Pattern Recognition

Yukyung Choi

yk.choi@rcv.sejong.ac.kr

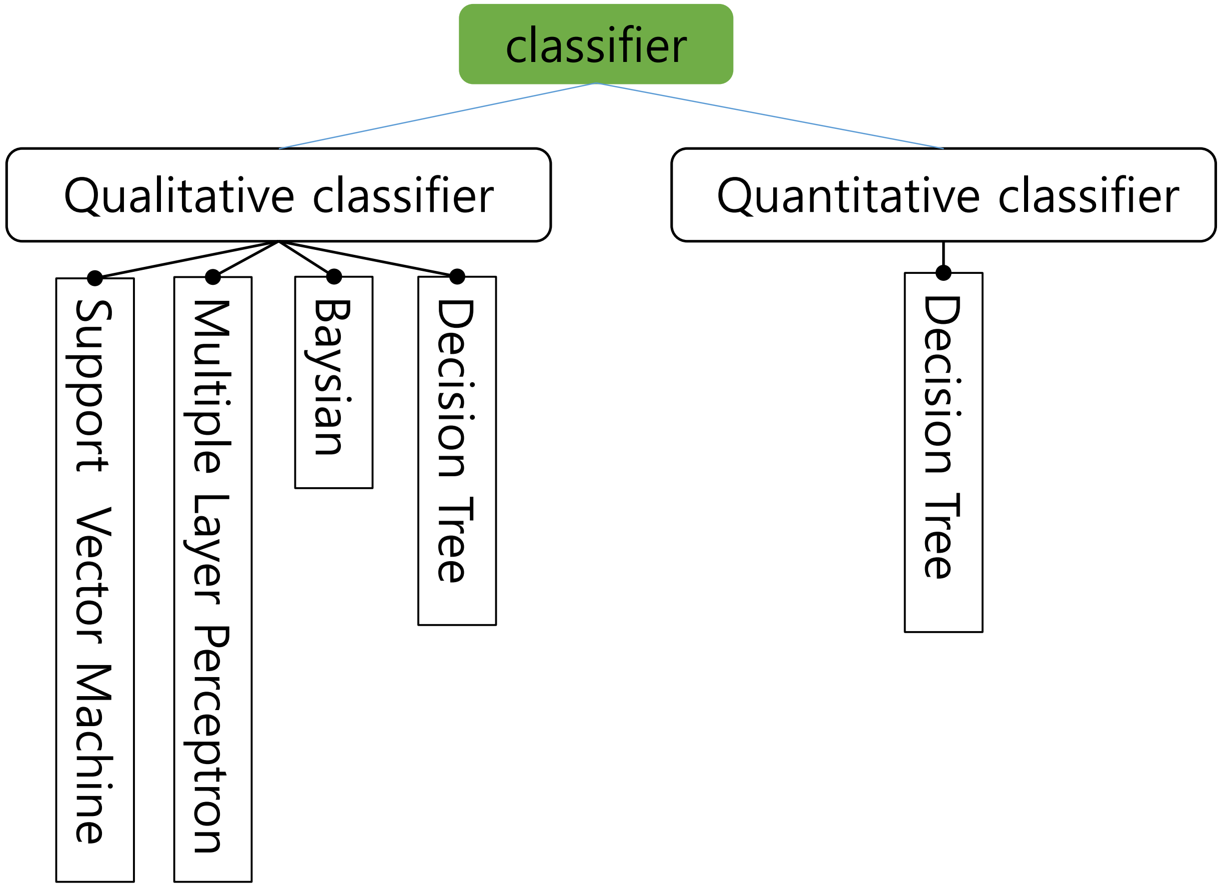
Summary

- Decision Tree & Random Forests
    - Theory
    - Hand on labs: How to use DT & RF (Lv.0)
        - DT vs RF
    - Hand on labs: Data Classification using library (Lv.1)
        - DIGIT Classification (Lv.1)
        - IRIS Classification (Lv.1)
    - Hand on labs: Data Regression using library
        - Salary Prediction (Lv.1)
    - Hand on labs : Bike Sharing Demand (Lv.2)
    - Hand on Labs: Implementation of DT-RF (Lv.3)

Content

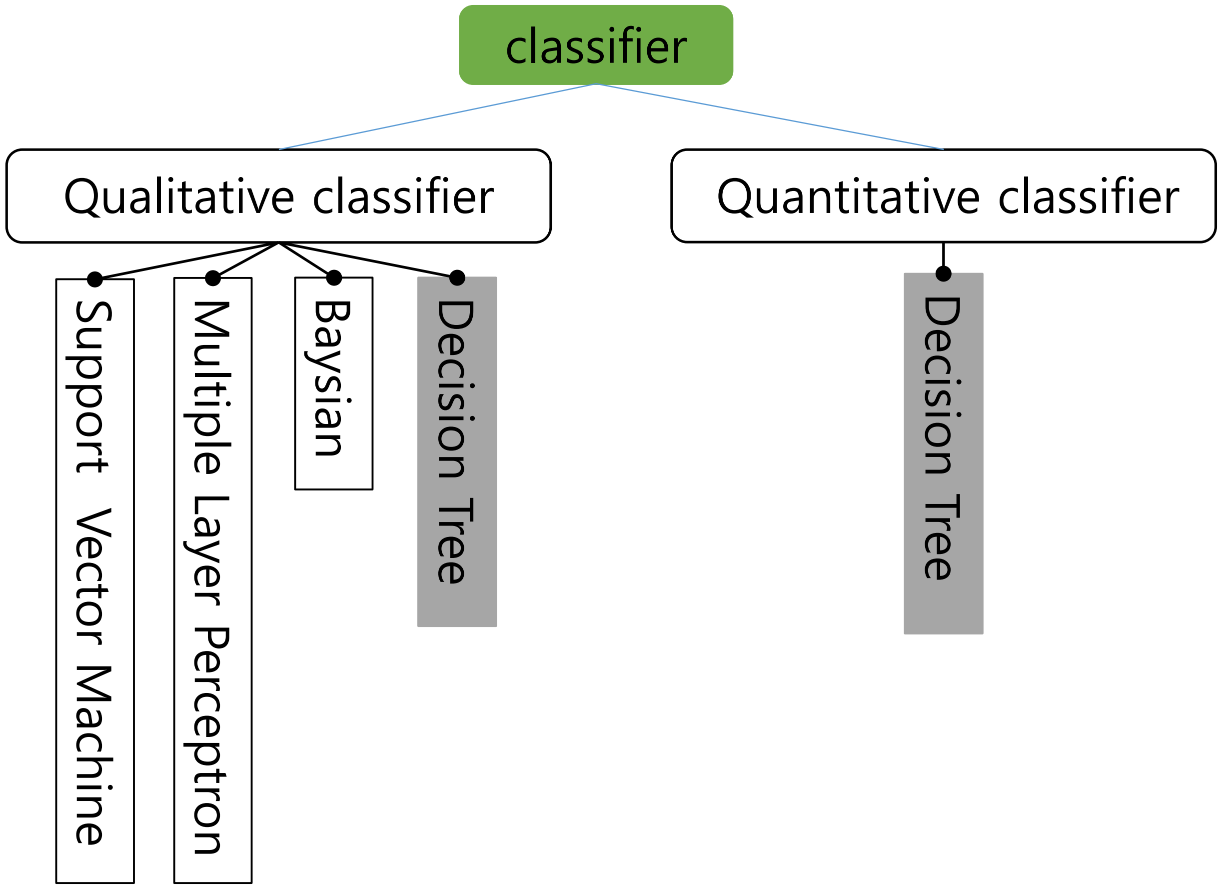
- Decision Tree
- Random Decision Trees
- Random Forests
- Random Forests with discriminative decision tree

Classifier



Metric/Non-Metric data : 어떤 사물이나 개념을 양적/질적 으로 표시한 데이터

Classifier



Metric/Non-Metric data : 어떤 사물이나 개념을 양적/질적 으로 표시한 데이터

Decision Tree

Definition

Tree structure based Classifier

Classification method by using several rules and constrains.

tree [1] : a tree is a widely-used data structure that emulates a hierarchical tree structure with a set of linked nodes.

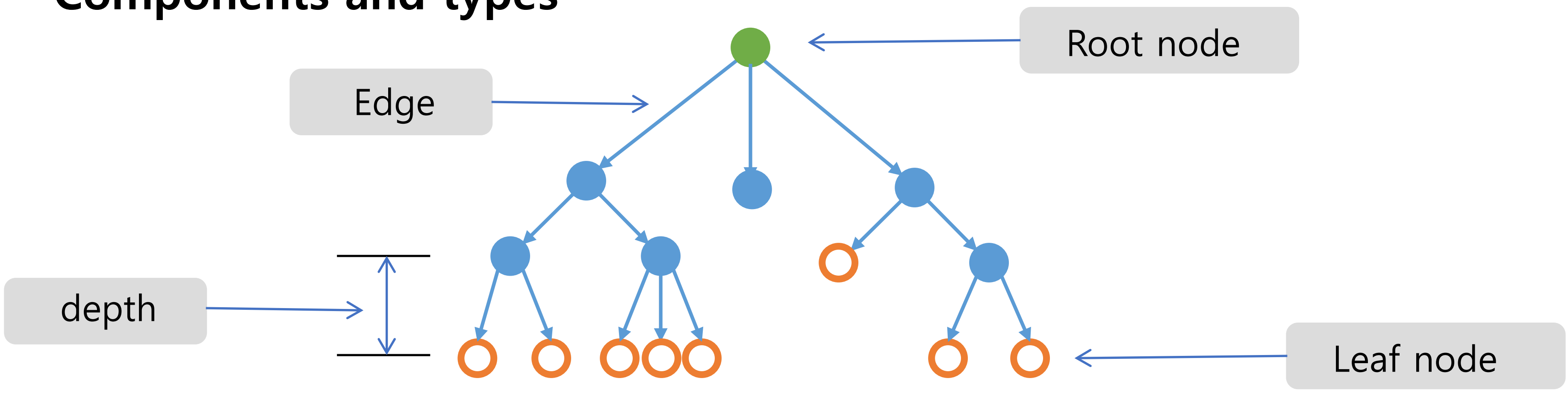
Decision Tree

Definition

Tree structure based Classifier

Classification method by using several rules and constrains.

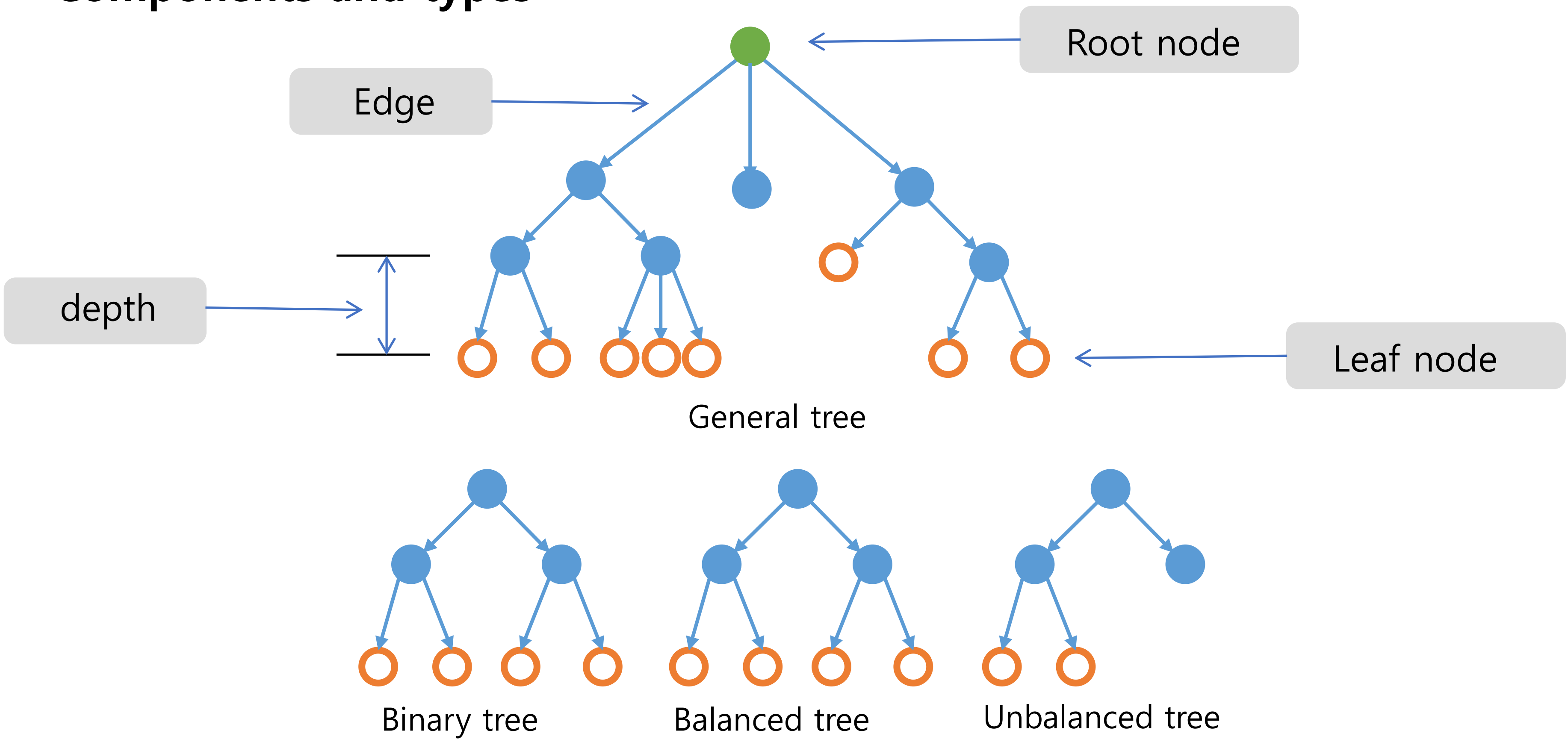
Components and types



General tree

Decision Tree

Components and types



Random Forest

- Advantage
  - 성능이 우수함
  - 파라미터 튜닝이 적음
  - 데이터 스케일링 없이도 동작
  - 큰 데이터 셋에서 잘 동작함
  - 시각화를 통한 모델의 이해
  - Missing 데이터에 강함

- Disadvantage
  - 매우 차원이 높은 희소한 데이터에서 잘 동작하지 않음
    - 이런 데이터는 선형 모델이 더욱 적합
  - Noisy한 데이터에 취약

Decision Tree

Training

build the tree

Consideration in this step

1) How many splits should there be at each node?
2) How to select the query attribute?
3) When to stop growing the tree?
4) Which class to allow leaf nodes?

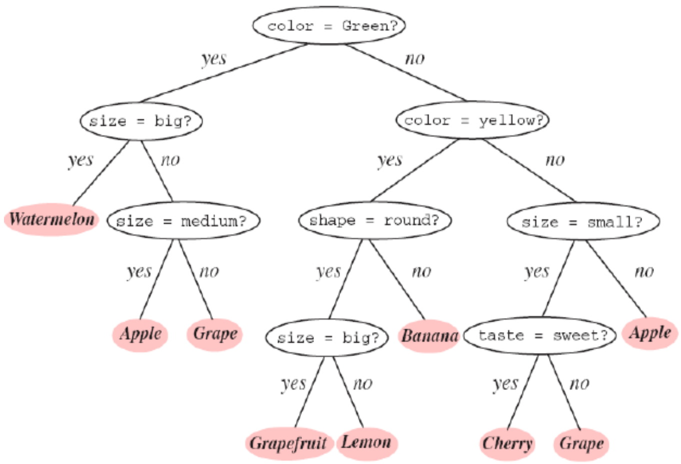
Decision Tree

Training

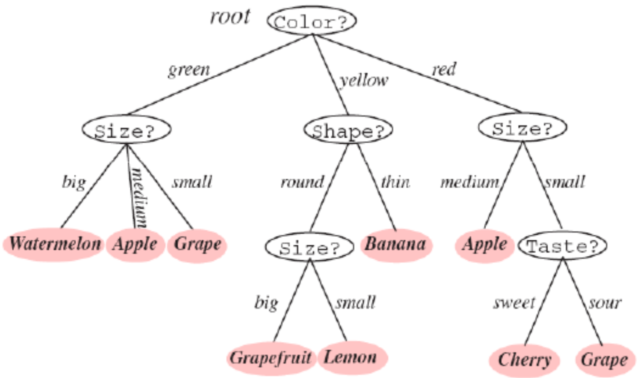
build the tree

Consideration in this step

1) How many splits should there be at each node?



Multi-valued tree



Binary tree

Decision Tree

Training

build the tree

Consideration in this step

2) How to select the query attribute?

Ready to Candidate query attributes :
Collect all possible attributes

Decision Tree

Training

build the tree

Consideration in this step

2) How to select the query attribute?

Ready to Candidate query attributes :
Collect all possible attributes

Select the best query attributes :
Find the maximal decrease in *impurity(entropy)*

Decision Tree

Training

build the tree

Consideration in this step

2) How to select the query attribute?

Ready to Candidate query attributes :
Collect all possible attributes

Select the best query attributes :
Find the maximal decrease in *impurity(entropy)*

$$\Delta im(T) = im(T) - \frac{|X_{Tleft}|}{|X_T|} im(T_{left}) - \frac{|X_{Tright}|}{|X_T|} im(T_{right})$$

$$\Delta im(T) = \frac{|X_{Tleft}|}{|X_T|} \frac{|X_{Tright}|}{|X_T|} \left(\sum_{i=1}^{M} | p(w_i | T_{left}) - p(w_i | T_{right}) | \right)^2$$

Decision Tree

Training

build the tree

Consideration in this step

2) How to select the query attribute?

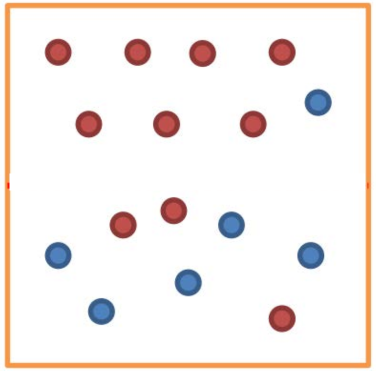
Impurity functions

Entropy, Gini, Misclassification

Entropy	$im(T) = 1 - \sum_{i=1}^{M} p(w_i \mid T)^2 = \sum_{i \neq j} p(w_i \mid T) p(w_j \mid T)$
Gini	$im(T) = -\sum_{i=1}^{M} p(w_i \mid T) \log_2 p(w_i \mid T)$
Misclassification	$im(T) = 1 - \max_{i} p(w_i \mid T)$

Decision Tree

Entropy Example



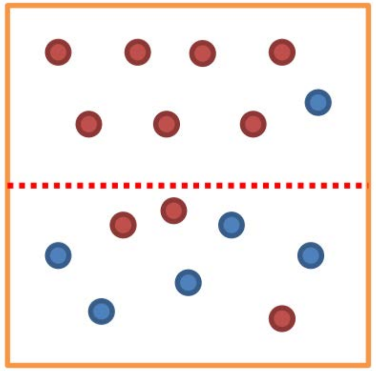
1st step

Log: scale normalization effect

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

$$Entropy(A) = -\frac{10}{16}\log_2\left(\frac{10}{16}\right) - \frac{6}{16}\log_2\left(\frac{6}{16}\right) \approx 0.95$$

RED: 10, BLUE: 6



2nd step

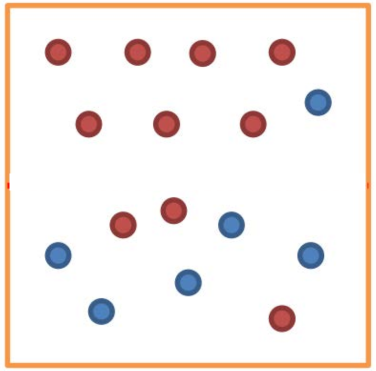
$$Entropy(A) = \sum_{i=1}^{d} R_i \left(-\sum_{k=1}^{m} p_k \log_2(p_k)\right)$$

R: split region, d: # region, m: # class

$$Entropy(A) = 0.5 \times \left(-\frac{7}{8}\log_2\left(\frac{7}{8}\right) - \frac{1}{8}\log_2\left(\frac{1}{8}\right)\right) + 0.5 \times \left(-\frac{3}{8}\log_2\left(\frac{3}{8}\right) - \frac{5}{8}\log_2\left(\frac{5}{8}\right)\right) \approx 0.75$$

Decision Tree

GINI Example

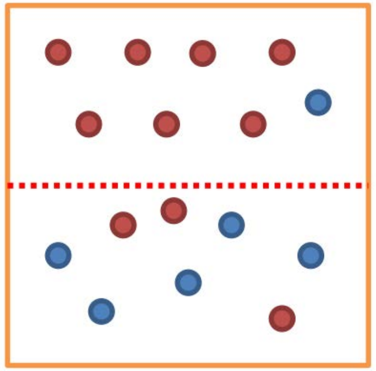


1st step

$$\text{Gini}(A) = \left(1 - \sum_{k=1}^{m} p_{ik}^2\right)$$

Gini(A) = (1-((10/16)^2 + (6/16)^2)

RED: 10, BLUE: 6



2nd step

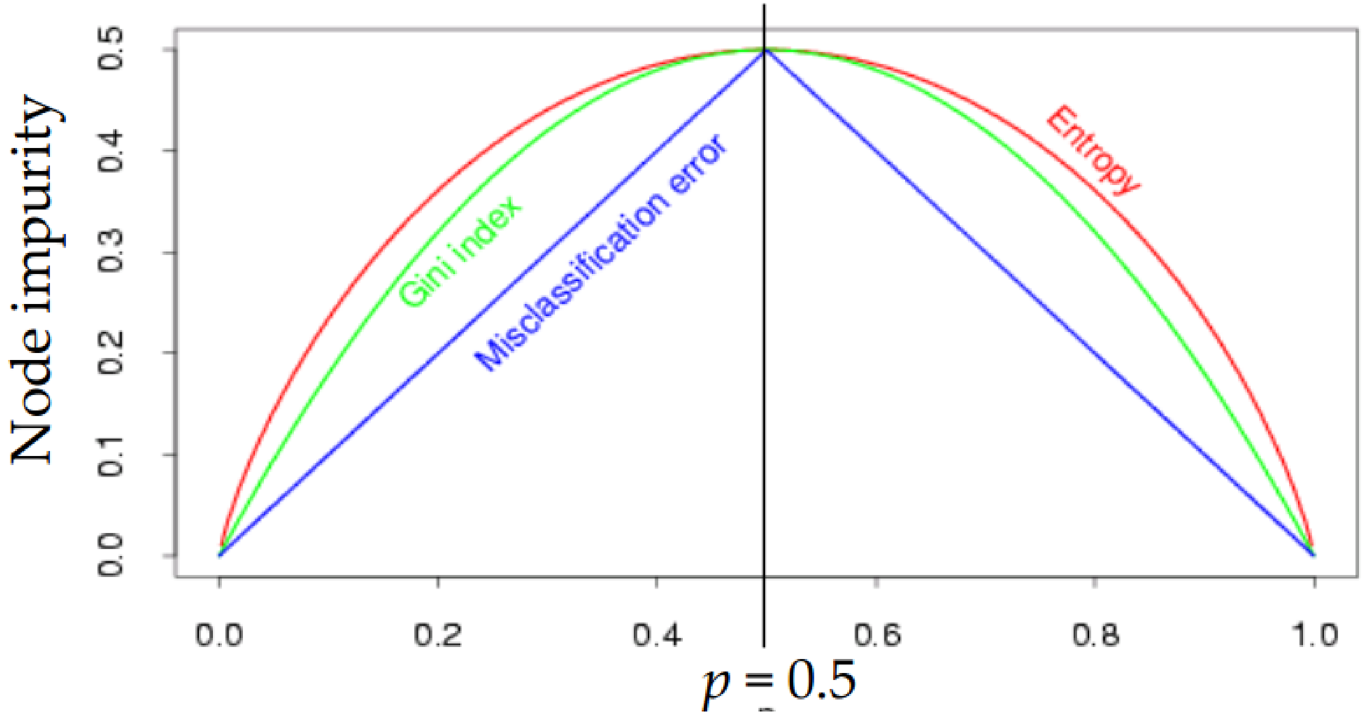
$$G.I(A) = \sum_{i=1}^{d}\left(R_i\left(1 - \sum_{k=1}^{m} p_{ik}^2\right)\right)$$

R: split region, d: # region, m: # class

Gini(A) = 0.5*(1- ((7/8)^2 + (1/8)^2)) + 0.5*(1 – ((5/8)^2 + (3/8)^2))

Decision **T**ree

Miss-classification is not differential.
Thus, this metric is not useful.



Decision Tree

Training

build the tree

Consideration in this step

3) When to stop growing the tree?

Stop conditions

$im(T) = 0$ → **Overfitting**

Decision Tree

Training

build the tree

Consideration in this step

3) When to stop growing the tree?

Stop conditions

$im(T) = 0$ → **Overfitting**

the number of X_t <= threshold → **Premature convergence**

argmax $\triangle im(T)$ <= threshold

Decision Tree

Training

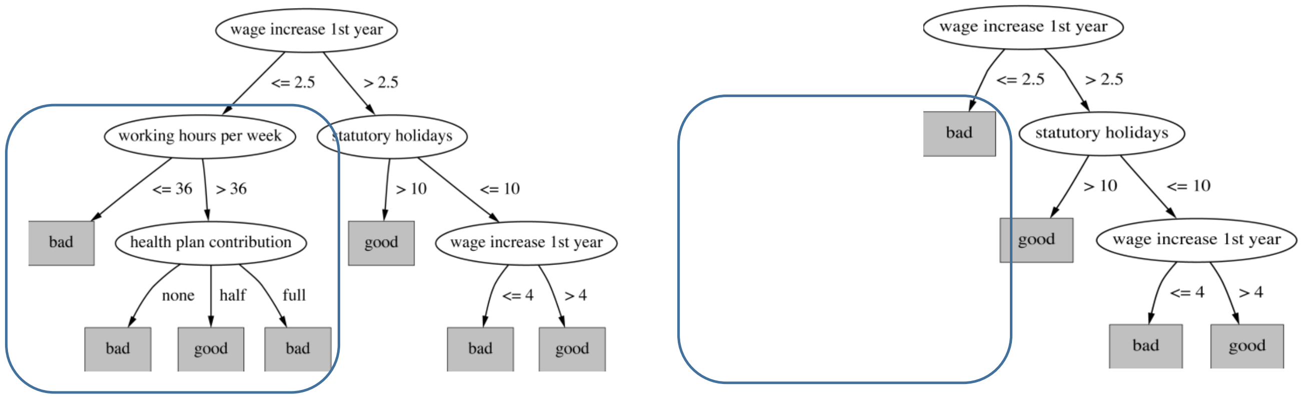
build the tree

Consideration in this step

3) When to stop growing the tree?

Avoiding overfitting & premature convergence

Make the largest tree and then do *pruning*



Example of pruning

Decision Tree

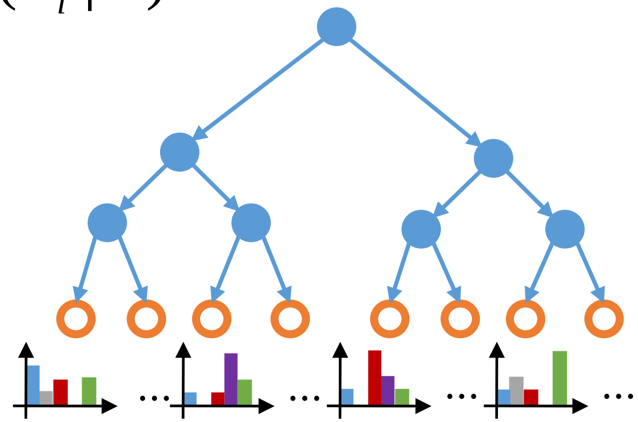
Training

build the tree

Consideration in this step

4) Which class to allocate leaf nodes?

$$K = \arg\max_{i} p(w_i \mid T)$$



Decision Tree

Classification

recognize the sample

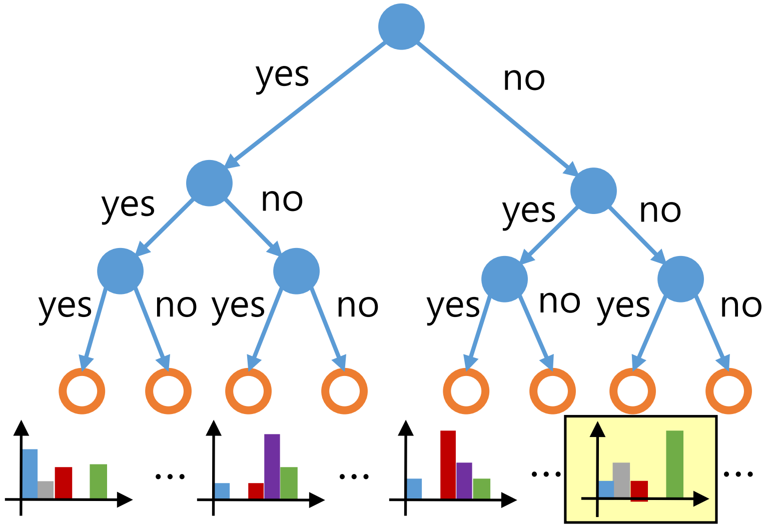
Decision Tree

Classification

recognize the sample

Advantages

Very fast "Yes/NO" operation
Just h-1 times comparison (h : depth level)



Example of classification

Decision Tree

Characteristics

+Handling the metric and non-metric data

weight
height
distance
cost

Metric data

gender
Blood type
job
Brand name

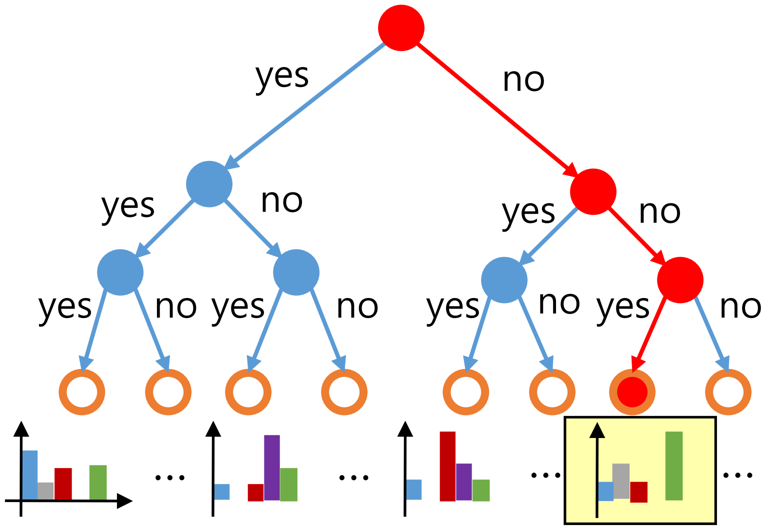
Non-Metric data

Decision Tree

Characteristics

+Handling the metric and non-metric data

+Visualization



Decision Tree

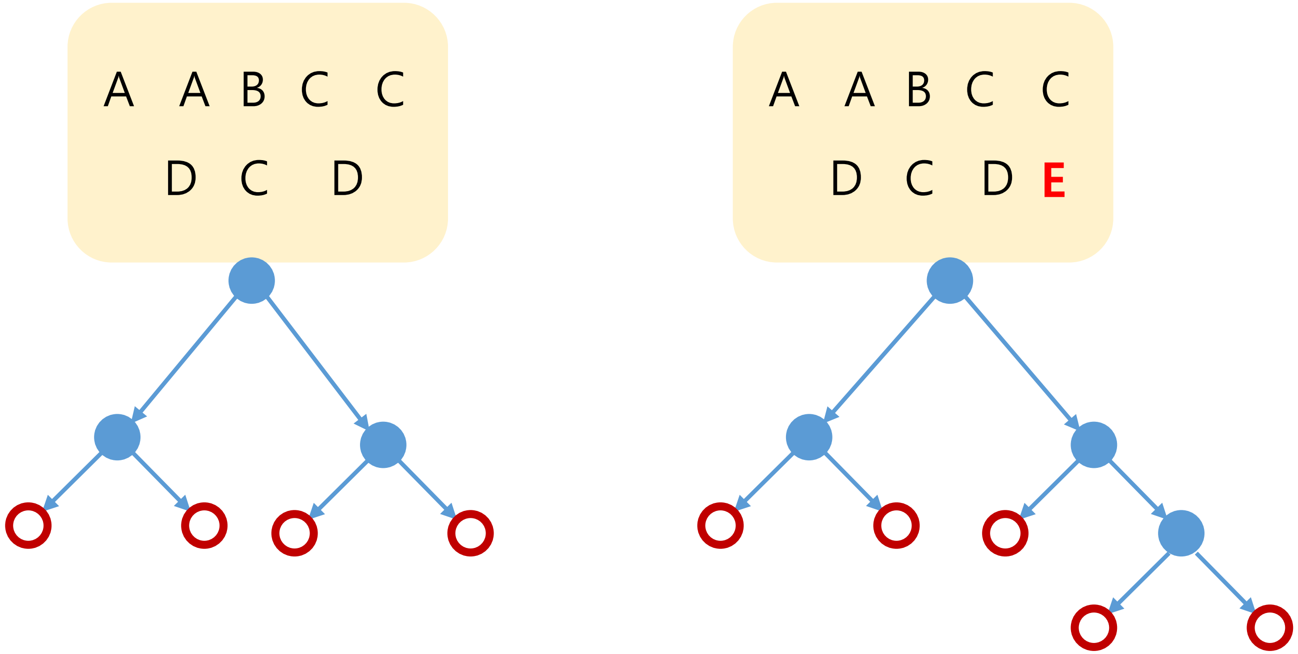
Characteristics

+Handling the metric and non-metric data
+Visualization
+Fast recognition

Decision Tree

Characteristics

+Handling the metric and non-metric data
+Visualization
+Fast recognition
-Instability(sensitive to noise data)



Decision Tree

Characteristics

+Handling the metric and non-metric data
+Visualization
+Fast recognition
-Instability
-Greedy algorithms (미리 정한 기준에 따라 매번 가장 좋아 보이는 답을 선택하는 알고리즘)

Decision **T**ree

Characteristics

+Handling the metric and non-metric data
+Visualization
+Fast recognition
-Instability
-Greedy algorithms
+Handling the missing data

Decision Tree

Algorithms

CART, ID3, C4.5

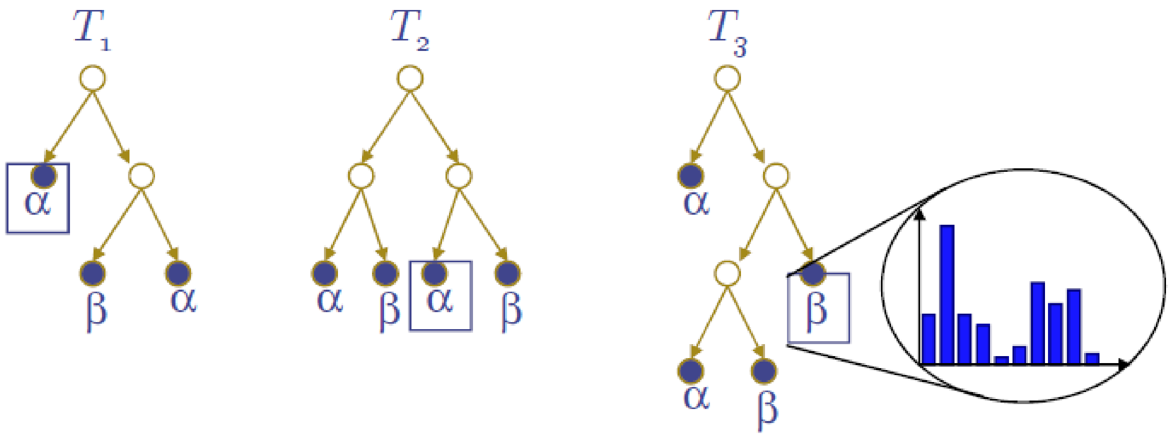
L.Breiman	R.Quinlan
CART	ID3
	↓ extension
	C4.5
	↓ commercialization
	C5.0

Comparisons

Property	CART	ID3	C4.5
Float data	O	X	O
Tree type	Binary	Multi	Multi
Prune	O	X	O
Classification	O	O	O
Regression	O	X	X
Missing data	Surrogate split	X	skip
Multi variable	O	X	X

Randomized Decision Trees

Randomized Decision Trees (Amit & German 1997)
Multiple classifier of several trees



Randomized decision trees

Randomized Decision Trees

Randomized Decision Trees (Amit & German 1997)
Multiple classifier of several trees

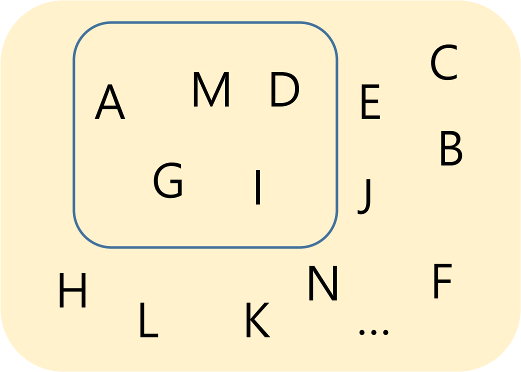
Idea : randomized attribute selection

Randomized Decision Trees

Randomized Decision Trees (Amit & German 1997)
Multiple classifier of several trees

Idea : randomized attribute selection

Instead randomly use subset of K attributes (K << All cases)
- Reduce correlation between different trees
- Typical choice : K = 10 for root node, K = 100*d (d : depth level)



All cases of possible attributes

Randomized Decision Trees

Randomized Decision Trees (Amit & German 1997)
Multiple classifier of several trees

Idea : randomized attribute selection

Instead randomly use subset of K attributes (K << All cases)
- Typical choice : K = 10 for root node, K = 100*d (d : depth level)
- Reduce correlation between different trees.

Choose best splitting attribute
- Minimizing entropy (impurity)

Randomized Forests

Randomized Forests (Breiman 2001)
Multiple classifier of several trees

Randomized Forests

Randomized Forests (Breiman 2001)
Multiple classifier of several trees

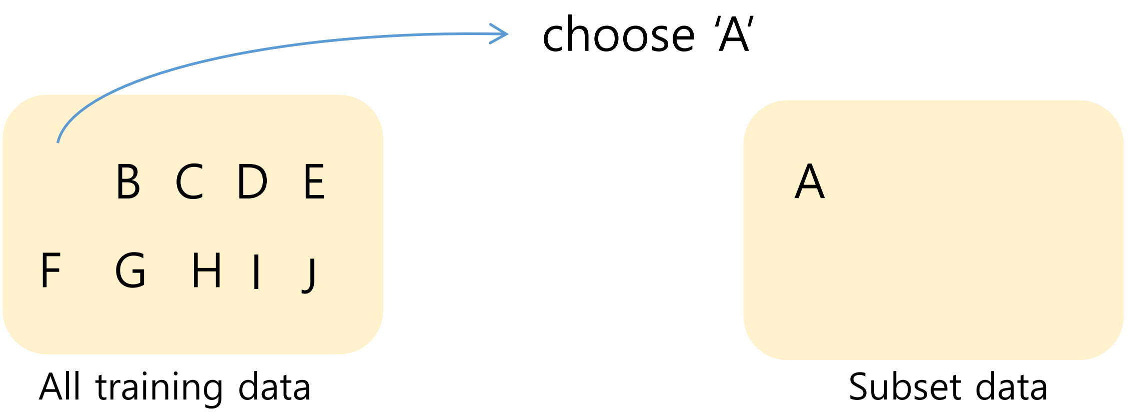
Idea : Randomness = 1)Bootstrap sampling + 2)randomized attribute selection

Randomized Forests

Randomized Forests (Breiman 2001)
Multiple classifier of several trees

Idea : Randomness = [1]Bootstrap sampling + [2]randomized attribute selection

Bootstrap sampling
Select a subset by choosing N times with replacement from all training data.

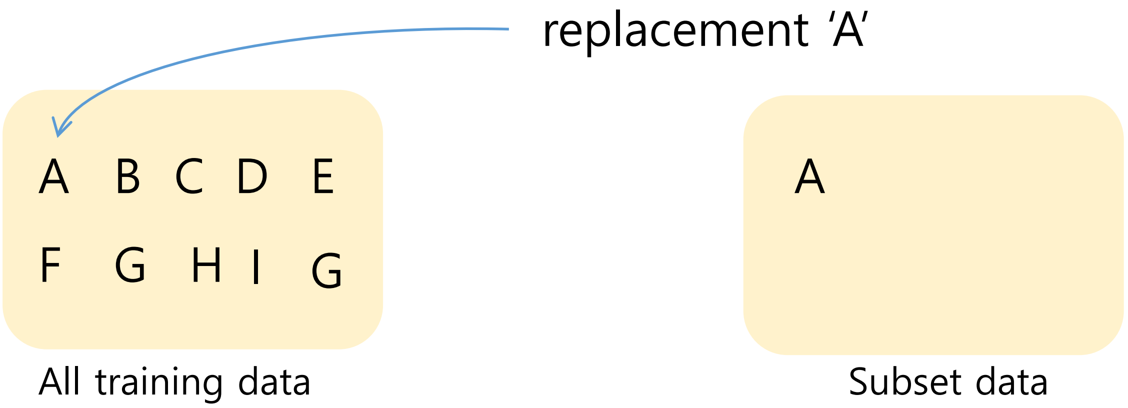


Randomized Forests

Randomized Forests (Breiman 2001)
Multiple classifier of several trees

Idea : Randomness = [1]Bootstrap sampling + [2]randomized attribute selection

Bootstrap sampling
Select a subset by choosing N times with replacement from all training data.



Randomized Forests

Randomized Forests (Breiman 2001)
Multiple classifier of several trees

Idea : Randomness = [1]Bootstrap sampling + [2]randomized attribute selection

Bootstrap sampling
Select a subset by choosing N times with replacement from all training data.

Randomized attribute selection
Instead randomly use subset of K attributes
- Typical choice : K = $\sqrt{N}$ **(** N : the number of subset)

Randomized Forests

Randomized Forests (Breiman 2001)
Multiple classifier of several trees

Idea : Randomness = [1]Bootstrap sampling + [2]randomized attribute selection

Advantage
Resistant to Overfitting

Randomized Forests

Randomized Forests (Breiman 2001)
Multiple classifier of several trees

Idea : Randomness = [1]Bootstrap sampling + [2]randomized attribute selection

Advantage

Resistant to Overfitting
Well suited for large training data

Randomized Forests

Randomized Forests (Breiman 2001)
Multiple classifier of several trees

Idea : Randomness = [1]Bootstrap sampling + [2]randomized attribute selection

Advantage

Resistant to Overfitting
Well suited for large training data
Empirically very good results. (≥ SVM, ≥ Boosting)

Randomized Forests

Randomized Forests (Breiman 2001)
Multiple classifier of several trees

Idea : Randomness = [1]Bootstrap sampling + [2]randomized attribute selection

Advantage
Resistant to Overfitting
Well suited for problems with large training data
Empirically very good results. (≥ SVM, ≥ Boosting)

Disadvantage
Memory consumption

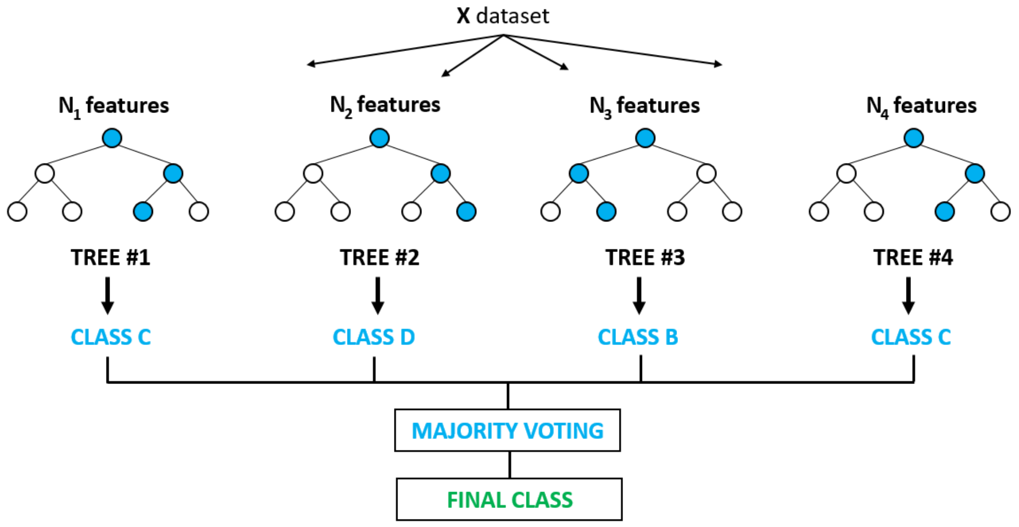
Comparison with various classifier

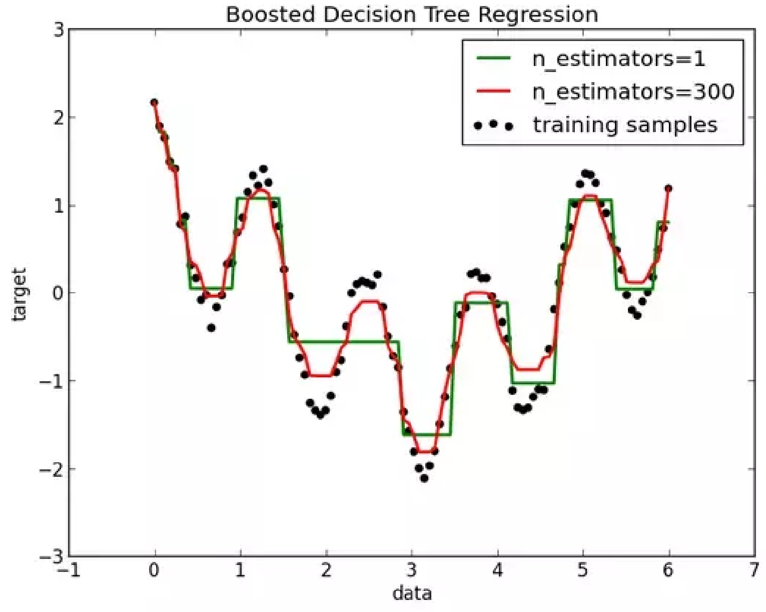
- CIFAR-10: Image Classification
- Comparison with various classifier
  - https://github.com/PhilippeCodes/Image-Classifier
  - https://github.com/PhilippeCodes/Image-Classifier/blob/master/Decision%20trees%20and%20random%20forests.ipynb

	Estimator	Test Accuracy
0	Baseline (dummy)	0.222
1	KNeighbors	0.776
2	DecisionTree	0.646
3	RandomForest	0.800
4	LogisticRegression	0.840
5	SVM Linear Kernel	0.817
6	SVM RBF Kernel	0.823
7	Multilayer Neural Network	0.821
8	Convolutional Neural Network	0.777

Classification & Regression with RF

- How to Build Tree
    - Classification: using entropy, information gain, Gini index
    - Regression: using MSE(mean square error)





Classification & Regression with RF

- How to decide final value
  - Classification: argmax class of **P**
  - Regression: average of **Y**

Classification & Regression with Tree

- How to Build Tree
  - Classification: using entropy, information gain, Gini index
  - Regression: using MSE(mean square error)

Examples

```
>>> from sklearn.ensemble import RandomForestClassifier
>>> from sklearn.datasets import make_classification
```

```
>>> X, y = make_classification(n_samples=1000, n_features=4,
...                            n_informative=2, n_redundant=0,
...                            random_state=0, shuffle=False)
>>> clf = RandomForestClassifier(n_estimators=100, max_depth=2,
...                              random_state=0)
>>> clf.fit(X, y)
RandomForestClassifier(bootstrap=True, class_weight=None, criterion='gini',
            max_depth=2, max_features='auto', max_leaf_nodes=None,
            min_impurity_decrease=0.0, min_impurity_split=None,
            min_samples_leaf=1, min_samples_split=2,
            min_weight_fraction_leaf=0.0, n_estimators=100, n_jobs=None,
            oob_score=False, random_state=0, verbose=0, warm_start=False)
>>> print(clf.feature_importances_)
[0.14205973 0.76664038 0.0282433  0.06305659]
>>> print(clf.predict([[0, 0, 0, 0]]))
[1]
```

https://scikit-learn.org/stable/modules/generated/sklearn.ensemble.RandomForestClassifier.html#sklearn.ensemble.RandomForestClassifier

Examples

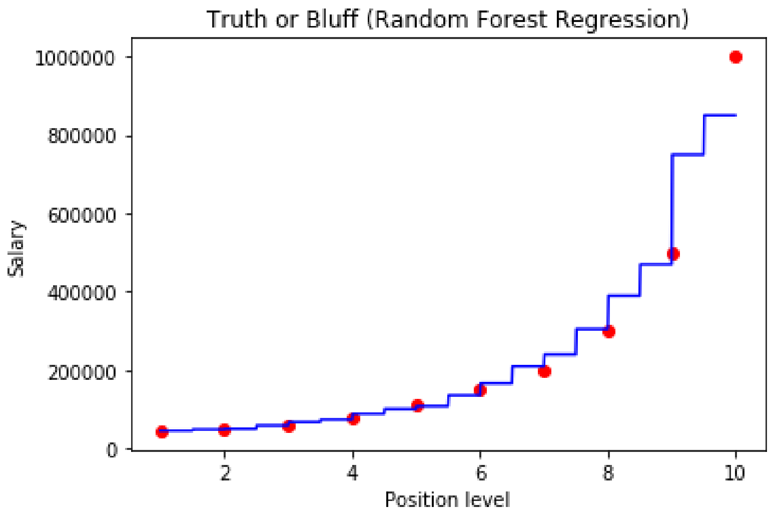
```
>>> from sklearn.ensemble import RandomForestRegressor
>>> from sklearn.datasets import make_regression
```

```
>>> X, y = make_regression(n_features=4, n_informative=2,
...                        random_state=0, shuffle=False)
>>> regr = RandomForestRegressor(max_depth=2, random_state=0,
...                              n_estimators=100)
>>> regr.fit(X, y)
RandomForestRegressor(bootstrap=True, criterion='mse', max_depth=2,
           max_features='auto', max_leaf_nodes=None,
           min_impurity_decrease=0.0, min_impurity_split=None,
           min_samples_leaf=1, min_samples_split=2,
           min_weight_fraction_leaf=0.0, n_estimators=100, n_jobs=None,
           oob_score=False, random_state=0, verbose=0, warm_start=False)
>>> print(regr.feature_importances_)
[0.18146984 0.81473937 0.00145312 0.00233767]
>>> print(regr.predict([[0, 0, 0, 0]]))
[-8.32987858]
```

https://scikit-learn.org/stable/modules/generated/sklearn.ensemble.RandomForestRegressor.html

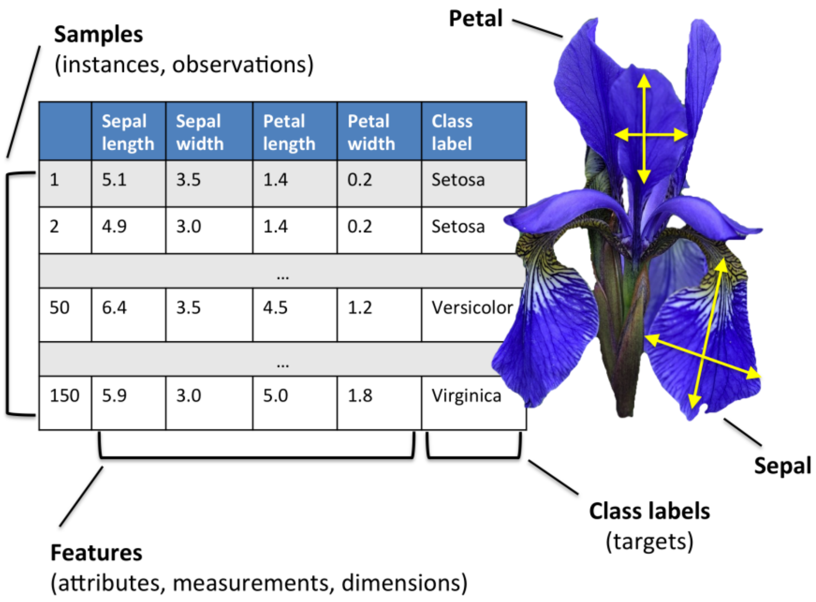
Application (Lv.1)

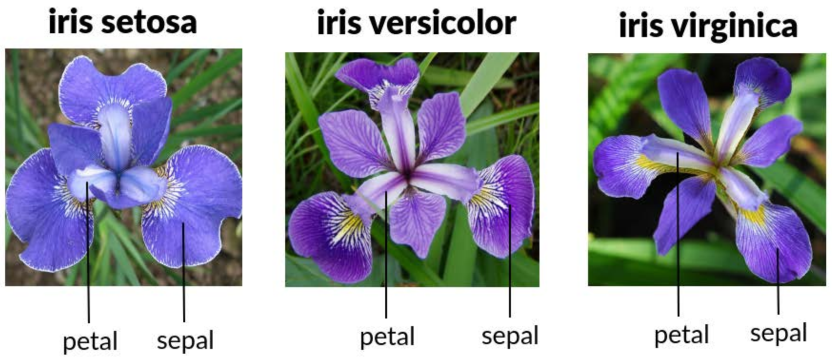
- Position Salaries
- Problem
  - https://www.kaggle.com/akram24/position-salaries
- Solution: Random Forests - regression
  - https://colab.research.google.com/drive/1oSEU7znIkwsfYuIHY6CzURI8pVYVxDmv

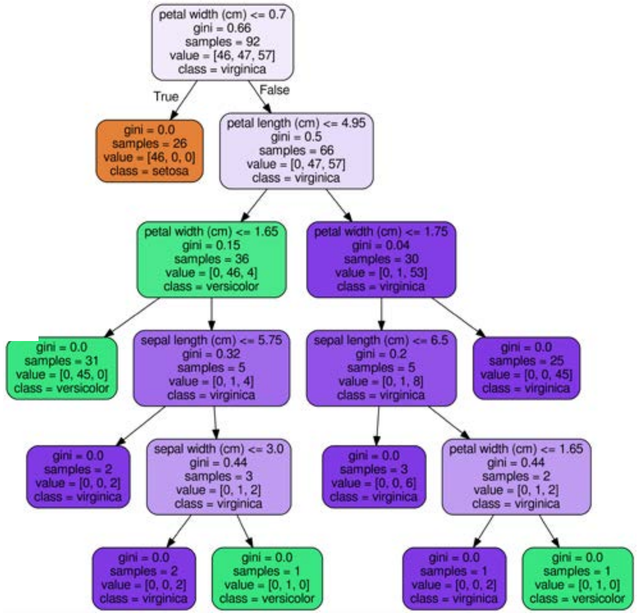


Application (Lv.1)

- Iris classification
- Solution: Random Forests - classification
  - https://colab.research.google.com/drive/1NXMRygCxXob0TCY5lJa8SltpFQjCOz5_

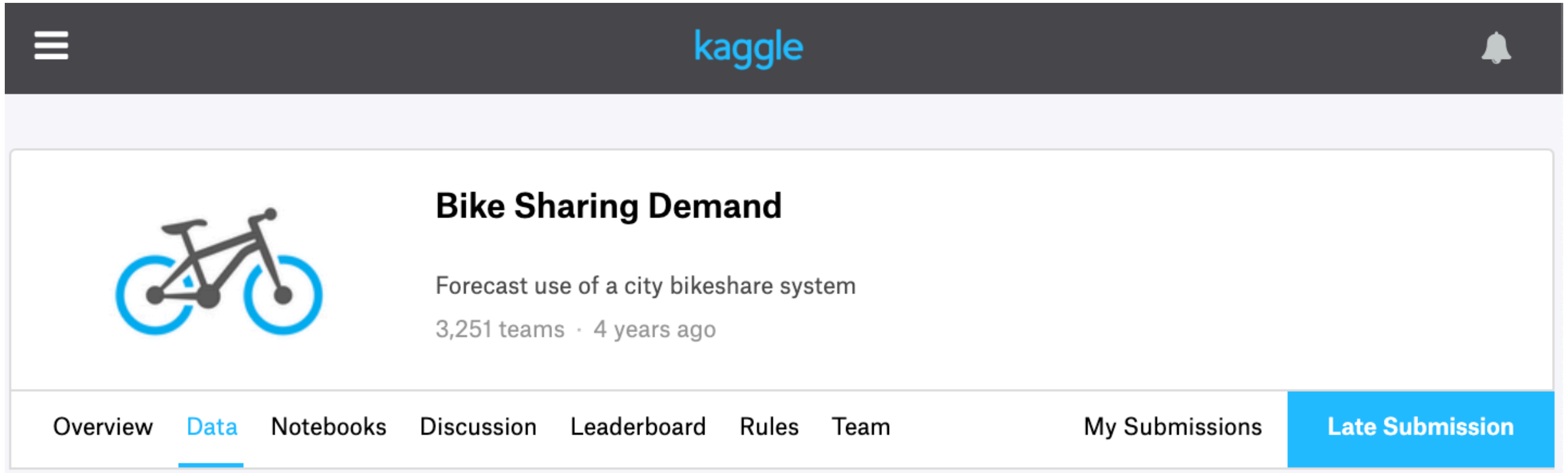






Application (Lv.2)

- Bike Sharing Demand
- Problem
  - https://www.kaggle.com/c/bike-sharing-demand
- Solution: Random Forests
  - https://colab.research.google.com/drive/1tz_SfvfXUkh6jd8AeGCx9LWFplXkBbiZ



Hands on Lab (Lv.2)

- Decision Tree from scratch
- Random Forests from scratch