Decision tree classifier

Hung-Hsuan Chen 陳弘軒
Computer Science and Information Engineering
National Central University
hhchen@ncu.edu.tw

Many slides are taken from Jiawei Han at UIUC

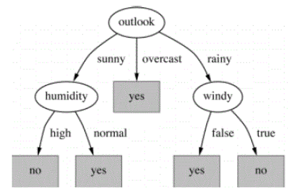
Decision Tree (1/2)

- Training set

feature vector (x_i) y_i: +1:Yes, -1: No

Outlook	Temperature	Humidity	Windy	Play
Sunny	Hot	High	False	No
Sunny	Hot	High	True	No
Overcast	Hot	High	False	Yes
Rainy	Mild	Normal	False	Yes
…	…	…	…	…

- Learned decision tree



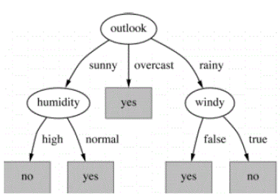
[Note]
Only one feature will be involved at a node

Decision Tree (2/2)

- Example:

(A test instance)

Outlook	Temperature	Humidity	Windy	Play
Rainy	Hot	High	True	?

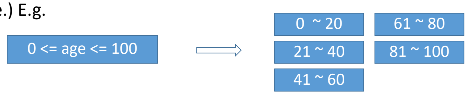


Ans: no

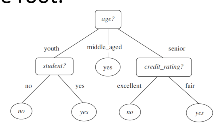
How to generate a classification tree?

Algorithm for Decision Tree Induction

- Basic algorithm (a **greedy algorithm**)
  - Tree is constructed in a top-down recursive divide-and-conquer manner.
  - Attributes are categorical.
    (if an attribute is a continuous number, it needs to be discretized in advance.) E.g.



  - At start, all the training examples are at the root.
  - Examples are partitioned recursively based on selected attributes.

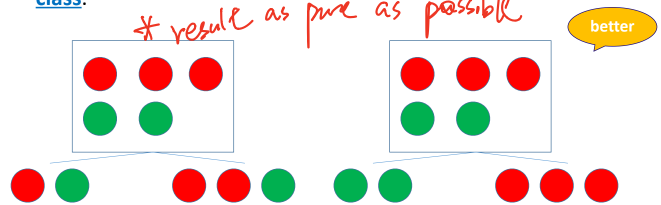


* which Q be first?
question

Algorithm for Decision Tree Induction

- Basic algorithm (a **greedy algorithm**)
  - Test attributes are selected on the basis of a heuristic or statistical measure (e.g., information gain): maximizing an information gain measure, i.e., **favoring the partitioning which makes the majority of examples belong to a single class**.

* result as pure as possible



Algorithm for Decision Tree Induction

- Basic algorithm (a **greedy algorithm**)
  - Test attributes are selected on the basis of a heuristic or statistical measure (e.g., information gain): maximizing an information gain measure, i.e., **favoring the partitioning which makes the majority of examples belong to a single class**.
- Examples of conditions for stopping partitioning:
  - All samples for a given node belong to the same class
  - There are no remaining attributes for further partitioning – majority voting is employed for classifying the leaf
  - There are no samples left

Primary Issues in Tree Construction (1/2)

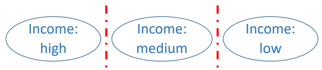
(多 pure)

- **Split criterion**: *Goodness function*
  - Used to select the attribute to be split at a tree node during the tree generation phase
  - Different algorithms may use different goodness functions:
    - Information gain (used in ID3)
    - Gain ratio (used in C4.5)
    - Gini index (used in CART)

not only

Primary Issues in Tree Construction (2/2)

- **Branching scheme**:
  - Determining the tree branch to which a sample belongs
  - Binary vs. *k-ary* splitting



- When to stop the further splitting of a node? e.g. impurity measure
- Labeling rule: a node is labeled as the class to which most samples at the node belongs.

How to Use a Tree?

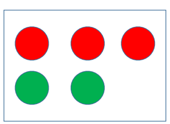
- Directly
  - Test the attribute value of unknown sample against the tree.
  - A path is traced from root to a leaf which holds the label.
- Indirectly
  - Decision tree is converted to classification rules.
  - One rule is created for each path from the root to a leaf.
  - **IF-THEN** might be easier for humans to understand .

Expected Information (Entropy)

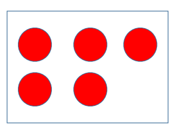
- Expected information (entropy) needed to classify a tuple in D:

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

(p_i: probability that a tuple in D belongs to class C_i, m: number of classes)



$$Info(D) = I(3,2) = -\frac{3}{5}\log_2(\frac{3}{5}) - \frac{2}{5}\log_2(\frac{2}{5})$$
$$\approx -\frac{3}{5}\times(-0.737) - \frac{2}{5}\times(-1.322)$$
$$\approx 0.971$$

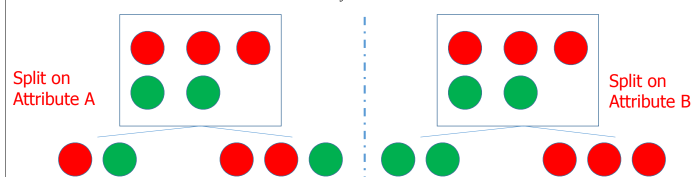


$$Info(D) = I(5,0) = -\frac{5}{5}\log_2(\frac{5}{5}) - \frac{0}{5}\log_2(\frac{0}{5})$$
$$= 0 - 0 = 0$$

Expected Information (Entropy)

- Information needed (after using A to split D into v partitions) to classify D:

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

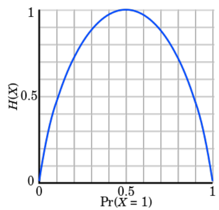


$$Info_A(D) = \frac{2}{5} Info(1,1) + \frac{3}{5} Info(2,1) = 0.918$$
$$Gain(A) = Info(D) - Info_A(D)$$
$$= 0.971 - 0.918 = 0.053$$

$$Info_B(D) = \frac{2}{5} Info(2,0) + \frac{3}{5} Info(3,0) = 0$$
$$Gain(B) = Info(D) - Info_B(D)$$
$$= 0.971 - 0 = 0.971$$

Expected Information (Entropy)

- Entropy is a measurement of uncertainty (or randomness, untidiness)
- Entropy H(X) of a coin flip
  - X: the probability of getting a head
- If the coin is fair, then entropy of the next flip is maximized
  - This is the situation of maximum uncertainty, since it is most difficult to predict the outcome
- If the coin is unfair, there is less uncertainty
  - One side is more likely to come up than the other
- Extreme case: a double-headed or a double-tailed coin
  - There is no uncertainty
  - The entropy is zero



Attribute Selection Measure: Information Gain (ID3)

- Select the attribute with the highest information gain
  - To minimize # of tests needed to classify a given tuple
- Let p_i be the probability that an arbitrary tuple in D belongs to class C_i; p_i is estimated by $|C_{i,D}|/|D|$
- Expected information (entropy) needed to classify a tuple in D:

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

- Information needed (after using A to split D into v partitions) to classify D:

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

- Information gained by branching on attribute A

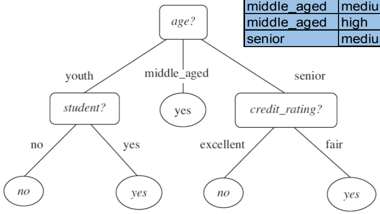
We'd like to maximize Gain(A), i.e., to minimize $Info_A(D)$, the information still required to finish classifying the tuples

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

Decision Tree Induction: An Example

- Training data set: Buys_computer
- Resulting tree:

age	income	student	credit_rating	buys_computer
youth	high	no	fair	no
youth	high	no	excellent	no
middle_aged	high	no	fair	yes
senior	medium	no	fair	yes
senior	low	yes	fair	yes
senior	low	yes	excellent	no
middle_aged	low	yes	excellent	yes
youth	medium	no	fair	no
youth	low	yes	fair	yes
senior	medium	yes	fair	yes
youth	medium	yes	excellent	yes
middle_aged	medium	no	excellent	yes
middle_aged	high	yes	fair	yes
senior	medium	no	excellent	no



Attribute Selection: Information Gain

ID
1
2
3
14

age	income	student	credit_rating	buys_computer
youth	high	no	fair	no
youth	high	no	excellent	no
middle_aged	high	no	fair	yes
senior	medium	no	fair	yes
senior	low	yes	fair	yes
senior	low	yes	excellent	no
middle_aged	low	yes	excellent	yes
youth	medium	no	fair	no
youth	low	yes	fair	yes
senior	medium	yes	fair	yes
youth	medium	yes	excellent	yes
middle_aged	medium	no	excellent	yes
middle_aged	high	yes	fair	yes
senior	medium	no	excellent	no

- Class P: buys_computer = "yes"
- Class N: buys_computer = "no"

yes

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

no

Attribute Selection: Information Gain

age	p_i	n_i	$I(p_i, n_i)$
youth	2	3	0.971
middle_aged	4	0	0
senior	3	2	0.971

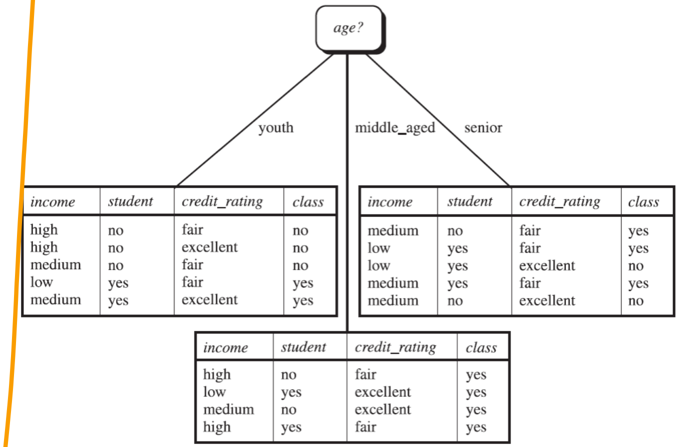
- Class P: buys_computer = "yes"
- Class N: buys_computer = "no"

- $Info_{age}(D) = \frac{5}{14}I(2,3) + \frac{4}{14}I(4,0) + \frac{5}{14}I(3,2) = 0.694$
- $\frac{5}{14}I(2,3)$: age="youth" appears in 5 out of 14 samples, with 2 positive and 3 negative examples
- $Gain(age) = Info(D) - Info_{age}(D) = 0.246$
- Similarly, we can get
- $Gain(income) = 0.029$
- $Gain(student) = 0.151$

比較如果用其它分的 gain

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

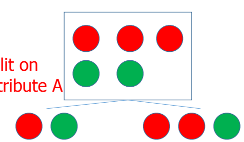
income	student	credit_rating	class
high	no	fair	yes
low	yes	excellent	yes
medium	no	excellent	yes
high	yes	fair	yes

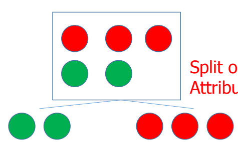
income	student	credit_rating	class
medium	no	fair	yes
low	yes	fair	yes
low	yes	excellent	no
medium	yes	fair	yes
medium	no	excellent	no

Quiz

- Which of the following has a higher entropy?
  - (O,O,X,X) vs (O,O,X,X,X) vs (O,O,O)
- Which attribute (A or B) will be selected by a decision tree classifier based on information gain?

Split on Attribute A





Split on Attribute B

※ 若用 Info gain 分類, feature 如有 ID 会很没用

Gain Ratio for Attribute Selection (C4.5)

- Information gain measure is biased towards attributes with a large number of values
  - E.g., unique pID -> split on pID results in large number of partitions, each containing just one tuple => each partition is pure => information required to classify this partition would be ***$Info_{pID}(D)=0$***, i.e., the information gain is maximal!!
- C4.5 (a successor of ID3) uses gain ratio to overcome the problem (normalization to information gain)

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

, If A has many 不同 value, splitinfo tend to high

  - GainRatio(A) = Gain(A)/SplitInfo(A)
- The attribute with the maximum gain ratio is selected as the splitting attribute

Example of Gain Ratio

age	income	student	credit_rating	buys_computer
youth	high	no	fair	no
youth	high	no	excellent	no
middle_aged	high	no	fair	yes
senior	medium	no	fair	yes
senior	low	yes	fair	yes
senior	low	yes	excellent	no
middle_aged	low	yes	excellent	yes
youth	medium	no	fair	no
youth	low	yes	fair	yes
senior	medium	yes	fair	yes
youth	medium	yes	excellent	yes
middle_aged	medium	no	excellent	yes
middle_aged	high	yes	fair	yes
senior	medium	no	excellent	no

income=high: 4
income=medium: 6
income=low: 4

- $SplitInfo_{income}(D) = -\frac{4}{14}\log_2\left(\frac{4}{14}\right) - \frac{6}{14}\log_2\left(\frac{6}{14}\right) - \frac{4}{14}\log_2\left(\frac{4}{14}\right) \approx 1.557$
- $GainRatio(income) = Gain(income)/SplitInfo_A(D) = \frac{0.029}{1.557} \approx 0.019$

* If ID, $SplitInfo_{ID}(D) = \sum_{i=1}^{14} \frac{1}{14} \log 14 \rightarrow GainRatio = \frac{0.029}{\log 14}$ 很低

Quiz

- Given students' ID, height, weight, and gender as the training data, you are asked to build a decision tree classifier to predict a student's gender based on her/his ID, height, and weight
  - Which attribute (ID, height, or weight) is likely to be selected first if you use information gain as the attribute selection method?

22

Gini Index (CART, IBM IntelligentMiner)

expect higher　　　lower

- If a data set D contains examples from n classes, impurity measure is calculated by Gini index, $Gini(D)$

ex: $p_1 = p_2 = \frac{1}{2}$　　$p_1 = \frac{1}{4}$ $p_2 = \frac{3}{4}$

$1-(\frac{1}{2})^2-(\frac{1}{2})^2 = \frac{1}{2}$　　$1-(\frac{1}{4})^2-(\frac{3}{4})^2 = \frac{6}{16}$

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

where p_i is the probability of class C_i in D, estimated by $|C_{i,D}|/|D|$

- If a data set D is split on A into subsets D_i, the *Gini* index $Gini_A(D)$ given the split on A is:

$$Gini_A(D) = \sum_i \frac{|D_i|}{|D|} Gini(D_i)$$

- Reduction in Impurity:

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

- The attribute provides the largest reduction in impurity, i.e., maximized $\Delta Gini(A)$, is chosen to split the node (*need to enumerate all the possible cut-point for each attribute*)

Example of Gini index

age	income	student	credit_rating	buys_computer
youth	high	no	fair	no
youth	high	no	excellent	no
middle_aged	high	no	fair	yes
senior	medium	no	fair	yes
senior	low	yes	fair	yes
senior	low	yes	excellent	no
middle_aged	low	yes	excellent	yes
youth	medium	no	fair	no
youth	low	yes	fair	yes
senior	medium	yes	fair	yes
youth	medium	yes	excellent	yes
middle_aged	medium	no	excellent	yes
middle_aged	high	yes	fair	yes
senior	medium	no	excellent	no

- D has **9** tuples in **buys_computer = "yes"** and **5** in **"no"**
- $Gini(D) = 1 - \left(\frac{9}{14}\right)^2 - \left(\frac{5}{14}\right)^2 = 0.459$

Example of Gini index

age	income	student	credit_rating	buys_computer
youth	high	no	fair	no
youth	high	no	excellent	no
middle_aged	high	no	fair	yes
senior	medium	no	fair	yes
senior	low	yes	fair	yes
senior	low	yes	excellent	no
middle_aged	low	yes	excellent	yes
youth	medium	no	fair	no
youth	low	yes	fair	yes
senior	medium	yes	fair	yes
youth	medium	yes	excellent	yes
middle_aged	medium	no	excellent	yes
middle_aged	high	yes	fair	yes
senior	medium	no	excellent	no

When income = “high”
➔ 2 “yes” and 2 “no”

When income = “medium”
➔ 4 “yes” and 2 “no”

When income = “low”
➔ 3 “yes” and 1 “no”

- $Gini_{income}(D) = \frac{4}{14}\left(1-\left(\frac{2}{4}\right)^2-\left(\frac{2}{4}\right)^2\right)+\frac{6}{14}\left(1-\left(\frac{4}{6}\right)^2-\left(\frac{2}{6}\right)^2\right)$

$$+\frac{4}{14}\left(1-\left(\frac{3}{4}\right)^2-\left(\frac{1}{4}\right)^2\right)=0.325$$

- $\Delta Gini(income) = Gini(D) - Gini_{income}(D) = 0.134$

Quiz

- Can we apply Decision Tree Classifier on the datasets with only numerical attributes?

HW

Computing Information-Gain for Continuous-Valued Attributes

- Let attribute A be a continuous-valued attribute
- Must determine the *best split point* for A
  - Sort the value A in increasing order
  - Typically, the midpoint between each pair of adjacent values is considered as a possible *split point*
    - $(a_i+a_{i+1})/2$ is the midpoint between the values of a_i and a_{i+1}
  - The point with the *minimum expected information requirement* for A is selected as the split-point for A
- Split:
  - D1 is the set of tuples in D satisfying A ≤ split-point, and D2 is the set of tuples in D satisfying A > split-point

Example of Information-Gain for Continuous-Valued Attributes

Temperature:	40	48	60	72	80	90
PlayTennis:	No	No	Yes	Yes	Yes	No

看怎分 purity 最高

- Sort the continuous-valued attributes
- Determine the classifier’s changing points
  - E.g., (48-60) and (80-90) in the above example
- Take the mid-points of the changing points as the candidates for discretization
  - E.g., (48+60)/2, (80+90)/2
- Use “54” to split
  - If Temperature <= 54 ➔ No; Else ➔ Yes
  - Gain(T=54) = Info(D) – $Info_{T=54}$(D) = 1 – 0.811 = 0.189
- Use “85” to split
  - If Temperature <= 85 ➔ Yes; Else ➔ No
  - Gain(T=85) = Info(D) – $Info_{T=85}$(D) = 1 – 0.696 = 0.304

Comparing Attribute Selection Measures

- The three measures, in general, return good results but
  - **Information gain**:
    - biased towards multivalued attributes
  - **Gain ratio**:
    - tends to prefer unbalanced splits in which one partition is much smaller than the others
  - **Gini index**:
    - biased to multivalued attributes
    - Usually faster than information gain (because information gain requires logarithm computation)

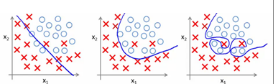
Why 很多人用 Gini? ∵ $O(\text{Gini}) = n^2 > O(\text{Gain, IG}) = \log n$?

Many Attribute Selection Measures

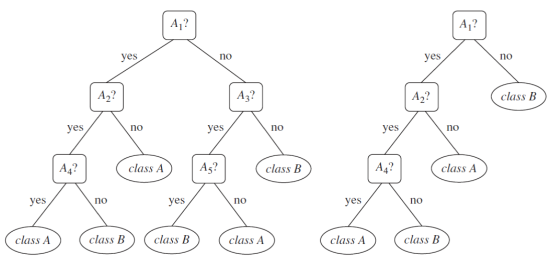
- **Which attribute selection measure is the best?**
  - **Most give good results, none is significantly superior than others**

Overfitting and Tree Pruning

- Overfitting: An induced tree may overfit the training data
  - Too many branches, some may reflect anomalies due to noise or outliers
  - Poor accuracy for unseen samples, i.e., lose the ability of generalization



underfitting overfitting



Overfitting and Tree Pruning

- Perfect decision tree performs 100% accuracy on the training data
  - Assuming that in the training data if two instances have the same feature sets then they must have the same label
- Prevent overfitting
  - Pre-prune the tree
    - Stop before a tree is fully grown
      - E.g., limit the tree height; stop when the number of instances in a node is small; when misclassification rate is low enough
    - The method is short-sighted
      - A seemly worthless early split may be followed by a very good split
  - Post-prune the tree
    - Grow a perfect tree and prune the nodes from bottom up
    - $R_\alpha(T) = R(T) + \alpha \cdot |f(T)|$

Training error # leaves

small small

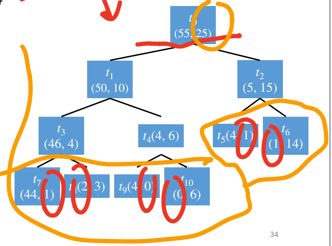
Post-pruning considerations

- Goal: cut a large portion of a tree, but only increase few error rate
- The two requests are usually against each other
- We may define the goodness of a cut by
  - $\alpha(t_i) = \dfrac{\#\text{ error after cut} - \#\text{ error before cut}}{\#\text{ leaves been cut} - 1}$ (小) (大)

(小)

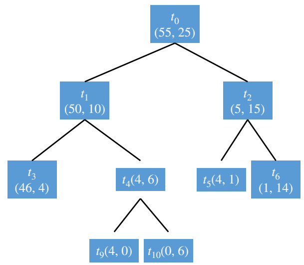
Example

- $\alpha(t_i) = \dfrac{\#\text{ error after cut} - \#\text{ error before cut}}{\#\text{ leaves been cut} - 1}$
- $\alpha(t_0) = ?$
  - If we cut the tree and leave only t_0, # errors = 25
  - If we don't cut the tree, # errors = 1+2+0+0+1+1=5
  - If we cut the tree, 6 leaves are cut
  - $\alpha(t_0) = \dfrac{25 - (1+2+0+0+1+1)}{6-1} = 4$
- By a similar manner, we may obtain
  - $\alpha(t_1) = \dfrac{10 - (1+2+0+0)}{4-1} = 2.33$
  - $\alpha(t_2) = \dfrac{5 - (1+1)}{2-1} = 3$
  - $\alpha(t_3) = \dfrac{4 - (1+2)}{2-1} = 1$ ➔ Cut here!
  - $\alpha(t_4) = \dfrac{4 - (0+0)}{2-1} = 4$



Example

- $\alpha(t_0) = \dfrac{25 - (4+0+0+1+1)}{5-1} = 4.75$
- $\alpha(t_1) = \dfrac{10 - (4+0+0)}{3-1} = 3$ ➔ Cut here!
- $\alpha(t_2) = \dfrac{5 - (1+1)}{2-1} = 3$
- $\alpha(t_4) = \dfrac{4 - (0+0)}{2-1} = 4$



Classification tree is constructed in a "greedy" manner

- Greedy: pick a feature to split the data best on the ***current*** information
  - This may lead to a local optimal

Example

A	B	C	label
1	0	1	1
0	1	0	1
0	1	1	1
0	0	0	0
1	1	1	0
…			

We build an imaginary dataset as follows

- The dataset has three binary features A, B, and C
- The label is constructed by A xor B
- C is set to label for 80% of the time and the inverse of the label for 20% of the time

On average, $Info_A(D) = 1$ entropy

- When A=1, 50% of the labels are 1 and 50% of the labels are 0
- When A=0, 50% of the labels are 1 and 50% of the labels are 0

Similarly, $Info_B(D) = 1$

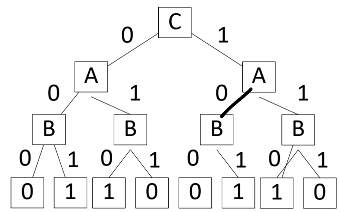
On average, $Info_C(D) = 0.722$

- When C=1, 80% of the labels are 1; 20% of the labels are 0
- When C=0, 80% of the labels are 0; 20% of the labels are 1

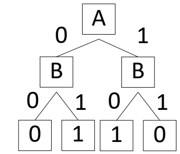
In the case, we will choose feature *C* as the feature to split data

However, an oracle should first select one feature from A or from B, and select the other feature as the second feature

Which one is better?



✓

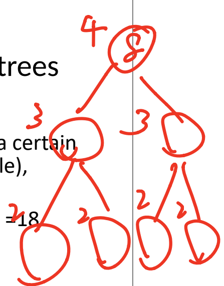


- Occam's razor
  - Among competing hypotheses, the one with the fewest assumptions should be selected.

Why not try all possible attribute splitting sequences?

- The best tree requires to test all possible sequences
  - Very large search space
  - If the training data have *d* binary features, consider one path from root to one leaf:
    - *d* possible roots
    - *d*-1 possible level-1 nodes
    - *d*-2 possible level-2 nodes
    - …
    - This is just one path from root to one leaf, there are *d* different leaves
  - If there are numerical attributes, the number of trees is even larger
    - Why?

An example of testing all trees vs decision trees

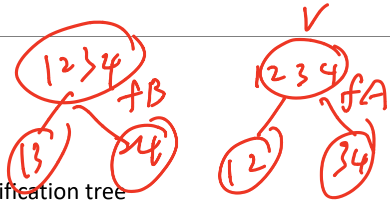


level-1

- Suppose we want to predict whether an individual has a certain disease based on 4 binary features: gender (male/female), height (tall/short), weight (fat/thin), age (young/old)
- Decision tree: need to test approximately 4 + 3*2 + 2*4 =18 splits
  - 4: root split test 4 features
  - 3*2: each level-1 node test 3 features; 2 level-1 nodes
  - 2*4: each level-1 node test 2 features; 4 level-1 nodes

level-2

- List all tress: need to test approximately 4 * 3*2 * 2*4 = 192
  - 4: root split test 4 features
  - 3*2: each level-1 node test 3 features; 2 level-1 nodes
  - 2*4: each level-1 node test 2 features; 4 level-1 nodes

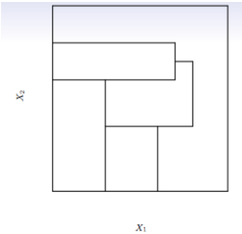


Regression tree

- A brief review of classification tree
  - Select a feature *A* and a cut-point *v* to split the original dataset *D* into sub-groups such that **the labels** in each sub-group are as pure as possible
  - Repeat the above step
- Regression tree
  - Select a feature *A* and a cut-point *v* to split the original dataset *D* into sub-groups such that **the target values** in each group is as pure as possible
    - There could be multiple ways to define "purity"
    - Possible choices: RSS (residual sum of squares), max-min, variance
  - Repeat the above step

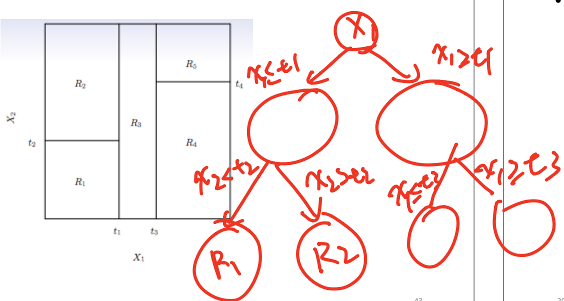
Quiz

- You are asked to build a decision tree classifier based on two features x1 and x2. How to partition the space into the following figure?



Quiz

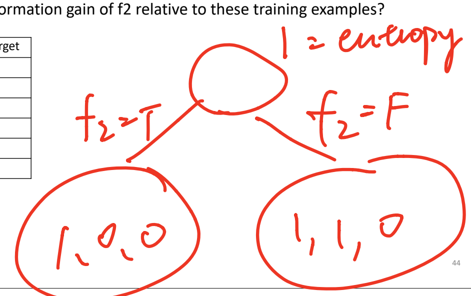
- You are asked to build a decision tree classifier based on two features x1 and x2. How to separate the space into the following figure?



Quiz

- True or false: if decision tree T2 is a pruned tree of tree T1, then T1 is less likely to overfit the training data
- Consider the following training data
  - What is the entropy of the training examples? one
  - What is the information gain of f2 relative to these training examples?

f1	f2	Target
T	T	1
T	F	1
T	T	0
F	F	1
F	T	0
F	F	0



$\frac{3}{6}\left(-\frac{1}{3}\log\frac{1}{3}-\frac{2}{3}\log\frac{2}{3}\right)+\frac{3}{6}\left(-\frac{1}{3}\log\frac{1}{3}-\frac{2}{3}\log\frac{2}{3}\right)$

Random forest

- A random forest is a **meta estimator** that fits numerous decision tree classifiers on various sub-samples of the dataset
  - Each tree can use only part of the available features
    - A common practice is using sqrt(n_features)
  - Sub-sampling the training data for each tree
    - A common practice: the sub-sample size is always the same as the original input sample size, but the samples are ***<u>drawn with replacement</u>***
- The prediction is based on the majority voting of all the generated decision trees
  - Prevent overfitting
  - Usually yield higher test accuracy
- Highly parallelizable during training and testing
- One of the best predictor in many applications and competitions

Info gain = entropy 1 − entropy 2

分前 − 分後

Summary

- Decision tree generates a set of classification/ regression rules based on the training data
- "Interpretable" prediction

* P.12

$= 0.918$

$Info_A(D) = \frac{2}{5} Info(1,1) + \frac{3}{5} Info(2,1)$ $\quad Info_B(D) = \frac{2}{5} Info(2,0) + \frac{3}{5} Info(3,0) = 0$

$Gain(A) = Info(D) - Info_A(D) = \cancel{0.918}$

$= 0.971 - 0.918 = 0.053$

$Gain(B) = Info(D) - Info_B(D)$

$= 0.971 - 0 = 0.971$

befor分 − afeer分 = Improvement of puriey