Ensemble Learning

Business Analytics (IME654)

2022. 11. 21

Team: 동기사랑

Member: 김창현, 정진용

Ensemble Learning

해당 발표자료는

고려대학교 산업경영공학과

강필성 교수님: 비즈니스 애널리틱스(IME654)

김성범 교수님: 다변량 통계분석 및 데이터 마이닝(IME567)

의 강의자료를 사용했음을 미리 밝힙니다.

Ensemble Learning

❖ Ensemble learning

✓ Model ensemble은 여러 모델들을 함께 사용하여 기존보다 성능을 더 올리는 방법을 말함

1. Bagging

- Bootstrap Aggregating의 약자이며 bootstrap을 이용하는 방법
  - ✓ Bootstrap: 주어진 데이터셋에서 random sampling을 거쳐 새로운 데이터셋을 만들어내는 과정
  - ✓ 만들어진 여러 데이터셋을 바탕으로 결과를 voting
- ex) Random Forest

2. Voting

- Voting은 크게 Hard voting과 soft voting으로 나눌 수 있음
  - ✓ Hard voting: 각 하위 학습 모델(weak learner)들의 예측 결과값을 바탕으로 다수결 투표하는 방식
  - ✓ Soft voting: 각 하위 학습 모델(weak learner)들의 예측 확률값의 평균 또는 가중치 합을 사용하는 방식

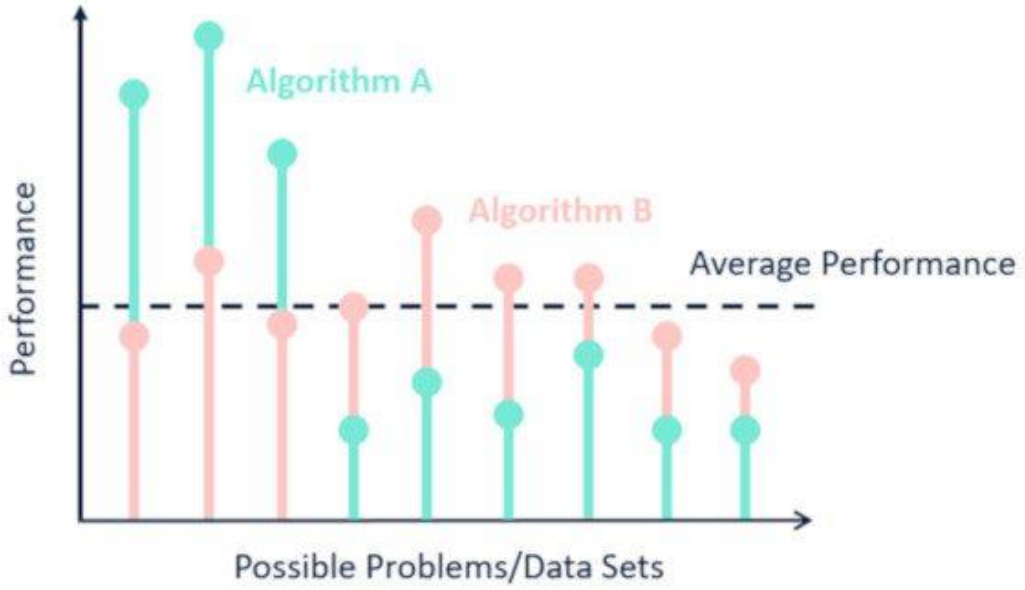
3. Boosting

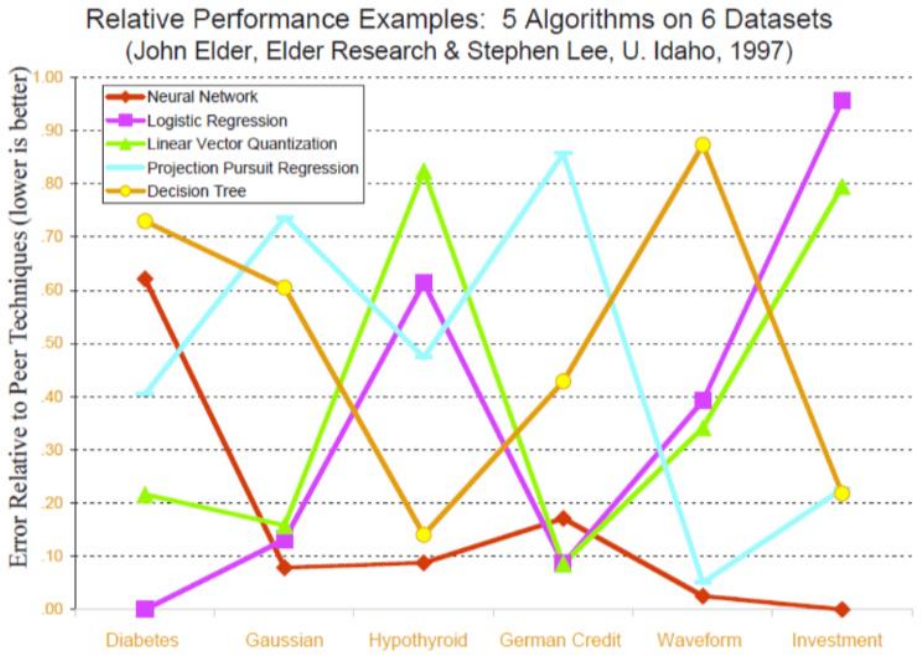
- 모델 iteration의 결과에 따라 데이터셋 샘플에 대한 가중치를 부여하며 모델을 업데이트하는 방식
- 반복할 때마다 각 샘플의 중요도에 따라 다른 분류기가 만들어지고 최종적으로는 모든 iteration에서 생성된 모델의 결과를 votin함
- Adaptive Boosting(AdaBoost)와 Gradient Boosting Model(GBM) 계열로 나눌 수 있음

Background

❖ No Free Lunch Theorem?

- 머신러닝은 다양한 샘플 데이터에 학습(fitting)을 시킴으로써 일반화되기를 목적으로 함
- '모델이 학습을 한다'라는 의미는 샘플 데이터로 구성된 가설 공간속에서 데이터에 알맞은 가설을 채택하는 것이라 볼 수 있음
- 다양한 가설이 존재하는 만큼 귀납적 편향의 문제에 마주침





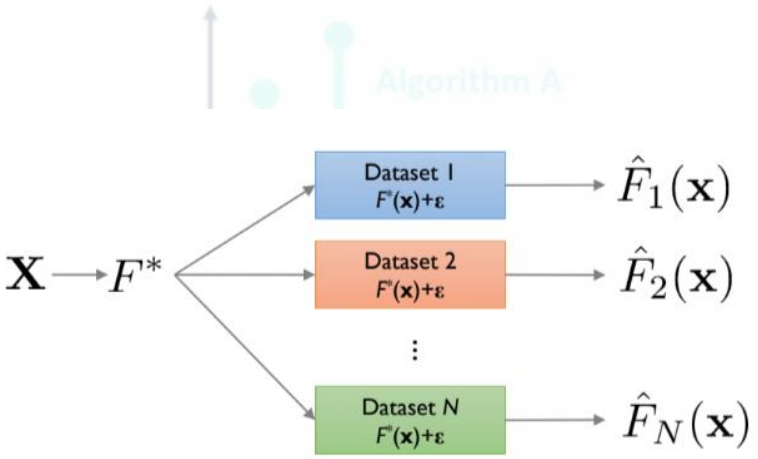
다양한 데이터셋에 대한 각 알고리즘들 성능 비교

https://www.kdnuggets.com/2019/09/no-free-lunch-data-science.html

Background

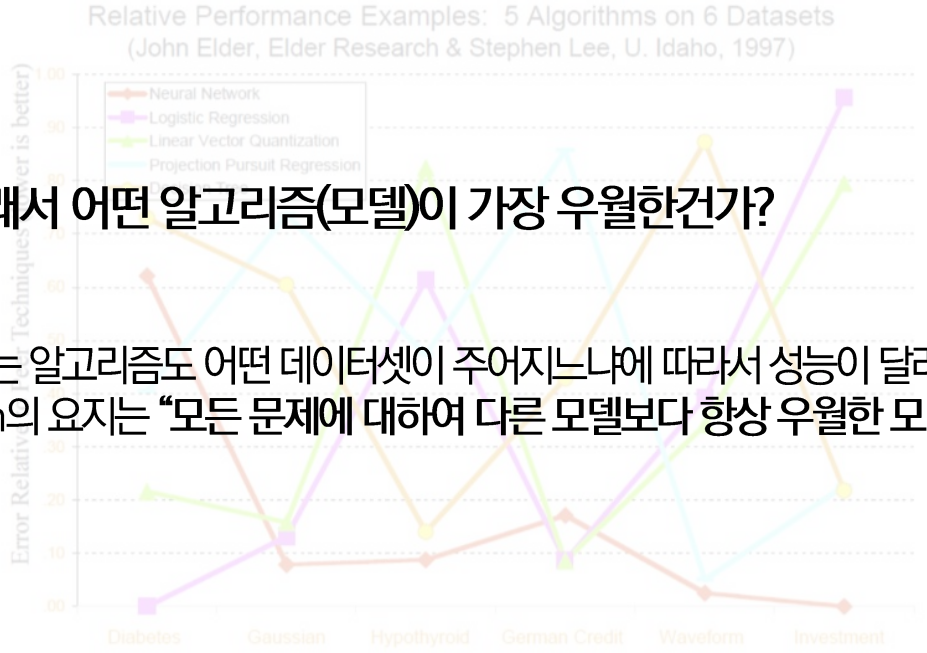
❖ No Free Lunch Theorem?

- 머신러닝은 다양한 샘플 데이터에 학습(fitting)을 시킴으로써 일반화되기를 목적으로 함
- '모델이 학습을 한다'라는 의미는 샘플 데이터로 구성된 가설 공간속에서 데이터에 알맞은 가설을 채택하는 것이라 볼 수 있음
- 다양한 가설이 존재하는 만큼 귀납적 편향의 문제에 마주침



그래서 어떤 알고리즘(모델)이 가장 우월한건가?

성능이 좋다고 하는 알고리즘도 어떤 데이터셋이 주어지느냐에 따라서 성능이 달라짐
No Free Lunch Theorem의 요지는 **"모든 문제에 대하여 다른 모델보다 항상 우월한 모델은 없다"**



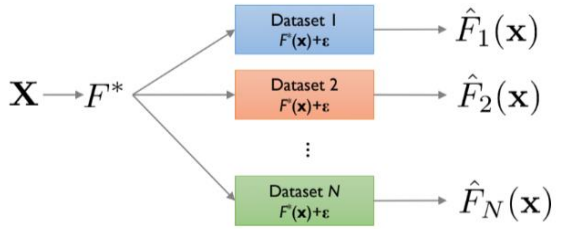
다양한 데이터셋에 대한 각 알고리즘들 성능 비교

https://www.kdnuggets.com/2019/09/no-free-lunch-data-science.html

Background

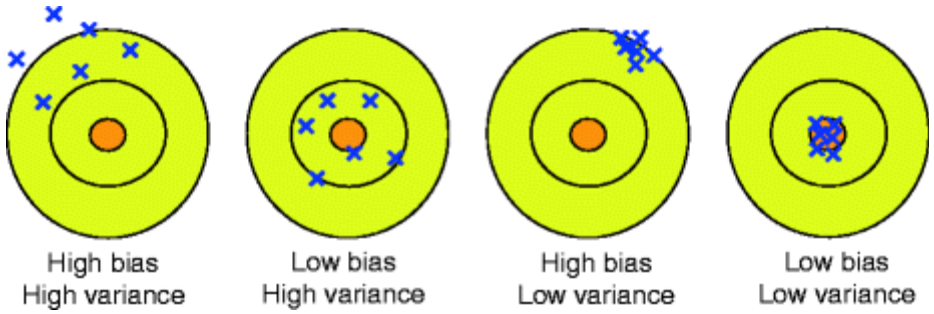
❖ Bias-Variance Decomposition

- 특정 데이터에 대한 오차를 편향과 분산에 의한 에러로 나눌 수 있음
- 편향이 높으면 과소적합이 발생하며 분산이 높으면 과적합이 발생함
- Bias는 정답과 평균 추정치 차이, Variance는 평균 추정치와 특정 데이터셋에 대한 추정치 차이



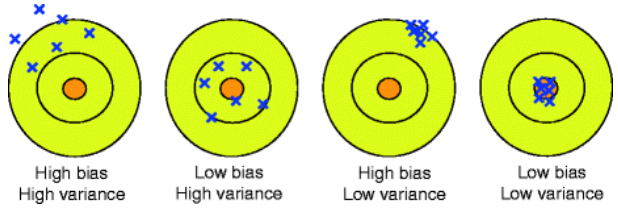
$$y = F^*(\mathbf{x}) + \epsilon, \quad \epsilon \sim N(0, \sigma^2)$$

$$\bar{F}(\mathbf{x}) = E[\hat{F}_D(\mathbf{x})]$$



$$\begin{aligned}
Error(X_0) &= E[(y - \hat{F}(X)|X = X_0)^2] \\
&= E[(F^*(X) + \varepsilon - \hat{F}(X))^2] \qquad (\because y = F^*(X) + \varepsilon) \\
&= E[(F^*(X) - \hat{F}(X))^2] + \sigma^2 \\
&= E[(F^*(X) - \bar{F}(X) + \bar{F}(X) - \hat{F}(X))^2] + \sigma^2 \\
&= E\left[\left(F^*(X) - \bar{F}(X)\right)^2 + \left(\bar{F}(X) - \hat{F}(X)\right)^2 + 2\left(F^*(X) - \bar{F}(X)\right)\left(\bar{F}(X) - \hat{F}(X)\right)\right] + \sigma^2 \\
&= E\left[\left(F^*(X) - \bar{F}(X)\right)^2\right] + E\left[\left(\bar{F}(X) - \hat{F}(X)\right)^2\right] + \sigma^2 \\
&= Bias^2\left(\hat{F}(X_0)\right) + Var^2\left(\hat{F}(X_0)\right) + \sigma^2
\end{aligned}$$

Background



❖ Ensemble Learning

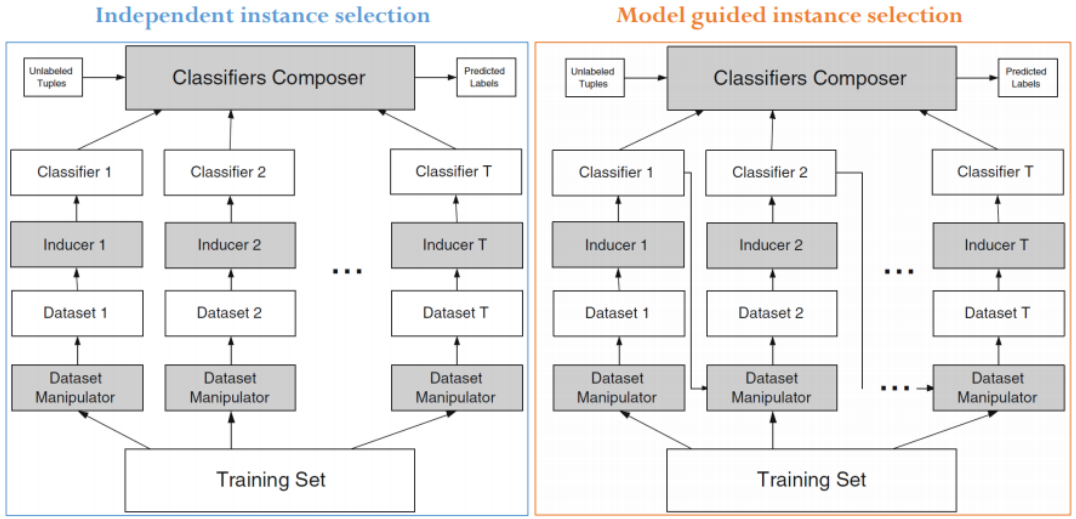
- 앙상블의 목적은 각 단일 모델의 좋은 성능을 유지하면서 다양성(diversity)을 확보하는 데 있음
  - ✓ Implicit diversity를 확보: 전체 데이터셋의 부분집합에 해당하는 여러 데이터셋을 준비한 뒤 따로 학습
  - ✓ Explicit diversity를 확보: 먼저 생성된 모델의 측정값으로부터 새로운 모델을 생성하여 학습

단일 모델 : Decision Tree, ANN, SVM, k값이 작은 K-NN

Bagging이나 Random Forest 등을 통해서 분산을 줄이자!

단일 모델 : Logistic Regression, k값이 큰 K-NN

Boosting을 통해서 분산을 줄이자!



Background

❖ Ensemble Learning

- 다양성을 확보한다면 단일 모델보다 앙상블 모델이 좋은 성능을 보이는 건지?
  - ✓ 수식 증명

Background

- ❖ Ensemble learning VS single algorithm learning
  - 실제 앙상블 모델들이 단일 모델보다 좋은 성능을 보임

What is SQuAD?

Stanford **Q**uestion **A**nswering **D**ataset (SQuAD) is a reading comprehension dataset, consisting of questions posed by crowdworkers on a set of Wikipedia articles, where the answer to every question is a segment of text, or *span*, from the corresponding reading passage, or the question might be unanswerable.

SQuAD2.0 combines the 100,000 questions in SQuAD1.1 with over 50,000 unanswerable questions written adversarially by crowdworkers to look similar to answerable ones. To do well on SQuAD2.0, systems must not only answer questions when possible, but also determine when no answer is supported by the paragraph and abstain from answering.

Explore SQuAD2.0 and model predictions

SQuAD2.0 paper (Rajpurkar & Jia et al. '18)

SQuAD 1.1, the previous version of the SQuAD dataset, contains 100,000+ question-answer pairs on 500+ articles.

Explore SQuAD1.1 and model predictions

SQuAD1.0 paper (Rajpurkar et al. '16)

Getting Started

We've built a few resources to help you get started with the dataset.
Download a copy of the dataset (distributed under the CC BY-SA 4.0 license):

Leaderboard

SQuAD2.0 tests the ability of a system to not only answer reading comprehension questions, but also abstain when presented with a question that cannot be answered based on the provided paragraph.

Rank	Model	EM	F1
	Human Performance *Stanford University* (Rajpurkar & Jia et al. '18)	86.831	89.452
1 Jun 04, 2021	IE-Net (ensemble) *RICOH_SRCB_DML*	**90.939**	**93.214**
2 Feb 21, 2021	FPNet (ensemble) *Ant Service Intelligence Team*	90.871	93.183
3 May 16, 2021	IE-NetV2 (ensemble) *RICOH_SRCB_DML*	90.860	93.100
4 Apr 06, 2020	SA-Net on Albert (ensemble) *QIANXIN*	90.724	93.011
5 May 05, 2020	SA-Net-V2 (ensemble) *QIANXIN*	90.679	92.948
5 Apr 05, 2020	Retro-Reader (ensemble) *Shanghai Jiao Tong University* http://arxiv.org/abs/2001.09694	90.578	92.978
5 Feb 05, 2021	FPNet (ensemble) *YuYang*	90.600	92.899
6 Apr 18, 2021	TransNets + SFVerifier + SFEnsembler (ensemble) *Senseforth AI Research* https://www.senseforth.ai/	90.487	92.894
6 Dec 01, 2020	EntitySpanFocusV2 (ensemble) *RICOH_SRCB_DML*	90.521	92.824

https://rajpurkar.github.io/SQuAD-explorer/

✓2016

Object detection (DET)[top]

Task 1a: Object detection with provided training data

Ordered by number of categories won

Team name	Entry description	Number of object categories won	mean AP
CUImage	Ensemble of 6 models using provided data	109	0.662751
Hikvision	Ensemble A of 3 RPN and 6 FRCN models, mAP is 67 on val2	30	0.652704
Hikvision	Ensemble B of 3 RPN and 5 FRCN models, mean AP is 66.9, median AP is 69.3 on val2	18	0.652003

Object localization (LOC)[top]

Task 2a: Classification+localization with provided training data

Ordered by localization error

Team name	Entry description	Localization error	Classification error
Trimps-Soushen	Ensemble 3	0.077087	0.02991
Trimps-Soushen	Ensemble 4	0.077429	0.02991
Trimps-Soushen	Ensemble 2	0.077668	0.02991
Trimps-Soushen	Ensemble 1	0.079068	0.03144

✓2017

Object detection (DET)[top]

Task 1a: Object detection with provided training data

Ordered by number of categories won

Team name	Entry description	Number of object categories won	mean AP
BDAT	submission4	85	0.731392
BDAT	submission3	65	0.732227
BDAT	submission2	30	0.723712
DeepView(ETRI)	Ensemble_A	10	0.593084
NUS-Qihoo_DPNs (DET)	Ensemble of DPN models	9	0.656932
KAISTNIA_ETRI	Ensemble Model5	1	0.61022
KAISTNIA_ETRI	Ensemble Model4	0	0.609402
KAISTNIA_ETRI	Ensemble Model2	0	0.608299
KAISTNIA_ETRI	Ensemble Model1	0	0.608278
KAISTNIA_ETRI	Ensemble Model3	0	0.60631

Object localization (LOC)[top]

Task 2a: Classification+localization with provided training data

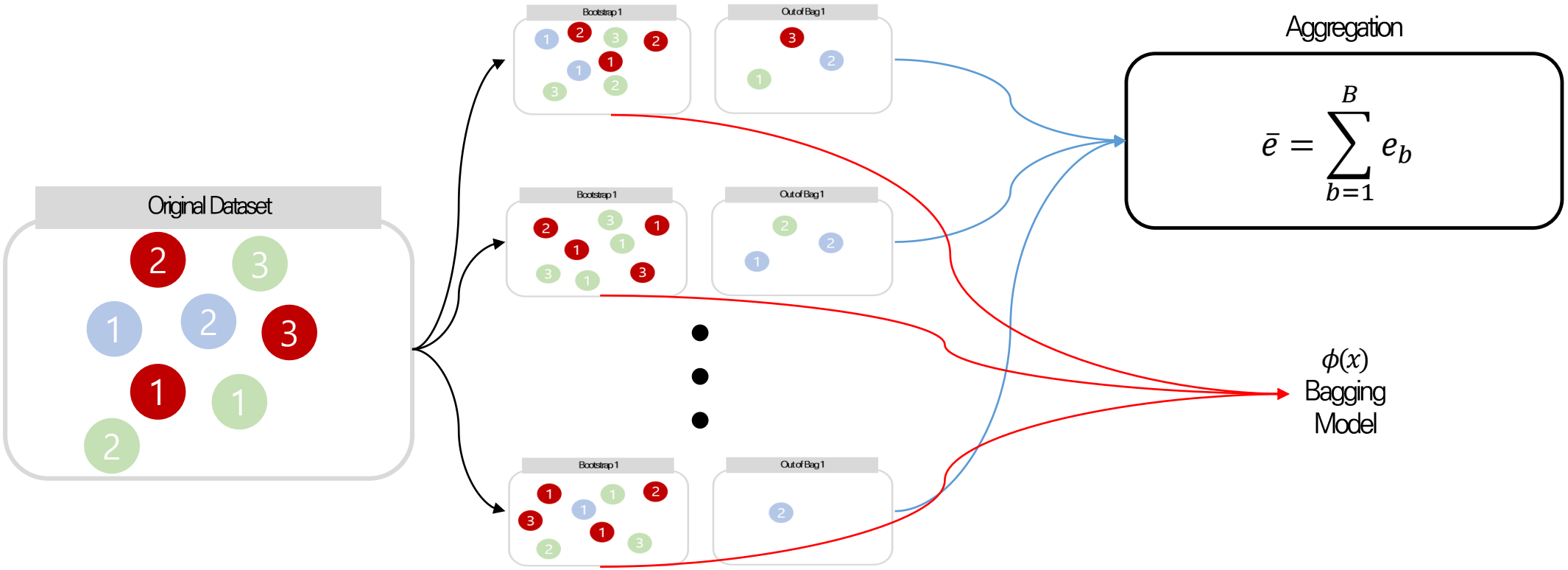
Ordered by localization error

Team name	Entry description	Localization error	Classification error
NUS-Qihoo_DPNs (CLS-LOC)	[E3] LOC: Dual Path Networks + Basic Ensemble	0.062263	0.03413
Trimps-Soushen	Result-3	0.064991	0.02481
Trimps-Soushen	Result-2	0.06525	0.02481
Trimps-Soushen	Result-4	0.065261	0.02481
Trimps-Soushen	Result-5	0.065302	0.02481
Trimps-Soushen	Result-1	0.067698	0.02481

Ensemble Learning

❖ Booststrap Aggregating(Bagging)이란?

- 배깅은 주어진 데이터셋에 대해 bootstrap 샘플링을 이용하여 단일 알고리즘 모델보다 더 좋은 모형을 만들 수 있는 앙상블 기법임
- 각 데이터셋은 복원추출을 통해 기존 데이터셋만큼의 크기를 갖도록 샘플링됨
- 개별 샘플링 된 데이터셋은 bootstrap이라 함



Ensemble Learning

❖ Booststrap Aggregating(Bagging)이란?

- 배깅은 주어진 데이터셋에 대해 bootstrap 샘플링을 이용하여 단일 알고리즘 모델보다 더 좋은 모형을 만들 수 있는 앙상블 기법임
- 각 데이터셋은 복원추출을 통해 기존 데이터셋만큼의 크기를 갖도록 샘플링됨
- 개별 샘플링 된 데이터셋은 bootstrap이라 함

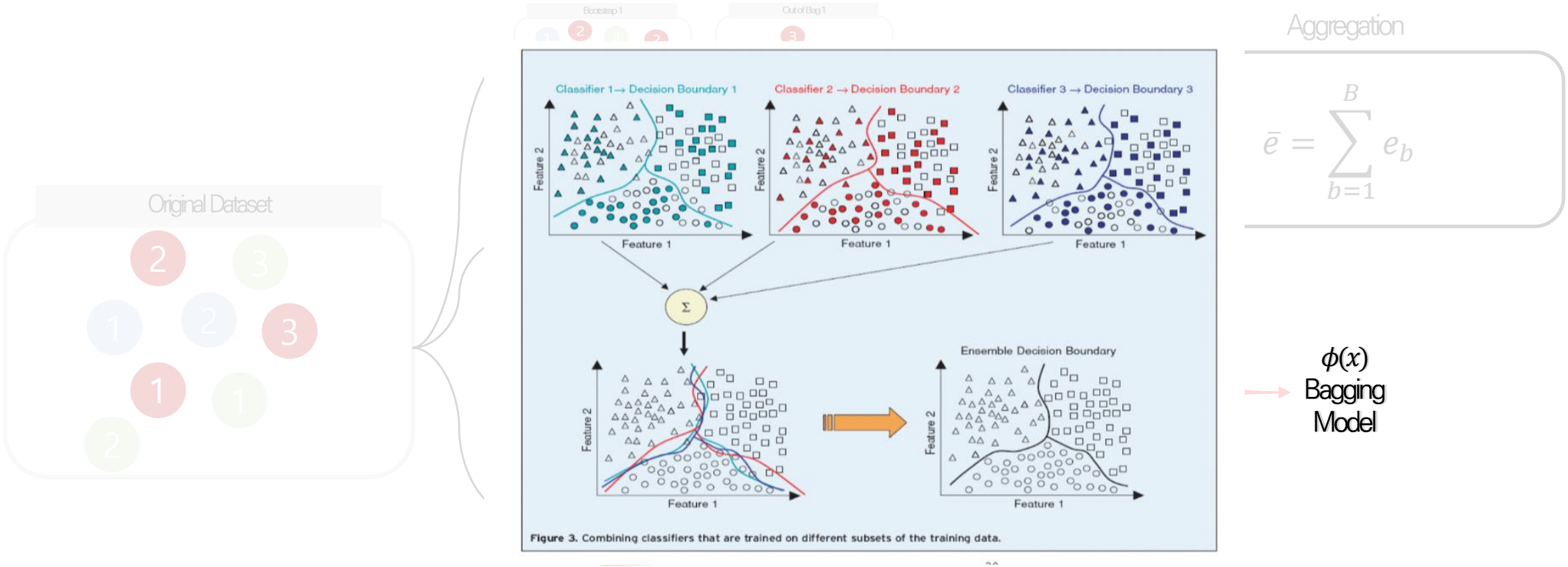
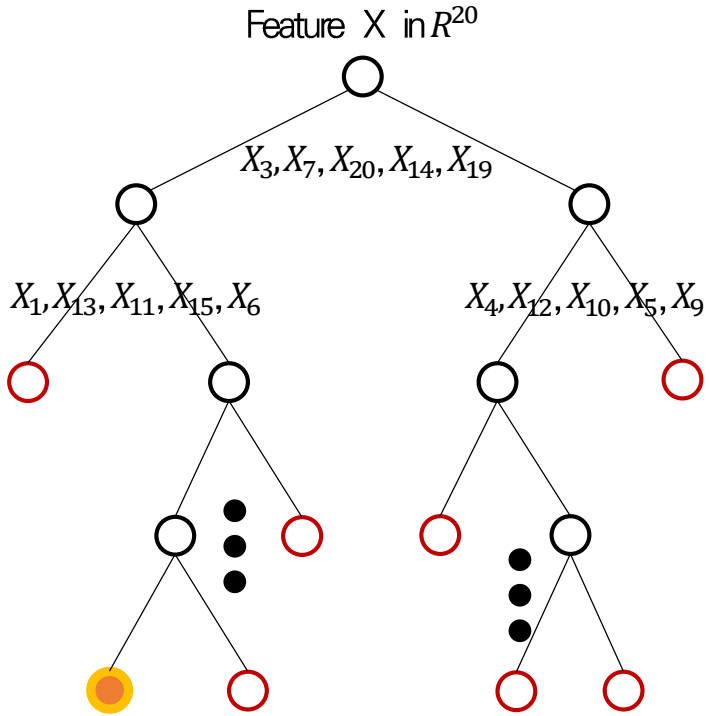


Figure 3. Combining classifiers that are trained on different subsets of the training data.

Ensemble Learning

❖ Randomly chosen predictor variables이란?

- 앙상블의 diversity를 확보하기 위한 기법
- 각 decision tree 분기점을 탐색할 때, 기존 변수 수보다 적은 변수 수를 임의로 선택하여 분기함



(1) 변수 X_i가 tree split에 한번도 사용되지 않았다면,
OOB Error of the original Data e_i = OOB Error of the Permuted Data p_i

(2) 변수 X_i가 tree split에 중요하게 사용되었다면,
OOB Error of the original Data e_i 〈 OOB Error of the Permuted Data p_i

- m번째 tree에서 변수 i에 대한 Random permutation 전후 OOB error의 차이

$$d_i^m = p_i^m - e_i^m$$

- 전체 Tree들에 대한 OOB error 차이의 평균 및 분산

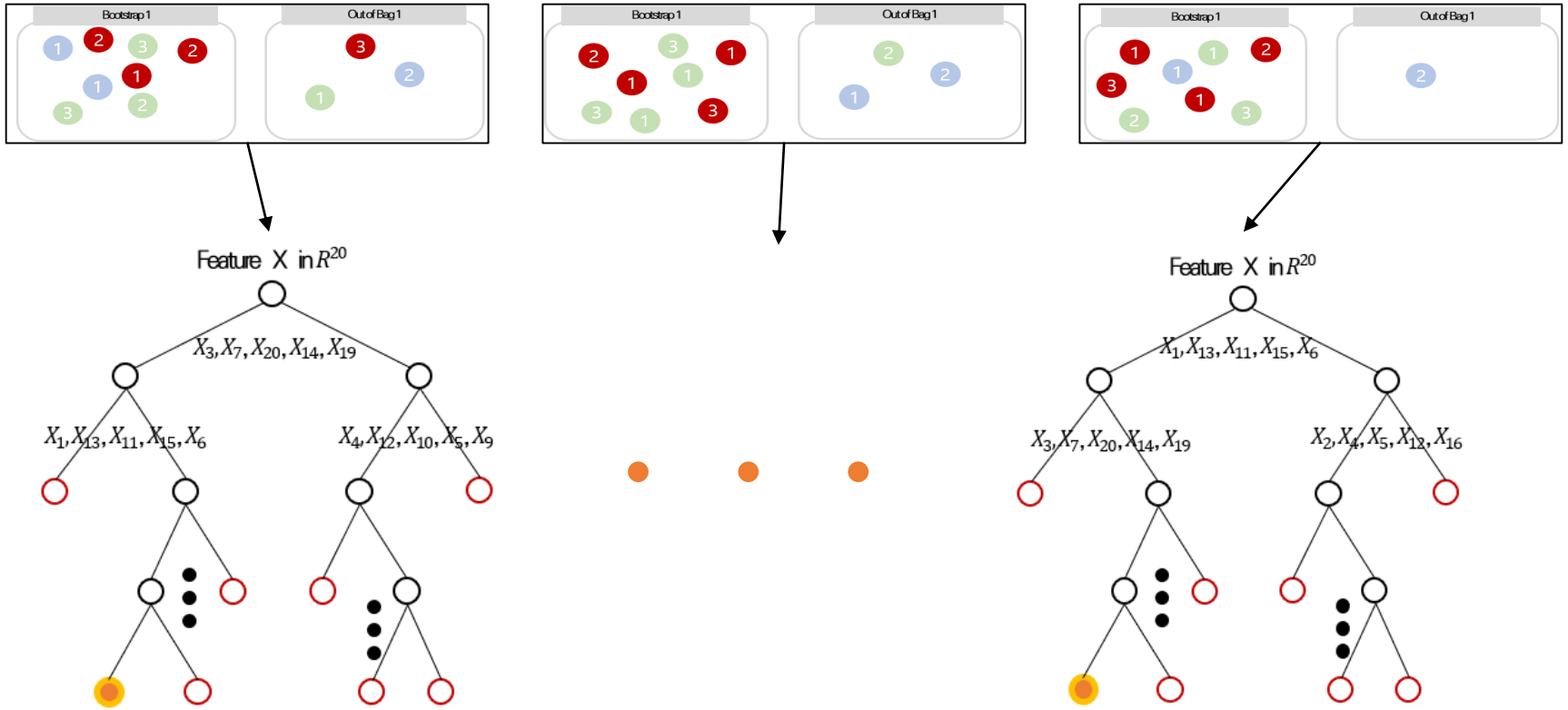
$$\bar{d}_i = \frac{1}{m}\sum_{i=1}^{m} d_i^m, \quad s_i^2 = \frac{1}{m-1}\sum_{i=1}^{m}(d_i^m - \bar{d}_i)^2$$

- i번째 변수의 중요도: $v_i = \dfrac{\bar{d}_i}{s_i}$

Ensemble Learning

❖ Random Forest

- Decision tree으로 ensemble을 하는 기법
- Ensemble을 할 때, 다양성을 확보하기 위하여 bagging과 randomly chosen predictor variables 두 가지 기법을 사용함



Ensemble Learning

❖ Boosting

"Can a set of weak learners create a single strong learner?"

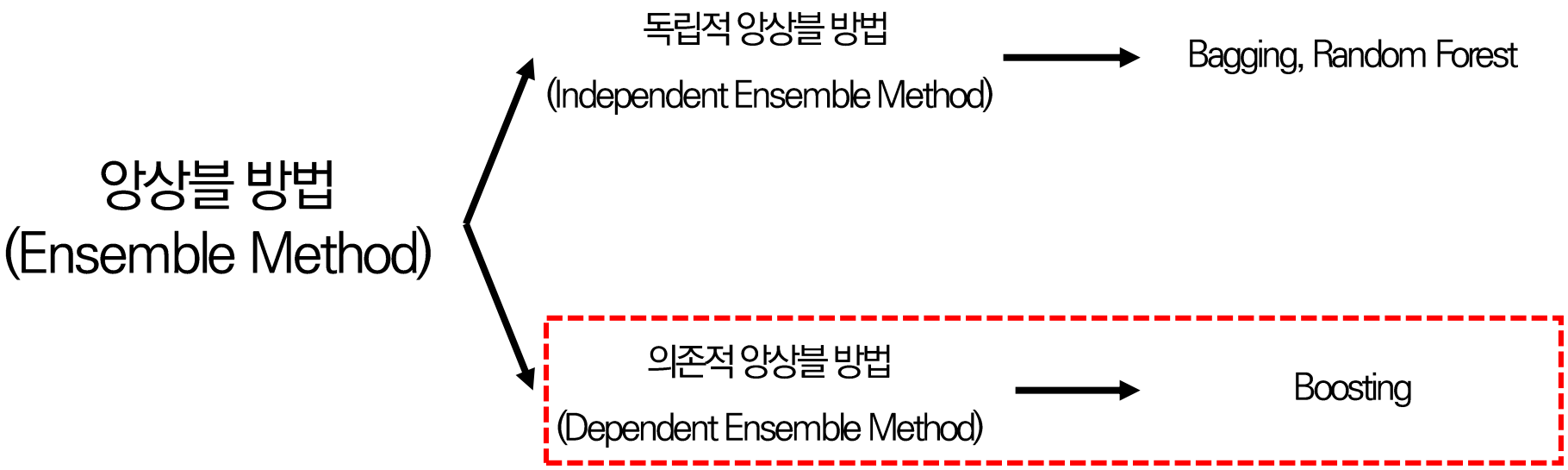
"Yes! Boosting!" –1997

Robert Schapire
- Princeton 컴퓨터과학과 교수
- Microsoft Research

Ensemble Learning

❖ Boosting

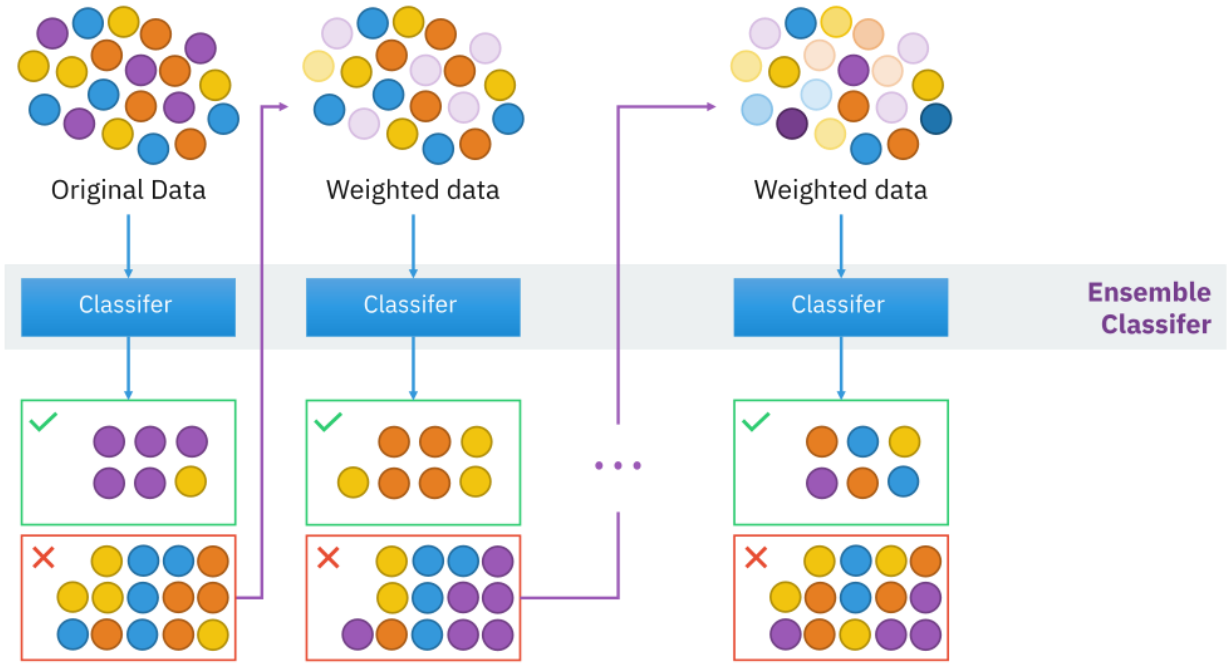
- 의존적 앙상블 방법: 개별 학습기들이 서로 독립이 아닌 경우를 의미



Ensemble Learning

❖ Boosting

- 의존적 앙상블 방법: 개별 학습기들이 서로 독립이 아닌 경우를 의미
- 핵심 아이디어: 분류하기 어려운 데이터에 학습을 집중함
- 학습 초기: 모든 데이터에 대해 동일한 가중치를 할당 / 학습 진행: 올바르게 분류된 데이터 가중치 줄이고, 잘못 분류된 데이터의 가중치 증가
- 배깅: 각 데이터가 추출될 확률이 모두 동일 / 부스팅: 각 데이터에 할당된 가중치에 비례해 추출



Ensemble Learning

❖ Boosting – AdaBoost

- 부스팅 모델의 시초
- 이전 모델이 오분류한 데이터의 가중치를 수정 → 오분류한 경우에 더 높은 가중치 부여

Algorithm 2 Adaboost

Input: Required ensemble size T

Input: Training set $S = \{(x_1, y_1), (x_2, y_2), ..., (x_N, y_N)\}$, where $y_i \in \{-1, +1\}$

Define a uniform distribution $D_1(i)$ over elements of S.

for $t = 1$ to T **do**

Train a model h_t using distribution D_t.

Calculate $\epsilon_t = P_{D_t}(h_t(x) \neq y)$

If $\epsilon_t \geq 0.5$ break

Set $\alpha_t = \frac{1}{2} \ln\left(\frac{1-\epsilon_t}{\epsilon_t}\right)$

Update $D_{t+1}(i) = \frac{D_t(i) \exp(-\alpha_t y_i h_t(x_i))}{Z_t}$

where Z_t is a normalization factor so that D_{t+1} is a valid distribution.

end for

For a new testing point (x', y'),

$H(x') = sign\left(\sum_{t=1}^{T} \alpha_t h_t(x')\right)$

Ensemble Learning

이진(binary classifier) 기준, I = identical function

❖ Boosting – AdaBoost

1. 트레이닝 데이터의 각 데이터 포인트별 초기 가중치와 오차율 $w_i = \frac{1}{n}$ *(i=1, ..., n)* $\epsilon = \frac{1}{n}\sum_{i=1}^{n} I\left(y_i \neq f_j(X_i)\right)$

2. j번째 약한 학습기 $f_j(X_i)$를 이용해 트레이닝 데이터 학습

3. (2)에서 사용한 약한 학습기 $f_j(X_i)$의 가중치가 적용된 오차율 구함 $\epsilon = \frac{1}{n}\sum_{i=1}^{n} w_i I\left(y_i \neq f_j(X_i)\right)$

4. 약한 학습기 전체에 적용된 가중치 $\alpha_j = \frac{1}{2} log(\frac{1-e_j}{e_j})$

Ensemble Learning

이진(binary classifier) 기준, I = identical function

❖ Boosting – AdaBoost

5. 4에서 구한 가중치 α_j 와 현재 데이터 포인트에 적용 중인 가중치 w_i를 이용해 w_i를 업데이트

$$W_i <- W_i \exp[-\alpha_i y_i f_i(x_i)], i = 1, \ldots, n$$

예측값과 실제값이 동일하다면 $y_i f_i(x_i) = 1$이 되므로, 개별 데이터 x_i의 가중치 w_i는 작아짐
예측값과 실제값이 다르다면 $y_i f_i(x_i) = -1$이 되므로 개별 데이터 포인트 x_i 의 가중치가 커짐

6. 가중치 합이 1이 되도록 가중치를 정규화

$$W_i <- \frac{W_i}{\sum_{i=1}^{n} W_i}$$

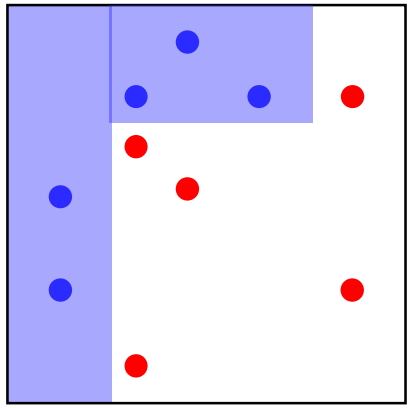
7. 2~6 단계를 약한 학습기 수만큼 반복 수행

8. 강한 학습기를 이용해 최종 예측값을 구함 $F(x) = sign(\sum_{j=1}^{m} \alpha_i f_i(X))$

Ensemble Learning

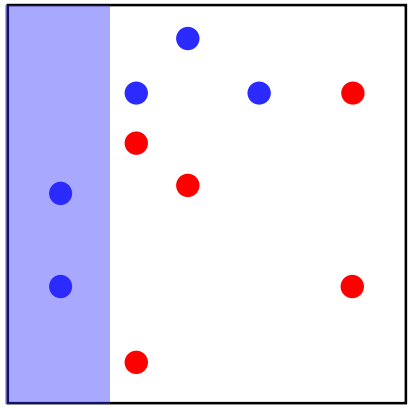
❖ Boosting – AdaBoost

AdaBoost



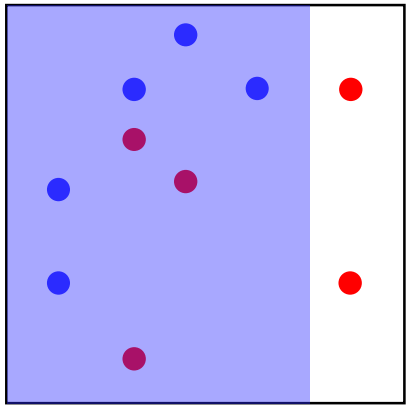
$= sign(\alpha_1$

약한 학습기1



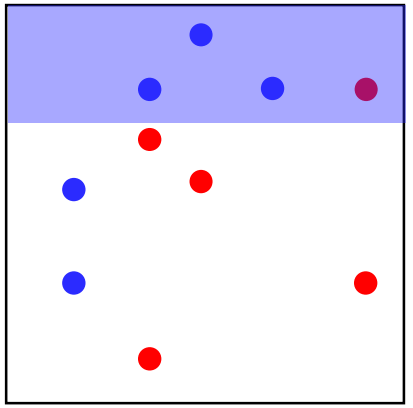
$+\alpha_2$

약한 학습기2



$+\alpha_3$

약한 학습기3

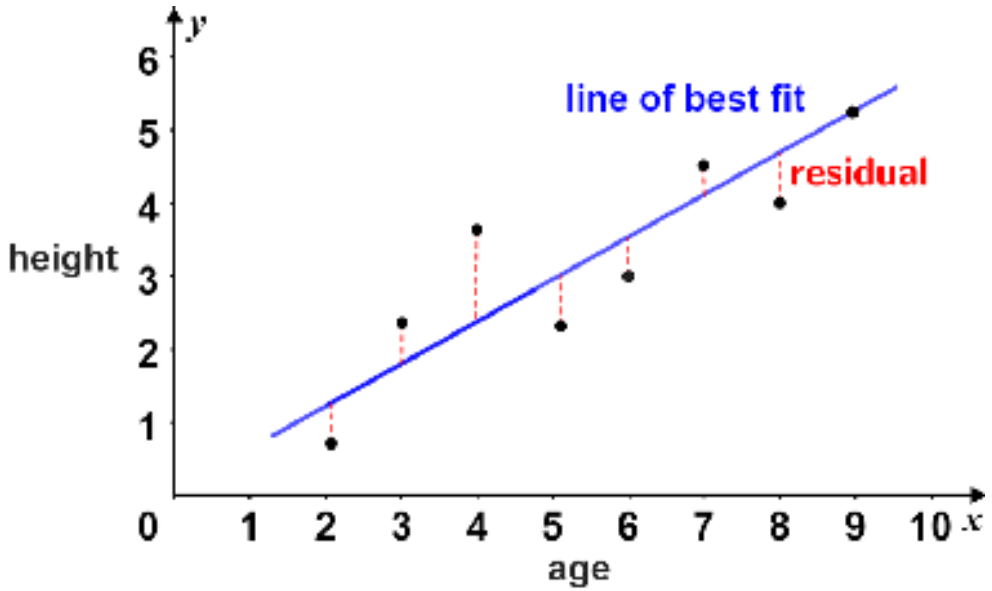


$)$

Ensemble Learning

❖ Boosting – Gradient Boosting Machine

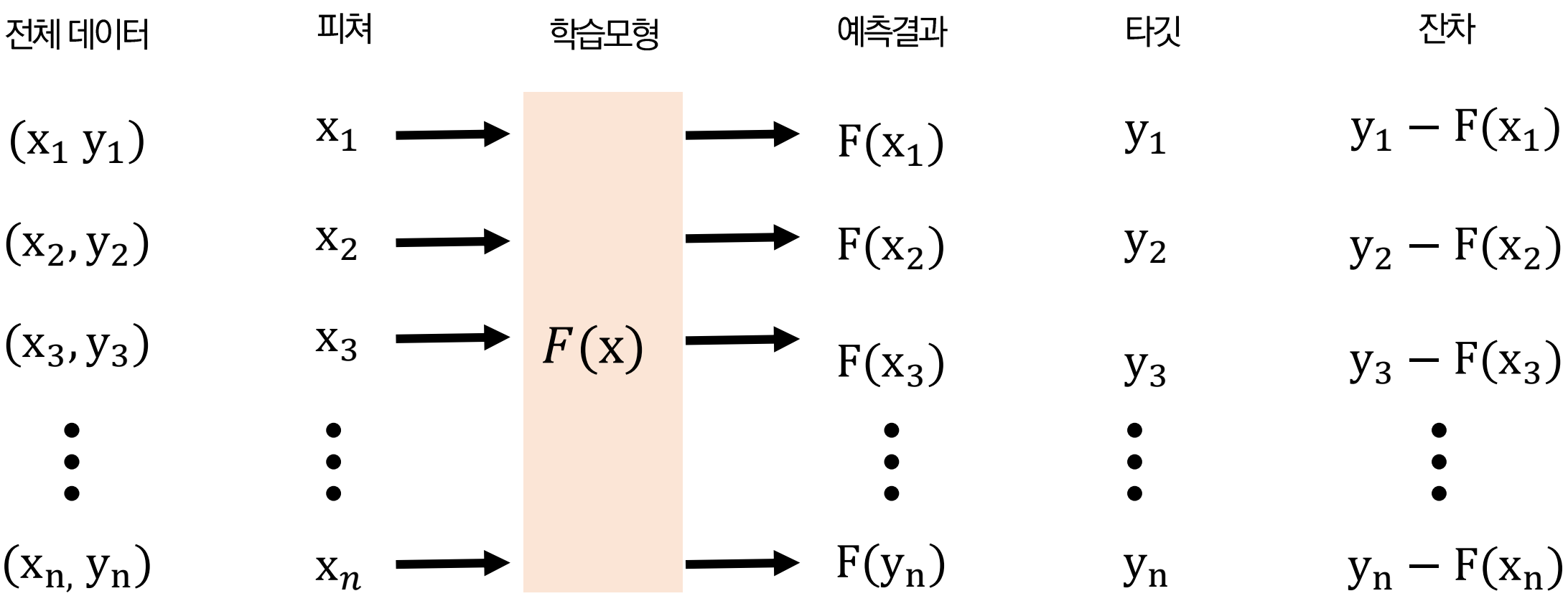
- Gradient Boosting은 말 그대로 gradient를 이용해 부스팅하는 방법
- 비용 함수를 최적화시킴으로써 학습 능력을 향상하는 알고리즘
- 만약 회귀모형의 잔차를 다음 단계에서 학습하는 모델을 구축할 수 있다면?



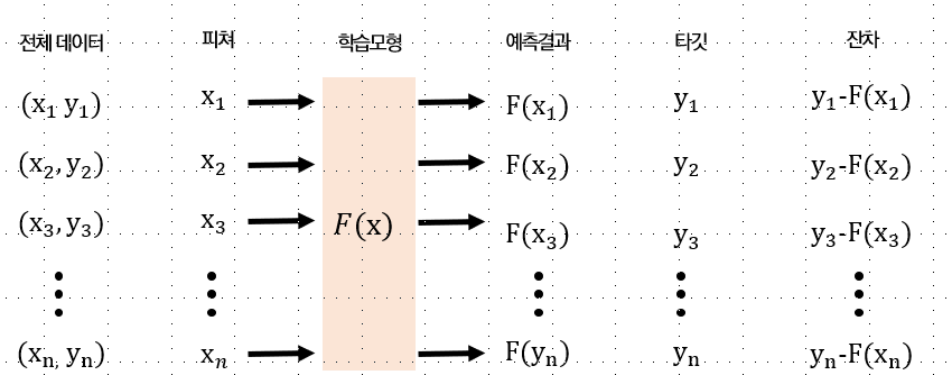
Ensemble Learning

❖ Boosting – Gradient Boosting Machine

- 일반적인 학습 과정



Ensemble Learning



❖ Boosting – Gradient Boosting Machine

- Gradient 학습 과정
- 새로운 모형 $f(x)$ 추가
- 기존의 데이터 쌍으로 학습하는 것이 아닌 데이터와 기존 모형의 잔차를 이용해서 학습

전체 데이터 피쳐 학습모형

$(x_1\ y_1 - \boldsymbol{F(x_1)})$ x_1 →

$(x_2, y_2 - \boldsymbol{F(x_2)})$ x_2 →

$(x_3, y_3 - \boldsymbol{F(x_3)})$ x_3 → f(x)

⋮ ⋮

$(x_n, y_n - \boldsymbol{F(x_n)})$ x_4

$$L(y_i, F(x_i)) = \frac{1}{2}(y_i - F(x_i))^2$$ 손실 함수

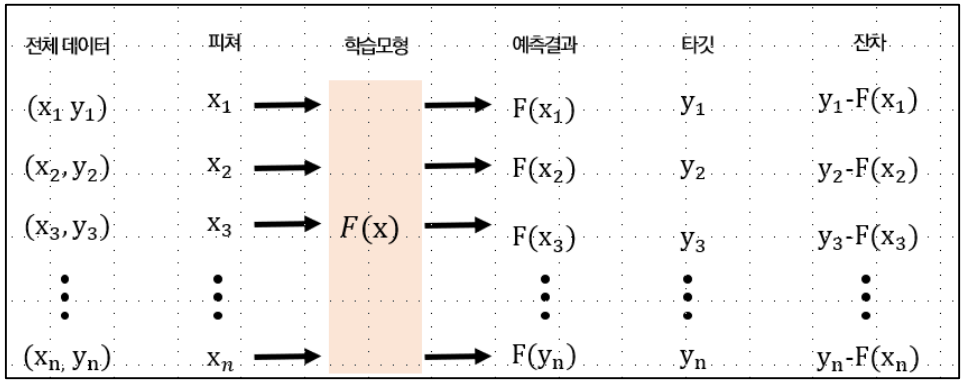
$$J = \sum_{i=1}^{n} L(y_i, F(x_i))$$ 목적 함수

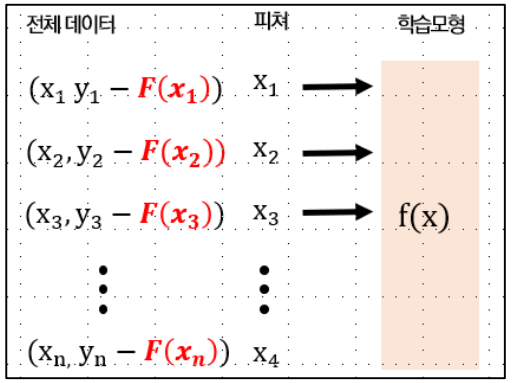
$$\frac{\delta J}{\delta F(x_i)} = \frac{\delta \sum_{i=1}^{n} L(y_i, F(x_i))}{\delta F(x_i)} = L(y_i, F(x_i)) = F(x_i) - y_i$$

Ensemble Learning

❖ Boosting – Gradient Boosting Machine

- Gradient 학습 과정
- m번째 모형 $F_m(x)$를 구하는 방법





$$L(y_i, F(x_i)) = \frac{1}{2}\left(y_i - F(x_i)\right)^2$$

$$J = \sum_{i=1}^{n} L\left(y_i, F(x_i)\right)$$

$$F_m(x) = F_{m-1}(x) + f(x)$$

$$= F_{m-1}(x) + y - F(x)$$

$$= F_{m-1}(x) + \frac{\delta J}{\delta F(x)} \rightarrow$$

Gradient!

$$\frac{\delta J}{\delta F(x_i)} = \frac{\delta \sum_{i=1}^{n} L(y_i, F(x_i))}{\delta F(x_i)} = \mathrm{L}\left(\mathrm{y_i}, \mathrm{F}(\mathrm{x_i})\right) = \mathrm{F}(\mathrm{x_i}) - \mathrm{y_i}$$

Ensemble Learning

❖ Boosting – XGBoost

- 극한의 효율성을 추구하는 알고리즘 ! (GBM은 느림)
- Split Finding Algorithm : 의사결정 나무에서의 basic exact greedy algorithm
- 장점: 가능한 모든 분기점을 전부 탐색하기 때문에 항상 최적의 분기점을 찾을 수 있음
- 단점: 데이터가 메모리에 저장되지 않을 경우 현실적으로 탐색이 불가능함 + 분산 컴퓨팅 환경에서 계산이 불가능

Algorithm 1: Exact Greedy Algorithm for Split Finding

Input: I, instance set of current node
Input: d, feature dimension
$gain \leftarrow 0$
$G \leftarrow \sum_{i \in I} g_i$, $H \leftarrow \sum_{i \in I} h_i$
for $k = 1$ *to* m **do**
  $G_L \leftarrow 0$, $H_L \leftarrow 0$
  for j *in sorted*$(I,$ *by* $\mathbf{x}_{jk})$ **do**
    $G_L \leftarrow G_L + g_j$, $H_L \leftarrow H_L + h_j$
    $G_R \leftarrow G - G_L$, $H_R \leftarrow H - H_L$
    $score \leftarrow \max(score, \frac{G_L^2}{H_L+\lambda} + \frac{G_R^2}{H_R+\lambda} - \frac{G^2}{H+\lambda})$
  end
end
Output: Split with max score

Ensemble Learning

❖ Boosting – XGBoost

- Approximate Algorithm for Split

Algorithm 2: Approximate Algorithm for Split Finding → 근사로 타협!

for $k = 1$ *to* m **do**
  Propose $S_k = \{s_{k1}, s_{k2}, \cdots s_{kl}\}$ by percentiles on feature k.
  Proposal can be done per tree (global), or per split(local). → 토막 낸다!
end
for $k = 1$ *to* m **do**
  $G_{kv} \leftarrow= \sum_{j \in \{j | s_{k,v} \geq \mathbf{x}_{jk} > s_{k,v-1}\}} g_j$
  $H_{kv} \leftarrow= \sum_{j \in \{j | s_{k,v} \geq \mathbf{x}_{jk} > s_{k,v-1}\}} h_j$
end
Follow same step as in previous section to find max score only among proposed splits.

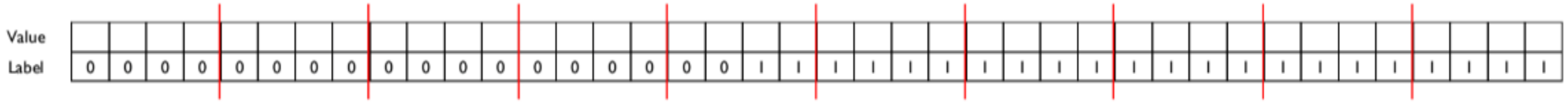
Ensemble Learning

❖ Boosting – XGBoost

- Approximate Algorithm for Split

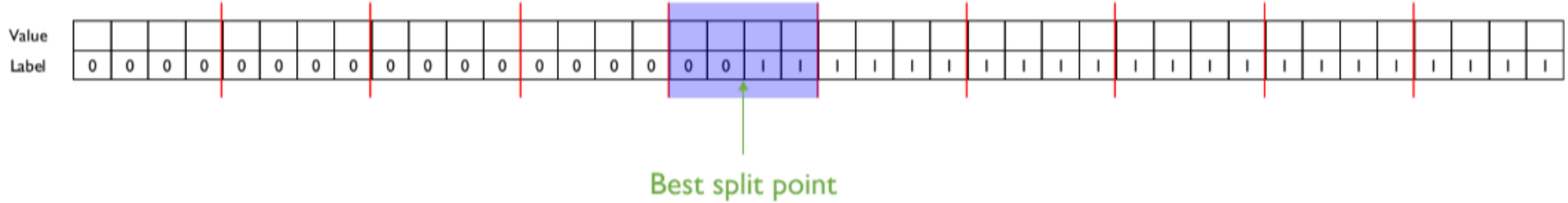
– 좌측으로부터 우측으로 변수 값이 오름차순으로 정렬되어 있다고 가정 (left: small, right: large)

– 전체 데이터를 10 분할



– 각 분할에 대해서 개별적으로 계산하여 최적 split을 찾음

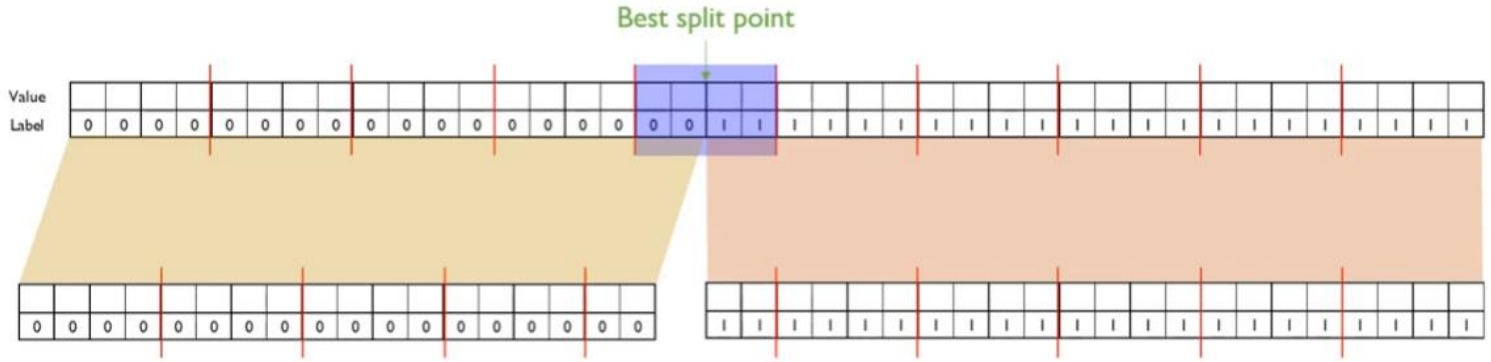
– 빠르지만 greedy algorithm에서 찾을 수 있는 최적의 분기점은 찾기 어려움



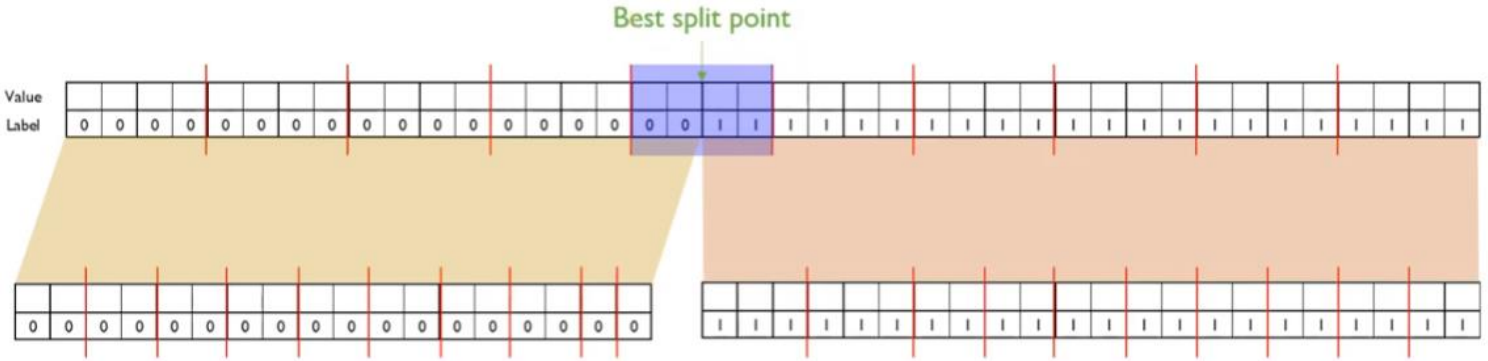
Ensemble Learning

❖ Boosting – XGBoost

- Approximate Algorithm for Split
- Global vs Local Variant



Global Variant
탐색해야되는 버킷의 개수는 줄어들고
데이터의 수는 동이하게 유지

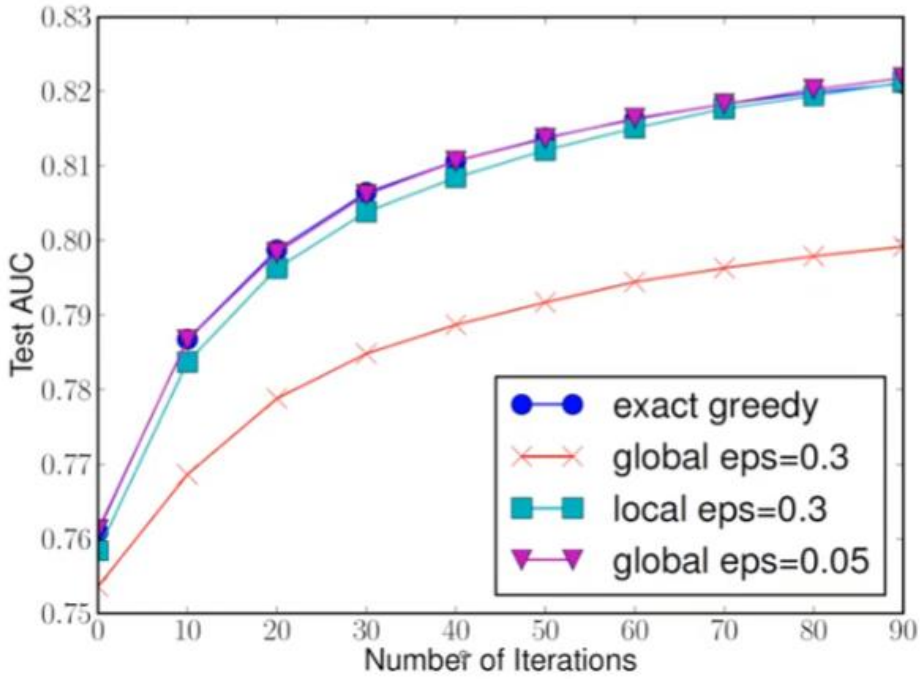


Local Variant:
자식 노드에도 버켓 사이즈를 유지
스플릿이 되더라도 버켓 사이즈가 유지
-> 각 버켓에 들어가는데이터는 감소

Ensemble Learning

❖ Boosting – XGBoost

- Approximate Algorithm for Split
- Global vs Local Variant



Epsilon = 버켓을 나누는 기준
Ex) eps = 0.3 이면 버켓은 3개, 0.05라면 버켓은 20개 생성 !

Global이 높은 성능에 도달하려면 실제로 local보다 더 많은 버켓을 생성해야됨

Ensemble Learning

❖ Boosting – LightGBM

- GBM의 최대 단점: 오래걸린다!
- 전통 GBM은 모든 데이터의 모든 피처를 탐색 후 정보 획득률을 조사
- LGBM은 모든 데이터에 대한 탐색과 모든 피처에 대한 완전 탐색을 완화 (버켓 개수를 사용한 XGBoost와는 독립)

1. Gradient-based One side Sampling(GOSS)

일부만 사용!

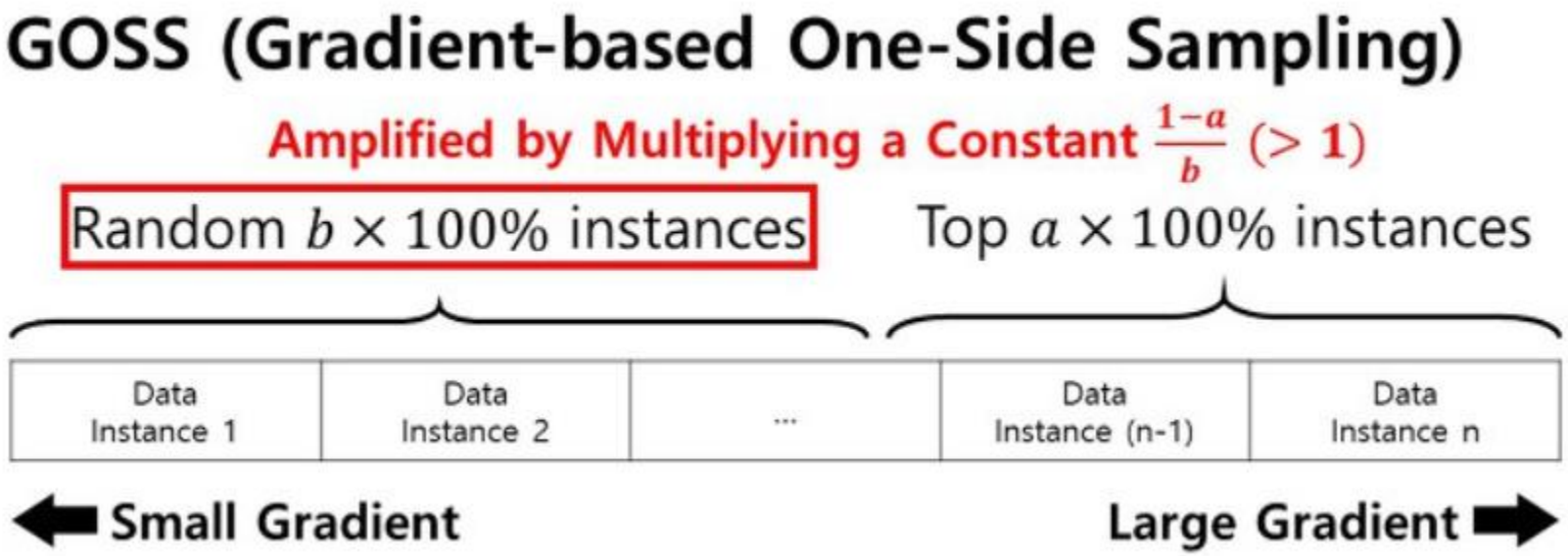
2. Exclusive Feature Bundling (EFB)

합칠 수 있는 애들 합치자!

Ensemble Learning

❖ Boosting – LightGBM

- Gradient-based One side Sampling(GOSS)
- 매우 쉬운 방법이지만 하이퍼파라미터 (a,b 선택 주의)



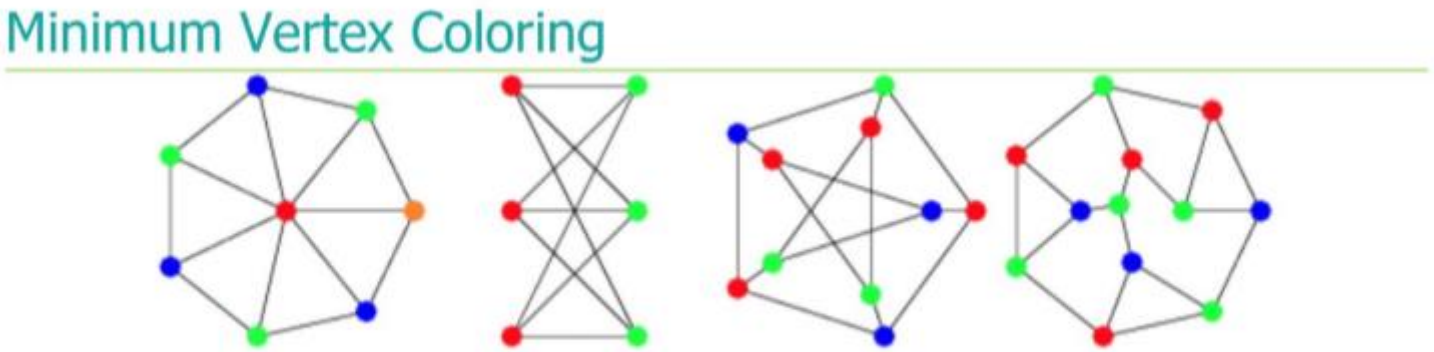
낮은 gradient = b 만큼 적게 사용

높은 gradient = a 만큼 많이 사용

Ensemble Learning

❖ Boosting – LightGBM

- Exclusive Feature Bundling (EFB)
- 배타적인 변수를 합쳐서 새로운 feature를 만들자



인접한 node는 같은 색을 사용하지 않을 때, 전부 색칠하는 최소한의 색은?

Ensemble Learning

❖ Boosting – LightGBM

- Exclusive Feature Bundling (EFB)
- 배타적인 변수를 합쳐서 새로운 feature를 만들자

Greedy Bundling

	x_1	x_2	x_3	x_4	x_5
I_1	1	1	0	0	1
I_2	0	0	1	1	1
I_3	1	2	0	0	2
I_4	0	0	2	3	1
I_5	2	1	0	0	3
I_6	3	3	0	0	1
I_7	0	0	3	0	2
I_8	1	2	3	4	3
I_9	1	0	1	0	0
I_{10}	2	3	0	0	2

X1, x2가 동시에 0이 아닌 값을 가지는 instance의 수

	x_1	x_2	x_3	x_4	x_5
x_1	-	6	2	1	6
x_2	6	-	1	1	6
x_3	2	1	-	3	4
x_4	1	1	3	-	3
x_5	6	6	4	3	-

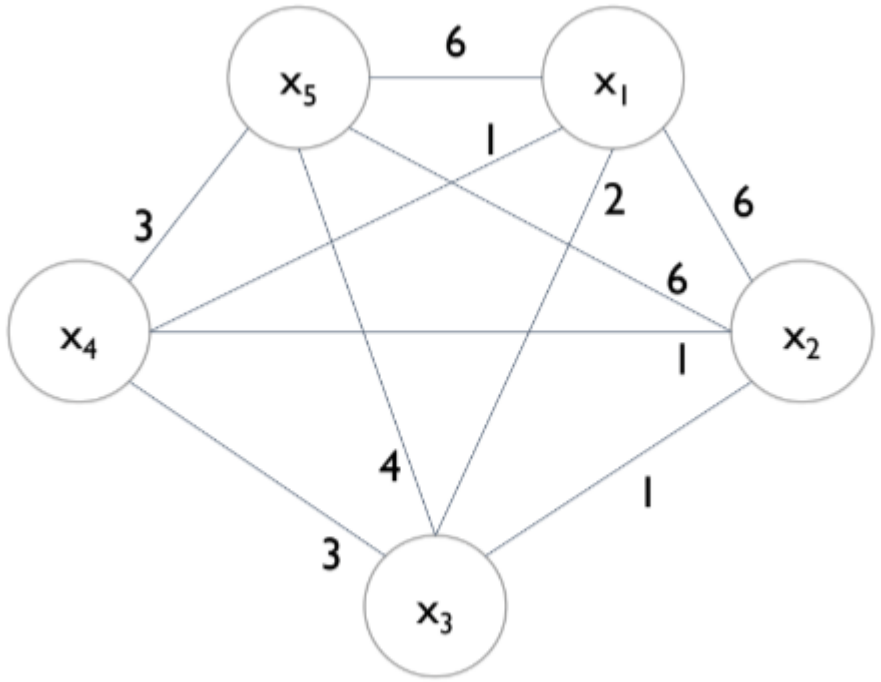
Column sum = degree (기준)

	x_5	x_1	x_2	x_3	x_4
d	19	15	14	10	8

Ensemble Learning

❖ Boosting – LightGBM

- Exclusive Feature Bundling (EFB)
- 배타적인 변수를 합쳐서 새로운 feature를 만들자



Greedy bundling example (cut-off = 0.2)

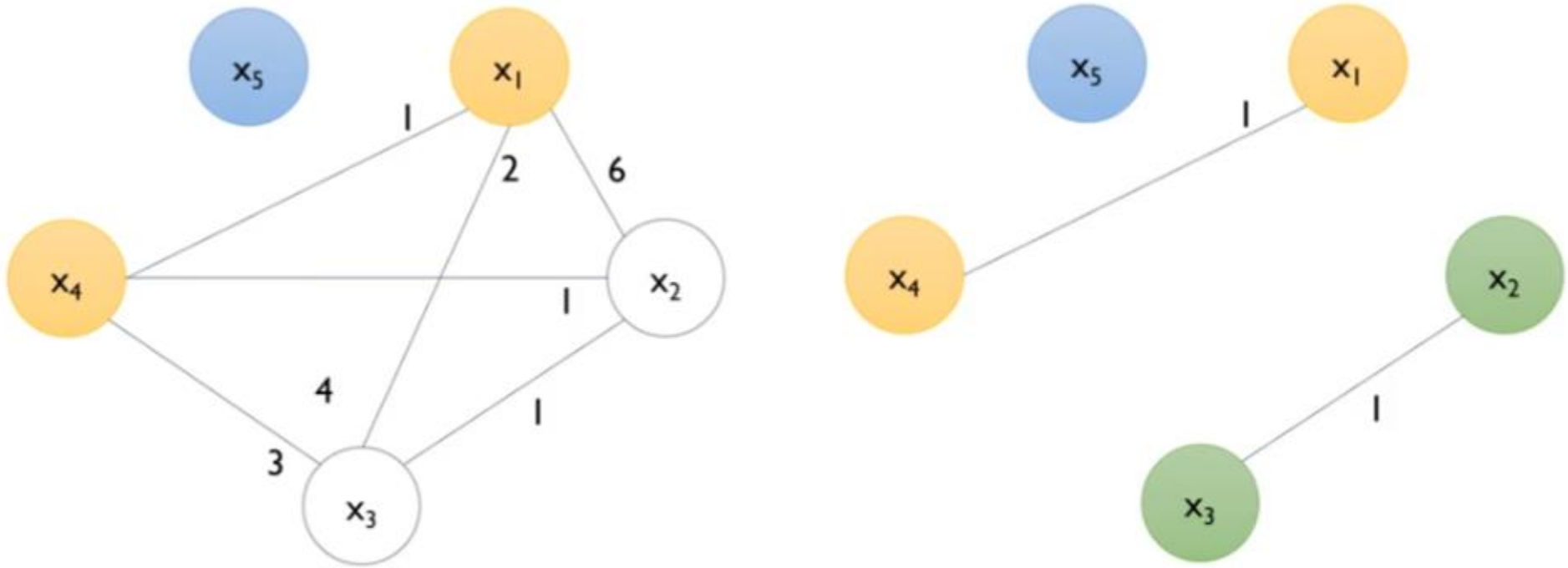
N = 100이면 10*0.2 = 2 가 기준이 됨 !

즉, 2번 이상 충돌하면 bundle하지 않겠다

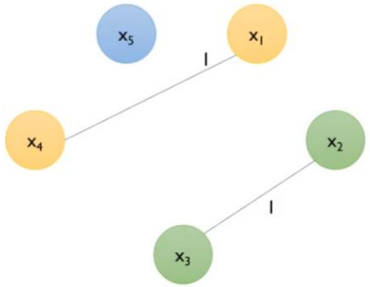
Ensemble Learning

❖ Boosting – LightGBM

- 배타적인 변수를 합쳐서 새로운 feature를 만들자



Ensemble Learning



❖ Boosting – LightGBM

- 배타적인 변수를 합쳐서 새로운 feature를 만들자

	x_5	x_1	x_4	x_2	x_3
I_1	1	1	0	1	0
I_2	1	0	1	0	1
I_3	2	1	0	2	0
I_4	1	0	3	0	2
I_5	3	2	0	1	0
I_6	1	3	0	3	0
I_7	2	0	0	0	3
I_8	3	1	4	2	3
I_9	0	1	0	0	1
I_{10}	2	2	0	3	0

	x_5	x_{14}	x_{23}
I_1	1	1	1
I_2	1	4	4
I_3	2	1	2
I_4	1	6	5
I_5	3	2	1
I_6	1	3	3
I_7	2	0	6
I_8	3	1	2
I_9	0	1	4
I_{10}	2	2	3

기준은 왼쪽

X1 의 offset (가장 큰 값) = 3

X2 의 offset (가장 큰 값) = 3

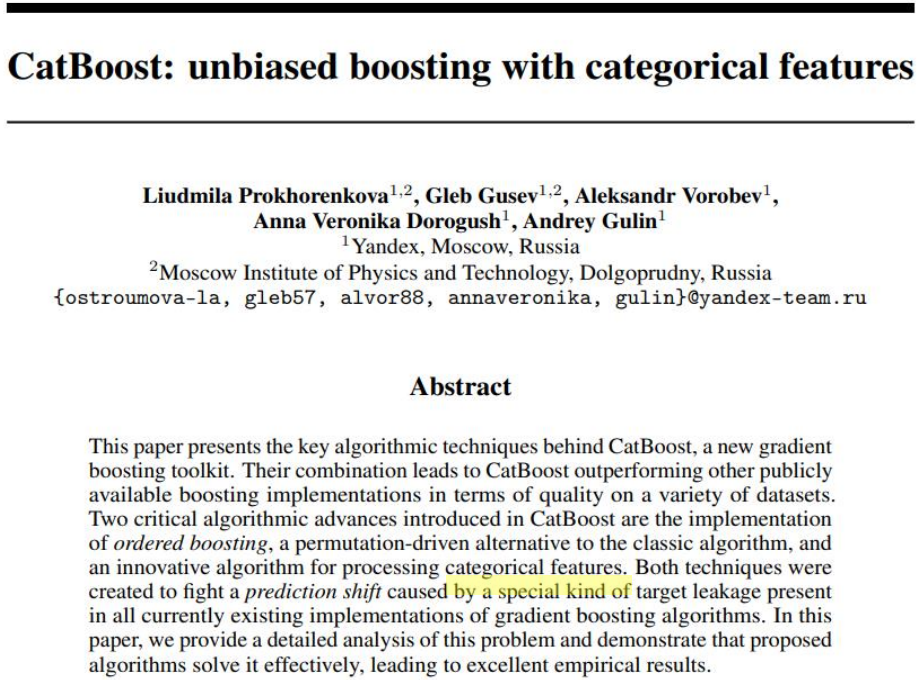
둘다 0인 경우 0

둘다 0이 아닌 경우 왼쪽 값

Ensemble Learning

❖ Boosting – Catboost

- 범주형(categorical) 데이터 처리에 효과적인 boosting 모델: Prediction Shift 해결
- 1. Ordered Boosting
- 2. Processing Categorical Features



CatBoost: unbiased boosting with categorical features

Liudmila Prokhorenkova[1,2], **Gleb Gusev**[1,2], **Aleksandr Vorobev**[1], **Anna Veronika Dorogush**[1], **Andrey Gulin**[1]
[1]Yandex, Moscow, Russia
[2]Moscow Institute of Physics and Technology, Dolgoprudny, Russia
{ostroumova-la, gleb57, alvor88, annaveronika, gulin}@yandex-team.ru

Abstract

This paper presents the key algorithmic techniques behind CatBoost, a new gradient boosting toolkit. Their combination leads to CatBoost outperforming other publicly available boosting implementations in terms of quality on a variety of datasets. Two critical algorithmic advances introduced in CatBoost are the implementation of *ordered boosting*, a permutation-driven alternative to the classic algorithm, and an innovative algorithm for processing categorical features. Both techniques were created to fight a *prediction shift* caused by a special kind of target leakage present in all currently existing implementations of gradient boosting algorithms. In this paper, we provide a detailed analysis of this problem and demonstrate that proposed algorithms solve it effectively, leading to excellent empirical results.

Ensemble Learning

❖ Boosting – Catboost

- Prediction Shift: 우리가 예측하고자 하는 변수가 데이터셋에 있는 데이터에 가까워지고 실제 값과는 멀어질 때 발생

$$\arg\min_{h \in H} \mathbb{E}\left(-g^t(\mathbf{x}, y) - h(\mathbf{x})\right)^2 \approx \arg\min_{h \in H} \frac{1}{n} \sum_{k=1}^{n} \left(-g^t(\mathbf{x}_k, y_k) - h(\mathbf{x}_k)\right)^2$$

$$\text{Training Dataset : } \mathcal{D} = (\mathbf{x}_k, y_k)_{k=1,\dots,n}$$

$$\text{where } \mathbf{x}_k = (x_k^1, \dots, x_k^m), \quad y_k \in \mathbb{R}$$

Conditional distribution $F^{t-1}(\mathbf{x}_k)|\mathbf{x}_k$ for training example $\mathbf{x}_k$

is SHFTED from

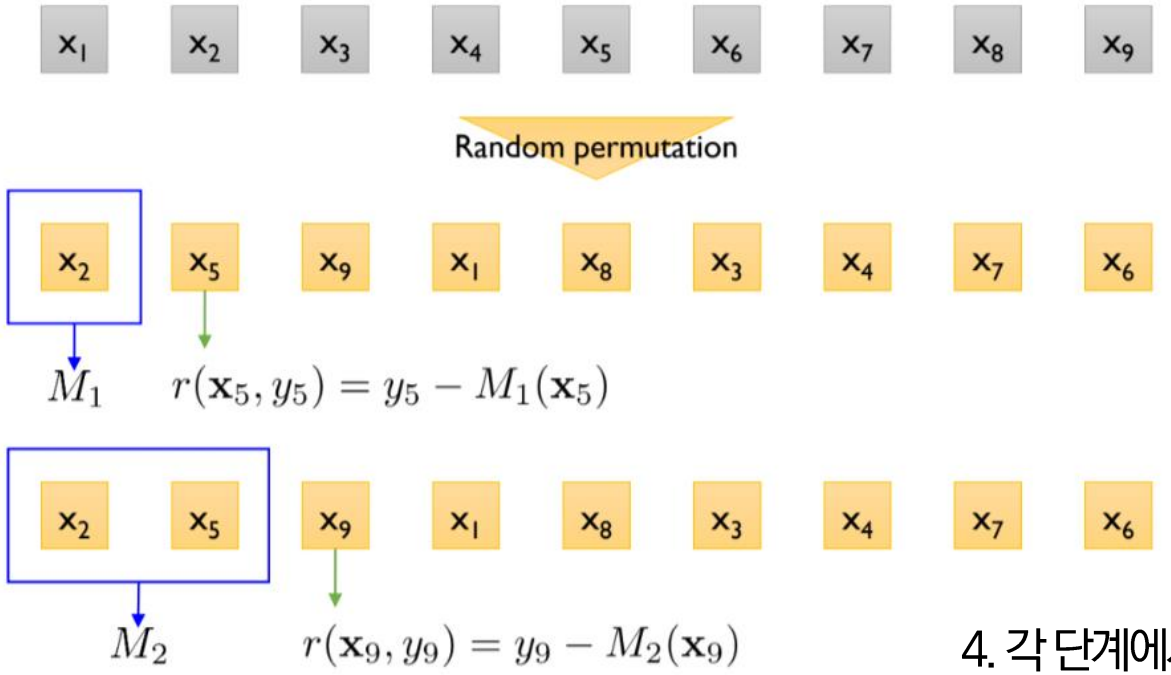
$\neq$

Conditional distribution $F^{t-1}(\mathbf{x})|\mathbf{x}$ for test example $\mathbf{x}$

Ensemble Learning

❖ Boosting – Catboost

- 1. Ordered Boosting
- 현실적으로 레이블이 있는 데이터는 제한되어 있음. 현실적으로 편향되지 않은 잔차를 사용하기 어려움
- 모델 훈련에 사용된 예제들에 따라 다양한 모델 집합을 유지하는 것이 가능 → 순서형 원칙 (ordering principle)



1. 무작위 순열을 가진다고 가정

2. 서로 다른 n개의 모델 M 생성

3. Mi는 순열의 처음 i번째 예제만 사용

4. 각 단계에서 표본의 잔차 얻기 위해 모델 Mi-1 사용

Ensemble Learning

❖ Boosting – Catboost

- 2.Categorical features
- 범주형 피처는 서로 비교가 불가능한 이산값의 집합
- Catboost에서는 one hot encoding과 비슷한 방식을 사용 (y값의 평균을 대체해서) → overfitting 문제
- overfitting을 방지하기 위해 Ordered TS(target statistics)를 사용해 범주형 피처를 처리

✓ Ordered TS: introduce an artificial time

- a random permutation σ of the training examples

$$\mathcal{D}_k = \{\mathbf{x}_j : \sigma(j) < \sigma(k)\}$$

- a = 0.1 (parameter) , p = 0 (computed from the training dataset)

	…	x^i	…	TS	y
I_1	…	A	…	0.000	1
I_2	…	B	…		1
I_3	…	C	…		1
I_4	…	A	…		0
I_5	…	B	…		1

$$\hat{x}_k^i = \frac{\sum_{\mathbf{x}_j \in \mathcal{D}_k} \mathbb{1}_{\{x_j^i = x_k^i\}} \cdot y_j + ap}{\sum_{\mathbf{x}_j \in \mathcal{D}_k} \mathbb{1}_{\{x_j^i = x_k^i\}} + a}$$

$$= \frac{0 + 0.1 \times 0}{0 + 0.1} = 0$$

Ensemble Learning

❖ Boosting – Catboost

안녕하세요
제3회 빅데이터•인공지능 스타트업 경진대회 운영사무국입니다.

다시 한 번 1차 심사 결과 2차 발표평가 대상팀으로 선정되신 것을 진심으로 축하드립니다.
오리엔테이션에서 안내드린 대로 최종 확정된 발표 일정을 보내드리오니
확인 부탁드리겠습니다.

" 가스공급형 수요예측 모델개발 "주제

구분	공급량
A	2497.129
A	2363.265
A	2258.505
A	2243.969
A	2344.105
A	2390.961
A	2378.457
A	2518.921
A	2706.481

Thank you