Applied inductive learning - Lecture 2

Louis Wehenkel

Department of Electrical Engineering and Computer Science
University of Liège

Montefiore - Liège - September 26, 2005

Find slides: http://montefiore.ulg.ac.be/∼lwh/AIA/

Batch-mode Supervised Learning

- Problem definition
  Given a learning sample (LS) of input/output pairs, build a model (i.e. an algorithm, or a rule) to compute outputs as a function of inputs.
  - Inputs are described by a set of attributes
  - Model must match input/output pairs of LS
  - Model must also predict correctly outputs for unseen inputs (test sample)
- General questions
  - Interpretability
  - Accuracy
  - Computational complexity

Decision tree induction

- Growing
- Scoring candidate splits (or tests)
- Pruning
- Demo
- Regression trees

Tree growing

- Top down induction of decision trees
  - Start with root node (i.e. top-node)
  - Use complete learning sample
  - Treat attributes one by one
    - Compute score for each split
    - Determine best split (highest score)
  - Determine best attribute and split (highest score)
  - Split learning sample
- Proceed in the same way with the two subsamples
- Stop procedure when subset is "pure"

Scoring a split based on a sample S

- Entropy estimate (Shannon): $H_C(S) = -\sum_c \frac{N_c(S)}{N(S)} \log_2 \frac{N_C(S)}{N(S)}$
- Entropy reduction by a split (Example)

Set	N_I	N_S	N	H_C
$S_1 : PU > 945.76$	31	21	52	0.973
$S_2 : PU \leq 945.76$	0	48	48	0.000
$S = S_1 \cup S_2$	31	69	100	0.893

- Information Gain of split

$$I_C^T(S) = H_C(S) - \frac{N_1(S)}{N(S)} H_C(S_1) - \frac{N_2(S)}{N(S)} H_C(S_2)$$

Here: $I = 0.893 - 0.52 \times 0.973 - 0.48 \times 0.000 = 0.387$

Normalized score measure (used in PEPITo)

- Entropy estimate (Shannon): $H_C(S) = \ldots$
- Split entropy: $H_T(S) = -\frac{N_1(S)}{N(S)} \log_2 \frac{N_1(S)}{N(S)} - \frac{N_1(S)}{N(S)} \log_2 \frac{N_1(S)}{N(S)}$
- Information Gain of split $I_C^T(S) = \ldots$
- SCORE: $C_C^T(S) = \frac{2 I_c^T(S)}{H_C(S) + H_T(S)}$

NB: This score measure is a normalized version of the information gain, i.e. $C_C^T(S) \in [0; 1]$

Total information quantity, complexity, quality

- Tree complexity: $C(\mathcal{T}) = \#testnodes$
- Denote by $i = 1, \ldots, C(\mathcal{T})$ the test nodes of the tree $\mathcal{T}$
- Denote by S_i and T_i the learning sample and test at node i, and let $N_i = \#S_i$
- Denote by N the size of the complete learning sample LS
- Information provided by a tree: $I_C^{\mathcal{T}} = \sum_{i=1}^{C(\mathcal{T})} \frac{N_i}{N} I_C^{T_i}(S_i)$
- Tree quality: $Q(\mathcal{T}, LS) = NI_C^{\mathcal{T}} - \beta C(\mathcal{T})$, $\beta \geq 0$.

The quality measure is used for tree pruning (both pre- and post-pruning); see notes.

Attribute importance measure

- Let $a(T)$ denote the attribute used by the test T.
- Information provided by attribute a in tree $\mathcal{T}$:

$$I_C^a = \sum_{i=1}^{C(\mathcal{T})} 1(a(T_i) = a) \frac{N_i}{N} I_C^{T_i}(S_i)$$

- NB: $I_C^{\mathcal{T}} = \sum_{att} I_C^{att}$
- Importance of attribute: $Imp(a) = \frac{I_C^a}{I_C^{\mathcal{T}}}$.

The more important attributes are those which lead to high scores at nodes which have a large number of samples.

Tree pruning

Why to prune trees ?

- To avoid overfitting
- To simplify interpretation, use

How to prune trees ?

- Early stopping using hypothesis test
- Post-pruning using cross-validation sample

Regression trees

Same principle

Entropy is replaced by the variance of the output variable y:

$$Var_Y(S) = N(S)^{-1} \sum_{o \in S} (y(o) - \mu_Y(S))^2$$

with $\mu_Y(S) = N(S)^{-1} \sum_{o \in S} y(o)$

$$\Delta Var_Y^T(S) = Var_Y(S) - \frac{N_1(S)}{N(S)} Var_Y(S_1) - \frac{N_2(S)}{N(S)} Var_Y(S_2)$$

Score: $\frac{\Delta Var_Y^T(S)}{Var_Y(S)}$

Quality measure and attribute importances are derived in the same way as in classification.

Homework

- Screen chapters 1 and 2, and read Chapter 5 of course notes
- Create PEPITo project from omib.jdb file provided (see web page).
- Play with PEPITo using OMIB database.