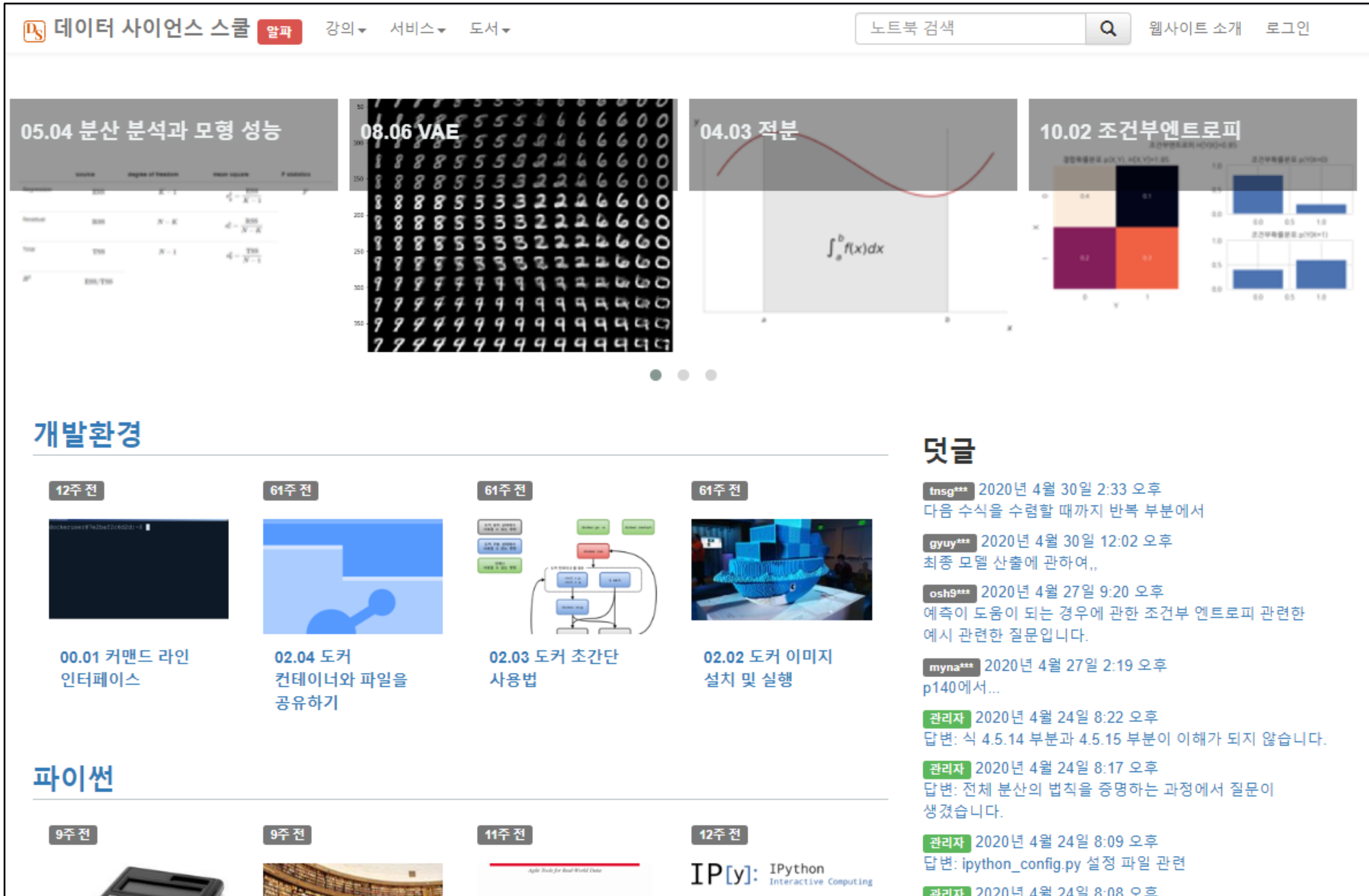
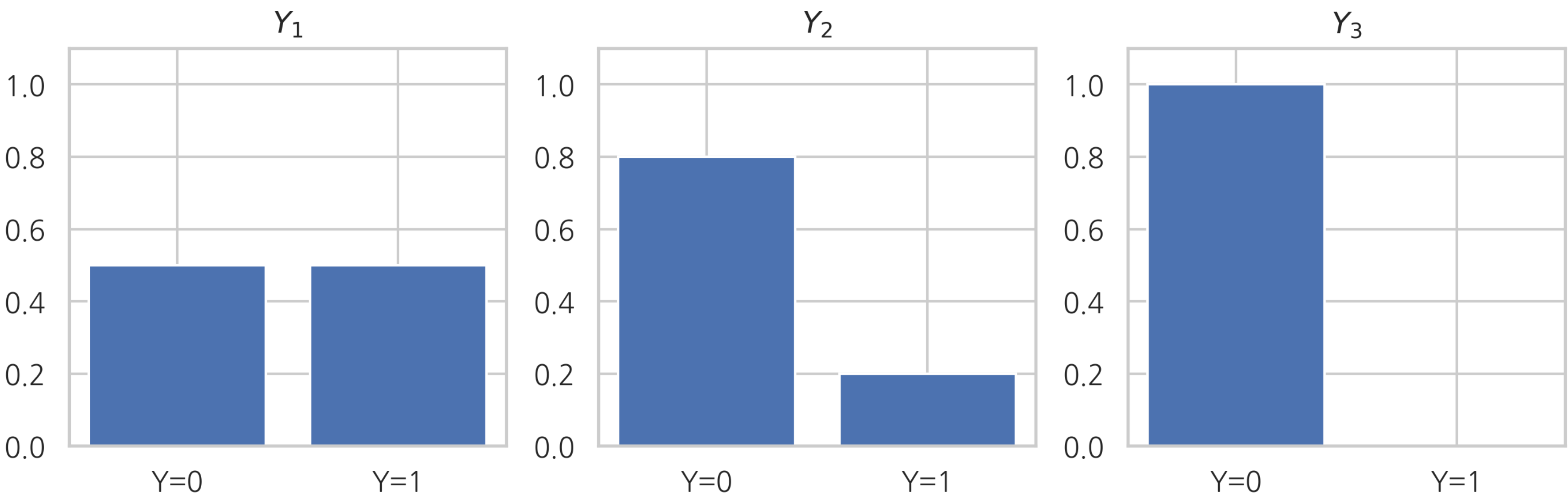
Tree-based algorithms

Data Science School (by Dr. Dohyeong Kim, datascienceschool.net)



Entropy



- $P(Y_1 = 0) = 0.5, P(Y_1 = 1) = 0.5$
- $P(Y_2 = 0) = 0.8, P(Y_2 = 1) = 0.2$
- $P(Y_3 = 0) = 1.0, P(Y_3 = 1) = 0.0$

Entropy

$$H[Y] = E_Y[-\log_2 P(y)]$$

- Y: discrete

$$H[Y] = -\sum_{k=1}^{K} p(y_k)\log_2 p(y_k)$$

- Y: continuous

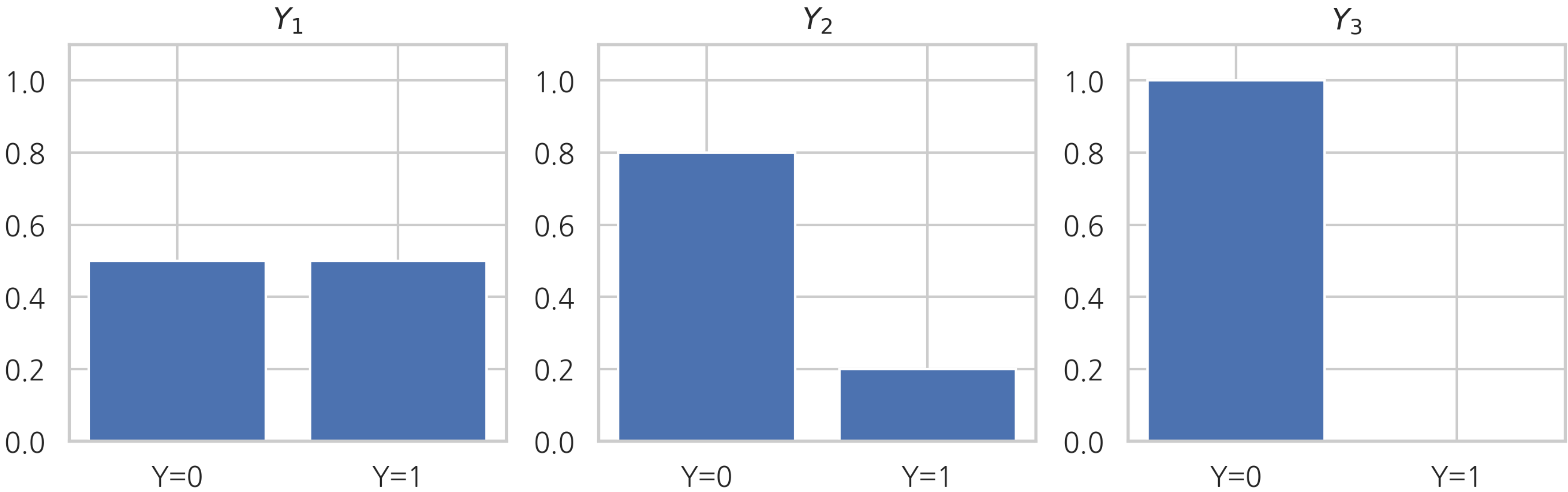
$$H[Y] = -\int_D p(y)\log_2 p(y)dy$$

$$\lim_{p\to 0+} p\log_2 p = \lim_{p\to 0+} \frac{\log_2 p}{\frac{1}{p}} = \lim_{p\to 0+} \frac{\frac{1}{p\ln 2}}{-\frac{1}{p^2}} = 0$$

Entropy

$$H[Y_i] = -p_0^{(i)} \log_2 p_0^{(i)} - p_1^{(i)} \log_2 p_1^{(i)}$$

where $p_0^{(i)} = P(Y_i = 0)$, $p_1^{(i)} = P(Y_i = 1)$



- $H[Y_1] = -0.5 \log_2 0.5 - 0.5 \log_2 0.5 = 1$
- $H[Y_2] = -0.8 \log_2 0.8 - 0.2 \log_2 0.2 = 0.72$
- $H[Y_3] = -1 \log_2 1 - 0 \log_2 0 = 0$

Exercise 1

Let Y be a Bernoulli random variable, that is, $Y \sim \mathrm{Ber}(p)$.
Draw the graph of entropy $H[Y]$ with respect to p.

Exercise 2

Calculate the entropy $H[Y]$ of the random variable Y following the distribution below.

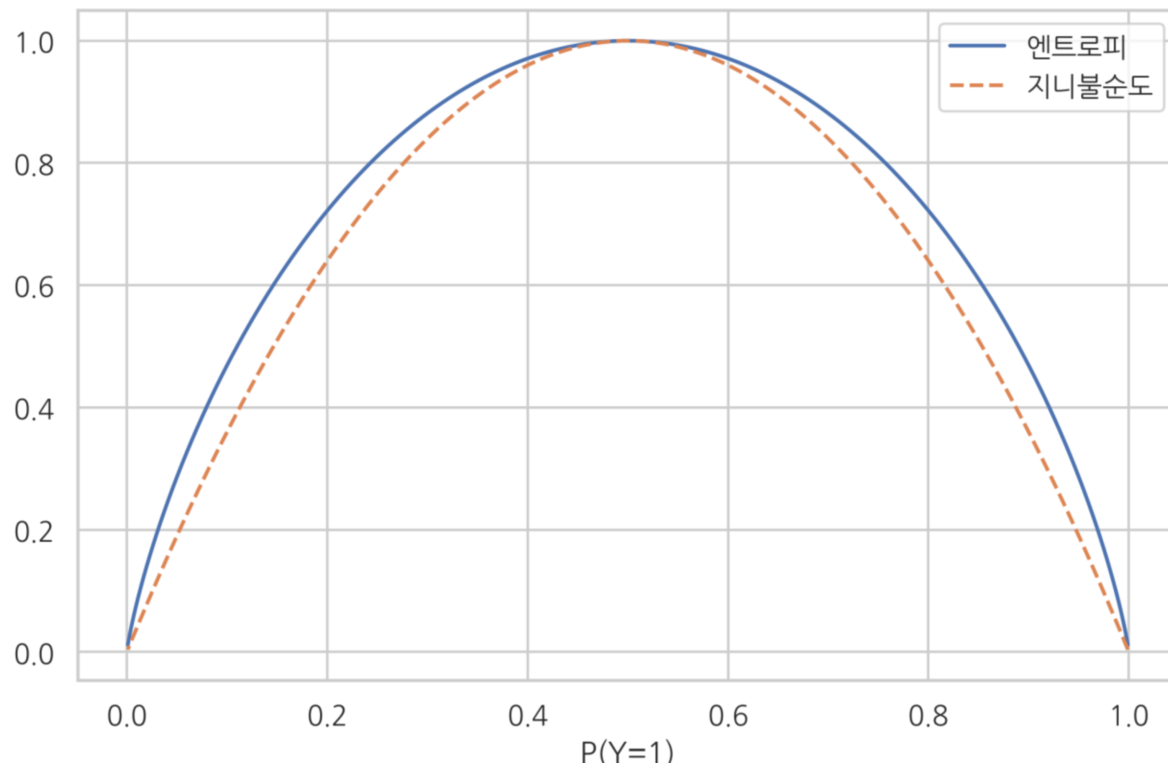
(a) $P(Y=0)=\frac{1}{8}$, $P(Y=1)=\frac{1}{8}$, $P(Y=2)=\frac{1}{4}$, $P(Y=3)=\frac{1}{2}$

(b) $P(Y=0)=1$, $P(Y=1)=0$, $P(Y=2)=0$, $P(Y=3)=0$

(c) $P(Y=0)=\frac{1}{4}$, $P(Y=1)=\frac{1}{4}$, $P(Y=2)=\frac{1}{4}$, $P(Y=3)=\frac{1}{4}$

Gini impurity

An alternative to entropy, which requires much less computation



Gini impurity

$$G[Y] = E_Y[1 - P(y)]$$

Entropy

$$H[Y] = E_Y[-\log_2 P(y)]$$

Joint entropy

$$H[X,Y] = E_{(X,Y)}[-\log_2 P(x,y)]$$

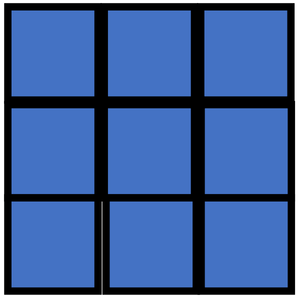
- X, Y: discrete

$$H[X,Y] = -\sum_{i=1}^{K_x} \sum_{j=1}^{K_y} p(x_i, y_j) \log_2 p(x_i, y_j)$$

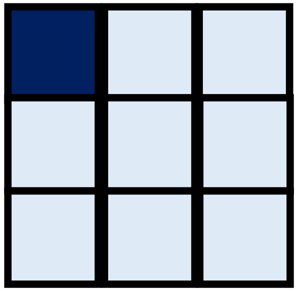
- X, Y: continuous

$$H[X,Y] = -\iint_{D_x \times D_y} p(x,y) \log_2 p(x,y) dxdy$$

high entropy



low entropy



Conditional entropy

$$H[Y|X=x] = E_{Y|x}[-\log_2 P(y|x)]$$

- X, Y: discrete

$$H[Y|x] = -\sum_{j=1}^{K_y} p(y_j|x) \log_2 p(y_j|x)$$

- X, Y: continuous

$$H[Y|x] = -\int_{D_y} p(y|x) \log_2 p(y|x)\, dxdy$$

Conditional entropy

$$H[Y|X] = E_X\big[H[Y|X=x]\big]$$

- X, Y: discrete

$$H[Y|X] = -\sum_{i=1}^{K_x} p(x_i)H(Y|x=x_i) = \sum_{i=1}^{K_x}\sum_{j=1}^{K_y} p(x_i)p\big(y_j|x\big)\log_2 p\big(y_j|x\big)$$

- X, Y: continuous

$$H[Y|X] = -\int_{D_x} p(x)H(Y|x)dx = \iint_{D_x \times D_y} p(x)p(y|x)\log_2 p(y|x)\,dxdy$$

Conditional entropy

	Y=0	Y=1
X=0	0.4	0.0
X=1	0.0	0.6

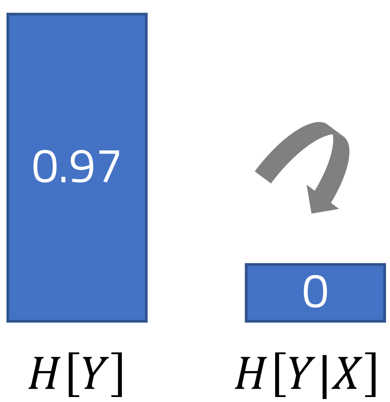
$$H[Y] = -p(Y=0)\log_2 p(Y=0) \\ -p(Y=1)\log_2 p(Y=1) \approx 0.97$$

Y=0	Y=1	| X=0
1.0	0.0	

$$H[Y|X=0] = -p(Y=0|X=0)\log_2 p(Y=0|X=0) \\ -p(Y=1|X=0)\log_2 p(Y=1|X=0) = 0$$

Y=0	Y=1	| X=1
0.0	1.0	

$$H[Y|X=1] = -p(Y=0|X=1)\log_2 p(Y=0|X=1) \\ -p(Y=1|X=1)\log_2 p(Y=1|X=1) = 0$$



$$H[Y|X] = p(X=0)H(Y|X=0) + p(X=1)H(Y|X=1) = 0$$

Conditional entropy

	Y=0	Y=1
X=0	1/9	2/9
X=1	2/9	4/9

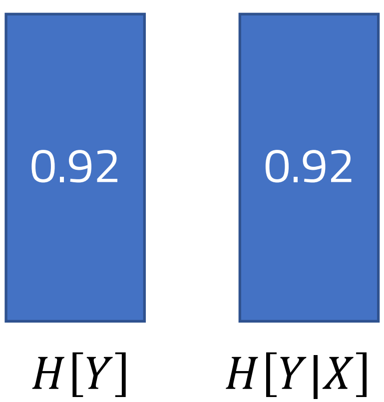
$$H[Y] = -p(Y=0)\log_2 p(Y=0)$$
$$-p(Y=1)\log_2 p(Y=1) \approx 0.92$$

Y=0	Y=1	| X=0
1/3	2/3	

$$H[Y|X=0] = -p(Y=0|X=0)\log_2 p(Y=0|X=0)$$
$$-p(Y=1|X=0)\log_2 p(Y=1|X=0) \approx 0.92$$

Y=0	Y=1	| X=1
1/3	2/3	

$$H[Y|X=1] = -p(Y=0|X=1)\log_2 p(Y=0|X=1)$$
$$-p(Y=1|X=1)\log_2 p(Y=1|X=1) \approx 0.92$$



$$H[Y|X] = p(X=0)H(Y|X=0) + p(X=1)H(Y|X=1) \approx 0.92$$

Information Gain

$Y = \begin{matrix} y_1 & \text{(fit)} \\ y_2 & \text{(unfit)} \end{matrix}$

X_1 : age

X_2 : pizza (y/n)

X_3 : exercise (y/n)

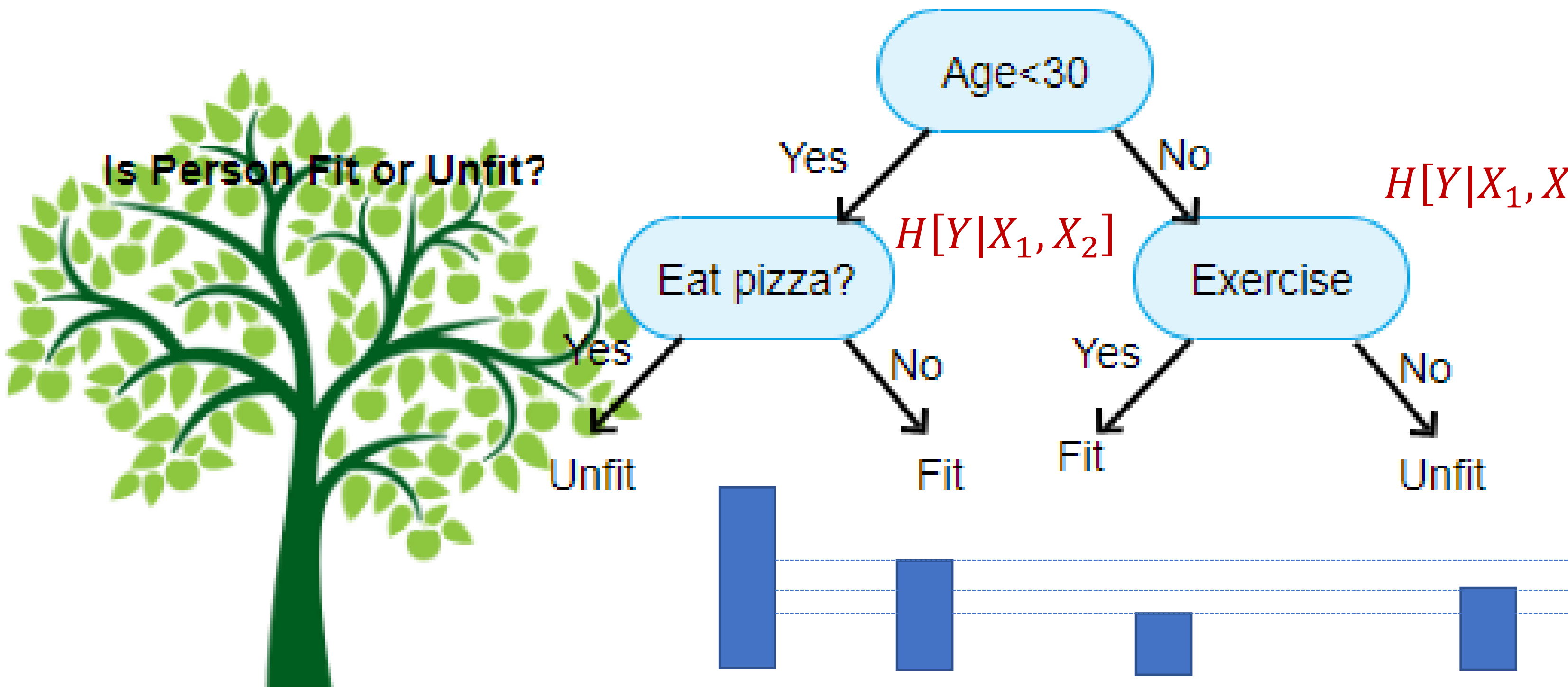
$$IG[Y;X] = H(Y) - H(Y|X)$$

$H[Y]$

$H[Y|X_1]$

$H[Y|X_1, X_2]$

$H[Y|X_1, X_3]$



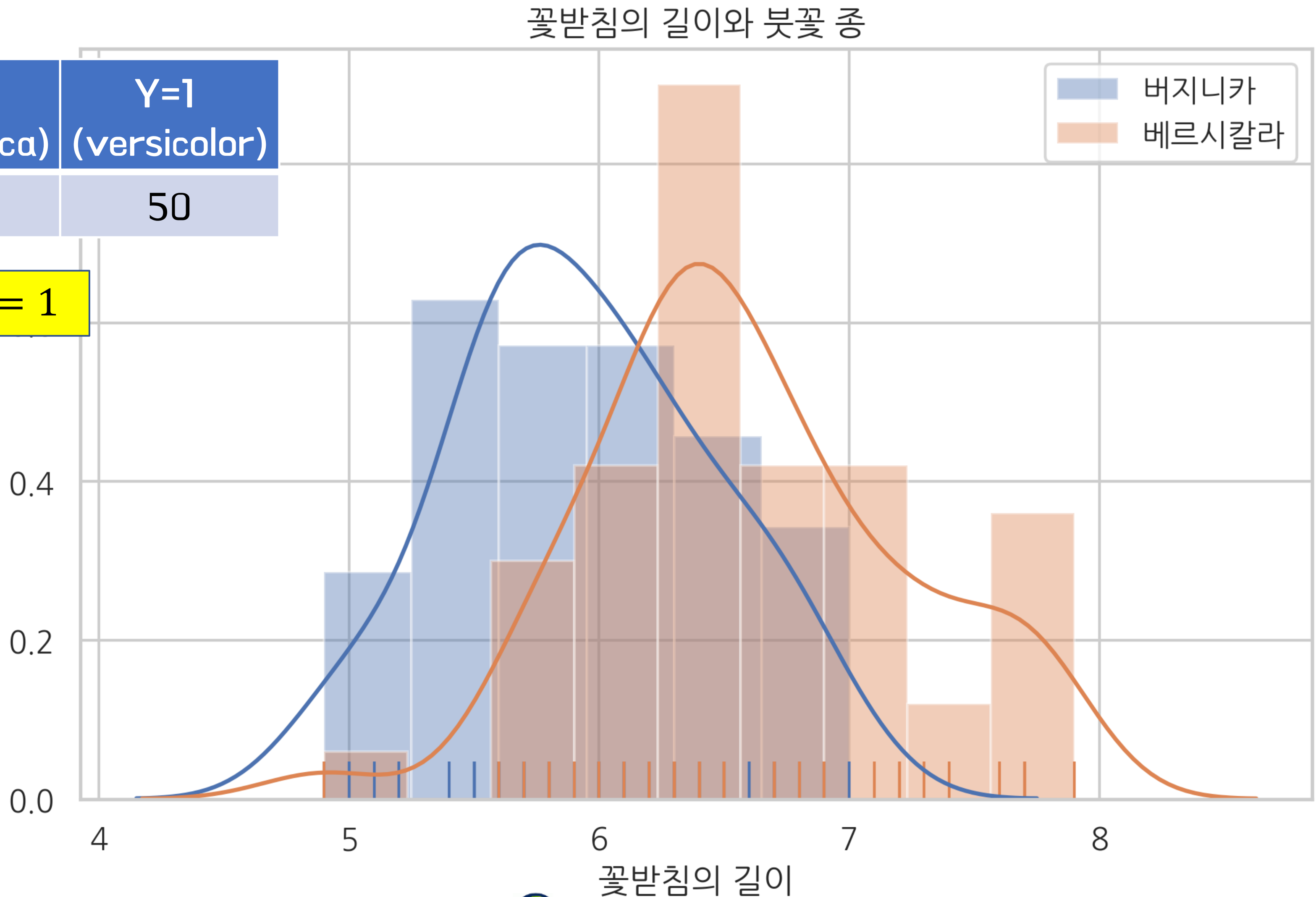
Decision Tree

$H[Y]$ $H[Y|X_1]$ $H[Y|X_1, X_2]$ $H[Y|X_1, X_3]$

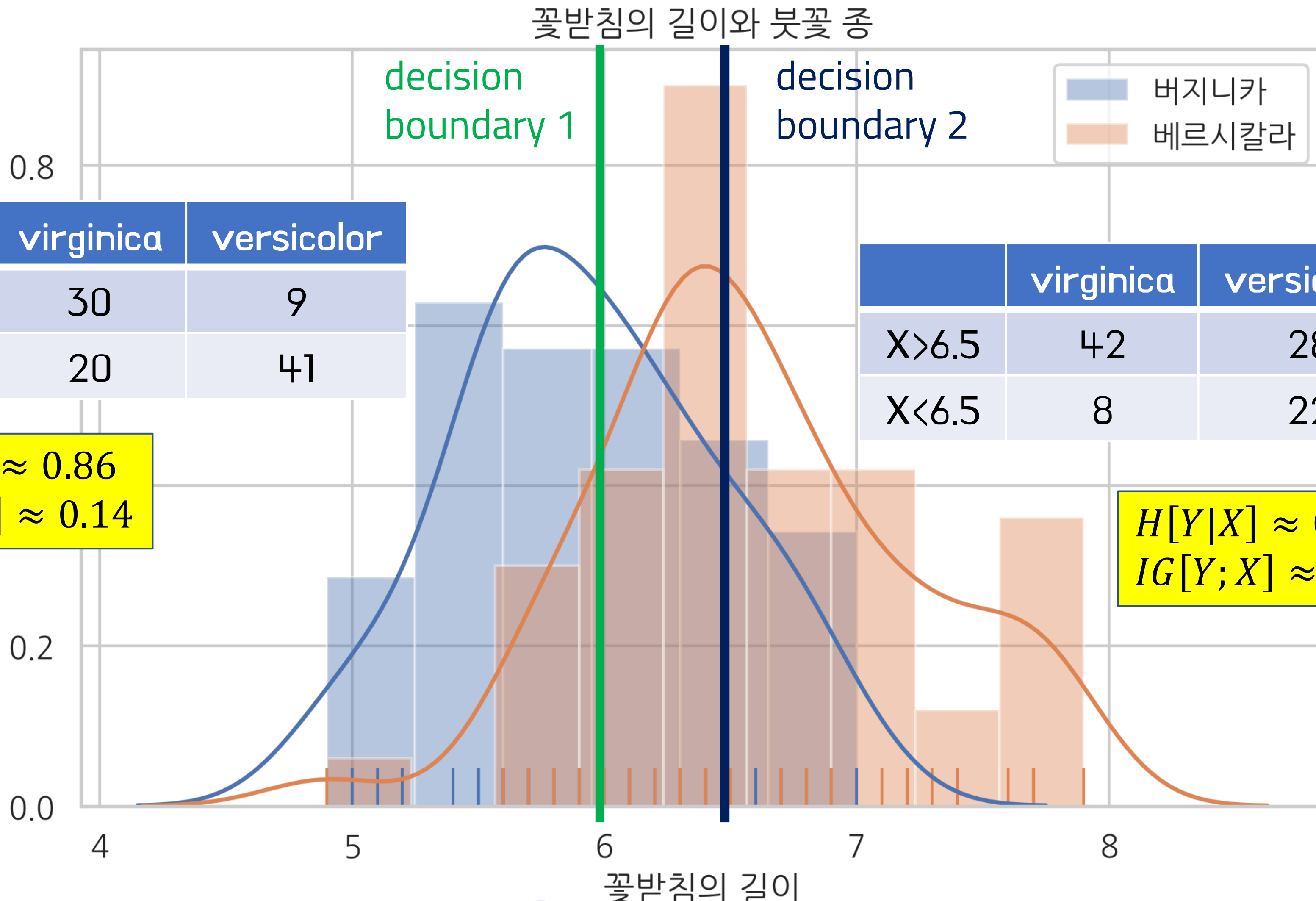
Iris classification

Y=0 (virginica)	Y=1 (versicolor)
50	50

$H[Y] = 1$



Iris classification



	virginica	versicolor
X>6	30	9
X<6	20	41

$H[Y|X] \approx 0.86$
$IG[Y;X] \approx 0.14$

	virginica	versicolor
X>6.5	42	28
X<6.5	8	22

$H[Y|X] \approx 0.93$
$IG[Y;X] \approx 0.07$

Exercise 4

(1) In the iris data, divide the interval between the minimum and maximum of sepal lengths by 0.05 intervals and draw a graph of how the conditional entropy changes when each value is used as the decision boundary value.

(2) When the sepal length is used as a feature, which value is best to use as a decision boundary value?

(3) Perform the above analysis for the sepal width. What is the best decision boundary value in this case?

(4) If only one of sepal length and sepal width had to be selected as a feature, which one should it be selected?

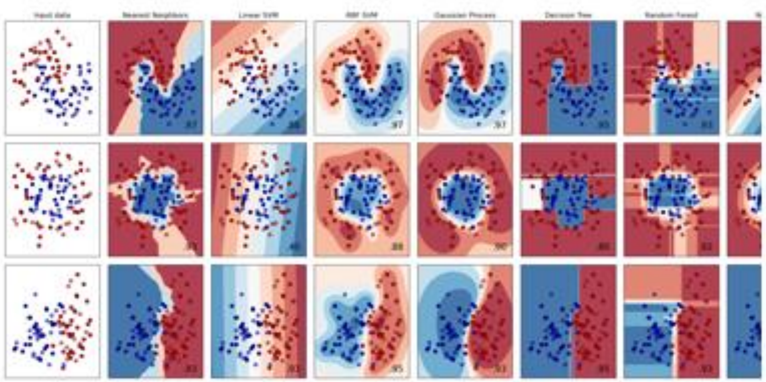
Scikit-Learn

scikit-learn.org

Classification

Identifying which category an object belongs to.

Applications: Spam detection, image recognition.
Algorithms: SVM, nearest neighbors, random forest, and more...

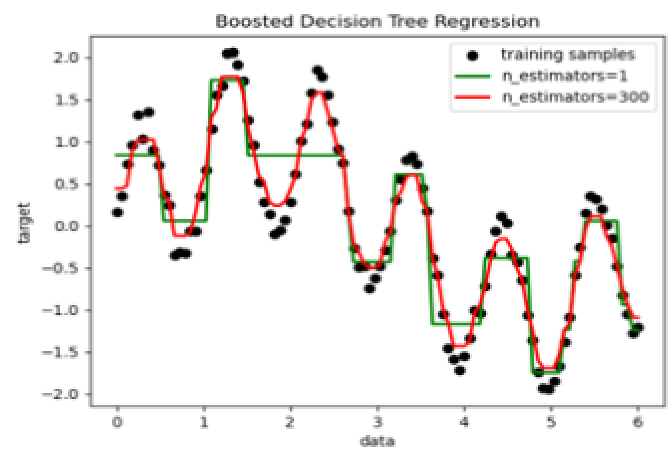


Examples

Regression

Predicting a continuous-valued attribute associated with an object.

Applications: Drug response, Stock prices.
Algorithms: SVR, nearest neighbors, random forest, and more...



Examples

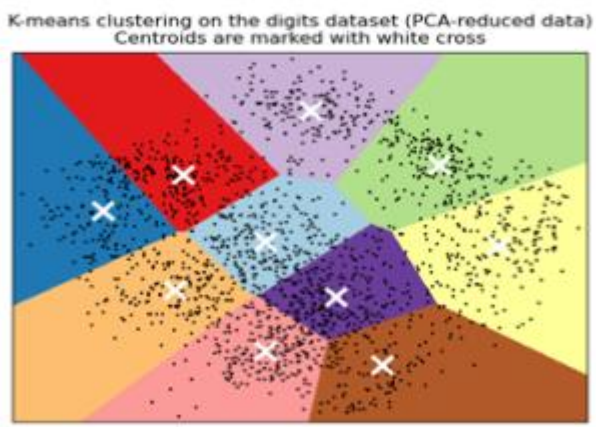
Scikit-Learn

scikit-learn.org

Clustering

Automatic grouping of similar objects into sets.

Applications: Customer segmentation, Grouping experiment outcomes
Algorithms: k-Means, spectral clustering, mean-shift, and more...

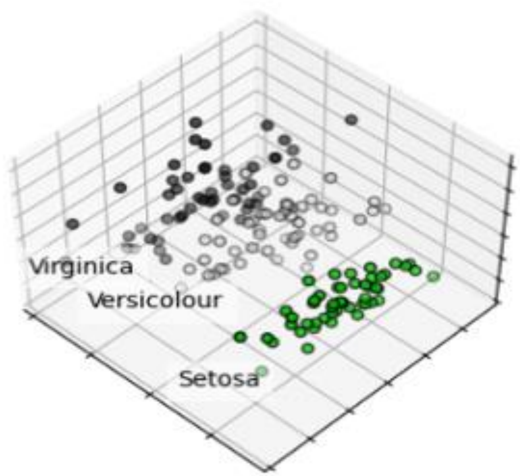


Examples

Dimensionality reduction

Reducing the number of random variables to consider.

Applications: Visualization, Increased efficiency
Algorithms: PCA, feature selection, non-negative matrix factorization, and more...



Examples

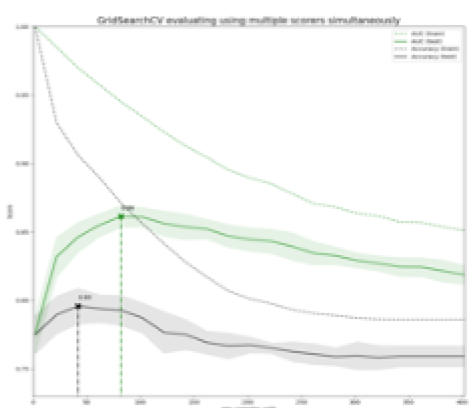
Scikit-Learn

scikit-learn.org

Model selection

Comparing, validating and choosing parameters and models.

Applications: Improved accuracy via parameter tuning
Algorithms: grid search, cross validation, metrics, and more...

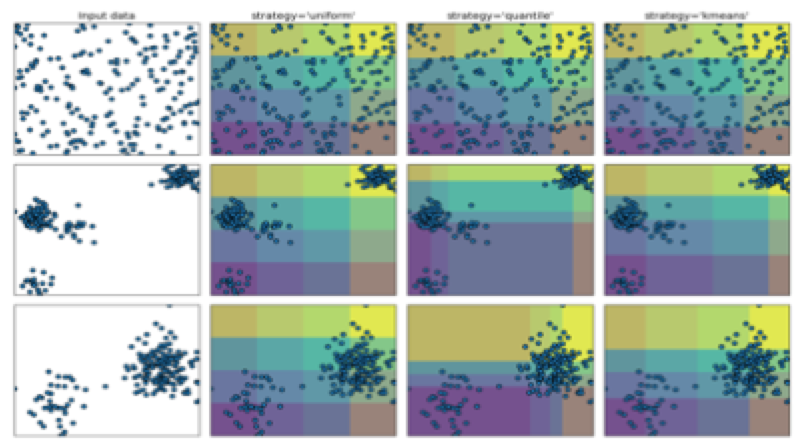


Examples

Preprocessing

Feature extraction and normalization.

Applications: Transforming input data such as text for use with machine learning algorithms.
Algorithms: preprocessing, feature extraction, and more...



Examples

Scikit-Learn

scikit learn

scikit-learn.org

User Guide

1. Supervised learning

- 1.10. Decision Trees
- 1.11. Ensemble methods

2. Unsupervised learning

3. Model selection and evaluation

4. Inspection

5. Visualizations

6. Dataset transformations

7. Dataset loading utilities

Decision tree

scikit-learn.org

```
from sklearn.datasets import load_iris
from sklearn import tree

iris = load_iris()
X, y = iris.data, iris.target
```

```
clf = tree.DecisionTreeClassifier(max_depth=3)
```

build a classifier

```
clf = clf.fit(X, y)
```

train the classifier with data

```
clf.predict(X)
```

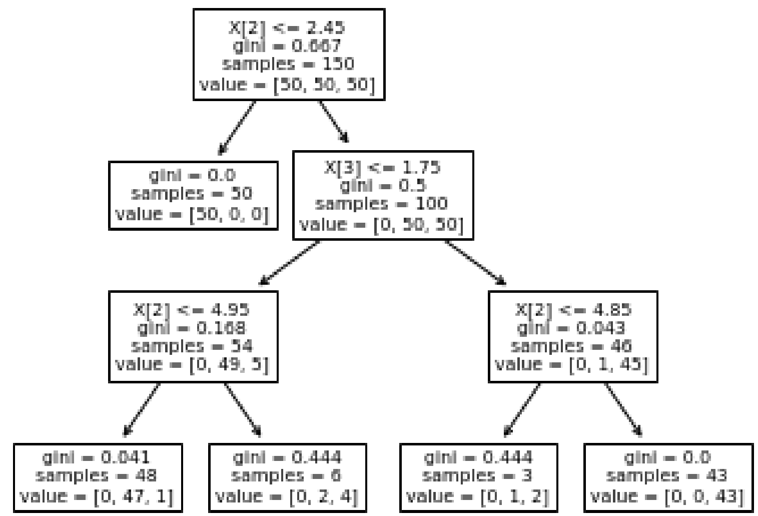
predict the class

```
clf.predict_proba(X)
```

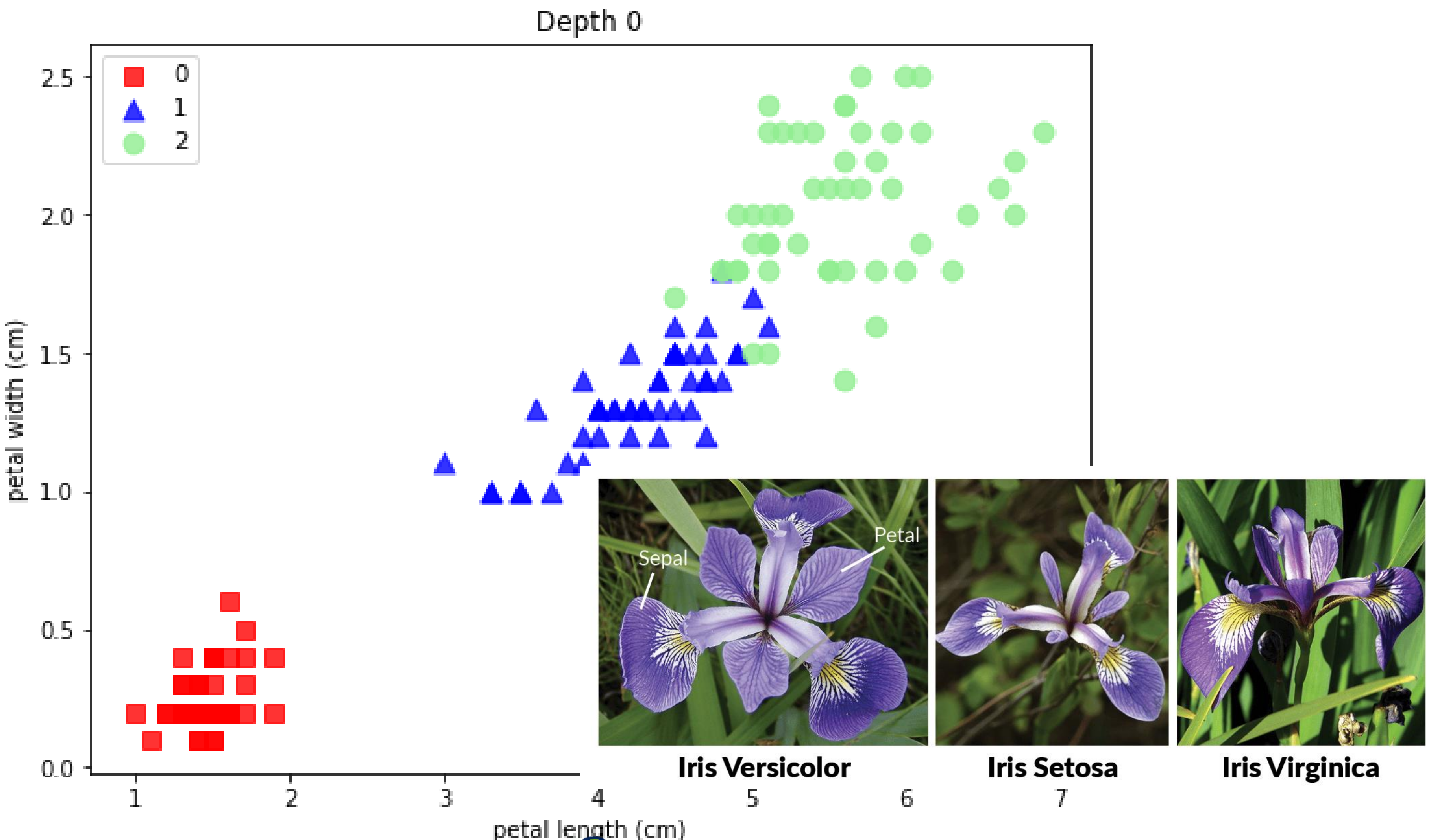
predict the probability of each class

```
clf.score(X, y)
```

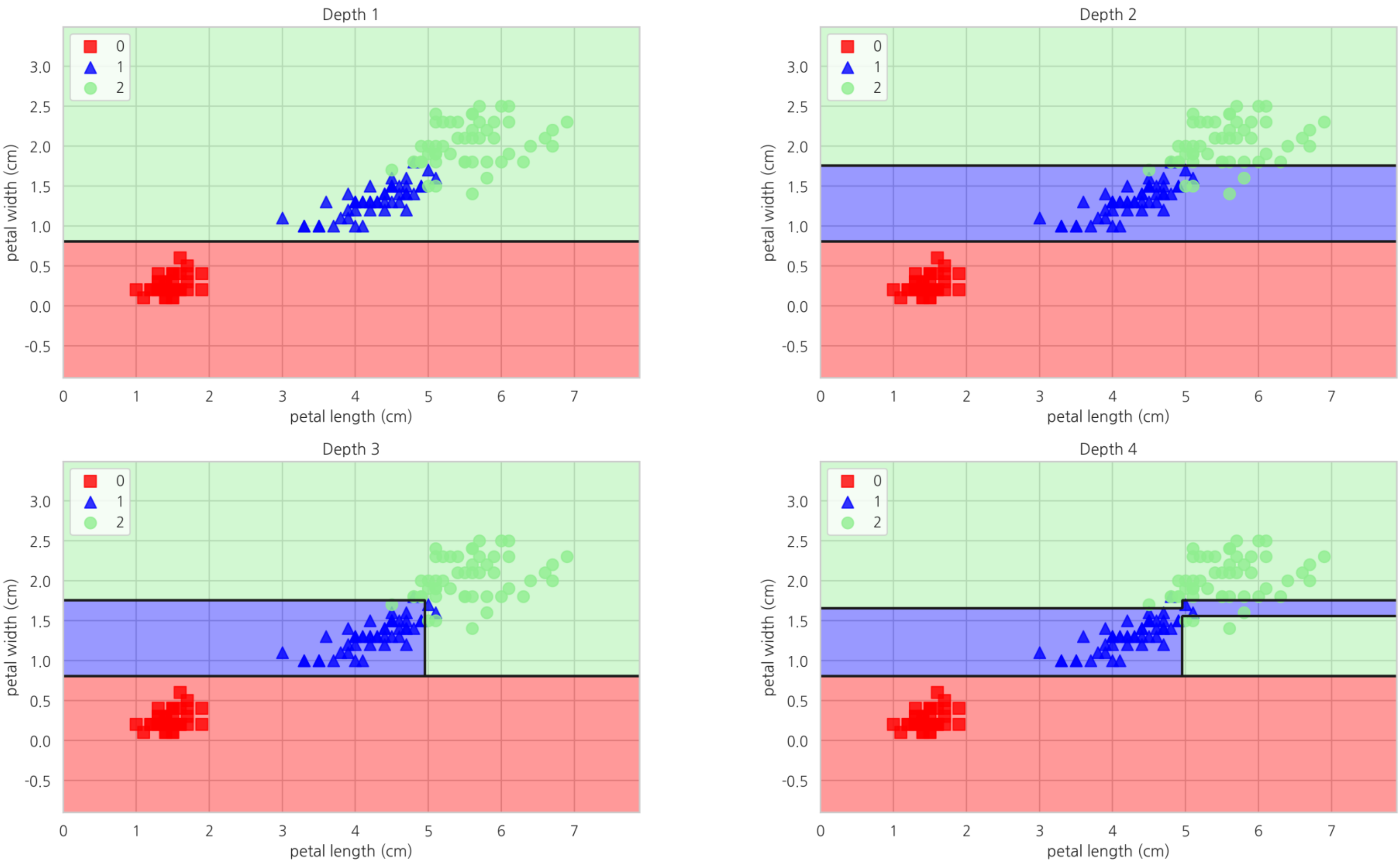
test the performance of the classifier



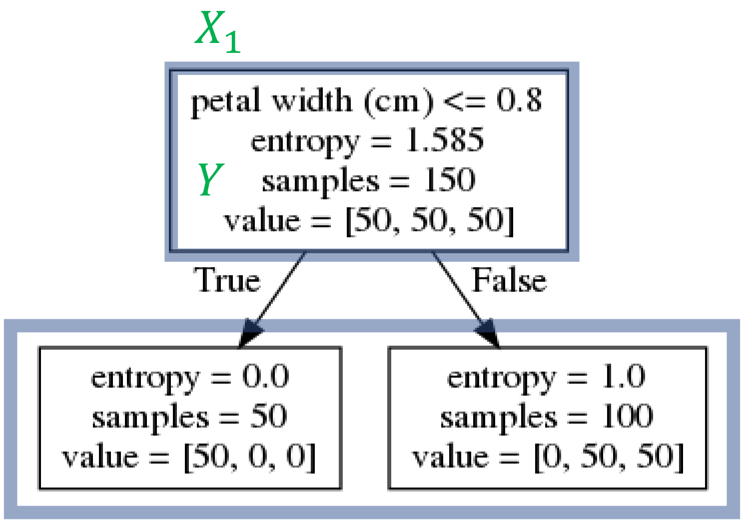
Decision tree



Decision tree



Decision tree



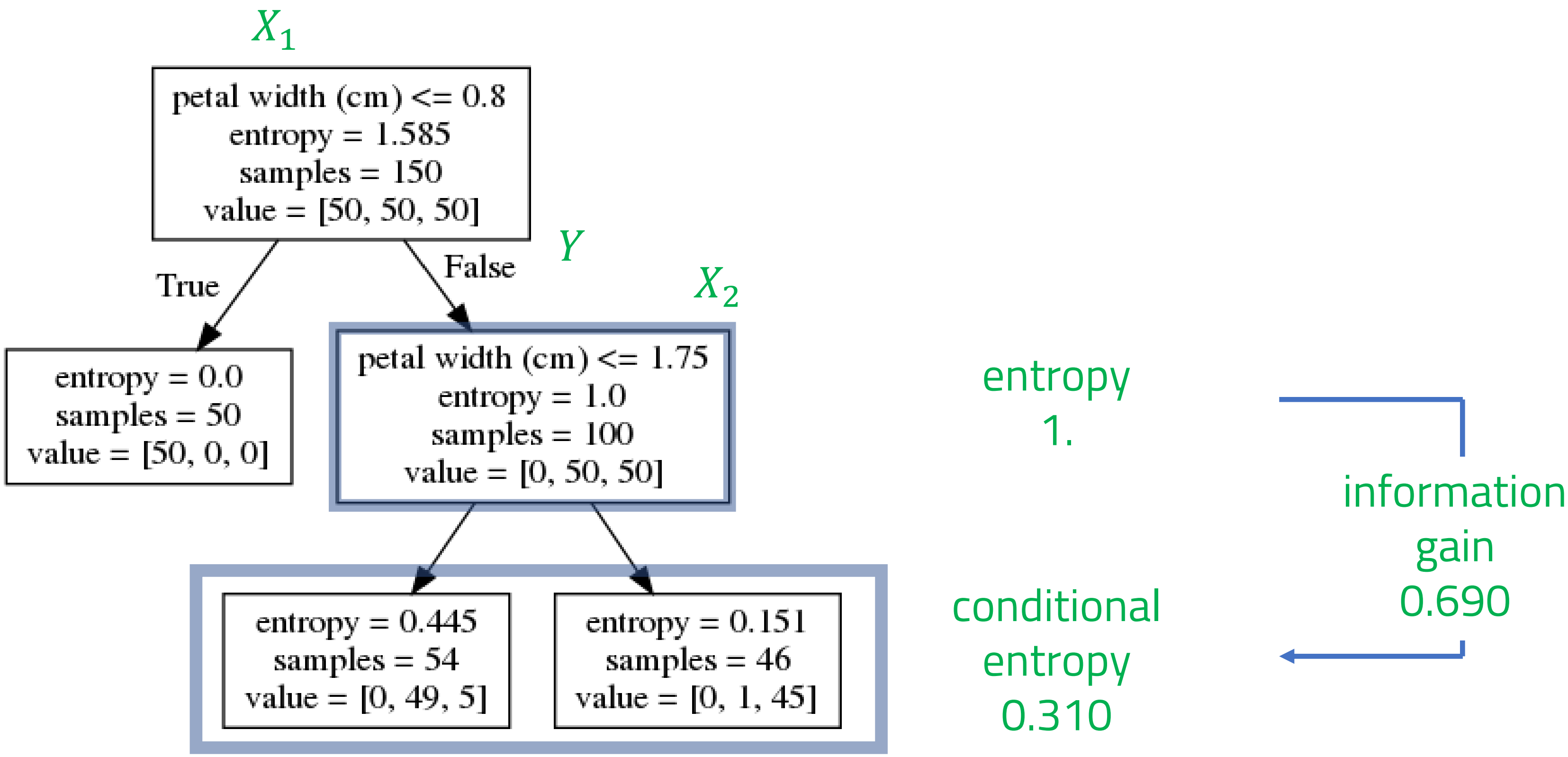
entropy
1.585

information gain
0.918

conditional
entropy
0.667

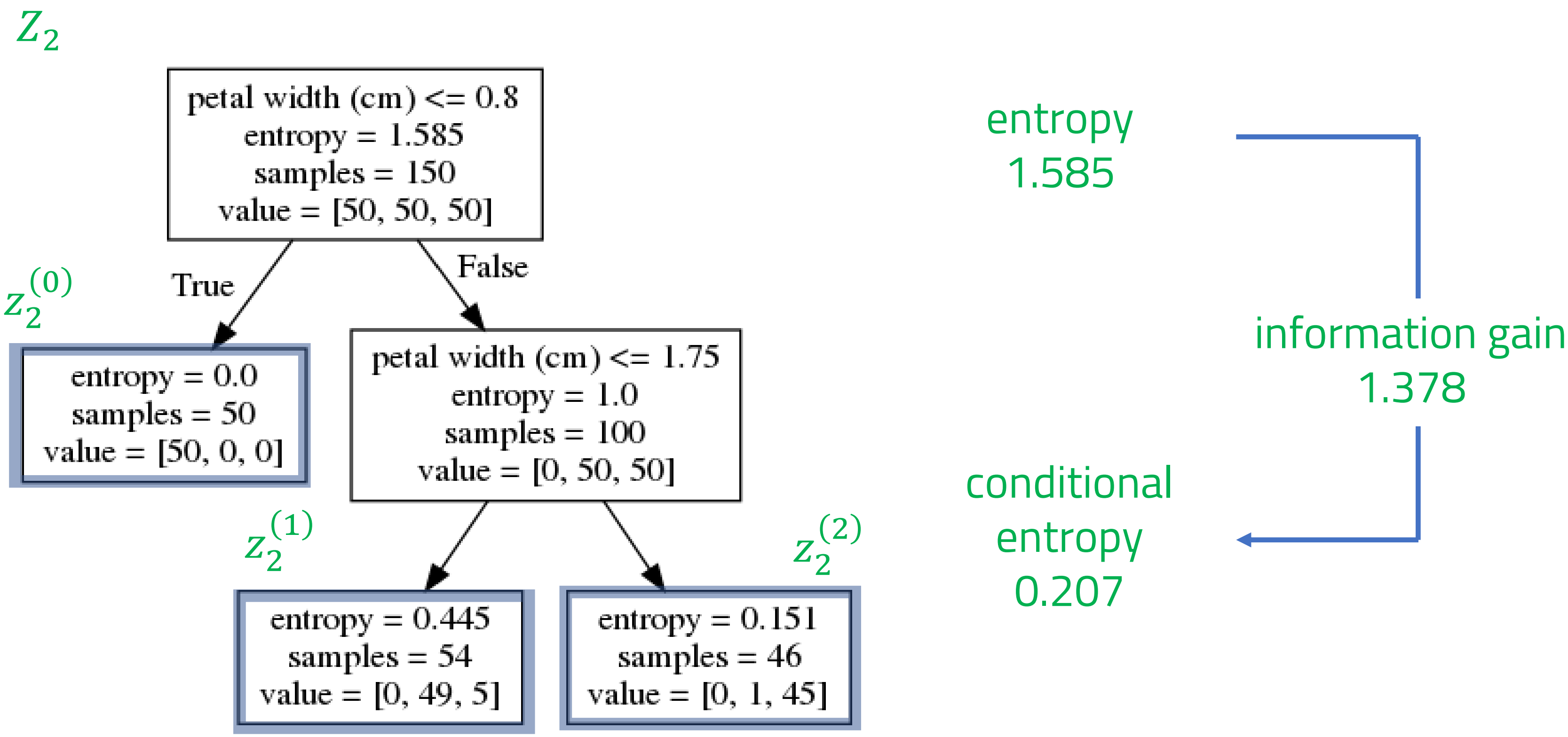
- $H[Y] = -\frac{50}{150}\log_2\frac{50}{150} - \frac{50}{150}\log_2\frac{50}{150} - \frac{50}{150}\log_2\frac{50}{150} \approx 1.585$
- $H[Y|X_1 \le 0.8] = -\frac{50}{50}\log_2\frac{50}{50} - \frac{0}{50}\log_2\frac{0}{50} - \frac{0}{50}\log_2\frac{0}{50} = 0$
- $H[Y|X_1 > 0.8] = -\frac{0}{100}\log_2\frac{0}{100} - \frac{50}{100}\log_2\frac{50}{100} - \frac{50}{100}\log_2\frac{50}{100} = 1$
- $H[Y|X_1] = H[Y|X_1 \le 0.8] \times \frac{50}{150} + H[Y|X_1 > 0.8] \times \frac{100}{150} \approx 0.667$

Decision tree



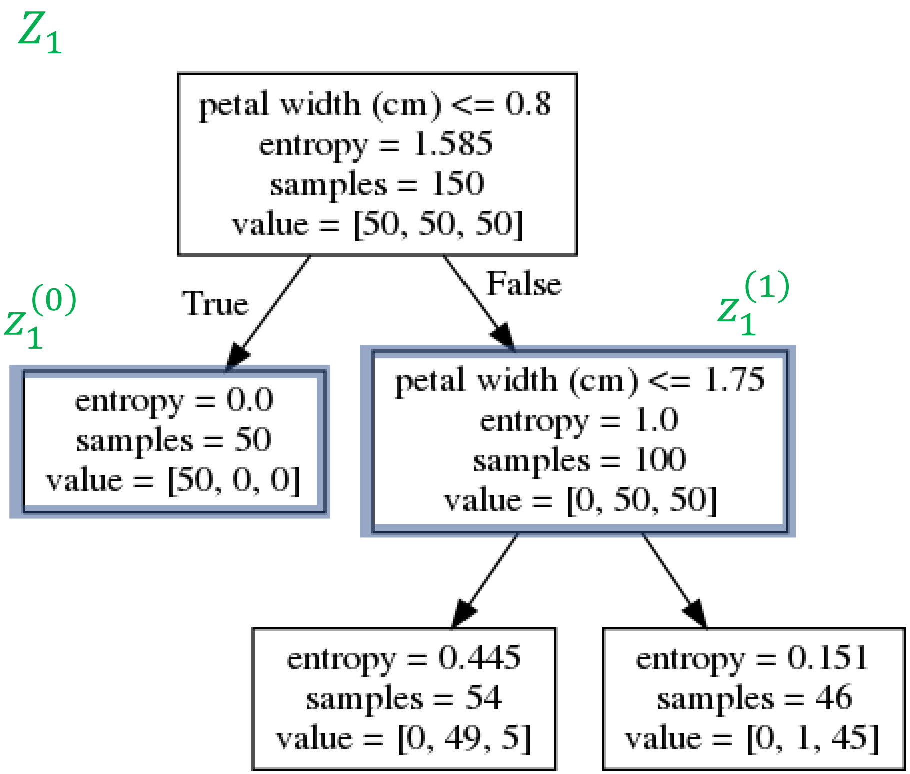
- $H[Y|X_1 > 0.8] = 1$
- $H[Y|X_2 \leq 1.75] \approx 0.445,\ H[Y|X_2 > 1.75] \approx 0.151$
- $H[Y|X_1 > 0.8, X_2] \approx 0.445 \times \frac{54}{100} + 0.151 \times \frac{46}{100} \approx 0.310$

Decision tree



- $H\left[Y|Z_2 = z_2^{(0)}\right] = 0,\ H\left[Y|Z_2 = z_2^{(1)}\right] \approx 0.445,\ H\left[Y|Z_2 = z_2^{(2)}\right] \approx 0.151$
- $H[Y|Z_2] = \frac{50}{150} H\left[Y|Z_2 = z_2^{(0)}\right] + \frac{54}{150} H\left[Y|Z_2 = z_2^{(1)}\right] + \frac{46}{150} H\left[Y|Z_2 = z_2^{(2)}\right] \approx 0.207$

Decision tree



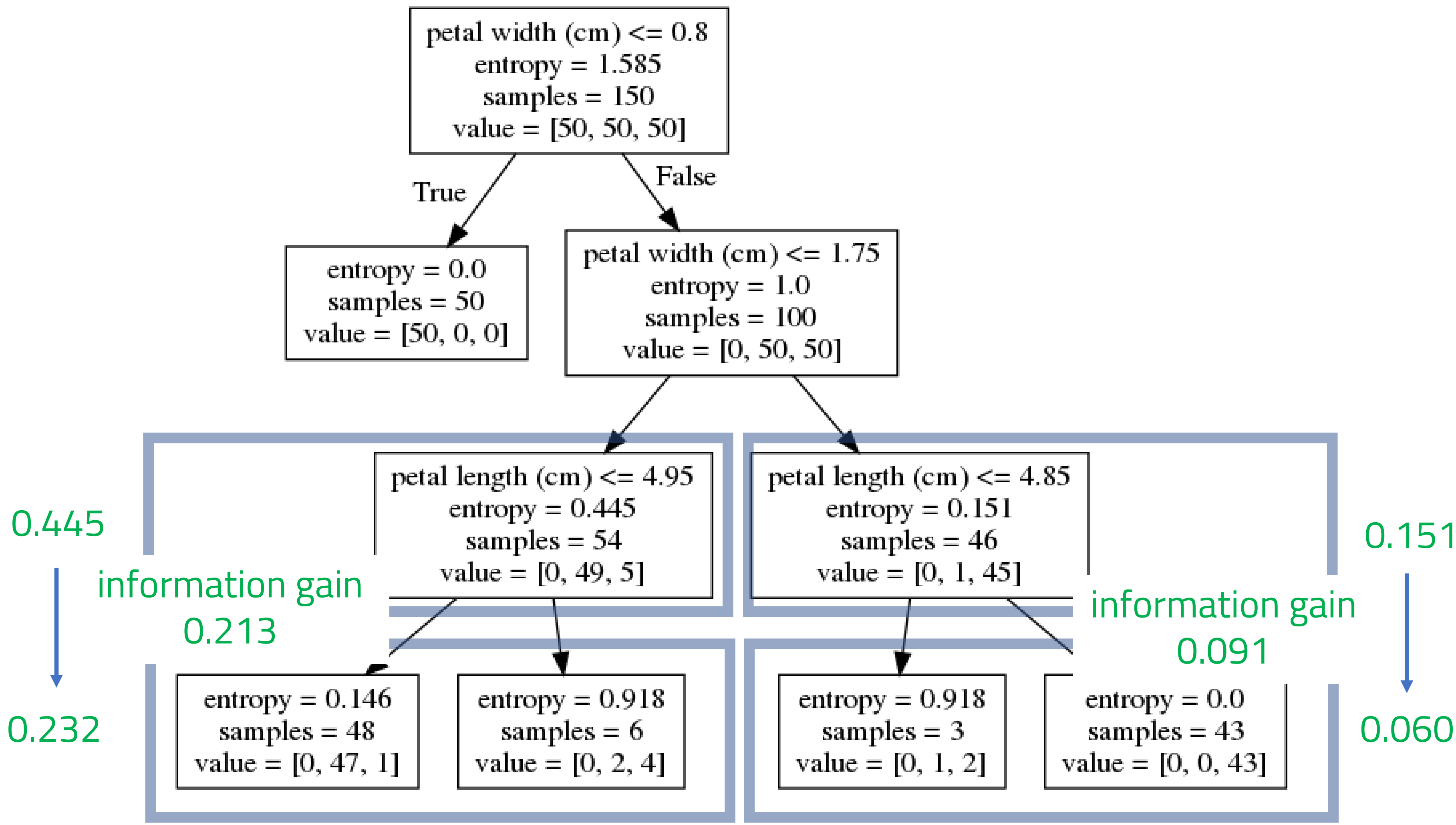
entropy
1.585

information gain
0.918

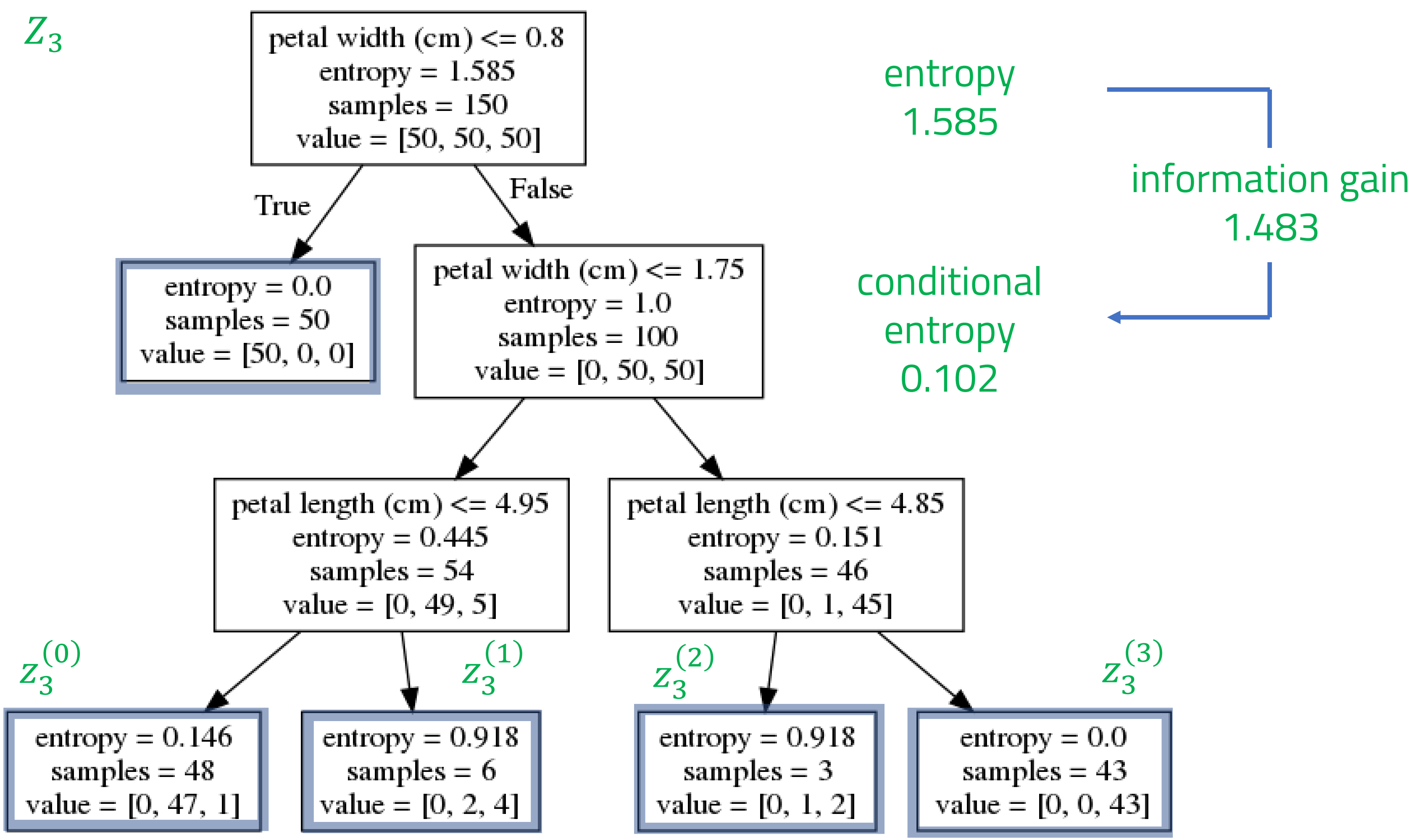
conditional
entropy
0.667

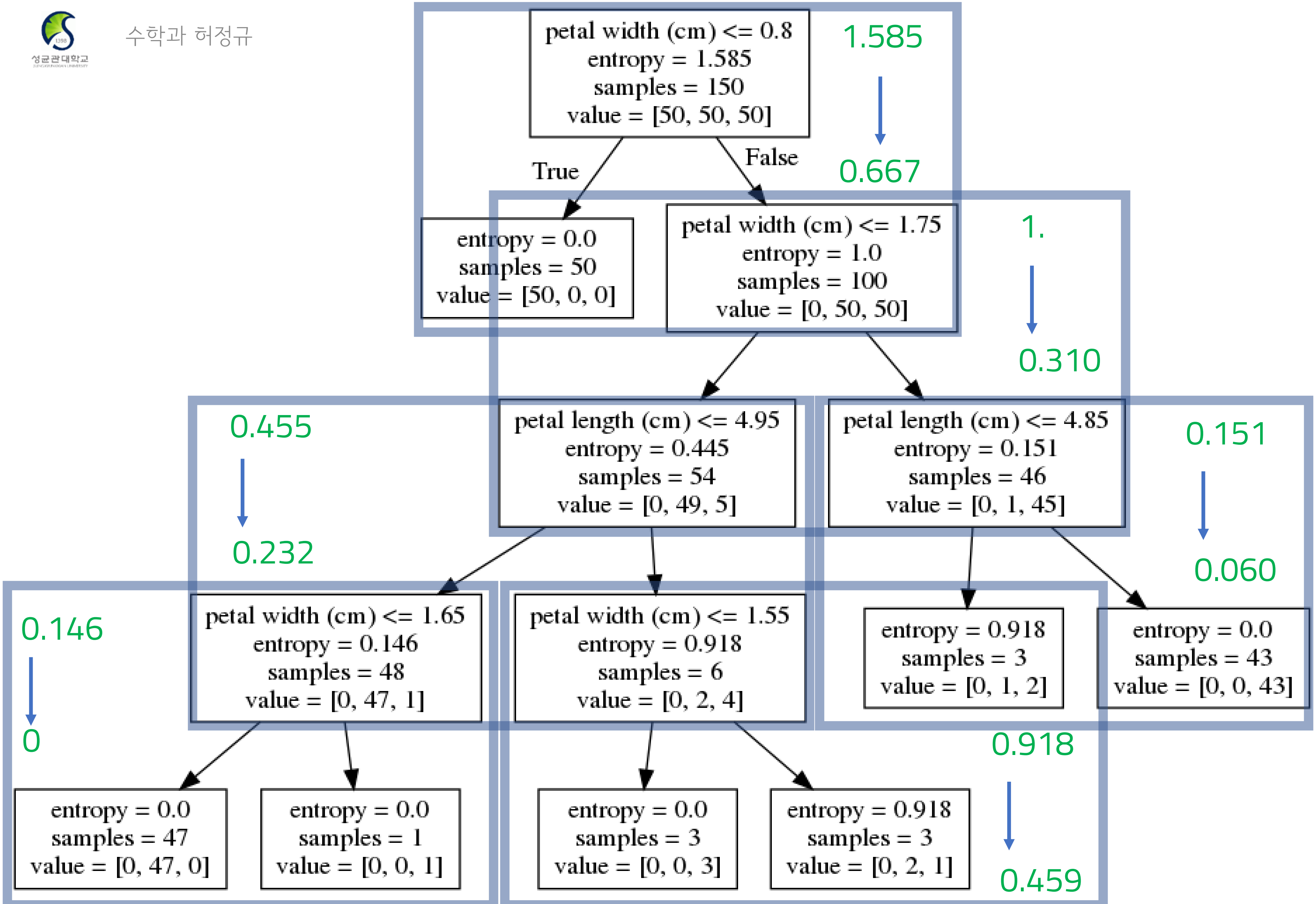
- $H\left[Y|Z_1 = z_1^{(0)}\right] = 0, H\left[Y|Z_1 = z_1^{(1)}\right] = 1$
- $H[Y|Z_1] = \frac{50}{150} H\left[Y|Z_1 = z_1^{(0)}\right] + \frac{100}{150} H\left[Y|Z_1 = z_1^{(1)}\right] \approx 0.667$

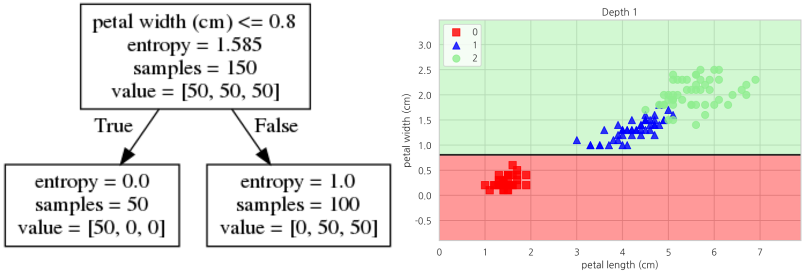
Decision tree



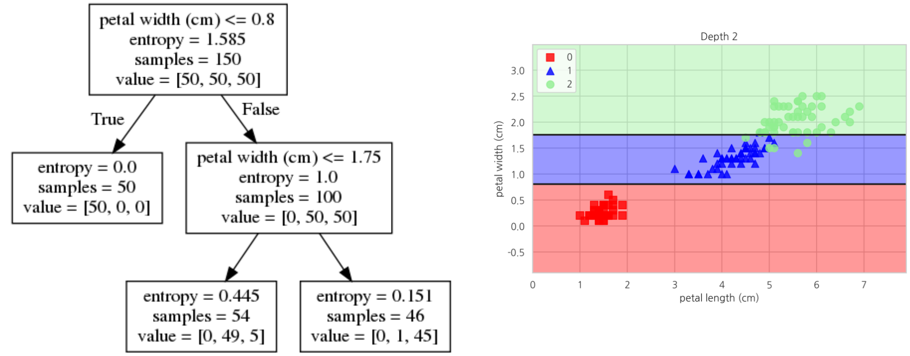
Decision tree



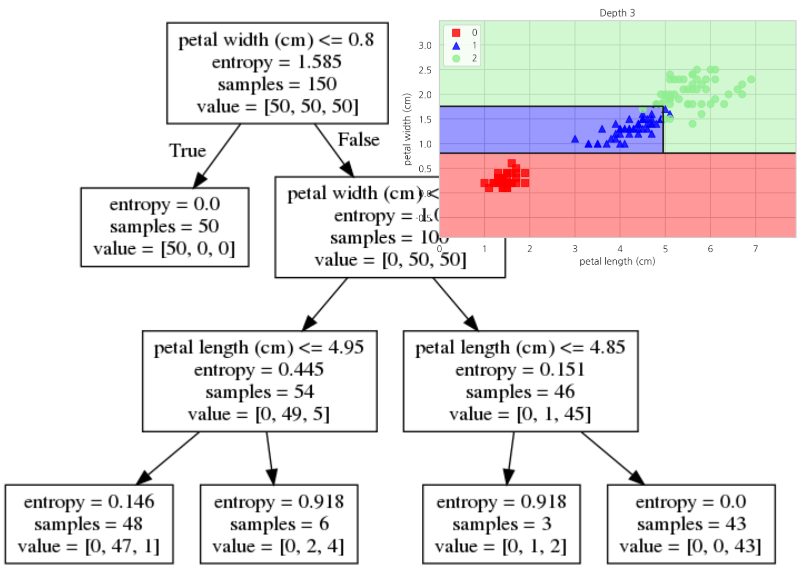

petal width (cm) <= 0.8
entropy = 1.585
samples = 150
value = [50, 50, 50]
True
False
1.585
0.667
entropy = 0.0
samples = 50
value = [50, 0, 0]
petal width (cm) <= 1.75
entropy = 1.0
samples = 100
value = [0, 50, 50]
1.
0.310
petal length (cm) <= 4.95
entropy = 0.445
samples = 54
value = [0, 49, 5]
0.455
0.232
petal length (cm) <= 4.85
entropy = 0.151
samples = 46
value = [0, 1, 45]
0.151
0.060
petal width (cm) <= 1.65
entropy = 0.146
samples = 48
value = [0, 47, 1]
0.146
0
petal width (cm) <= 1.55
entropy = 0.918
samples = 6
value = [0, 2, 4]
entropy = 0.918
samples = 3
value = [0, 1, 2]
entropy = 0.0
samples = 43
value = [0, 0, 43]
entropy = 0.0
samples = 47
value = [0, 47, 0]
entropy = 0.0
samples = 1
value = [0, 0, 1]
entropy = 0.0
samples = 3
value = [0, 0, 3]
entropy = 0.918
samples = 3
value = [0, 2, 1]
0.918
0.459



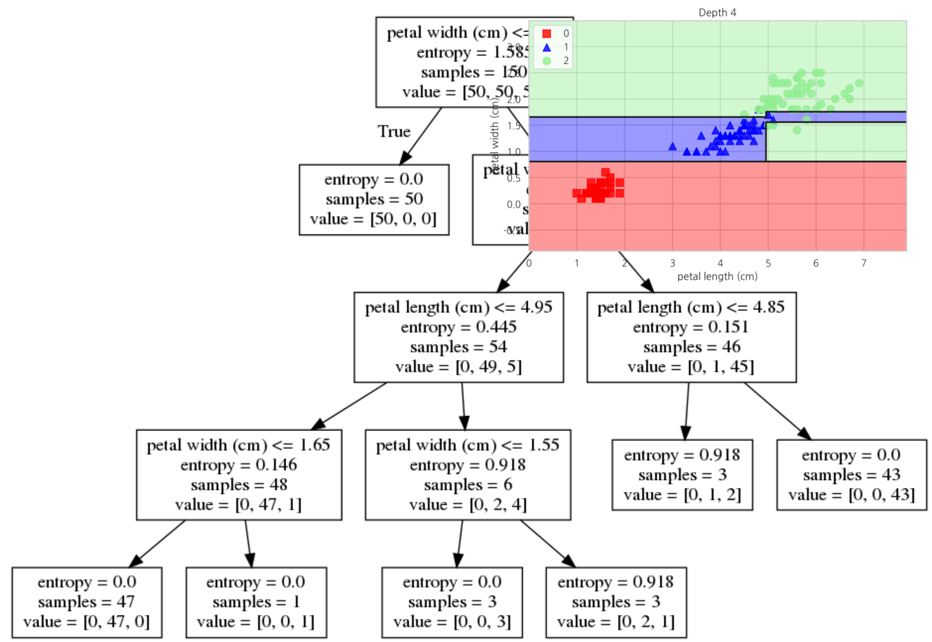
1.585 → 0.667



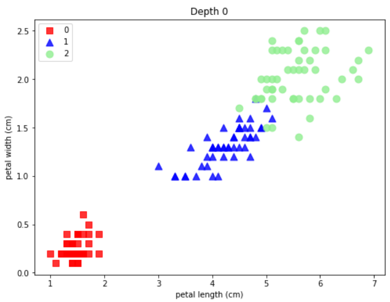
1.585 → 0.207

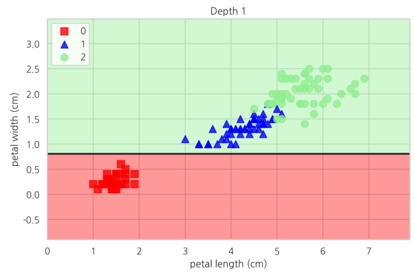


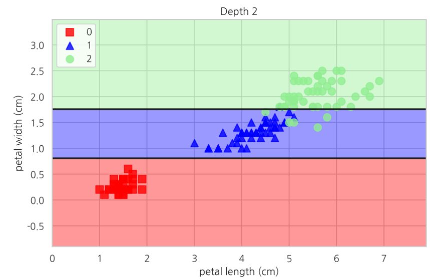
1.585 → 0.102

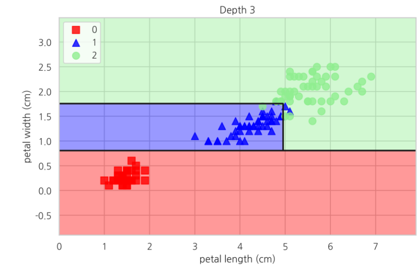


1.585 → 0.0367









high
entropy

low
entropy

A decision tree increases its depth and decrease the entropy of the data.

Exercise 5

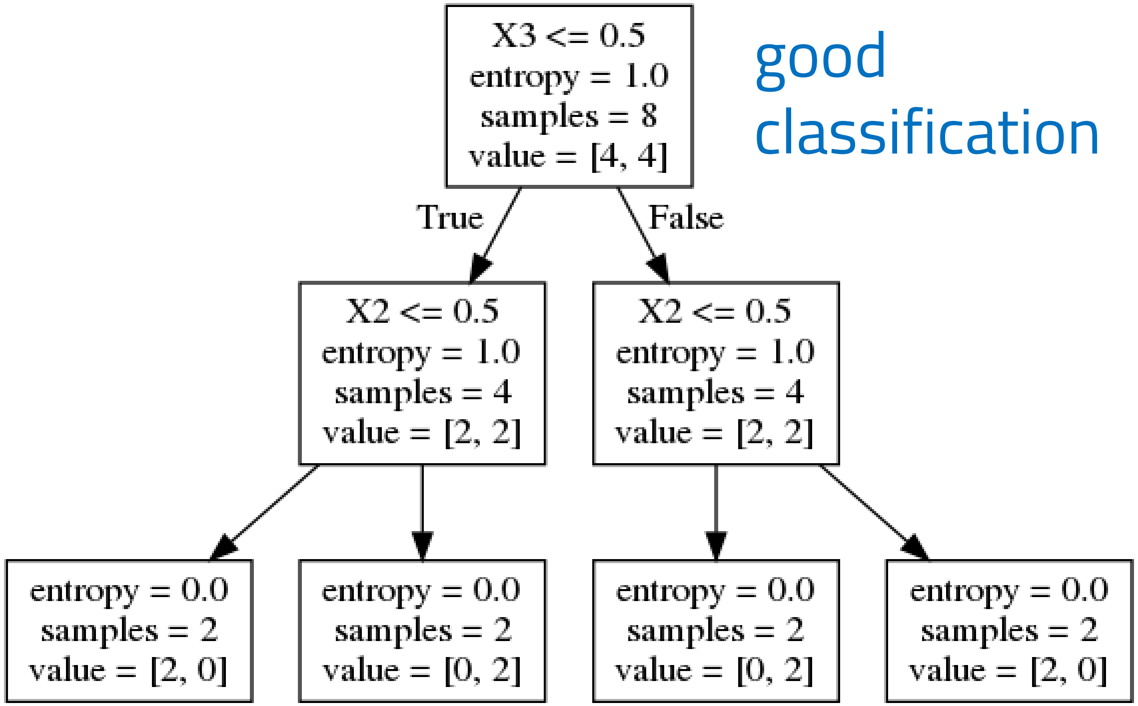
(1) For the iris classification problem, build a decision tree with max_depth = 3 using the sepal length and width and calculate the accuracy.

(2) Measure the test performance through cross-validation with K=5.

(3) While changing the max_depth argument, find which value of the argument gives the best test performance.

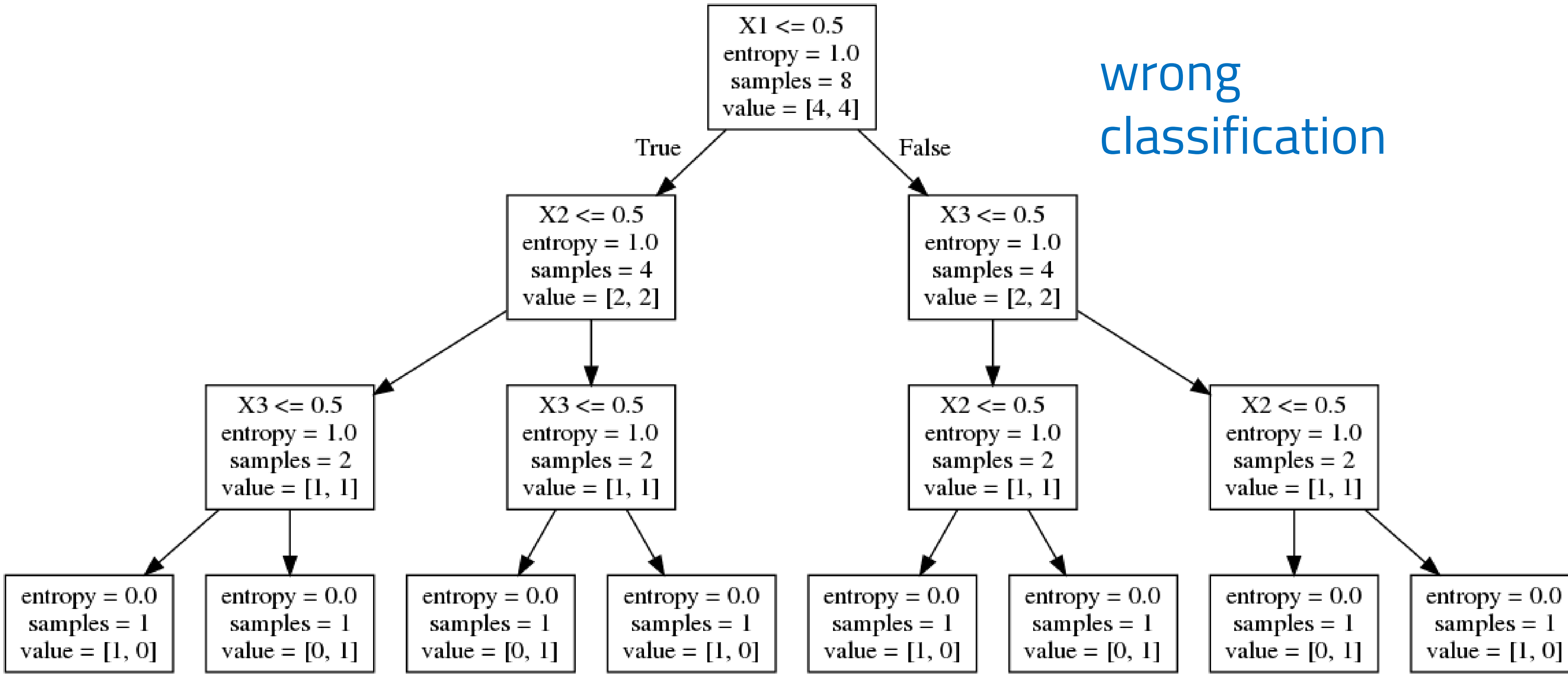
Problem of greedy algorithms

	x_1	x_2	x_3	y
1	0	0	0	0
2	1	0	0	0
3	0	0	1	1
4	1	0	1	1
5	0	1	0	1
6	1	1	0	1
7	0	1	1	0
8	1	1	1	0



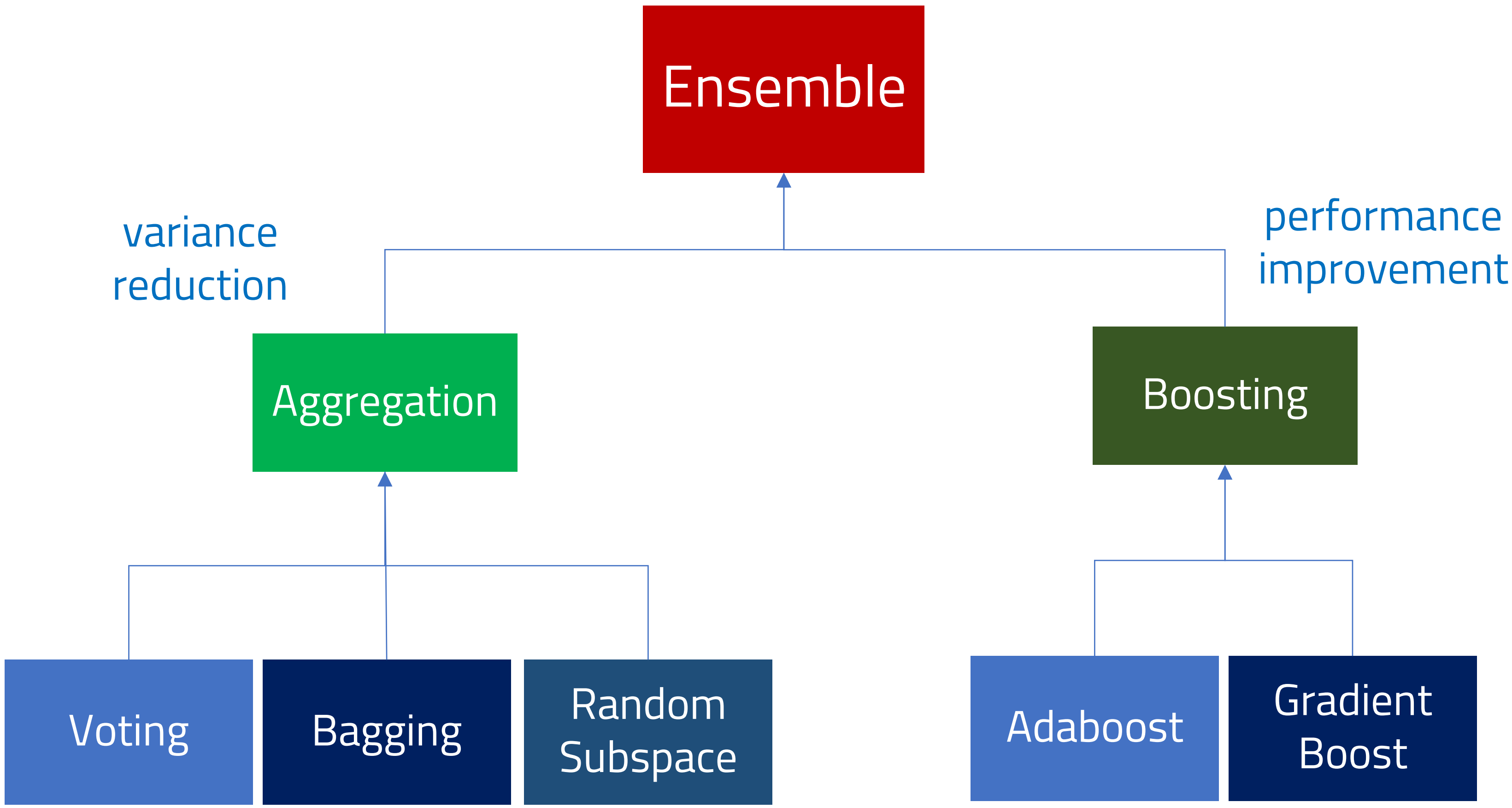
good classification

Problem of greedy algorithms

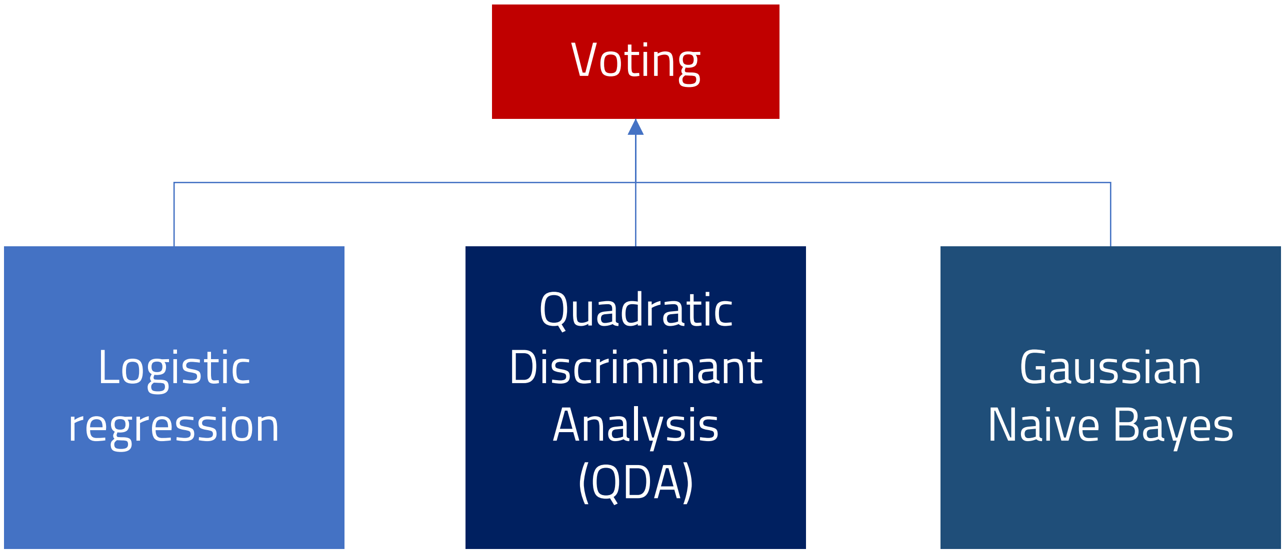


wrong classification

Ensemble


Ensemble
variance reduction
performance improvement
Aggregation
Boosting
Voting
Bagging
Random Subspace
Adaboost
Gradient Boost

Voting



- hard voting : every individual classifier votes for a class, and the majority wins.
- soft voting : every individual classifier provides a probability value that a specific data point belongs to a particular target class. The predictions are weighted by the classifier's importance and summed up. Then the target label with the greatest sum of weighted probabilities wins the vote.

Voting

```
from sklearn.linear_model import LogisticRegression
from sklearn.naive_bayes import GaussianNB
from sklearn.discriminant_analysis import QuadraticDiscriminantAnalysis
from sklearn.ensemble import VotingClassifier

model1 = LogisticRegression(random_state=1)
model2 = QuadraticDiscriminantAnalysis()
model3 = GaussianNB()
```

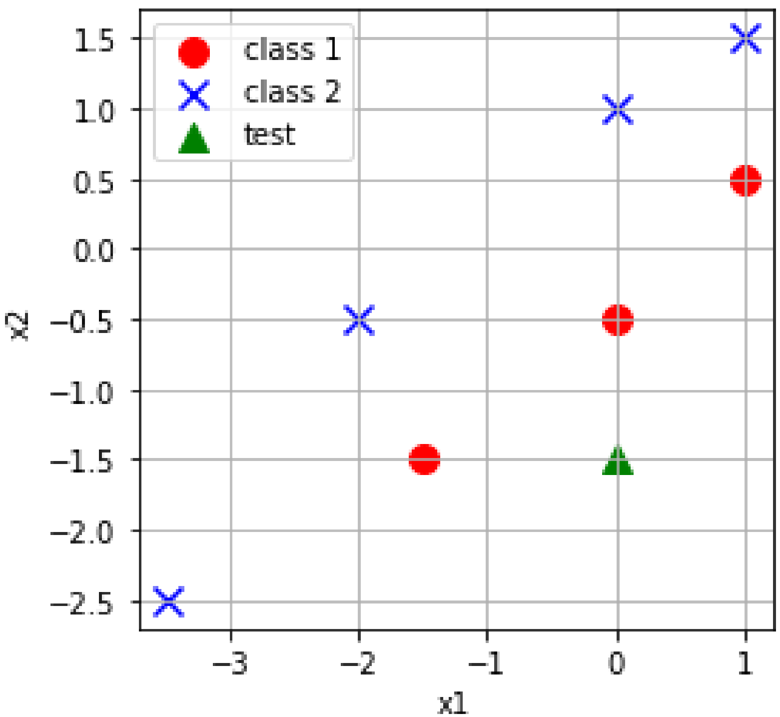
build various classifiers

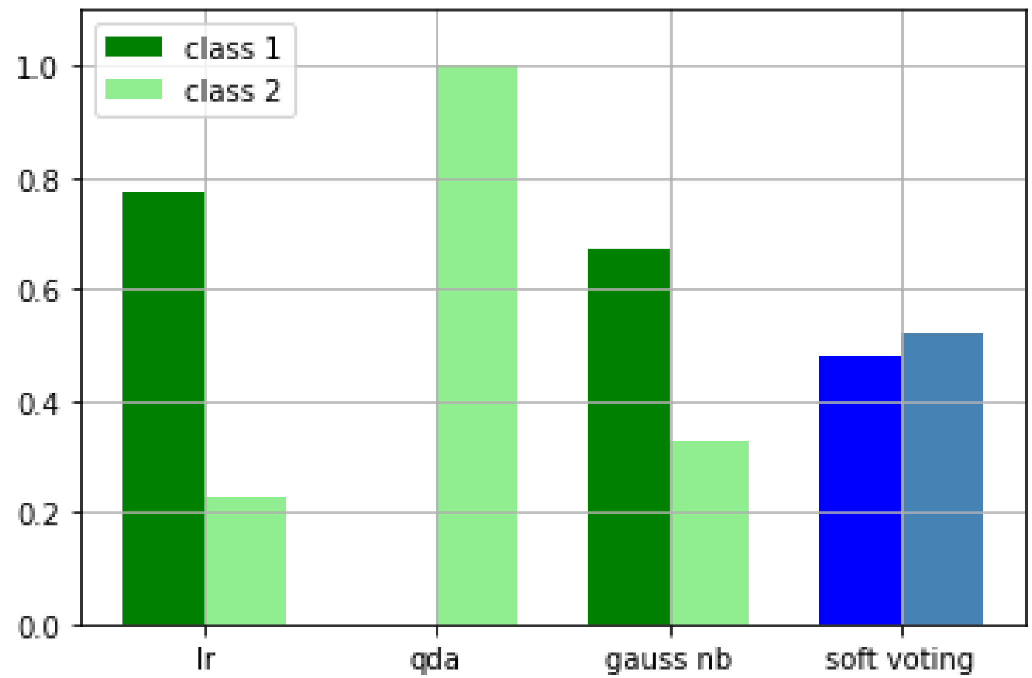
```
ensemble = VotingClassifier(
    estimators=[('lr', model1), ('qda', model2), ('gnb', model3)], voting='soft')
```

build the voting classifier, an ensemble of the classifiers

```
ensemble.fit(X,y)
ensemble.predict(X)
ensemble.predict_proba(X)
ensemble.score(X,y)
```

Voting

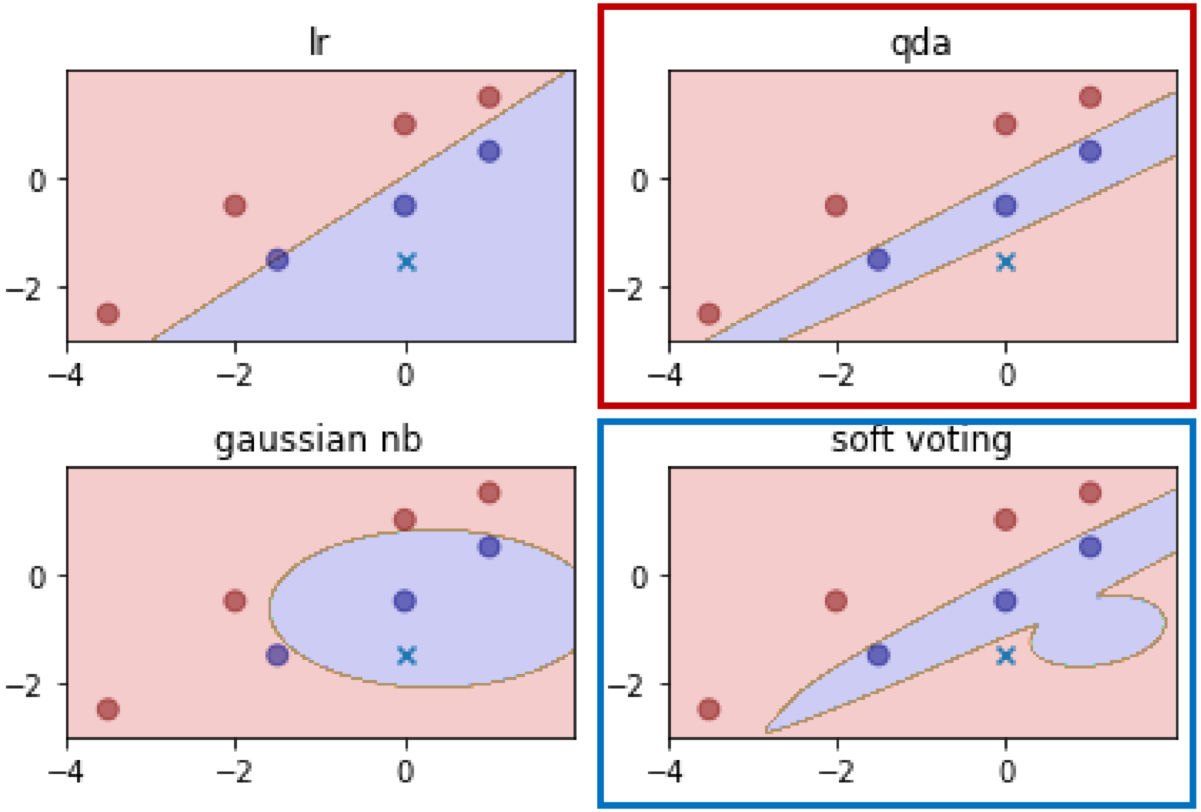




correct wrong correct wrong

- If "hard voting" method was chosen, the ensemble classifier would provide correct result.

Voting



Excessive influence

Voting

$X_i \sim \mathrm{Ber}(p)$ ⟶ $n\bar{X} \sim B(n, p)$ ⟶ $\bar{X} \sim N\left(p, \frac{p(1-p)}{n}\right)$

- $E[\bar{X}] = p$
- $Var[\bar{X}] = p(1-p)$

- $E[\bar{X}] = p$
- $Var[\bar{X}] = \frac{p(1-p)}{n}$

classifier

$X_i \sim \mathrm{Ber}(0.6)$

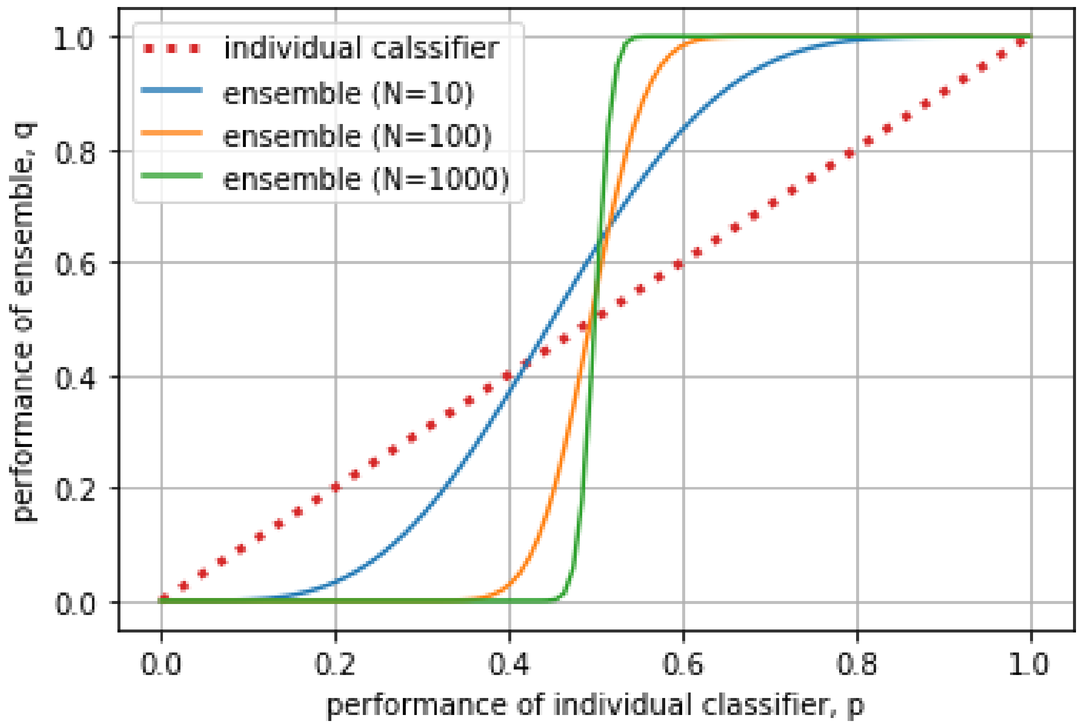
0	1	0	0	0	1	0	1	1	1	0	1	0	0	1	1	1	1	1	0	1	0	0	0	1	1	0	1	0	1
-1	0	-1	-1	-1	-1	-1	-1	-1	0	-1	0	-1	-1	-1	0	1	1	1	1	1	1	1	0	1	1	1	1	1	1

$\bar{X}_{1:i}$

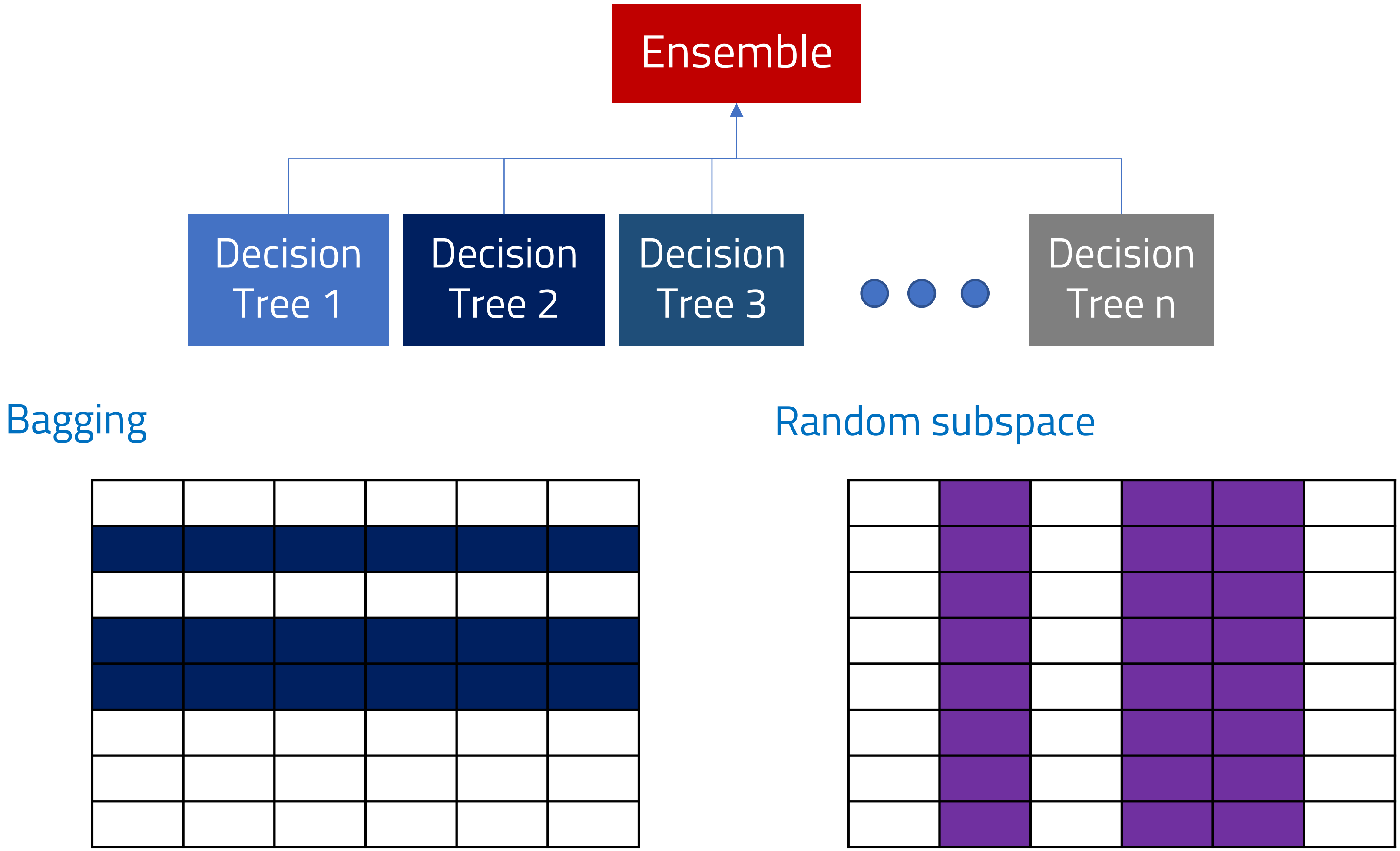
ensemble

Exercise 6

Draw the following graphs. Individual classifiers provide correct predictions with probability p, independent of each other. The ensemble classifier consists of the individual classifiers according to the soft voting method, and it gives correct prediction with probability q due to the ensemble effect.



Bagging, Random subspace



Bagging, Random subspace

Bagging

```
from sklearn.ensemble import BaggingClassifier
model = BaggingClassifier(DecisionTreeClassifier(max_depth=2), n_estimators=100)
```

Random subspace

```
from sklearn.ensemble import RandomForestClassifier
model = RandomForestClassifier(max_depth=2, n_estimators=100)
```

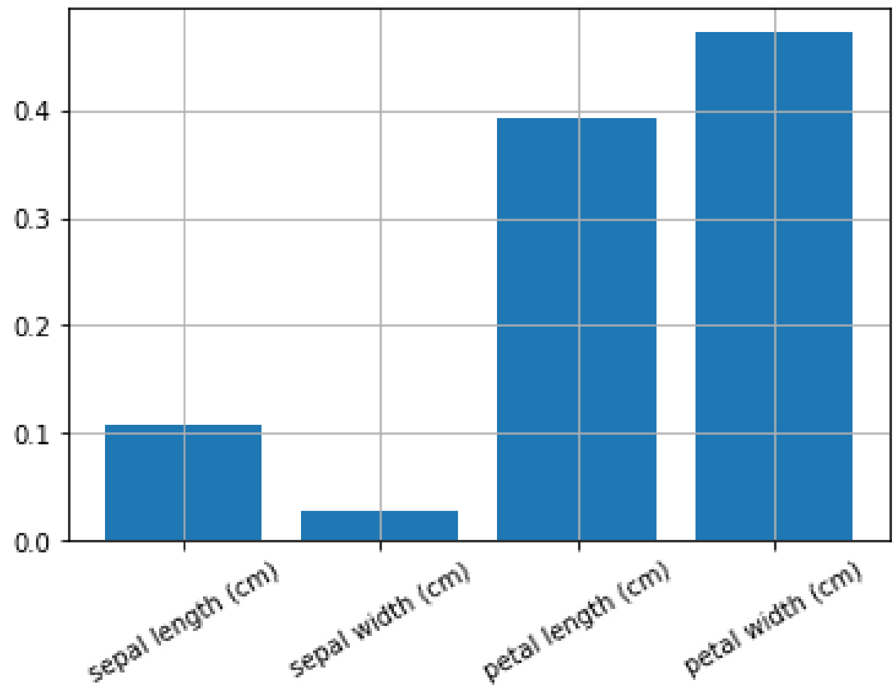
Feature importance

```
from sklearn.datasets import load_iris
from sklearn.ensemble import RandomForestClassifier

iris = load_iris()
X, y = iris.data, iris.target

model = RandomForestClassifier()
model = model.fit(X,y)

model.feature_importances_
```



Boosting

Commitee

- $\boxed{C_1} = \{k_1\}$

Classifier

- $C_2 = C_1 \cup \{k_2\} = \{k_1, \boxed{k_2}\}$
- $C_3 = C_2 \cup \{k_3\} = \{k_1, k_2, k_3\}$

…

- $C_m = C_{m-1} \cup \{k_m\} = \{k_1, k_2, \ldots, k_m\}$

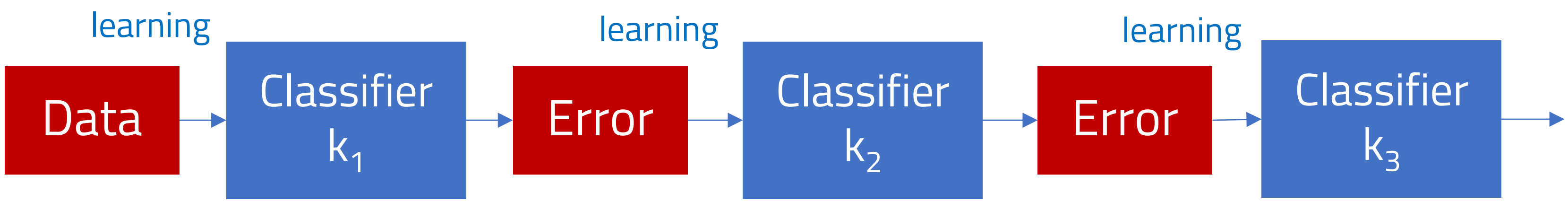
Boosting!

$y_i = -1$ or 1

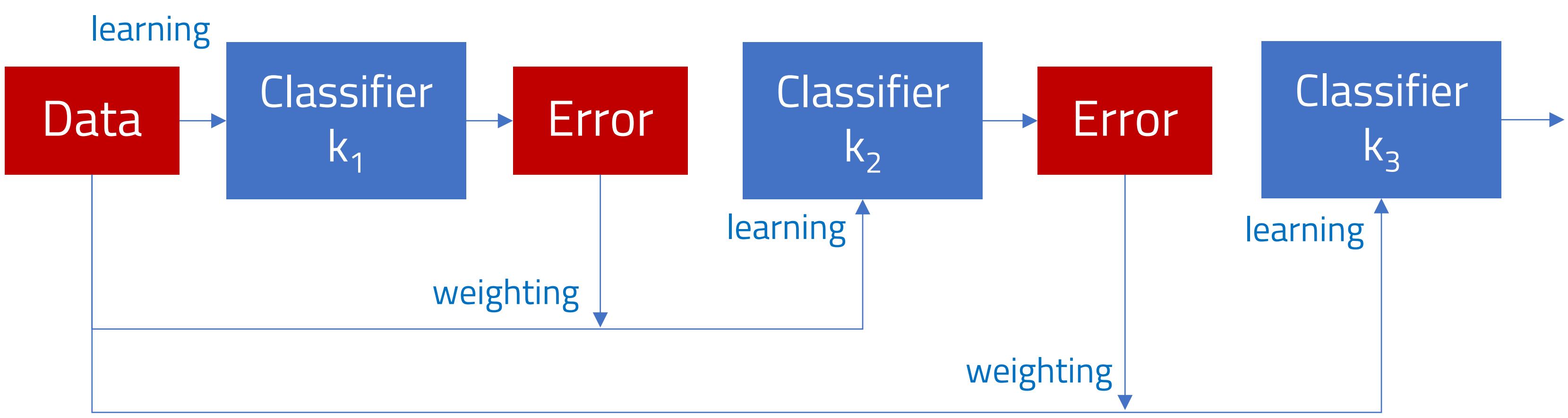
$$C_m(x_i) = \operatorname{sign}\big(\alpha_1 k_1(x_i) + \alpha_2 k_2(x_i) + \cdots + \alpha_m k_m(x_i)\big)$$

Boosting

Gradient boost



Adaboost



AdaBoost

Data

- i=1,2,3,4,5,6,7,8
- N=8

- Adaboost classifier

$$C_m(x_i) = \mathrm{sign}\big(C_{m-1}(x_i) + \alpha_m k_m(x_i)\big)$$

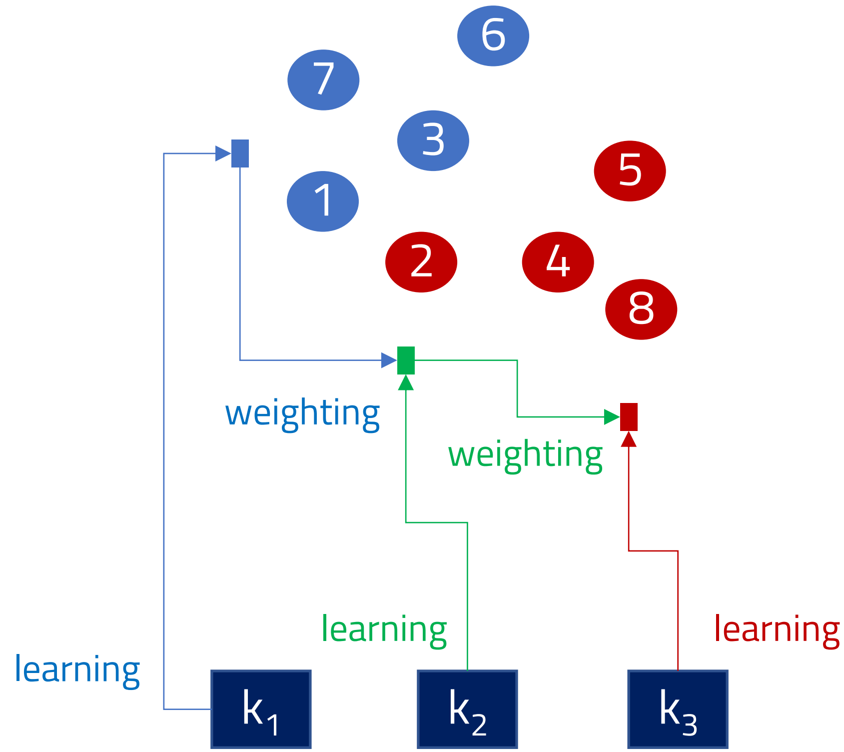
- Loss function for the m th classifier

$$L_m = \sum_{i=1}^{N} w_{m,i} I(k_m(x_i) \neq y_i)$$

where

$$w_{m,i} = w_{m-1,i} \exp\big(-y_i C_{m-1}(x_i)\big)$$

$$= \begin{cases} w_{m-1,i} e^{-1} \text{ if } C_{m-1}(x_i) = y_i \\ w_{m-1,i} e^{+1} \text{ if } C_{m-1}(x_i) \neq y_i \end{cases}$$



Classifier

- m=1,2,3

AdaBoost

Data
- i=1,2,3,4,5,6,7,8
- N=8

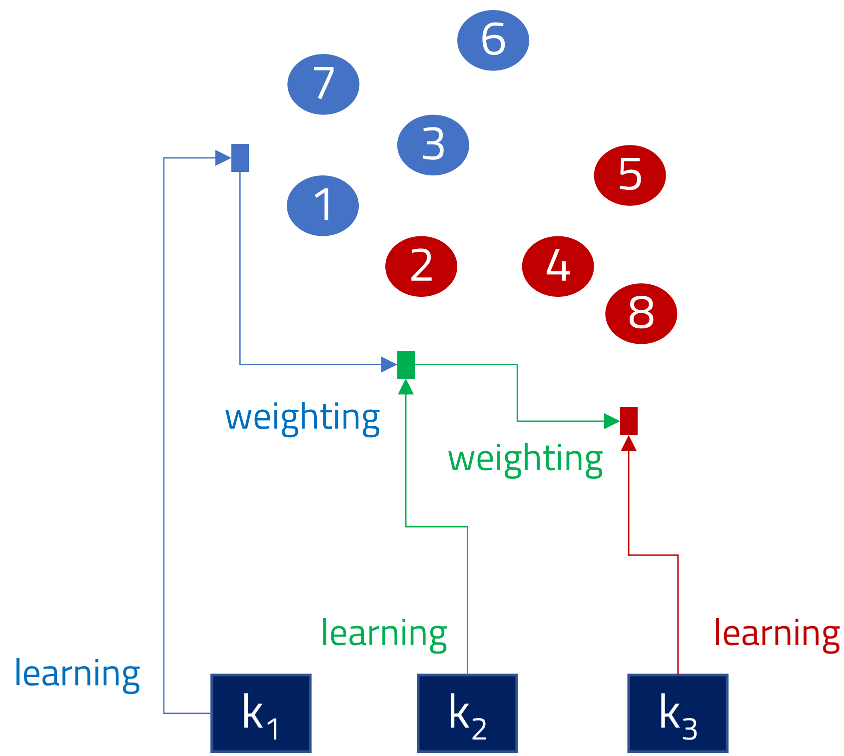
- Adaboost classifier

$$C_m(x_i) = \mathrm{sign}\big(C_{m-1}(x_i) + \alpha_m k_m(x_i)\big)$$

where

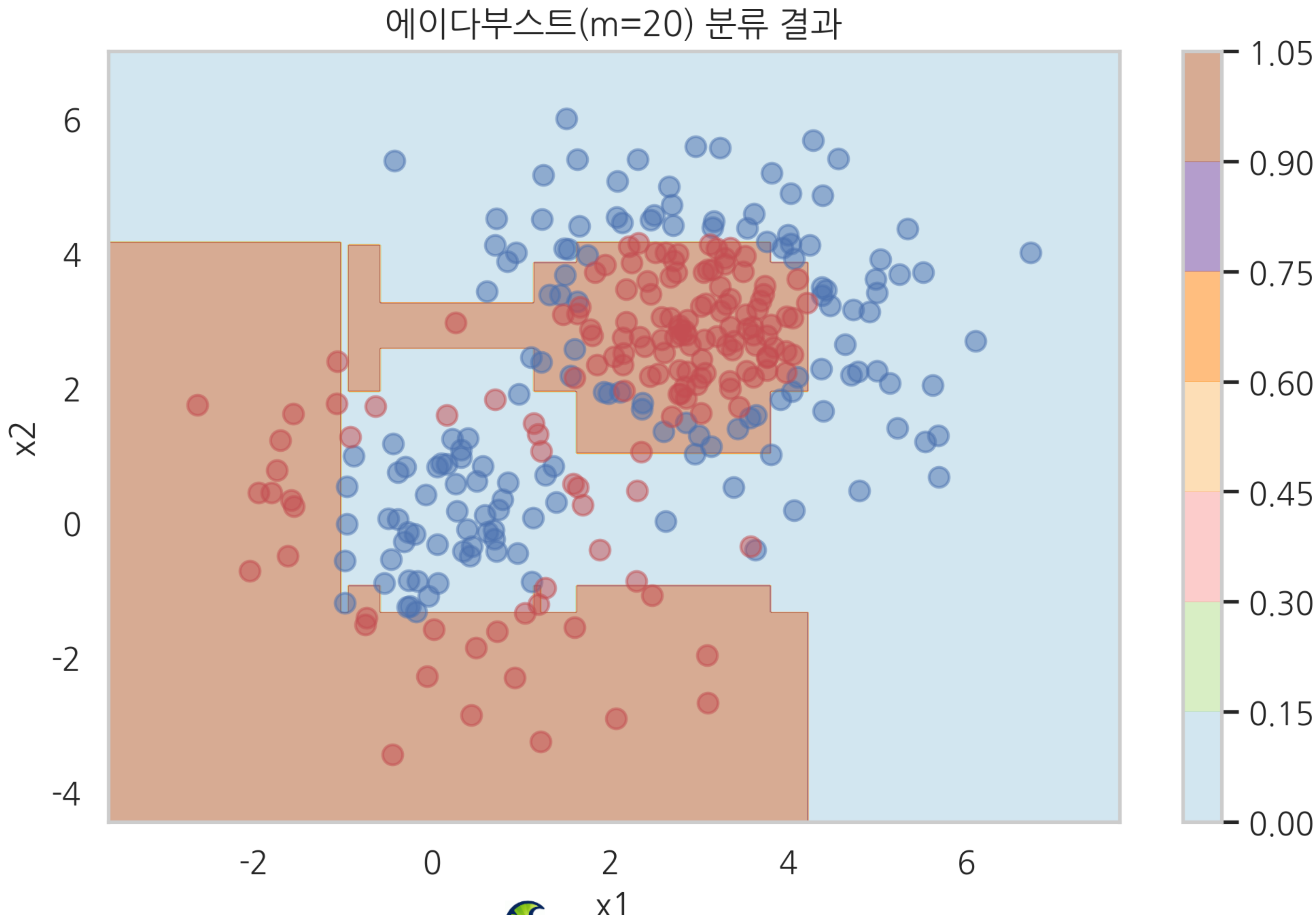
$$\alpha_m = \frac{1}{2}\log\left(\frac{1-\epsilon_m}{\epsilon_m}\right)$$

$$\epsilon_m = \frac{L_m}{\sum_{i=1}^{N} w_{m,i}}$$



Classifier
- m=1,2,3

AdaBoost

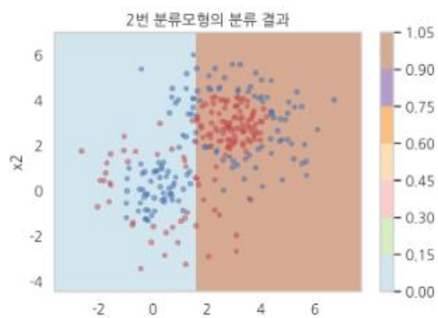


에이다부스트(m=20) 분류 결과
6
4
2
0
-2
-4
x2
-2
0
2
4
6
x1
1.05
0.90
0.75
0.60
0.45
0.30
0.15
0.00

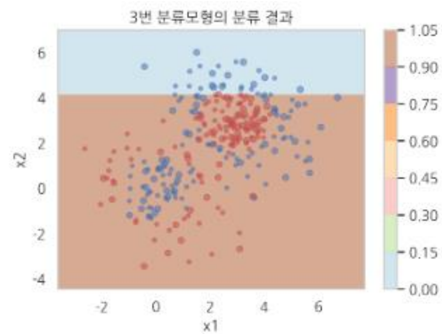
AdaBoost

k_1

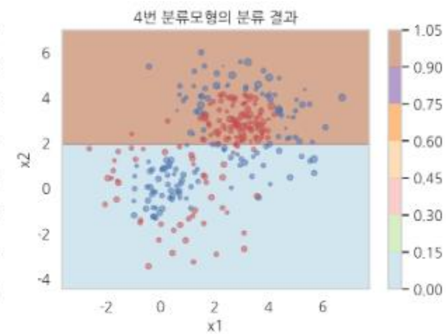
k_2



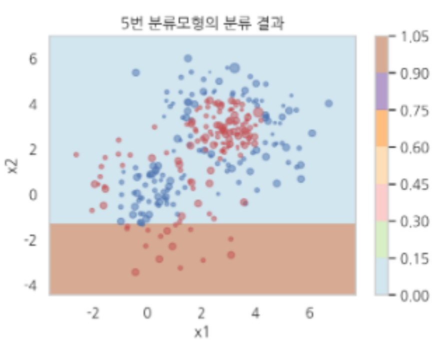
k_3



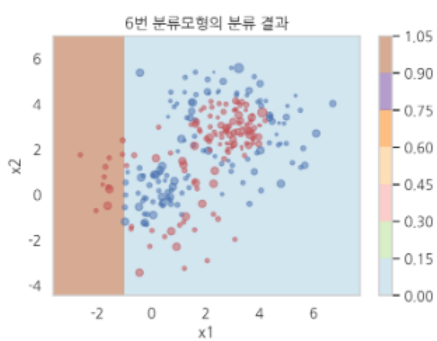
k_4



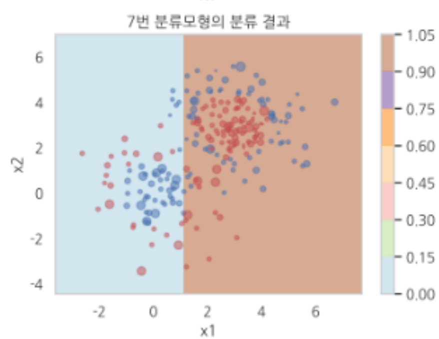
k_5



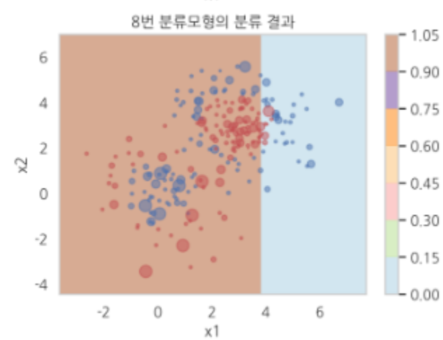
k_6



k_7



k_8



Gradient Boost

- Gradient classifier

$$C_m(x_i) = \mathrm{sign}\big(C_{m-1}(x_i) - \alpha_m k_m(x_i)\big)$$

- the m th classifier

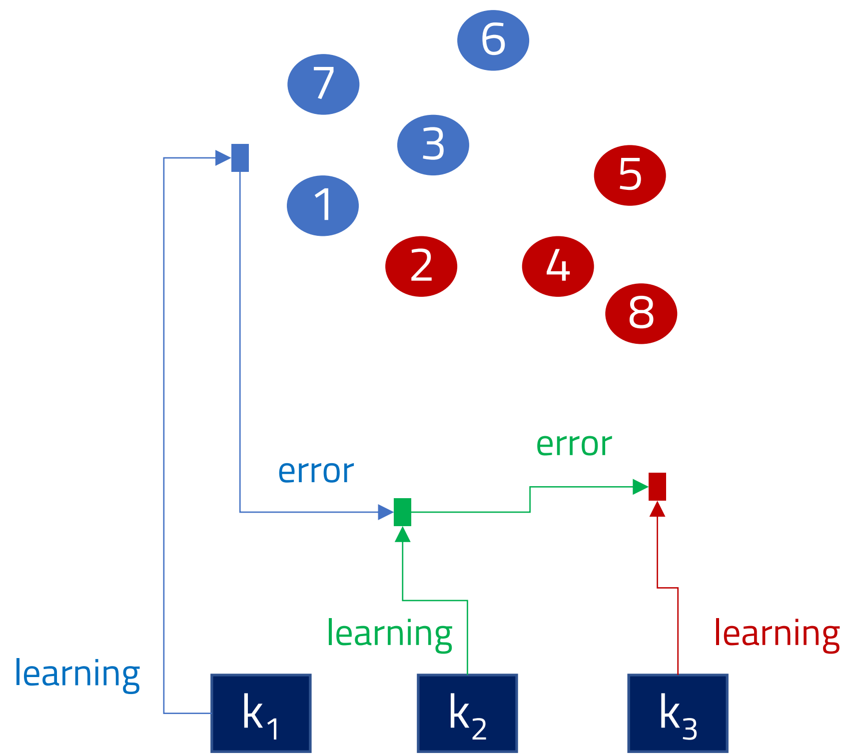
$$k_m = -\frac{\delta L(y, C_{m-1})}{\delta C_{m-1}}$$ calculus of variations

so that $C_m = C_{m-1} - \alpha_m \frac{\delta L(y,C_{m-1})}{\delta C_{m-1}}$

ex) $L(y, C_{m-1}) = \sum_{i=1}^{N}\big(y_i - C_{m-1}(x_i)\big)^2$

$$\frac{\delta L(y, C_{m-1})}{\delta C_{m-1}} = \frac{\partial L(y, C_{m-1})}{\partial C_{m-1}} = \sum_{i=1}^{N}\big(y_i - C_{m-1}(x_i)\big)$$

Data
- i=1,2,3,4,5,6,7, 8
- N=8



Classifier
- m=1,2,3

Gradient Boost

- Gradient classifier

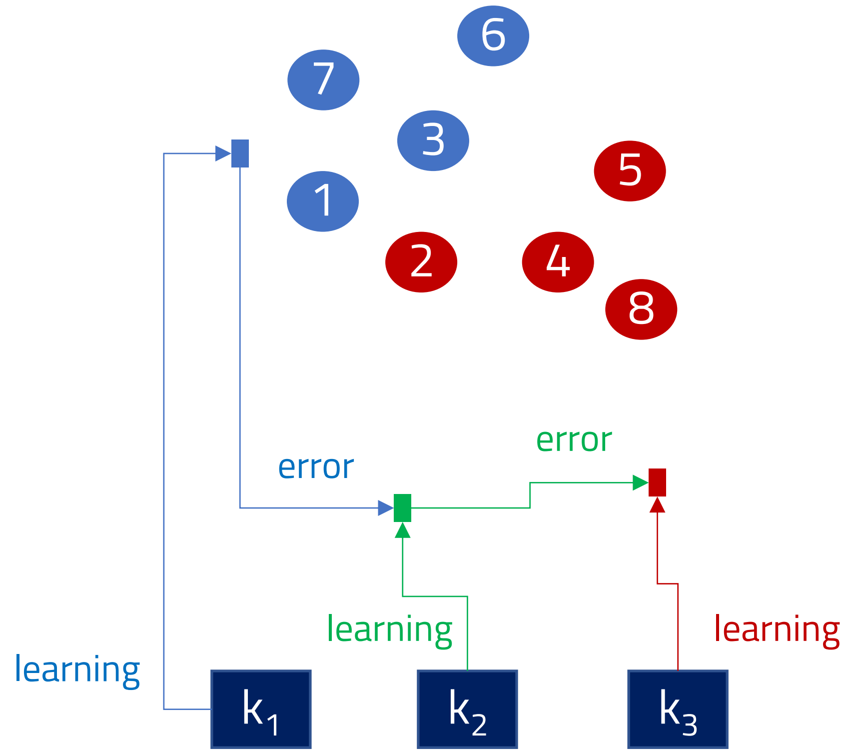
$$C_m(x_i) = \mathrm{sign}\big(C_{m-1}(x_i) - \alpha_m k_m(x_i)\big)$$

where

$$\alpha_m = \underset{\alpha}{\mathrm{Argmin}}\, L(y, C_m)$$

Data

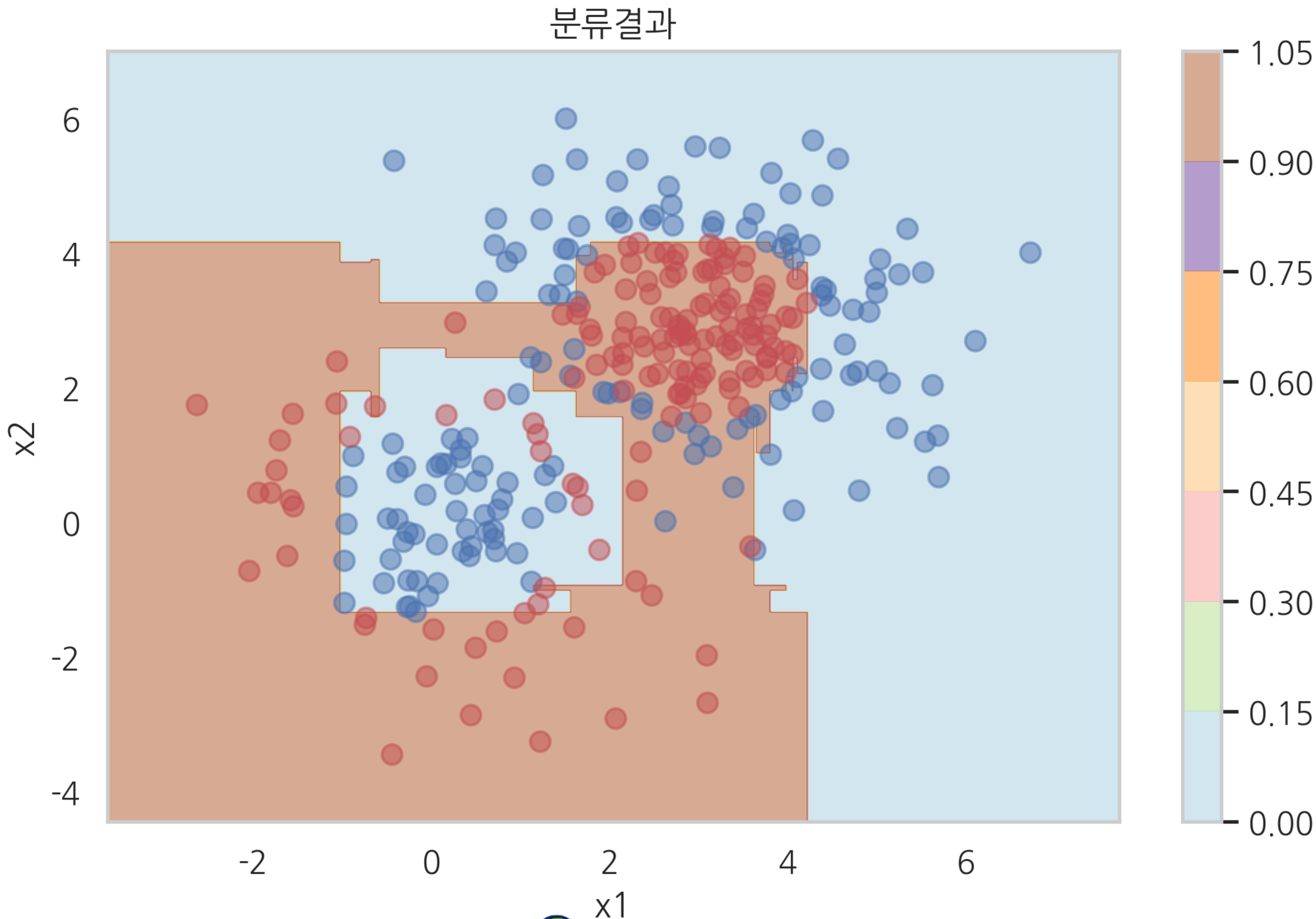
- i=1,2,3,4,5,6,7,8
- N=8



Classifier

- m=1,2,3

Gradient Boost



Boosting

Adaboost

```
from sklearn.ensemble import AdaBoostClassifier
model1 = AdaBoostClassifier(DecisionTreeClassifier(max_depth=1), n_estimators=100)
```

Gradient boosting

```
from sklearn.ensemble import GradientBoostingClassifier
model = GradientBoostingClassifier(n_estimators=100, max_depth=2)
```