인공지능 보안

이수미

Decision Tree 개요 및 실습

순서

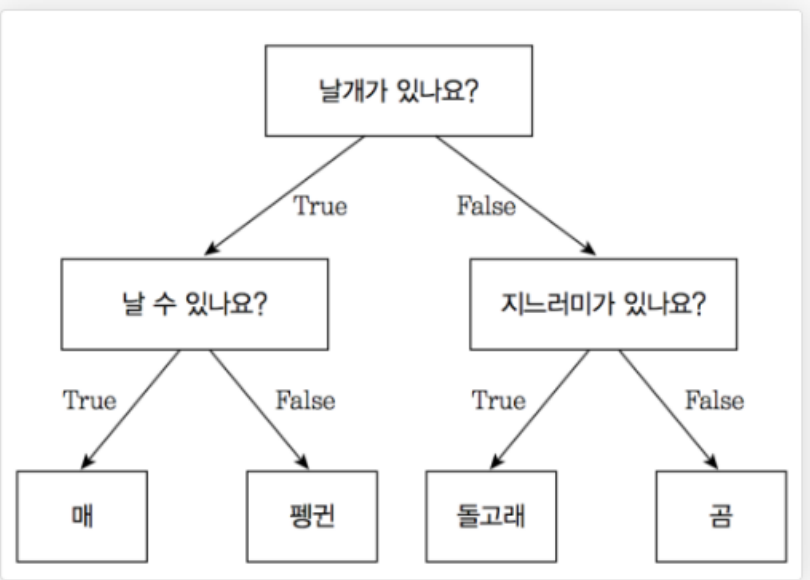
1. Decision Tree 개념
2. Decision Tree 알고리즘
3. Ensemble 이해
4. 투표 분류기
5. 배깅과 랜덤 포레스트

Decision Tree 개념

Decision Tree(결정 트리) 개념

✓ 결정 트리(Decision Tree, 의사결정트리, 의사결정나무라고도 함)는 분류(Classification)와 회귀(Regression) 모두 가능한 지도 학습 모델

- 규칙은 'if-else' 문으로 표현이 가능
- 트리는 일종의 경로를 표현하는 것
- 트리 구조의 마지막 노드에는 분류 문제에서 클래스, 회귀 문제에서는 예측치가 들어감

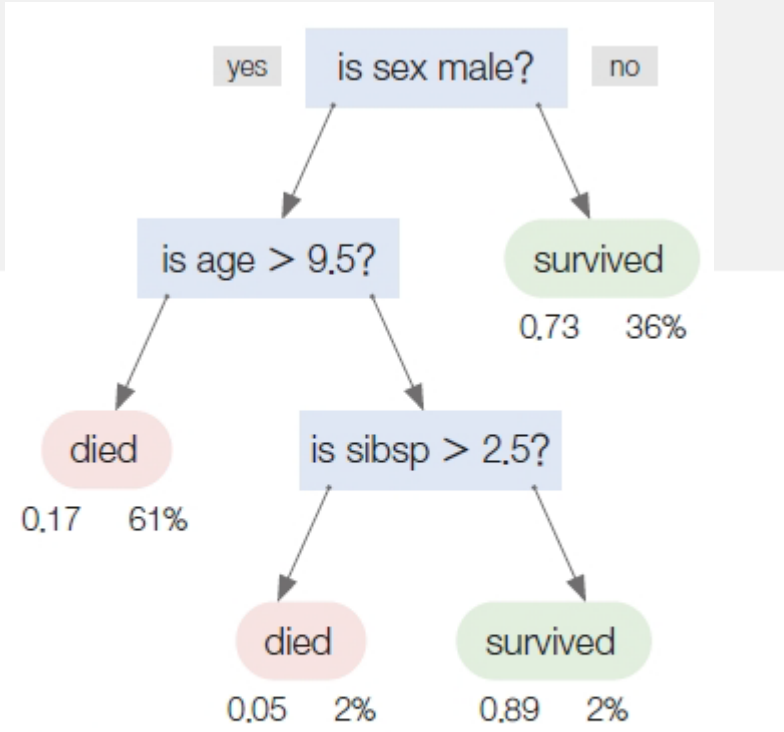


출처: 텐서 플로우 블로그

Decision Tree(결정 트리) 개념

```
if age > 30:
    return True
else:
    return False
```

(a) if-else문의 예



(b) if-else문의 경로 표현

그림 12-1 의사결정트리의 이해

그림 12-2 아키네이터(akinator) 게임 © https://kr.akinator.com/

Decision Tree(결정 트리) 개념

✓딥러닝 기반을 제외한 전통적인 통계 기반의 머신러닝 모델 중 효과와 실용성이 가장 좋음

- 테이블형 데이터에 있어 설명력과 성능의 측면에서 딥러닝 모델들과 대등하게 경쟁
- 앙상블(ensemble) 모델이나 부스팅(boosting) 같은 새로운 기법들이 모델들의 성능을 대폭 향상시키고 있음

Decision Tree(결정 트리) 개념

✓결정 트리 모델: 특정 기준(질문)에 따라 데이터를 구분하는 모델, 전체적인 모양이 나무를 뒤집어 놓은 것과 같아서 Decision Tree라고 함

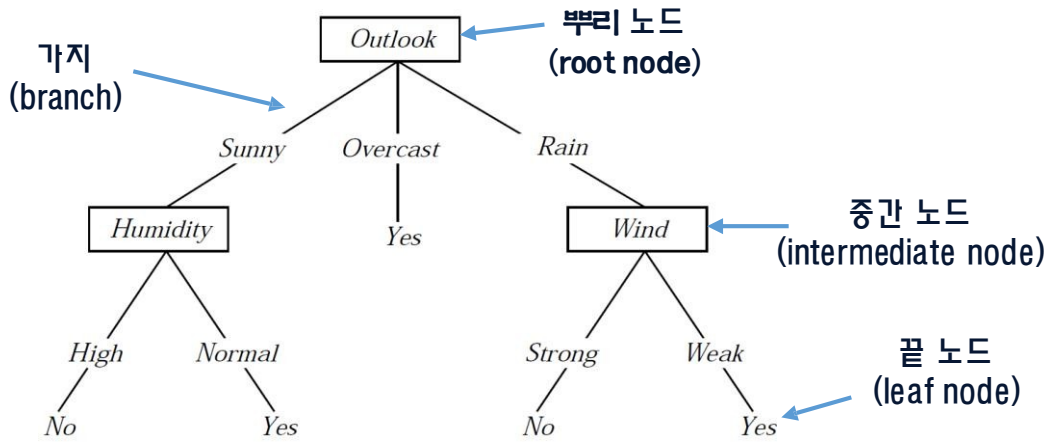
✓노드(Node): 결정 트리에서 질문이나 정답을 담은 네모 상자

Root Node: 맨 처음 분류 기준 (즉, 첫 질문)

Terminal Node 혹은 Leaf Node: 맨 마지막 노드

중간 노드(intermediate node) : 뿌리와 끝 노드를 제외한 노드

✓가지(branch) : 마디와 마디 사이를 연결하는 연결선



Decision Tree(결정 트리) 분류기

✓ 의사결정트리의 노드(node) 구성이 가장 중요

✓ 마지막 노드에 클래스나 예측치를 기입하고 상위의 부모 노드들에는 if-else문의 조건에 해당하는 정보 기입

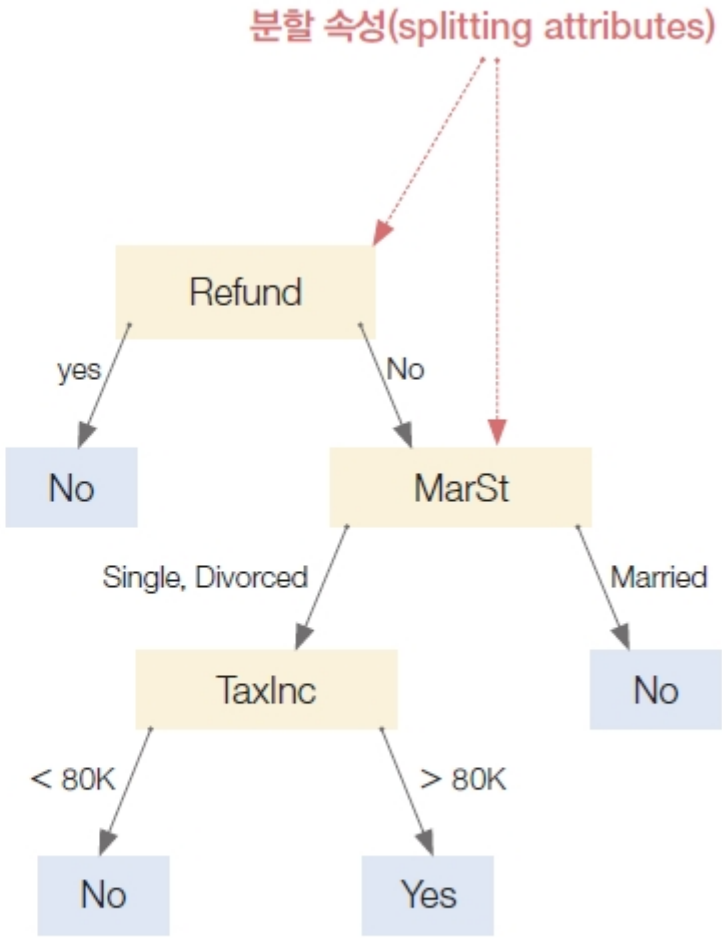
✓분할 속성(splitting attributes) : 부모 노드에 들어가는 if-else문의 조건들

- 어떤 분할 속성이 가장 모호성을 줄일 것인지 파악
  - 예시: 1부터 100까지의 숫자 중 하나를 맞추는 '숫자 예측 게임'
- '재귀적 지역 최적화 방법' : 첫 문제로 분할 속성을 설정하고, 그 다음 남은 데이터 속에서 최적의 분할 속성을 찾아냄

Decision Tree(결정 트리) 분류기

Tid	Refund	Marital Status	Taxable Income	Cheat
1	Yes	Single	125K	No
2	No	Married	100K	No
3	No	Single	70K	No
4	Yes	Married	120K	No
5	No	Divorced	95K	Yes
6	No	Married	60K	No
7	Yes	Divorced	220K	No
8	No	Single	85K	Yes
9	No	Married	75K	No
10	No	Single	90K	Yes

(a) 훈련 데이터(training data)



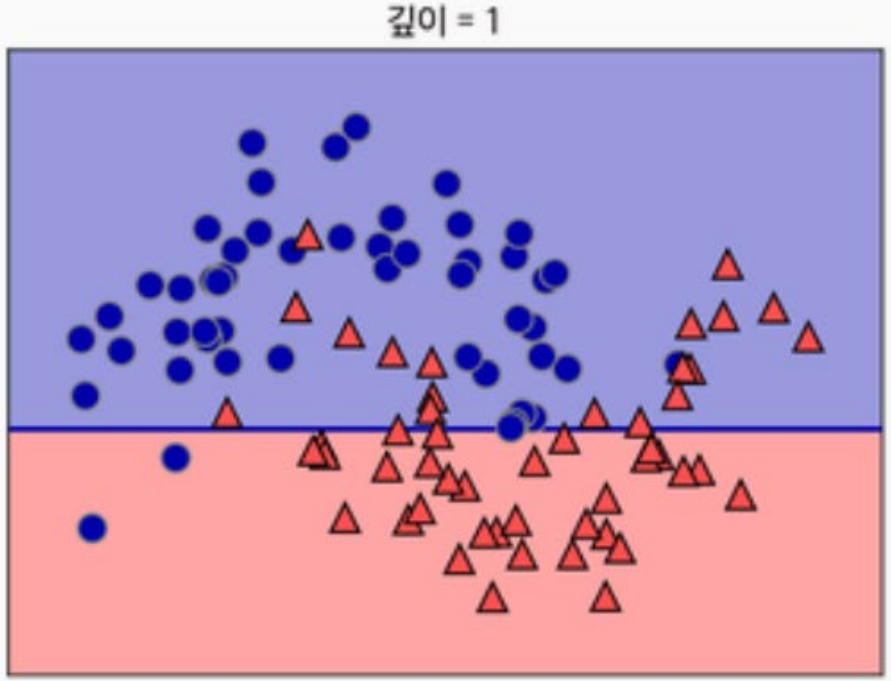
(b) 의사결정트리(decision tree)

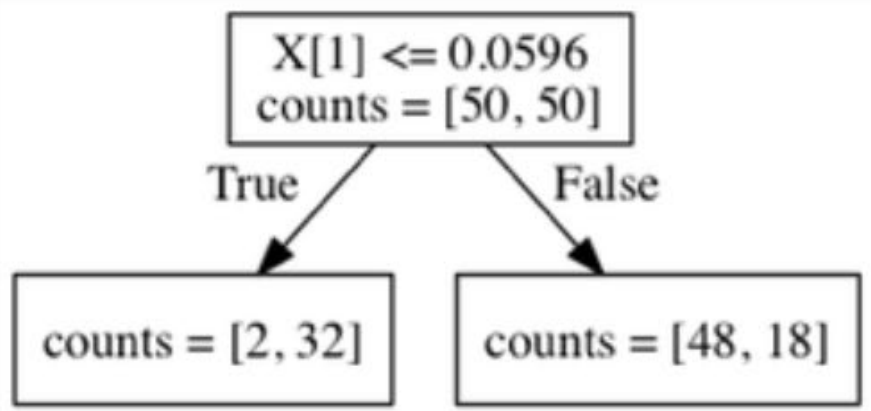
그림 12-3 의사결정트리 만들기

프로세스

✓결정 트리 알고리즘의 프로세스

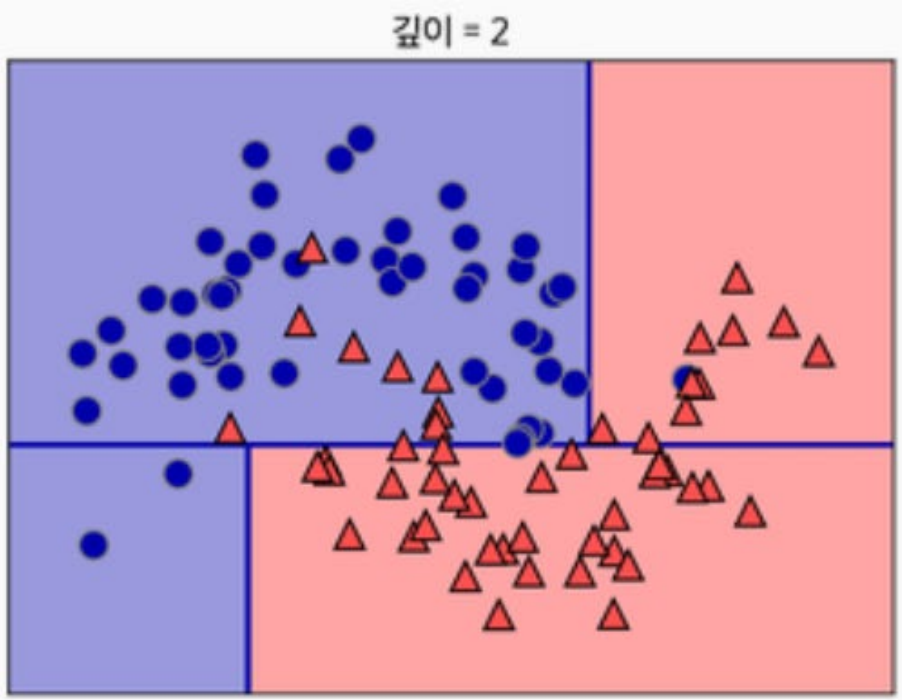
✓ 데이터를 가장 잘 구분할 수 있는 질문을 기준으로 나눈다

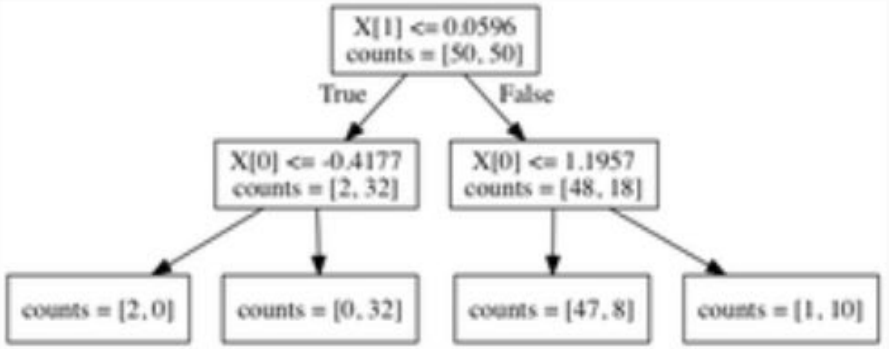




프로세스

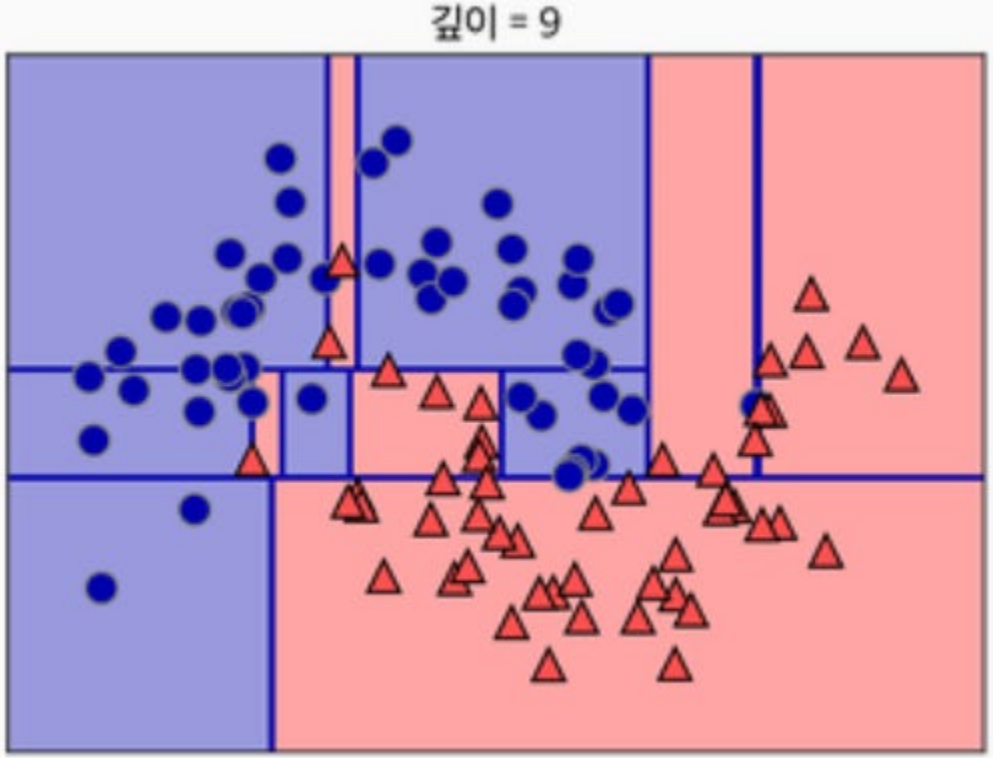
✓나뉜 각 범주에서 또 다시 데이터를 가장 잘 구분할 수 있는 질문을 기준으로 나눕니다





프로세스

✓이를 지나치게 많이 하면 아래와 같이 오버피팅이 된다.



Decision Tree 알고리즘

엔트로피의 이해

✓정보이론이란?

: 최대한 많은 데이터를 매체에 저장하거나 채널을 통해 통신하기 위해 특정 시그널 및 사건에 존재하는 정보의 양을 측정하는 응용 수학의 한 분야

핵심 idea : " 잘 일어나지 않는 사건은 자주 발생하는 사건보다 정보량이 많다 "

(한국전기는 매일 MLCC 제품을 만들고 있다 vs. 한국전기는 수율 100%의 MLCC 제품을 매일 만들고 있다.)

엔트로피의 이해

✓특정 사건에 대한 정보량을 함수화 시키면 다음과 같음

$$Information(x) = -\log_2 P(x)$$

사건이 일어날 확률

✓동전을 던져 앞면이 나오는 사건과 주사위를 던져 눈이 1이 나오는 사건, 두개의 정보량을 비교해보면

전자 는 $-\log_2 1/2 = 1$, 후자는 $-\log_2 1/6 = 2.5849$로 후자가 더 정보량이 높음

✓이때 Entropy는 어떠한 데이터를 표현하기 위한 평균 정보량을 뜻하며 수식은 다음과 같음

$$Entropy(S) = -\sum_x P(x) \log_2 P(x)$$

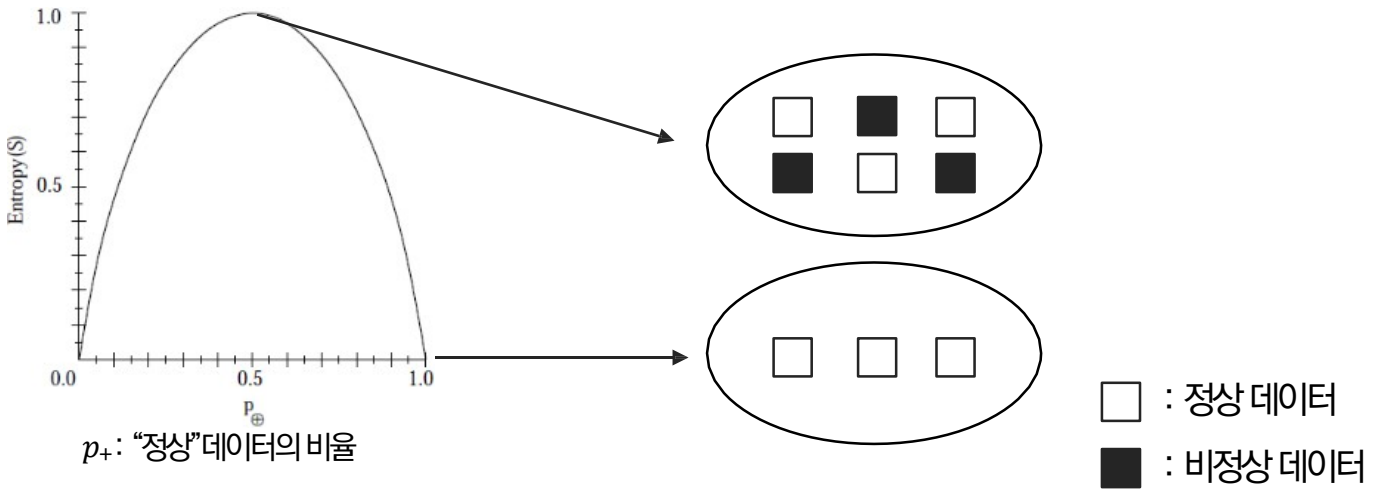
사건이 일어날 확률

사건의 정보량

엔트로피의 이해

✓ p+가 0 또는 1인 경우 전부 비정상 혹은 정상인 의미이므로 필요한 정보량이 적어 entropy가 낮고, 0.5인 경우에는 정상과 비정상이 동일한 비율로 혼재되어 있으므로 필요한 정보량이 많아 entropy가 높음

✓ 즉, entropy는 혼잡도의 개념으로 볼 수 있음

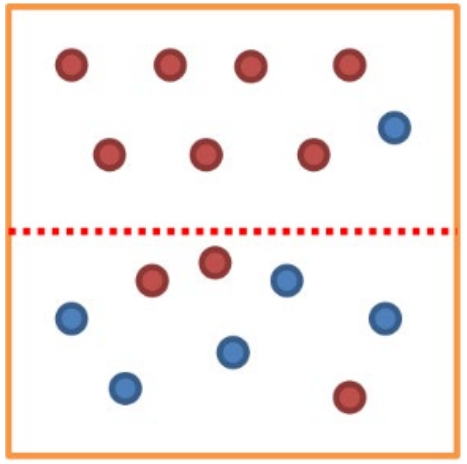


〈p+의 변화에 따른 entropy의 변화〉

엔트로피의 이해

✓불순도(Impurity): 해당 범주 안에 서로 다른 데이터가 얼마나 섞여 있는지

- 아래 그림에서 위쪽 범주는 불순도가 낮고, 아래쪽 범주는 불순도가 높다.
- 바꾸어 말하면 위쪽 범주는 순도(Purity)가 높고, 아래쪽 범주는 순도가 낮다.
- 위쪽 범주는 다 빨간점인데 하나만 파란점이므로 불순도가 낮다고 할 수 있다.
- 반면 아래쪽 범주는 5개는 파란점, 3개는 빨간점으로 서로 다른 데이터가 많이 섞여 있어 불순도가 높다



엔트로피의 이해

✓한 범주에 하나의 데이터만 있다면 불순도가 최소(혹은 순도가 최대)이고, 한 범주 안에 서로 다른 두 데이터가 정확히 반반 있다면 불순도가 최대(혹은 순도가 최소)입니다.

결정 트리는 불순도를 최소화(혹은 순도를 최대화)하는 방향으로 학습을 진행

✓엔트로피가 높다:

- 불순도가 높다, 순도가 낮다
- 엔트로피가 1(한 범주 안에 서로 다른 데이터가 정확히 반반 있다)

✓엔트로피가 낮다:

- 불순도가 낮다, 순도가 높다
- 엔트로피가 0(한 범주 안에 하나의 데이터만 있다)

엔트로피 예제

✓경사, 표면, 속도 제한을 기준으로 속도가 느린지 빠른지 분류해놓은 표

✓첫 줄을 보면 경사가 가파르고(steep), 표면이 울퉁불퉁하고(bumpy), 속도 제한이 있다면(yes) 속도가 느리다(slow)라고 분류

✓ X variables가 경사, 표면, 속도 제한이고, Y variable이 속도

경사	표면	속도 제한	속도
steep	bumpy	yes	slow
steep	smooth	yes	slow
flat	bumpy	no	fast
steep	smooth	no	fast

엔트로피 예제

경사	표면	속도 제한	속도
steep	bumpy	yes	slow
steep	smooth	yes	slow
flat	bumpy	no	fast
steep	smooth	no	fast

✓Pi는 한 영역 안에 존재하는 데이터 가운데 범주 i에 속하는 데이터의 비율

✓P_slow = 2/4 = 0.5 (slow 라벨 개수 = 2개, 전체 examples 수 = 4개)

P_fast = 2/4 = 0.5 (fast 라벨 개수 = 2개, 전체 examples 수 = 4개)

✓현재 범주 전체의 엔트로피는 얼마일까? “1” ← 서로 다른 데이터가 정확히 반반 있기 때문

$$Entropy(S) = -\sum_{x} P(x) \log_2 P(x)$$

사건이 일어날 확률

사건의 정보량

- Entropy = − P_slow * $\log_2$ (P_slow) − P_fast * $\log_2$ (P_fast) = −0.5 * $\log_2$ (0.5)−0.5 * $\log_2$ (0.5) = 1

정보 획득(Information gain)

✓엔트로피가 1인 상태에서 0.7인 상태로 바뀌었다면 정보 획득(information gain)은 0.3입니다.

✓정보 획득(Information Gain): 분기 이전의 엔트로피에서 분기 이후의 엔트로피를 뺀 수치

Information gain = entropy(parent) - [weighted average] entropy(children)

- entropy(parent)는 분기 이전 엔트로피이고, entropy(children)은 분기 이후 엔트로피
- [weighted average] entropy(children)는 entropy(children)의 가중 평균
  - 분기 이후 엔트로피에 대해 가중 평균을 하는 이유는 분기를 하면 범주가 2개 이상으로 쪼개지기 때문
  - 범주가 하나라면 위 엔트로피 공식으로 바로 엔트로피를 구할 수 있음.
  - 하지만 범주가 2개 이상일 경우 가중 평균을 활용하여 분기 이후 엔트로피를 구함

정보 획득(Information gain)

① 전체 엔트로피: $$Info(D) = -\sum_{i=1}^{n} p_i \log_2(p_i)$$

② 속성별 엔트로피: $$Info_A(D) = -\sum_{j=1}^{v} \frac{|D_j|}{|D|} \times Info(D_j)$$

- 속성 A로 데이터를 분류했을 때 속성 A가 가진 모든 클래스의 각 엔트로피를 계산한 후, 데이터의 개수만큼 가중치를 줌

③ 속성별 정보 이득: $$Gain(A) = Info(D) - Info_A(D)$$

- 정보 이득이 크면 클수록 A를 기준으로 데이터를 분류했을 때 얻을 수 있는 정보량이 많다는 뜻

- A를 기준으로 데이터를 나눌 때 엔트로피가 작다면 해당 속성을 기준으로 데이터를 나누기 좋다고 볼 수 있음

ID3 알고리즘

✓성장(grow) : 일반적으로 결정트리를 생성하는 방법을 성장이라고 부름. 트리를 성장시키는 개념

✓ID3(Iterative Dichotomiser 3) : 반복적으로(iteratively) 데이터를 나누는(divides) 알고리즘

- 톱다운(top-down) 방식으로 데이터를 나누면서 탐욕적(greedy)으로 현재 상태에서 최적화를 추진하는 방법을 선택

✓기본적인 ID3 알고리즘

```
if 데이터 집합에 있는 모든 항목이 같은 레벨임 :
  분류 항목 표시를 반환함(ex. buy_yes)
else :
  Find Best Split_branch_attribute(ex, attribute-age)
  해당 속성(attribute)을 기준으로 데이터셋 분할
  가지 노드(branch node) 생성
    for each branch
      branch_node.add(Recursive branch split)
  return branch node
```

ID3 알고리즘

✓결정 트리 알고리즘은 정보 획득을 최대화하는 방향으로 학습이 진행됩니다.

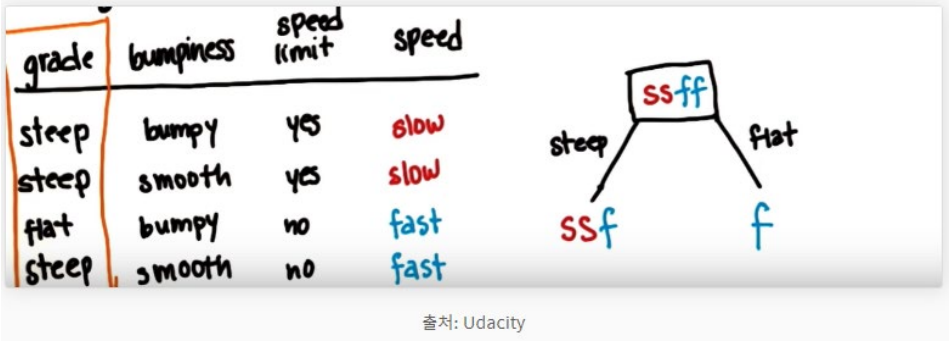
어느 feature의 어느 분기점에서 정보 획득이 최대화되는지 판단을 해서 분기가 진행됩니다

- 위에서 예로 든 속도 문제를 다시 보겠습니다. 맨 처음 전체 엔트로피 = 1 이라고 했습니다.

  즉, entropy of parent = 1입니다.

경사	표면	속도 제한	속도
steep	bumpy	yes	slow
steep	smooth	yes	slow
flat	bumpy	no	fast
steep	smooth	no	fast

ID3 알고리즘



✓경사(grade) 기준 분기

- entropy(steep)= $-P\_slow*\log_2(P\_slow)-P\_fast*\log_2(P\_fast)=-(2/3)*\log_2(2/3)-(1/3)*\log_2(1/3)=0.9184$
- entropy(flat) = 0 이고, entropy(steep) = 0.9184입니다. 이제 분기 이후 노드에 대한 가중 평균을 구해보겠습니다.
  - [weighted average] entropy(children)

    = weighted average of steep * entropy(steep) + weighted average of flat * entropy(flat) = 3/4 * (0.9184) + 1/4 * (0)

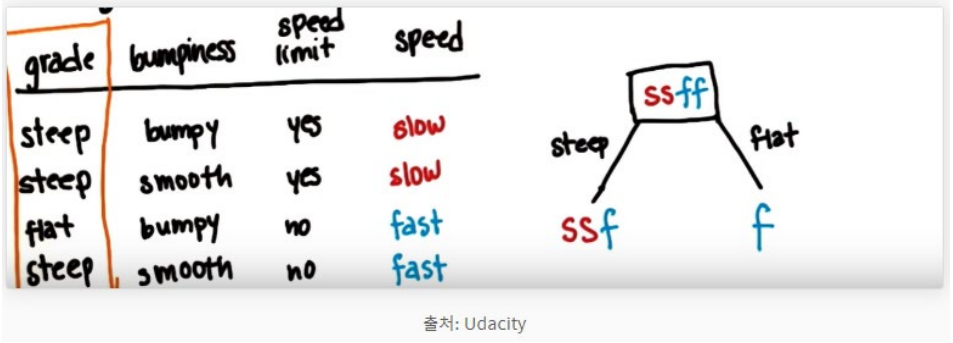
    = 0.6888

  ※ weighted average of steep = ¾인 이유는 4개의 데이터 중 steep에 해당하는 데이터는 3개이기 때문

  마찬가지로 weighted average of flat = 1/4인 이유는 4개의 데이터 중 flat에 해당하는 데이터는 1개이기 때문

  - 따라서, 경사(grade)를 기준으로 분기한 후의 엔트로피는 0.6888
- information gain = entropy(parent) – [weighted average] entropy(children) = 1 – 0.6888 = 0.3112
- 경사 feature를 기준으로 분기를 했을 때는 0.3112만큼의 정보 획득(information gain)이 있다는 뜻

ID3 알고리즘



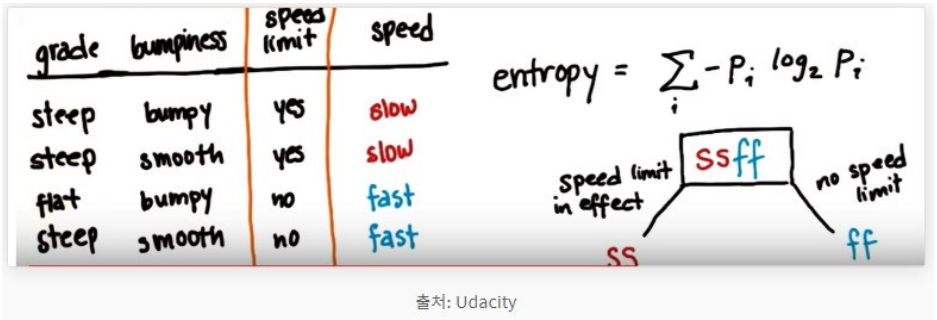
출처: Udacity

✓표면(bumpiness) 기준 분기

- bumpy에는 slow/fast, smooth에도 slow, fast가 있다.
  나의 범주에 서로 다른 데이터가 정확히 반반 있습니다. 이럴 때 엔트로피는 1
- entropy(bumpy) = − P_slow * log2(P_slow) − P_fast * log2(P_fast)
  = − (1/2) * log2(1/2) − (1/2) * log2(1/2) = 1
- entropy(smooth) = − P_slow * log2(P_slow) − P_fast * log2(P_fast)
  = − (1/2) * log2(1/2) − (1/2) * log2(1/2) = 1

- information gain = entropy(parent) − [weighted average] entropy(children)
  = 1 − (2/4) * 1 − (2/4) * 1 = 0
- 표면을 기준으로 분기했을 때는 정보 획득이 전혀 없다는 뜻

ID3 알고리즘



출처: Udacity

✓ 속도제한 기준 분기

- ✓ entropy(yes) = − P_slow * log2(P_slow) − P_fast * log2(P_fast) = − (1) * log2(1) − (0) * log2(0) = 0
- ✓ entropy(no) = − P_slow * log2(P_slow) − P_fast * log2(P_fast) = − (0) * log2(0) − (1) * log2(1) = 0

- ✓ information gain = 1 − (2/4) * 0 − (2/4) * 0 = 1

- ✓ 경사, 표면, 속도제한 기준으로 분기 했을 때 정보 획득은 각각 0.3112, 0, 1
  결정트리는 정보 획득이 가장 많은 방향으로 학습이 진행된다.
  따라서 첫 분기점을 속도제한 기준으로 설정 (max_depth나 min_sample_split으로 설정한 범위까지 분기)
  ➔ 이것이 바로 결정트리의 전체적인 알고리즘

의사결정트리 알고리즘의 특징

① 재귀적 작동 : 가지가 되는 속성을 선택한 후 해당 가지로 데이터를 나누면, 이전에 적용되었던 알고리즘이 남은 데이터에 적용됨

✓ 남은 데이터에서만 최적의 모델을 찾는 방법으로 작동

② 속성 기준으로 가지치기 수행 : 가장 불확실성이 적은 속성을 기준으로 가지치기를 수행

③ 중요한 속성 정보 제공 : 처음 분리 대상이 되는 속성이 가장 중요한 속성

✓ 이 특징을 '해석 가능한 머신러닝'이라고 부름

Ensemble의 이해

앙상블의 이해

✓ 대중적인 데이터 분석 알고리즘

- 최근 머신러닝/딥러닝 분야에서 딥러닝 다음으로 부스팅(boosting) 알고리즘이 핵심적으로 사용됨
- 선형회귀나 로지스틱 회귀는 가장 대중적인 알고리즘이고, 그 다음이 의사결정트리와 앙상블 계열 알고리즘, 딥러닝

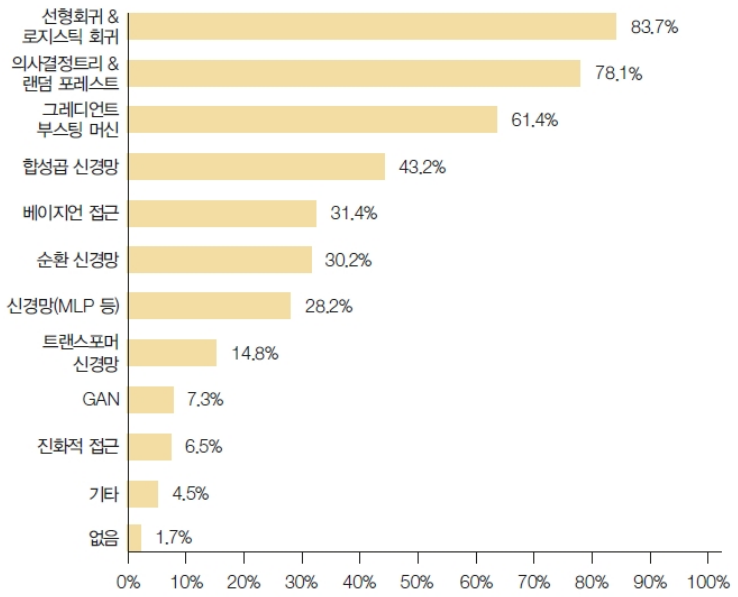


그림 13-1 데이터 분석 시 사용하는 알고리즘 설문조사

앙상블의 개념

✓앙상블(ensemble) : 여러 개의 알고리즘들이 하나의 값을 예측하는 기법을 통칭하여 말함

- 회귀 문제에서는 가중 평균이나 단순 평균을 구하는 방식으로 Y 값을 예측

✓메타 분류기(meta-classifier)라고도 부름

- 메타(meta)는 일종의 상위 또는 추상화라는 개념.

  여러 분류기들을 모아 하나의 분류기를 만들어 이를 메타 분류기라고 부른다

✓시간이 굉장히 오래 걸리지만 비교적 좋은 성능을 냄

앙상블의 개념

✓하나의 데이터를 넣음 → 이를 여러 모델에 학습시키고 → 테스트 데이터를 각 모델에 입력 →

투표 또는 여러 가중치 기법을 적용하여 최종 선택

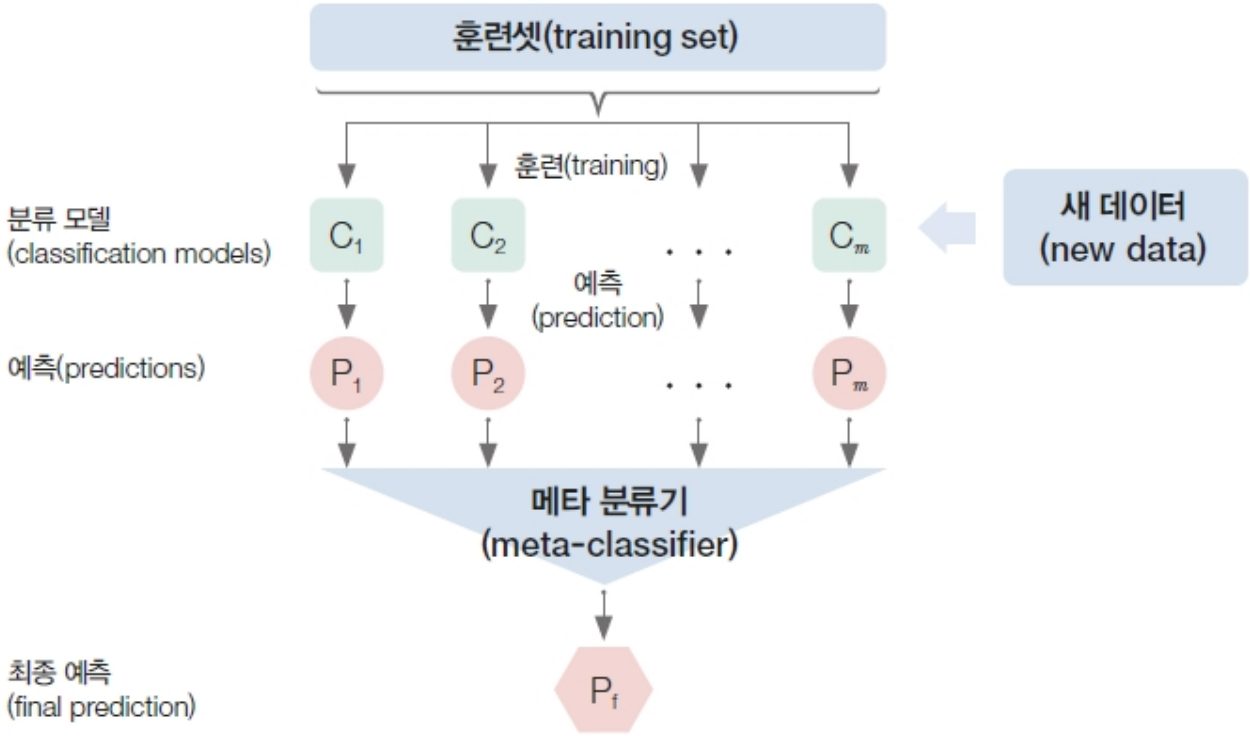


그림 13-2 앙상블 모델

앙상블 기법

✓바닐라 앙상블 : 가장 기본적인 앙상블 기법. 바닐라라고 하면 아이스크림에서 아무것도 첨가되지 않은 맛인데, 바닐라 앙상블도 아무것도 처리하지 않은 앙상블 모델을 의미. 일반적으로 가중치 평균이나 투표 방식으로 만들어지는 앙상블 모델

✓부스팅 : 하나의 모델에서 여러 데이터를 샘플링한 다음 그 샘플링 된 데이터로 각각의 모델을 만드는 기법

✓배깅 : ‘boosting aggregation(부스팅 집합)’의 줄임말로 부스팅을 좀 더 발전시킨 기법

투표 분류기

투표 분류기의 개념

Colab 실습

- ✓투표 분류기(voting classifier) : 여러 개의 모델을 만들어 모두 같은 데이터를 넣고 결과를 취합하여 가장 많이 선택된 결과를 취함
- ✓앙상블 모델의 가장 기본적인 형태
- ✓다수결 분류기(majority voting classifier)라고도 부름
- ✓또는 각 분류기마다 가중치를 주고 해당 가중치를 각 모델에 곱하여 가중치의 합을 구하는 방식
- ✓장점 : 다양한 모델을 만든 후, 다음 단계로 매우 쉽게 만들 수 있음

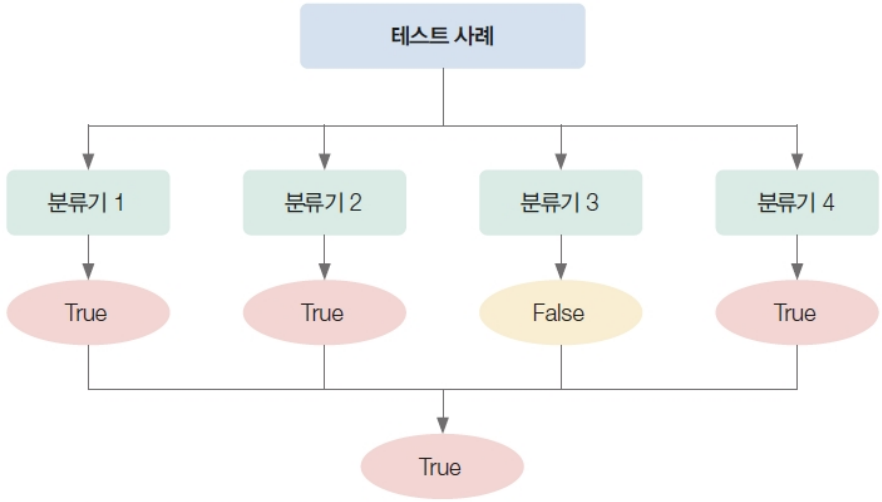


그림 13-3 투표 분류기의 기본 형태

배깅과 랜덤 포레스트

배깅의 개념

✓ 배깅(bagging) : 하나의 데이터셋에서 샘플링을 통해 여러 개의 데이터셋을 만든 다음 각 데이터셋 마다 모델을 개발하여 투표 분류기로 만드는 기법

- 단순하면서 성능이 높아 특히 트리 계열 알고리즘과 함께 많이 사용되며 통계적인 샘플링 기법이나 딥러닝 기법과도 함께 사용

✓샘플링(sampling): 다루고자 하는 데이터가 전체 모수에서 일부분을 뽑아서 데이터를 분석

배깅의 개념

✓ 배깅의 장점 : 다양한 데이터셋에서 강건한 모델(robust model)을 개발할 수 있다

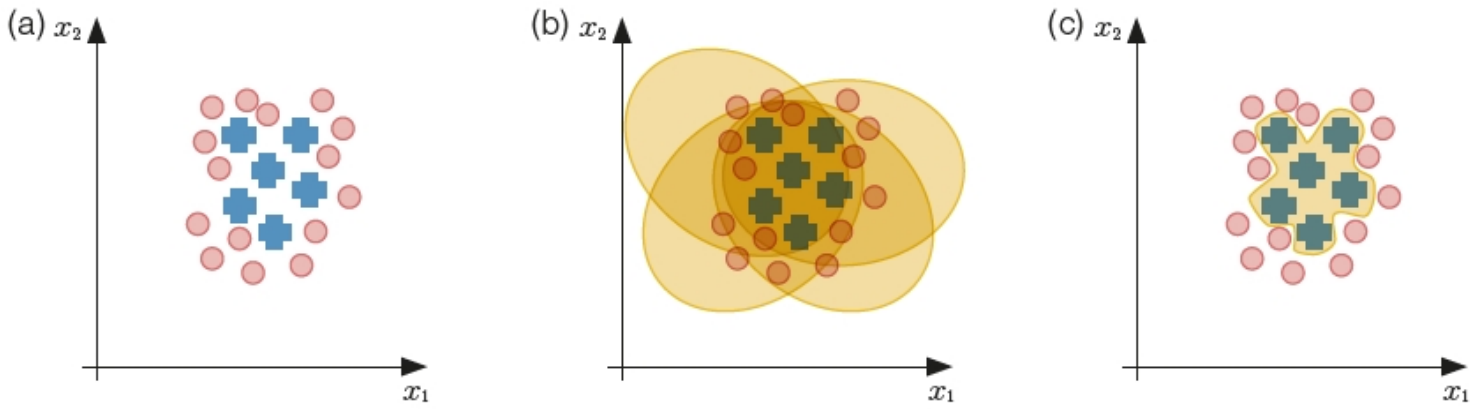


그림 13-4 배깅의 모델링

부트스트래핑

✓ 부트스트래핑(bootstrapping) : 모수 데이터로부터 학습 데이터를 추출할 때 임의의 데이터를 추출한 후 복원 추출하는 여러 번의 과정

✓복원추출 : 전체 데이터에서 먼저 일부를 추출하여 이를 '학습 데이터셋 1'이라고 부른 다음 다시 그 데이터를 모수에 집어넣고 '학습 데이터 셋 2'를 뽑는 방식

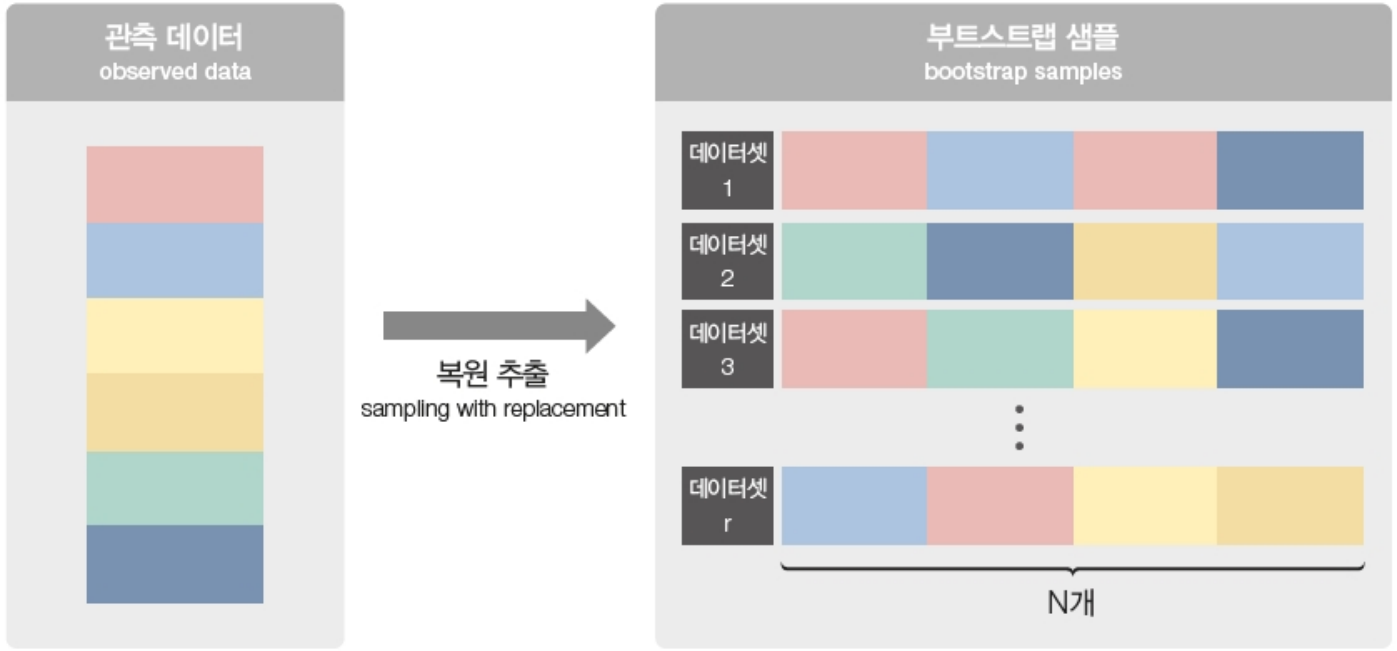


그림 13-5 복원추출

부트스트래핑

✓ 부트스트래핑의 유래

- 부트스트래핑은 원래 카우보이 워크화의 뒤에 달려 있는 조그마한 끈을 말한다. 그림과 같이 해당 끈에 손을 넣어서 워크화를 착용했다.
- 남에게 도움 받지 않고 스스로 무엇인가를 시작할 때 이를 지칭하는 대표적인 용어로 사용한다. 데이터 과학에서는 처음 모수 데이터에서만 일부 데이터를 추출하여 사용하는 행위를 의미하며, 컴퓨터 과학에서는 외부데이터의 주입 없이 컴퓨터가 메모리에 저장된 정보만으로 부팅되는 것을 의미

그림 13-6 워크화의 부트스트래핑

부트스트래핑

✓ '.632 부트스트래핑' 기법 : 전체 데이터 S에서 n번의 데이터를 추출할 때, 이를 많이 추출할수록 각 데이터가 나타날 확률이 63.2%에 가까워 짐

$$1-\prod\left(1-\frac{1}{d}\right)=1-\left(1-\frac{1}{d}\right)^{d}\approx 1-e^{-1}$$

배깅(bagging)

✓ 배깅(bagging)은 부트스트랩 집합이라는 의미의 ‘bootstrap aggregation’의 약자로,

말 그대로 부트스트랩 연산의 집합이라는 개념

✓ 데이터셋으로부터 부분집합 n개를 추출 → 앙상블 방법과 달리 하나의 모델에 다양한 데이터셋을 넣어서 n개의 모델을 생성

✓ 높은 분산으로, 일반적인 모델로 만들 경우 과적합이 심한 데이터셋에 좀 더 강건

- 각 모델들은 해당 데이터셋에 맞춰진 과적합 모델

배깅(bagging)

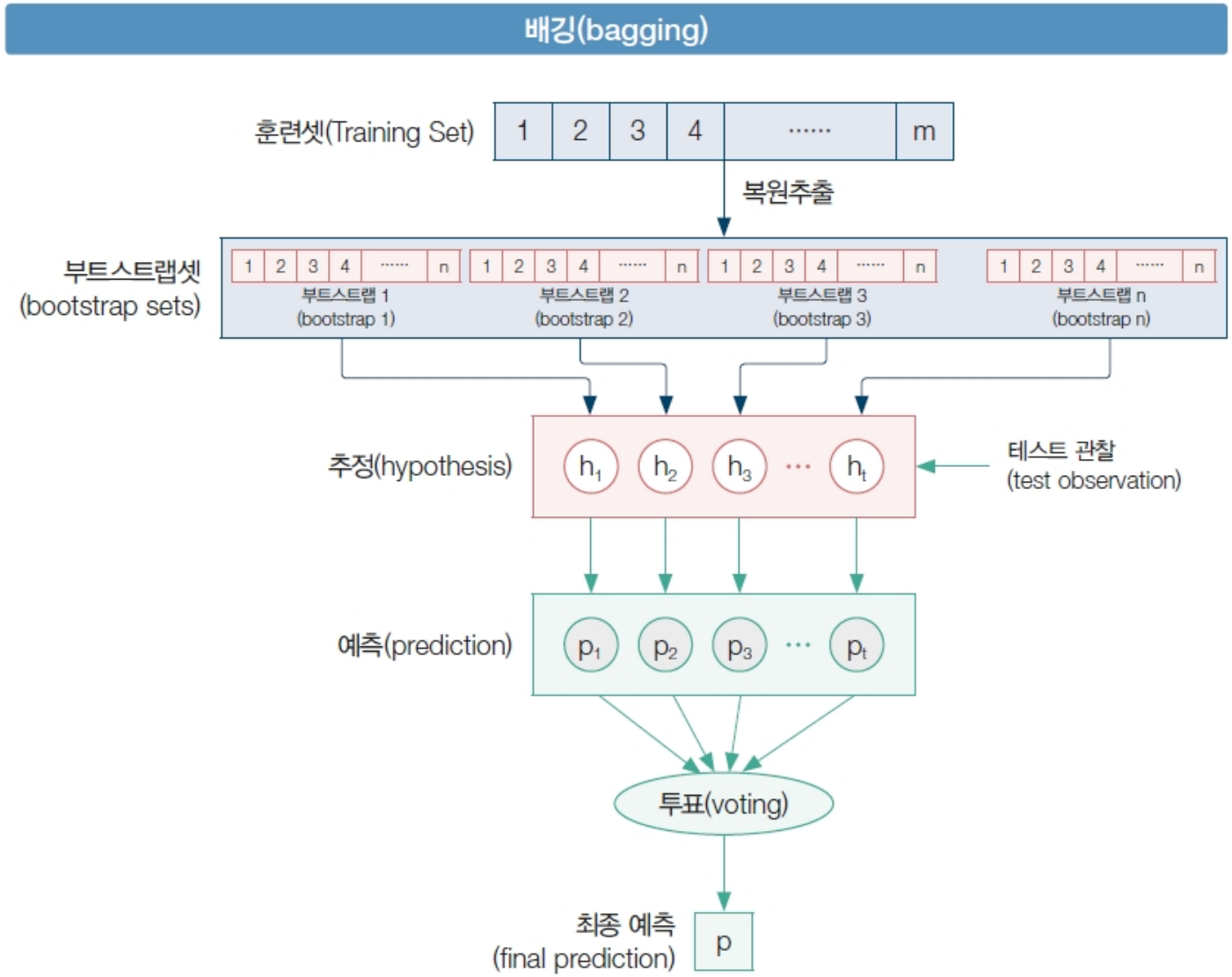


그림 13-7 배깅 업무 순서도

배깅(bagging)

✓Out-of-bag Error

✓배깅 모델의 성능을 측정하기 위해서 정확도나 정밀도 외에 'Out-of-bag Error'라는 지표를 사용한다. 일반적으로 'OOB error estimation'이라고 부른다.

✓ 배깅에서 부분집합을 생성할 때 일부 데이터만 학습에 사용되는데, 각 부분집합에서 학습에 사용되지 않은 데이터셋에 대해서만 성능을 측정하여 배깅 모델의 효과를 측정하는 것이다.

✓ 기본적으로 검증셋(validation set)과 유사한 방식으로 학습에 사용하지 않은 데이터를 가지고 학습의 성능을 측정한다고 이해할 수 있다.

$$Acc(M) = \frac{1}{k}\sum_{i=1}^{k}(0.632 \times Acc(M_i)_{test\_set} + 0.368 \times Acc(M_i)_{train\_set})$$

랜덤 포레스트

✓랜덤 포레스트(random forest) : 하나의 모델을 나무(tree)라고 한다면 이러한 나무들을 이용해 랜덤하게 데이터를 뽑아서 숲을 생성하는 알고리즘

✓배깅 알고리즘을 의사결정트리에 적용한 모델

✓사이킷런 배깅 분류기 BaggingClassifier

- base_estimator : 사용될 수 있는 모델(default=None)
- n_estimators : int, optional(default=10), subset으로 생성되는 모델의 개수
- max_samples : int or float, optional(default=1.0), 최대 데이터 개수 또는 비율
- max_features : int or float, optional(default=1.0), 최대 사용 피쳐 또는 비율
- bootstrap : boolean, optional(default=True), bootstrap 사용 여부
- oob_score : boolean, oob score 산출 여부
- warm_start : booeanl, optional(default=False), 이전에 학습된 모델을 사용할 것인가에 대한 정보

Q&A