Data Mining - Classification II

Nhat-Quang Doan

University of Science and Technology of Hanoi

nq.doan@gmail.com

Introduction

Ensemble models

- Random Forest
- Single Linear Regression models
- etc.

What are ensemble models?

- Ensemble models consist of various learning models to obtain better predictive performance
- Models can be a similar type or different

Ensemble models

Why ensemble models?

- Build different classifiers (experts) and let them vote
- Intuitively, ensemble of classifiers must be more accurate than any of its individual ones

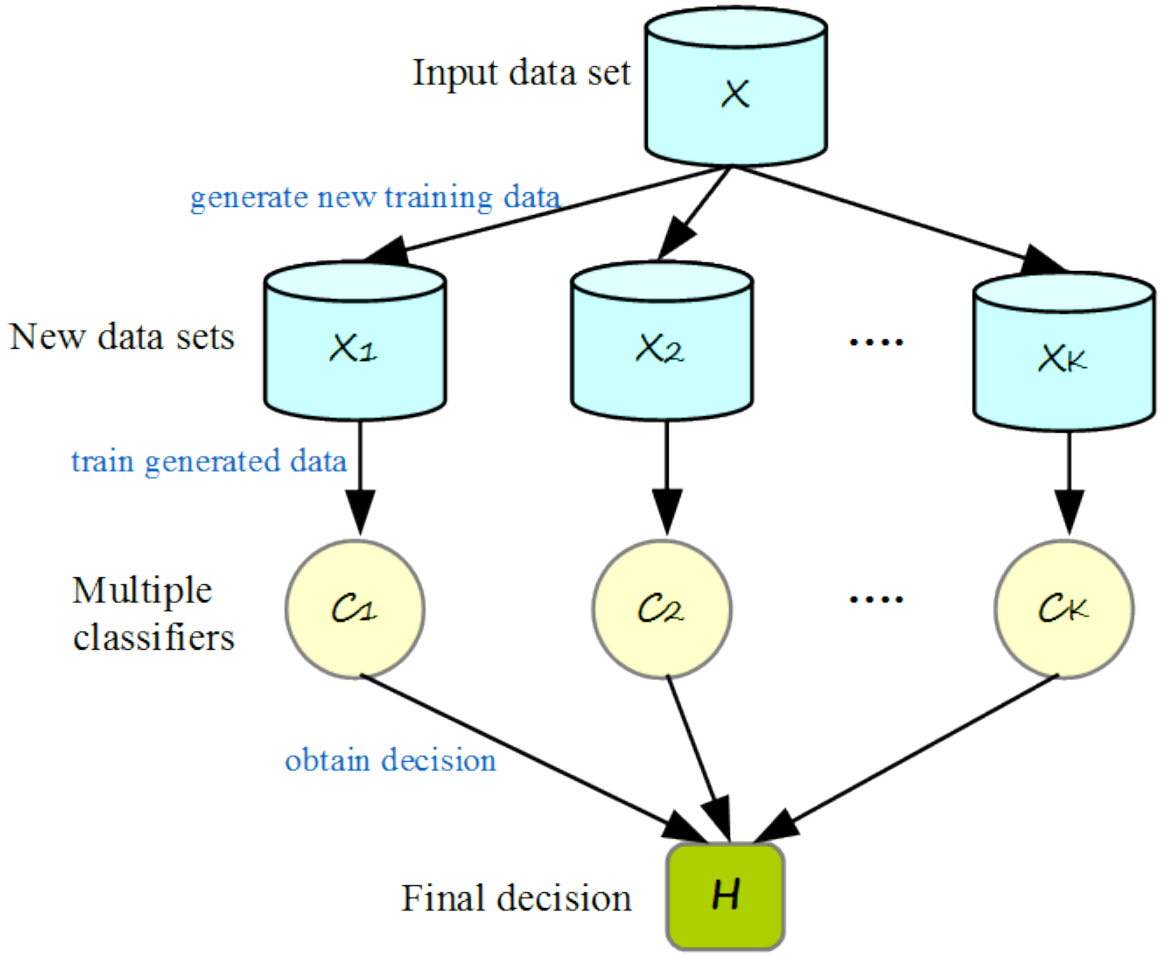
Ensemble models

How does it work?

- Suppose that there are 25 classifiers
- Each classifier has error rate $e = 0.35$
- Assume independence among classifiers
- Probability that the ensemble classifiers make a wrong prediction

$$e_{ensemble} = \sum_{i=13}^{25} \binom{25}{i} e^i (1-e)^{25-i} = 0.06$$

Ensemble models



Introduction

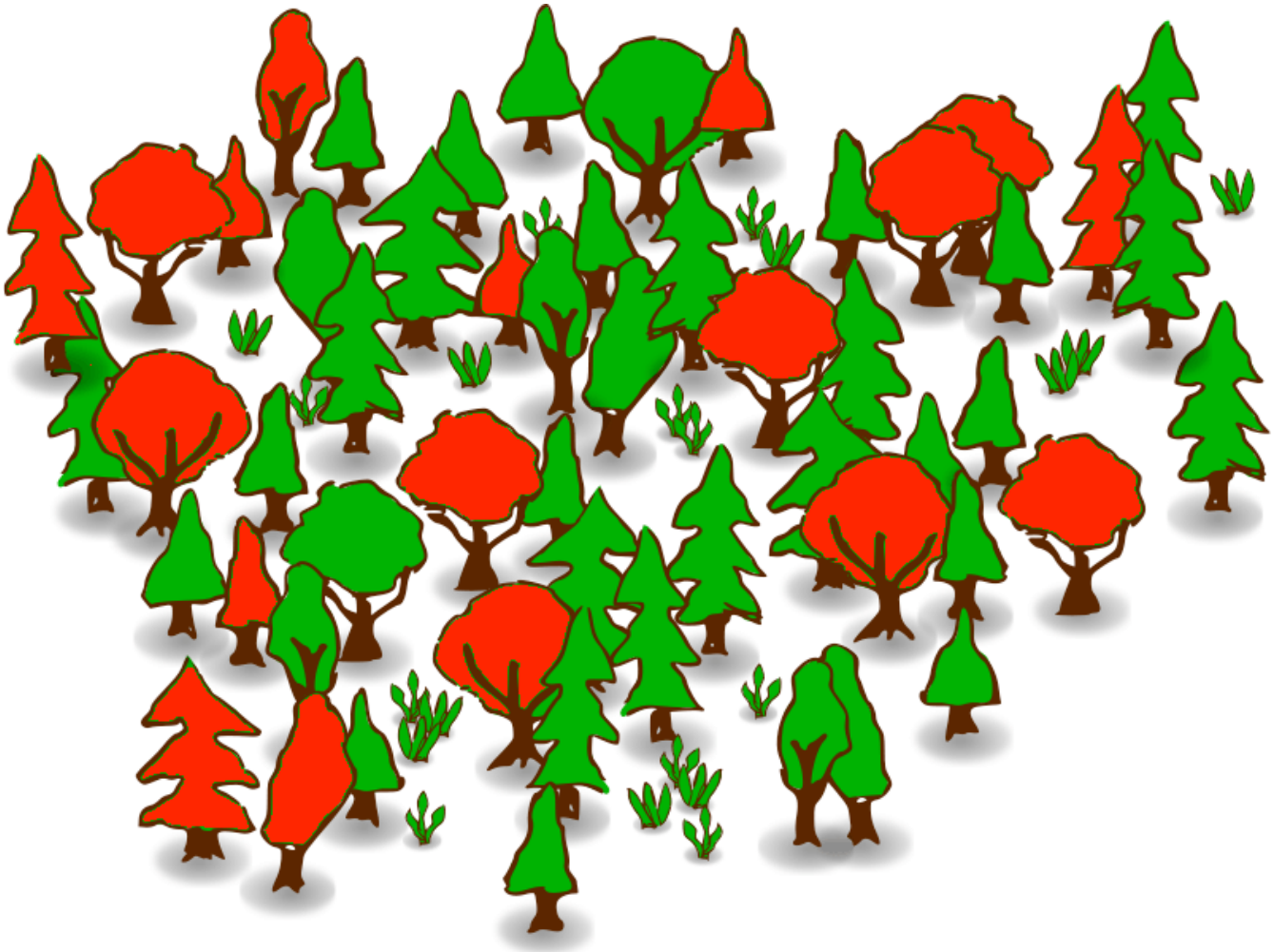
Pros

- Improve the predictive performance (the law of large numbers).
- Avoid the overfitting problem.
- Robust to outliers and noise.

Cons

- Not a compact presentation.
- Not easy to compute the complexity.

Principles

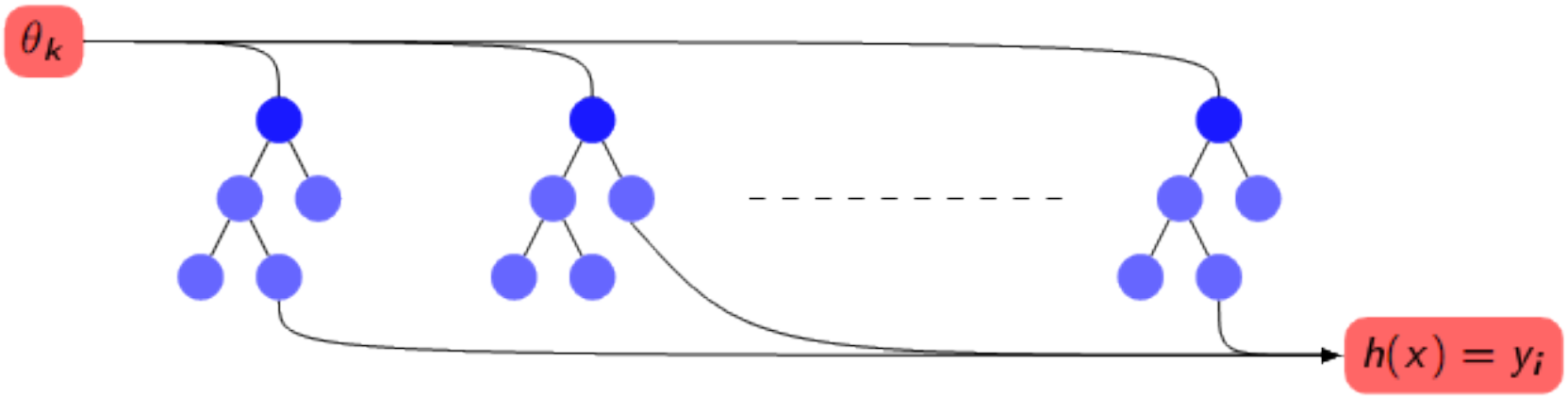


Random Forests

Random Forests

Principles

- Random Forests consist of an ensemble of **decision trees** such that each tree represents a classifier.
- The generalization error of a forest depends on the strength of the individual trees in the forest and the correlation between them.



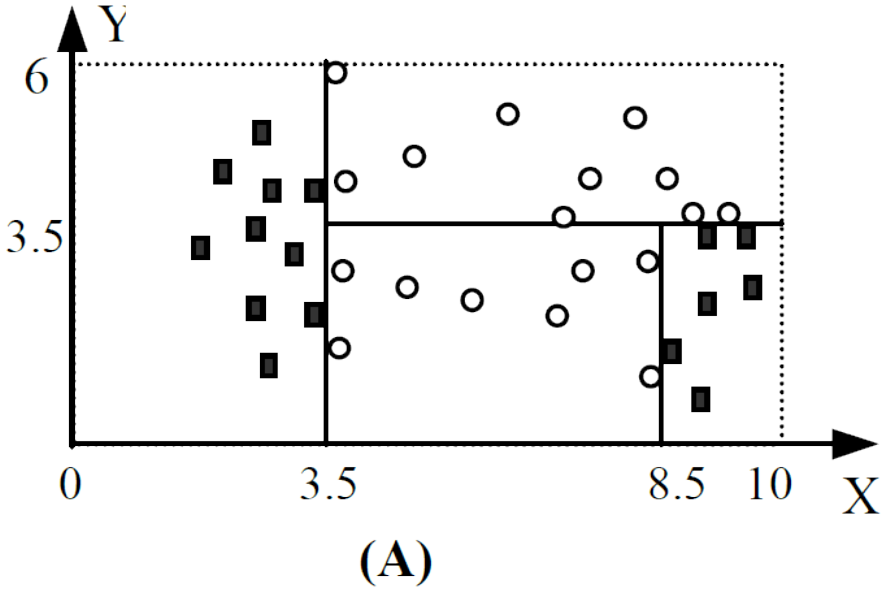
Decision Tree

Principles

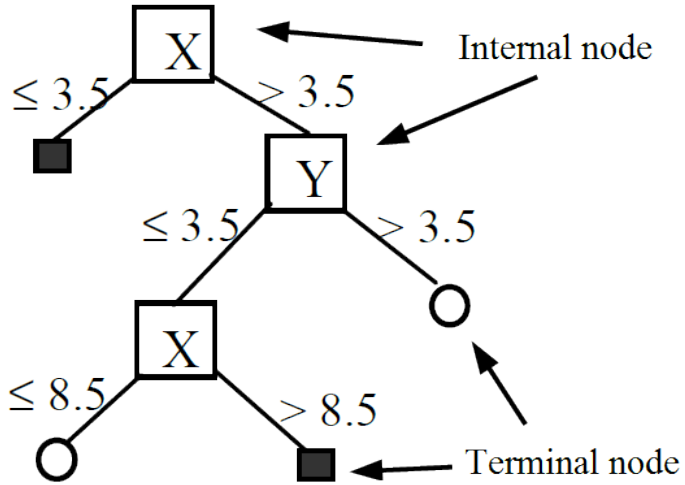
- A predictive model uses a set of binary rules applied to calculate a target value.
- Can be used for classification (categorical variables) or regression (continuous variables) applications.
- Different algorithms are proposed to determine the best split at a node

Decision Tree

Example:



(A)

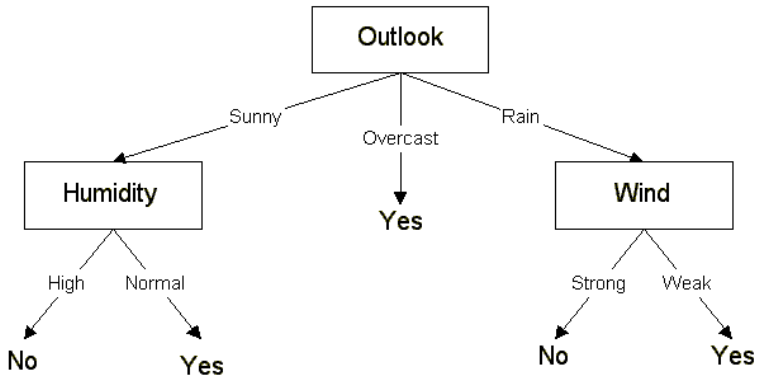


(B)

- Internal node represents a split rule: the feature is used to split and the value of the split.
- Terminal (leaf) node consists of data objects of a class. Each leaf node is marked with a class label.

Decision Tree

Day	Outlook	Temp.	Humidity	Wind	Play Tennis
D1	Sunny	Hot	High	Weak	No
D2	Sunny	Hot	High	Strong	No
D3	Overcast	Hot	High	Weak	Yes
D4	Rain	Mild	High	Weak	Yes
D5	Rain	Cool	Normal	Weak	Yes
D6	Rain	Cool	Normal	Strong	No
D7	Overcast	Cool	Normal	Weak	Yes
D8	Sunny	Mild	High	Weak	No
D9	Sunny	Cool	Normal	Weak	Yes
D10	Rain	Mild	Normal	Strong	Yes
D11	Sunny	Mild	Normal	Strong	Yes
D12	Overcast	Mild	High	Strong	Yes
D13	Overcast	Hot	Normal	Weak	Yes
D14	Rain	Mild	High	Strong	No



Algorithm

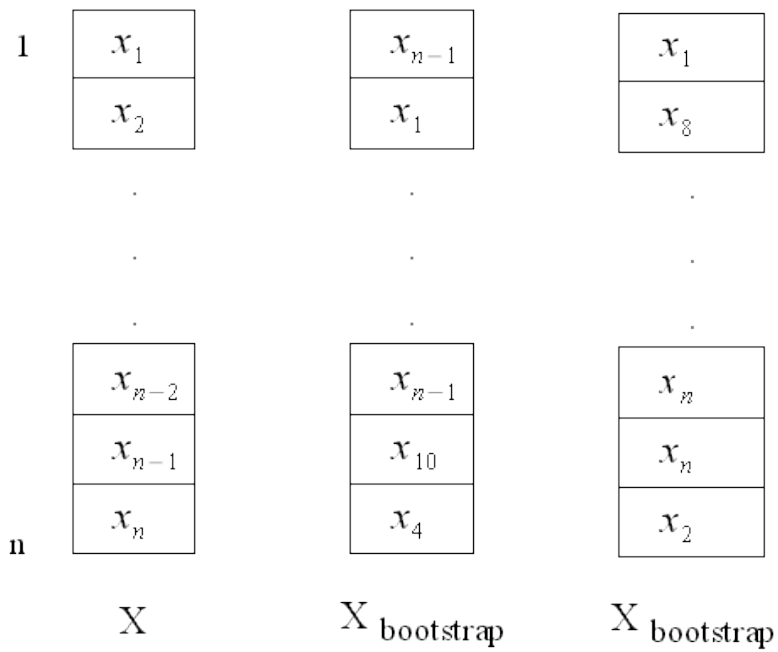
INPUT: $\mathcal{X} = \{\mathbf{x}_1, .., \mathbf{x}_n\} \in \mathbb{R}^d$

1. Create K bootstrap sets $\mathcal{X}_1, \mathcal{X}_2, ..., \mathcal{X}_K$ from the input sample $\mathcal{X}$
2. Learn an un-pruned decision tree on each learning set.
   Learning: at each internal node
   - Randomly select $m < d$ features
   - Determine the best split using only these selected features
3. Prediction: Use the rules given from all trees in the forest on the test set
   - Majority vote
   - Average of responses

Algorithm

Step 1: Create K boostrap sets

- Consider input set $\mathcal{X}$ with n training instances
- Create a new training set $\mathcal{X}_i$ by drawing n objects with replacement



Step 2: Determine the best split Suppose that $\mathcal{X}$ consists of n

instances of C classes and n_c $(c = 1, .., C)$ is denoted the cardinality of class c.

After the splitting of a feature, the input set is divided into two subsets. We denote respectively the cardinality of class c in these two subsets by n_{c_1} and n_{c_2} and $n_i = \sum_{c=1}^{C} n_{c_i}, \forall i = 1, 2.$

Step 2: Determine the best split

We want to determine which feature in a given set of training feature vectors is the most useful for discriminating the classes.

Step 2: Determine the best split

We want to determine which feature in a given set of training feature vectors is the most useful for discriminating the classes.

1. Choose the feature r providing the highest information gain to split the training set into two subsets.

Step 2: Determine the best split

We want to determine which feature in a given set of training feature vectors is the most useful for discriminating the classes.

1. Choose the feature r providing the highest information gain to split the training set into two subsets.
2. Construct child nodes after the splitting. Each child node contains a respective subset.

Step 2: Determine the best split

We want to determine which feature in a given set of training feature vectors is the most useful for discriminating the classes.

1. Choose the feature r providing the highest information gain to split the training set into two subsets.
2. Construct child nodes after the splitting. Each child node contains a respective subset.
3. Repeat 1 and 2 until each child node consists of only instances from a pure class (terminal or leaf node).

Step 2: Determine the best split

We want to determine which feature in a given set of training feature vectors is the most useful for discriminating the classes.

1. Choose the feature r providing the highest information gain to split the training set into two subsets.
2. Construct child nodes after the splitting. Each child node contains a respective subset.
3. Repeat 1 and 2 until each child node consists of only instances from a pure class (terminal or leaf node).

It is possible to use another criterion as the split rule such as: Gini index, distance measures, etc...

Information gain and Entropy

Information gain I_r is to calculate the relevance of a feature r. The goal is to maximize I_r:

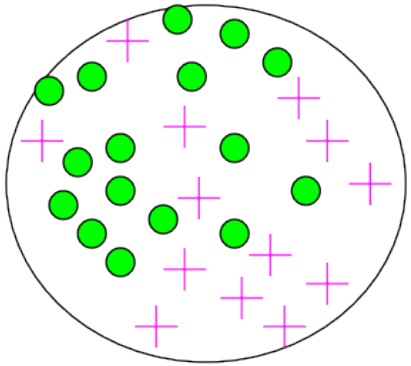
$$I_r = E_0 - \sum_{i=1}^{k} \frac{n_i}{n} E_i$$

where E is the entropy that measures the impurity in a group of objects

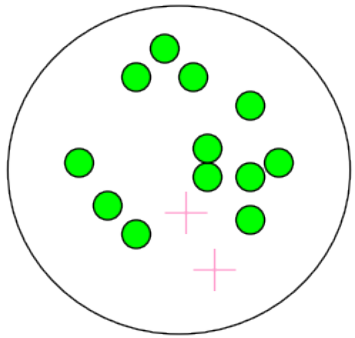
$$E_0 = \sum_{c=1}^{C} -\frac{n_c}{n} \log_2 \frac{n_c}{n}$$

$$E_i = \sum_{c=1}^{C} -\frac{n_{c_i}}{n_i} \log_2 \frac{n_{c_i}}{n_i}$$

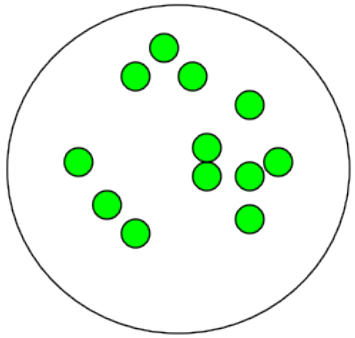
Example: 2-class case



Very impure group of objects



Less impure



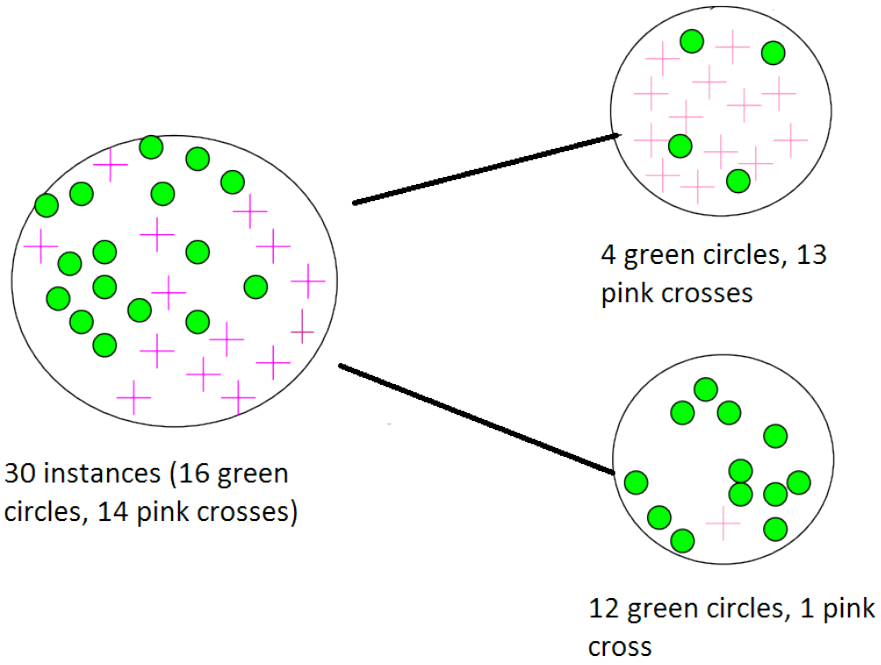
Pure group of objects

Case 1: $E = -\frac{16}{30}\log_2\frac{16}{30} - \frac{14}{30}\log_2\frac{14}{30} = 0.99$ (Impure set, need training for learning)

Case 2: $E = -\frac{2}{14}\log_2\frac{2}{14} - \frac{12}{14}\log_2\frac{12}{14} = 0.59$

Case 3: $E = -1\log_2 1 = 0$ (Pure set, no need training)

Example: 2-class case

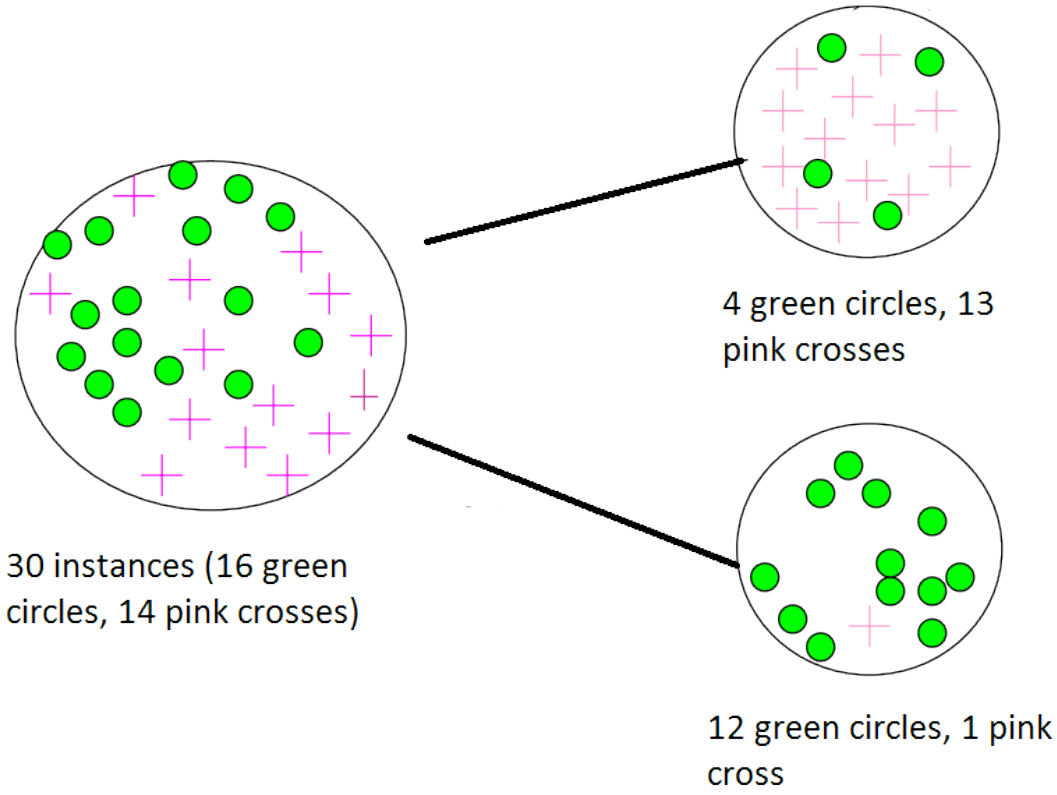


We have $E_0 = 0.99$

$E_1 = -\frac{13}{17}\log_2\frac{13}{17} - \frac{4}{17}\log_2\frac{4}{17} = 0.787$ and

$E_2 = -\frac{1}{13}\log_2\frac{1}{13} - \frac{12}{13}\log_2\frac{12}{13} = 0.391$

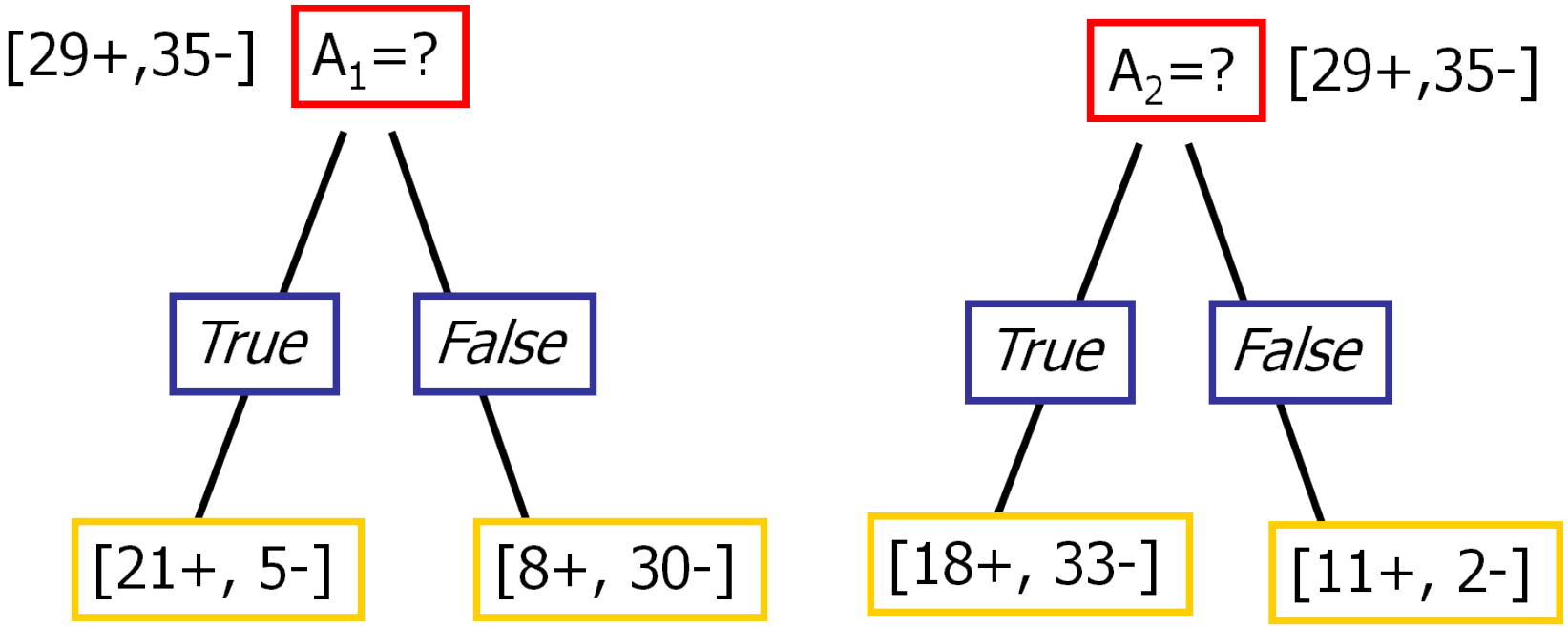
Example: 2-class case



Information gain:

$$I_r = E_0 - \frac{17}{30}E_1 - \frac{13}{30}E_2 = 0.99 - \frac{17}{30}0.787 - \frac{13}{30}0.391 = 0.38$$

Decision Tree



For A_1

$$E_0([29+, 35-]) = -\frac{29}{64} log_2 \frac{29}{64} - \frac{35}{64} log_2 \frac{35}{64} = 0.99$$

$$E([21+, 5-]) = 0.71$$

$$E([8+, 30-]) = 0.74$$

$$I(A_1) = E_0 - \frac{26}{64} * E([21+, 5-]) - \frac{38}{64} * E([8+, 30-]) = 0.27$$

For A_2

$$E_0([29+, 35-]) = -\frac{29}{64} log_2 \frac{29}{64} - \frac{35}{64} log_2 \frac{35}{64} = 0.99$$

$$E([18+, 33-]) = 0.94$$

$$E([8+, 30-]) = 0.62$$

$$I(A_2) = E_0 - \frac{51}{64} * E([18+, 33-]) - \frac{13}{64} * E([11+, 2-]) = 0.12$$

Algorithm

Step 3: Predict instance class

Each instance from the test set

- In each built tree in forests, it moves top-down from the root and gets verified by the rules in internal nodes.
- When it reaches a leaf node, this instance will be assigned to the class label of this leaf.
- A majority vote is performed. The majority class will be finally decided to label this instance

Pros and Cons

Advantages

- No overffitting, low variance due to bagging ensemble
- Natural and fast to parallelize
- No pruning required in order to generalize well
- Better prediction accuracy than single decision trees
- Low complexity $\theta kdnlogn$
- Simple to understand and to interpret

Disadvantages

- Duplication of trees is possible. Some trees can be bad.
- No optimal tree.
- Each tree can be visualized but not the forest.