# Computational Intelligence: Methods and Applications

Lecture 20

SSV & other trees

Włodzisław Duch

SCE, NTU, Singapore

Google: Duch

## GhostMiner Philosophy

GhostMiner, data mining tools from our lab.

http://www.fqspl.com.pl/ghostminer/

or write "Ghostminer" in Google.

- Separate the process of model building and knowledge discovery from model use =>

  GhostMiner Developer & GhostMiner Analyzer.
- There is no free lunch – provide different type of tools for knowledge discovery.
  Decision tree, neural, neurofuzzy, similarity-based, committees.
- Provide tools for visualization of data.
- Support the process of knowledge discovery/model building and evaluating, organizing it into projects.

## SSV in GM

SSV = Separability Split Value, simple criterion measuring how many pairs of samples from different classes are correctly separated.

Define subsets of data $D$ using a binary test $f(\mathbf{X},s)$ to split the data into left and right subset $D = LS \cup RS$.

$$LS(s,f,D) = \{\mathbf{X} \in D : f(\mathbf{X},s) = \mathrm{T}\}$$
$$RS(s,f,D) = D - LS(s,f,D)$$

Tests: defined on vector X, usually on a single attribute $X_i$ for continuous values comparing it with a threshold $s$,

$$f(\mathbf{X},s) = \mathrm{T} \Leftrightarrow \mathbf{X}_i < s$$

or a subset of values for discrete attributes.

Another type of tests giving quite different shapes of decision borders is based on distances from prototypes.

## SSV criterion

Separability = the number of samples that are in *LS* subset and are from class $\omega_c$ times the number of elements in *RS* from all the remaining classes, summed over all classes.

If several tests/thresholds separate the same number of pairs (this may happen for discrete attributes) select the one that separates a lower number of pairs from the same class.

$$SSV(s) = 2\sum_{c\in C}\left|LS(s,f,D)\cap D_c\right|\cdot\left|RS(s,f,D)\cap(D-D_c)\right|$$
$$-\sum_{c\in C}\min\left(\left|LS(s,f,D)\cap D_c\right|,\left|RS(s,f,D)\cap D_c\right|\right)$$

SSV is maximized; first part should dominate, hence factor 2.

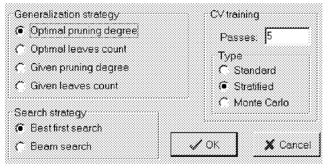
Simple to compute, creates full tree using top-down algorithm with best-first search or beam search procedures to find better trees.

Uses cross-validation training to select nodes for backward pruning.

## SSV parameters

Generalization strategy: defines how the pruning is done.

First, try to find optimal parameters for pruning:
final number of leaf nodes, or
pruning degree k = remove all nodes that increased accuracy by k samples only.



Generalization strategy: [✓ Optimal pruning degree] Optimal leaves count / Given pruning degree / Given leaves count

CV training: Passes: 5

Type: Standard [✓ Stratified] Monte Carlo

Search strategy: [✓ Best first search] Beam search

Given pruning degree or given nodes count define these parameters by hand.

"Optimal" uses cross-validation training to determine these parameters: the number of CV folds and their type has to be selected.

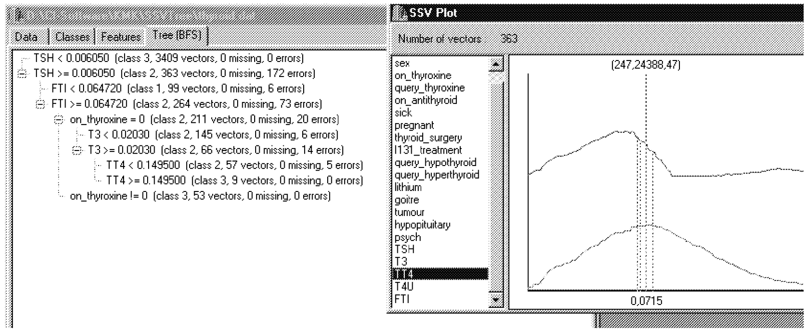
Optimal numbers: minimize sum of errors in the test parts of CV training.

**Search strategy:** use the nodes of tree created so far and:

use the best-first search, i.e. select the next node for splitting;
use beam search: keep ~10 best trees, expand them; avoids local maxima of the SSV criterion function but rarely gives better results.

## SSV example

Hypothyroid disease, screening data with 3772 training (first year) and 3428 test (second year) examples, majority 92.5% are normal, the rest are primary hypothyroid or compensated hypothyroid cases.



```
TSH < 0.006050  (class 3, 3409 vectors, 0 missing, 0 errors)
TSH >= 0.006050  (class 2, 363 vectors, 0 missing, 172 errors)
  FTI < 0.064720  (class 1, 99 vectors, 0 missing, 6 errors)
  FTI >= 0.064720  (class 2, 264 vectors, 0 missing, 73 errors)
    on_thyroxine = 0  (class 2, 211 vectors, 0 missing, 20 errors)
      T3 < 0.02030  (class 2, 145 vectors, 0 missing, 6 errors)
      T3 >= 0.02030  (class 2, 66 vectors, 0 missing, 14 errors)
        TT4 < 0.149500  (class 2, 57 vectors, 0 missing, 5 errors)
        TT4 >= 0.149500  (class 3, 9 vectors, 0 missing, 0 errors)
    on_thyroxine != 0  (class 3, 53 vectors, 0 missing, 0 errors)
```

TT4 attribute: red is # errors; green - # correctly separated pairs; blue is # pairs separated from the same class (here always zero).

## Wine data example

Chemical analysis of wine from grapes grown in the same region in Italy, but derived from three different cultivars.
Task: recognize the source of wine sample.
13 quantities measured, continuous features:

- alcohol content
- ash content
- magnesium content
- flavanoids content
- proanthocyanins phenols content
- OD280/D315 of diluted wines
- malic acid content
- alkalinity of ash
- total phenols content
- nonanthocyanins phenols content
- color intensity
- hue
- proline.

Wine robot

## C4.5 tree for Wine

J48 pruned tree: using reduced error (3x crossvalidation) pruning:

```
------------------
OD280/D315 <= 2.15
|   alkalinity <= 18.1: 2 (5.0/1.0)
|   alkalinity > 18.1: 3 (31.0/1.0)
OD280/D315 > 2.15
|   proline <= 750: 2 (43.0/2.0)
|   proline > 750: 1 (40.0/2.0)

Number of Leaves  :     4
Size of the tree :      7
Correctly Classified Instances       161        90.44 %
Incorrectly Classified Instances      17         9.55 %
Total Number of Instances            178
```

## WEKA/YALE output

WEKA output contains confusion matrix, YALE has transposed matrix

```
  a  b  c  <= classified as
 56  3  0 |  a = 1 (class number)
  4 65  2 |  b = 2
  3  1 44 |  c = 3
```

P(true|predicted)

2x2 matrix:

$$\begin{array}{ccc} TP & FN & P_+ \\ FP & TN & P_- \\ P_+(M) & P_-(M) & 1 \end{array}$$

Pr=Precision=TP/(TP+FP)

R =Recall = TP/(TP+FN)

F-Measure = 2Pr*R/(P+R)

WEKA:

```
TP Rate  FP Rate  Precision  Recall  F-Measure  Class
  0.949    0.059     0.889    0.949    0.918      1
  0.915    0.037     0.942    0.915    0.929      2
  0.917    0.015     0.957    0.917    0.936      3
```

## WEKA/YALE info

Other output information:

Kappa statistic 0.8891

(corrects for chance agreement)

$$\kappa = \frac{N\sum_{i=1}^{K} N_{ii} - \sum_{i=1}^{K} N_i N_i(M)}{N^2 - \sum_{i=1}^{K} N_i N_i(M)}$$

Mean absolute error 0.0628

$$MAE = \frac{\sum_{k=1}^{n} \left| P(\omega|\mathbf{X};M) - Y(\mathbf{X}) \right|}{n}$$

Root mean squared error 0.2206

$$RMS = \sqrt{\frac{\sum_{k=1}^{n} \left| P(\omega|\mathbf{X};M) - Y(\mathbf{X}) \right|^2}{n}}$$

Root relative squared error 47.0883%

Relative absolute error 14.2992%

$$RAE = \frac{\sum_{k=1}^{n} \left| P(\omega|\mathbf{X};M) - Y(\mathbf{X}) \right|}{\sum_{k=1}^{n} \left| Y(\mathbf{X}) - \overline{Y} \right|}$$

## Simplest SSV rules

Decision trees provide rules of different complexity, depending on the pruning parameters.

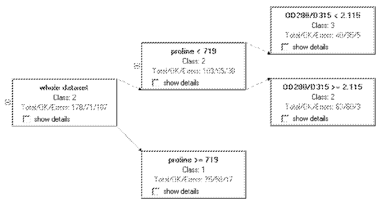
Simpler trees make more errors but help to understand data better.

In SSV pruning degree or pruning nodes may be selected by hand.

Start from small number of nodes, see how the number of errors in CV will change.

Simplest tree: 5 nodes, corresponding to 3 rules;

25 errors, mostly Class2/3 wines mixed.



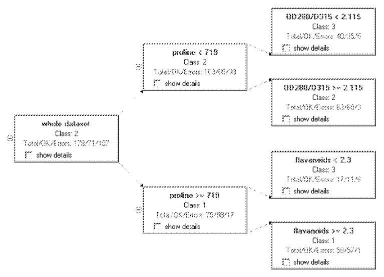
## Wine – SSV 5 rules

Lower pruning leads to more complex but more accurate tree.

7 nodes, corresponding to 5 rules;

10 errors, mostly Class2/3 wines mixed.

Try to lower the pruning degree or increase the node number and observe the influence on the error.



| | | |
|---|---|---|
| av. 3 nodes train 10x: | 87,0±2,1% | test 80,6±5,4±2,1% |
| av. 7 nodes train 10x: | 98,1±1,0% | test 92,1±4,9±2,0% |
| av. 3 nodes train 10x: | 99,7±0,4% | test 90,6±5,1±1,6% |

## Wine – SSV optimal rules

What is the optimal complexity of rules?
Use crossvalidation to estimate optimal pruning for best generalization.

Various solutions may be found, depending on the search parameters:
5 rules with 12 premises, making 6 errors,
6 rules with 16 premises and 3 errors,
8 rules, 25 premises, and 1 error.

if OD280/D315 $> 2.505 \wedge$ proline $> 726.5 \wedge$ color $> 3.435$ then class 1

if OD280/D315 $> 2.505 \wedge$ proline $> 726.5 \wedge$ color $< 3.435$ then class 2

if OD280/D315 $< 2.505 \wedge$ hue $> 0.875 \wedge$ malic-acid $< 2.82$ then class 2

if OD280/D315 $> 2.505 \wedge$ proline $< 726.5$ then class 2

if OD280/D315 $< 2.505 \wedge$ hue $< 0.875$ then class 3

if OD280/D315 $< 2.505 \wedge$ hue $> 0.875 \wedge$ malic-acid $> 2.82$ then class 3

## DT summary

DT: fast and easy, recursive partitioning.

### Advantages:

easy to use, very few parameters to set, no data preprocessing;

frequently give very good results, easy to interpret, and convert to logical rules, work with nominal and numerical data.

Applications: classification and regression.

Almost all Data Mining software packages have decision trees.

### Some problems with DT:

few data, large number of continuous attributes, unstable;
lower parts of trees are less reliable, splits on small subsets of data;
DT knowledge expressive abilities are rather limited, for example it is hard to create a concept: “majority are for it”, easy for M-of-N rules.