Introduction

Breaking news, the lottery draw for this week is 45-28-12-31-22-7.

Breaking news, the lottery draw for this week is 1-2-3-4-5-6.

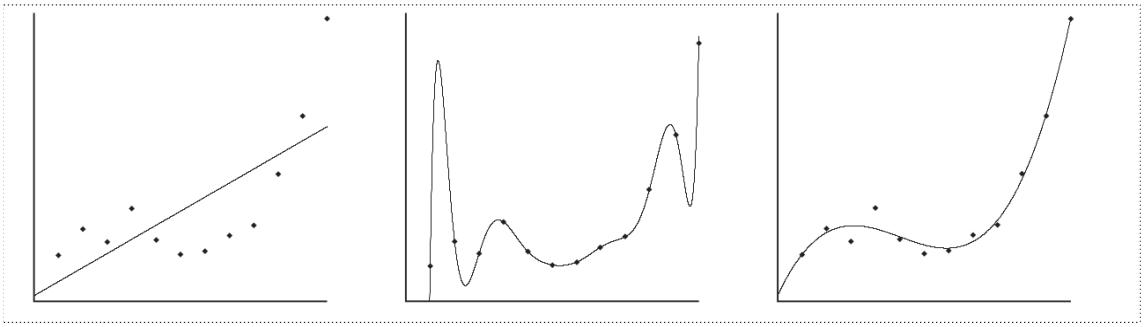
We would all agree that the second news is worth tweeting about, it seems so unlikely that it would be the subject of many cafeteria talks the day after. However, upon careful consideration, one would argue that the probability is the same for all combinations, and it is very low anyway (about 1/14 000 000 for 6 numbers drawn between 1 and 49 without order. This distance between human intuition and mathematical probability theory is at the core of the article I have chosen is titled “A Computational Theory of Subjective Probability” of Maguire & al. The starting assumption made by the authors is that in real life, there’s often less data than needed to compute probability in classical mathematical fashion; instead people rely on “subjective probability”: The more an event is uprisings, the less likely it is to occur.

Subjective Probability

To meet the shortcuts human brain seem to use when weighting probabilities of event, advances in algorithmic statistics derived a definition of an optimal model of an outcome as the one which yields the Minimum Description Length. This is no more than a formal definition of Occam’s razor. If an observation is no longer typical with respect to the MDL model, it calls for an update of the beliefs to meet the randomizes deficiency of the data. This adjustment is associated to a cost that is the information required given the original model to derive the update. Therefore the main contribution of the paper is shifting the focus from an undetermined probability measure function to the immutable mechanism of representational updating.

Kolmogorov complexity

According to the MDL principle, learning is seen as data compression. The MDL of a given binary string, is the length of the shortest instruction set to produce the string. It sheds light therefore on the concept of regularity. Indeed, the first example sequence of the introduction can only be described by: print(“45-28-12-31-22-7”);halt; . When the second sequence can be described by : print([1-6]);halt;. The latter is then “less random” or more appropriately more regular than the first. Application of the MDL principle becomes then a coding problem as defined by Information theory. We seek then for a set H of competing hypotheses H_i to select the one that minimizes the description of the data D given H_i, meaning $L(Hi) + L(D|H_i)$. This can be illustrated by the following figure where the best hypothesis is the one on the right, the polynomial H_i that makes a good compromise between the complexity of H_i and the complexity of the errors to be encoded with the help of H_i



Subjective information and probability

Given a hypothesis P a probability density function, there are some strings we 'humanly' expect to be output, where as others are surprising. Let let $\alpha > 0$ be a constant, called the surprise threshold, which represents the level of randomness deficiency that necessitates representational updating. A string x is said to be α-P typical if it cannot be described by less than L(x|p) – α : There's no randomness deficiency of x with respect to P, ie: $K(x|p^*) \geq -\log p(x) - \alpha.$

Suppose an observer experiences observations $d_1, d_2, \ldots$ generated by some source with computable probability density. Then the best model given the data for the source is the one that yields the MDL of d_i, ie:

$$p_n = \arg\min\{|p^*| : p \text{ is optimal for } d_1, d_2, \ldots, d_n \text{ and } d_1, d_2, \ldots, d_n \text{ are } (p, \alpha)\text{-typical}\}.$$

We therefore define an observation d_{n+1} to be α-surprising can be explained by less bits than what is needed to encode it given p_n. This calls for a representational updating yielding p_{n+1}. Therefore, the subjective information brought by d_{n+1} is $K(p_{n+1} | p_n)$. This expressed in Shanon-fano code, its probability is $2^{-K(pn+1 \mid pn)}$. This statement is the author's major contribution in the paper : providing a computational formula for subjective probability of an event.

Question by Prof Suchanek

In the Q&A section of my presentation, professor Suchanek asked a question I failed to give a satisfying answer to. The question being "To compute a subjective probability of an event, one has to be able to encode hypothesis and data given the said hypothesis. How does one in the same context of a problem encode categorical data and continuous data?". Thankfully professor Dessalles, pointed to the answer by stating that continuous data is brought to compatibility with categorical data by discretization and thus allowing a justifiable code length comparison. What follows is some of my findings and an application regarding, continuous variable discretization.

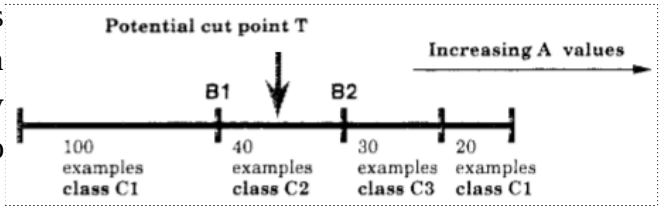
The bibliographical resources I sought to understand the answer to the question revolves around an application of the minimum description length principle in the context of multi-interval discretization of continuous-valued attributes for classification learning. Programs that learn from pre-classified 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.

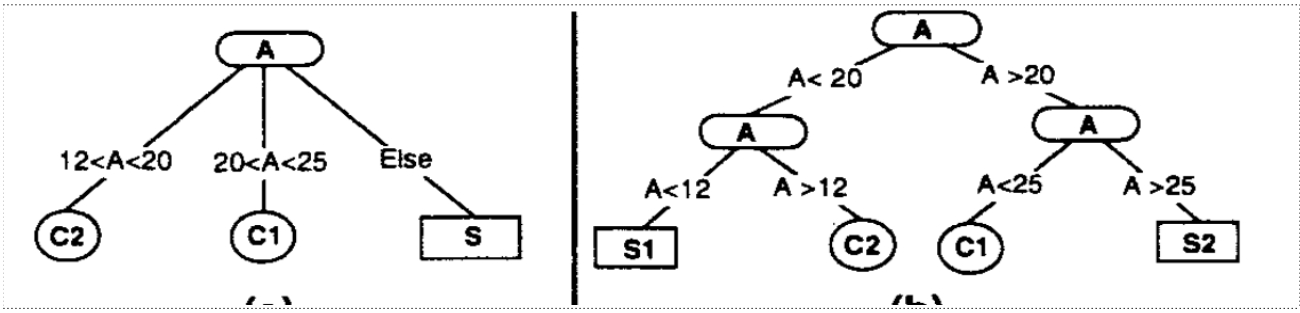
$$E(A,T;S) = \frac{|S_1|}{|S|}\mathrm{Ent}(S_1) + \frac{|S_2|}{|S|}\mathrm{Ent}(S_2)$$

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 answering prof Suchanek's question and was also the topic I have chosen 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