

정 규 세션 3 주차

ToBig's 10기 이준걸

Ensemble2

Bagging, Boosting, XGBOOST, LightGBM, Stacking

Content

Unit 01 | Ensemble Model Overview & Voting Classifier

Unit 02 | Bagging, Boosting

Unit 03 | AdaBoost, Gradient Boosting

Unit 04 | XGBoosting, GBM & LightGBM

Unit 05 | Stacking

Unit 01 | Ensemble Model OverView

성능을 어떻게 올릴 것인가?
&
no free lunch Theorem

Unit 01 | Ensemble Model OverView

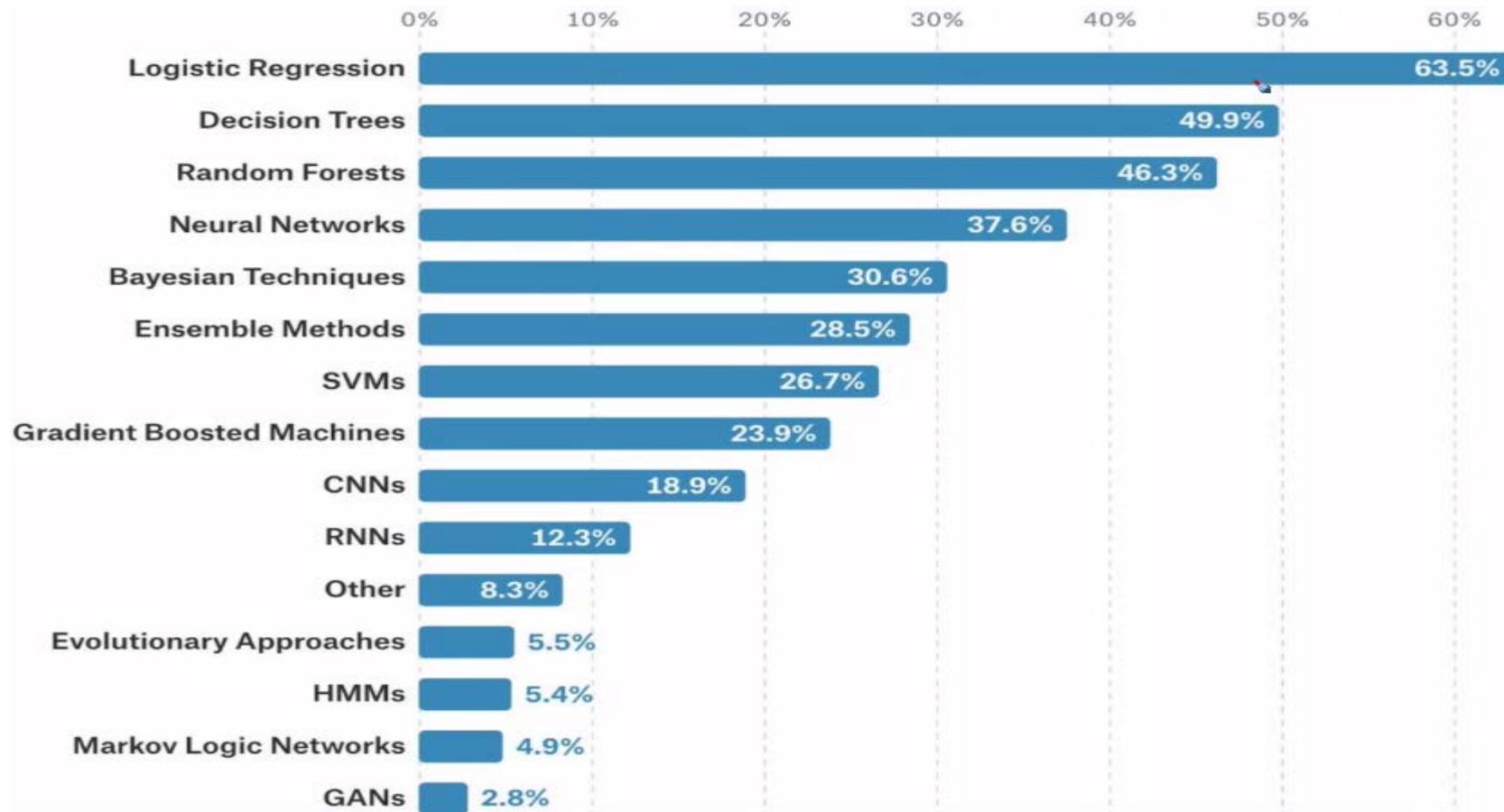
no free lunch Theorem

We have dubbed the associated results “No Free Lunch” theorems because they demonstrate that if an algorithm performs well on a certain class of problems then it necessarily pays for that with degraded performance on the set of all remaining problems. ('No Free Lunch Theorem')

-> 간단히 말해 특정한 문제에 최적화된 알고리즘은 다른 문제에서 는 그렇지 않다는 것을 수학적으로 증명한 정리이다.

Unit 01 | Ensemble Model OverView

Kaggle에서 많이 쓰이는 모델들



Unit 01 | Ensemble Model OverView

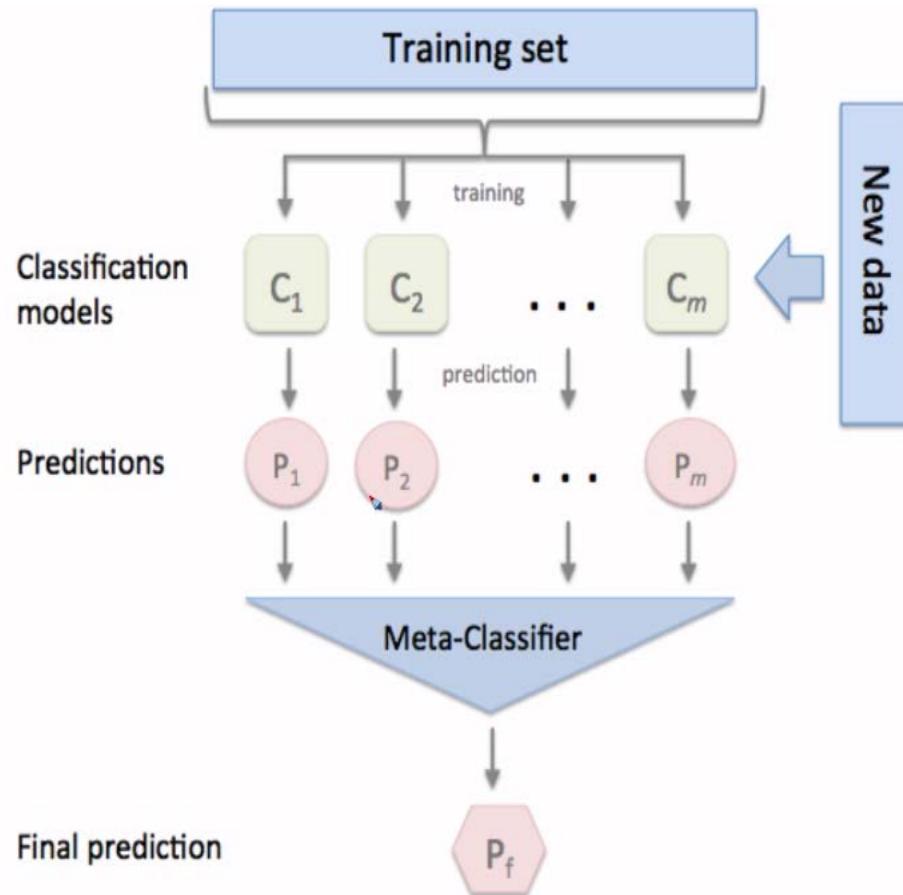
Ensemble Model OverView

- 하나의 모델이 아니라 여러 개의 모델의 투표로 Y값 예측
- Regression 문제에서는 평균값으로 예측
- Meta – Classifier : 여러 모델의 연계를 통해 문제 해결
- Stacking(Meta – ensemble) : 앙상블을 여러 개를 섞어서 연결해 문제 해결
- 학습은 오래 걸리나 성능이 매우 매우 좋음!
- Kaggle에서 Structured Dataset에서의 대세 기법



Unit 01 | Ensemble Model OverView

Meta - Classifier



ex1) 모두 다른 모델

C_1 : Logistic Regression

C_2 : KNN

C_3 : SVM

...

C_m : DT

ex2) 모두 같은 모델

(단 hyperparameter는 다른)

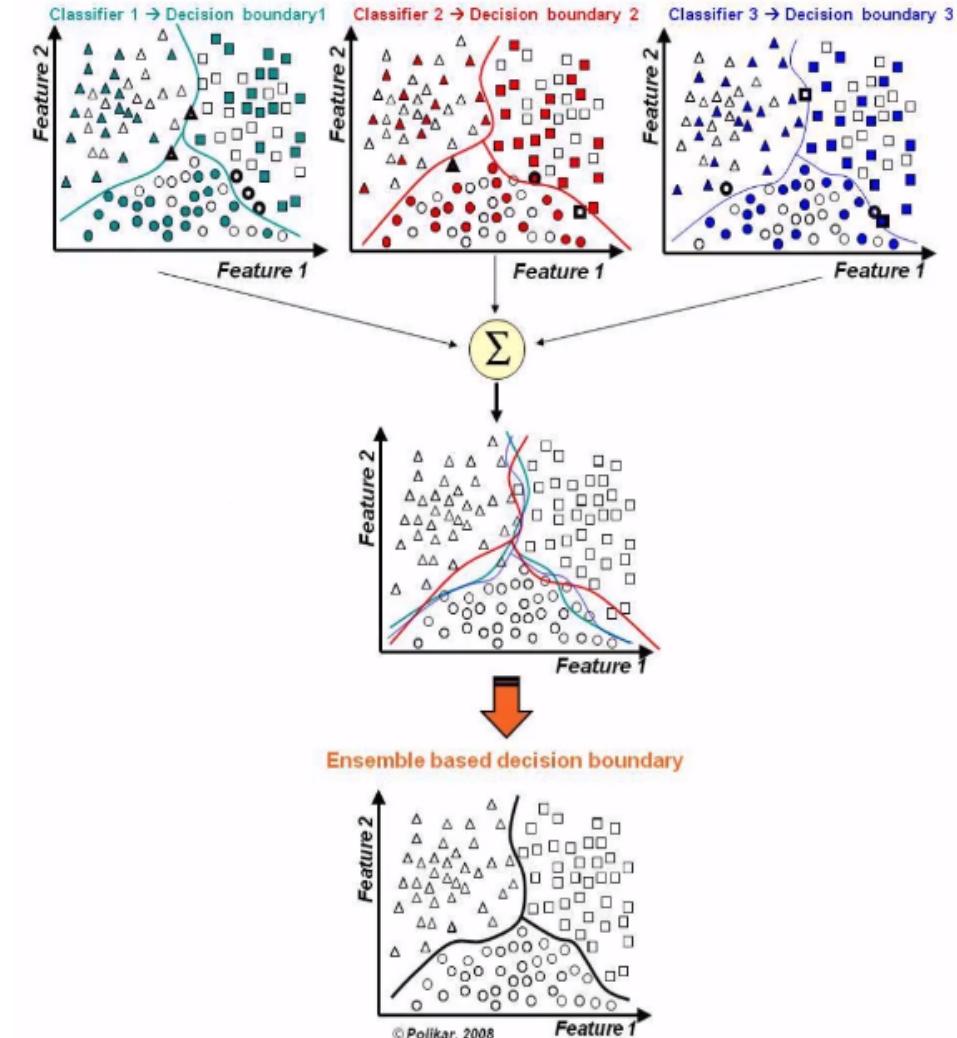
C_1 : DT

C_2 : DT

C_3 : DT

...

C_m : DT



Unit 01 | Ensemble Model OverView

오늘 배울 내용은?

- Vanilla Ensemble : 기본적인 Ensemble(분류 – Voting, 회귀 – Mean, Median)
- Bagging : Data Sampling 1
- Boosting : Data Sampling 2
- Adaptive Boosting (AdaBoost)
- XGBoost
- Light GBM



Boosting의 발전된 Model

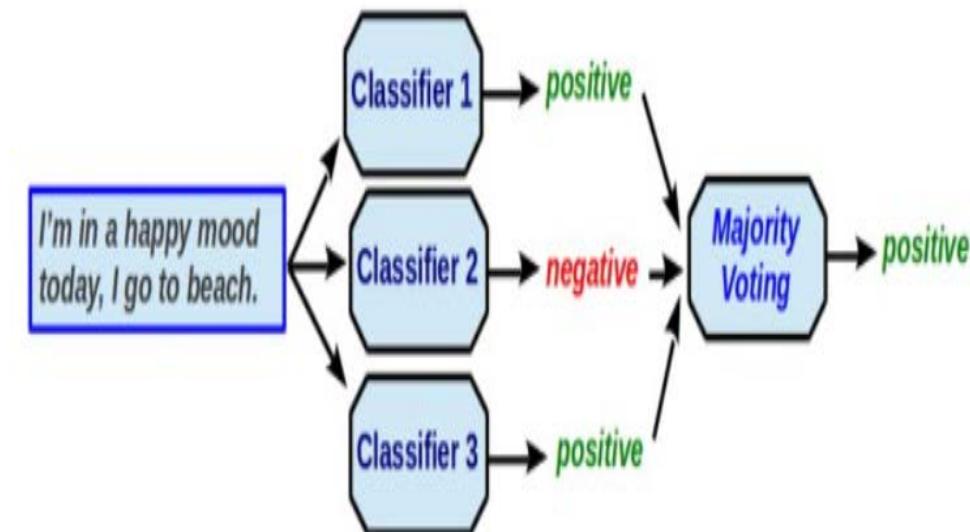
Unit 01 | Voting Classifier

Voting Classifier = Majority Voting = Vanilla Ensemble

- 가장 기본적인 Ensemble Classifier
- 여러 개의 Model의 Voting을 통해 예측

Code

```
clf1 = LogisticRegression(random_state = 1)
clf2 = DecisionTreeClassifier(random_state = 1)
clf3 = GaussianNB()
Eclf = VotingClassifier(
    Estimators = [('lr', clf1), ('dt', clf2), ('gnb', clf3)],
    Voting = 'hard')
```



Unit 01 | Voting Classifier

[실습1-Voting_Ensemble] 켜주세요!

Unit 01 | Ensemble Model Overview & Voting Classifier

Unit 02 | Bagging – Random Forest

Unit 03 | Boosting – AdaBoost, Gradient Boosting

Unit 04 | XGBoosting, GBM & LightGBM

Unit 05 | Stacking

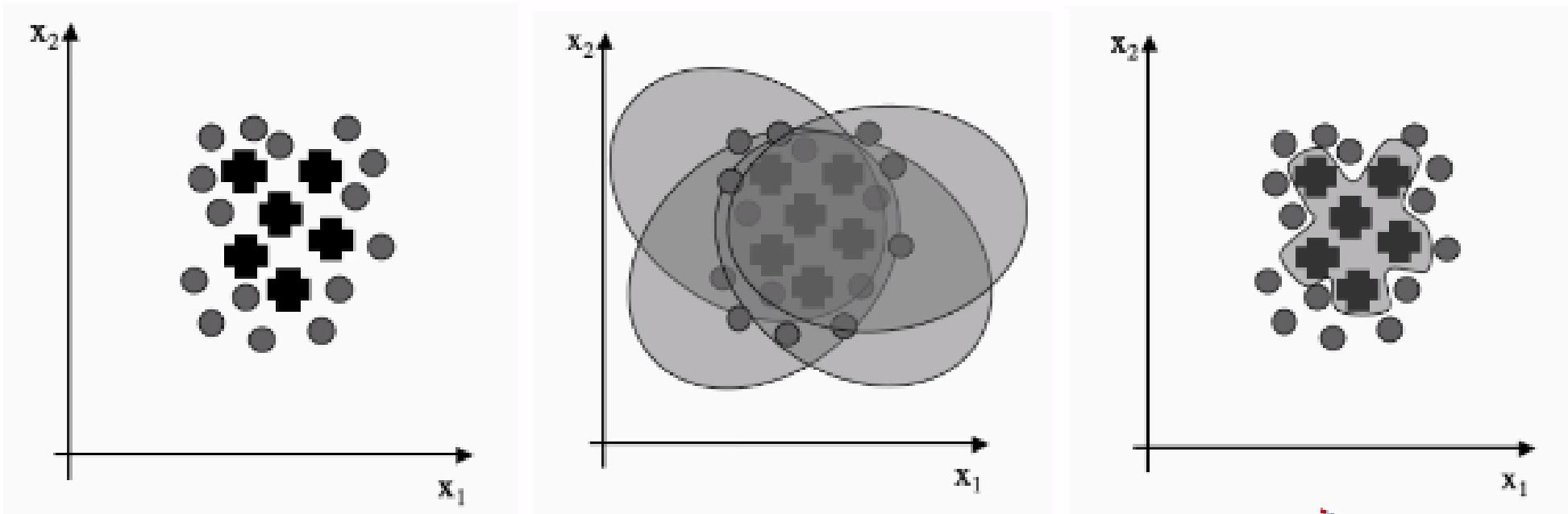
Unit 02 | Bagging

What is Bagging? – Sampling

- 단순히 같은 데이터 셋으로 만드는 Classifier는 의미가 없다. Ex) 같은 Dataset으로 만든 여러 개의 Tree를 생각해 보면 매우 비슷하게 만들어질 것이므로 성능에 영향을 미치지 않는다.
- 우리가 보고 있는 데이터는 매우 큰 부분의 일부에 지나지 않는다. 마찬가지로 우리가 갖고 있는 샘플 자체를 하나의 모집단이라고 생각을 하고 이 모집단을 샘플링을 해서 모델을 만든다면 매우 Robust한 모델을 만들 수 있다.
- 즉, 다양한 Sampling Dataset으로 다양한 Classifier를 만들어 보자!!

Unit 02 | Bagging

Ensemble Sampling

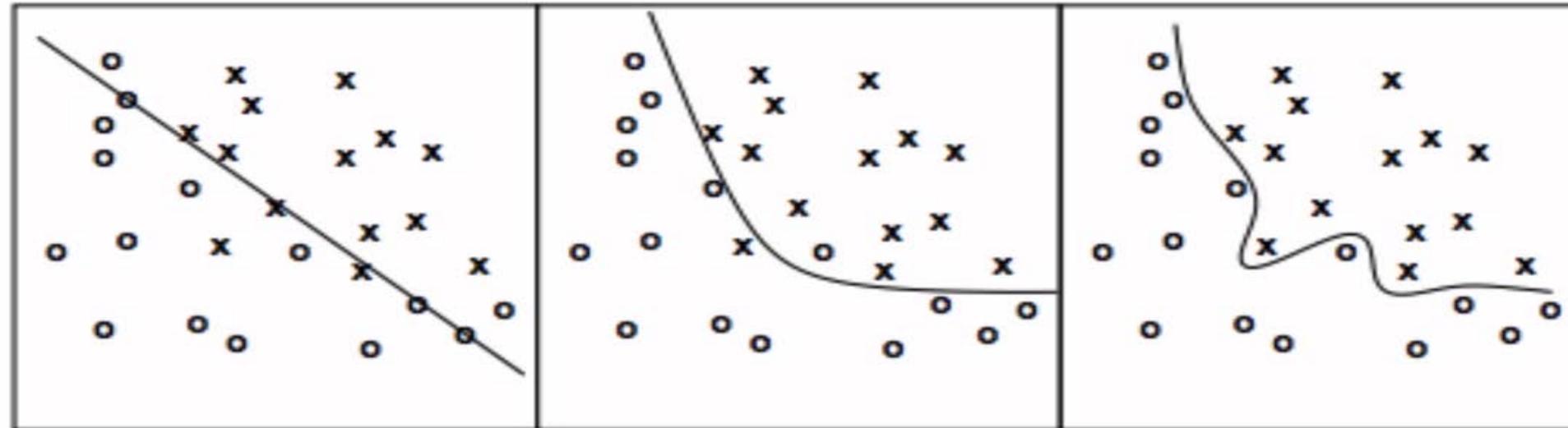


→ 분류기를 여러 개 모아서 학습을 시키면 강한 분류기를 만들 수 있다.

Unit 02 | Bagging

Variance 관점

“The most likely hypothesis is the simplest one consistent with the data.”



inadequate

good compromise

over-fitting

-> 데이터의 샘플링은 Variance를 줄여준다. 따라서 모델을 사용할 때 Variance가 높은 모델(Neural Net, Decision Tree)을 사용하게 되면 효과가 극대화가 된다.

Unit 02 | Bagging

Bootstrapping

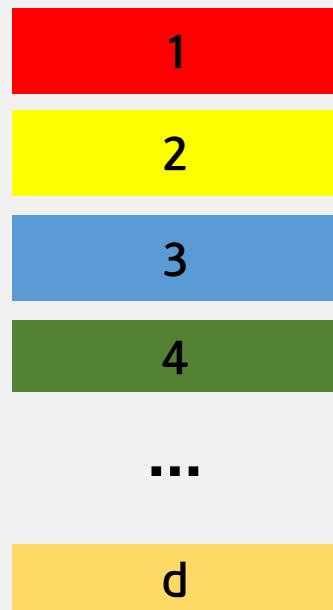
- 데이터를 외부 추가 없이 추출하는 것 -> 외부의 input 없이 구성된 데이터셋으로만 성능을 최대화 시키는 방법.
- 학습데이터 Subset을 구성하여 n개를 추출하자! Ex) 밑에는 $n = 3$



Unit 02 | Bagging

.632 Bootstrap

Observed Data



나눠진 subset을 d번 뽑는 시행을 있다고 하자.

한 데이터가 뽑힐 확률 : $\frac{1}{d}$ / 뽑히지 않을 확률 : $1 - \frac{1}{d}$

적어도 한번 뽑힐 확률 :

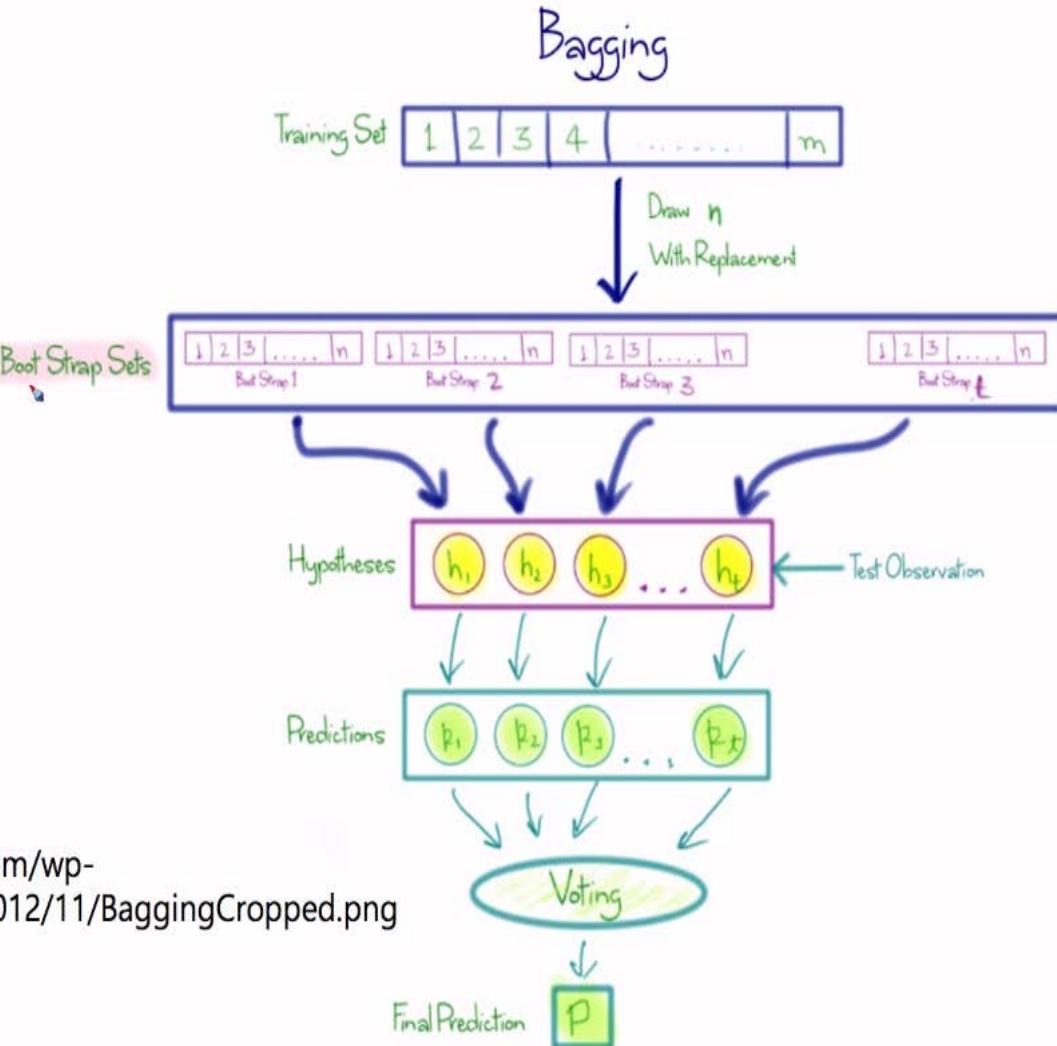
$$1 - \prod\left(1 - \frac{1}{d}\right) = 1 - \left(1 - \frac{1}{d}\right)^d \approx 1 - e^{-1}$$

$$= 0.632$$

Unit 02 | Bagging

Bagging

- Bootstrap의 Subset Sample로 **동일한** 모델 n개를 학습
→ 양상불
- High Variance(Overfitting이 심함) 모델이 적합
- Regressor(평균 Or 중위수), Classifier 모두 존재



<http://manish-m.com/wp-content/uploads/2012/11/BaggingCropped.png>

Unit 02 | Bagging

Bagging = Bootstrap Aggregation

- Bootstrap의 Subset Sample로 **동일한** 모델 n개를 학습 → 양상을
- High Variance(Overfitting이 심함) 모델이 적합
- Regressor(평균 Or 중위수), Classifier 모두 존재

Unit 02 | Bagging

Out of Bag error = OOB error

- Bagging 실행 시, Bag에 미포함 데이터로 성능 측정
- Validation set, K-fold 와 처리하는 방법과 유사
- Bagging 성능 측정을 위한 굉장히 좋은 지표

Unit 02 | Bagging

[실습2-Bagging] 켜주세요!

Unit 02 | RandomForest

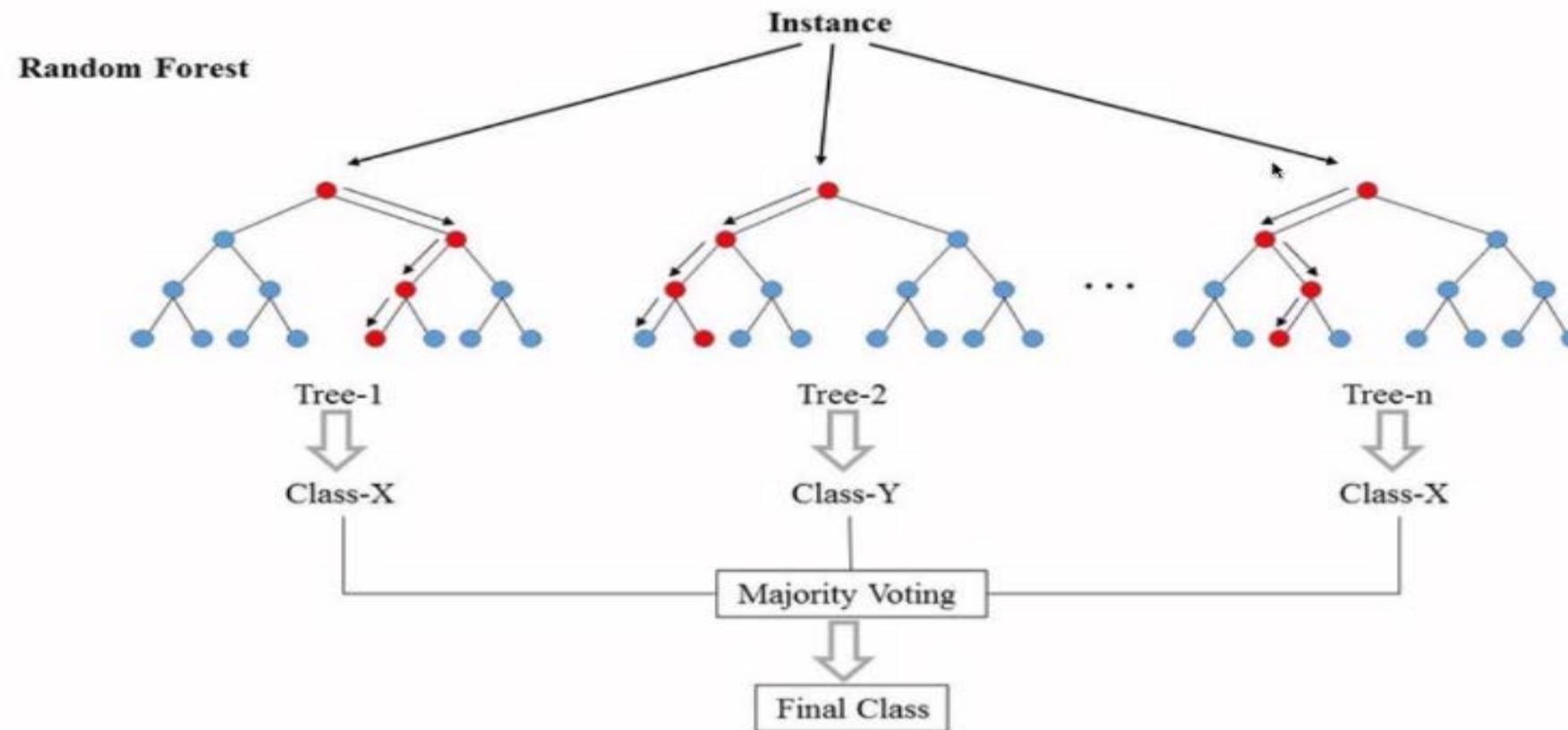
What is Random Forest?

- Bagging + Randomized Decision Tree
- Variance가 높은 DT들의 Ensemble
- 가장 간단하면서 높은 성능을 보이는 사기적인 모델
- Regression과 Classifier 모두 가능



Unit 02 | Bagging

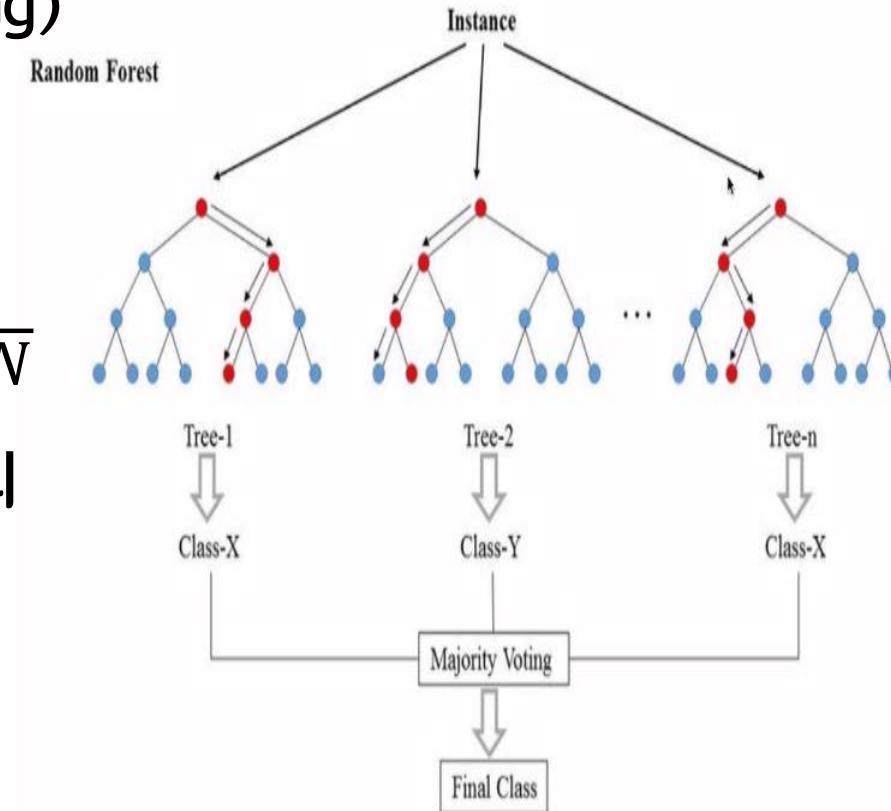
Random Forest



Unit 02 | RandomForest

Random Forest

- Correlation이 낮은 m개의 Subset Data로 학습(= Bagging)
- Tree의 구성은 Binary로 구성
- Subset Data를 만들 때 Feature도 Bagging을 적용하여 N 개를 selection 또한 Feature의 재사용이 가능하고 보통 \sqrt{N} 로 Subset을 만든다. (Feature가 K개 있을 때 Subset Data의 Feature가 K=N이면 Bagging)
- Variance가 높은 트리가 좋다.



Unit 02 | RandomForest

Random Forest

```
class sklearn.ensemble. RandomForestRegressor (n_estimators=10, criterion='mse', max_depth=None,  
min_samples_split=2, min_samples_leaf=1, min_weight_fraction_leaf=0.0, max_features='auto',  
max_leaf_nodes=None, min_impurity_decrease=0.0, min_impurity_split=None, bootstrap=True, oob_score=False,  
n_jobs=1, random_state=None, verbose=0, warm_start=False) ¶ [source]
```

max_features : int, float, string or None, optional (default="auto")

The number of features to consider when looking for the best split:

- If `int`, then consider `max_features` features at each split.
- If `float`, then `max_features` is a percentage and `int(max_features * n_features)` features are considered at each split.
- If "auto", then `max_features=sqrt(n_features)`.
- If "sqrt", then `max_features=sqrt(n_features)` (same as "auto").
- If "log2", then `max_features=log2(n_features)`.
- If `None`, then `max_features=n_features`.

Unit 02 | RandomForest

[실습3-Randomforest] 켜주세요!

Unit 01 | Ensemble Model Overview & Voting Classifier

Unit 02 | Bagging – Random Forest

Unit 03 | Boosting – AdaBoost, Gradient Boosting

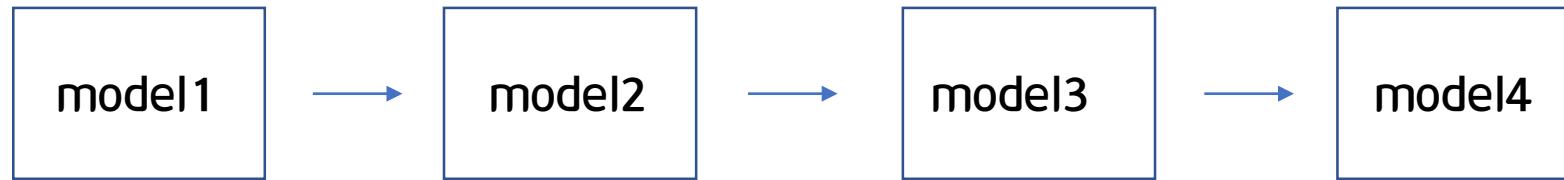
Unit 04 | XGBoosting, GBM & LightGBM

Unit 05 | Stacking

Unit 03 | Boosting

What is Boosting?

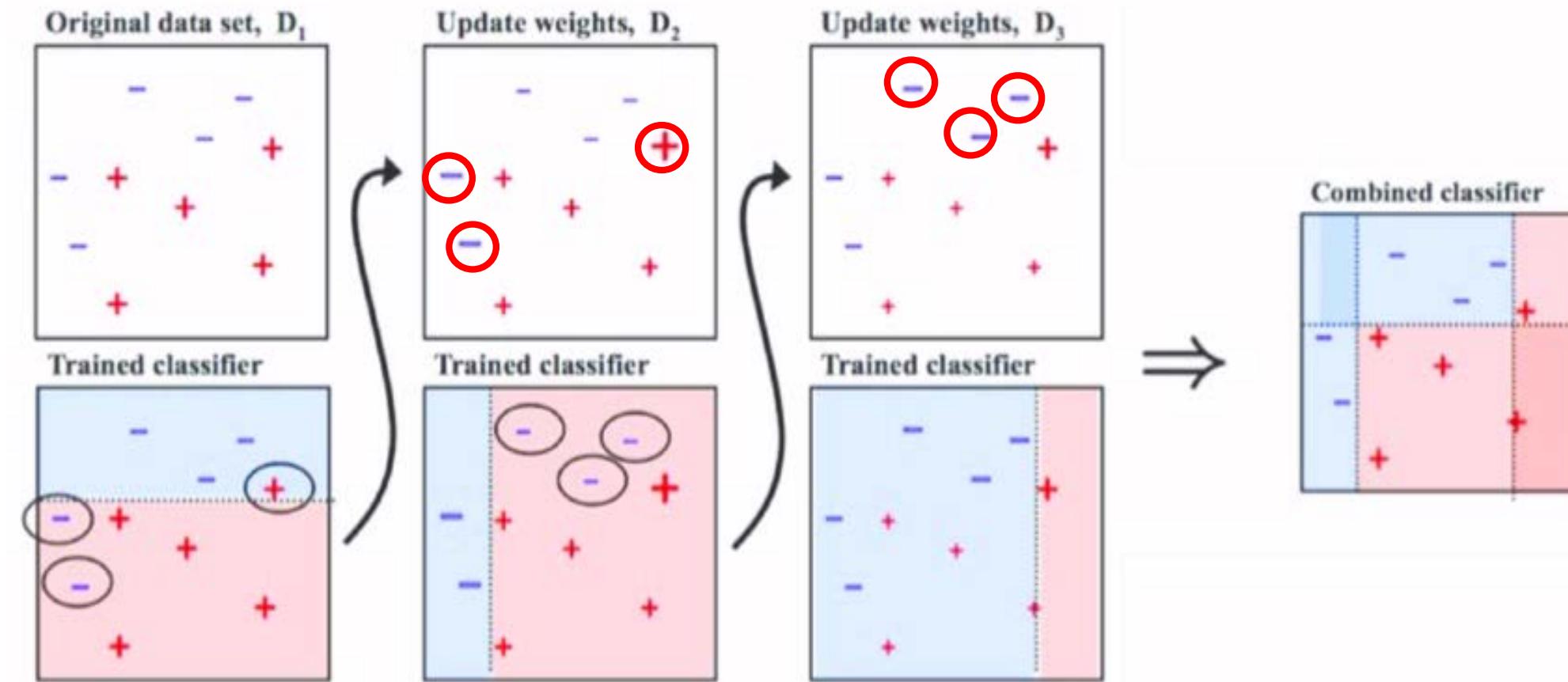
- 학습 Round을 진행하면서 모델을 생성 -> 모델에 의해 각 row의 Weight를 업데이트 함.



- Instance Weight가 높은(오분류된) row를 중심으로 모델을 생성
- 해당 모델들로 양상을 모델을 만듦
- 잘못 분류된 데이터를 더 잘 분류해보는 것이 목적

Unit 03 | Boosting

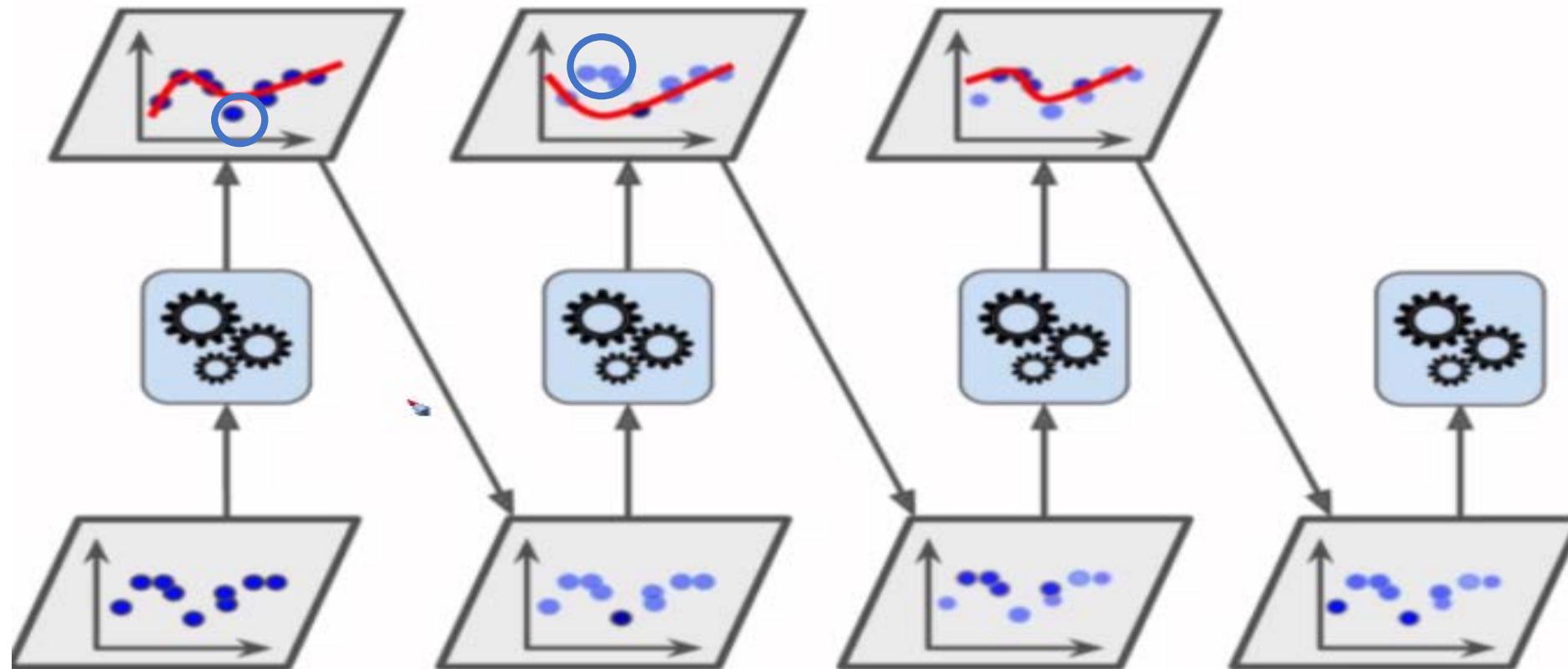
Boosting - Classifier



오분류된
Instance에 가중
치를 줘서 완벽
해져가는 모델.

Unit 03 | Boosting

Boosting – Regression



Unit 03 | AdaBoost

What is Adaboost?

- Adaptive Boosting = 에이다 부스트
- 매 Round마다 Instance의 Weight값을 계산
- 틀리는 Instance의 Weight up -> weight 기준 Resampling
- Instance의 Weight합이 클수록, Model의 Weight를 줄임
- 약한 분류기에 입력값을 변화시켜가며 강한 분류기를 만듦(Weak learner)
- High-Depth DT, NN (High Variance Model)에는 적합하지 않음

Unit 03 | AdaBoost

Adaboost

Algorithm 10.1 AdaBoost.M1.

값 샘플의 weight를 $1/N$ 로 초기화

1. Initialize the observation weights $w_i = 1/N$, $i = 1, 2, \dots, N$.

N 은 데이터의 갯수

2. For $m = 1$ to M : **M은 모델의 갯수**

Weight값을 기준으로 분류기 생성(resampling)

(a) Fit a classifier $G_m(x)$ to the training data using weights w_i .

(b) Compute 해당 분류기의 에러 계산

$$I(y_i, G_m(x_i)) = \begin{cases} 0 & \text{if } y_i = G_m(x_i) \\ 1 & \text{if } y_i \neq G_m(x_i) \end{cases}$$

$$\text{err}_m = \frac{\sum_{i=1}^N w_i I(y_i \neq G_m(x_i))}{\sum_{i=1}^N w_i}.$$

(c) Compute $\alpha_m = \log((1 - \text{err}_m)/\text{err}_m)$. 해당 분류기의 분류기 가중치 생성

(d) Set $w_i \leftarrow w_i \cdot \exp[\alpha_m \cdot I(y_i \neq G_m(x_i))]$, $i = 1, 2, \dots, N$.

Instance의 weight 업데이트

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

Unit 03 | AdaBoost

Adaboost

값 샘플의 weight를 $1/N$ 로 초기화

1. Initialize the observation weights $w_i = 1/N, i = 1, 2, \dots, N.$
 N 은 데이터의 갯수

모든 Instance의 초기값은 다 동일하게 시작을 함.

Ex) Instance가 100개 이면 모든 $W_i = 1/100$

Unit 03 | AdaBoost

Adaboost

2. For $m = 1$ to M : M은 모델의 갯수

Weight값을 기준으로 분류기 생성(resampling)

(a) Fit a classifier $G_m(x)$ to the training data using weights w_i .

Instance의 Weight값을 기준으로 Sampling을 진행

최초 샘플링 : 모두 뽑힐 확률이 동일함.

n round 샘플링 : 틀린 값에 가중치가 부여되므로 뽑힐 확률이 높아짐

!지금부터 for문이 돌아간다고 생각하면 됩니다!!!

Unit 03 | AdaBoost

Adaboost

(b) Compute 해당 분류기의 에러 계산

$$\text{err}_m = \frac{\sum_{i=1}^N w_i I(y_i \neq G_m(x_i))}{\sum_{i=1}^N w_i}, \quad I(y_i, G_m(x_i)) = \begin{cases} 0 & \text{if } y_i = G_m(x_i) \\ 1 & \text{if } y_i \neq G_m(x_i) \end{cases}$$

예측값과 실제값이 같으면 1이 되고, 다르면 0이 되게 함.

분자 : 틀린 예측 값의 instance의 weigh만 남게 되어 모두 더함

분모 : 모든 가중치를 더 함(Normalizer 역할)

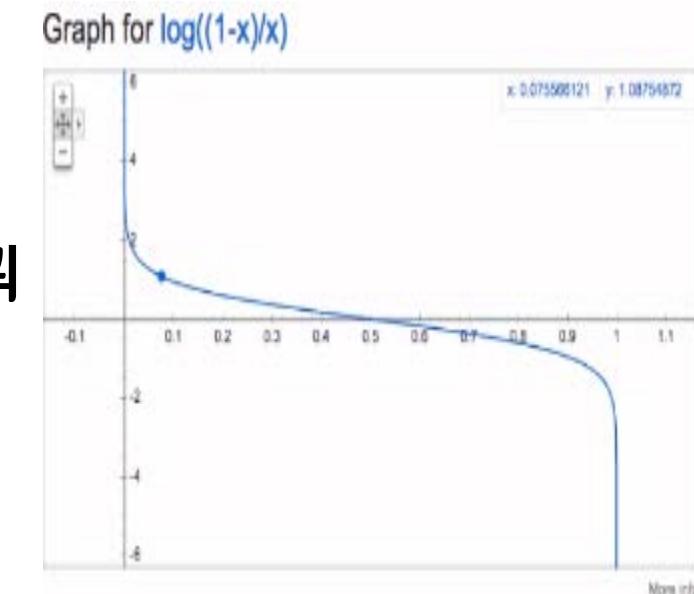
따라서 err의 범위는 $0 < \text{err} < 1$ 이다.

Unit 03 | AdaBoost

Adaboost

(c) Compute $\alpha_m = \log((1 - \text{err}_m)/\text{err}_m)$. 해당 분류기의 분류기 가중치 생성

- 에러가 높으면 m번째 모델은 좋지 않는 모델
- 에러가 낮으면 m번째 모델은 좋은 모델
- 따라서 반비례 관계이므로 $(1-\text{err})/\text{err}$ 를 통해 반대방향으로 바꿔 모델의 가중치를 계산한다.
- 전체 양상블에 얼마만큼의 역할을 반영할지 계산하는거임!!



Unit 03 | AdaBoost

Adaboost

$$I(y_i, G_m(x_i)) = \begin{cases} 0 & \text{if } y_i = G_m(x_i) \\ 1 & \text{if } y_i \neq G_m(x_i) \end{cases}$$

- (d) Set $w_i \leftarrow w_i \cdot \exp[\alpha_m \cdot I(y_i \neq G_m(x_i))]$, $i = 1, 2, \dots, N$.
- - - Instance의 weight 업데이트

모델의 가중치의 exp를 하여 틀린 놈들만 가중치를 업데이트 시켜버림.

이 짓을 for문이 Classifier 개수만큼 돌아가게 합니다.

Unit 03 | AdaBoost

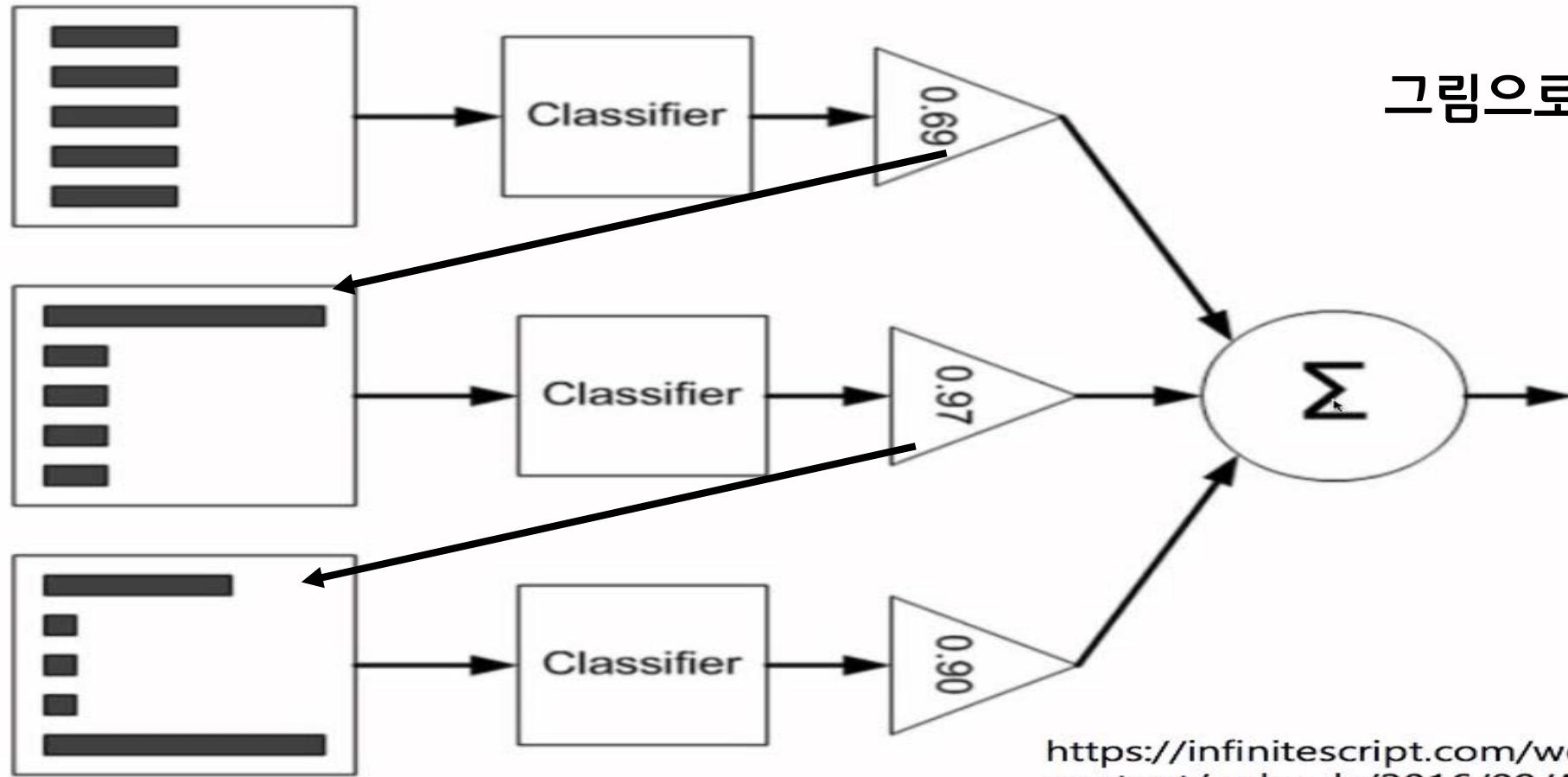
Adaboost

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

그러면 Classifier M개를 돌렸겠죠? 그리고 각각의 모델의 가중치가 있을 겁니다. 그 가중치를 곱하여 각각의 역할 비중을 줘서 Sumation을 통해 예측을 하는겁니다.

Unit 03 | AdaBoost

Adaboost

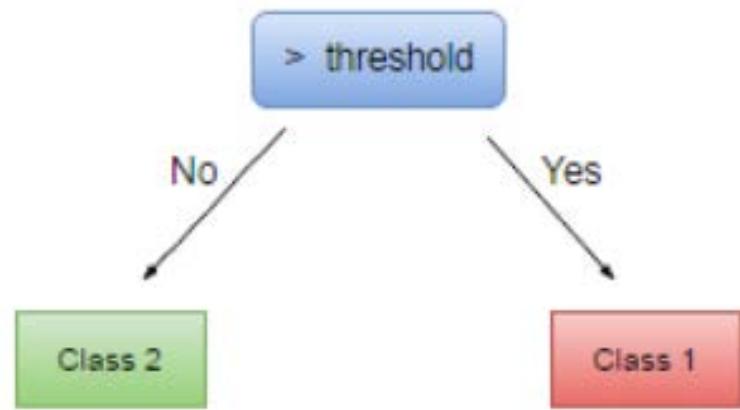


<https://infinitescript.com/wordpress/wp-content/uploads/2016/09/AdaBoost.jpg>

Unit 03 | AdaBoost

Adaboost with Stump

Adaboost의 사용되는 Classifier를 1Depth Tree를 사용하는 방법



Why? Boosting은 약한 모델(High Bias)을 점점 성장 시켜 좋은 분류를 하게 하는 것이기 때문이다. 물론, 다양한 분류 모델을 사용해도 된다! 하지만, DT의 경우 bias, Variance를 조정하기 쉬운 모델이기 때문에 대부분 Tree모델을 사용한다고 보면 된다.

Unit 03 | AdaBoost

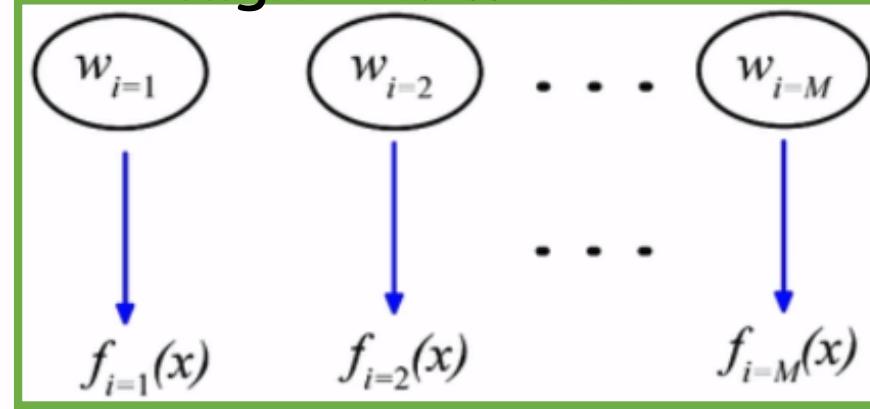
Adaboost

- 가장 인기있는 Boosting 알고리즘. 높은 성능을 보임.
- 계속 틀리는 Instance만 조져서 Model을 향상시킴
- 물론, Hyperparameter 조정을 통해 성능을 향상시켜야 한다.

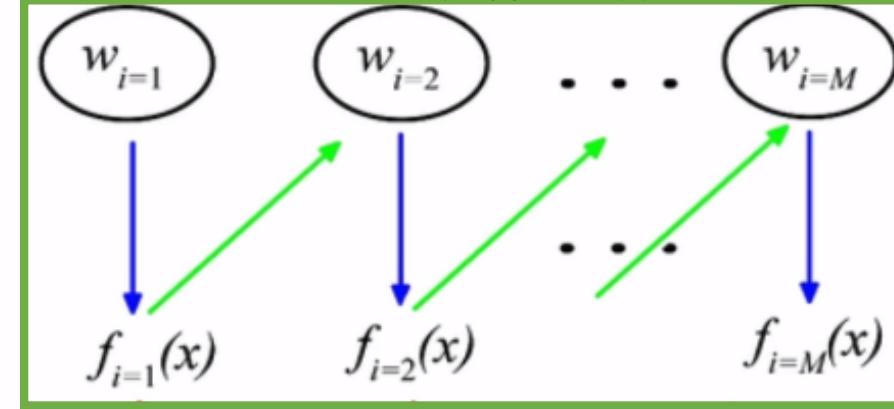
Unit 03 | Bagging VS Boosting

Bagging VS Boosting

High Variance Tree



Low Variance Tree



$$F_M(x) = \text{sign}(\sum_{i=1}^M f_i(x))$$

b. Bagging

$$F_M(x) = \text{sign}(\sum_{i=1}^M f_i(x))$$

a. Boosting

Unit 03 | Bagging VS Boosting

Bagging VS Boosting

특성	Bagging	Boosting
병렬화	O	X
Variance	High Variance Tree	Low Variance Tree (High Bias)
Sampling	All Same	Not Same ->Bagging보다 좋은 성능

Unit 03 | Adaboost

[실습4-Adaboost] 켜주세요!

Unit 03 | Gradient Boosting

※주의 지금부터 수식적인 내용이 많아서 개념 위주로 가겠습니다.

Unit 03 | Gradient Boosting

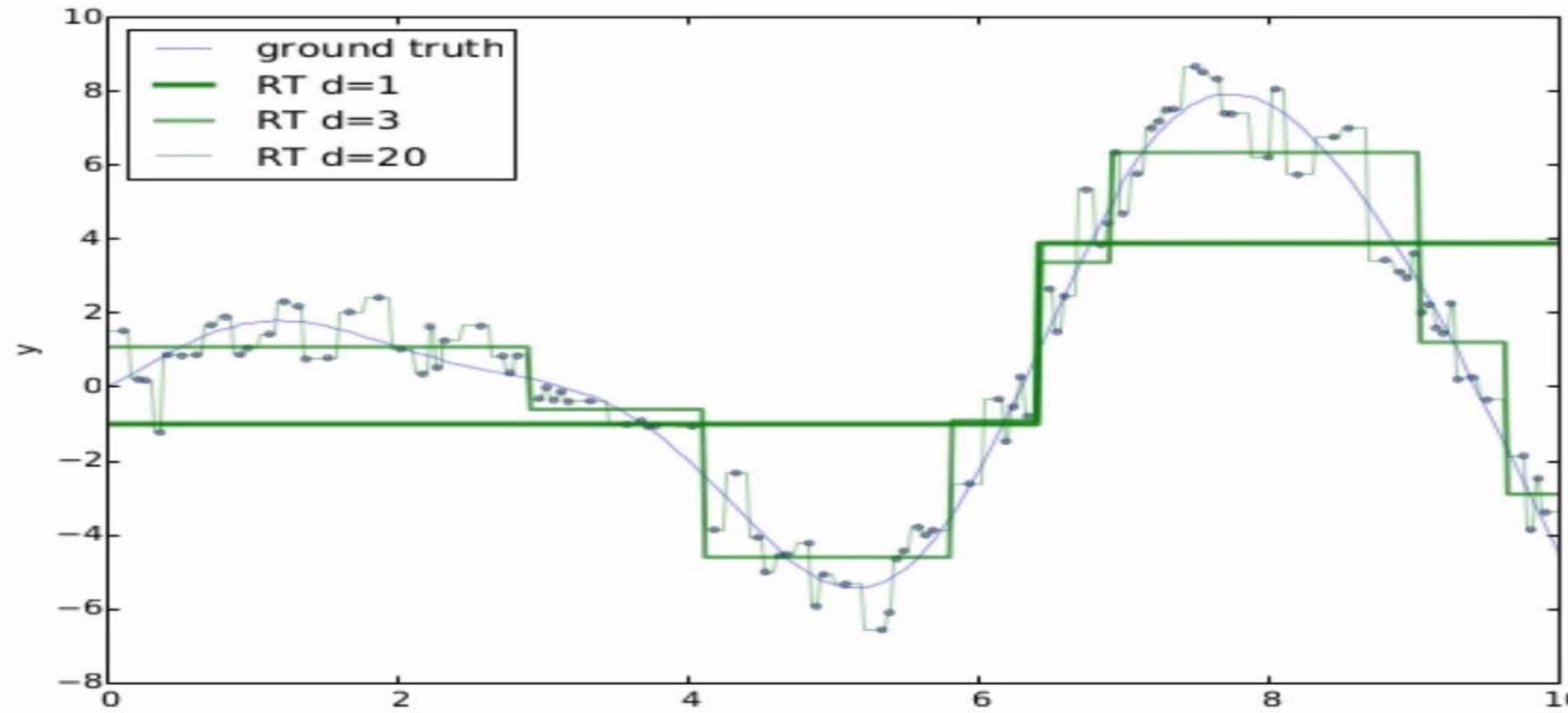
What is Gradient Boosting

- Adaboost와 같은 Boosting 기법의 일종
- Regression, Classification 등 모두 사용
- Sequential + Additive Model
- 이전 모델의 Residual을 가지고 Model를 강화시킴
- Residual을 예측하여 발전하는 Weak learner

Unit 03 | Gradient Boosting

Boosting with regression

Function approximation with Regression Trees



Unit 03 | Gradient Boosting

Boosting Example

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

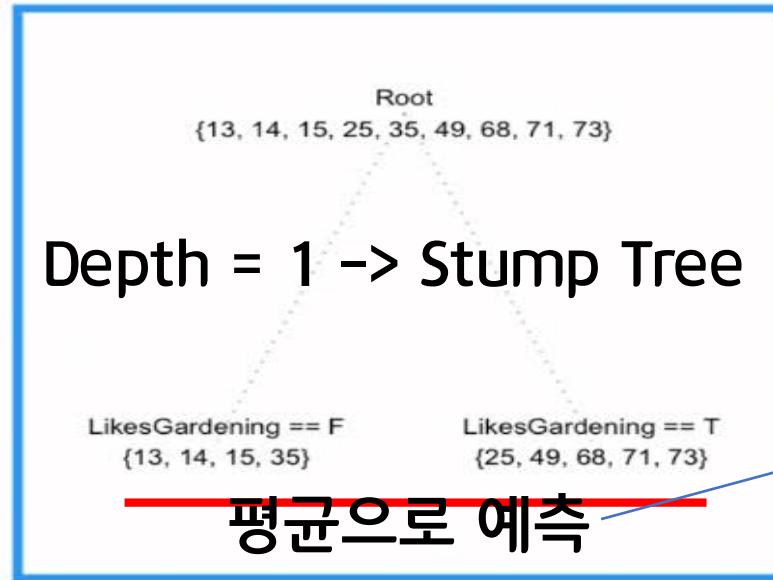


<http://blog.kaggle.com/2017/01/23/a-kaggle-master-explains-gradient-boosting/>

Unit 03 | Gradient Boosting

Boosting Example

Tree 1



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

추정값

Tree1 Prediction
19.25
19.25
19.25
57.2
19.25
57.2
57.2
57.2
57.2

잔차 = 참값 - 추정값

Tree1 Residual
-6.25
-5.25
-4.25
-32.2
15.75
-8.2
10.8
13.8
15.8

Unit 03 | Gradient Boosting

Boosting Example



Unit 03 | Gradient Boosting

Boosting Example

$$\hat{y}_i^{(0)} = 0$$

$$\hat{y}_i^{(1)} = f_1(x_i) = \hat{y}_i^{(0)} + f_1(x_i)$$

$$\hat{y}_i^{(2)} = f_1(x_i) + f_2(x_i) = \hat{y}_i^{(1)} + f_2(x_i)$$

...

$$\hat{y}_i^{(t)} = \sum_{k=1}^t f_k(x_i) = \hat{y}_i^{(t-1)} + f_t(x)$$

$$M\left(x, y - \sum_{k=1}^t f_k(x)\right) = f_t(x)$$

Unit 03 | Gradient Boosting

Boosting Example

$$\hat{y}_i^{(0)} = 0$$

$$\hat{y}_i^{(1)} = f_1(x_i) = \hat{y}_i^{(0)} + f_1(x_i)$$



Tree1 Prediction
19.25
19.25
19.25
57.2
19.25
57.2
57.2
57.2
57.2

Unit 03 | Gradient Boosting

Boosting Example

Age	Tree1 Prediction	Tree1 Residual
13	19.25	-6.25
14	19.25	-5.25
15	19.25	-4.25
25	57.2	-32.2
35	19.25	15.75
49	57.2	-8.2
68	57.2	10.8
71	57.2	13.8
73	57.2	15.8

Unit 03 | Gradient Boosting

Boosting Example

 $f_2(x_i)$ →

Tree1 Residual	LikesGardening	PlaysVideoGames	LikesHats
-6.25	FALSE	TRUE	TRUE
-5.25	FALSE	TRUE	FALSE
-4.25	FALSE	TRUE	FALSE
-32.2	TRUE	TRUE	TRUE
15.75	FALSE	TRUE	TRUE
-8.2	TRUE	FALSE	FALSE
10.8	TRUE	TRUE	TRUE
13.8	TRUE	FALSE	FALSE
15.8	TRUE	FALSE	TRUE

잔차를 예측한다.

Unit 03 | Gradient Boosting

Boosting Example

$$\hat{y}_i^{(2)} = f_1(x_i) + f_2(x_i) = \boxed{\hat{y}_i^{(1)}} + \boxed{f_2(x_i)}$$



Tree1 Prediction
19.25
19.25
19.25
57.2
19.25
57.2
57.2
57.2
57.2

+

Tree2 Prediction
-3.567
-3.567
-3.567
-3.567
-3.567
-3.567
7.133
-3.567
7.133
7.133

Unit 03 | Gradient Boosting

Boosting Code

```
from sklearn.tree import DecisionTreeRegressor

tree_reg1 = DecisionTreeRegressor(max_depth=2)
tree_reg1.fit(X, y)

r1 = y - tree_reg1.predict(X)
tree_reg2 = DecisionTreeRegressor(max_depth=2)
tree_reg2.fit(X, r1)

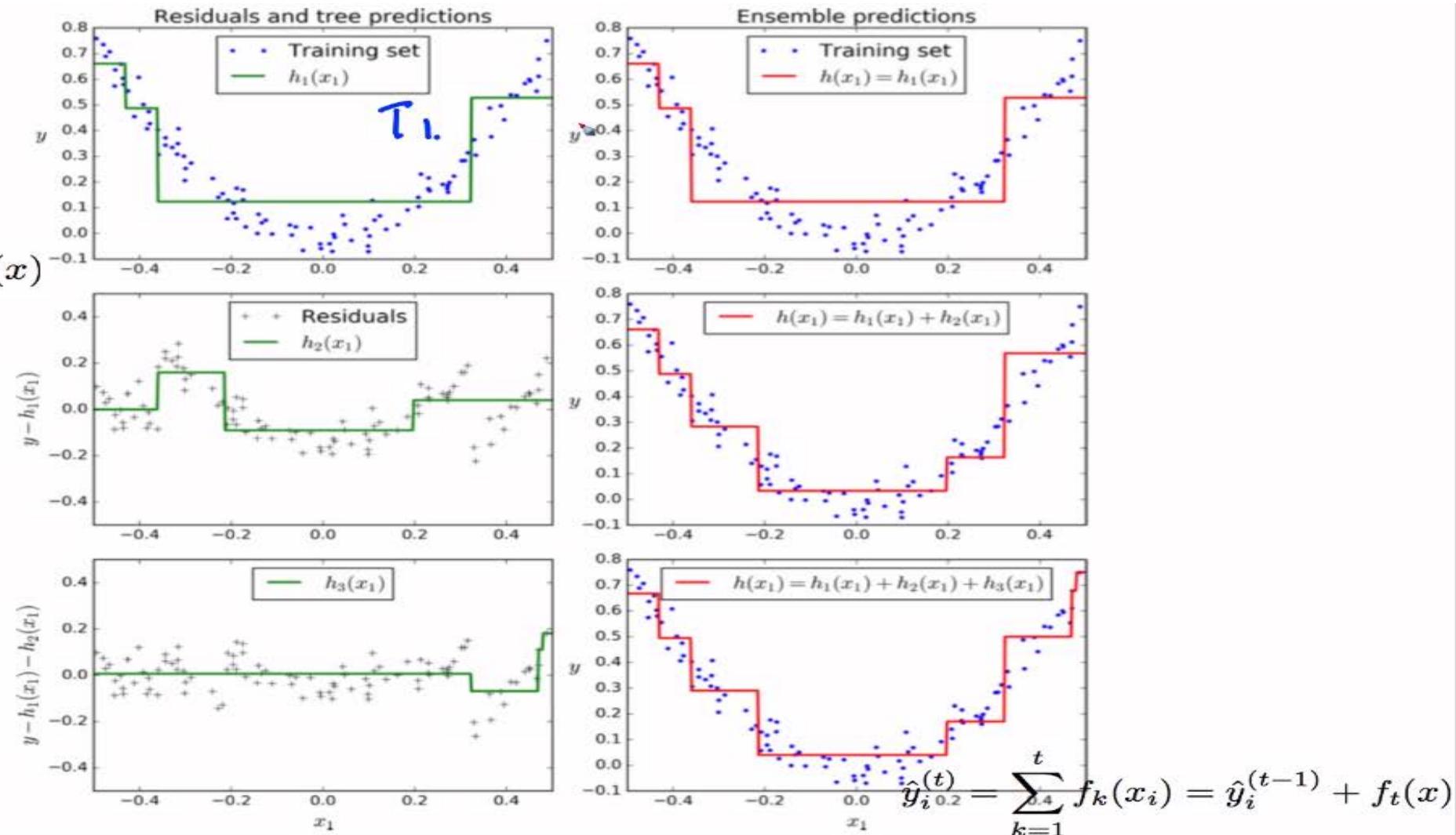
r2 = r1 - tree_reg2.predict(X)
tree_reg3 = DecisionTreeRegressor(max_depth=2)
tree_reg3.fit(X, r2)

y_pred = sum(tree.predict(X_new) for tree in (tree_reg1, tree_reg2, tree_reg3))
```

Unit 03 | Gradient Boosting

Boosting GRAPH

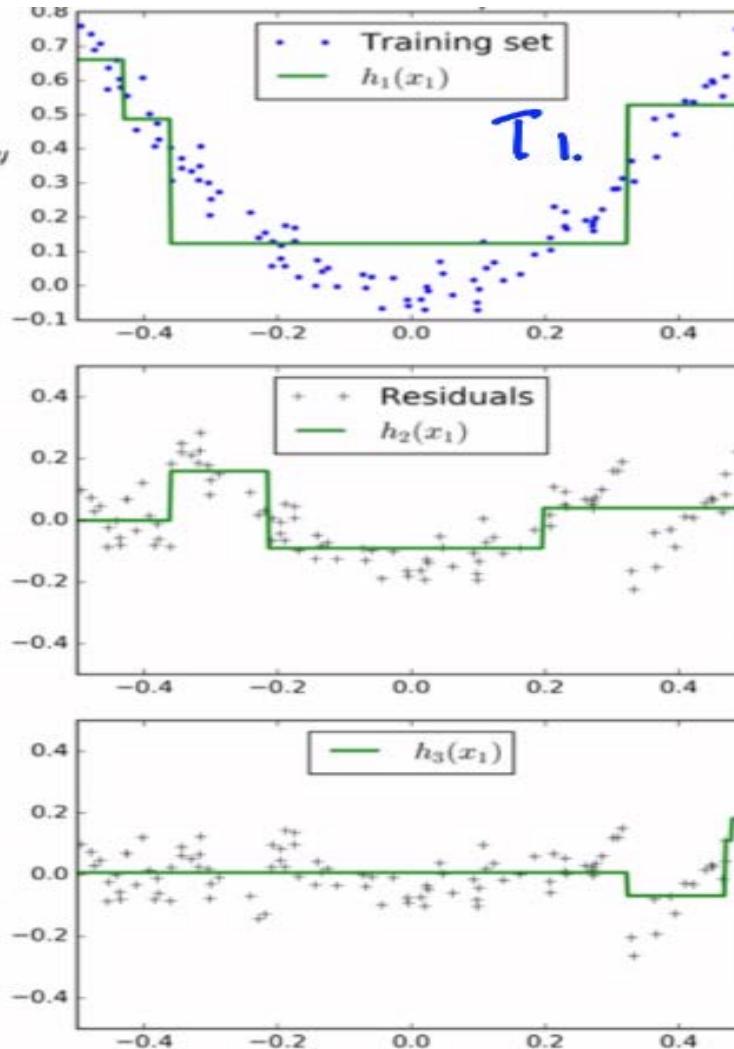
$$M(x, y - \sum_{k=1}^t f_{k-1}(x)) = f_t(x)$$



Unit 03 | Gradient Boosting

근데 왜 Gradient라고 할까????

Unit 03 | Gradient Boosting



Gradient Boosting은 잔차를 없애는
방향으로 학습을 하게 된다.
→ 이때 학습 방법을 Gradient
Method를 쓰기 때문이다.

Unit 03 | Gradient Boosting

Tuning parameters

- Number of Tree : 얼마나 많은 Sequential한 Tree를 만들 것인가?
- Depth of Tree : 트리의 복잡성 -> 2,3,4 정도로 짧을 수록 좋다.
- Subsampling : Random Forest의 Sampling과 같음
- Shrinkage Parameter λ : 보통 람다는 작고 estimator는 큰 것이 좋음
- Fitting to low variance

Unit 03 | Gradient Boosting

Sklearn with Gradient Boosting

3.2.4.3.6. `sklearn.ensemble.GradientBoostingRegressor`

```
class sklearn.ensemble.GradientBoostingRegressor (loss='ls', learning_rate=0.1, n_estimators=100,  
subsample=1.0, criterion='friedman_mse', min_samples_split=2, min_samples_leaf=1, min_weight_fraction_leaf=0.0,  
max_depth=3, min_impurity_decrease=0.0, min_impurity_split=None, init=None, random_state=None,  
max_features=None, alpha=0.9, verbose=0, max_leaf_nodes=None, warm_start=False, presort='auto') [source]
```

loss : {'ls', 'lad', 'huber', 'quantile'}, optional (default='ls')

learning_rate : float, optional (default=0.1)

n_estimators : int (default=100)

max_depth : integer, optional (default=3)

subsample : float, optional (default=1.0)

Unit 03 | Gradient Boosting

[실습6-Gradient Boosting] 켜주세요!

Unit 01 | Ensemble Model Overview & Voting Classifier

Unit 02 | Bagging – Random Forest

Unit 03 | Boosting – AdaBoost, Gradient Boosting

Unit 04 | XGBoosting, GBM & LightGBM

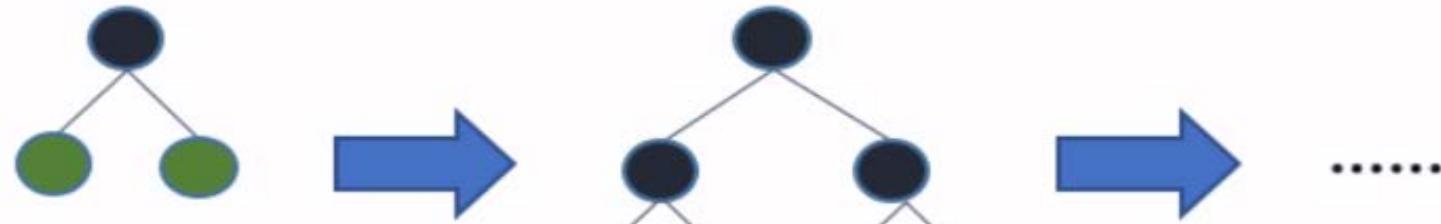
Unit 05 | Stacking

Unit 04 | XGBoost and LightGBM

Various boosting method

- GBM 연상량과 병렬처리를 위한 패키지가 존재
- 대표적인 패키지는 XGBoost와 LightGBM
- XGBoost – eXtreme Gradient Boosting
- LightGBM – Light Gradient Boosting Machine
- 구현 알고리즘은 일부 상이하나 목표는 비슷함
- 둘다 Tree기반은 Gradient Boosting임!!

Unit 04 | XGBoost and LightGBM

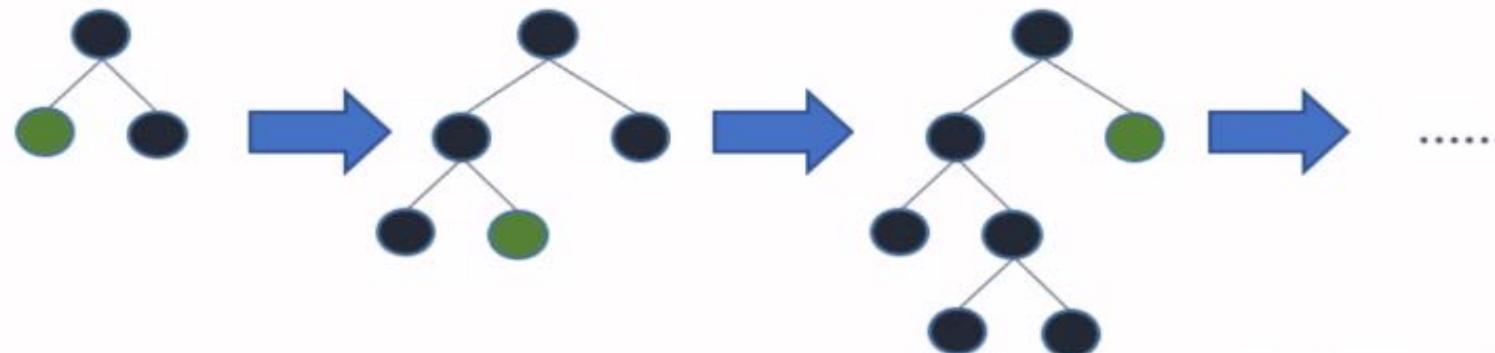


Level-wise tree growth

GBRT

XGBOOST

XGBOOST – 균형적으로 성장



Leaf-wise tree growth

Light GBM

Light GBM – 한쪽으로만 성장

Unit 04 | XGBoost and LightGBM

		XGBoost	Light GBM
Parameters Used		max_depth: 50 learning_rate: 0.16 min_child_weight: 1 n_estimators: 200	max_depth: 50 learning_rate: 0.1 num_leaves: 900 n_estimators: 300
Training AUC Score	0.999	Without passing indices of categorical features 0.992	Passing indices of categorical features 0.999
Test AUC Score	0.789	0.785	0.772
Training Time	970 secs	153 secs	326 secs
Prediction Time	184 secs	40 secs	156 secs
Parameter Tuning Time (for 81 fits, 200 iteration)	500 minutes	200 minutes	

Unit 01 | Ensemble Model Overview & Voting Classifier

Unit 02 | Bagging – Random Forest

Unit 03 | Boosting – AdaBoost, Gradient Boosting

Unit 04 | XGBoosting, GBM & LightGBM

Unit 05 | Stacking

Unit 05 | Stacking

What is Stacking

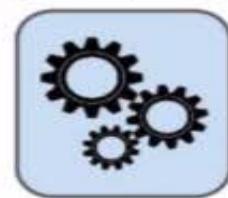
- Meta Ensemble -> 여러 모델의 결과를 묶어서 예측
- 컴퓨터 성능향상과 함께 최근 대중화 되고 있다.
- 키워드 : Stacking Kaggle Stacknet
- 복잡성으로 인해 완벽한 파이썬 구현체는 아직 없음
- 구현 알고리즘은 일부 상이하나 목표는 비슷함



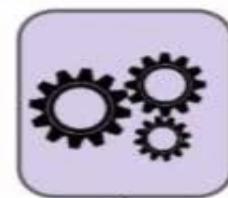
Unit 05 | Stacking

Basic Stacking

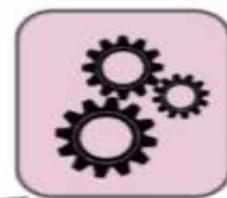
여러개의 모델을 생성



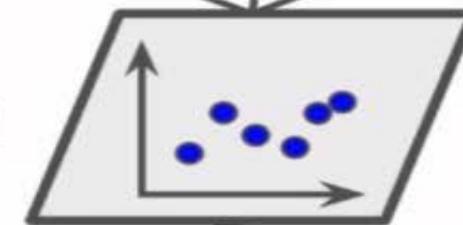
Train



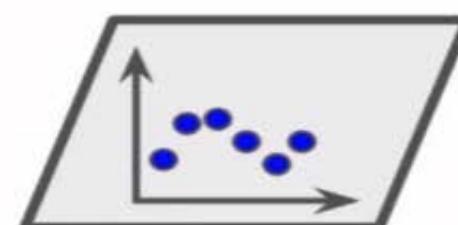
Subset 1로만 학습을 함



Subset 1

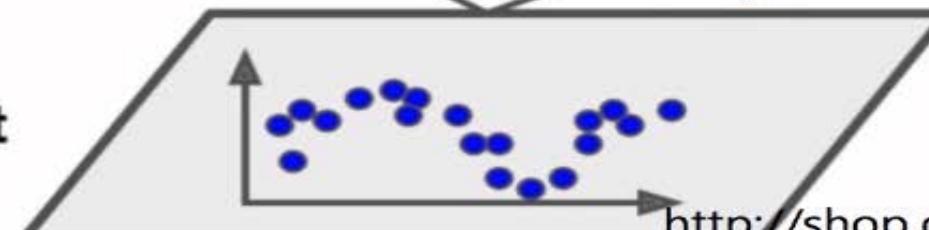


Split



Subset 2
데이터셋을 나눔

Training set

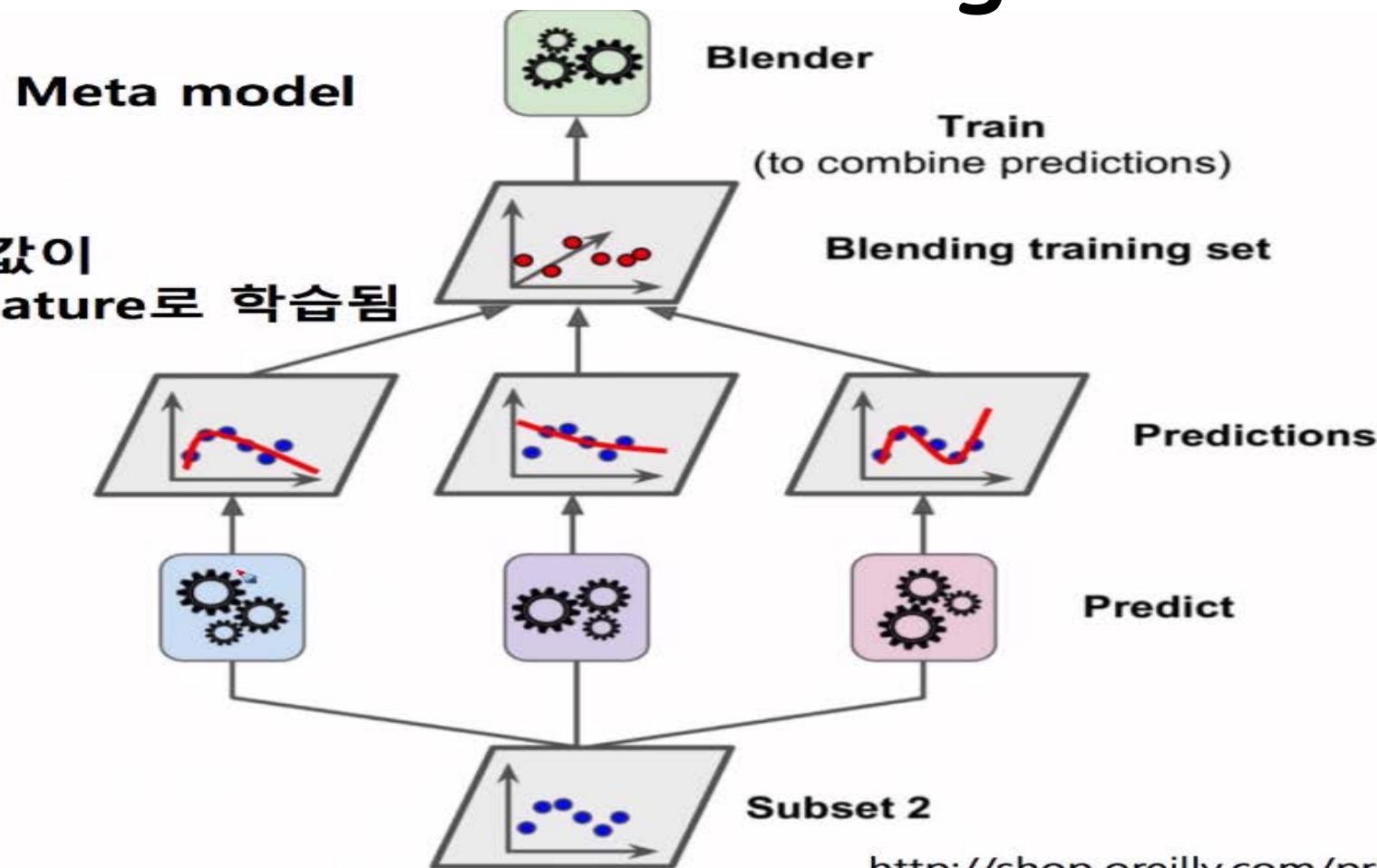


<http://shop.oreilly.com/product/0636920052289.do>

Unit 05 | Stacking

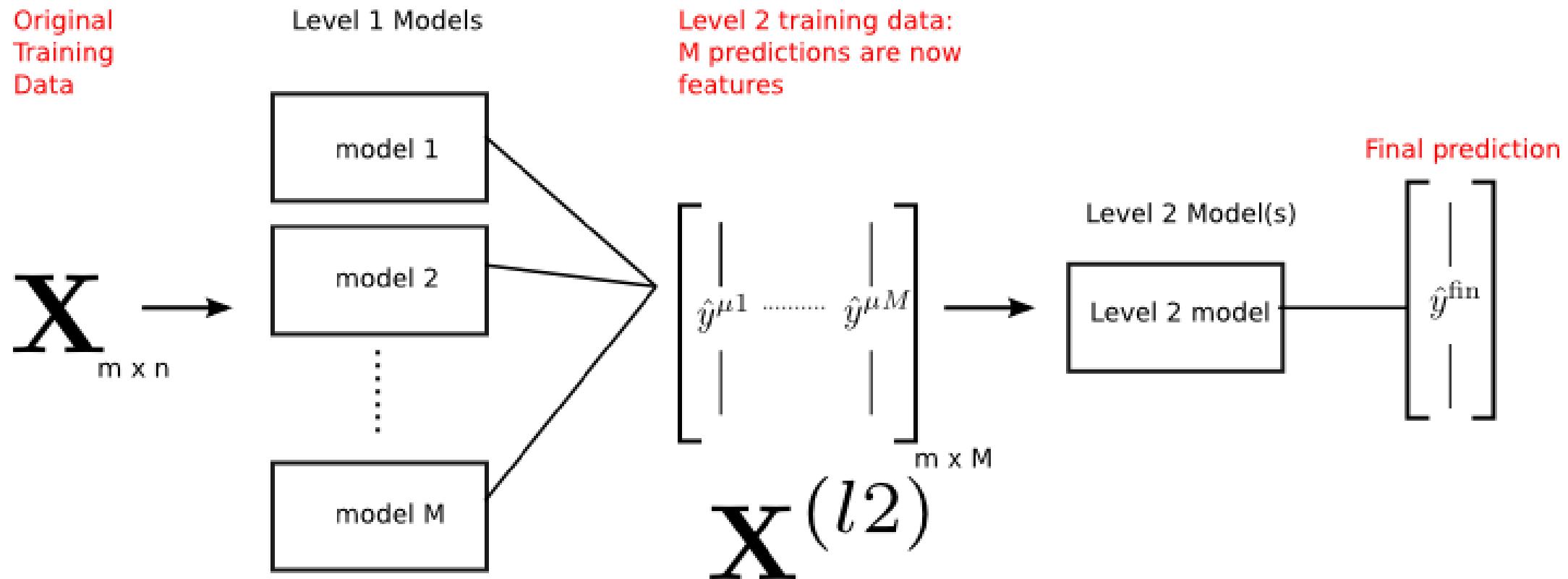
Basic Stacking

각 모델의 결과값이
Meta 모델의 feature로 학습됨



Unit 05 | Stacking

Basic Stacking



Unit 05 | Stacking

[실습7-Stacking] 켜주세요!

Q & A

들어주셔서 감사합니다.

이제 과제 설명드리겠습니다.

Assginment

원래 갖고 있던 무기



Assginment

오늘 갖게된 무기



Dual MP7



SG-870_v3



AK47 Aim



M4A1 Aim



MP5 Aim



Dragunov



KSG-12



KRISS



Tiger Stripe

Assignment

과제 : House Prices: Advanced Regression Techniques

Assignment : 지금까지 배운 아래 모델 모두를 활용하여 MSE를 구해보고 Test도 예측해서 케글에 제출 해보세요! 가장 성능이 좋은 모델로 리더보드를 찍어보고 캡처해서 올려주세요.

지금까지 배운 모델

1주차)Linear Regression

2주차)SVM, KNN, Naive Bayesian

3주차) DT(단일 모델), Random Forest, AdaBoost, XGBoost, LightGBM, **STACKING**

STACKING방법은 다양할 수 있으나, 생각이 안나시면 오늘 배운 정석적인 Stacking을 하시면 됩니다.

데이터 설명 : <https://www.kaggle.com/c/house-prices-advanced-regression-techniques#description>

Assignment

과제 : House Prices: Advanced Regression Techniques

변수 설명 및 데이터 다운 : 79개 변수

<https://www.kaggle.com/c/house-prices-advanced-regression-techniques/data>

데이터는 올려드리지 않습니다. Kaggle이랑 친해지기 위해 직접 받으세요! 꼭 들어가서 설명 되어있는 부분 많이 읽어 보시고 Kernels에 가면 전처리도 참고할만한 코드도 완전 널려있습니다.

Pipeline : 전처리 -> Feature Selection -> Modeling -> Tuning 순으로 진행하여 보세요

1. 전처리
2. Feature Selection
3. Modeling
4. Tuning

이라고 꼭 주석을 달아주셔야 합니다.

본과제는 따로 설과제를 주어지지 않는 대신 진행됩니다. 따라서 2월 12일(화)까지 진행하여 주세요. 11기 분 중 가장 높은 리더 보드를 찍으시는 분은 스타벅스 기프티콘을 드리겠습니다.