

Ensemble Methods

5기 허유진

목차

- ▶ 01 앙상블 모델 등장 배경
- ▶ 02 Bagging
- ▶ 03 Boosting
- ▶ 04 Random Forest

01 앙상블 모델 등장 배경

Ensemble Method(앙상블)이란?

통계학과 기계학습에서 앙상블 학습법(Ensemble Learning Method)이란, 학습 알고리즘을 따로 사용하는 경우에 비해 더 좋은 예측 성능을 얻기 위해 다수의 학습 알고리즘을 사용하는 방법이다.

즉, 여러 개의 분류기를 생성하고, 그 예측을 결합하여 보다 정확한 예측을 도출하는 기법.

Machine Learning에서 예측력과 정확도를 높이기 위한 방법으로 사용된다. (해석, 설명력 ↓)

앙상블 등장 배경

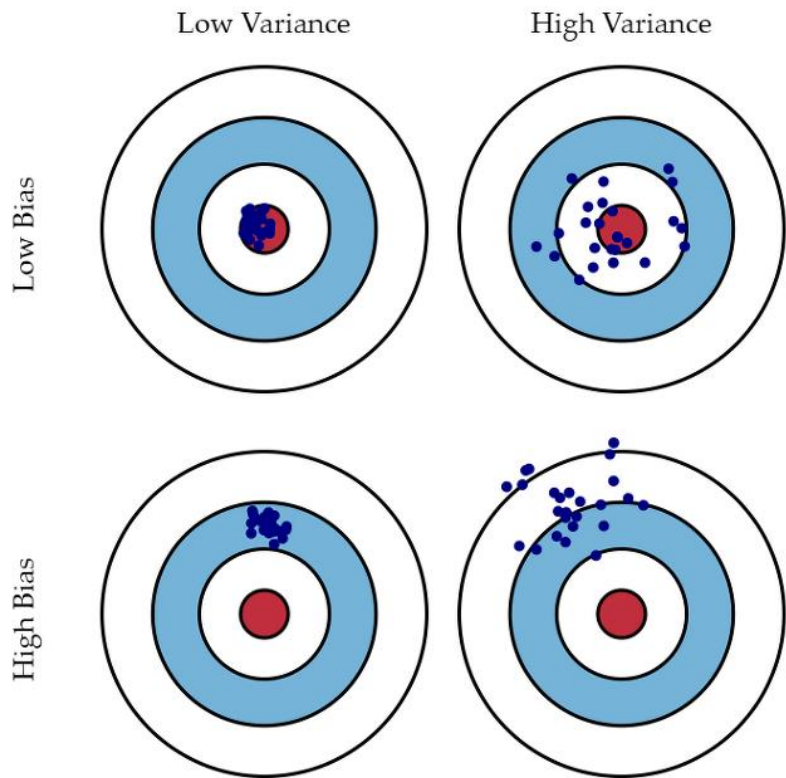


Fig. 1 Graphical illustration of bias and variance.

편향과 분산에 대해 알아야 합니다.

편향 ↓ 분산 ↑ : ex) decision tree

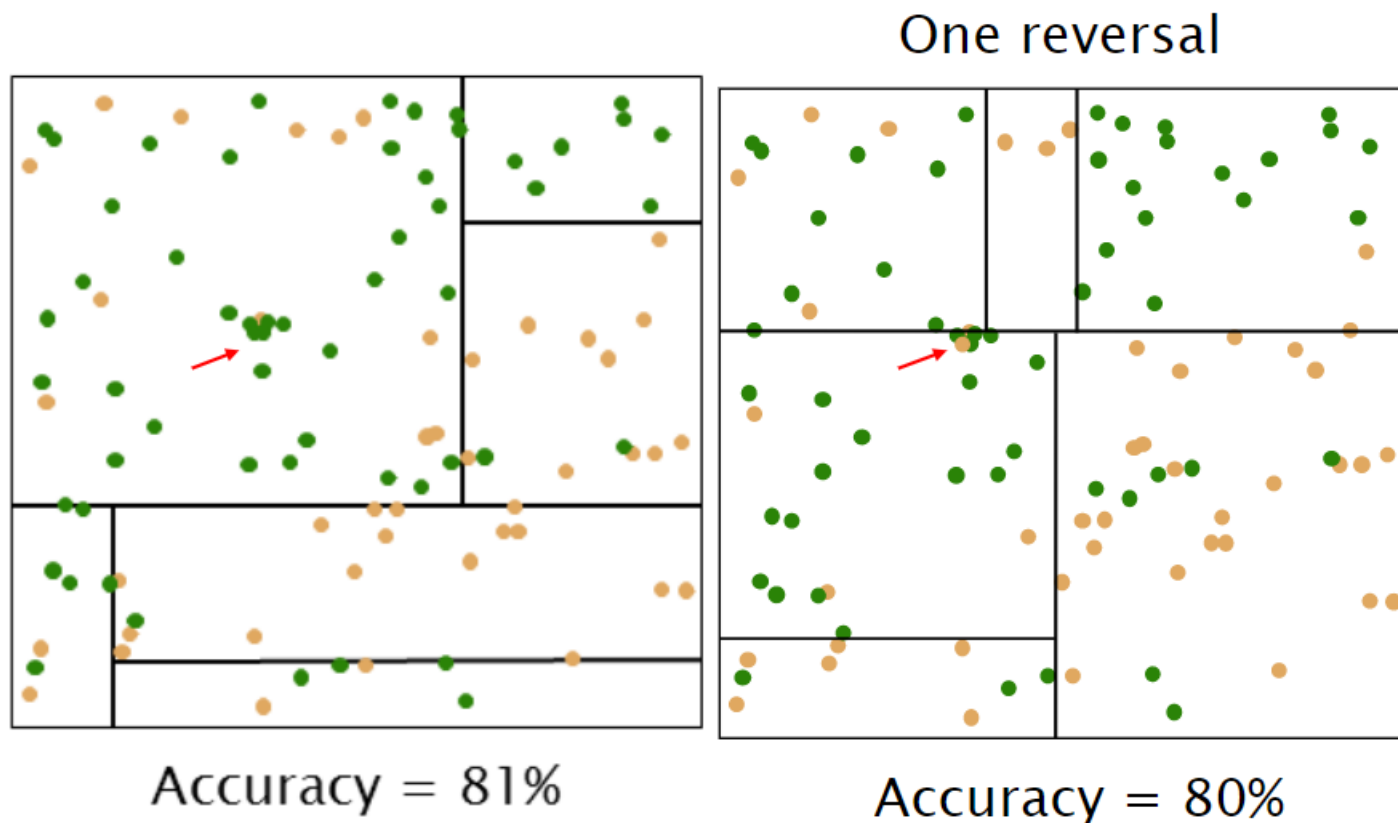
편향 ↑ 분산 ↓ : ex) logistic regression

편향 ↓ 분산 ↓ : 예측 정확도 측면에서 가장 좋다.

앙상블: 의사결정나무의 **불안정성**을 장점으로 만듭니다.
B를 평균 낸 게 A의 점 한 개.

Decision tree를 여러 개 만들어서 평균 낸 것을 하나의 모델로 사용하면 되겠다.

Decision Tree의 불안정성



유사도가 99%임에도 불구하고 완전 다른 결과가 나타난다. (**불안정성**)

왜 그럴까?

뿌리 노드를 분할할 때 greedy search를 하기 때문에 1,2등이 뒤바뀔 수 있다.
(모든 경우를 다 search 하여 불순도를 가장 많이 낮추는 것 선택)

유사도 99%, 딱 한 개의 점만 다름
X변수 2개, Y변수는 점의 색깔, 영역 내 다수결 분류

앙상블 등장 배경

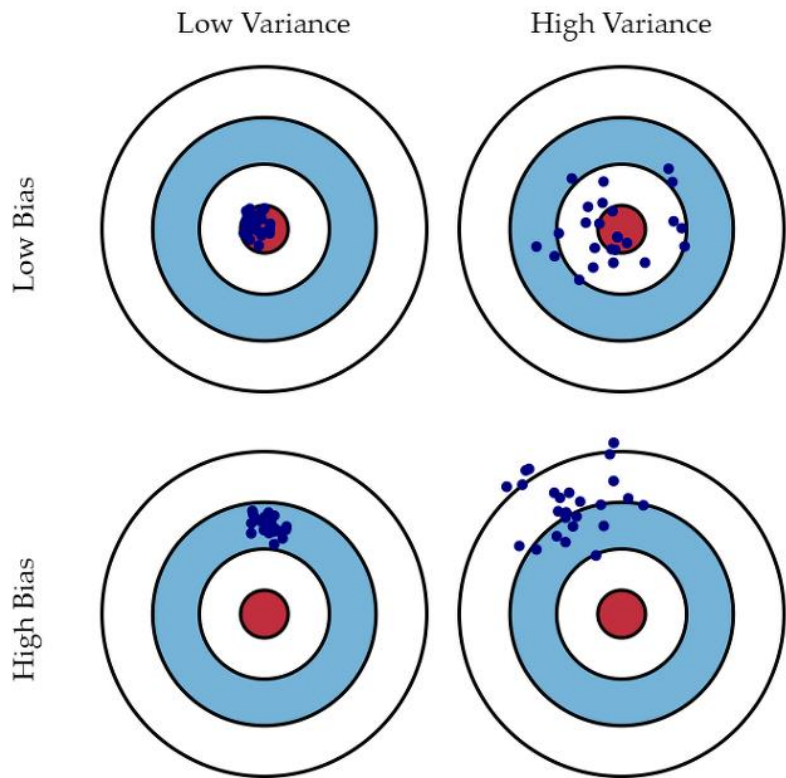


Fig. 1 Graphical illustration of bias and variance.

B 모델

불안정적이다 = 분산이 크다

이러한 모델에 앙상블을 적용하면 좋겠다!

불안정성은 Ensemble Technique의 필수 요소이다.

지금부터 Decision Tree를 이용하여 앙상블을 해보자.

02 Bagging

Bootstrapping

나에게는 original dataset 1개 밖에 없다. 이것로 여러 개의 Decision Tree를 만들어야 한다! 어떻게 만들까?

Bootstrapping!

= 복원 추출

x^1	y^1
x^2	y^2
x^3	y^3
x^4	y^4
x^5	y^5
x^6	y^6
x^7	y^7
x^8	y^8
x^9	y^9
x^{10}	y_0^1

Original Dataset
Y 변수는 0과 1 Binary

Bootstrap 1

x^3	y^3
x^6	y^6
x^2	y^2
x^{10}	y_0^1
x^8	y^8
x^7	y^7
x^7	y^7
x^3	y^3
x^2	y^2
x^7	y^7

Bootstrap 2

x^7	y^7
x^1	y^1
x^{10}	y_0^1
x^1	y^1
x^8	y^8
x^6	y^6
x^2	y^2
x^6	y^6
x^4	y^4
x^9	y^9

...

Bootstrap B

x^9	y^9
x^5	y^5
x^2	y^2
x^4	y^4
x^7	y^7
x^2	y^2
x^5	y^5
x^{10}	y_0^1
x^8	y^8
x^2	y^2

데이터의 개수는 동일. Random하게 복원추출 하니까 원래 데이터와 같지는 않다.
이 데이터를 이용하여 서로 다른 Tree를 만들면 되겠다.

Bootstrapping

결과 취합은 다수결로!

Bootstrap 1		Bootstrap 2			Bootstrap B	
x^3	y^3	x^7	y^7		x^9	y^9
x^6	y^6	x^1	y^1		x^5	y^5
x^2	y^2	x^{10}	y^1_0		x^2	y^2
x^{10}	y^1_0	x^1	y^1		x^4	y^4
x^8	y^8	x^8	y^8	...	x^7	y^7
x^7	y^7	x^6	y^6		x^2	y^2
x^7	y^7	x^2	y^2		x^5	y^5
x^3	y^3	x^6	y^6		x^{10}	y^1_0
x^2	y^2	x^4	y^4		x^8	y^8
x^7	y^7	x^9	y^9		x^2	y^2
1		0		...		0

평균을 구했더니 0.6 => cutoff가 0.5일 때 최종 predicted class label은 1로 예측

Bootstrapping

예를 들어,

Bootstrap의 개수가 8인 경우 -> Decision Tree가 8개 만들어짐.

Classification 결과: {1, 0, 1, 0, 1, 1, 1, 1}이면 최종 앙상블 결과는 1

Bagging

Bagging = 배깅 = Bootstrapping Aggregating

- 전체 데이터 집합에서 복원추출을 통해 원래 데이터 수만큼의 크기를 갖도록 새로운 데이터셋 생성
- 이렇게 복원추출된 데이터셋이 bootstrap
- 각 bootstrap을 이용하여 의사결정나무를 만들기
- 최종 예측은 각 의사결정나무의 예측 결과를 취합하여 생성
- 서로 다른 decision tree가 만들어지므로 **다양성을 확보**할 수 있다.

Bagging

Bootstrap sample training set: $T^{(1)}, T^{(2)}, \dots, T^{(B)}$

Classifiers: $C(x, T^{(1)}), \dots, C(x, T^{(B)})$

Let N_j be the number of times that classified as j

$$N_j = \sum_{b=1}^B I[C(x, T^{(b)}) = j] \quad \text{for } j \in \{1, \dots, J\}$$

Bagging: $C_B(x) = \operatorname{argmax}_j N_j$

Bagging uses unweighted majority voting. 즉, 다수결 투표를 한다.

Bagging reduce the variance. 분산을 감소시킨다.

Bagging은 분류와 회귀에 모두 사용 가능하다. 회귀에서는 Regression Tree를 이용하면 된다.

Bagging의 장점과 단점

Bagging의 장점: 예측력이 우수하다

Bagging의 문제점: 해석력이 약해진다

앙상블의 목적: 다수의 모델을 학습하여 오류의 감소를 추구

오류의 감소란? 분산과 편향의 감소

- 분산의 감소에 의한 오류의 감소: Bagging, Random Forest
- 편향의 감소에 의한 오류 감소: Boosting

이때, 다수의 모델 간에는 다양성이 있어야 한다. 똑같은 모델을 여러 개 사용하는 것은 아무런 효과가 없다.
다양성을 어떻게 확보하는가?

Bagging에서는 Bootstrapping을 이용하여 다양성 확보.

03 Boosting

Boosting

Boosting의 기본 idea

1. Train data로 decision tree를 만들 때, 분류가 어려운 data의 가중치를 높여서 의사결정나무를 만들면 조금 더 정교한 tree가 나오지 않을까?!
2. Tree 여러 개를 통합할 건데, equal weight가 아니라 가중치를 다르게 줘서 통합한다. 더욱 정교한 tree의 가중치를 높인다.

Bagging과 비교

Bagging: Tree를 여러 개 만들기 위해 data에 변형을 가한다. Tree들은 모두 독립적.

Boosting: Tree를 여러 개 만들기 위해 data의 가중치를 변형한다. Tree들은 직전 결과에 영향을 받으므로 독립적이지 않다.

Boosting

Boosting의 방법

1. Iterative search를 통해 예측하기 어려운 regions를 찾는다.
2. 각 iteration마다 오분류 가중치를 두어 오류를 개선해 나가며 학습한다.
3. 모든 iteration을 통합한다.

Boosting의 알고리즘

1. Adaboost
2. Gradient Boost
3. XGBoost
4. LightGBM
5. Catboost

Adaboost

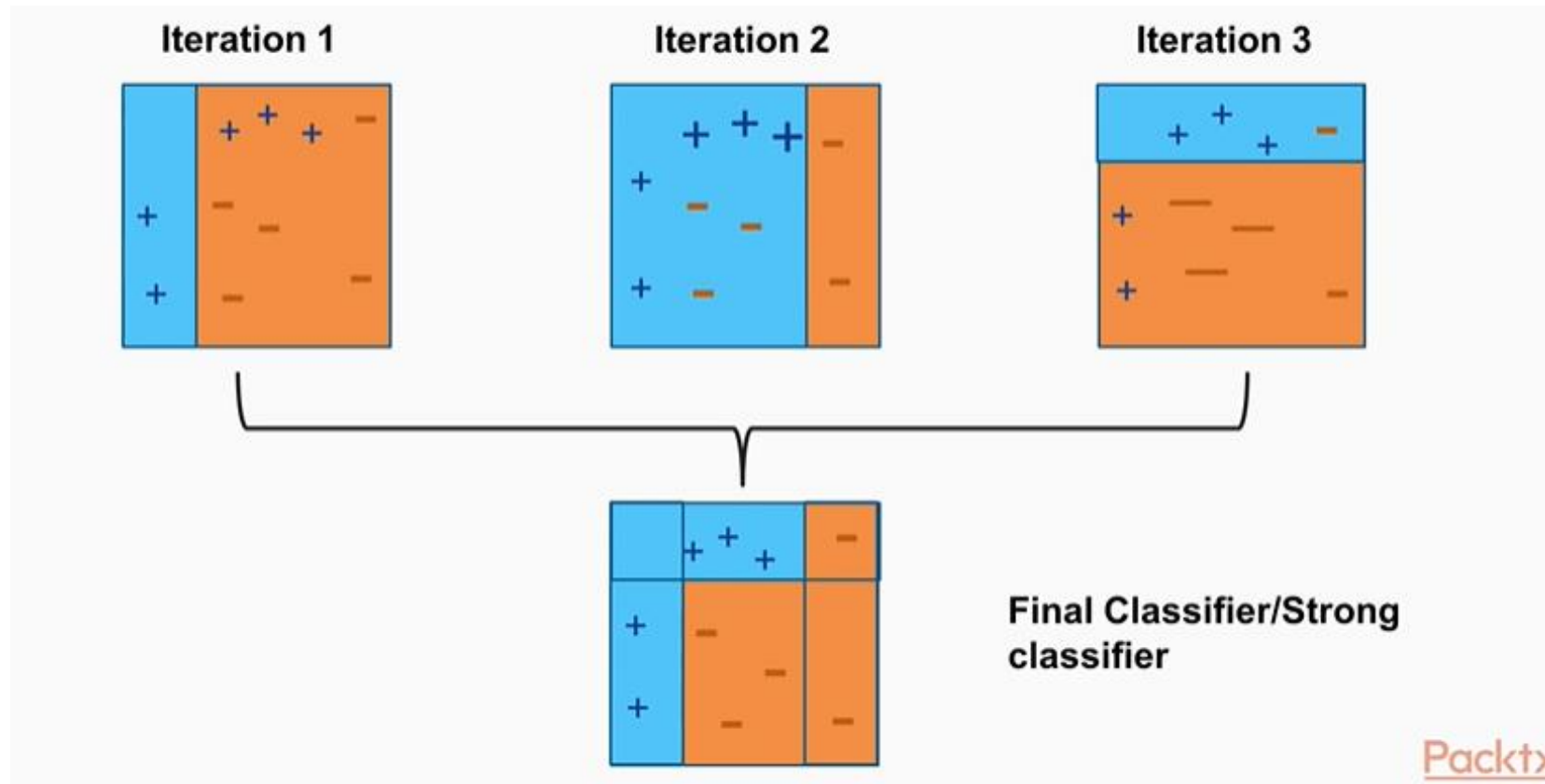
Adaboost(Adaptive Boost)는 편향과 분산을 모두 줄이는 boosting 방법이다.

- Train a classifier f_1 on data, then train a f_2 to correct the errors of f_1 , then train a f_3 to correct the errors of f_2 , etc
- Classification on unequal weighted training data
- Put unequal weights on observations so that misclassified observation have higher weight
- Reduce the variance: Tree 여러 개를 통합하여 평균내기 때문에 분산이 줄어든다.
- Reduce the bias: Tree가 점점 진화하기 때문

Adaboost

Adaboost(Adaptive Boost)

약한 학습기의 오류 데이터에 가중치를 부여하면서 boosting을 수행. 약한 학습기로 decision tree를 사용한다.



Adaboost

1. Initialize the weight $w_i = \frac{1}{N}$, $i = 1, 2, \dots, N$

2. For $m = 1$ to M :

1) Fit a classifier $G_m(x)$ to T using weight w_i

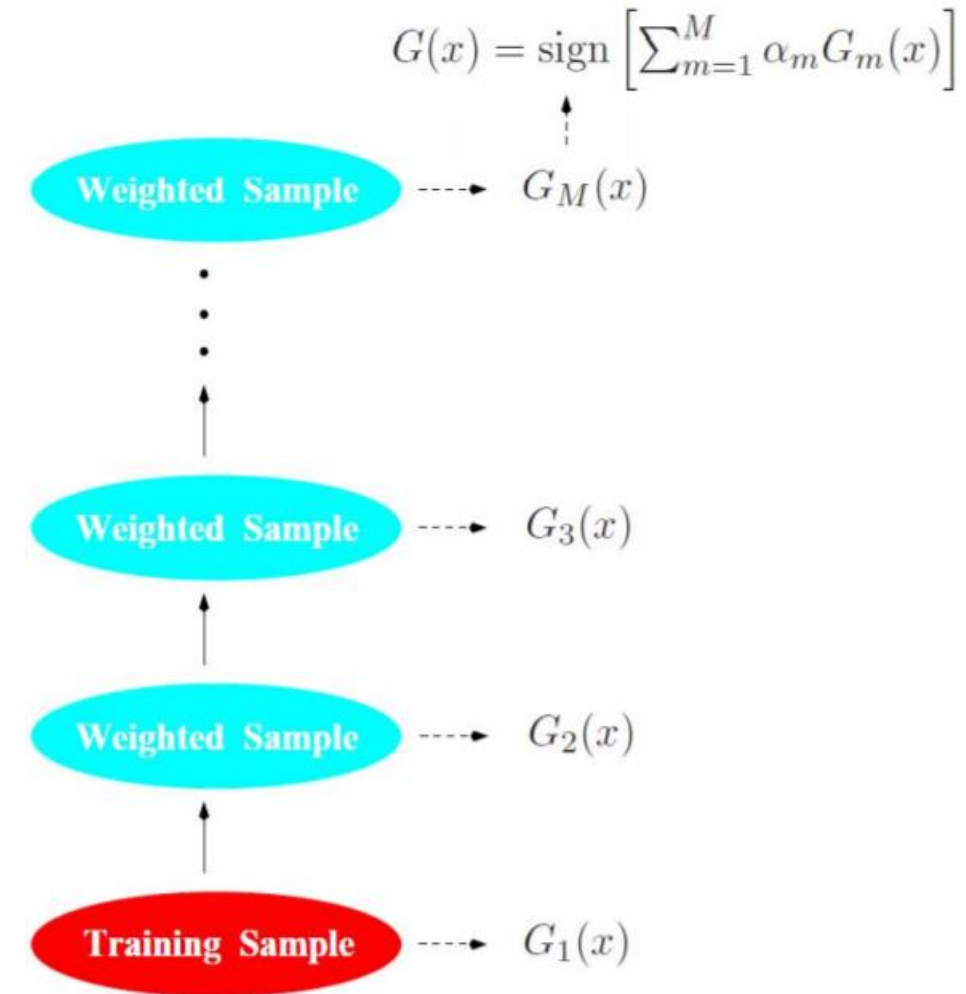
2) Compute

$$err_m = \frac{\sum_{i=1}^N w_i I(y_i \neq G_m(x_i))}{\sum_{i=1}^N w_i}$$

3) Compute $\alpha_m = \log\left(\frac{1-err_m}{err_m}\right)$

4) Set $w_i = w_i \times \exp[\alpha_m I(y_i \neq G_m(x_i))]$, $i = 1, 2, \dots, N$

3. Output $G_{Ad}(x) = \text{sign}[\sum_{m=1}^M \alpha_m G_m(x)]$



Gradient Boosting

Gradient Boosting

Adaboost의 기본 idea와 동일. 다만, weight을 update할 때 “Gradient Descent Algorithm”을 이용하여 최적 weight을 계산한다는 점에서 Adaboost를 보완.

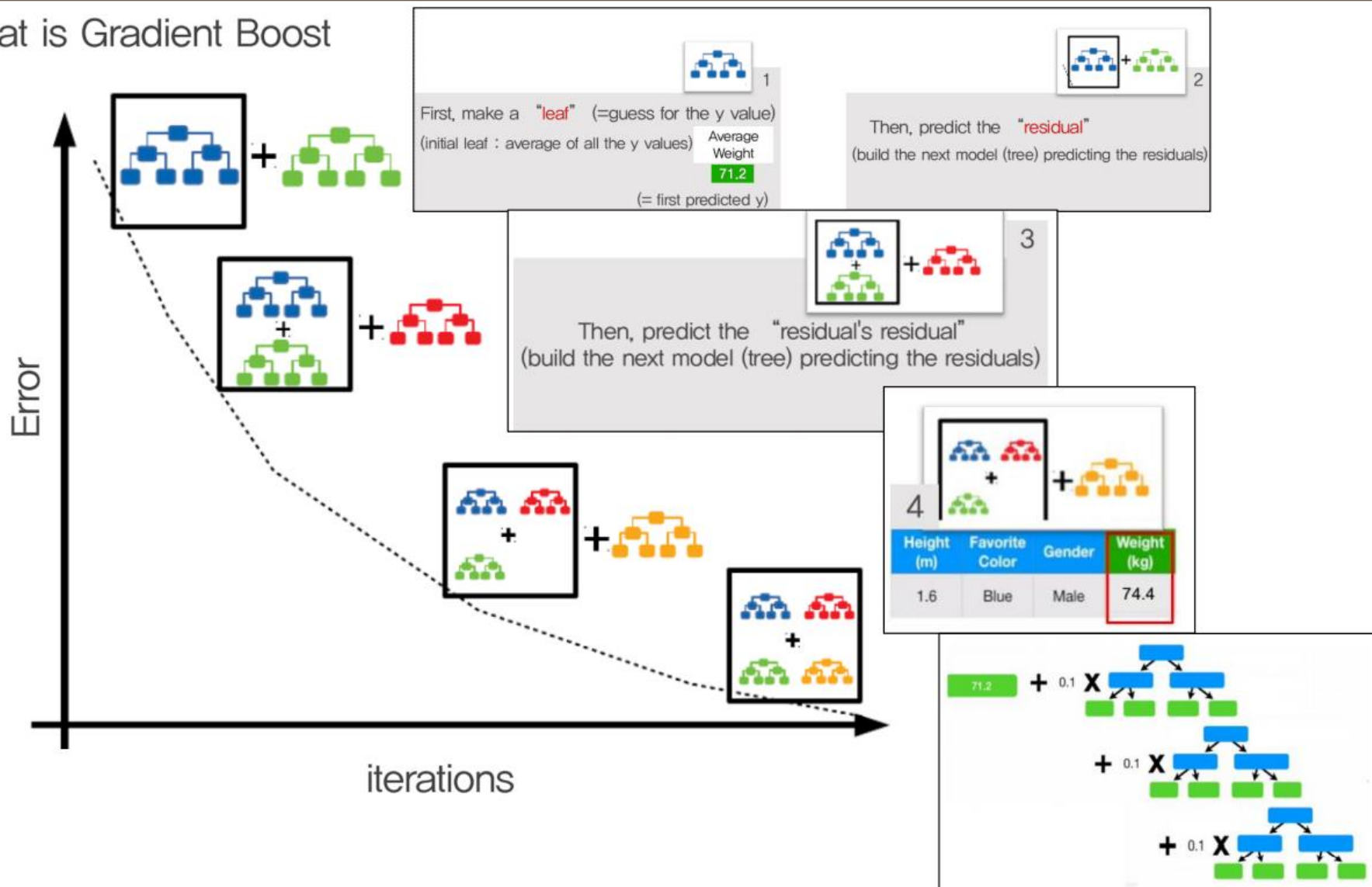
기본 idea: 직전 단계 tree의 error (오분류된 관찰값)을 y 라고 생각하고, 이것을 설명하는 tree를 또 만들자.

- $Y = h_1(x) + err1$
- $err1 = h_2(x) + err2$
- $err2 = h_3(x) + err3$
- $Y = h_1(x) + h_2(x) + h_3(x) + err3$
- $\hat{Y} = w_1 h_1(x) + w_2 h_2(x) + w_3 h_3(x) + \dots + M$

$$F_m(x) = F_{m-1}(x) + w_m h_m(x), \quad m = 1, \dots, M (\text{앙상블 크기})$$

Gradient Boosting

What is Gradient Boost



04 Random Forest

Random Forest

Random Forest (랜덤 포레스트)

2001년 Leo Breiman이 랜덤 포레스트 개념을 도입

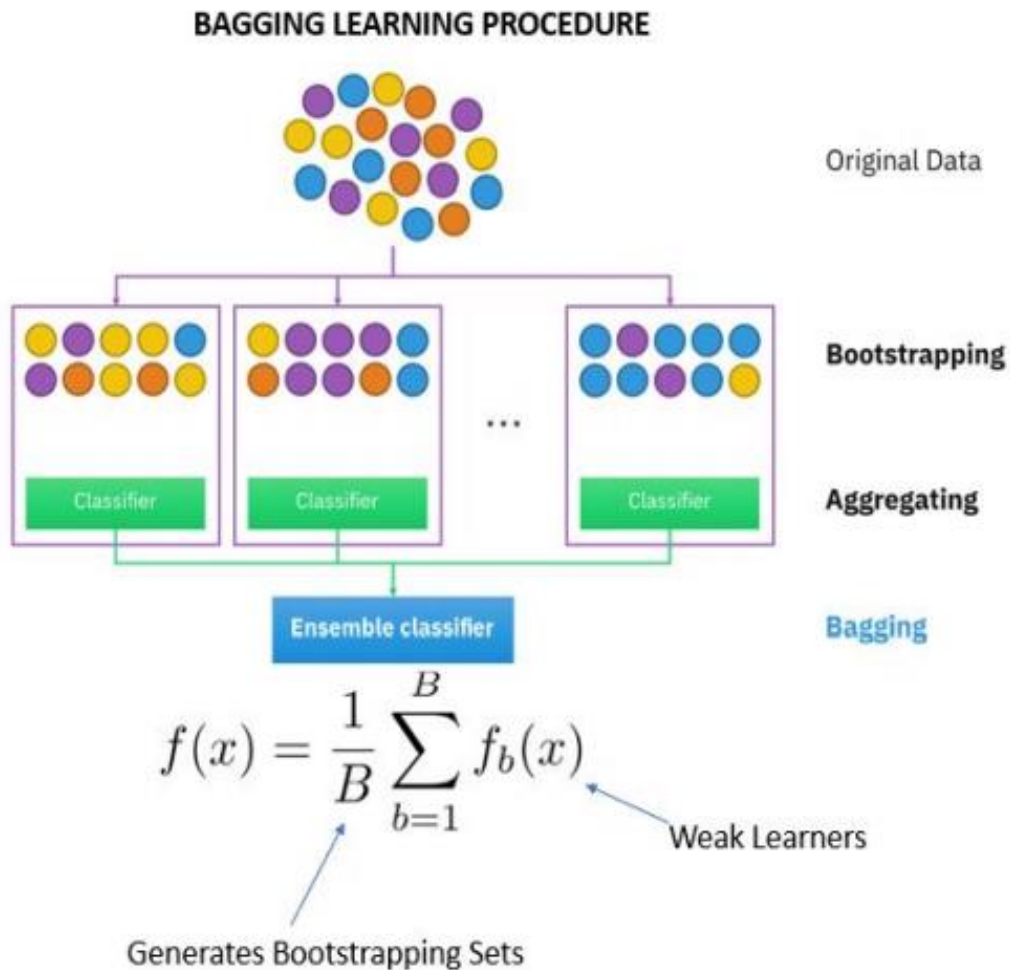
어렵지 않고 사용하기 쉬워 Machine Learning 초보자, 현업에서 많이 사용된다.

Boosting 계열에 비해 hyperparameter에 민감하지 않다.

의사결정나무를 이용한 Bagging의 특수한 형태를 취하는 앙상블 기법

Random Forest에서는 앙상블의 다양성을 확보하기 위해 **변수를 임의 추출**하여 bagging보다 더 다양한 decision tree를 만들어낸다.

Bagging의 문제점



X변수가 25개 있다고 가정하자.

CART에서는 25개의 모든 변수 중 지니 불순도를 가장 낮추는 것을 선택해서 뿌리 노드를 분할하고, 이때 Greedy search 진행.

만약 중요한 변수가 정해져 있다면, bagging에서 뿌리 노드를 분할할 때 계속해서 한 변수가 분할할 것이다.

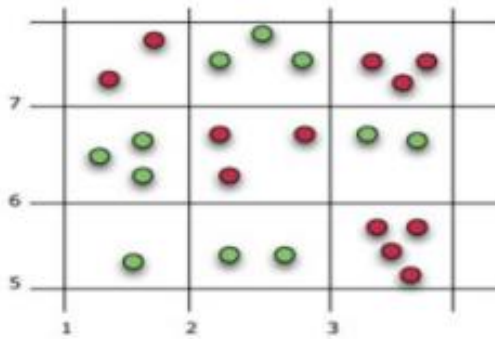
뿌리 노드에 선택되는 x 변수가 겹치는 경우가 많아 tree의 다양성이 낮아진다.

-> tree의 상관성이 높다면 앙상블의 의미가 줄어든다!
개선 필요.

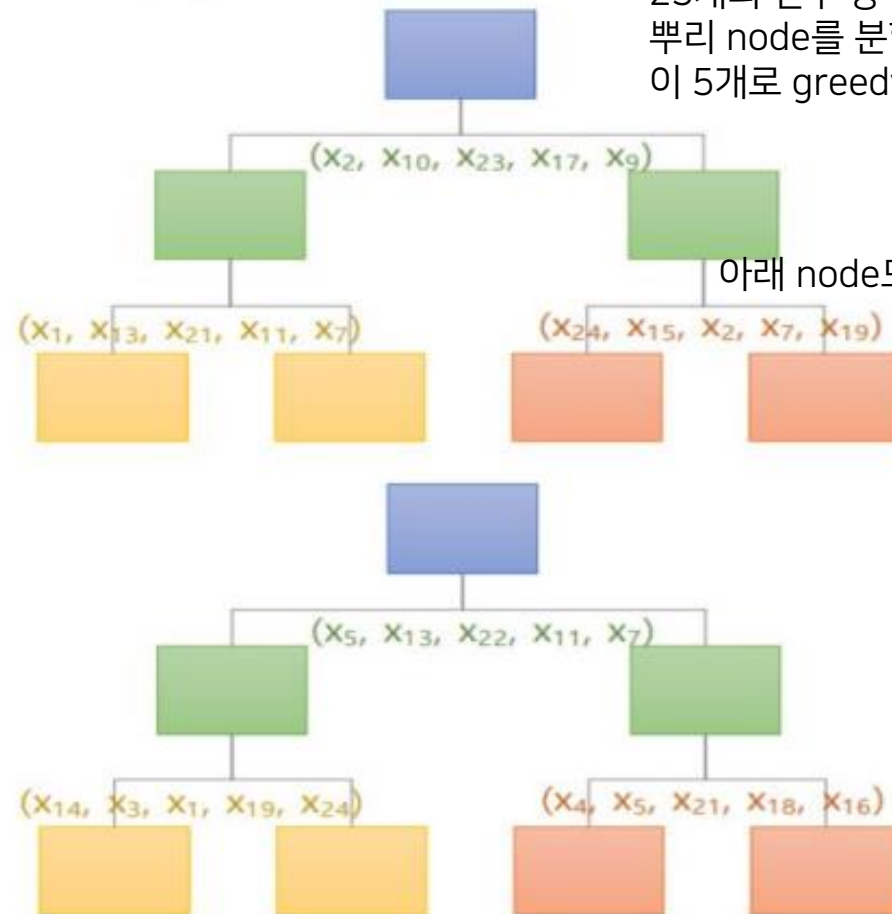
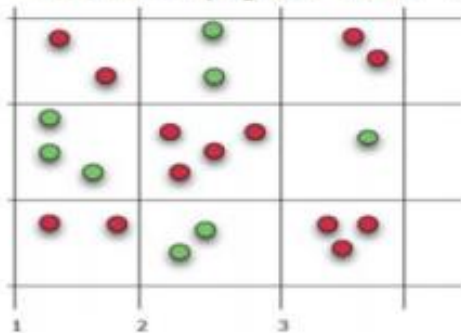
Random Forest

Random Forest의 문제 해결 방법:
변수의 임의추출을 통해 tree의 다양성을 확보하자.

Bootstrap i (X in R^{25})



Bootstrap j (X in R^{25})



25개의 변수 중 random하게 5개 선택해서
뿌리 node를 분할하자.
이 5개로 greedy search를 진행!

아래 node도 마찬가지로 random 추출

Random Feature Selection

Random Forest (랜덤 포레스트)의 Random Feature Selection(변수의 임의추출)

- 각 node를 분할할 때, 전체 변수 p 개 중 일부인 $m(m < p)$ 개의 변수만을 임의로 선정하여 가장 최적으로 분할하는 변수를 선택
- **Bootstrapping(샘플의 다양화) + Random Feature Selection(트리의 다양화)**를 통해 개별 tree의 다양성을 극대화한다.
- m 이 작으면 tree 간의 상관관계가 감소한다. But, 너무 작으면 나무 자체의 변별력이 없기 때문에 주의해야 한다.
- 보통 $m = \sqrt{p}$ (Classification), $m = \frac{p}{3}$ (Regression)
이것이 default 값이고, 데이터마다 최적의 m 은 모두 다르다.

Result Aggregating

Result Aggregating 방법 1: Majority voting 다수결

Training Accuracy	Ensemble population	P(y=1) for a test instance	Predicted class label
0.80	Model 1	0.90	1
0.75	Model 2	0.92	1
0.88	Model 3	0.87	1
0.91	Model 4	0.34	0
0.77	Model 5	0.41	0
0.65	Model 6	0.84	1
0.95	Model 7	0.14	0
0.82	Model 8	0.32	0
0.78	Model 9	0.98	1
0.83	Model 10	0.57	1

10개의 decision tree

cutoff = 0.5

$$\sum_{j=1}^n \delta(\hat{y}_j = 0) = 4$$

$$\sum_{j=1}^n \delta(\hat{y}_j = 1) = 6$$

↓ 다수결

$\hat{y}_{\text{Ensemble}} = 1$

$$\hat{y}_{\text{Ensemble}} = \arg \max_i \left(\sum_{j=1}^n \delta(\hat{y}_j = i), i \in \{0, 1\} \right)$$

Result Aggregating

Result Aggregating 방법 2: Weighted voting 가중치 (1)

★ Training Accuracy	Ensemble population	P(y=1) for a test instance	Predicted class label
0.80	Model 1	0.90	1
0.75	Model 2	0.92	1
0.88	Model 3	0.87	1
0.91	Model 4	0.34	0
0.77	Model 5	0.41	0
0.65	Model 6	0.84	1
0.95	Model 7	0.14	0
0.82	Model 8	0.32	0
0.78	Model 9	0.98	1
0.83	Model 10	0.57	1

성능에 따른 가중치 반영

10개의 decision tree

cutoff = 0.5

$\hat{y}_{\text{Ensemble}} = 1$


$$\frac{\sum_{j=1}^n (\text{Trn Acc}_j) \cdot \delta(\hat{y}_j=0)}{\sum_{j=1}^n (\text{Trn Acc}_j)} = 0.424$$

$$\frac{\sum_{j=1}^n (\text{Trn Acc}_j) \cdot \delta(\hat{y}_j=1)}{\sum_{j=1}^n (\text{Trn Acc}_j)} = 0.576$$

$$\hat{y}_{\text{Ensemble}} = \arg \max_i \left(\frac{\sum_{j=1}^n (\text{Trn Acc}_j) \cdot \delta(\hat{y}_j=i)}{\sum_{j=1}^n (\text{Trn Acc}_j)} \right), i \in (0, 1)$$

Result Aggregating

Result Aggregating 방법 2: Weighted voting 가중치 (2)

Training Accuracy	Ensemble population	 $P(y=1)$ for a test instance	Predicted class label	예측 확률 평균내기
0.80	Model 1	0.90	1	$\sum_{j=1}^n P(y=0) = 0.375$
0.75	Model 2	0.92	1	
0.88	Model 3	0.87	1	
0.91	Model 4	0.34	0	$\sum_{j=1}^n P(y=1) = 0.625$
0.77	Model 5	0.41	0	
0.65	Model 6	0.84	1	
0.95	Model 7	0.14	0	
0.82	Model 8	0.32	0	
0.78	Model 9	0.98	1	
0.83	Model 10	0.57	1	

성능에 따른 가중치 반영

10개의 decision tree

cutoff = 0.5

$\hat{y}_{\text{Ensemble}} = 1$

$$\hat{y}_{\text{Ensemble}} = \arg \max_i \left(\frac{1}{n} \sum_{j=1}^n P(y=i), i \in \{0,1\} \right)$$

Out of bag error (OOB error)

개별 학습 데이터셋 구성 시 Bootstrap으로 추출되지 않은 데이터를 검증용 데이터로 사용



In Bag: Random하게 복원추출을 하면, 전체 데이터 중 unique하게 63.2%의 데이터가 뽑힌다.

Out of Bag: 나머지 36.8%는 아예 bootstrapping에 뽑히지 못한다. 이들은 decision tree를 만들 때 활용되지 못하기 때문에 이것 이용해서 training accuracy를 계산하자.

변수 중요도

변수 중요도 (Feature Importance)

랜덤 포레스트는 회귀분석처럼 개별 변수가 통계적으로 얼마나 유의한지에 대한 정보를 제공하지 않는다. 즉, p-value 값이 없다.

그렇다면 변수의 중요도를 어떻게 특정할 수 있을까?

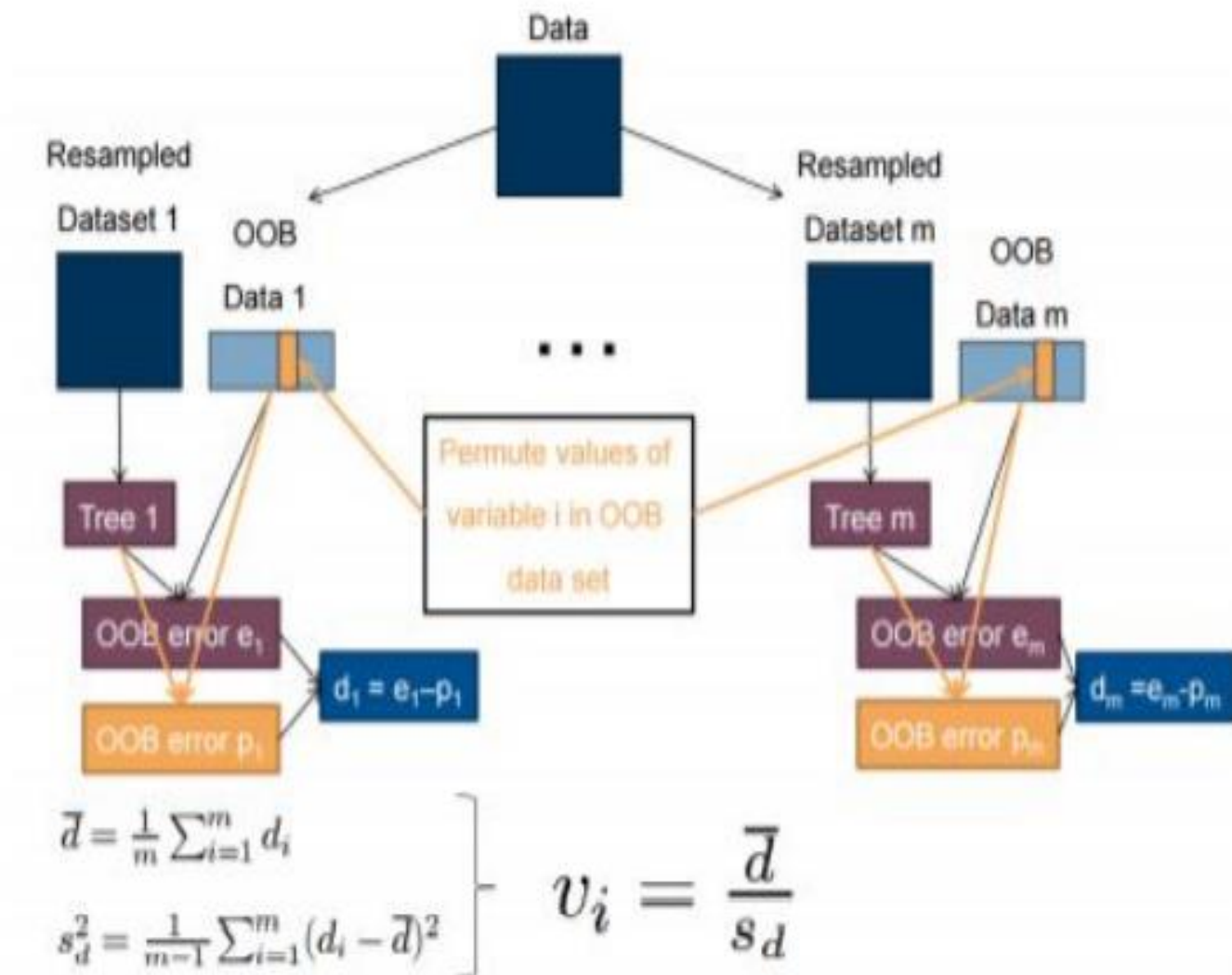
1. 원래 OOB 데이터 집합에 대해서 OOB Error를 구한다. (OOB 데이터로 validation을 진행하는 것)
2. 특정 변수의 값을 임의로 뒤섞은 OOB 데이터 집합에 대해서 OOB error를 구한다.
3. 개별 변수의 중요도는 2단계와 1단계 OOB error 차이의 평균과 분산을 고려하여 추정한다.

기본 idea: 이때 계산한 error의 차이가 클수록 해당 변수가 tree에서 중요한 역할을 한다.

: 모든 tree에서의 error 차이에 대해 평균/표준편차를 계산한다. 이 값이 변수 중요도.

변수 중요도 계산 필기 slide

변수 중요도 계산



변수 중요도

변수 중요도 (Feature Importance)

Random Forest의 변수 중요도는 실제로 다른 분석을 할 때에도 많이 쓰인다.

Random Forest는 해석력이 낮다는 단점이 있는데, 이때 변수 중요도가 결국 해석이 될 수 있다.

감사합니다

5기 허유진