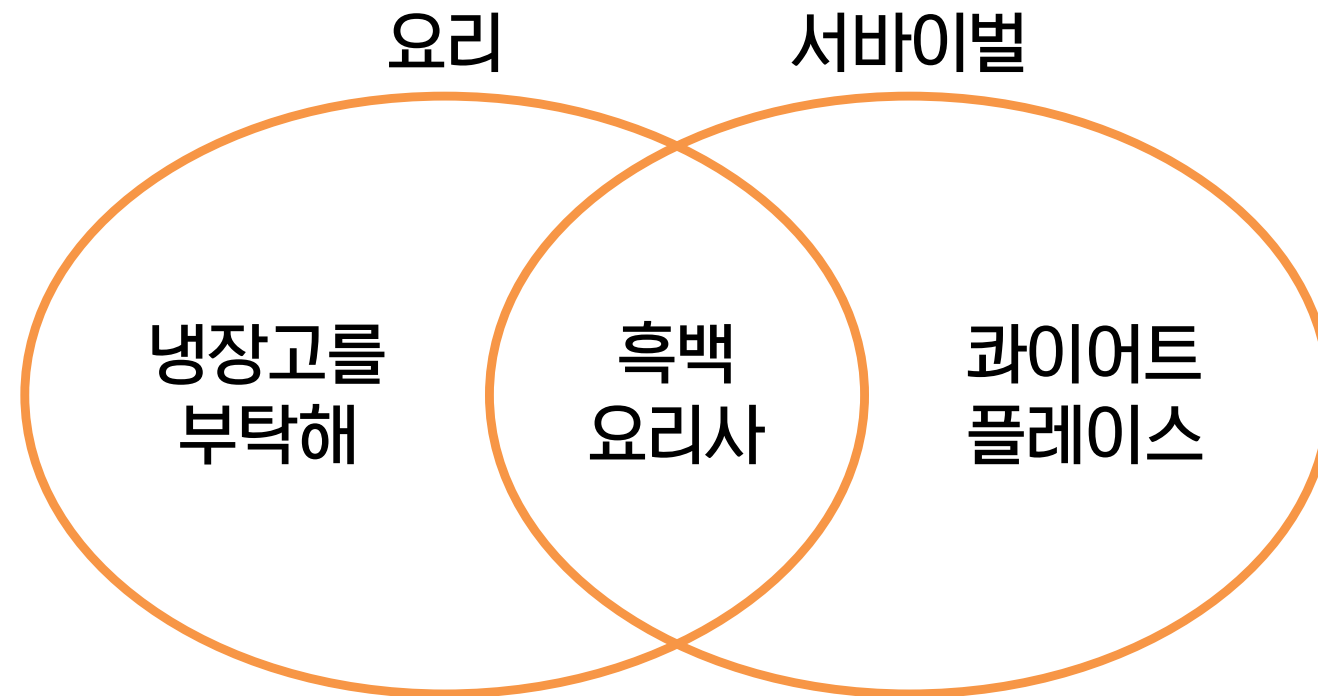
Ex)



머신러닝 문제 구조화하기 – 다중 레이블 분류(cont')

□ 다중 클래스 분류: 데이터가 여러 클래스에 동시 속함

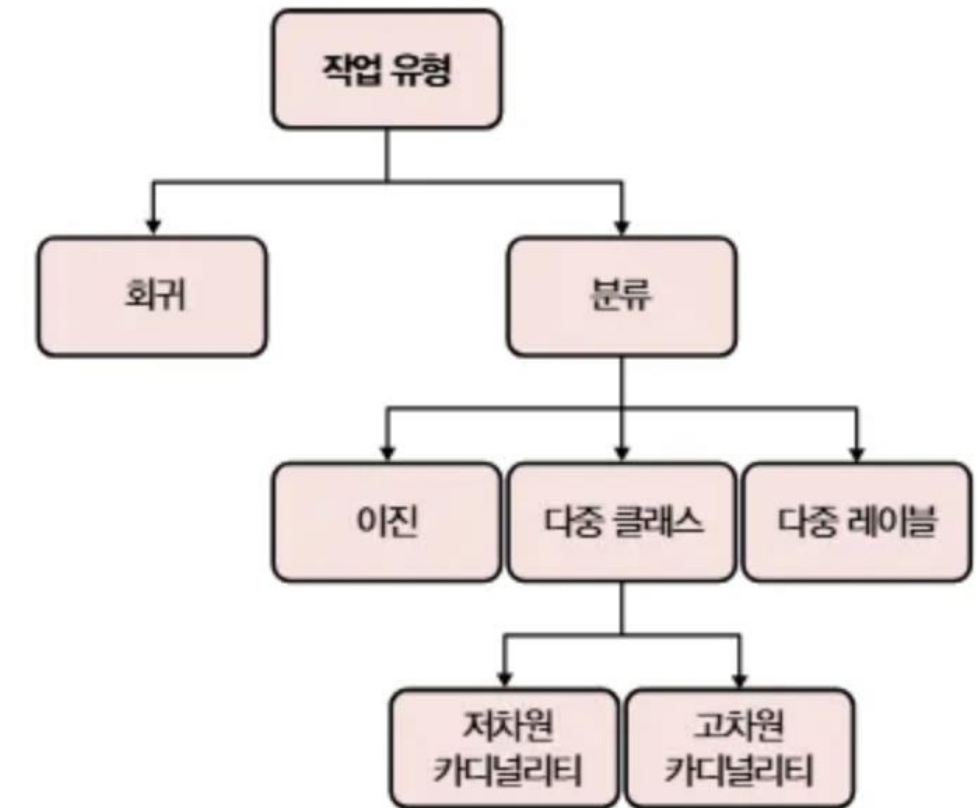
Ex)



방법1)

만약 클래스가 [요리, 미스터리, 서바이벌, 액션] 이라면

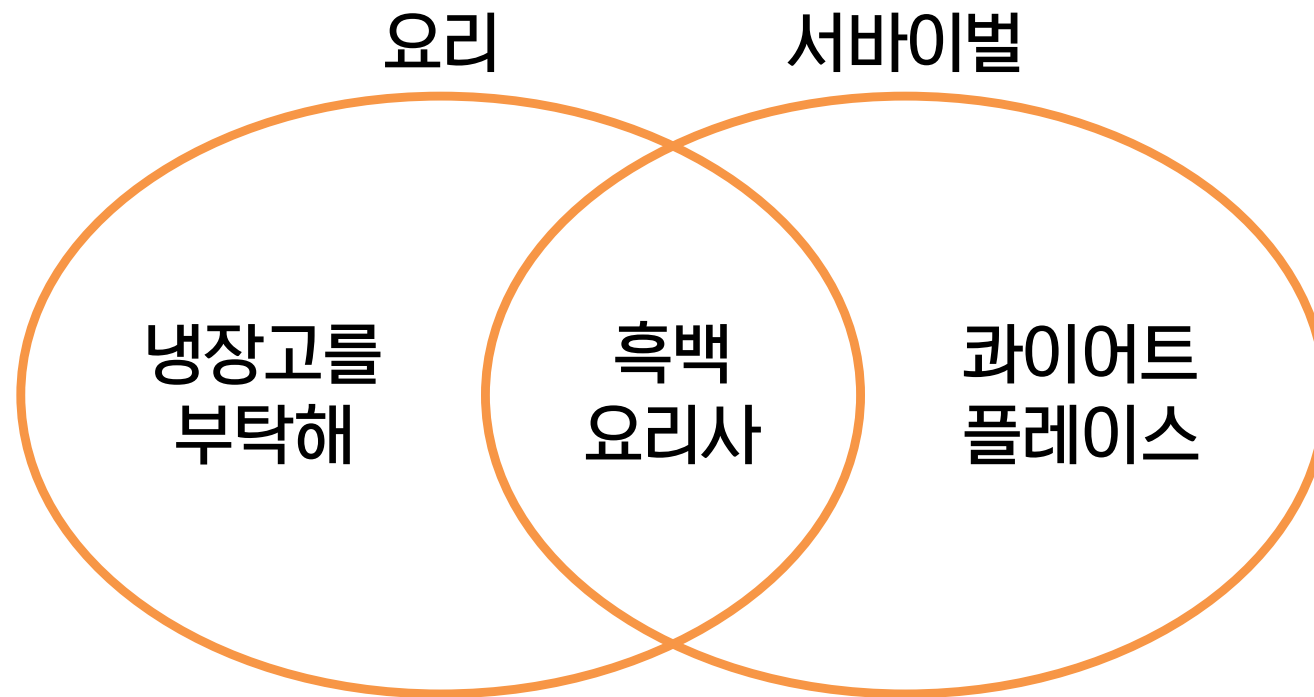
흑백 요리사는 [1, 0, 1, 0] 으로 표현



머신러닝 문제 구조화하기 – 다중 레이블 분류 (cont')

□ 다중 클래스 분류: 데이터가 여러 클래스에 동시 속함

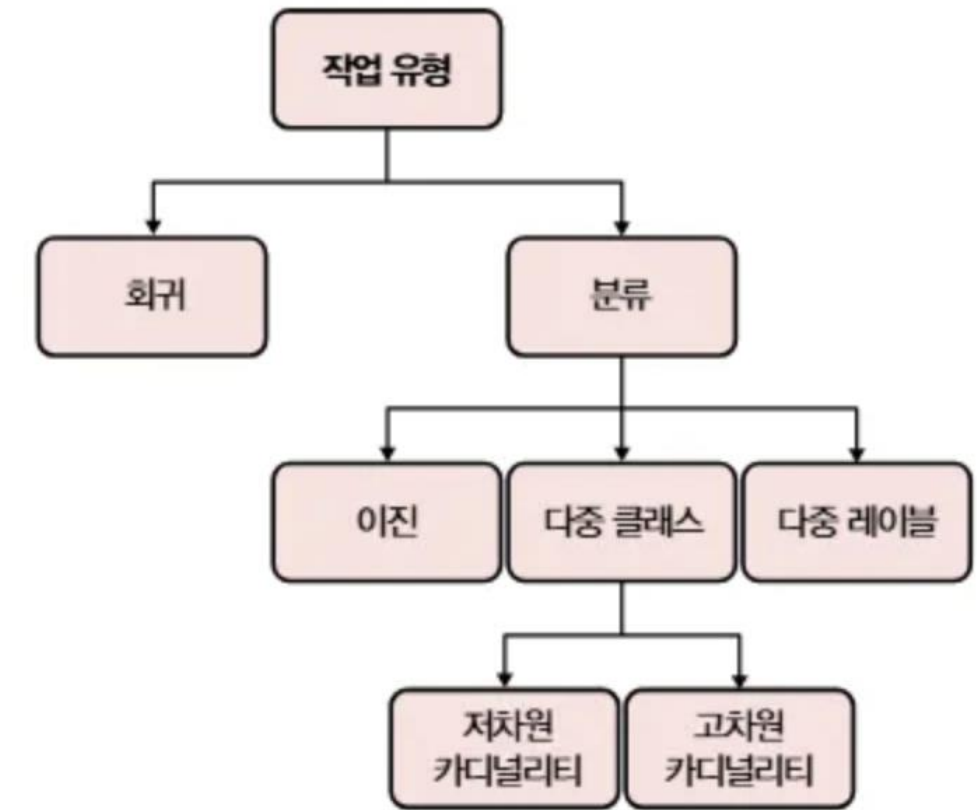
Ex)



방법2)

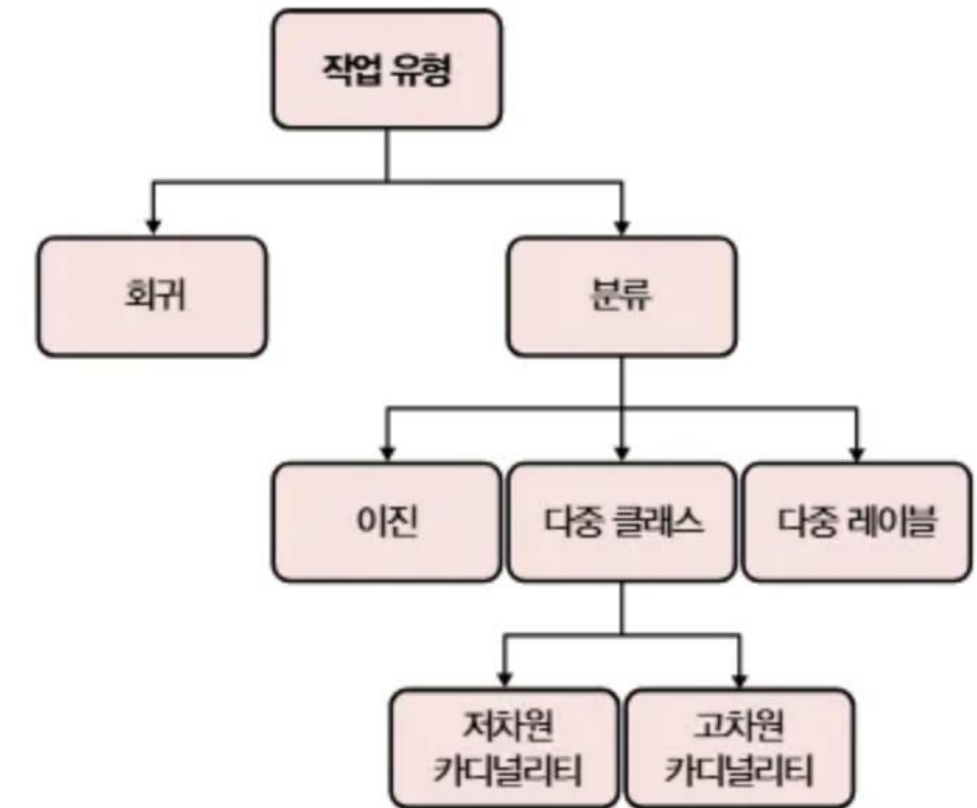
만약 클래스가 [요리, 미스터리, 서바이벌, 액션] 이라면

각각의 주제에 대해 모델을 만든 후 주제에 속하는지 여부 출력



머신러닝 문제 구조화하기 – 다중 분류 (cont')

- 다중 클래스 분류 => 확률이 가장 높은 범주 선택
- 다중 레이블 분류 => 몇 가지 범주에 속하는지 모름
가장 높은 범주 2개 혹은 3개 선택



- 구조화를 분류하는 것과 회귀하는 것에 따라 모델 개발에 어려움 존재
입력데이터의 종류, 프로세스 형태를 통해 분류, 회귀 중 결정

손실함수

- ML에서 손실을 계산할 때 모델 출력, Ground Truth 레이블 비교 후 RMSE(Root Mean Square Error) 혹은 Cross Entropy 사용

- $$\text{RMSE} = \sqrt{\sum_{i=1}^n \frac{(\hat{y}_i - y_i)^2}{n}}$$

- $$\text{Cross Entropy} = H(\mathbf{x}) = -\sum_{i=1}^n P(x_i) \log P(x_i)$$

손실함수

□ 회귀: 평균 제곱근 오차(RMSE)

평균 절대 오차(MAE, Mean Absolute Error)

□ 이진분류: 로지스틱 손실

□ 다중 클래스 분류: Cross Entropy

□ 목적 함수가 여러개인 경우 ML 문제 구조화가 어려움

손실함수

Ex) 목적이 2가지일 때,

접근법1: 예측 품질 \leftrightarrow 실제 품질 처리 = quality loss

예측 클릭 수 \leftrightarrow 실제 클릭 수 = engagement loss

두 가지를 결합한 $loss = \alpha \text{ quality loss} + \beta \text{ engagement loss}$

를 최소화하는 방법을 사용함

α 와 β 값을 무작위로 테스트해보며 가장 잘 동작하는 값을 찾음

→ 조정할 때 마다 모델 다시 훈련

손실함수

Ex) 목적이 2가지일 때,

접근법2: 예측 품질 \leftrightarrow 실제 품질 처리 = quality loss

예측 클릭 수 \leftrightarrow 실제 클릭 수 = engagement loss

모델을 2개 만든 후 각 모델의 출력을 결합함

감사합니다.

