Introduction

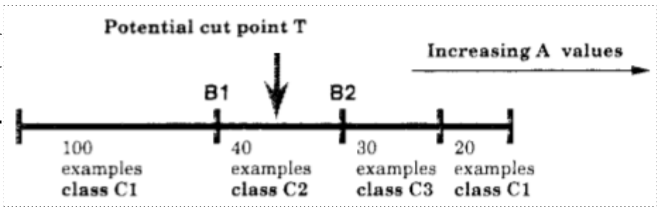
The theme I have chosen is linked to the « simplicity » lab and revolves around an application of the minimum description length principle in the context of multi-interval discretization of continuous-valued attributes for classification learning. Programs that learn from pre-classified examples are deployed as an alternative for expert human judgment on writing rules on example attributes for classification purposes. By examining large sets of labeled data, it is hoped that a learning program may discover the proper conditions under which each class is appropriate. This rules are represented as trees in which internal nodes represent split decisions based on the attribute values of the seen examples and are created by the use of heuristics to search through the space of possible relations and splitting criteria. Famous implementations as CART, ID3 or C4.5 proceed by selecting attributes that minimize the information entropy of the classes in the data-set. However, this attribute selection assumes that the selection process occurs among categorical/nominal attributes; continuous attributes must therefore be discretized beforehand. This involves choosing a particular binning-schema in which each bin is an interval in the continuous values that the attribute exhibits. The focus of Fayad & Irani in their 1992 paper “multi-interval discretization of continuous-valued attributes for classification learning” is to derive a gain in computational efficiency by employing the minimum description length to pick the most-likely hypothesis of bin boundaries in the case of continuous attribute discretization. My micro-research is to implement the idea they came up with.

Start-case : The binary discretization problem

During tree generation, a set S is partitioned in two bins with respect to a continuous valued attribute A by specifying a threshold value, T, to respect of which an example instance would be assigned to the left branch or to the right branch whether A< T or A => T respectively. T is chosen among the K possible values that A assumes by sorting the latter and computing the value that minimizes the most E(A,T,S) the class information entropy induced by choosing T as splitting criterion for S to form a left-branch set S1 and a right branch set S2.

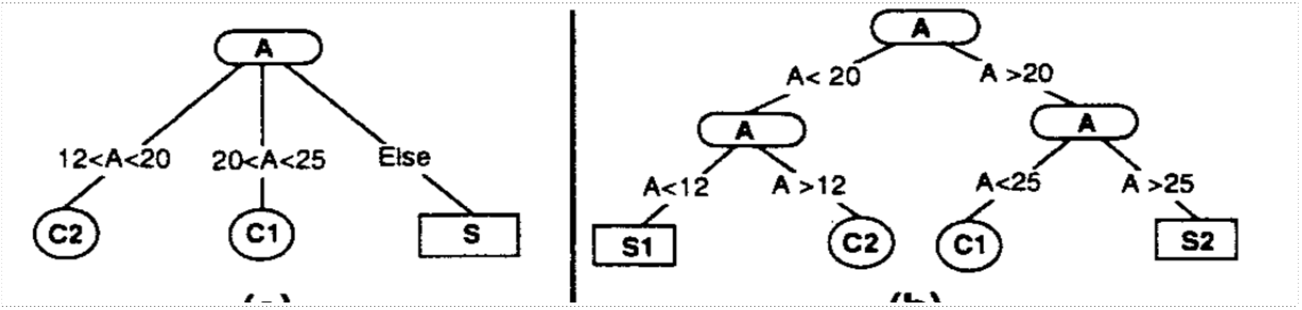
$$E(A, T; S) = \frac{|S_1|}{|S|}\mathrm{Ent}(S_1) + \frac{|S_2|}{|S|}\mathrm{Ent}(S_2)$$

Although the computation of E(A,T,S) is polynomial in time, it should be stressed that it is realized for every candidate value that A assumes – meaning K-1 times. And for every decision node in the tree. A first contribution made by the paper is to negate the necessity of checking every K-1 value but to only focus on class boundary such as B1 and B2 as potential candidates as they derive an implication that minimum E(A,T,S) only occurs at these points and not on arbitrary T as shown in the figure on the right. Although a very interesting result for tree generation speed-up this isn’t the topic of my micro-research.



Generalizing the start-case : An application of the MDL principle.

The second contribution of the paper and that catches my interest in the context of the “simplicity” theme lab for DK902 is an observation that occurs when the ‘interesting interval’ (as coined by the authors) may be an internal range within A. To get such an interval, binary-interval-at-a-time approach leads to unnecessary excessive partitioning leading to i) excessive memory usage and a loss in node purity and ii) overall tree performance. As shown in the following figure, instead of a 3-way split that would discriminate the interval [12-25], a binary split would generate unnecessary, possibly noise inducing S1 and S2.



To address this issue, the authors propose to view the problem of finding correct splitting values as an information coding problem. By formulating the problem as binary decision problem of cutting/not-cutting when it comes to a potential value T of attribute A. We therefore distinguish between two hypotheses:

- HT: The hypothesis that T induces if it is accepted as a splitting criterion.
- NT: The null hypothesis, T is rejected as a splitting criterion.

The best strategy calls for selecting the decision with the maximal probability. The MDL principle stating that the most likely model is the one yielding the minimum length of description, it’s therefore possible to chose among {HT,NT} by choosing the shortest of the two of the following sequences of describing the competing hypotheses:

[length(HT) + length(S|HT)] vs [length(NT) + length(S|NT)]

The author have therefor shifted the problem of choosing ideal splitting criteria to a coding problem where :

Coding NT : Coding the classes contained in the set S in sequence =

K*Average_code_length_of_class_labels + O(1)

Coding HT : Specifying the cut value $\log_2(K-1)$ +

K1*Average_code_length_of_class_labels_in_S1

K2*Average_code_length_of_class_labels_in_S1 + O(K)

This solves the problem at hand and allows to emit cut/not-cut for all candidate T’s, allowing therefore a multi-interval discretization of continuous-valued attributes using the MDL principle. Empirical test-runs realized by the authors prove indeed better tree generation and lower memory consumption on the same data.

My work

A MDLP.c is joined to this current report and can be compiled to be used as the command line. It’s a Cython compiled code, chosen for performance consideration. I will post on my git-hub the commented python source, before the end of the week of which the link I will share by email.

./MDLP [Parametres]

```
Parametres: ./MDLP.py --source=path --destination=path --features=f1,f2,f3... ' \
            '--labels_classes=temperature'
```

Where the parameters are respectively :

The path to dataset of which features f1,f2,f3 need to be discretized

The destination dataset copy of source with discretized features

The column names to be discretized

The Y (target) label class.

The required dependencies are :

Numpy and Pandas and also the python-dev package for debian machines / python-devel on redhat distros