의사결정나무

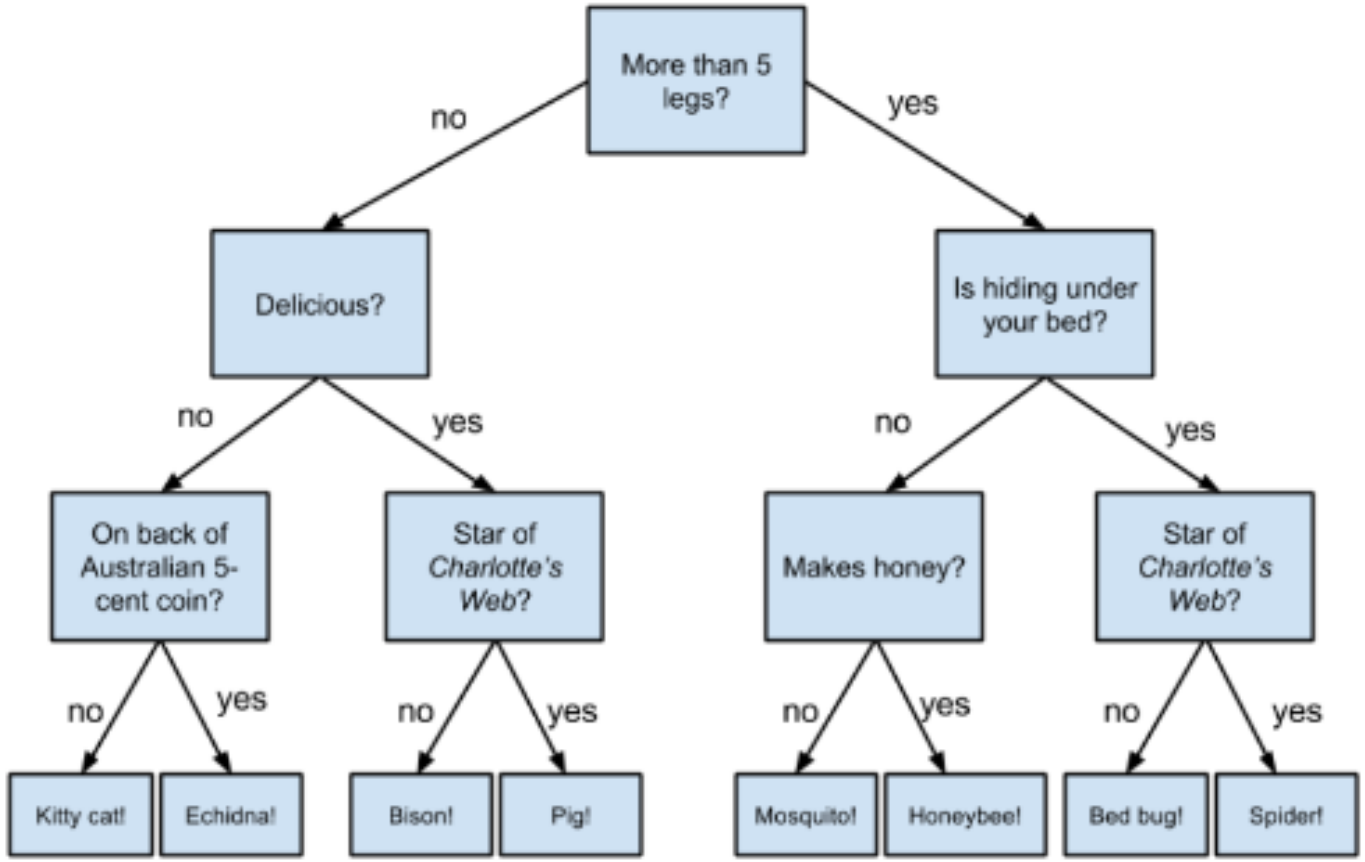
오동건

Growth Hackers

CONTENTS

Growth Hackers

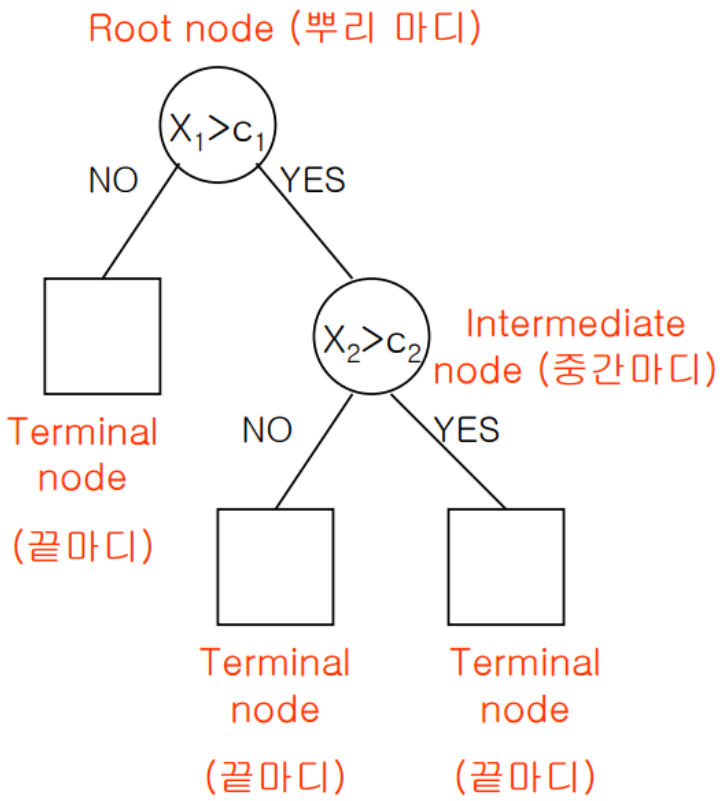
의사결정트리의 기초



스무고개를 떠올려 봅시다

Growth Hackers

의사결정트리의 기초



Root node: 시작 노드

Terminal node;잎: 최종 예측값

Internal node: 설명변수의 분기점

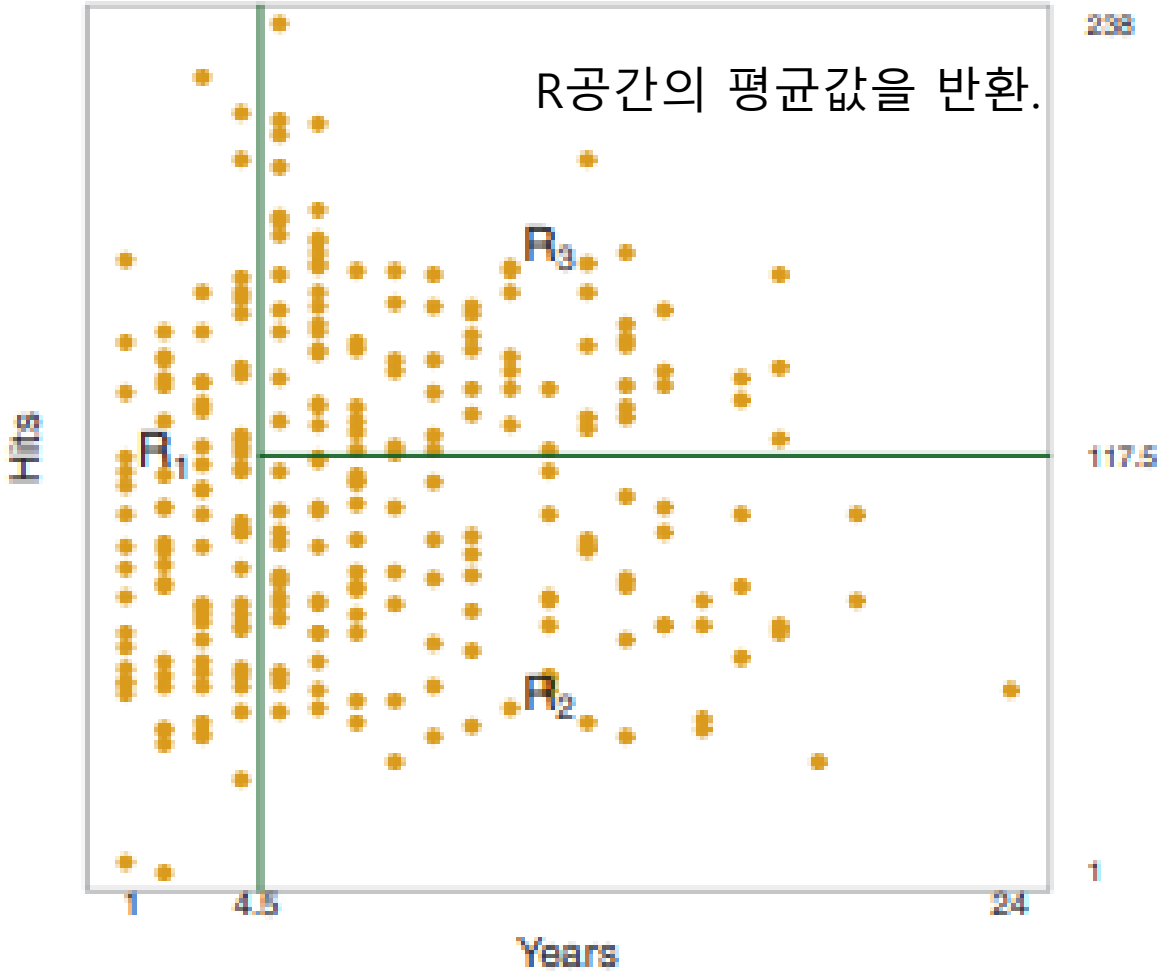
Branch: 가지

Tree size: 노드의 수

Tree depth: root node ~ terminal node까지의 깊이

Growth Hackers

회귀트리



$$\sum_{j=1}^{J} \sum_{i \in R_j} (y_i - \hat{y}_{R_j})^2,$$

회귀트리의 목적은 RSS를 최소화하는 공간 R(terminal node)의 분할을 찾는 것으로 시작

-> 선형에서는 정규방정식으로 찾을 수 있었지만,

트리에서는 어떻게??

Growth Hackers

회귀트리

R공간의 가능한 모든 분할을 계산하는 것은 실현 불가능!

(설명변수가 모두 질적이고 몇 개 안되면 모든 서브셋을 계산할 수 도 있겠지만…)

-> 재귀이진분할로 알려진 하향식의 **그리디(greedy)** 기법을 사용.

-> 나중은 고려하지 않고 즉각적으로 두 공간으로 분할했을 때, 최선의 결과가 나오도록 것

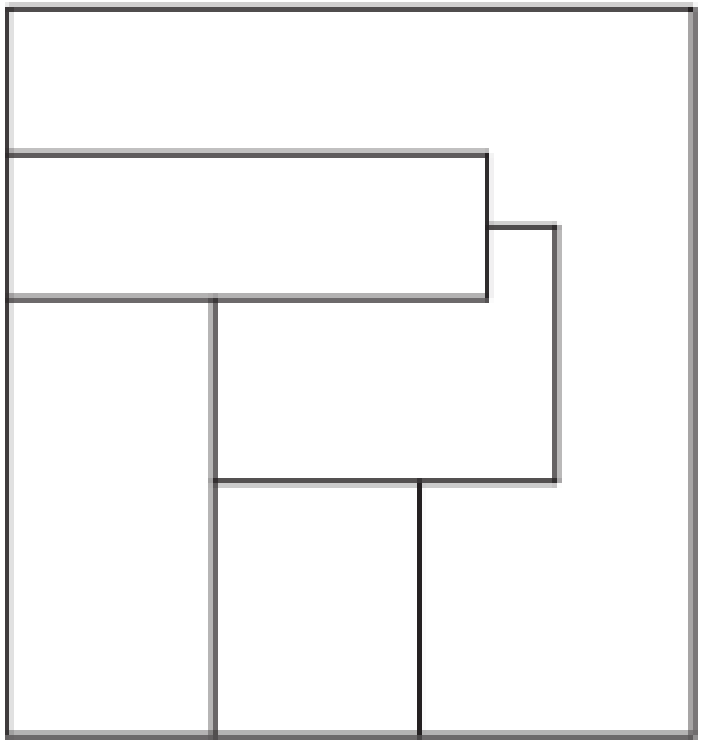
-> 최선의 결과: RSS가 가장 많이 줄어드는 것

-> RSS가 가장 많이 줄어드는 최선의 분할점과 설명변수를 선택

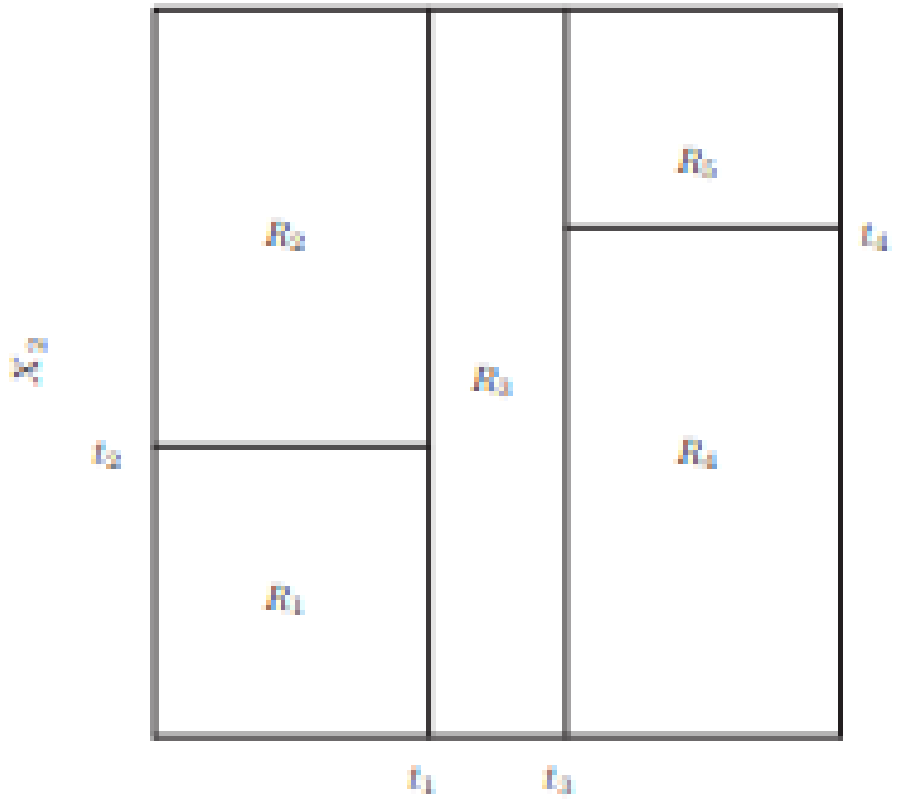
(모든 R의 분할을 고려하는 것에 비해 계산이 더 쉬워짐)

-> 이 과정을 반복

회귀트리



재귀이진분할 X



재귀이진분할 O

Growth Hackers

회귀트리

트리의 크기가 무한으로 가면 훈련 MSE는 0에 수렴.

Ex) 10명을 분류할 때, 잎이 10개면 오차없음.

그러나 과적합 가능성이 높아져서 검정MSE는 높을 수 있다.

깊은 트리는 높은 분산을 가짐.

대안: Pruning

-> 처음부터 작은 트리를 만들자

-> 잠깐! 그러나 뒤에 더 좋은 분할이 있을 수도 있다.

-> 트리를 아주 크게 만든 다음 가지를 치자!(Pruning)

회귀트리

가지를 어디서 치지? = 터미널 노드의 수(T)를 몇 개 남기지?

아래 식을 보면 lasso와 유사함을 알 수 있다.
-> 트리가 깊어짐에 따라서 패널티를 부과하고
-> 비용함수에 터미널 노드의 수를 반영함으로써 적절한 pruning을 할 수 있다.

$$\sum_{m=1}^{|T|} \sum_{i:\, x_i \in R_m} (y_i - \hat{y}_{R_m})^2 + \alpha |T|$$

*적절한 a를 모르더라도 cross validation을 통해 추정할 수 있다.

Growth Hackers

분류트리

회귀트리와 달리 비용함수의 차이가 있다.
RSS 대신에 지니지수 또는 엔트로피로 불순도를 측정.

- 집합 X에 두 종류의 원소가 있다.
  이 집합의 불순도는 얼마인가?

Ex) 흰 바둑알 W, 검정 바둑알 B
X = {B, B, B, B, W, W, W, W}
Y = {B, B, W, W, W, W, W, W}
Z = {B, B, B, B, B, B, B, B}
Z' = {B, W}

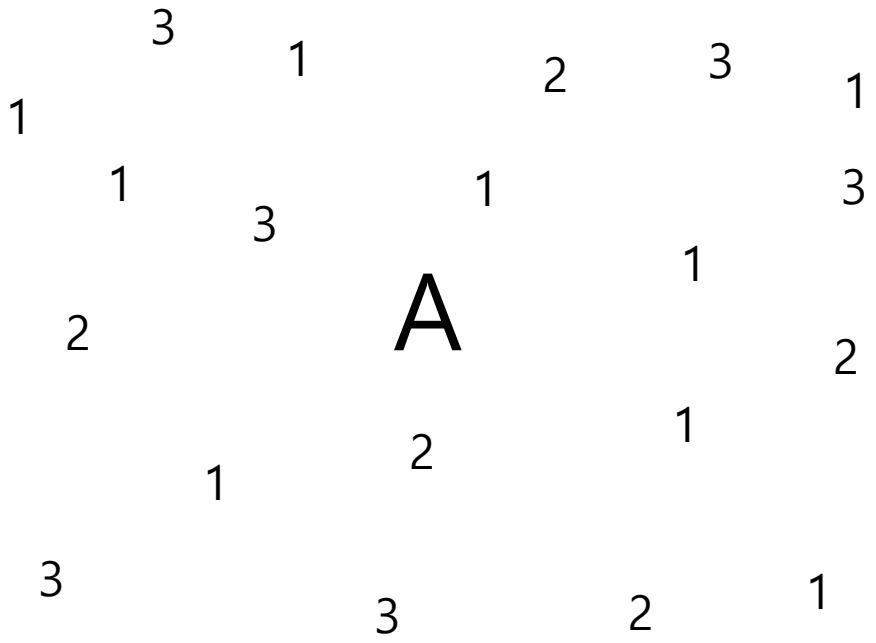
⇒ 불순도 측정하는 방법?

엔트로피

지니지수

Growth Hackers

분류트리



1 : 9 개
2 : 5 개
3 : 6 개
총 20 개

$$p_k = \frac{n_k}{20}$$ 하나의 터미널 노드에서 각 클라스의 비

$n_k = class\ k$ 에 속하는 원소의 개수

하나의 터미널 노드 A를 가정.

Growth Hackers

분류트리

지니 지수(weighted sum)

- Sckit-learn 의사결정나무 default option
- m개의 레코드를 지닌 end note A에 대한 지니 지수

$$I(A) = 1 - \sum_{k=1}^{m} p_k^2$$

- Q: 범주가 두 개일 때, 불순도를 최대로 만드는 비율은??

엔트로피(entropy)

- m개의 레코드를 지닌 end note A에 대한 엔트로피

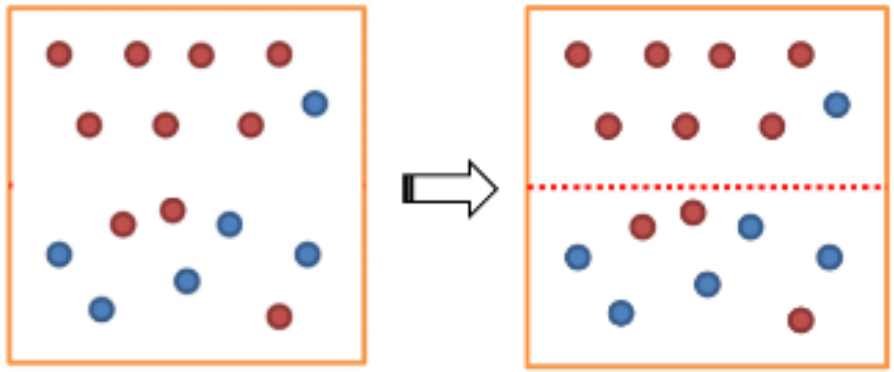
$$entropy(A) = -\sum_{k=1}^{m} p_k \log_2 p_k$$

- Q: 엔트로피가 언제 0이 될까요??

분류트리

정보 이득(information gain)

- 분할 후 불순도 측정



$$entropy(A) = -\sum_{k=1}^{m} p_k \log_2 p_k$$

Q: 분할 결과의 불순도를 엔트로피로 계산해볼까요?

A: $0.5 \times \left(-\frac{1}{8}\log_2\frac{1}{8} - \frac{7}{8}\log_2\frac{7}{8}\right) + 0.5 \times \left(-\frac{5}{8}\log_2\frac{5}{8} - \frac{3}{8}\log_2\frac{3}{8}\right)$

- **Information Gain** = 기존 불순도 – 분할 후 불순도

분류트리

지니 지수는 미분 불가능.. 엔트로피를 통해

알고리즘을 따라가봅시다 !

Outlook, Temp, Humidity, Windy 등의 날씨 조건을 보고 , 언제 테니스를 치는지에 대해 분류하는 예제

Play의 엔트로피 계산

$$E(Play) = \left(-\frac{5}{14}\log_2\frac{5}{14}\right)+\left(-\frac{9}{14}\log_2\frac{9}{14}\right)$$

$$E(Play) = 0.94$$

Outlook	Temperature	Humidity	Windy	Play?
sunny	hot	high	FALSE	No
sunny	hot	high	TRUE	No
overcast	hot	high	FALSE	Yes
rain	mild	high	FALSE	Yes
rain	cool	normal	FALSE	Yes
rain	cool	normal	TRUE	No
overcast	cool	normal	TRUE	Yes
sunny	mild	high	FALSE	No
sunny	cool	normal	FALSE	Yes
rain	mild	normal	FALSE	Yes
sunny	mild	normal	TRUE	Yes
overcast	mild	high	TRUE	Yes
overcast	hot	normal	FALSE	Yes
rain	mild	high	TRUE	No

분류트리

각 클래스로 분류 시 엔트로피는 ?

E(Play, Outlook)

		Play		
		Yes	No	Total
Outlook	sunny	3	2	5
	overcast	4	0	4
	rain	3	2	5

$$E(Play,\ Outlook) = p(sunny) \times E(3,2) + p(rainy) \times E(3,2) + p(overcast) \times E(4,0)$$

$$E(Play,\ Outlook) = \frac{5}{14} \times \left[\left(-\frac{3}{5} \times \log_2 \frac{3}{5}\right) + \left(-\frac{2}{5} \times \log_2 \frac{2}{5}\right) \right] + \cdots\cdots$$

$$E(Play,\ Outlook) = 0.36 \times 0.971 + 0.36 \times 0.971 + 0$$

$$E(Play,\ Outlook) = 0.6935$$

E(Play, Humidity)

		Play		
		Yes	No	Total
Humidity	high	3	4	7
	normal	6	1	7

$$E(Play,\ Humidity) = p(high) \times E(3,4) + p(normal) \times E(6,1)$$

$$E(Play,\ Humidity) = 0.7884$$

E(Play,Temperature)

		Play		
		Yes	No	Total
Temp..	hot	2	2	4
	mild	4	2	6
	cool	3	1	4

$$E(Play,\ Temp) = p(hot) \times E(2,2) + p(mild) \times E(4,2) + p(cool) \times E(3,1)$$

$$E(Play,\ Temp) = \frac{4}{14} \times \left[\left(-\frac{1}{2} \times \log_2 \frac{1}{2}\right) + \left(-\frac{1}{2} \times \log_2 \frac{1}{2}\right) \right] + \cdots\cdots$$

$$E(Play,\ Temp) = 0.2857 + 0.3935 + 0.2317$$

$$E(Play,\ Temp) = 0.911$$

E(Play, Windy)

		Play		
		Yes	No	Total
Windy	TRUE	3	3	6
	FALSE	6	2	8

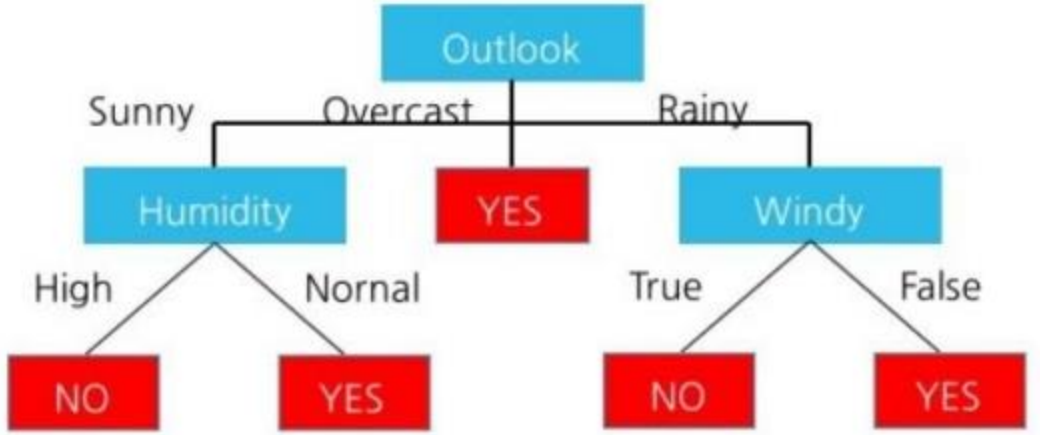
$$E(Play,\ Windy) = p(True) \times E(3,3) + p(False) \times E(6,2)$$

$$E(Play,\ Windy) = 0.8921$$

분류트리

분류 시 **INFORMATION GAIN** 이 가장 큰 클래스는 ?

$$E(Play) - E(Play, Outlook) = 0.25$$
$$E(Play) - E(Play, Temp) = 0.02$$
$$E(Play) - E(Play, Humidity) = 0.1514$$
$$E(Play) - E(Play, Windy) = 0.047$$



Growth Hackers

Decision Tree

- **의사결정나무의 장점**
    1. 규칙을 이해하기 쉽다
    2. 연속형 , 범주형 변수 모두 쉽게 처리된다
    3. 중요한 변수를 찾아준다
    4. 입력변수의 이상치에 강건 (robust) 하다

- **의사결정나무의 단점**
    1. 회귀모형에서 예측력이 떨어진다
    2. 나무의 크기가 크면 해석력이 떨어진다
    3. 불안정하다

앙상블을 사용하면

예측 성능 향상!!

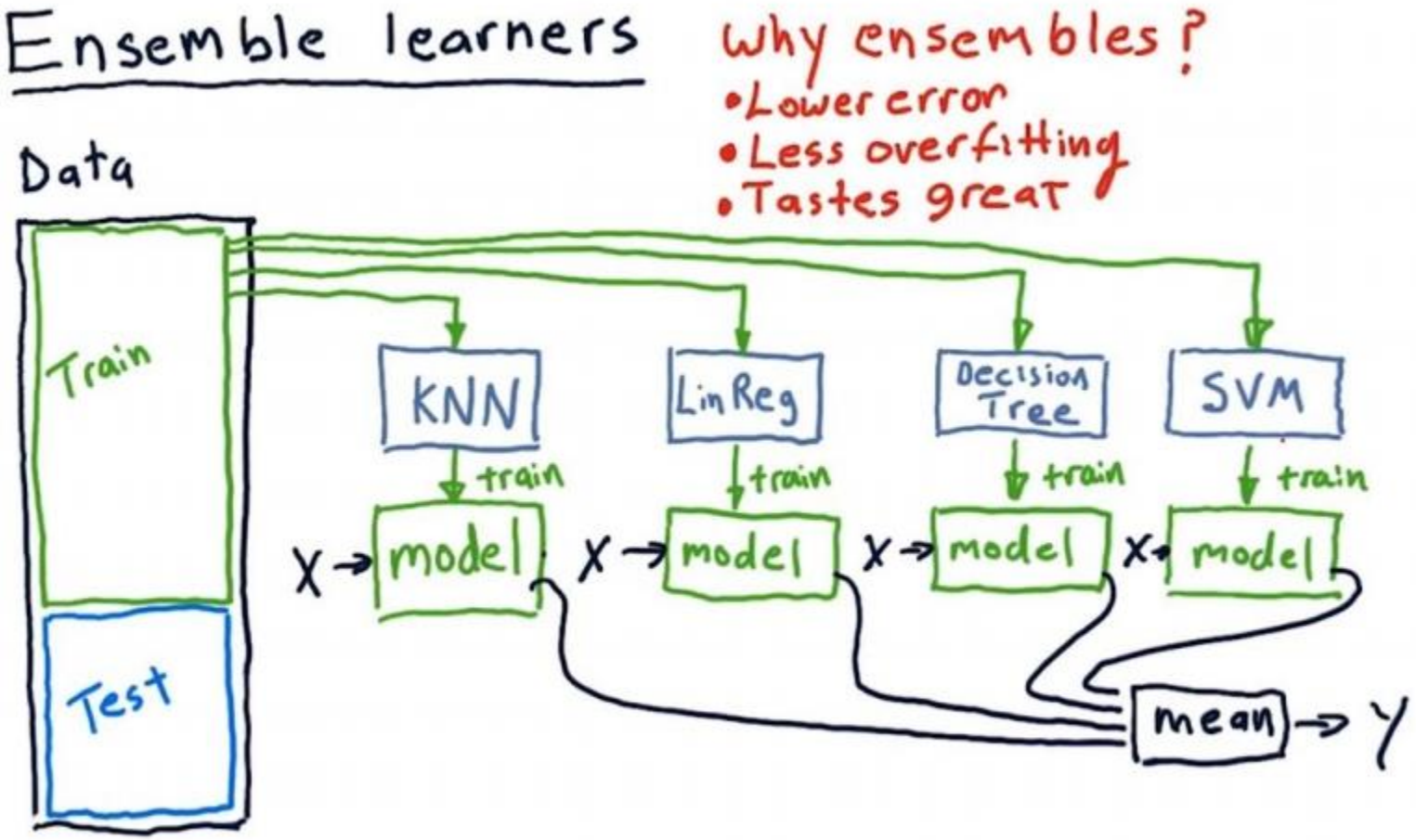
Ensemble

- **앙상블** 이란 하나의 자료에 대해서 여러 개의 예측모형을 만든 후, 이를 결합하여 최종 예측모형을 만드는 방법을 말한다

- 앙상블 방법의 예
  : **Bagging, Boosting**

- 실증적으로 앙상블 방법이 의사결정나무보다 훨씬 좋은 예측력을 갖는 것이 밝혀졌다

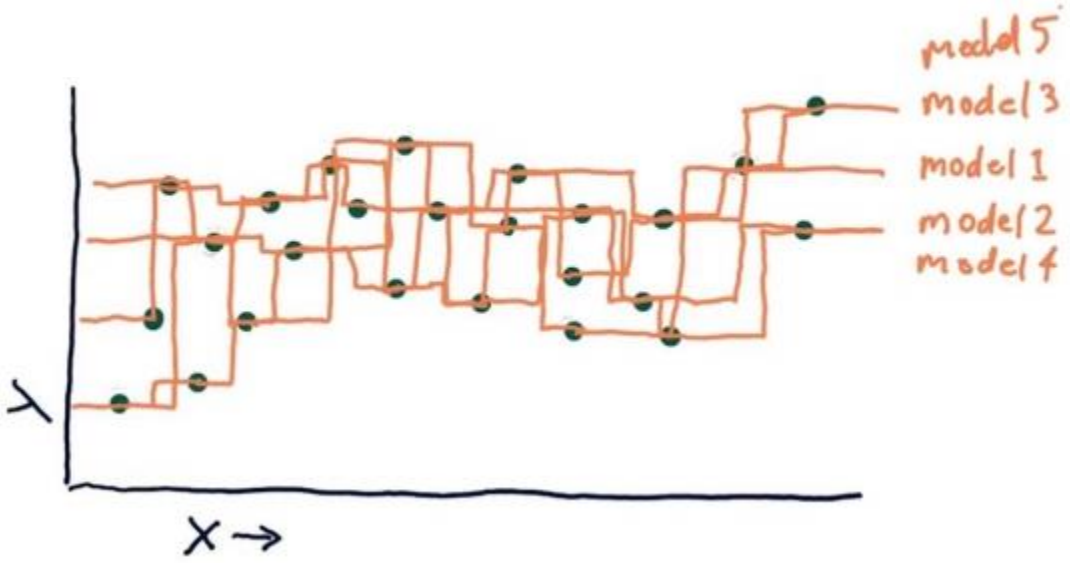
Ensemble

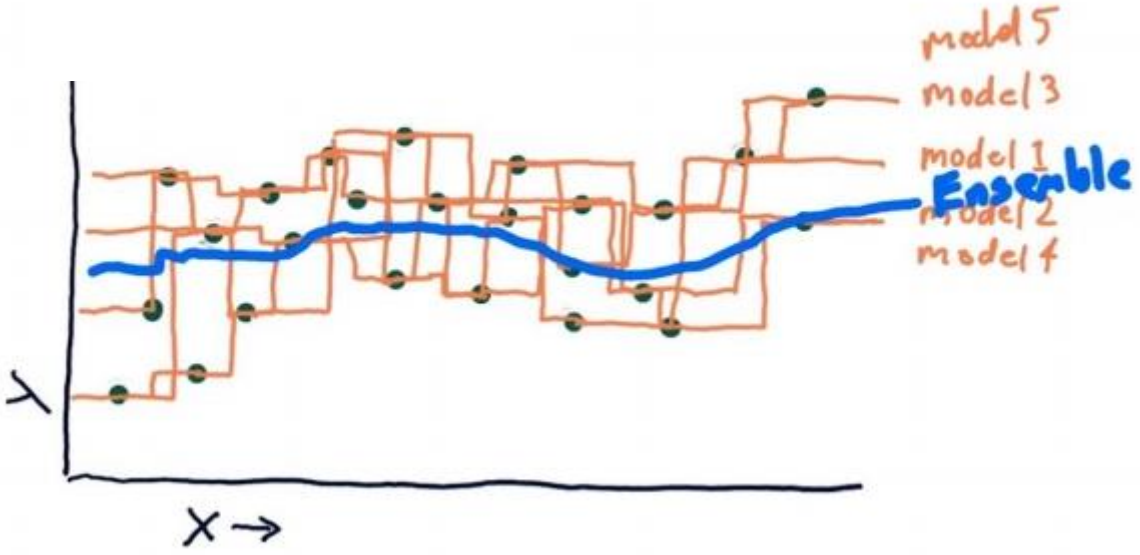


- **Error 최소화**
  단일 모형으로 분석했을 때보다 신뢰성 높은 예측값을 얻음
- **Variance 감소**
  여러 모형의 의견을 결합하여 평균 내면 variance 가 작아짐
- **Overfitting 감소**
  과적합의 가능성을 줄여줌

Growth Hackers

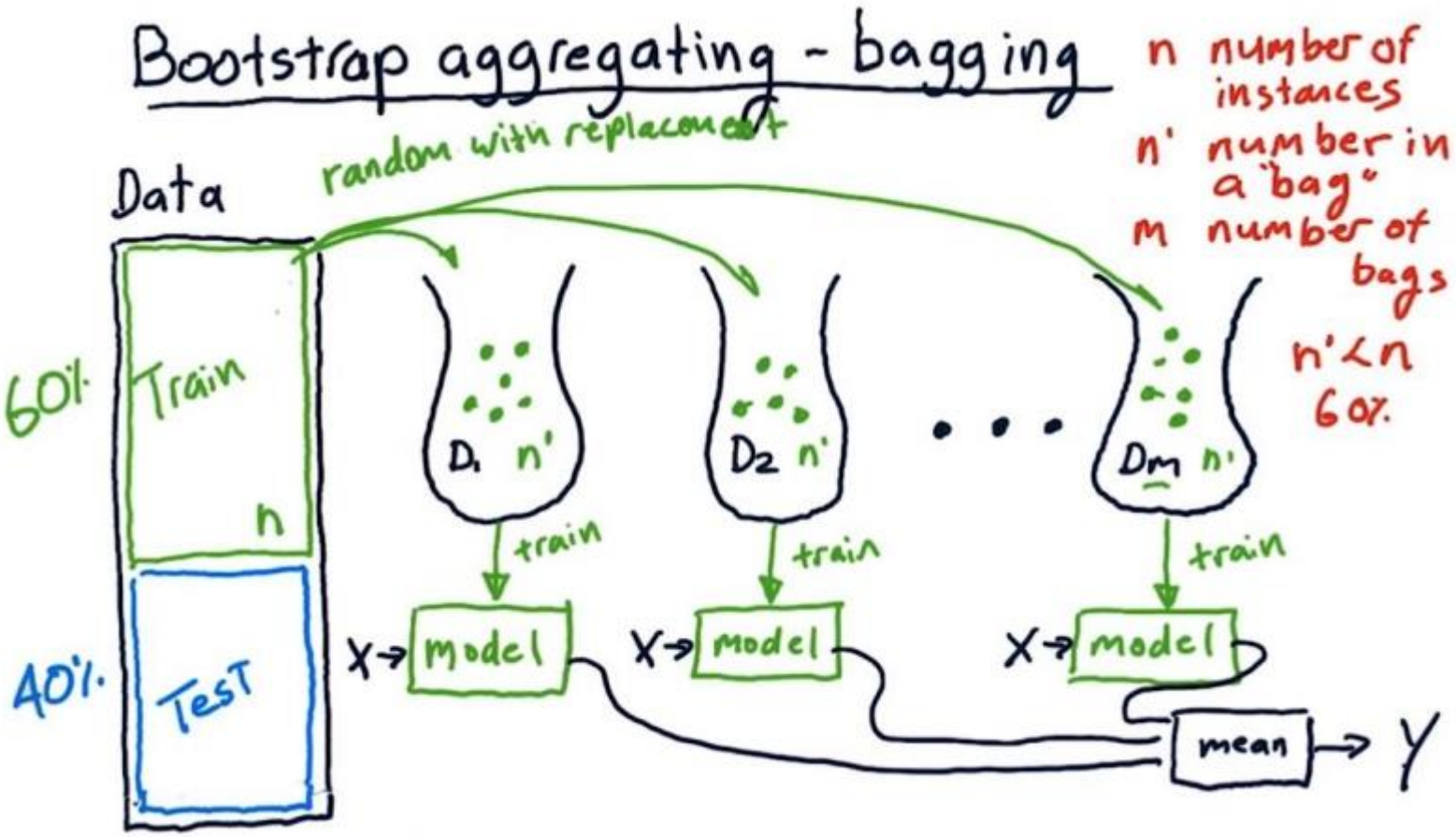
Ensemble





Growth Hackers

Bagging

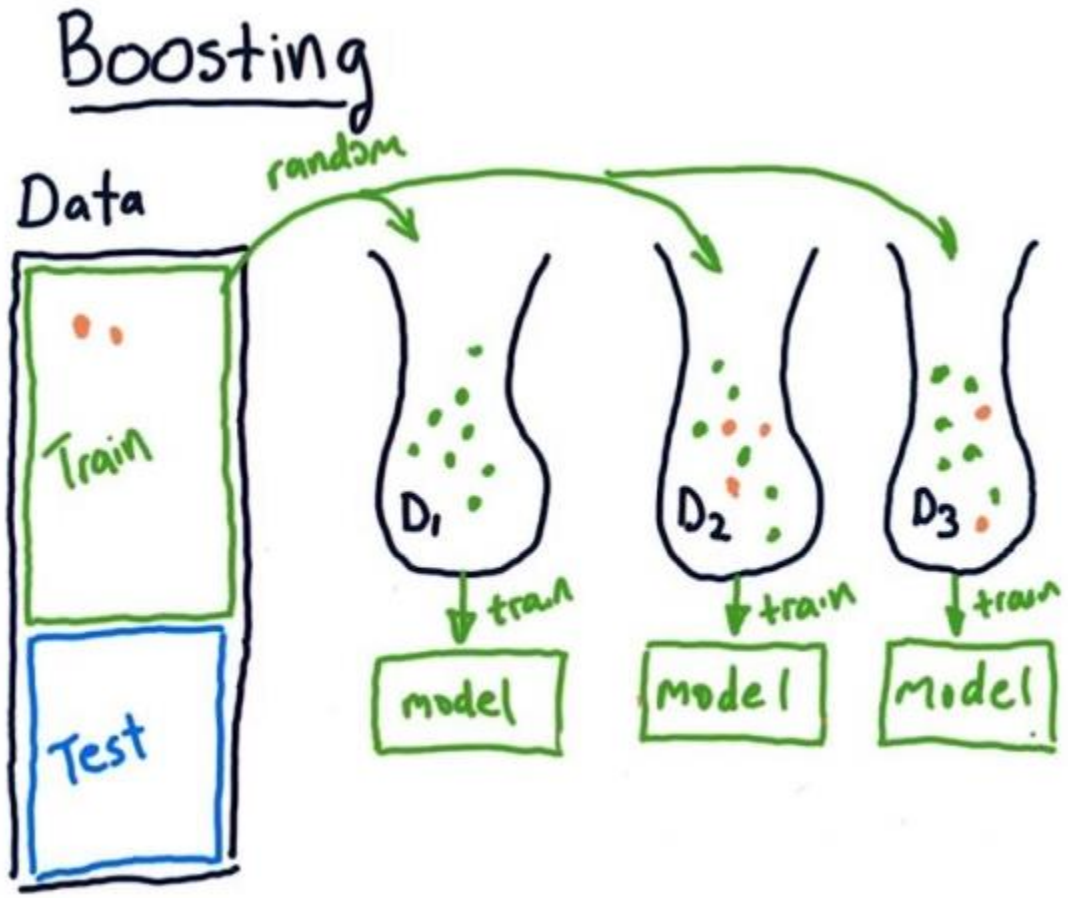


- **배깅** 은 bootstrap 을 사용하여 원래 훈련 자료에 대한 다수의 복사본을 만들고 , 각 복사본에 별도의 의사결정트리를 적합하며 그 다음에 이 트리들을 모두 결합 , 단일 예측 모형을 만든다
- **의사결정트리의 높은 분산 해결**
  : Bootstrap 된 훈련 데이터에 대한 예측 결과를 평균하여 얻으면 분산 감소

$$\hat{f}_{\text{bag}}(x) = \frac{1}{B}\sum_{b=1}^{B} \hat{f}^{*b}(x).$$

Growth Hackers

Boosting



- 부스팅:
  트리들이 순차적 으로 만들어진다
- 각 트리는 이전에 만들어진 트리들로부터의 정보를 사용하여 만들어진다 . 모델 적합 후 잔차 를 다음 모델의 반응변수 로 놓고 새로 적합 .
- 결과 Y 가 아니라 현재의 잔차들을 반응변수로 적합 -> 작은 트리들이 이 잔차들에 적합함으로 , 좋은 성능을 내지 못하는 영역을 천천히 개선한다

Growth Hackers

Bagging VS Boosting

Bagging	Boosting
다수의 boostrap 샘플	하나의 샘플
각각의 샘플을 적합한 후 평균	잔차를 종속변수로 두고 적합을 반복.
Random forest ($\sqrt{p}$ 개의 설명변수를 사용. 배깅은 p개)	Xgboost, LGBM

References

- GH 2 기 남정우 , 이영준 , 차유경 - 의사결정나무 세션 자료
- DECISION TREE 알고리즘 예제 http://jihoonlee.tistory.com/16
- BOOSTING 기법의 이해 www.slideshare.net/freepsw/boosting-bagging-vs-boosting
- ENSENBLE https://www.youtube.com/watch?v=Un9zObFjBH0&t=9s
- BAGGING https://www.youtube.com/watch?v=2Mg8QD0F1dQ&t=35s
- BAGGING EXAMPLE https://www.youtube.com/watch?v=sVriC_Ys2cw
- BOOSTING https://www.youtube.com/watch?v=GM3CDQfQ4sw
- Introduction to Statistical Learning https://www-bcf.usc.edu/~gareth/ISL/ISLR%20First%20Printing.pdf
- RANDOMFOREST PYTHON PRACTICE https://partrita.github.io/posts/random-forest-python/
- XGBOOST https://brunch.co.kr/@snobberys/137

Growth Hackers