밑바닥부터 시작하는 데이터 과학

5 회차

- 의사결정나무
- 모형 결합
- 불균형 데이터 문제
- 군집화
- 시스템 설계

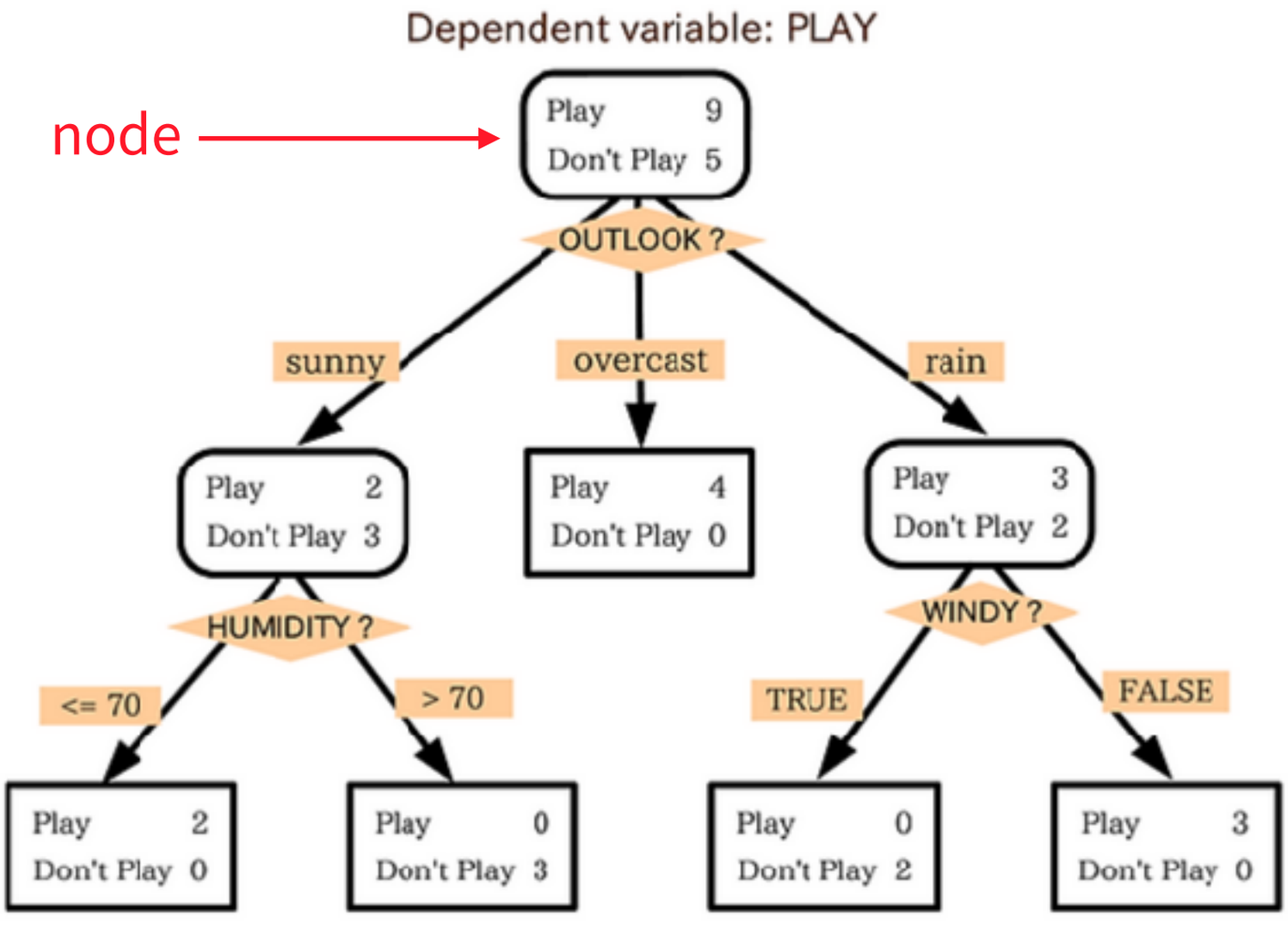
1. 의사결정나무

의사결정나무 (Decision Tree)

1. 의사결정나무

의사결정나무 (Decision Tree)

- 여러 규칙을 순차적으로 적용하면서 독립 변수의 공간을 분할하는 모델.
- 다양한 의사결정 경로(decision path)과 결과(outcome)를 나타내는데 나무 구조를 사용함.
  - 질문을 던져서 대상을 좁혀나가는 '스무고개' 놀이와 비슷한 개념



1. 의사결정나무

의사결정나무 (Decision Tree)

분류 방법

1. 여러가지 독립 변수 중 하나의 독립 변수를 선택하고 그 독립 변수에 대한 기준값(threshold)을 정한다. 이를 **분류 규칙**이라고 한다.
2. 전체 학습 데이터 집합(부모 노드)을 해당 독립 변수의 값이 기준값보다 큰 데이터 그룹(자식 노드 1)과 해당 독립 변수의 값이 기준값보다 작은 데이터 그룹(자식 노드 2)으로 나눈다.
3. 자식 노드에 있는 데이터의 클래스의 비율을 조사하여 가장 데이터 수가 많은 클래스를 그 자식 노드의 대표 클래스로 정한다.
4. 각각의 자식 노드에 대해 1~3의 단계를 반복하여 하위의 자식 노드를 만든다. 단, 자식 노드에 한가지 클래스의 데이터만 존재한다면 더 이상 자식 노드를 나누지 않고 중지한다.

규칙 결정 방법

- 불확실성을 최대한 낮추는 규칙을 정한다.

  = 정보획득 (Information Gain)이 가장 커지는 기준값.

  = Information Gain : 분할 전 엔트로피 (entropy) - 분할 후 엔트로피 (entropy)

  = 엔트로피를 가장 낮추는 규칙.

1. 의사결정나무

의사결정나무 (Decision Tree)

엔트로피 (Entropy)

- 상태가 분산되어 있는 정도 (= 얼마만큼의 정보를 담고 있느냐)
- 여러가지로 고루 분산되어 있을 수 있으면 엔트로피가 높고, 특정한 하나의 상태로 몰려있으면 엔트로피가 낮다.

확률분포 $Y_1: P(Y=0)=0.5, P(Y=1)=0.5$

확률분포 $Y_2: P(Y=0)=0.8, P(Y=1)=0.2$

확률분포 $Y_3: P(Y=0)=1.0, P(Y=1)=0.0$

엔트로피의 수학적 정의

- **Y가 이산확률 변수인 경우 :** $H[Y] = -\sum_{k=1}^{K} p(y_k) log_2 P(y_k)$
  - K : 클래스의 수

P(y)가 0 또는 1이면 엔트로피는 거의 0이 된다.

- **Y가 연속확률 변수인 경우 :** $H[Y] = -\int p(y) log_2 P(y) d_y$

1. 의사결정나무

재귀적 분할을 이용한 의사결정나무

재귀 용법

- 하나의 함수에서 자신을 다시 호출하여 작업을 수행하는 방식으로 주어진 문제를 푸는 방법.

어느 한 컴퓨터공학과 학생이 유명한 교수님을 찾아가 물었다.
"재귀함수가 뭔가요?"
"잘 들어보게. 옛날옛날 한 산 꼭대기에 이세상 모든 지식을 통달한 선인이 있었어. 마을 사람들은 모두 그 선인에게 수많은 질문을 했고, 모두 지혜롭게 대답해 주었지. 그의 답은 대부분 옳았다고 하네.
그런데 어느날, 그 선인에게 한 선비가 찾아와서 물었어.
"재귀함수가 뭔가요?"
"잘 들어보게. 옛날옛날 한 산 꼭대기에…

ID3 알고리즘
- 주어진 데이터를 활용해 귀납적 추론을 한다.

(1) 모든 데이터 포인트의 클래스 레이블이 동일하다면, 그 예측값이 해당 클래스 레이블인 자식 노드를 만들고 종료하라.
(2) 파티션을 나눌 수 있는 변수가 남아 있지 않다면, 가장 빈도 수가 높은 클래스 레이블로 예측하는 자식 노드를 만들고 종료하라.
(3) 그게 아니면, 각 변수로 데이터의 파티션을 나눠라.
(4) 파티션을 나눴을 때 엔트로피가 가장 낮은 변수를 택하라.
(5) 선택된 변수에 대한 결정 노드를 추가하라.
(6) 남아 있는 변수들로 각 파티션에 대한 위 과정을 반복하라.

1. 의사결정나무

의사 결정 나무의 장점, 단점

장점

- 매우 설명하기 쉽다.
- 인간의 의사결정 과정을 더 밀접하게 반영한다고 생각할 수 있다.
- 그래픽으로 나타내기 쉽고, 비전문가도 쉽게 해석할 수 있다. (트리의 크기가 작을 경우)
- 트리는 가변수들을 만들지 않고도 질적 설명변수들을 쉽게 처리할 수 있다.

단점

- 정확도가 낮을수 있다.
- 높은 성능 분산을 가질 수 있다.
  - **하나의 학습데이터를 적당히 반으로 나누어, 두 개의 의사결정나무 모델을 만든다면, 두 모델의 결과가 상당히 다를 수 있다는 것을 의미한다.**

2. 모형 결합

모형 결합

2. 모형 결합

모형 결합

모형 결합 기초

- 앙상블 방법론(ensemble methods)이라고도 한다.
- 이는 특정한 하나의 예측 방법이 아니라 복수의 예측 모형을 결합하여 더 나은 성능의 예측을 하려는 시도이다.

모형 결합 효과

- 단일 모형을 사용할 때 보다 성능 분산이 감소한다. (특히, 의사결정나무를 사용하면)
  - 의사결정 나무는 편향이 낮고, 분산이 높은 기법으로 앙상블을 통해 분산을 낮추게되면 결과적으로 편향도 낮고 분산도 낮은 모델이 만들어진다. (모델 분산 X, 모델 개수 n, n개의 모델 분산 X의 평균 분산은 X/n^2)
    - **편향** : 알고리즘에서 잘못된 가정을 했을 때 발생하는 오차
    - **분산** : 학습데이터에 내재된 작은 변동 때문에 발생하는 오차
- 과최적화를 방지한다

2. 모형 결합

모형 결합

취합 방법론 (1) : 다수결, 과반수 투표 (Majority Voting)

- 가장 단순한 방법으로 **서로 다른 모델**에도 적용 가능.
- n개의 모델을 동시에 학습시켜서, 모델 간의 결과값들을 다수결 투표 방식으로 선택한다.
  - hard voting: 단순 투표. 개별 모형의 결과 기준
  - soft voting: 가중치 투표. 개별 모형의 조건부 확률의 합 기준

[Hard Voting 예시]

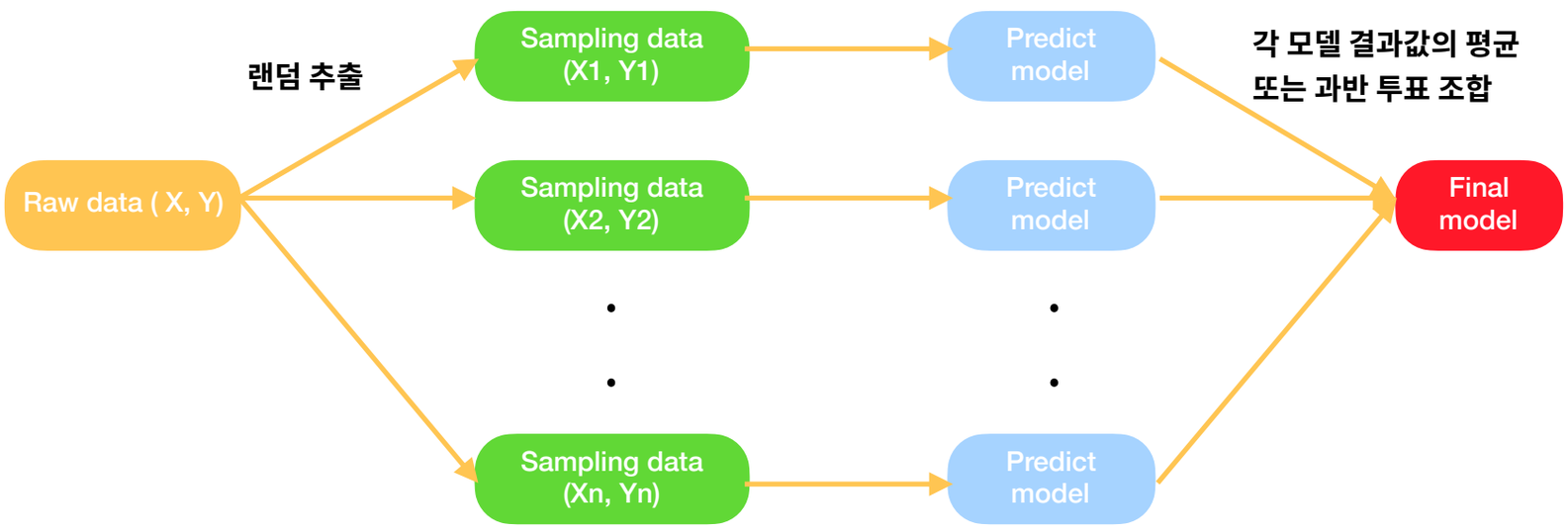
	실제 결과	모델1	모델2	모델3	Voting 결과
x1	1	1	0	1	1
x2	0	0	0	1	0
x3	0	1	1	1	1
x4	1	1	1	1	1
x5	1	1	1	1	1
x6	1	0	1	1	1
Accuracy	-	4/6	4/6	4/6	5/6

2. 모형 결합

모형 결합

취합 방법론 (2) : 배깅 (Bagging)

- **Bootstrap Aggregating**
- **배깅(bagging)**은 **동일한 모형**과 **모형 모수**를 사용하는 대신 부트스트래핑(bootstrapping)과 유사하게 트레이닝 데이터를 랜덤하게 선택해서 다수결 모형을 적용한다.
  - 같은 데이터 샘플을 중복사용(replacement)하지 않으면: Pasting
  - 같은 데이터 샘플을 중복사용(replacement)하면 Bagging
  - 데이터가 아니라 다차원 독립 변수 중 일부 차원을 선택하는 경우에는: Random Subspaces
  - 데이터 샘플과 독립 변수 차원 모두 일부만 랜덤하게 사용하면: Random Patches
- **병렬 학습 가능**
- **집계 방법**
  - 각 모델의 결과값에 대하여 회귀모형일때는 **평균**, 분류모형일때는 **과반수 투표**로 최종 결과를 반환해주는 방식.

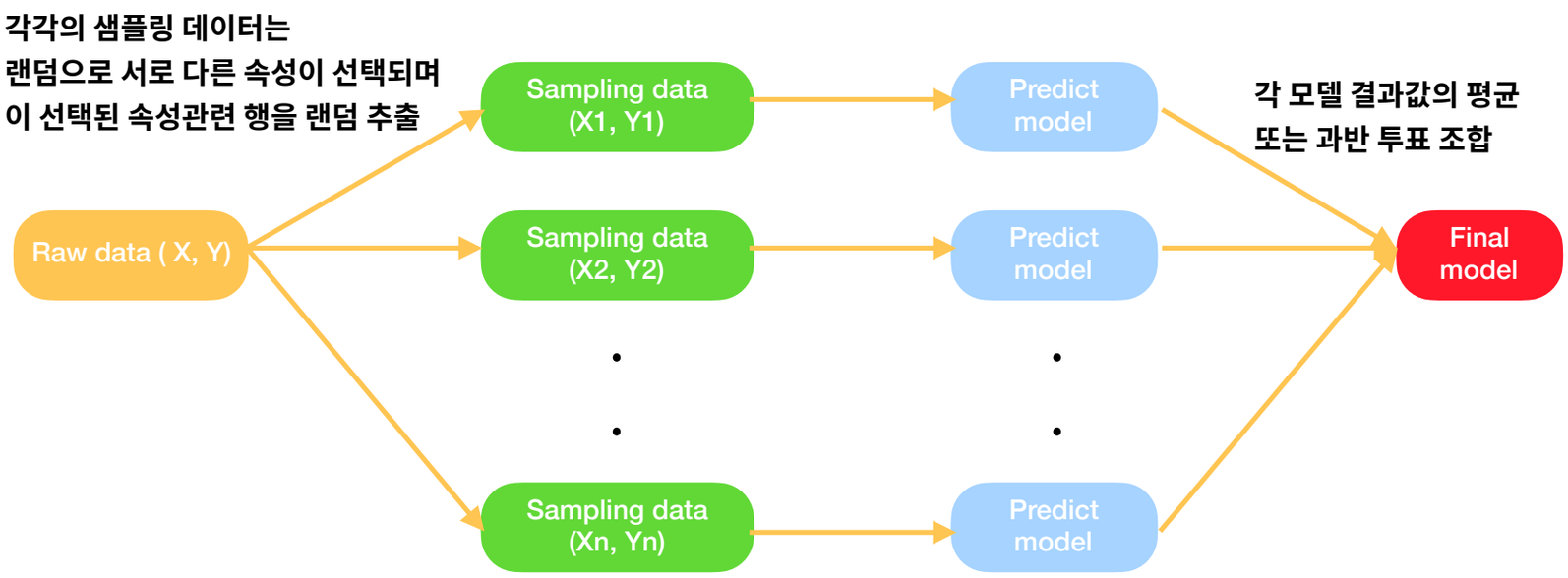


2. 모형 결합

모형 결합

취합 방법론 (3) : 랜덤 포레스트 (Random Forest)

- 의사 결정 나무(Decision Tree)를 개별 모형으로 사용하는 모형 결합 방법이다.
  - 배깅과 마찬가지로 **데이터 샘플의 일부만 선택**하여 사용한다.
    - 이렇게하면 약 1/3은 랜덤하게 빠지게된다. (https://tensorflow.blog/랜덤-포레스트에서-데이터가-누락될-확률/)
  - 노드 분리시 모든 독립 변수들을 비교하여 최선의 독립 변수를 선택하는 것이 아니라 **독립 변수 차원을 랜덤하게 감소**시킨 다음 그 중에서 독립 변수를 선택
- **Feature Importance (독립 변수 중요도)**
  - 포레스트 안에서 사용된 모든 노드에 대해 어떤 독립 변수를 사용하였고 그 노드에서 얻은 information gain을 구할 수 있으므로 각각의 독립 변수들이 얻어낸 information gain의 평균을 비교하면 어떤 독립 변수가 중요한지를 비교할 수 있다.



2. 모형 결합

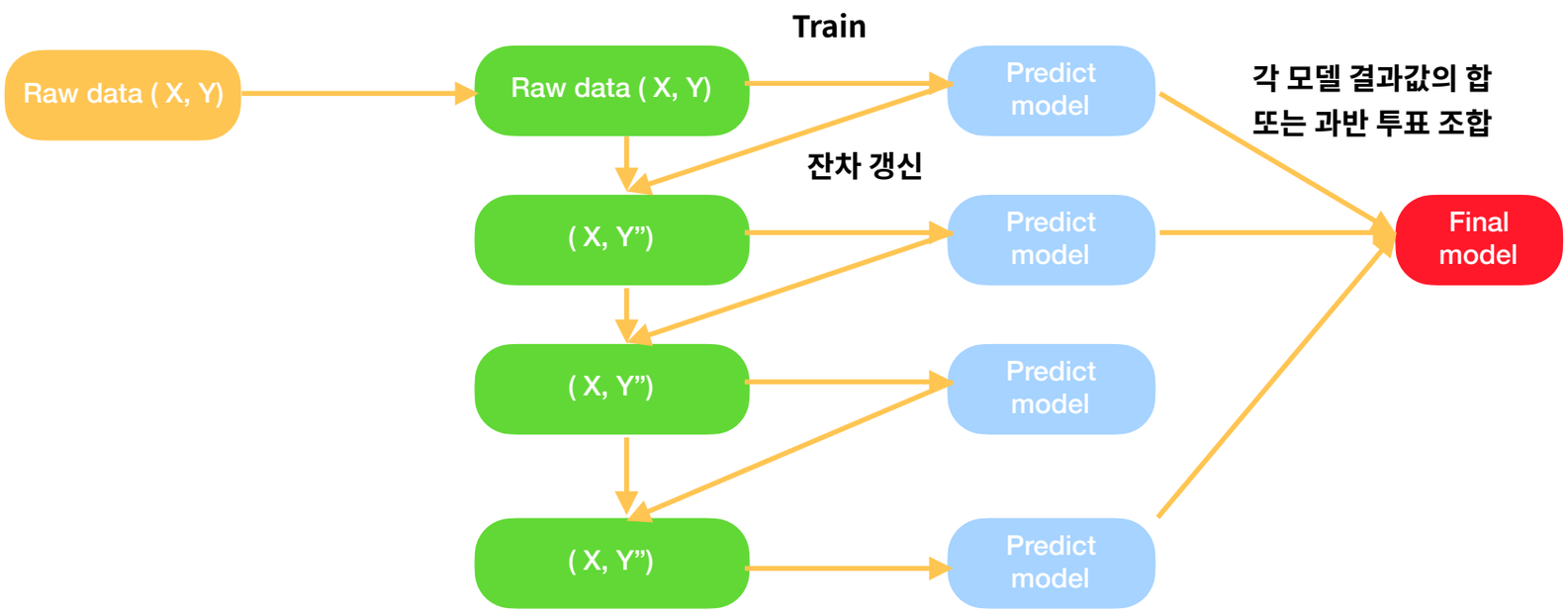
모형 결합

부스팅 방법론

- 부스트(boost) 방법은 미리 정해진 모형 집합을 사용하는 것이 아니라 단계적으로 모형 집합에 포함할 개별 모형을 선택한다. 부스트 방법에서는 성능이 떨어지는 개별 모형을 weak classifier라고 한다. 또한 다수결 방법을 사용하지 않고 각 weak classifier들을 **가중선형결합**하여 최종 모형인 boosted classifier **C**를 생성한다.

$$C_{(m-1)}(x_i)=\alpha_1 k_1(x_i)+\ldots+\alpha_{m-1}k_{m-1}(x_i)$$

$$C_m(x_i)=C_{(m-1)}(x_i)+\alpha_m k_m(x_i)$$



2. 모형 결합

모형 결합

부스팅 Process

- weak classifier들을 **가중선형결합**하여 최종 모형인 boosted classifier **C**를 생성한다. (???)
- 선형결합 과정을 회귀트리를 예시로 살펴보자.

  - 예시 문제 : 정원 손질, 비디오게임, 모자 를 좋아하는 여부로 사람 나이 예측

PersonID	Age	LikesGardening	PlaysVideoGames	LikesHats
1	13	FALSE	TRUE	TRUE
2	14	FALSE	TRUE	FALSE
3	15	FALSE	TRUE	FALSE
4	25	TRUE	TRUE	TRUE
5	35	FALSE	TRUE	TRUE
6	49	TRUE	FALSE	FALSE
7	68	TRUE	TRUE	TRUE
8	71	TRUE	FALSE	FALSE
9	73	TRUE	FALSE	TRUE

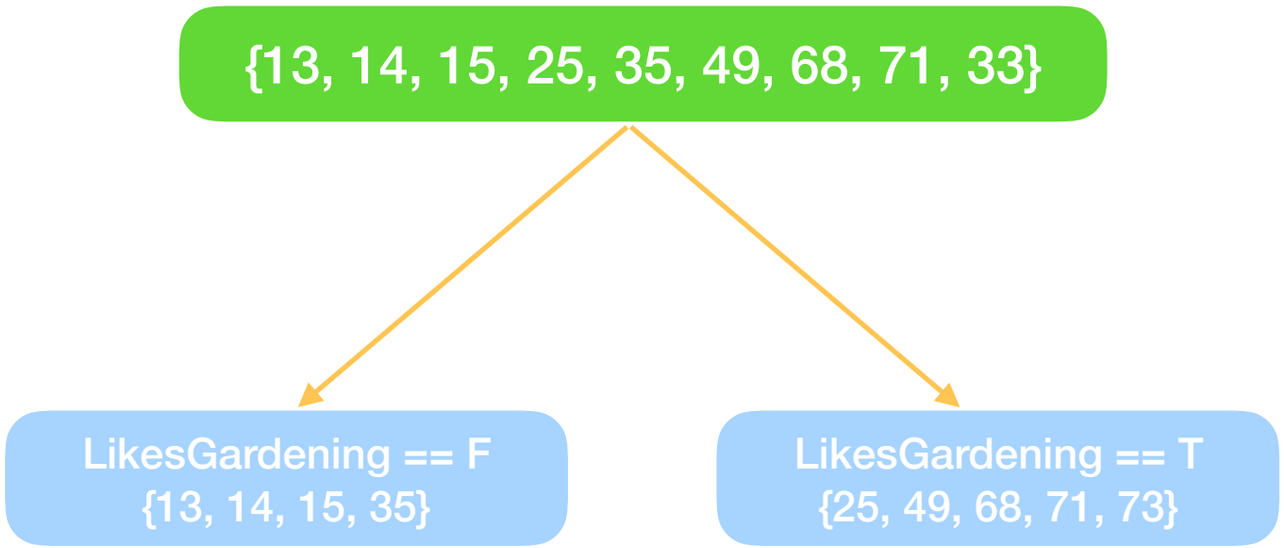
2. 모형 결합

모형 결합

부스팅 Process

1. 회귀 트리를 만들고 예측값, residual을 구한다.
   - **회귀트리의 예측값은 노드의 평균이다.**

Tree1



PersonID	Age	Tree 1 Prediction	Tree1 Residual
1	13	19.25	-6.25
2	14	19.25	-5.25
3	15	19.25	-4.25
4	25	57.2	-32.3
5	35	19.25	15.75
6	49	57.2	-8.2
7	68	57.2	10.8
8	71	57.2	13.8
9	73	57.2	15.8

2. 모형 결합

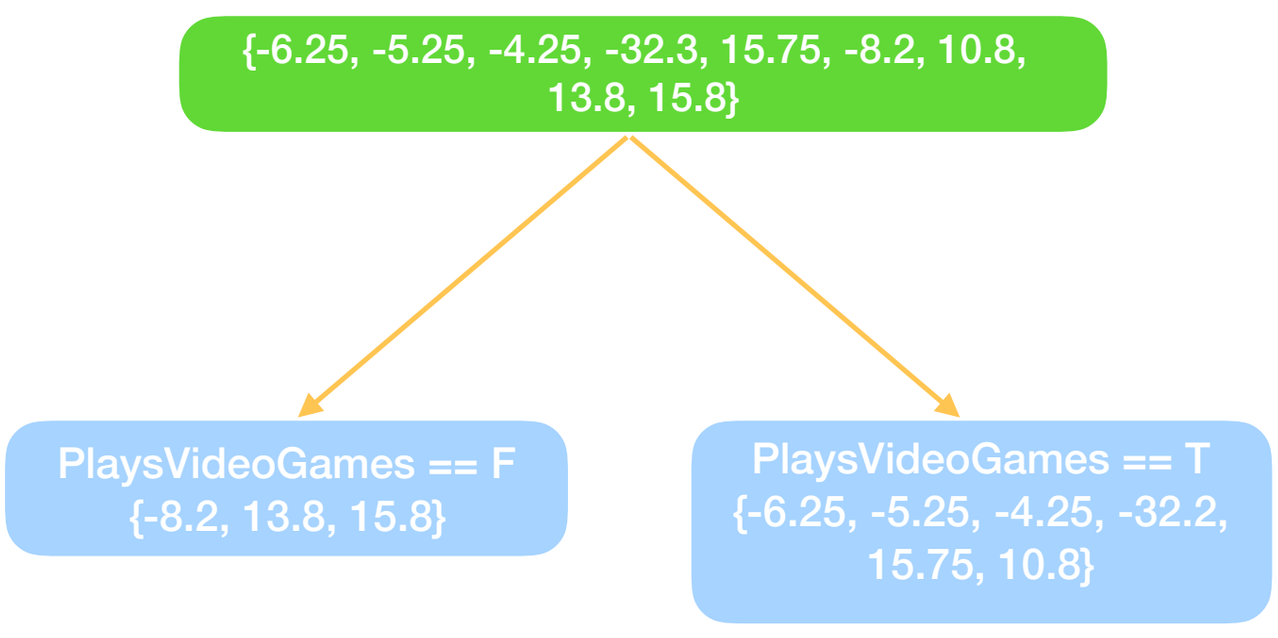
모형 결합

부스팅 Process

2. Tree1의 residual을 예측하는 Tree2를 만든다.
3. Tree2의 prediction 값과 Tree1의 prediction값을 합하여 최종 예측값을 만든다.

- 즉 최종 트리는 Tree1 (C_1) + Tree1의 residual을 예측하는 Tree2 (k_1)

Tree2



ID	Age	Tree 1 Prediction	Tree1 Residual	Tree2 Prediction	Combined Prediction	Final Residual
1	13	19.25	-6.25	-3.567	15.68	2.68
2	14	19.25	-5.25	-3.567	15.68	1.68
3	15	19.25	-4.25	-3.567	15.68	0.68
4	25	57.2	-32.3	-3.567	53.63	28.63
5	35	19.25	15.75	-3.567	15.68	-19.32
6	49	57.2	-8.2	7.133	64.33	15.33
7	68	57.2	10.8	-3.567	53.63	-14.37
8	71	57.2	13.8	7.133	64.33	-6.67
9	73	57.2	15.8	7.133	64.33	-8.67

2. 모형 결합

모형 결합

부스팅 Process

- Tree1의 RSS 는 1994였던 반면에, 부스팅 후 최종 RSS는 1765로 떨어졌다.
- 이로써 우리는 회귀트리들의 선형결합을 통해 모델의 성능을 높일 수 있다는 것을 알게되었다.
- 이 과정을 일반화 시키면 다음과 같이 나타낼 수 있다.

$$C_M = C_{M-1} + k_{M-1}$$

- 위 과정에서 결합 횟수 (M), 트리의 깊이 (depth)는 boosting 모델의 hyper parameter가 된다.

3. 불균형 데이터 문제

불균형 데이터 문제

3. 불균형 데이터 문제

불균형 데이터 문제

불균형 데이터 문제 (Imbalanced Data)

- 현실에서 굉장히 자주 발생하는 문제이다.
  - 예시 : 클릭률, 구매 전환율 등

- 데이터 클래스 비율이 너무 차이가 나면(highly-imbalanced data) 단순히 우세한 클래스를 택하는 모형의 정확도가 높아지므로 모형의 성능판별이 어려워진다. 즉, 정확도(accuracy)가 높아도 데이터 갯수가 적은 클래스의 재현율(recall-rate)이 급격히 작아지는 현상이 발생할 수 있다.

해결 방법

- 비대칭 데이터는 다수 클래스 데이터에서 일부만 사용하는 **언더 샘플링**이나 소수 클래스 데이터를 증가시키는 **오버 샘플링**을 사용하여 데이터 비율을 맞추면 재현율(recall)이 향상된다.

참고 자료 : http://sherry-data.tistory.com/22

4. 모델 최적화

모델 최적화

4. 모델 최적화

모델 최적화

머신러닝 모델이 완성된 이후, 모델의 하이퍼파라미터를 튜닝하는 최적화 과정을 통해 성능을 향상 시킬 수 있다.

- 최적의 하이퍼파라미터 조합을 찾아 최적화를 달성한다.

GridSearch

- 정해진 하이퍼파라미터의 모든 조합을 탐색해보는 방법

5. 군집화

군집화 (Clustering)

데이터의 종속 변수 값을 예측하는 것이 아니라 독립 변수의 특성만으로 데이터의 그룹
즉, 클러스터(cluster)를 형성하는 작업.

5. 군집화

K-Means

데이터에 대한 가정

- (공간 상에서) **가까운 곳**에 위치한 데이터는 서로 유사할 것이다.

K-Means의 목표

- 목적함수 값이 최소화될 때까지 클러스터의 수 K와 각 클러스터의 중심(centroid)을 반복해서 찾는 것이다

$$J = \sum_{k=1}^{K} \sum_{i \in C_k} d(x_i, \mu_k)$$

- d는 두 데이터의 **유사도 함수(Similarity Function)는** 유클리디안 거리(Distance)로 정의한다.

$$d(x_i, \mu_k) = \|x_i - \mu_k\|^2$$

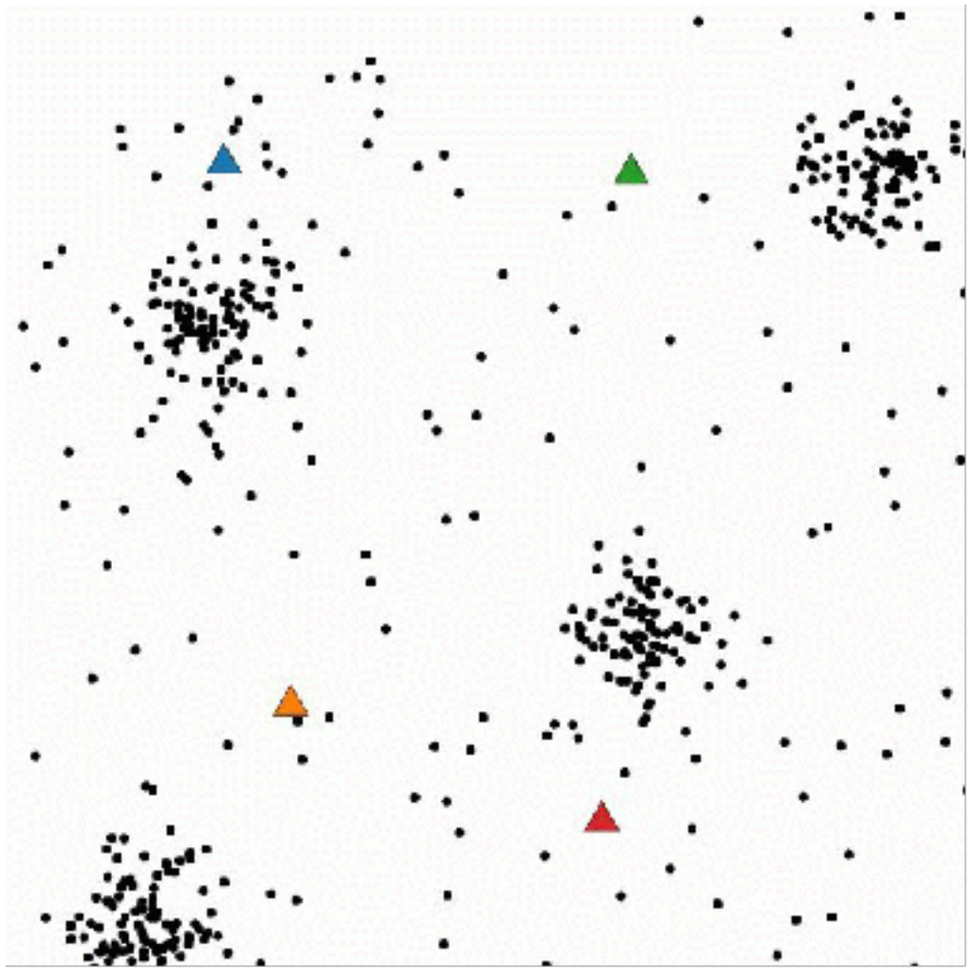
- **결국, 서로 가까운 곳에 위치한 데이터끼리 그룹 (cluster)를 만들겠다는 것.**

5. 군집화

K-Means

연산 절차

1. 임의의 중심값 μ_k 를 고른다. (보통 데이터 샘플 중의 하나를 선택)
2. 중심에서 각 샘플 데이터까지의 거리를 계산
3. 각 데이터 샘플에서 가장 가까운 중심을 선택하여 클러스터 갱신
4. 다시 만들어진 클러스터에 대해 중심을 다시 계산하고 1 ~ 4를 반복한다.



5. 군집화

K-Means

생각해볼 점

- 우리가 가진 데이터를 K-means로 군집화 시키기에 적합한가 ?

	사과 구매 개수	맥주 구매 개수
유저1	1	0
유저2	0	1
유저3	2	1

- 유저 1과 유저 2의 거리 = $\sqrt{2}$
- 유저 1과 유저 3의 거리 = $\sqrt{2}$

코사인 거리로 생각해보면 어떨까?

6. 머신러닝 시스템 설계

시스템 설계

6. 머신러닝 시스템 설계

배치 처리와 배치 학습

배치 처리
- 어떤 처리를 일괄로 하는 것.
  - 실시간 처리와 반대되는 개념.

배치 학습
- 새로운 가중치를 계산하는데 훈련 데이터를 모두 사용하는 것.
  - 온라인 학습과 반대되는 개념
    - 온라인 학습은 새로 만들어진 훈련 데이터만을 이용해서 새로운 가중치를 계산한다.

6. 머신러닝 시스템 설계

배치 처리를 이용한 학습 패턴

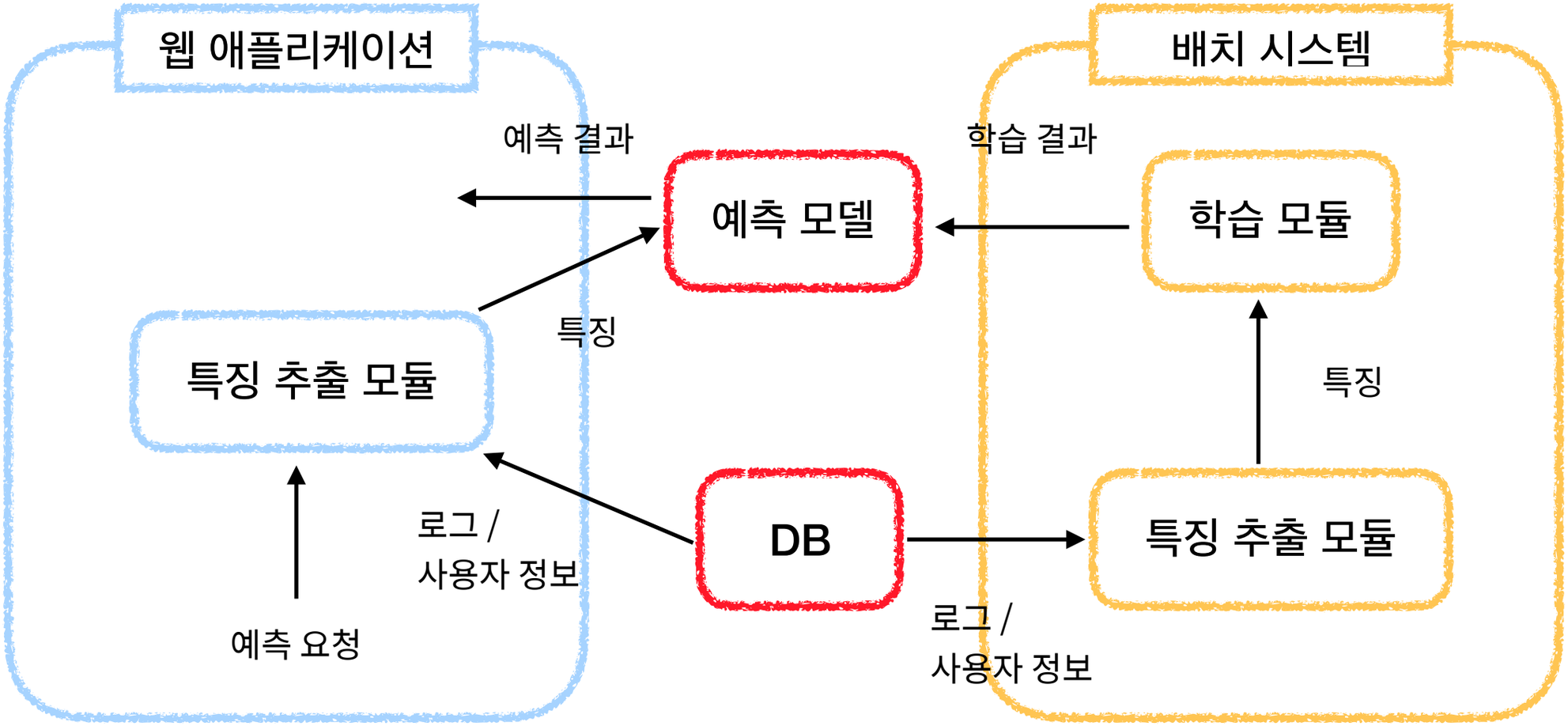
1. 배치 처리로 학습 + 예측 결과를 웹 애플리케이션에서 직접 산출 (예측을 실시간 처리)
2. 배치 처리로 학습 + 예측 결과를 API를 통해 사용 (예측을 실시간 처리)
3. 배치 처리로 학습 + 예측 결과를 DB에 저장하고 사용 (예측을 배치 처리)

6. 머신러닝 시스템 설계

배치 처리를 이용한 학습 패턴

1. 배치 처리로 학습 + 예측 결과를 웹 애플리케이션에서 직접 산출 (예측을 실시간 처리)

- 어플리케이션과 머신러닝 시스템의 결합이 강해짐
- 머신러닝 환경 제약이 생김 (예측 모델의 크기, 연산 부하 등을 고려)



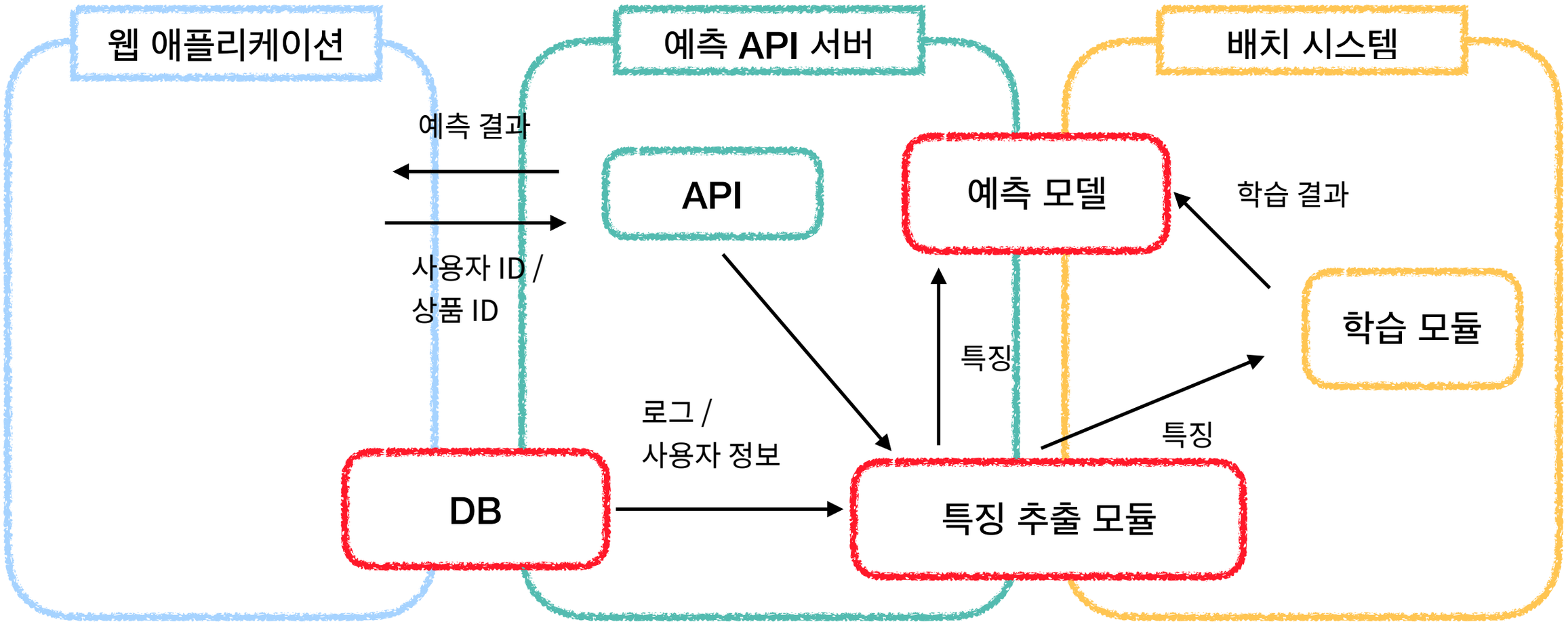
참고 자료 : 머신러닝 실무 프로젝트

6. 머신러닝 시스템 설계

배치 처리를 이용한 학습 패턴

2. 배치 처리로 학습 + 예측 결과를 API를 통해 사용 (예측을 실시간 처리)

- 머신러닝 환경을 자유롭게 구축 가능
- 시스템 규모가 커진다



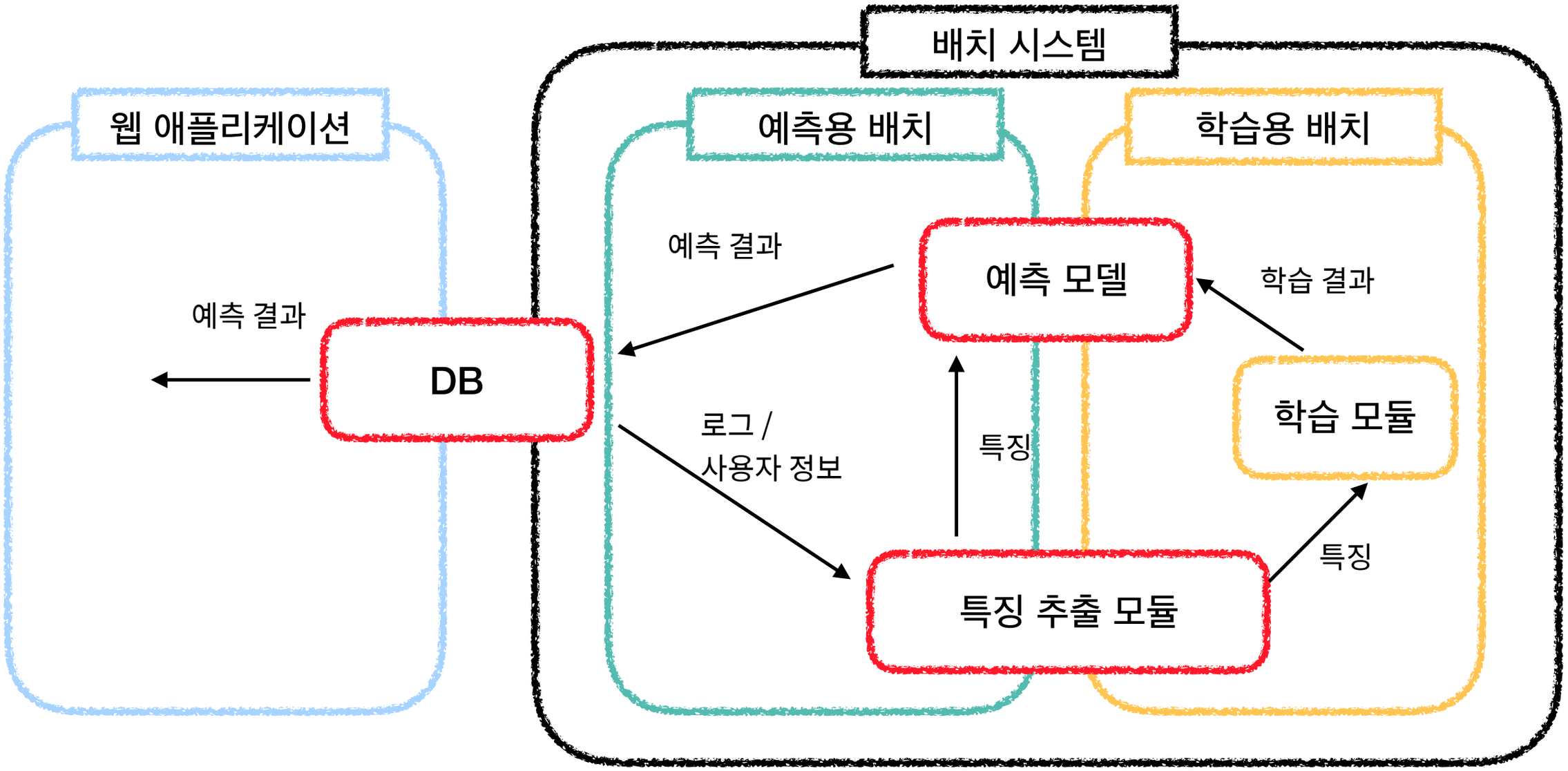
참고 자료 : 머신러닝 실무 프로젝트

6. 머신러닝 시스템 설계

배치 처리를 이용한 학습 패턴

3. 배치 처리로 학습 + 예측 결과를 DB에 저장하고 사용 (예측을 배치 처리)

- 웹 애플리케이션의 응답 속도를 신경쓰지 않아도 된다.
- 각 시스템의 구현 언어가 달라도 된다.



참고 자료 : 머신러닝 실무 프로젝트

다음에
또!
같이!
만나요!

learnit Flip your study