Computational Intelligence: Methods and Applications

Lecture 19
Pruning of decision trees

Włodzisław Duch
SCE, NTU, Singapore
Google: Duch

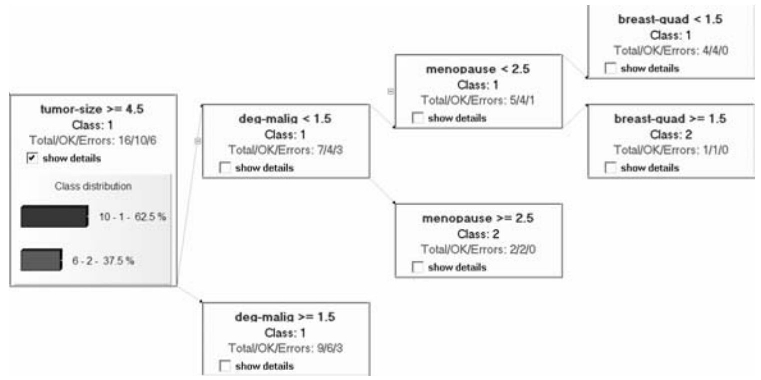
Pruning

How to avoid overfitting and deal with the noise in data?

1. Stop splitting the nodes if the number of samples is too small to make reliable decisions.
2. Stop if the proportion of samples from a single class (node purity) is larger than a given threshold - forward pruning.
3. Create a tree that fits all data and then simplify it - backward pruning.

- Prune to improve the results on a validation set or on a crossvalidation test partition, not on the training set!
- Use the MDL (Minimum Description Length) principle: Minimize Size(Tree) + Size(Tree(Errors))
- Evaluate splits looking not at the effects on the next level but a few levels deeper, using beam-search instead of best-first search; for small trees exhaustive search is possible.

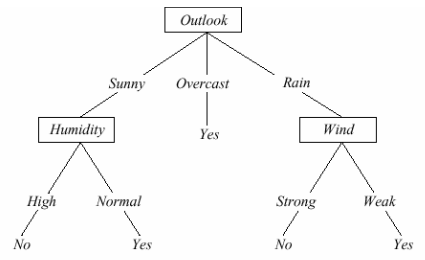
DT for breast cancer



Leave only the most important nodes, even if they are not pure, prune the rest.

DT ➔ logical rules

Each path from the root to a leaf in DT is a rule: the number of rules is equal to the number of rules, but some conditions and rules may be spurious and should be deleted.



IF $(Outlook = Sunny) \wedge (Humidity = High)$ THEN $PlayTennis = No$

IF $(Outlook = Sunny) \wedge (Humidity = Normal)$ THEN $PlayTennis = Yes$

General tree algorithm

TDIDT - Top Down Iterative Decision Tree algorithm

```
function DT(D: training set) returns tree;
        Tree' := construct_tree(D);
        Tree:= prune_tree(Tree');
        return Tree;

function construct_tree(D: training set) returns Tree;
•       T := generate_test_results(D);
•       t := select_best_test(T, D);
•       P := partition D induced by the test t;
•       if stop_condition(D, P)
•       then return Tree=leaf(info(E))
•       else
•             for all D_j in P: t_j := construct_tree(D_j);
•             return node(t, {(j,t_j)};
```

ID 3

ID3: Interactive Dichotomizer, version 3, initially called CLS (Concept Learning System), R. Quinlan (1986)

Works only with nominal attributes.
For numerical attributes separate discretization step is needed.

Splits selected using the information gain criterion Gain(D,X).

The node is divided into as many branches, as the number of unique values in attribute X.

ID3 creates trees with high information gain near the root, leading to locally optimal trees that are globally sub-optimal.

No pruning has been used.

ID3 algorithm evolved into a very popular C4.5 tree (and C5, commercial version).

C4.5 algorithm

One of the most popular machine learning algorithms (Quinlan 1993)

- TDIDT tree construction algorithm, several variants of the algorithm are in use, but textbooks do not describe it well.

- Tests:
  X=? for nominal values,
  $X<t$, for $t=(X_i+X_{i+1})/2$ (only those pairs of X values should be checked where the class changes.

- Evaluation criterion – information gain ratio $G_R(Data,Attribute)$
- $I(D)$ - information (entropy) contained in class distribution

$$I(D) = -\sum_{i=1}^{K} P(\omega_i) \lg_2 P(\omega_i)$$

C4.5 criterion

Information gain: calculate information in the parent node and in the children nodes created by the split, subtract this information weighting it by the percentage of data that falls into *k* children's nodes:

$$G(D,X) = I(D) - \sum_{i=1}^{k} \frac{|D_i|}{|D|} I(D_i)$$

- Information gain ratio $G_R(D$,X) is equal to the information gain divided by the amount of information in the split-generated data distribution:

$$I_S(D,X) = -\sum_{i=1}^{k} \frac{|D_i|}{|D|} \times \lg_2\left(\frac{|D_i|}{|D|}\right)$$
$$G_R(D,X) = G(D,X) / I_S(D,X)$$

Why ratio? To avoid preferences of attributes with many values.

I_S decreases the information gain for nodes that have many children.

CHAID

CHi-squared Automatic Interaction Detection, in SPSS (but not in WEKA), one of the most popular trees in data mining packages.

Split criterion for the attribute X is based on χ^2 test that measures correlations between two distributions.

For a test, for example selecting a threshold $X<X_0$ (or $X=X_0$) for each attribute, a distribution of classes $N(\omega_c|\text{Test=True})$ is obtained; it forms a contingency table: class vs. tests.

If there is no correlation with the class distribution then

$P(\omega_c|\text{Test=True})=P(\omega_c)P(\text{Test=True})$

$N(\omega_c|\text{Test=True})=N(\omega_c)N(\text{Test=True})$

Compare the actual results n_{ij} obtained for each test with these expectation e_{ij}; if they match well then the test is not worth much.

χ^2 test measures the probability that they match by chance: select tests with the largest χ^2 value for $(n_{ij}-e_{ij})^2$

CHAID example

Hypothesis: test result $X<X_0$ (or $X=X_0$) is correlated with the class distribution; then χ^2 test has a small value (see *Numerical Recipes*).

Expectation: $e_{ij}= N_{i0} \times N_{gj} / N$

$$\chi^2 = \sum_{ij}\left(n_{ij}-e_{ij}\right)^2 / e_{ij}$$

χ^2 distribution for $k = (N_{i0}-1)\cdot(N_{gj}-1)$ degrees of freedom.

Example: class=species, X=tail length. Contingency table:

$$P\left(\chi^2 \mid k\right) = \mathrm{erf}\left(k,\chi\right)$$

Species	Long	Short	Missing	Sum
# birds	n_{11}	n_{21}	n_{31}	N_{g1}
# reptiles	n_{12}	n_{22}	n_{32}	N_{g2}
#mammal	n_{13}	n_{23}	n_{33}	N_{g3}
# fish	n_{14}	n_{24}	n_{34}	N_{g4}
Sum	N_{10}	N_{20}	N_{30}	N

Probability $P()$ that the disagreement is not by chance is given by erf = error function, integrated Gaussian.

CART

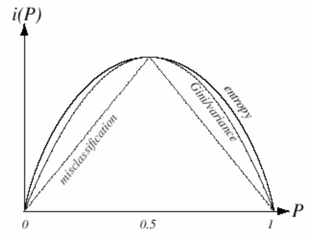
Classification and Regression Trees (Breiman 1984).

Split criterion: instead of information, uses the change in Gini index; in a given node p_c is % of samples from ω_c class; purity of node is:

$$Gini = \sum_{c\neq d}^{C} p_c p_d = 1-\sum_{c=1}^{C} p_c^2$$

Other possibility: misclassification rate

$$Mi = 1-\max_c p_c$$



Stop criterion: MDL, parameter α, tradeoff between complexity and accuracy:

here 2*Gini, 2*Mi, entropy

$\alpha\cdot$tree complexity + leaf entropy

$$\alpha Size(Tree) + \sum_{l\in leaf} I(l)$$

Other trees

See the entry in WIKIpedia on decision trees.

Several interesting trees for classification and regression are at the page of Wei-Yin Loh.

YaDT: Yet another Decision Tree builder.

RuleQuest has See 5, a new version of C4.5 with some comparisons.
Only a demo version is available.
Look also at their Magnum Opus software to discover interesting patters – association rules, and k-optimal rule discovery.

Occam Tree from Visionary Tools.