Decision trees

Thanh-Nghi Do

Can Tho University

dtnghi@cit.ctu.edu.vn

Can Tho
Dec. 2019

Content

- Introduction
- Decision trees
- Decision forests

Content

- **<u>Introduction</u>**
- Decision trees
- Decision forests

Top 10 Data Mining Algorithms
(Kdnuggets)

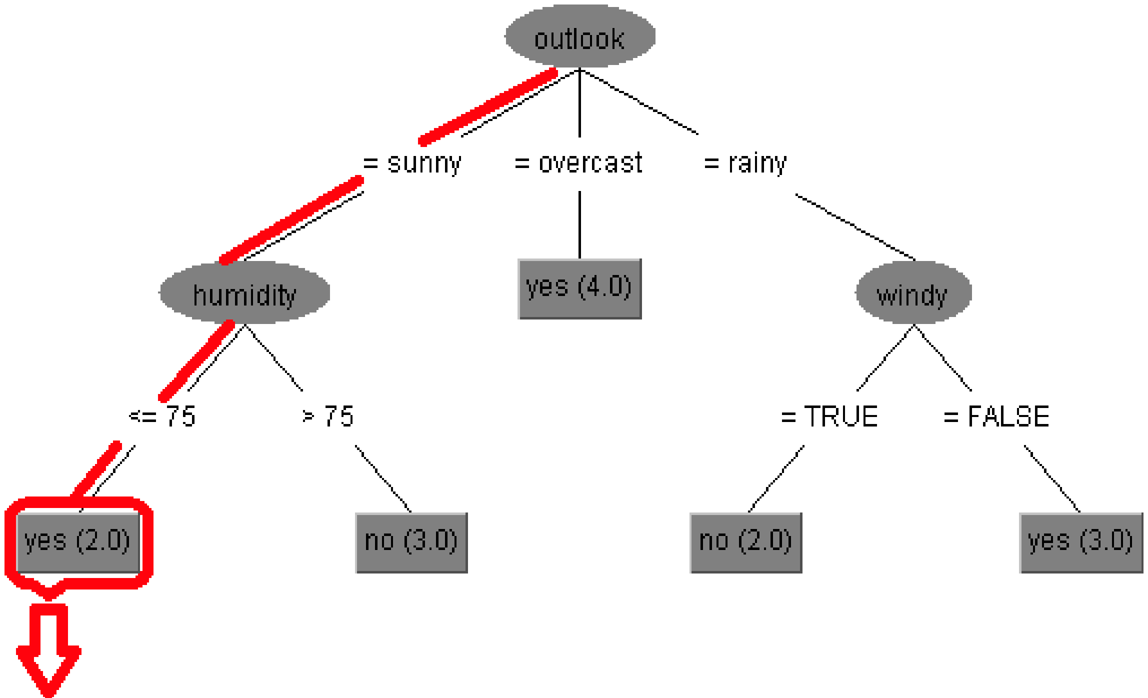
Here are the algorithms:

- 1. C4.5
- 2. k-means
- 3. Support vector machines
- 4. Apriori
- 5. EM
- 6. PageRank
- 7. AdaBoost
- 8. kNN
- 9. Naive Bayes
- 10. CART

Introduction

- Decision trees
  - Graphical representation of the classification procedure
  - Intuition and comprehension for user
  - Decision rules: **if … then …**
  - Training time: short
  - Dealing with classification and regression
  - Handling different data types
  - Applied into data analysis, bioinformatics, text mining, etc.
  - Algorithms: C4.5 (Quinlan, 1993), CART (Breiman et al., 1984)

Introduction



outlook	temp.	hum.	windy	Play
sunny	85	85	false	No
sunny	80	90	true	No
overcast	83	78	false	Yes
rain	70	96	false	Yes
rain	68	80	false	Yes
rain	65	70	true	No
...	...	...	...	...
...	...	...	...	...

IF (outlook=sunny) and (humidity <= 75) THEN play=yes

Introduction

- Decision tree
  - An internal node (non-terminal node, decision node): test on an attribute (variable, dimension, feature)
  - A branch represents an outcome of the test
  - A leaf node represents a class label or class label distribution
  - A new example is classified by following a matching path to a leaf node
  - A decision rule (if … then …) corresponds to a path from root to a leaf node

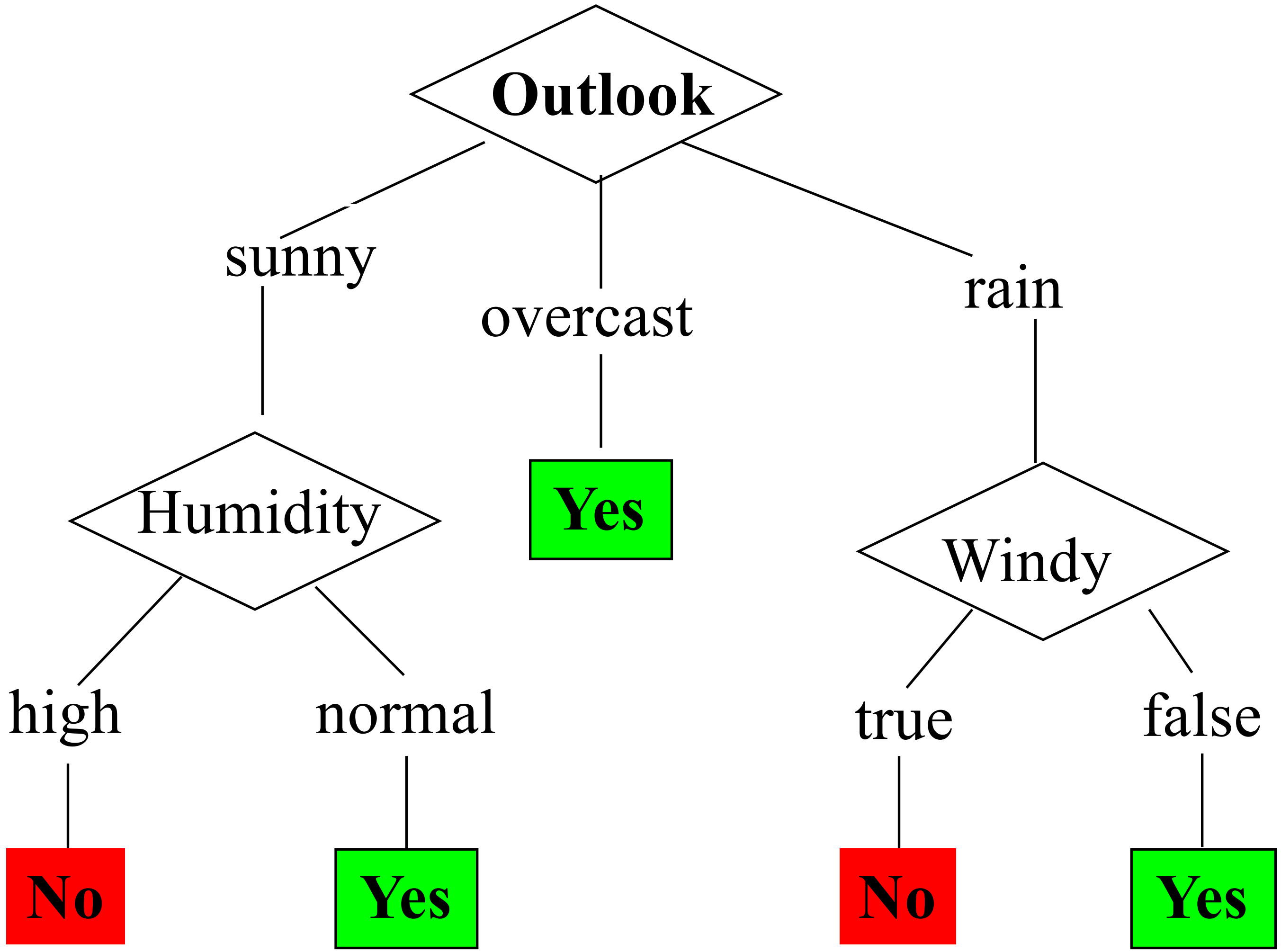
Content

- Introduction
- **<u>Decision trees</u>**
- Decision forests

Training dataset Weather [Outlook, Temp, Humidity, Windy] → Play

Outlook	Temp	Humidity	Windy	Play
Sunny	Hot	High	False	No
Sunny	Hot	High	True	No
Overcast	Hot	High	False	Yes
Rainy	Mild	High	False	Yes
Rainy	Cool	Normal	False	Yes
Rainy	Cool	Normal	True	No
Overcast	Cool	Normal	True	Yes
Sunny	Mild	High	False	No
Sunny	Cool	Normal	False	Yes
Rainy	Mild	Normal	False	Yes
Sunny	Mild	Normal	True	Yes
Overcast	Mild	High	True	Yes
Overcast	Hot	Normal	False	Yes
Rainy	Mild	High	True	No

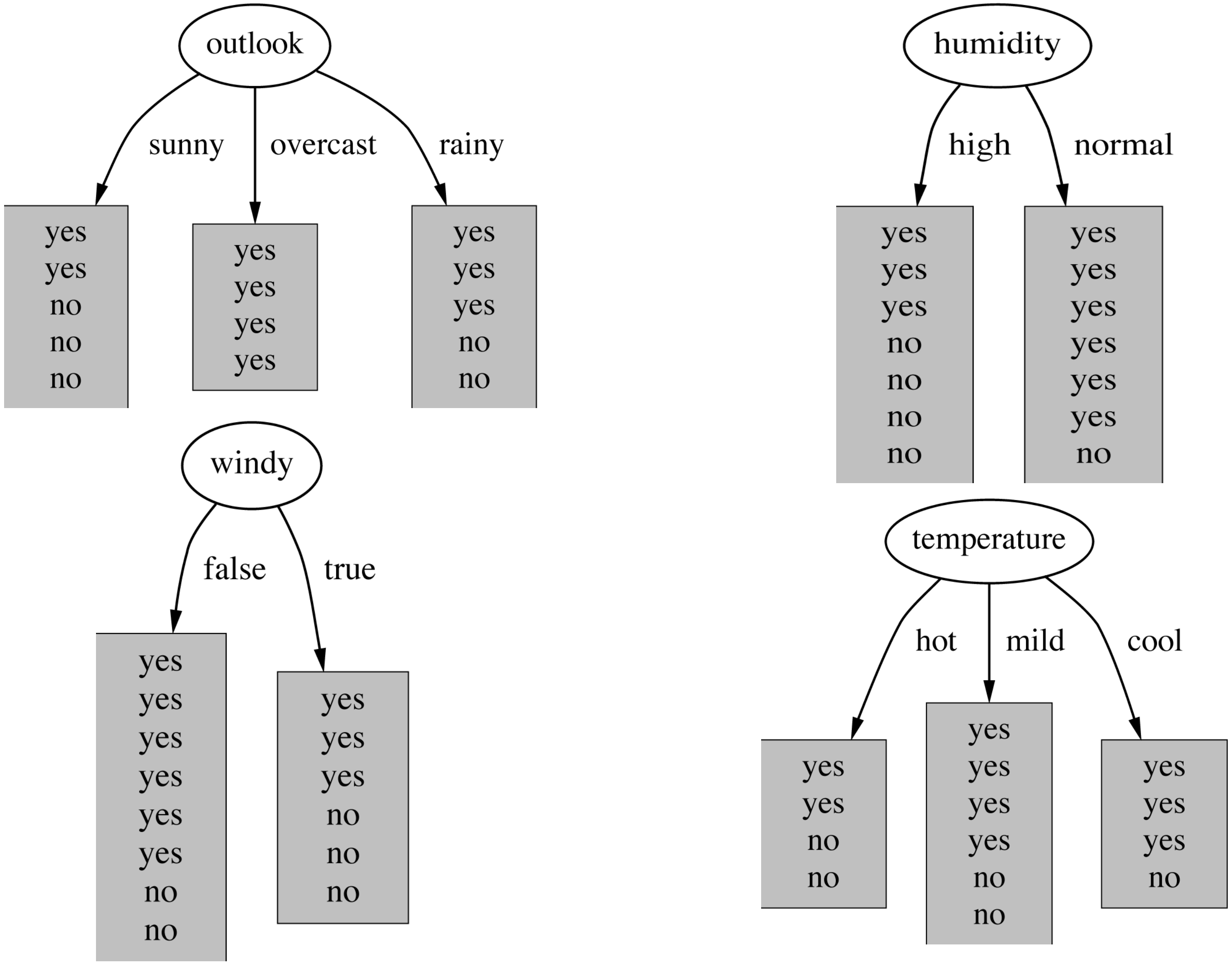
Decision tree for **weather**



Decision trees

- Learning algorithm (Top-down)
  - At start, all training examples are at the root
  - Partition the examples recursively by choosing one attribute each time
  - At each node, available attributes are evaluated on the basis of separating the classes of the training examples
  - A Goodness function is used for choosing the splitting attribute
  - Typical goodness functions: information gain (ID3/C4.5), information gain ratio, gini index

Choosing the splitting attribute?



Choosing the splitting attribute?

- Which is the best attribute?
  - The one which will result in the smallest tree
  - Heuristic: choosing the attribute that produces the "purest" nodes
- Goodness functions
  - Impurity criterion
  - When node is pure, measure should be zero
  - When impurity is maximal (i.e. all classes equally likely), measure should be maximal
  - Measure should obey multistage property (i.e. decisions can be made in several stages)
  - Shannon entropy, Gini index

Entropy (C4.5)

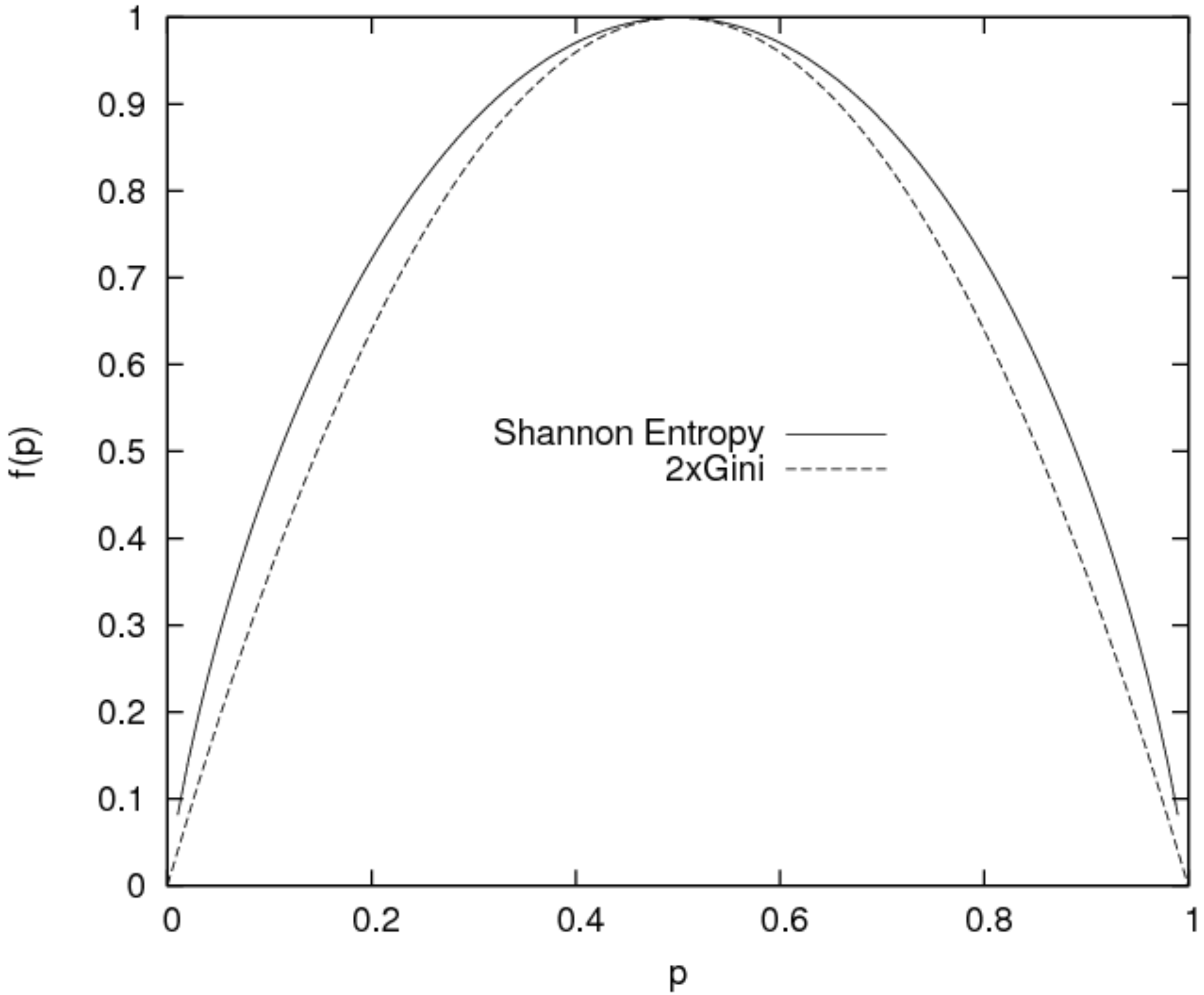
Claude Shannon

Born: 30 April 1916
Died: 23 February 2001

"Father of information theory"

$$\text{entropy}(p_1, p_2, \ldots, p_n) = -p_1 \log p_1 - p_2 \log p_2 \ldots - p_n \log p_n$$

Goodness functions for impurity criterion



Computing information

- Splitting node D (m examples) into k sub-partitions $D_1, \ldots, D_k$ ($m_1, \ldots, m_k$ examples) :

$$\text{info}(\text{part./D}) = (m_1 / m) \times \text{info}(D_1) + \ldots + (m_k / m) \times \text{info}(D_k)$$

Training dataset Weather [Outlook, Temp, Humidity, Windy] → Play

Outlook	Temp	Humidity	Windy	Play
Sunny	Hot	High	False	No
Sunny	Hot	High	True	No
Overcast	Hot	High	False	Yes
Rainy	Mild	High	False	Yes
Rainy	Cool	Normal	False	Yes
Rainy	Cool	Normal	True	No
Overcast	Cool	Normal	True	Yes
Sunny	Mild	High	False	No
Sunny	Cool	Normal	False	Yes
Rainy	Mild	Normal	False	Yes
Sunny	Mild	Normal	True	Yes
Overcast	Mild	High	True	Yes
Overcast	Hot	Normal	False	Yes
Rainy	Mild	High	True	No

Attribute **outlook**

- "Outlook" = "Sunny":

$$\text{info}([2,3]) = \text{entropy}(2/5,3/5) = -2/5\log(2/5) - 3/5\log(3/5) = 0.971\text{ bits}$$

- "Outlook" = "Overcast":

0*log(0) = 0

$$\text{info}([4,0]) = \text{entropy}(1,0) = -1\log(1) - 0\log(0) = 0\text{ bits}$$

- "Outlook" = "Rainy":

$$\text{info}([3,2]) = \text{entropy}(3/5,2/5) = -3/5\log(3/5) - 2/5\log(2/5) = 0.971\text{ bits}$$

- Info(outlook):

$$\text{info}([3,2],[4,0],[3,2]) = (5/14)\times 0.971 + (4/14)\times 0 + (5/14)\times 0.971$$
$$= 0.693\text{ bits}$$

Attribute **outlook**

- Information gain of attribute **outlook**

(information before split) – (information after split)

$$\text{gain}(\text{"Outlook"}) = \text{info}([9,5]) - \text{info}([2,3],[4,0],[3,2]) = 0.940 - 0.693$$
$$= 0.247 \text{ bits}$$

Attribute **humidity**

- "Humidity" = "High":

$$\text{info}([3,4]) = \text{entropy}(3/7,4/7) = -3/7\log(3/7) - 4/7\log(4/7) = 0.985 \text{ bits}$$

- "Humidity" = "Normal":

$$\text{info}([6,1]) = \text{entropy}(6/7,1/7) = -6/7\log(6/7) - 1/7\log(1/7) = 0.592 \text{ bits}$$

- Info(humidity):

$$\text{info}([3,4],[6,1]) = (7/14)\times 0.985 + (7/14)\times 0.592 = 0.788 \text{ bits}$$

- Gain informationnel de variable **humidity**

$$\text{info}([9,5]) - \text{info}([3,4],[6,1]) = 0.940 - 0.788 = 0.152$$

Information gain of attributes

$$\text{gain}(\text{"Outlook"}) = 0.247 \text{ bits}$$

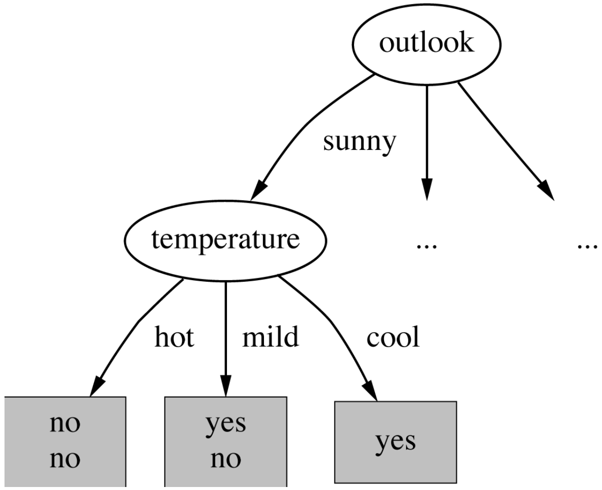
$$\text{gain}(\text{"Temperature"}) = 0.029 \text{ bits}$$

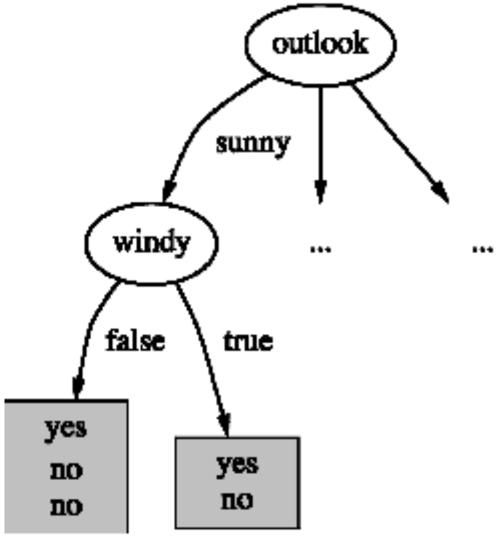
$$\text{gain}(\text{"Humidity"}) = 0.152 \text{ bits}$$

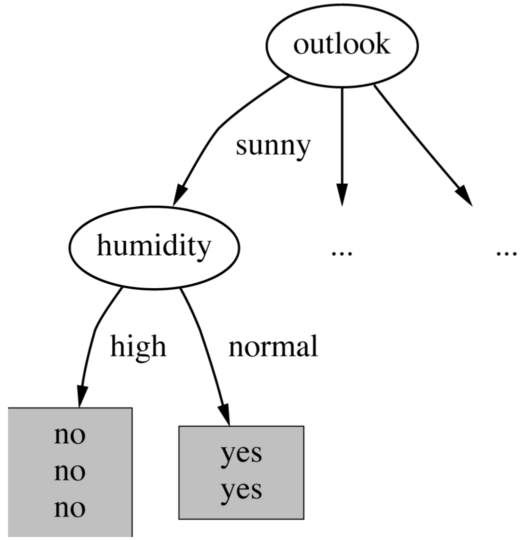
$$\text{gain}(\text{"Windy"}) = 0.048 \text{ bits}$$

- Selection: **max (gain)** or **min(info)** **=> outlook**

Branch **outlook = sunny**
Choosing temperature, humidity, windy?





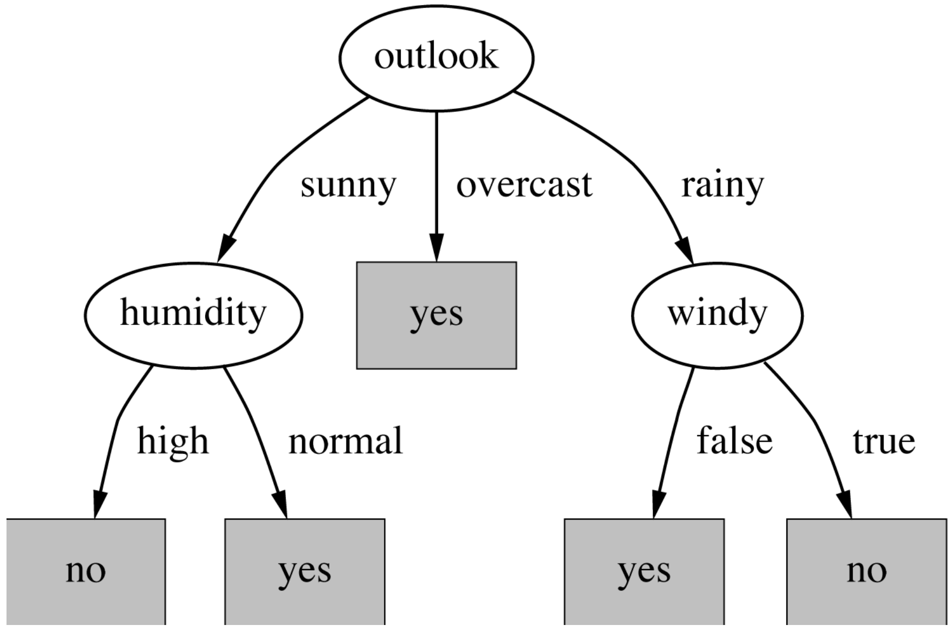


$$\text{gain}(\text{"Humidity"}) = 0.971 \text{ bits}$$

$$\text{gain}(\text{"Temperature"}) = 0.571 \text{ bits}$$

$$\text{gain}(\text{"Windy"}) = 0.020 \text{ bits}$$

Resulting tree



- Assign the class to a leaf node (terminal node): majority class

Gini (CART)

- A node D with n classes (relative frequencies $p_1, p_2, \ldots, p_n$)

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

- Splitting node D (m examples) into k sub-partitions $D_1, \ldots, D_k$ ($m_1, \ldots, m_k$ examples) :

$$\text{Gini(part./D)} = (m_1 / m) \times Gini(D_1) + \ldots + (m_k / m) \times Gini(D_k)$$

- Selection: **min(Gini)**

Continuous attribute

- Best split
  - Sort examples by the values of the numeric attribute
  - Binary split: halfway between values
  - Computing information for left, right partitions
  - Computing information for all split points

Continuous attribute

- Attribute **temperature**

64	65	68	69	70	71	72	72	75	75	80	81	83	85
Yes	No	Yes	Yes	Yes	No	No	Yes	Yes	Yes	No	Yes	Yes	No

temperature < 71.5 (left) yes/4, no/2

temperature ≥ 71.5 (right) yes/5, no/3

- Info([4,2],[5,3]) = 6/14 info([4,2]) + 8/14 info([5,3])
  = 0.939 bits

- Computing information for all split points in one pass!

Continuous attribute

- Heuristic (Fayyad & Irani, 1992): only needs to be evaluated between points of different classes

64	**65**	**68**	**69**	**70**	**71**	**72**	**72**	**75**	**75**	**80**	**81**	**83**	**85**
Yes	No	Yes	Yes	Yes	No	No	Yes	Yes	Yes	No	Yes	Yes	No

X

Breakpoints between values of the same class cannot be optimal

Overfitting

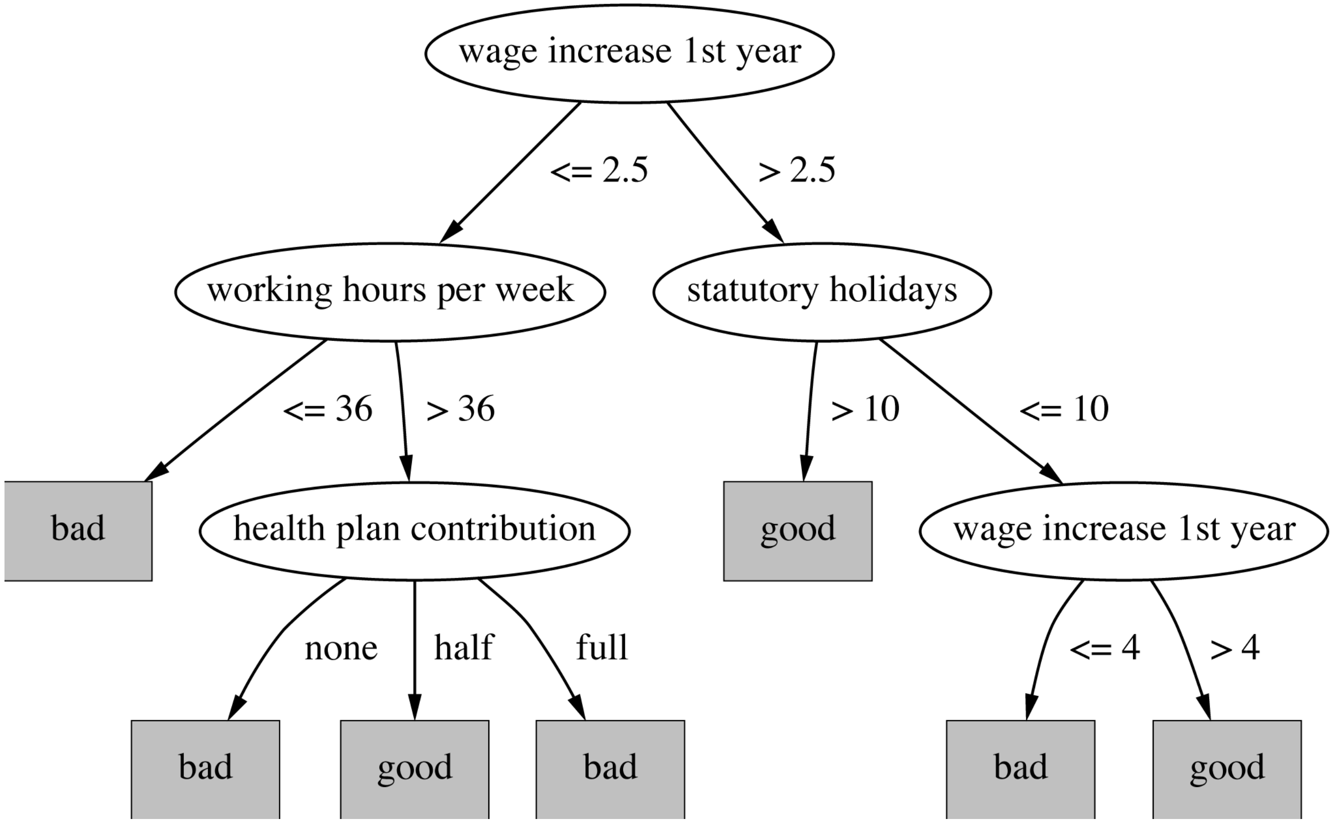
- Decision tree model fits the training data too precisely - usually leads to poor results on new data
- Gap between training and test error
- Training error is very low or even zero
- Test error is higher than training error
- Approaches for avoiding overfitting
  - Pre-pruning: stop growing the tree earlier
  - Post-pruning: allow the tree to perfectly classify the training set, and then post-prune the resulting tree

Post-pruning for avoiding overfitting

- Pruning
  - Heuristic: estimating error rates for sub-trees
  - Sub-tree replacement

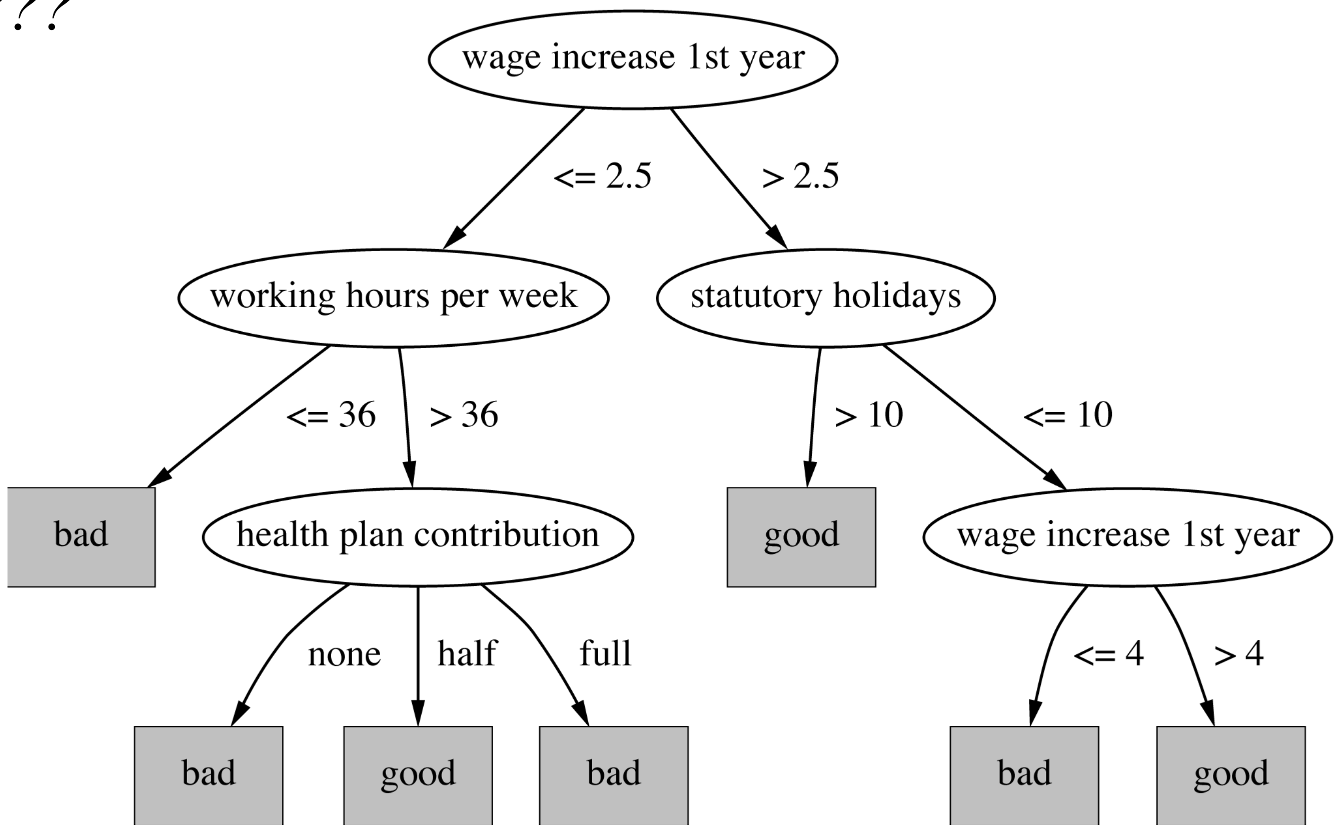
Post-pruning for avoiding overfitting

- *Bottom-up*

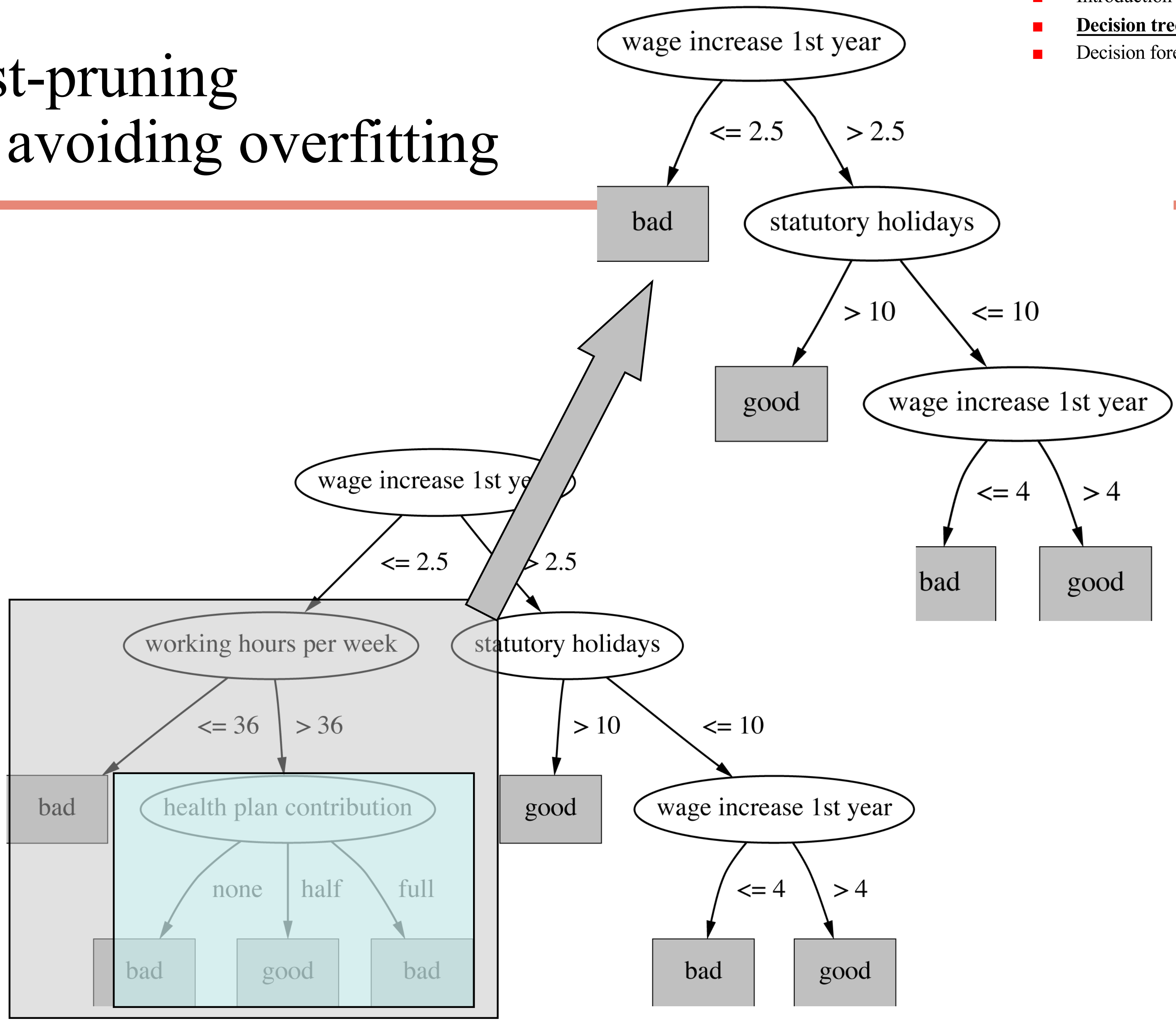


Post-pruning for avoiding overfitting

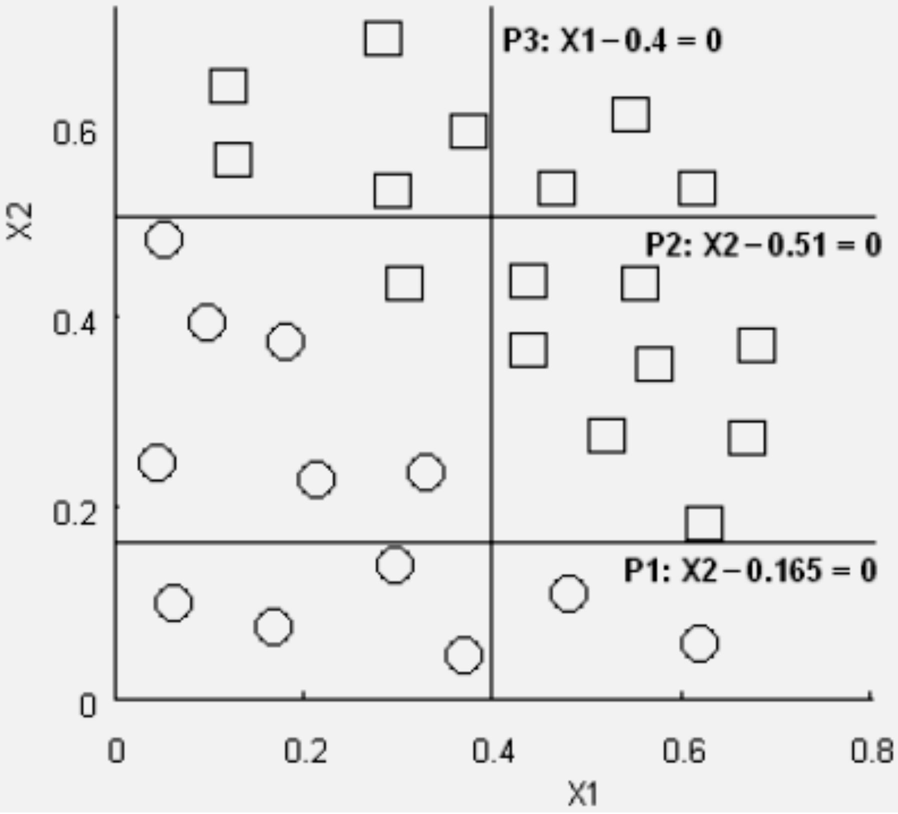
- *Bottom-up*
- *Sub-trees???*

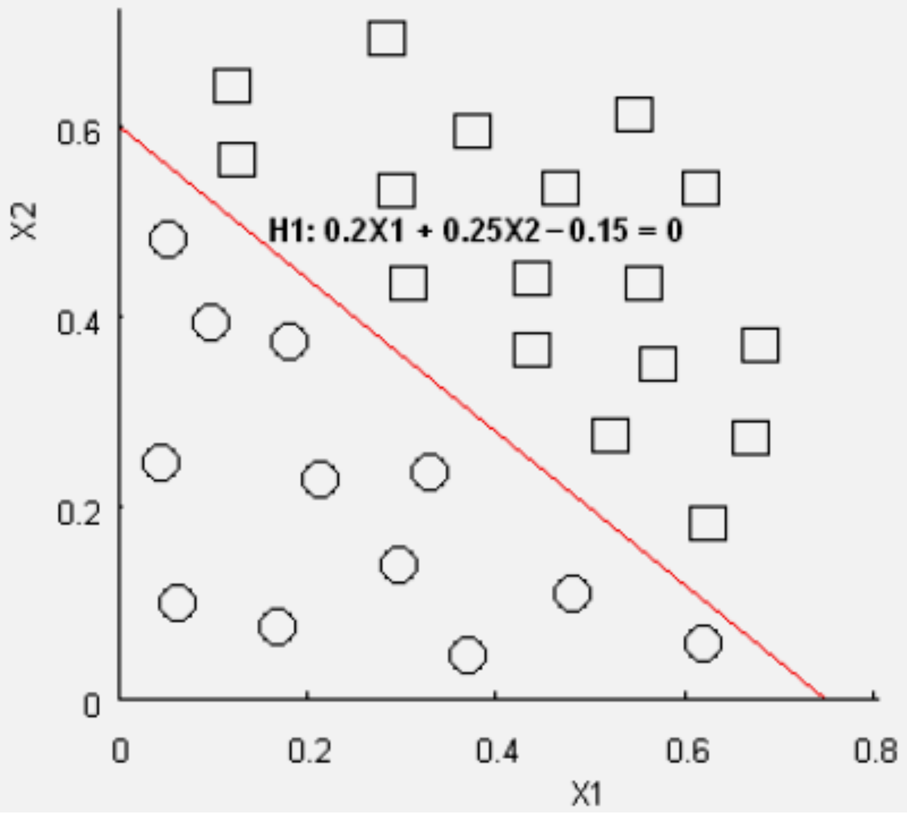


Post-pruning for avoiding overfitting



Multi-variate split



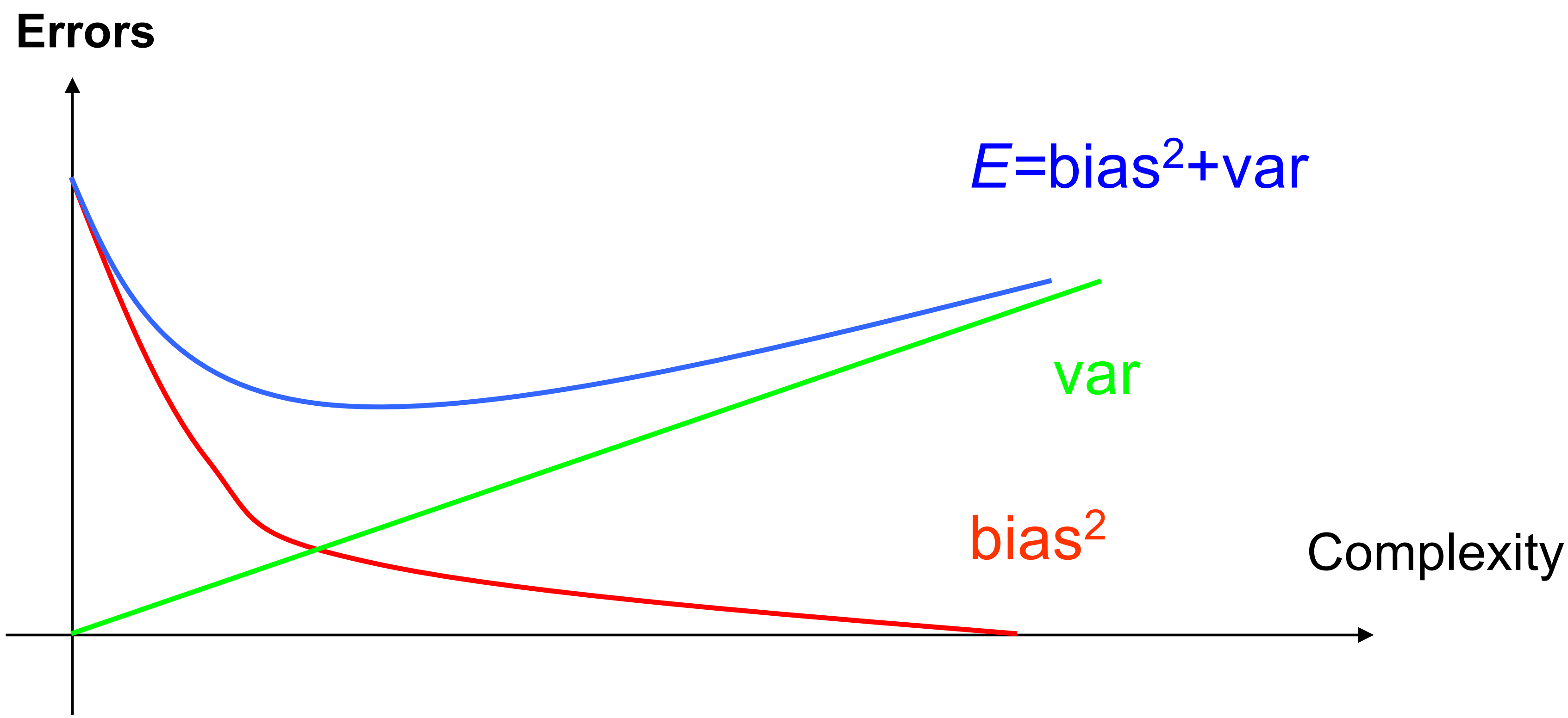


Content

- Introduction
- Decision trees
- **<u>Decision forests</u>**

Bias-Variance dilemma

- Error = Bias^2 + Variance
  - Bias: error from erroneous assumptions in the learning algorithm
  - Variance: error from sensitivity to small fluctuations in the training set



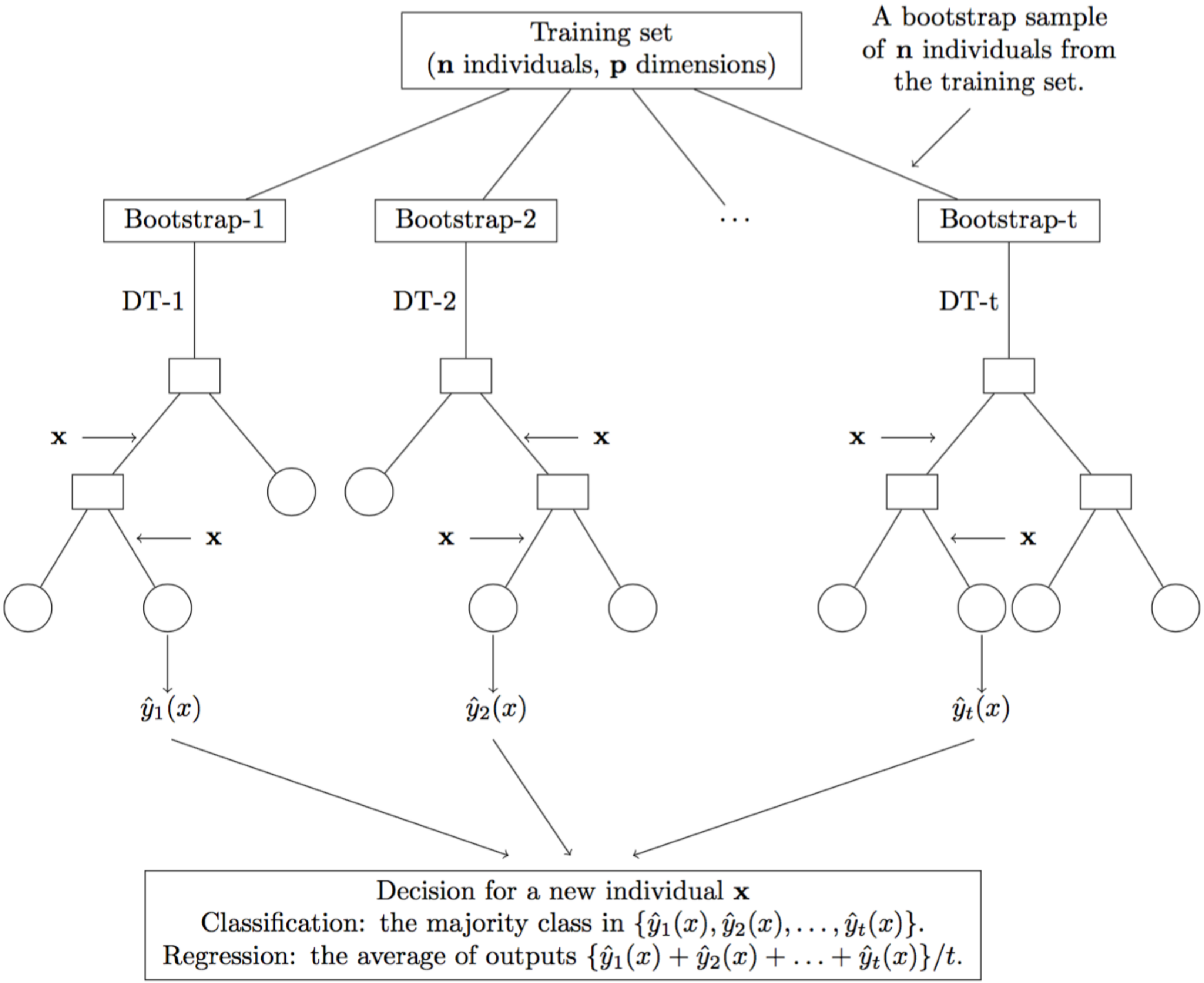
Ensemble-based learning

- Principle
  - Try to reduce bias and/or variance
  - Combine weak classifiers (not too bad) and sufficiently diverse classifiers
  - Weak classifier: decision tree, naive Bayes, etc.
  - Bagging (Breiman, 1996)
  - Boosting (Freund & Schapire, 1995), (Breiman, 1997)
  - Random forests (Breiman, 2001)

Bagging (Breiman, 1996)

- Principle
  - Try to reduce variance without increasing too much bias
  - Bagged trees: learning T decision tree models (weak classifiers) from different bootstrap samples
  - Classification: majority vote
  - Regression: average

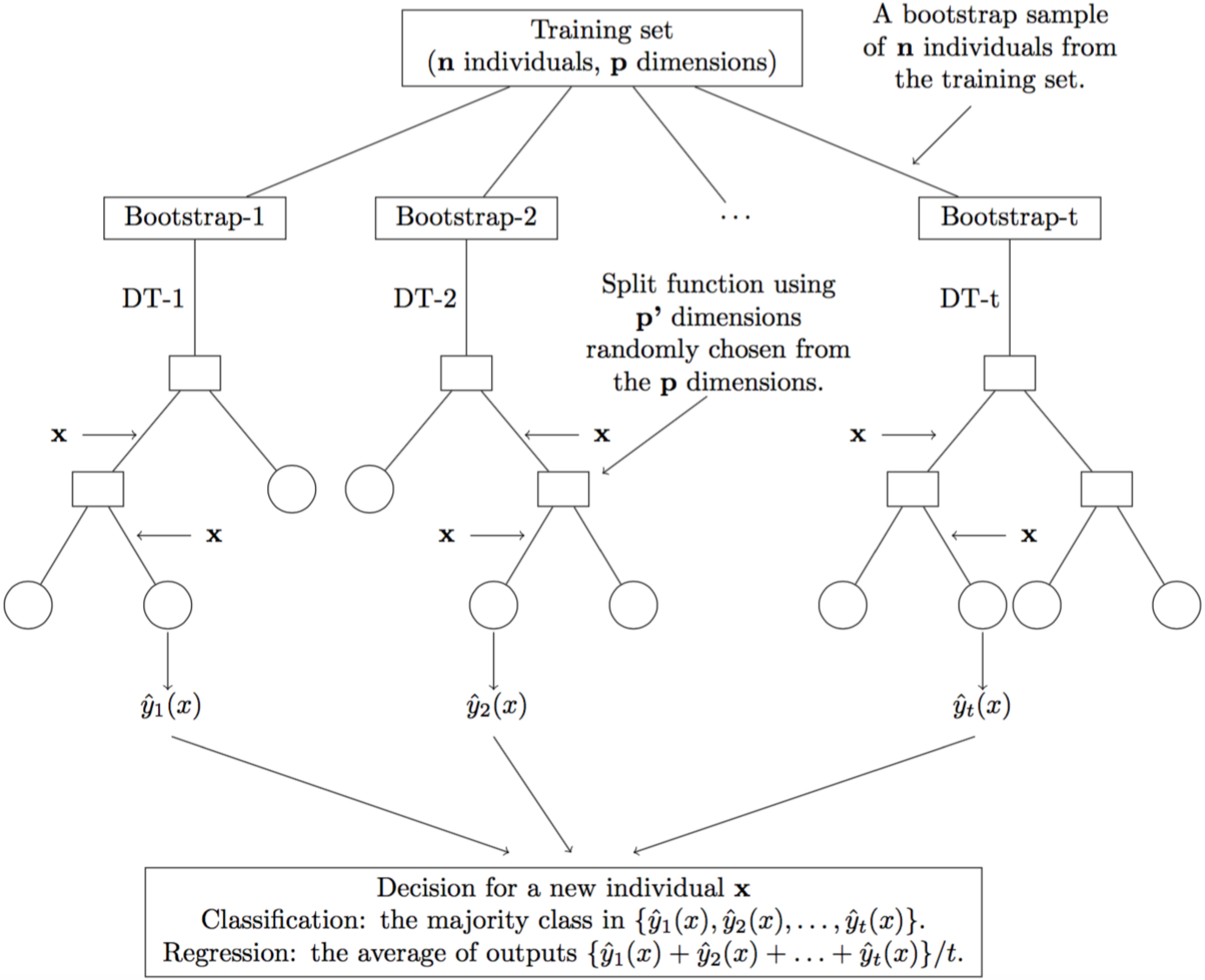
Bagging (Breiman, 1996)



Random forests (Breiman, 2001)

- Principle
  - Try to reduce variance and keep low bias
  - Random forests: learning T random and unpruned decision trees (weak classifiers) from different bootstrap samples
  - Unpruned tree (grown to maximum depth): low bias
  - Random (bootstrap, random subset of attributes for choosing the best split): high diversity
  - Classification: majority vote
  - Regression: average

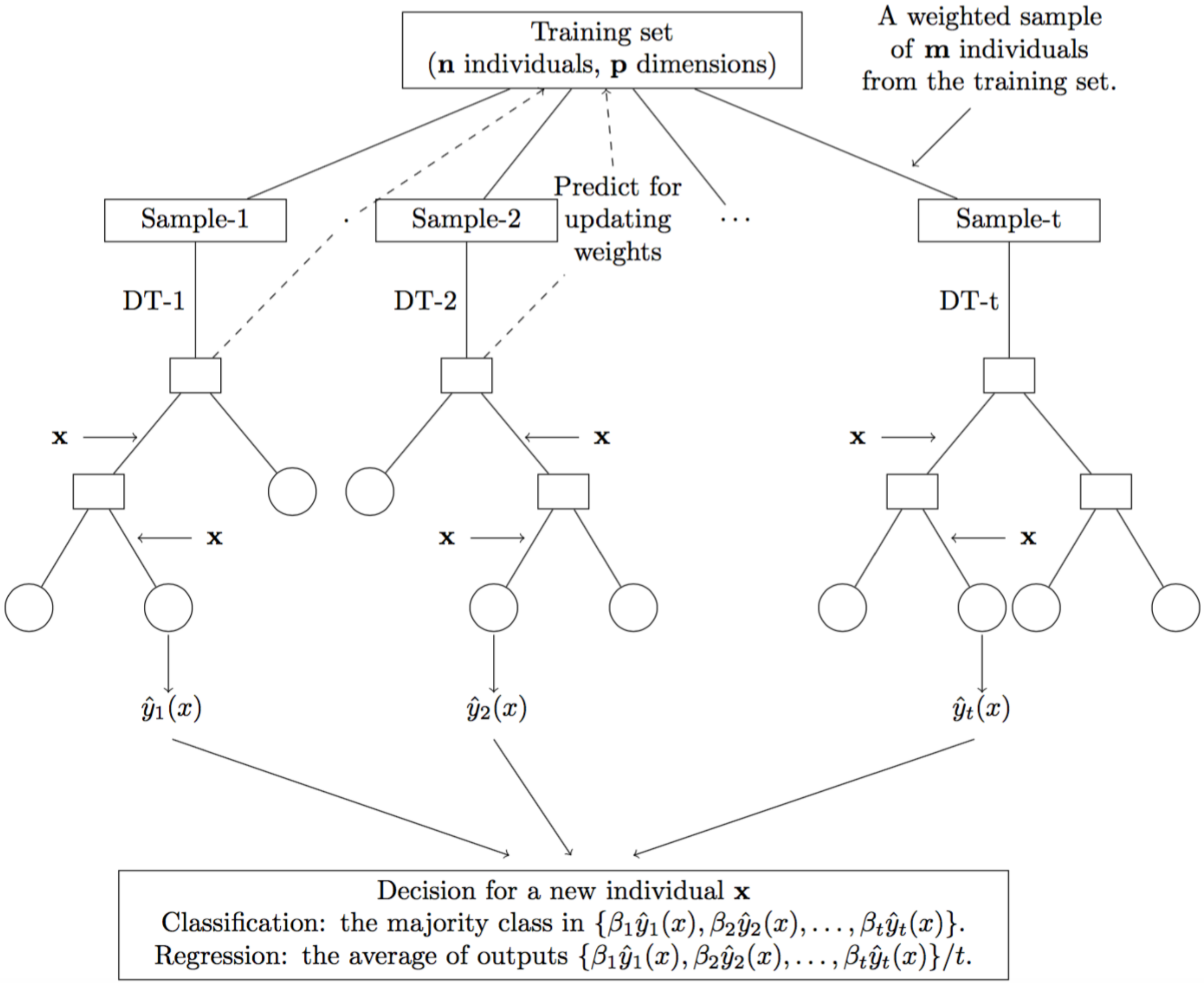
Random forests (Breiman, 2001)



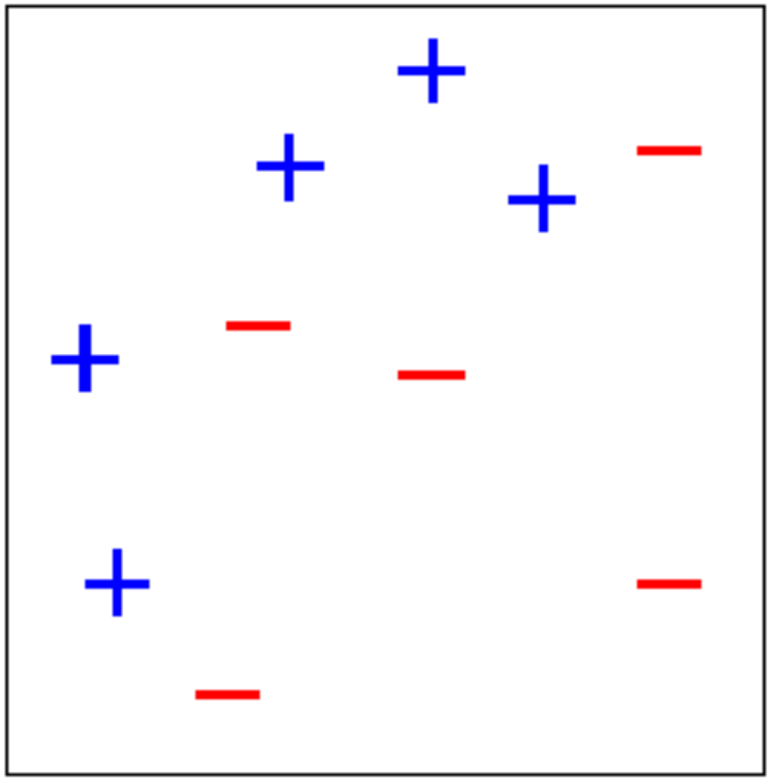
Boosting (Freund & Schapire, 1995)

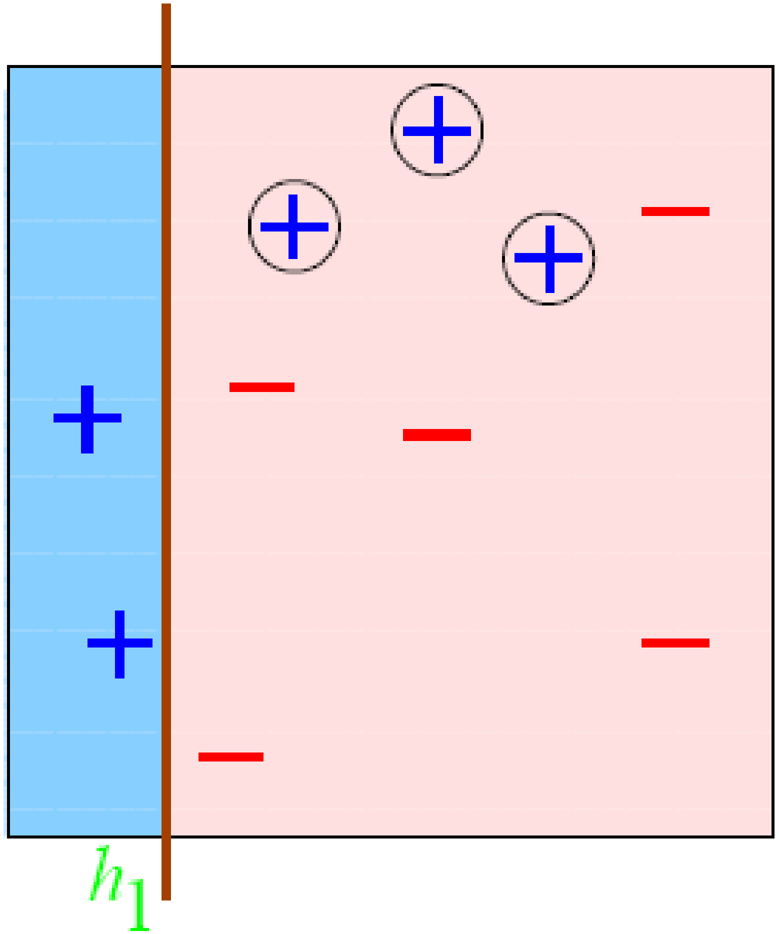
- Principle
  - Try to reduce bias and variance
  - Boosting: improving classification correctness of weak classifiers (not too bad, more accurate than random guess), for example decision stumps
  - Consecutively train *T* weak classifiers (trees) so that the i-th classifier concentrates mostly errors produced by previous ones
  - Classification: majority vote (weighted predictions)
  - Regression: average (weighted predictions)

Boosting (Freund & Schapire, 1995)

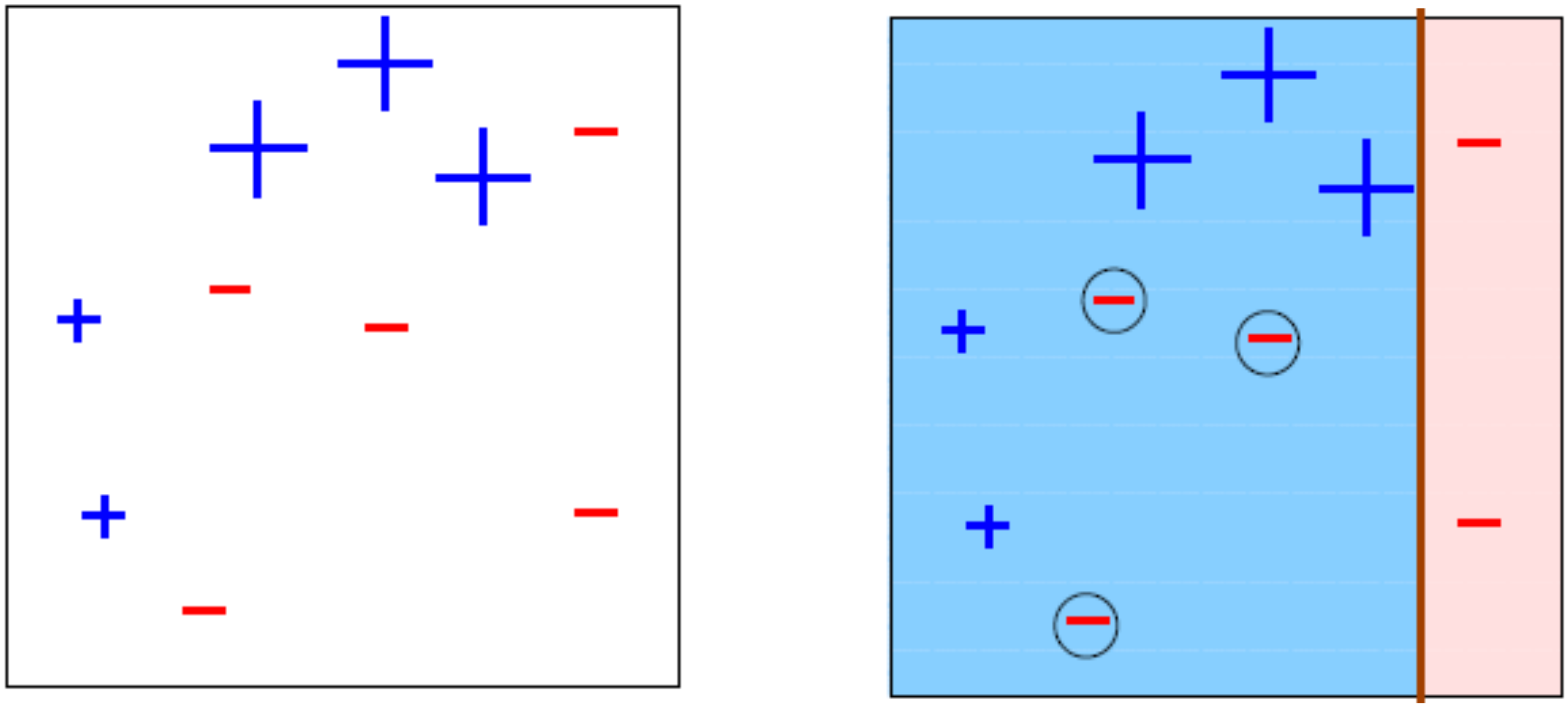


Boosting (Freund & Schapire, 1995)

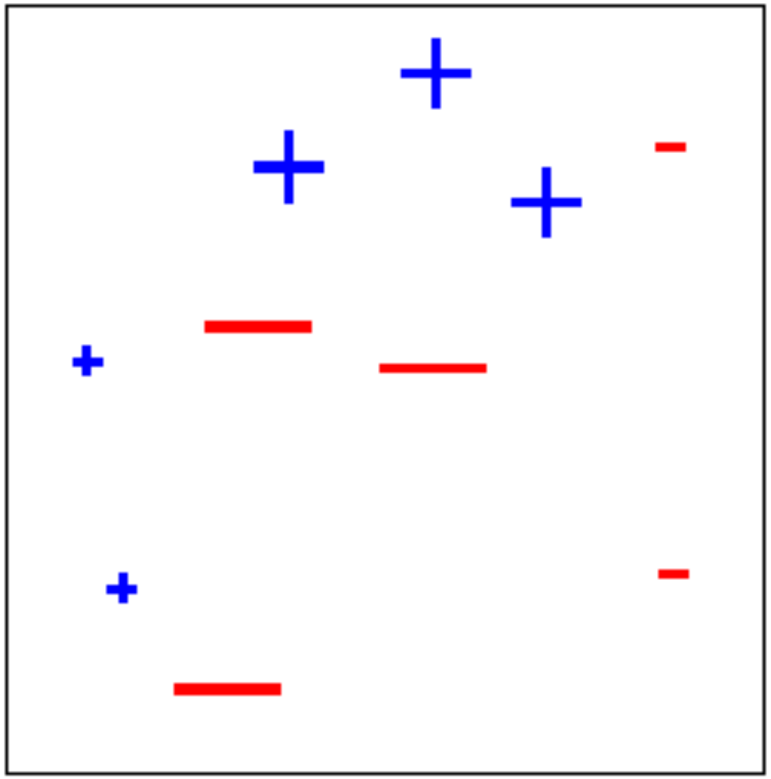


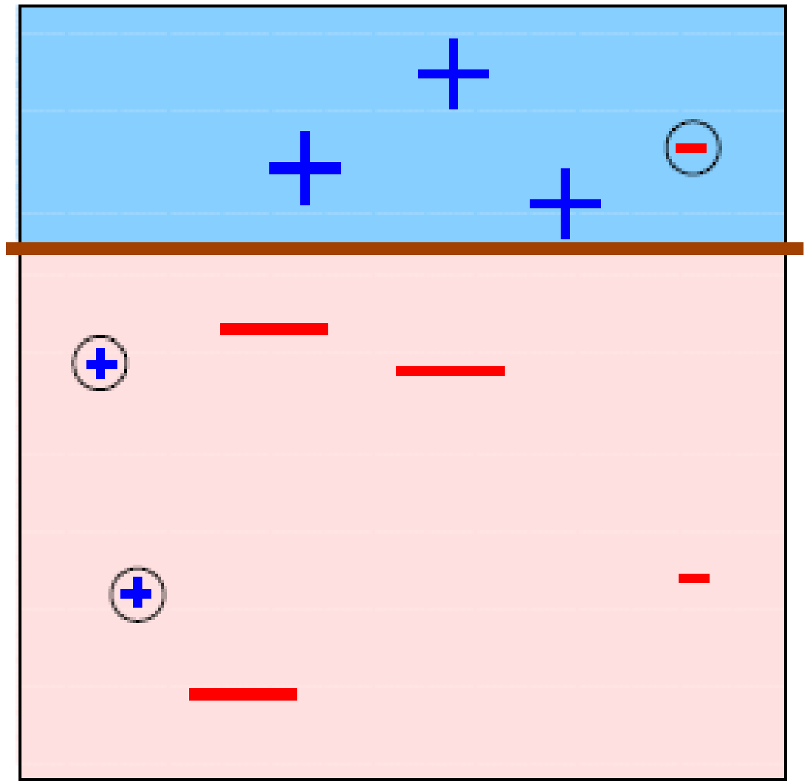


Boosting (Freund & Schapire, 1995)

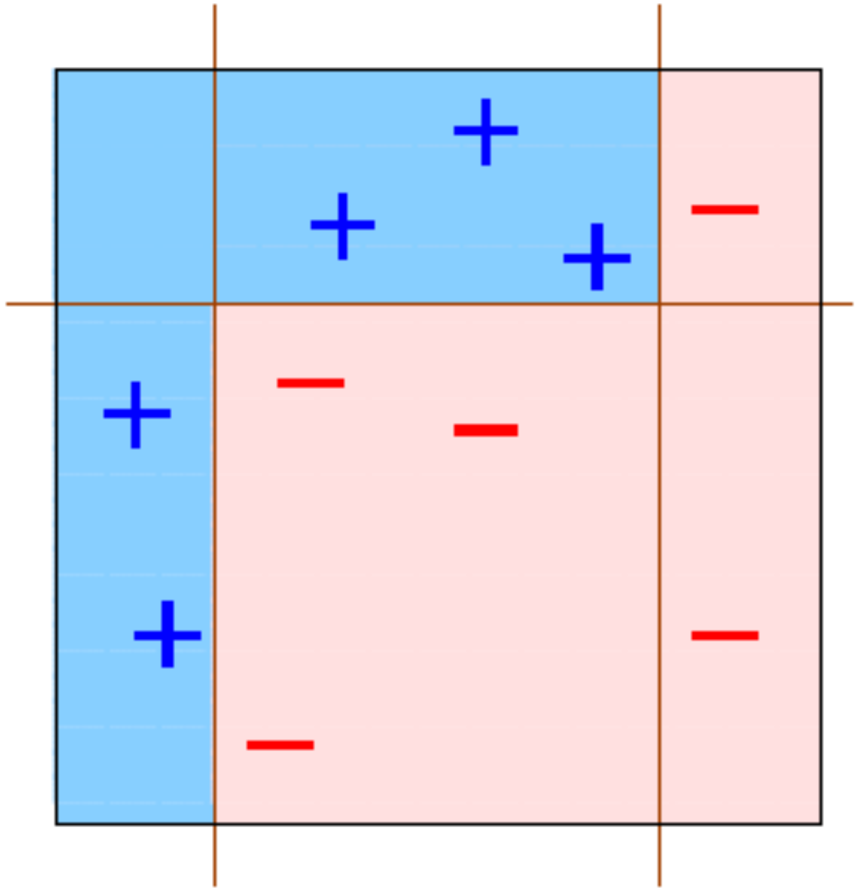


Boosting (Freund & Schapire, 1995)





Boosting (Freund & Schapire, 1995)



Comments

- Ensemble-based learning
  - Reducing bias and/or variance by combining different weak classifiers
  - Blackbox models
  - Bagging, random forests: can be parallelized
  - Boosting, random forests: most accurate

Merci !