Pruning, Limitations, and Generating Decision Trees/Rules

M.M. Pedram
pedram@khu.ac.ir
Kharazmi University
(Fall 2009)

1

1. Pruning Decision Trees

- Note that decision trees are made though a *recursive-partitioning method*.
- When a decision tree is built, many of the branches will reflect anomalies in the training data due to noise or outliers. Tree pruning methods address this problem of *overfitting* the data. Such methods typically use statistical measures to remove the least reliable branches.
- Main task in decision-tree *pruning*:
  remove parts of the tree (sub-trees) that do not contribute to the classification accuracy of unseen testing samples, producing a less complex and thus more comprehensible tree.

2

1. Pruning Decision Trees

- There are 2 ways to modify the recursive-partitioning method:
  - *Prepruning* or *Stop Splitting* : Deciding not to divide a set of samples any further under some conditions. The stopping criterion is usually based on some statistical tests, such as the χ^2 test: *If there are no significant differences in classification accuracy before and after division, then represent a current node as a leaf*. The decision is made in advance, before splitting, and therefore this approach is called *prepruning*.
  - *Postpruning*: Removing retrospectively some of the tree structure using selected accuracy criteria. The decision in this process of *postpruning* is made after the tree has been built.
- *postpruning* is more common approach.

3

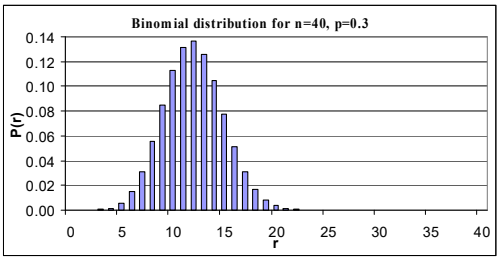
1. Pruning Decision Trees

- C4.5 follows the *postpruning* approach, but it uses a specific technique to estimate the predicted error rate. This method is called *pessimistic pruning*.
- *pessimistic pruning* : For every node in a tree, the estimation of the upper confidence limit U_{cf} is computed using the statistical tables for binomial distribution, i.e., it uses the training set to estimate (predict) error rates.
  - Parameter U_{cf} is a function of $|D_i|$ and E for a given node. C4.5 uses the default confidence level of 25% (or α=0.25).
  - compares $U_{25\%}$ (predicted $E/|D_i|$) for a given node D_i with a weighted confidence of its leaves. Weights are the total number of cases for every leaf. If the predicted error for a root node in a sub-tree is less than weighted sum of $U_{25\%}$ for the leaves (predicted error for the sub-tree), then a sub-tree will be replaced with its root node, which becomes a new leaf in a pruned tree.

4

Remember #1

- Binomial Probability Distribution



$$P(R=r)=\frac{n!}{r!(n-r)!}p^r(1-p)^{n-r}$$

Probability P(r) of r heads(errors)in n coin flips, if p = probability(*heads*)

$$E[X]\equiv\sum_{i=0}^{n} i\cdot P(i)=np$$

$$Var(X)\equiv E[(X-E[X])^2]=np(1-p)$$

$$\sigma_X\equiv\sqrt{E[(X-E[X])^2]}=\sqrt{np(1-p)}$$

5

Remember #1

We want to estimate: p = probability(*heads*)

Estimator: $Y = r/n$

Estimation Bias: $E(Y) - p\ (=0)$?

Note: The variance in this estimate arises from the variance in r, because n is a constant. Because r is Binomially distributed, its mean and variance are given by np and $np(1-p)$.

$$E(Y) = (1/n)\,.\,E(r)$$
$$= (1/n)\,.\,np = p$$

⇒ Y is an unbiased estimator (with binomial distribution)

Standard deviation: $\sigma_Y = \sigma_r/n = [np(1-p)]^{1/2}/n$

6

Normal Approximation to Binomial

- Y, as our estimator, follows a Binomial distribution, with

$$\mu_Y = p$$

$$\sigma_Y = \sqrt{\frac{p(1-p)}{n}}$$

Note that p is unknown, we can substitute $Y = r/n$ for it, i.e.

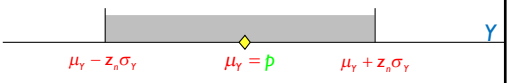
$$\mu_Y = p$$

$$\sigma_Y \approx \sqrt{\frac{Y(1-Y)}{n}}$$

Thus:

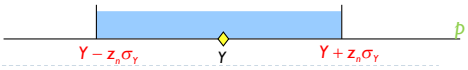
$$\mu_Y - z_n\sigma_Y \le Y \le \mu_Y + z_n\sigma_Y$$

$$p - z_n\sigma_Y \le Y \le p + z_n\sigma_Y$$



or

$$Y - z_n\sigma_Y \le p \le Y + z_n\sigma_Y$$



7

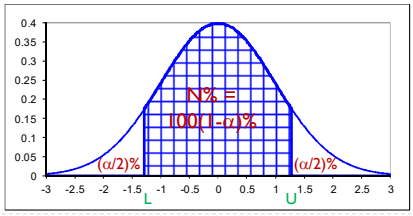
Remember #2

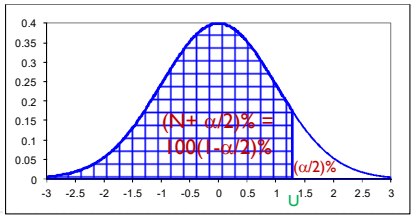
Two-Sided and One-Sided Bounds

A N% = 100(1-α)% confidence interval with lower bound ***L*** and upper bound ***U***, implies a (N+ α/2)% = 100(1- α/2)% confidence interval with lower bound *L* and no upper bound, or with upper bound *U* and no lower bound.

N% of area (probability) lies in $\mu \pm z_N\sigma$

N% :	50%	68%	80%	90%	95%	98%	99%
z_N :	0.67	1.00	1.28	1.64	1.96	2.33	2.53





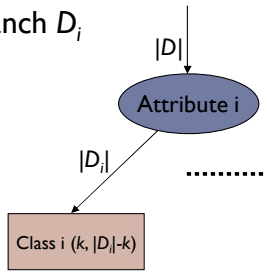
8

1. Pruning Decision Trees

- $p_{err,\max}$: (predicted) max error probability in branch D_i

$U_{cf} = p_{err,\max}$

$\alpha = 0.25$



Binomial Distribution:

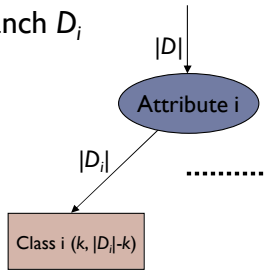
$$P(incorrect\_classification = |D_i| - k) = \binom{|D_i|}{|D_i| - k} p_{err,\max}^{|D_i|-k} (1 - p_{err,\max})^k$$

$$P(correct\_classification = k) = \binom{|D_i|}{k} (1 - p_{err,\max})^k p_{err,\max}^{|D_i|-k}$$

9

1. Pruning Decision Trees

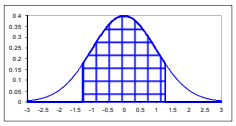
- $p_{err,\max}$: (predicted) max error probability in branch D_i



- Normal Distribution Approximation:

$$U_{cf} = U_{0.25} \Rightarrow 0.25 = \alpha$$

$$\Rightarrow z_N = 1.150 = z_{\alpha/2}$$

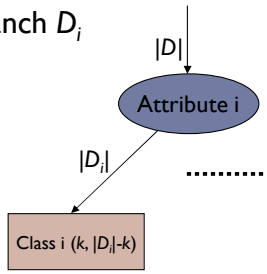


Confidence level N%	50%	68%	75%	80%	90%	95%	98%	99%
Constant z_N	0.67	1.00	1.15	1.28	1.64	1.96	2.33	2.58

10

1. Pruning Decision Trees

- $p_{err,\max}$: (predicted) max error probability in branch D_i

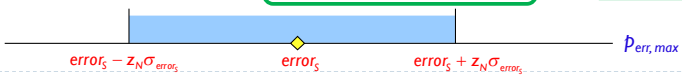


- Normal Distribution Approximation:

$$U_{cf} = U_{0.25} \Rightarrow 0.25 = \alpha$$
$$\Rightarrow z_N = 1.150 = z_{\alpha/2}$$

$$error_S - z_N\sigma_{error_S} \le p_{err,\max} \le error_S + z_N\sigma_{error_S}$$

the upper bound of this as our error rate estimate, i.e., U_{cf}



11

Parameter estimation

Reminder

- *Additive smoothing*, also called *Laplace smoothing* or *Lidstone smoothing*, is a technique used to smooth categorical data.
- Given an observation **x** = (x_1, …, $x_{no\_of\_classes}$) from a multinomial distribution with *n* trials and parameter vector

$$\theta = (\theta_1, \ldots, \theta_{no\text{-}of\text{-}classes})$$

a "smoothed" version of the data gives the estimator:

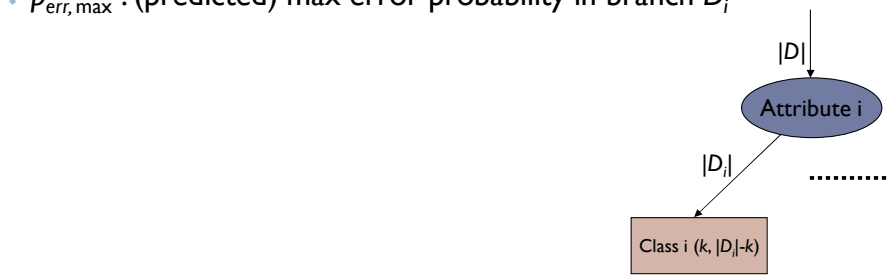
$$\hat{\theta}_i = \frac{x_i + \lambda}{n + \lambda .\ no\_of\_classes}$$

where $\lambda > 0$ is the smoothing parameter. $\lambda = 0$ corresponds to no smoothing. as the resulting estimate will be between the empirical estimate x_i/n (for $\lambda \to 0$), and the uniform probability $1/_{no\text{-}of\text{-}classes}$ (for $\lambda \to +\infty$).

12

1. Pruning Decision Trees

- $p_{err,\max}$: (predicted) max error probability in branch D_i



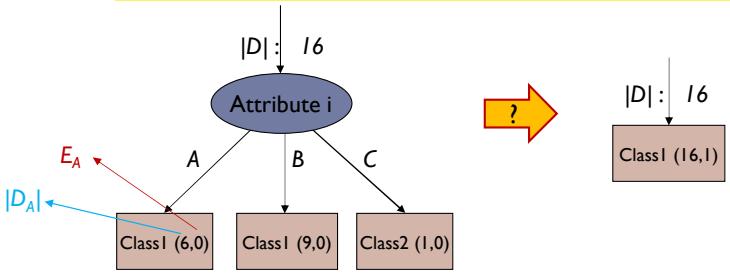
$$error_S = \frac{E_i}{|D_i|} \quad \text{smoothing} \quad error_S = \frac{E_i + 1}{|D_i| + 2}$$

$$\sigma_{error_S(h)} \approx \sqrt{\frac{error_S(h)(1 - error_S(h))}{n}} \quad \sigma_{error_S(h)} \approx \sqrt{\frac{error_S(h)(1 - error_S(h))}{|D_i| + 2}}$$

13

1. Pruning Decision Trees

Example: Analyze the possibility of replacing the following sub-tree with a leaf node shown below. (confidence of 25%)



- It is necessary to compute a predicted error PE for the initial sub-tree and for a replaced node. Thus, first we compute the followings :

$U_{25\%}(6, 0) = ?$ $U_{25\%}(9, 0) = ?$ $U_{25\%}(1, 0) = ?$

$U_{25\%}(16, 1) = ?$

14

1. Pruning Decision Trees

In a similar way, other upper confidence limits will be computed:

$U_{25\%}(6, 0) = 0.259$, $U_{25\%}(9, 0) = 0.191$, $U_{25\%}(1, 0) = 0.646$, and $U_{25\%}(16, 1) = 0.196$

(Note: values are approximated by normal distribution)

predicted errors for the initial sub-tree and the leaf node are:

$$PE_{sub\text{-}tree} = 6 \times 0.259 + 9 \times 0.191 + 1 \times 0.646 = 3.919$$

$$PE_{node} = 16 \times 0.196 = 3.136$$

Since the existing sub-tree has a higher value of predicted error than the replaced node, it is recommended that the decision tree be pruned and the sub-tree replaced with the leaf node.

15

2. Features of Decision Trees/Rules

Advantages of Decision Tries and Decision Rules:

- They are relatively simple, readable,
- Their generation is very fast,
- Unlike many statistical approaches, a logical approach does not depend on assumptions about distribution of attribute values or independence of attributes,
- This method tends to be more robust across tasks than most other statistical methods,
- Works with mixed data types,
- Models non-linear functions,
- Handles classification and regression,
- Many successful applications.

16

2. Features of Decision Trees/Rules

- If data samples are represented graphically in an N-dimensional space, where N is the number of attributes, then a logical classifier (decision trees or decision rules) divides the space into regions. Each region is labeled with a corresponding class. An unseen testing sample is then classified by determining the region into which the given point falls. Decision trees are constructed by successive refinement, splitting existing regions into smaller ones that contain highly concentrated points of one class.
- The number of training cases needed to construct a good classifier is proportional to the number of regions.
- More complex classifications require more regions that are described with more rules and a tree with higher complexity. All that will require an additional number of training samples to obtain a successful classification.

17

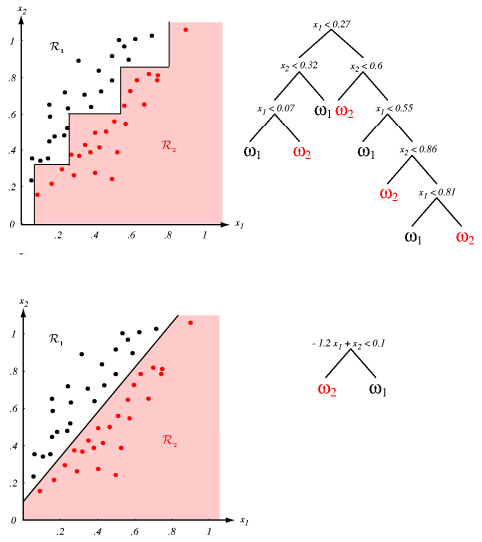
2. Features of Decision Trees/Rules

Disadvantages of Decision Tress and Decision Rules:

- They are based on heuristic search which are *greedy* in nature and thus are sensitive to local minima,
- A logical approach based on decision rules tries to approximate non-orthogonal, and sometimes, nonlinear classification with hyperrectangles; classification becomes extremely complex with large number of rules and a still larger error.

18

2. Features of Decision Trees/Rules



19

2. Features of Decision Trees/Rules

- The other type of classification problems, where decision rules are not the appropriate tool for modeling, have classification criteria in the form: *A given class is supported if n out of m conditions are present.* To represent this classifier with rules, it would be necessary to define $\binom{m}{n}$ regions only for one class. Medical diagnostic decisions are a typical example of this kind of classification. If 4 out of 11 symptoms support diagnosis of a given disease, then the corresponding classifier will generate 330 regions in an 11-dimensional space for positive diagnosis only. That corresponds to 330 decision rules.

20

2. Features of Decision Trees/Rules

- A possible solution: returning to the beginning of preprocessing phases, it is necessary to transform input features into new dimensions that are linear (or nonlinear) combinations of initial inputs. This transformation is based on some domain heuristics and requires emphasis on and effort in the data-preparation process; the reward is a simpler classification model with a lower error rate.

Example: A classification problem is described by nine binary inputs $\{A_1, A_2, \ldots, A_9\}$, and the output class C:

$$(A_1 \vee A_2 \vee A_3) \wedge (A_4 \vee A_5 \vee A_6) \wedge (A_7 \vee A_8 \vee A_9) \rightarrow C$$

The conjunctive form of above rule, will have 27 factors with only $\wedge$ operations:

$$((A_1 \wedge A_4 \wedge A_7) \vee (A_1 \wedge A_5 \wedge A_7) \vee \ldots) \rightarrow C$$

2. Features of Decision Trees/Rules

If new attributes are introduced:

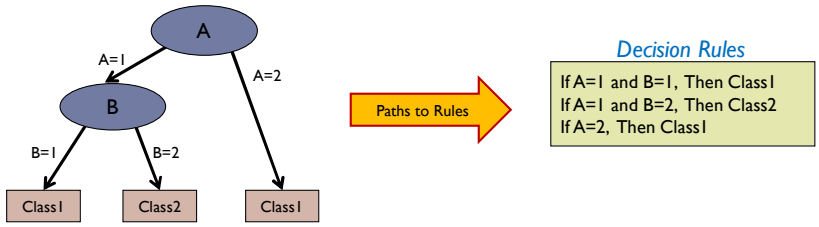
$$B_1 = A_1 \vee A_2 \vee A_3$$
$$B_2 = A_4 \vee A_5 \vee A_6$$
$$B_3 = A_7 \vee A_8 \vee A_9$$

the description of class C will be simplified into the logical rule

$$B_1 \wedge B_2 \wedge B_3 \rightarrow C$$

3. Generating Decision Rules

- Even though the pruned trees are more compact than the originals, they can still be very complex.
- A path to each leaf can be transformed into an IF-THEN production rule. The IF part consists of all tests on a path, and the THEN part is a final classification. Rules in this form are called *decision rules*.
- A collection of decision rules for all leaf nodes would classify samples exactly as the tree does.



23

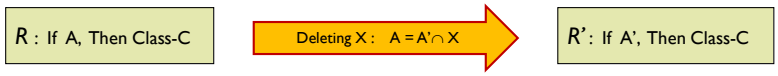
3. Generating Decision Rules

- The ways of reducing the complexity of decision rules:
  1. The antecedents of individual rules may contain irrelevant conditions, and the rules can be generalized by deleting these superfluous conditions without affecting rule-set accuracy.
  2. The other way is a process of grouping attribute values for categorical data to avoid extremely complex models.

24

3.1. Generalizing rules by deleting some conditions

- Consider rule R and a *more general* rule R′:



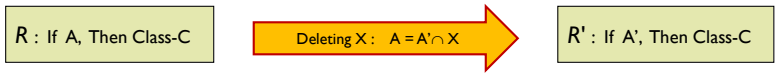
- Question: *What are criteria for deletion of rule conditions?*

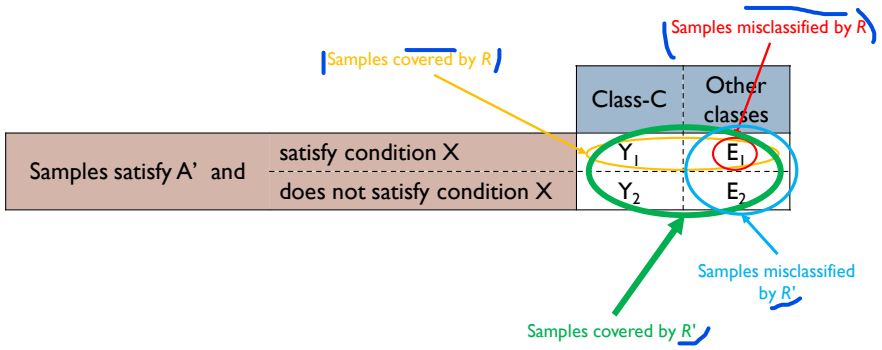
  answer: Elimination is based on a pessimistic estimate of the accuracy of rules R and R′, using *contingency table* for the rule R.

25

3.1. Generalizing rules by deleting some conditions

- *contingency table*





26

3.1. Generalizing rules by deleting some conditions

- The estimate of the error rate of rule *R*:

$$U_{cf}(Y_1 + E_1, E_1)$$

- The estimate of the error rate of rule R′:

$$U_{cf}(Y_1 + Y_2 + E_1 + E_2, E_1 + E_2)$$

- If the pessimistic (estimate of the) error rate of rule *R'* is no greater than that of the original rule *R*, then it makes sense to delete condition *X*, thus is *R* replaced with *R'*.

27

3.1. Generalizing rules by deleting some conditions

- Example:

		Class-C	Other classes
Samples satisfy A' and	satisfy condition X	8	1
	does not satisfy condition X	7	0

$$U_{cf}(Y_1 + E_1, E_1) = U_{cf}(9, 1) = 0.183$$

$$U_{cf}(Y_1 + Y_2 + E_1 + E_2, E_1 + E_2) = U_{cf}(16, 1) = 0.157$$

As the estimated error rate of the rule R′ is lower than the estimated error rate for the initial rule R, a rule set pruning could be done by simplifying the decision rule R and replacing it with R′.

28

3.1. Generalizing rules by deleting some conditions

- Some complications caused by a rule's generalization:
  - There will be the cases that satisfy the conditions of more than one rule,
    solution: The conflict resolution schema in C4.5 (details not given here) selects one rule when there is "multiple-rule satisfaction".
  - There will be the cases that satisfy the conditions of no rules.
    solution: the solution is a *default rule* or a *default class*.
    - One choice for *default class* would be the class that appears most frequently in the training set.
    - C4.5 uses a modified strategy and simply chooses as the default class the one that contains the most training samples not covered by any rule.