Supervised Learning

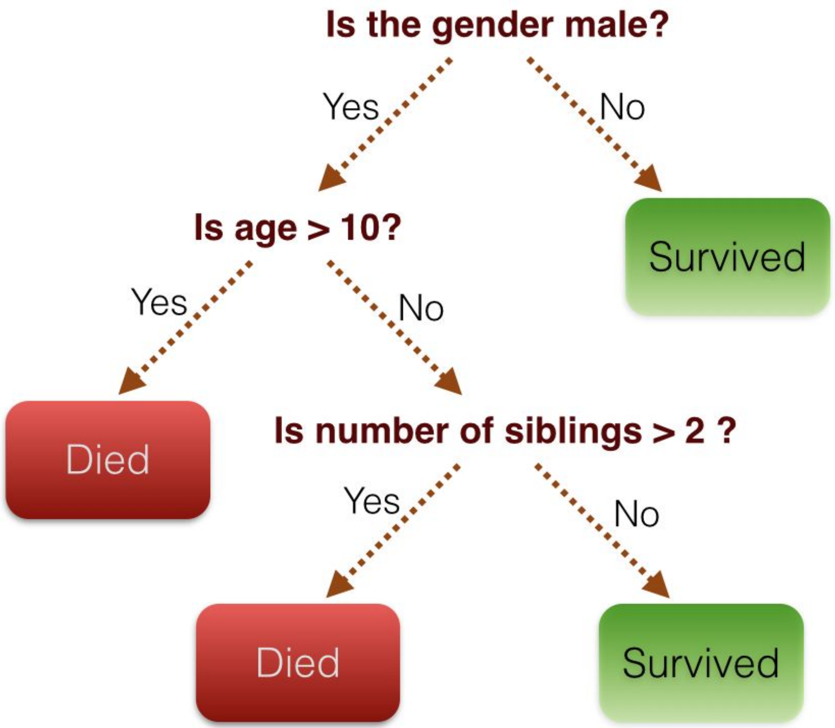
Amir Hosein Hadian

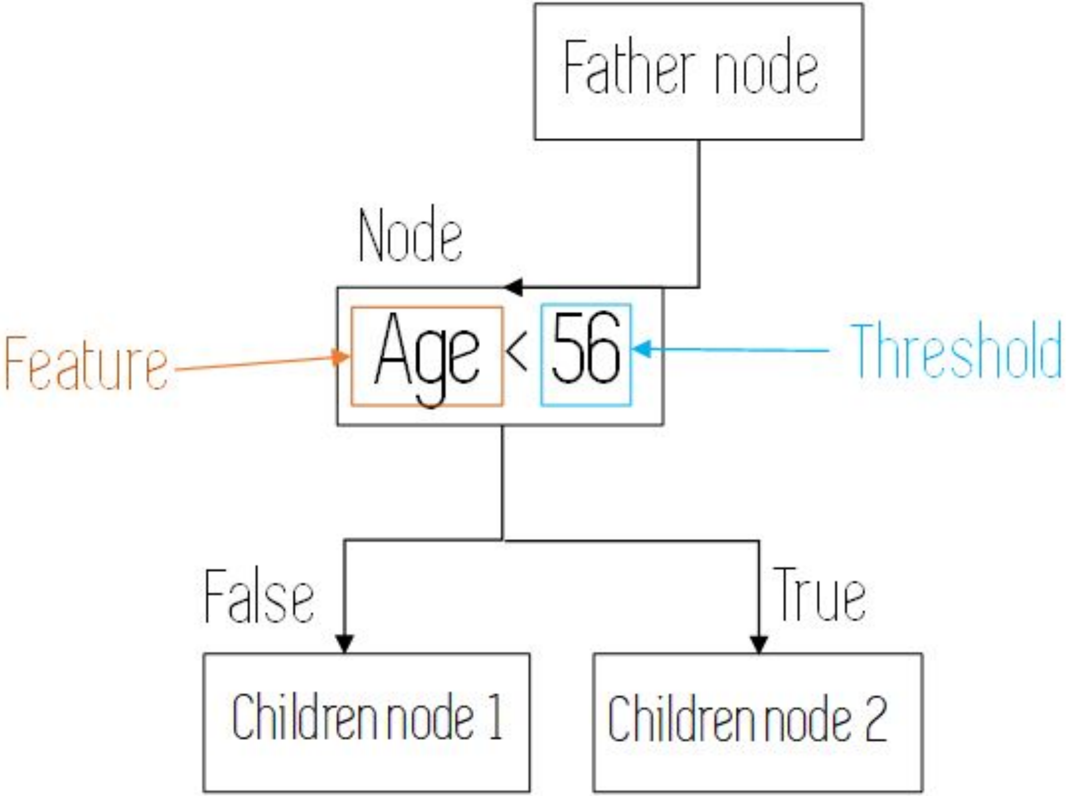
Outline

Decision tree

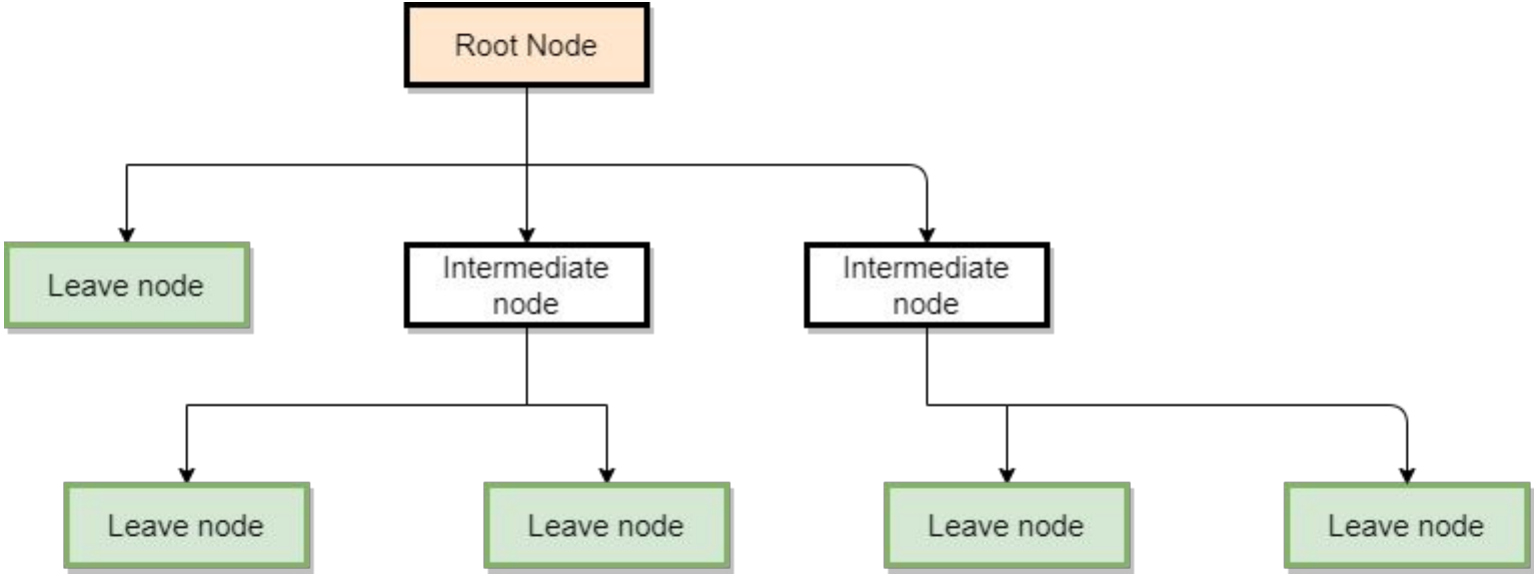
Random Forest

Model evaluation


Is the gender male?
Yes
No
Is age > 10?
Survived
Yes
No
Died
Is number of siblings > 2 ?
Yes
No
Died
Survived



Father node
Node
Feature
Age < 56
Threshold
False
True
Children node 1
Children node 2


Root Node
Leave node
Intermediate node
Intermediate node
Leave node
Leave node
Leave node
Leave node

- How are decision trees used for classification?

- Why are decision tree classifiers so popular?

Algorithm: Generate_decision_tree. Generate a decision tree from the training tuples of data partition, *D*.

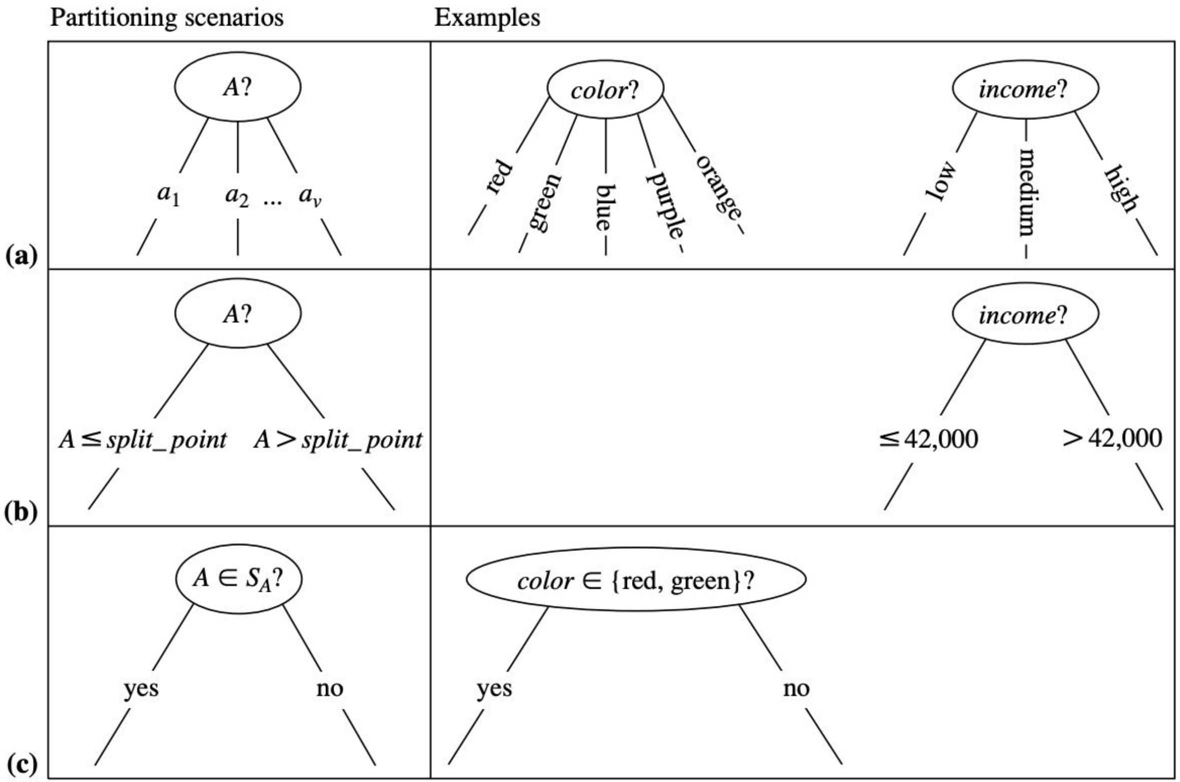
Input:

- Data partition, *D*, which is a set of training tuples and their associated class labels;
- *attribute_list*, the set of candidate attributes;
- *Attribute_selection_method*, a procedure to determine the splitting criterion that "best" partitions the data tuples into individual classes. This criterion consists of a *splitting_attribute* and, possibly, either a *split-point* or *splitting subset*.

Output: A decision tree.

Method:

(1) create a node *N*;
(2) **if** tuples in *D* are all of the same class, *C*, **then**
(3)   return *N* as a leaf node labeled with the class *C*;
(4) **if** *attribute_list* is empty **then**
(5)   return *N* as a leaf node labeled with the majority class in *D*; // majority voting
(6) apply **Attribute_selection_method**(*D*, *attribute_list*) to **find** the "best" *splitting_criterion*;
(7) label node *N* with *splitting_criterion*;
(8) **if** *splitting_attribute* is discrete-valued **and**
    multiway splits allowed **then** // not restricted to binary trees
(9)   *attribute_list* ← *attribute_list* − *splitting_attribute*; // remove *splitting_attribute*
(10) **for each** outcome *j* of *splitting_criterion*
// partition the tuples and grow subtrees for each partition
(11)   let D_j be the set of data tuples in *D* satisfying outcome *j*; // a partition
(12)   **if** D_j is empty **then**
(13)   attach a leaf labeled with the majority class in *D* to node *N*;
(14)   **else** attach the node returned by **Generate_decision_tree**(D_j, *attribute_list*) to node *N*;
endfor
(15) return *N*;


Partitioning scenarios
Examples
A?
a_1
a_2 ... a_v
color?
red
green
blue
purple
orange
income?
low
medium
high
(a)
A?
$A \leq split\_point$
$A > split\_point$
income?
$\leq 42{,}000$
$> 42{,}000$
(b)
$A \in S_A$?
yes
no
color ∈ {red, green}?
yes
no
(c)

Information Gain

$$Info(D) = -\sum_{i=1}^{m} p_i \log_2(p_i),$$

$$Info_A(D) = \sum_{j=1}^{v} \frac{|D_j|}{|D|} \times Info(D_j).$$

$$Gain(A) = Info(D) - Info_A(D).$$

Class-Labeled Training Tuples from the *AllElectronics* Customer Database

RID	*age*	*income*	*student*	*credit_rating*	*Class: buys_computer*
1	youth	high	no	fair	no
2	youth	high	no	excellent	no
3	middle_aged	high	no	fair	yes
4	senior	medium	no	fair	yes
5	senior	low	yes	fair	yes
6	senior	low	yes	excellent	no
7	middle_aged	low	yes	excellent	yes
8	youth	medium	no	fair	no
9	youth	low	yes	fair	yes
10	senior	medium	yes	fair	yes
11	youth	medium	yes	excellent	yes
12	middle_aged	medium	no	excellent	yes
13	middle_aged	high	yes	fair	yes
14	senior	medium	no	excellent	no

$$Info(D) = -\frac{9}{14}\log_2\left(\frac{9}{14}\right) - \frac{5}{14}\log_2\left(\frac{5}{14}\right) = 0.940$$

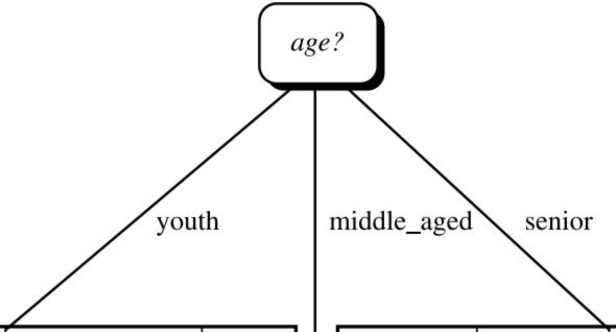
$$\begin{aligned} Info_{age}(D) = \frac{5}{14} \times \left(-\frac{2}{5}\log_2\frac{2}{5} - \frac{3}{5}\log_2\frac{3}{5}\right) &+ \frac{4}{14} \times \left(-\frac{4}{4}\log_2\frac{4}{4}\right) \\ &+ \frac{5}{14} \times \left(-\frac{3}{5}\log_2\frac{3}{5} - \frac{2}{5}\log_2\frac{2}{5}\right) \\ &= 0.694 \end{aligned}$$

$$Gain(age) = Info(D) - Info_{age}(D) = 0.940 - 0.694 = 0.246$$

$$Gain(income) = 0.029$$

$$Gain(student) = 0.151$$

$$Gain(credit\_rating) = 0.048$$



income	*student*	*credit_rating*	*class*
high	no	fair	no
high	no	excellent	no
medium	no	fair	no
low	yes	fair	yes
medium	yes	excellent	yes

income	*student*	*credit_rating*	*class*
medium	no	fair	yes
low	yes	fair	yes
low	yes	excellent	no
medium	yes	fair	yes
medium	no	excellent	no

income	*student*	*credit_rating*	*class*
high	no	fair	yes
low	yes	excellent	yes
medium	no	excellent	yes
high	yes	fair	yes

$$SplitInfo_A(D) = -\sum_{j=1}^{v} \frac{|D_j|}{|D|} \times \log_2\left(\frac{|D_j|}{|D|}\right).$$

$$GainRatio(A) = \frac{Gain(A)}{SplitInfo_A(D)}.$$

$$SplitInfo_{income}(D) = -\frac{4}{14} \times \log_2\left(\frac{4}{14}\right) - \frac{6}{14} \times \log_2\left(\frac{6}{14}\right) - \frac{4}{14} \times \log_2\left(\frac{4}{14}\right)$$

$$= 1.557.$$

$$Gain(income) = 0.029.$$

$$GainRatio(income) = 0.029/1.557 = 0.019.$$

Gini Index

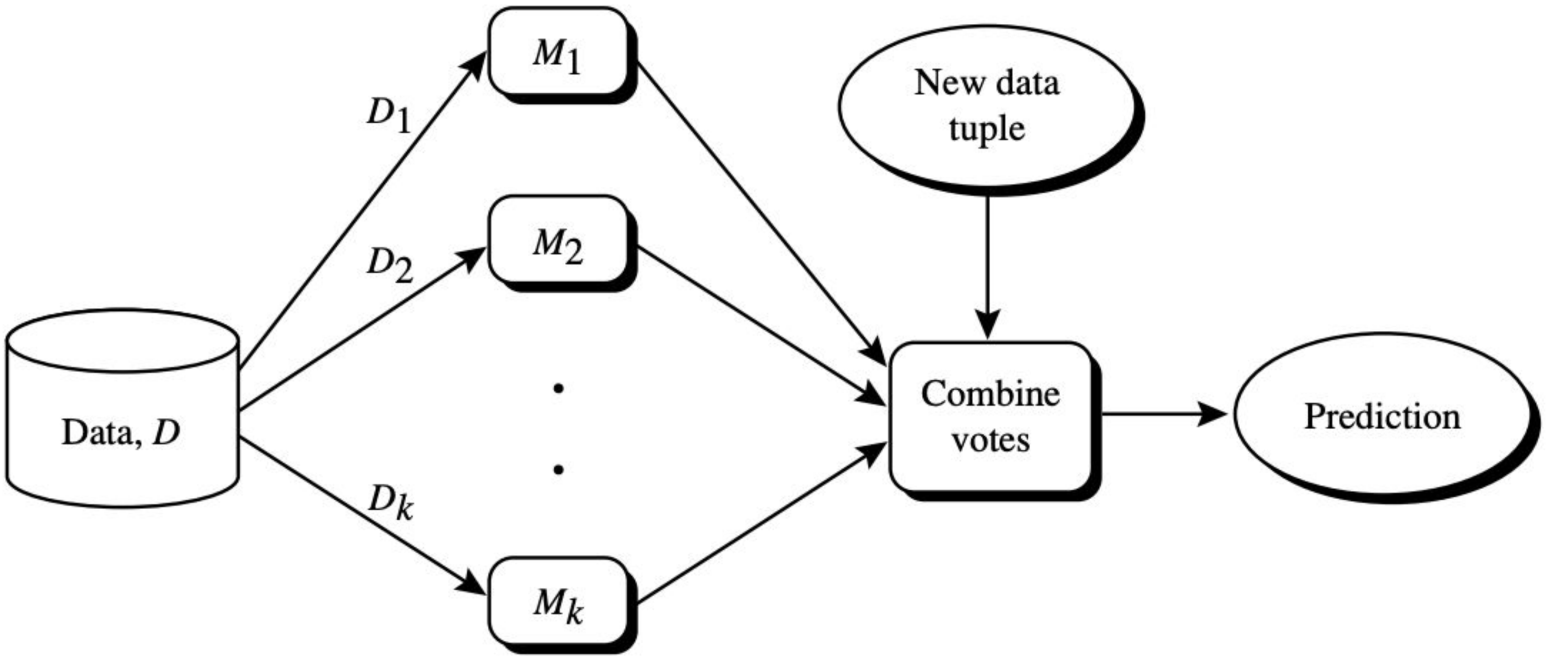
$$Gini(D) = 1 - \sum_{i=1}^{m} p_i^2,$$

$$Gini_A(D) = \frac{|D_1|}{|D|} Gini(D_1) + \frac{|D_2|}{|D|} Gini(D_2).$$

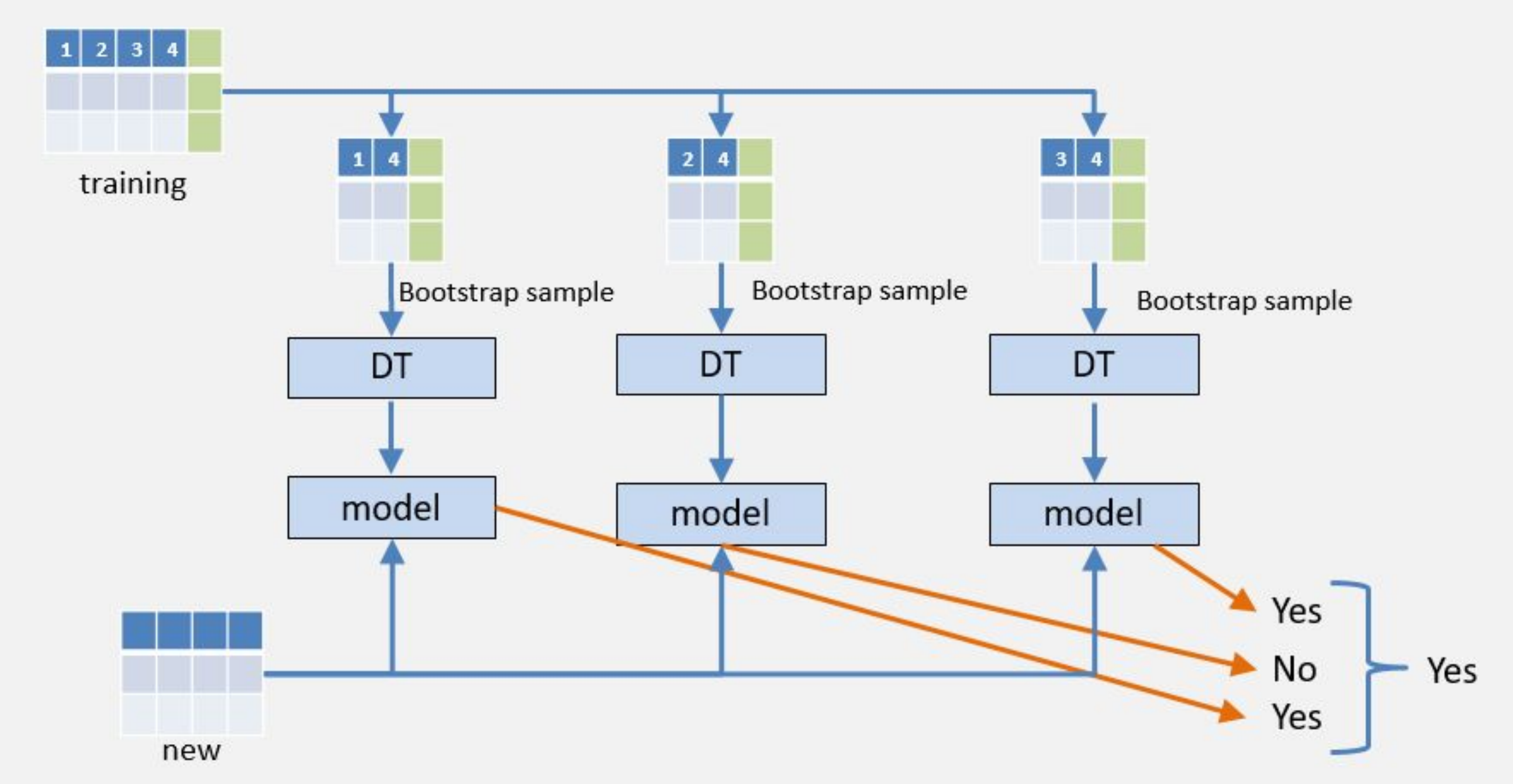
$$\Delta Gini(A) = Gini(D) - Gini_A(D).$$

$$Gini(D) = 1 - \left(\frac{9}{14}\right)^2 - \left(\frac{5}{14}\right)^2 = 0.459.$$

$$\begin{aligned}
&Gini_{income \in \{low, medium\}}(D) \\
&\quad = \frac{10}{14} Gini(D_1) + \frac{4}{14} Gini(D_2) \\
&\quad = \frac{10}{14}\left(1 - \left(\frac{7}{10}\right)^2 - \left(\frac{3}{10}\right)^2\right) + \frac{4}{14}\left(1 - \left(\frac{2}{4}\right)^2 - \left(\frac{2}{4}\right)^2\right) \\
&\quad = 0.443 \\
&\quad = Gini_{income \in \{high\}}(D).
\end{aligned}$$



Data, D
D_1
D_2
D_k
M_1
M_2
M_k
New data tuple
Combine votes
Prediction


1 2 3 4
training
1 4
2 4
3 4
Bootstrap sample
Bootstrap sample
Bootstrap sample
DT
DT
DT
model
model
model
new
Yes
No
Yes
Yes

Bagging

Algorithm: Bagging. The bagging algorithm—create an ensemble of classification models for a learning scheme where each model gives an equally weighted prediction.

Input:

- D, a set of d training tuples;
- k, the number of models in the ensemble;
- a classification learning scheme (decision tree algorithm, naïve Bayesian, etc.).

Output: The ensemble—a composite model, $M*$.

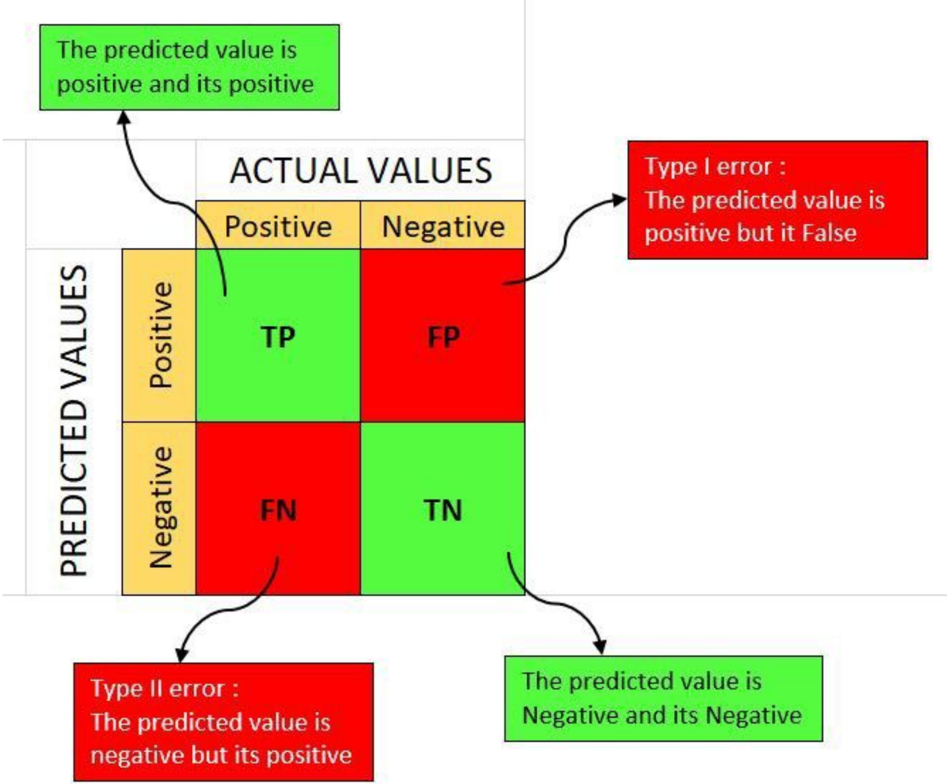
Method:

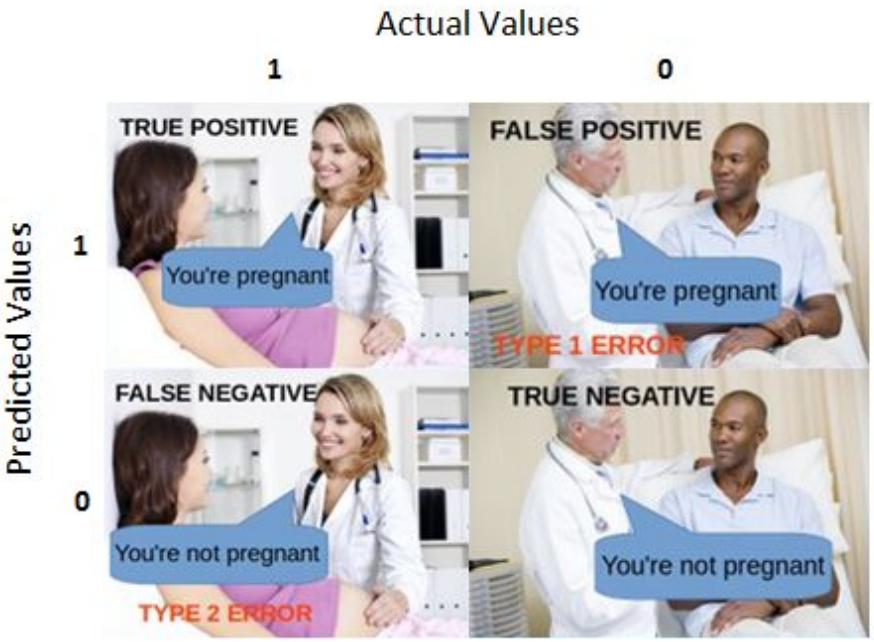
(1) **for** $i = 1$ to k **do** // create k models:
(2) create bootstrap sample, D_i, by sampling D with replacement;
(3) use D_i and the learning scheme to derive a model, M_i;
(4) **endfor**

To use the ensemble to classify a tuple, *X*:

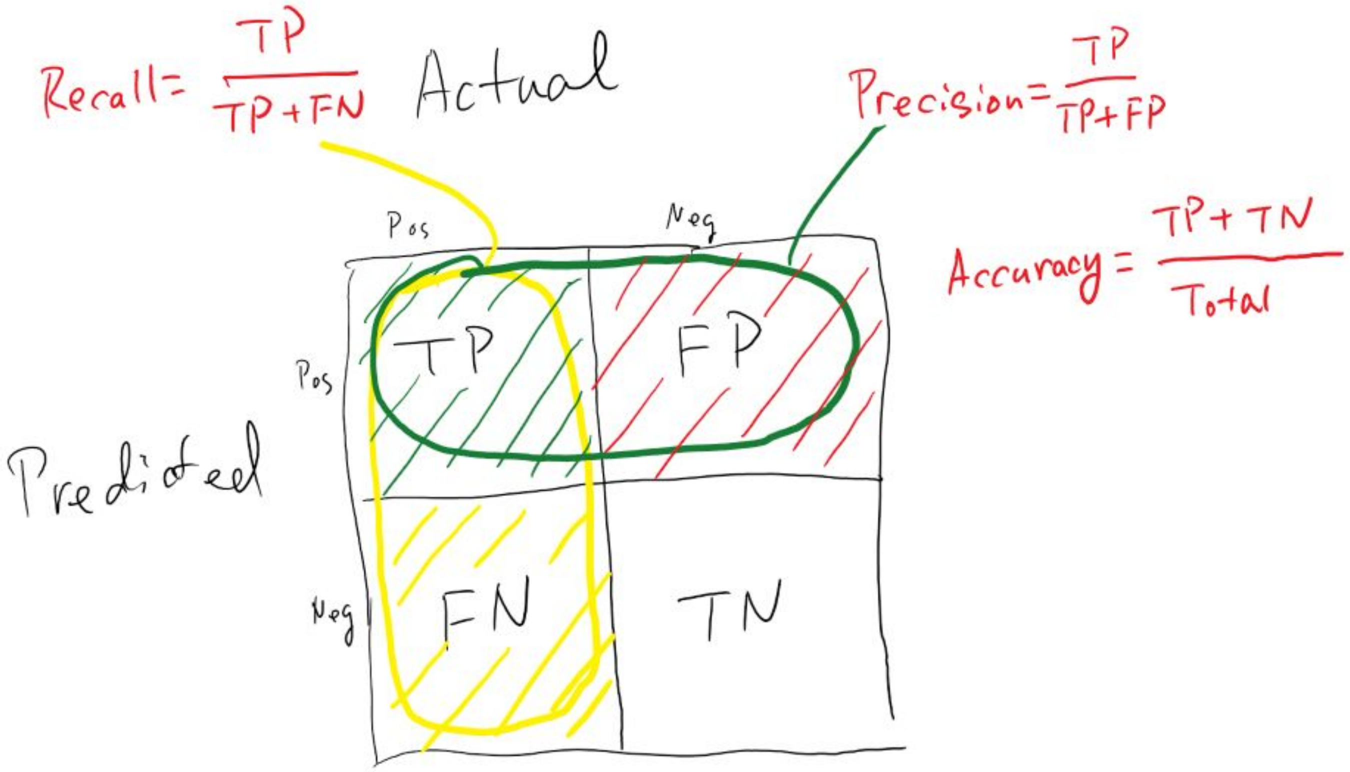
let each of the k models classify $\boldsymbol{X}$ and return the majority vote;

Model Evaluation

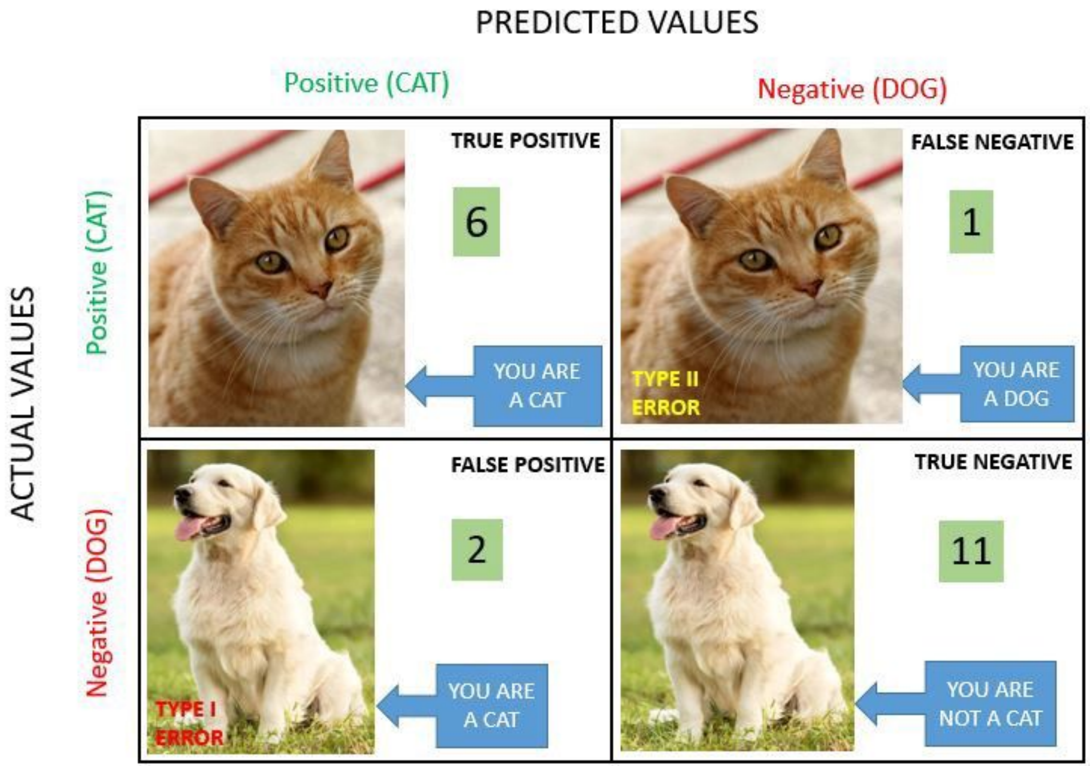




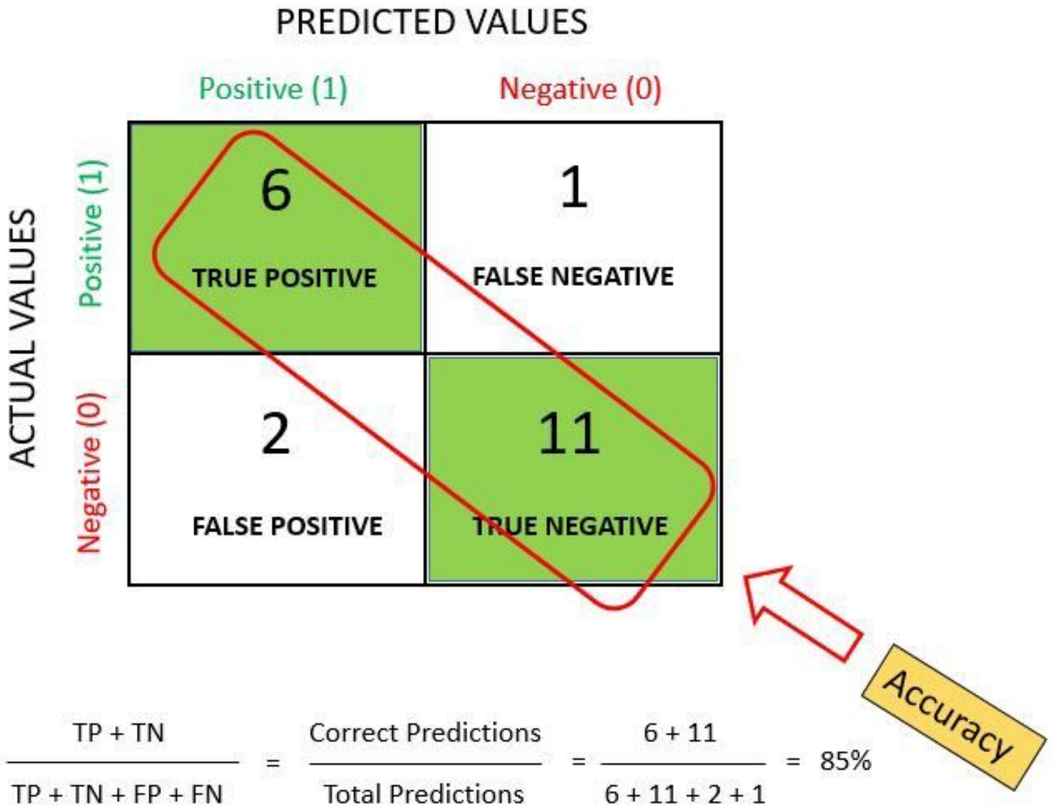
Model Evaluation



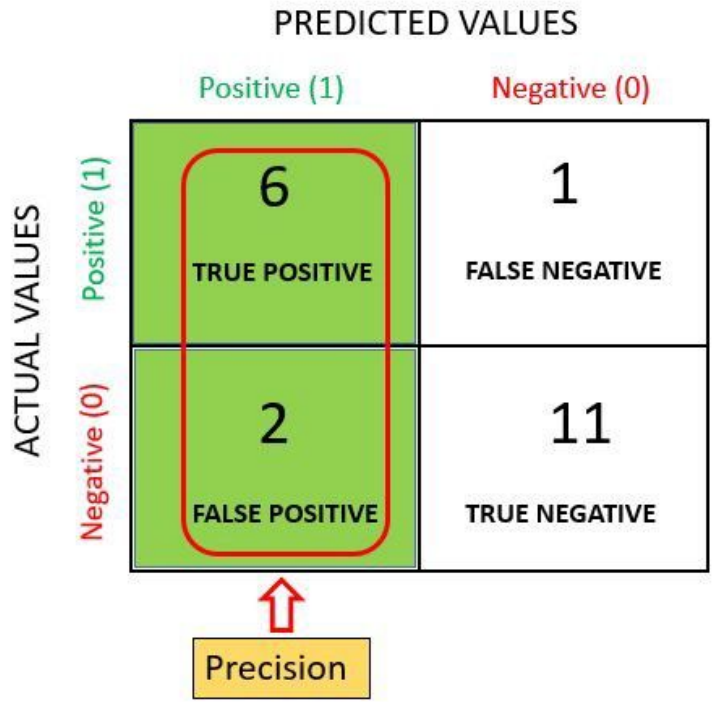
Model Evaluation



Model Evaluation

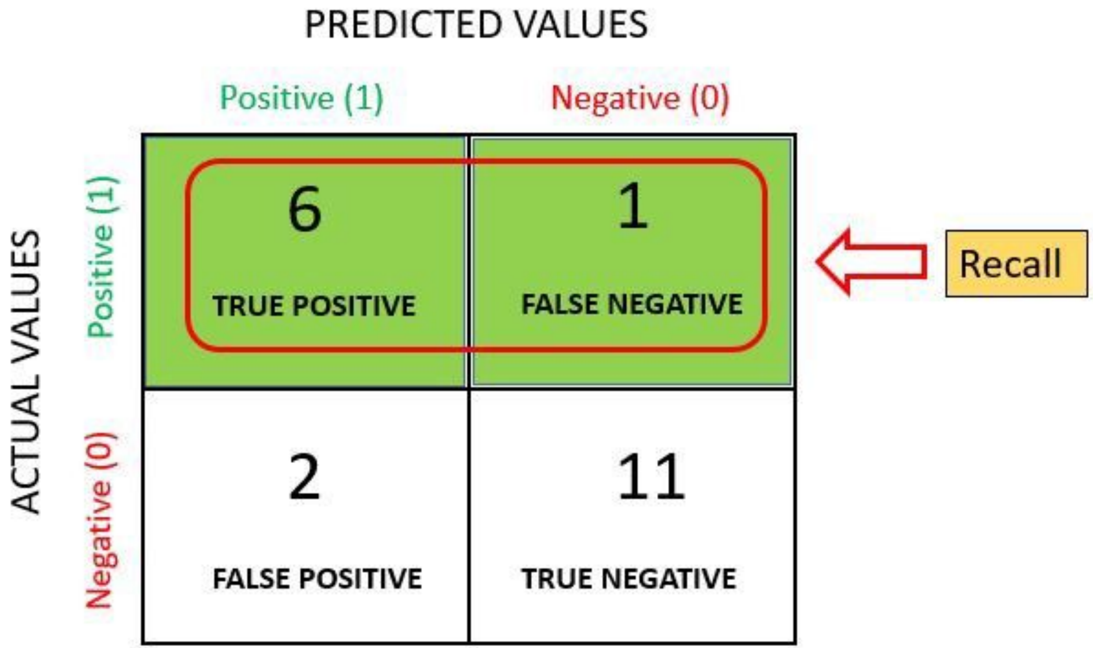


$$\text{Accuracy} = \frac{TP + TN}{TP + TN + FP + FN} = \frac{\text{Correct Predictions}}{\text{Total Predictions}} = \frac{6 + 11}{6 + 11 + 2 + 1} = 85\%$$



$$\text{Precision} = \frac{TP}{TP + FP} = \frac{\text{Predictions Actually Positive}}{\text{Total Predicted positive}} = \frac{6}{6+2} = 0.75$$

Model Evaluation



$$\text{Recall} = \frac{TP}{TP + FN} = \frac{\text{Predictions Actually Positive}}{\text{Total Actual positive}} = \frac{6}{6+1} = 0.85$$

Model Evaluation

$$\text{F1-Score} = 2 * \frac{(\text{Recall} * \text{Precision})}{(\text{Recall} + \text{Precision})} = 2 * \frac{(0.85 * 0.75)}{(0.85 + 0.75)} = 0.79$$

Model Evaluation