Prediction of benthic macro-invertebrate taxa (*Asellidae* and *Tubificidae*) in watercourses of Flanders by means of classification trees

Peter Goethals, Saso Dzeroski, Peter Vanrolleghem & Niels De Pauw

Introduction

Prediction of freshwater organisms based on machine learning is becoming more and more reliable due to the availability of appropriate datasets and modelling techniques. Machine learning techniques as neural networks, classification trees, evolutionary algorithms are often powerful tools to perform ecological modelling, especially when large datasets are involved. Dzeroski *et al*. (1997) illustrated the use of machine learning techniques in the reduction of subjectivity of river quality classification methods based on biological indices, but many other applications in integrated ecological river management will probably prove the convenience of data driven modelling in the near future.
Classification trees (Breiman *et al*., 1984), often referred to as decision trees (Quinlan, 1986) predict the value of a discrete dependent variable with a finite set of values (called class) from the values of a set of independent variables (called attributes), which may be either continuous or discrete. Data describing a real system, represented in the form of a table, can be used to learn or automatically construct a decision tree.
The predictive power of classification trees will be illustrated on a dataset of benthic macro-invertebrate taxa in Flemish watercourses. This dataset consists of physical-chemical and eco-toxicological measurements as well as abundances of benthic macro-invertebrates. In this paper, the results on two important taxa (*Asellidae* and *Tubificidae*) are presented.

Materials and methods

The common way to induce decision trees is the so-called Top-Down Induction of Decision Trees (TDIDT, Quinlan 1986). Tree construction proceeds recursively starting with the entire set of training examples (entire table). At each step, the most informative attribute is selected as the root of the (sub)tree and the current training set is split into subsets according to the values of the selected attribute. For discrete attributes, a branch of the tree is typically created for each possible value of the attribute. For continuous attributes, a threshold is selected and two branches are created based on that threshold. For the subsets of training examples in each branch, the tree construction algorithm is called recursively. Tree construction stops when all examples in a node are of the same class (or if some other stopping criterion is satisfied). Such nodes are called leaves and are labelled with the corresponding values of the class.
An important mechanism used to prevent trees from over-fitting data is tree pruning. Pruning can be employed during tree construction (pre-pruning) or after the tree has been constructed (post-pruning). Typically, a minimum number of examples in branches can be prescribed for pre-pruning and confidence level in accuracy estimates for leaves for post-pruning.
A number of systems exist for inducing classification trees from examples, e.g., CART (Breiman *et al*., 1984), ASSISTANT (Cestnik *et al*., 1987), and C4.5 (Quinlan, 1993). Of these, C4.5 is one of the most well known and widely-used decision tree induction systems. J48 (Witten & Frank, 1999) is a Java re-implementation of C4.5. It is a part of the machine learning package WEKA, which also includes some of the latest developments in machine learning. Among these, methods for inducing ensembles, such as bagging and boosting are especially important. Bagging (short for bootstrap aggregation) takes a dataset, and generates different classifiers by using the same learner on different (bootstrap) samples taken from the datasets. The predictions of these classifiers are combined by majority vote. Boosting uses error-proportionate sampling. It uses the base learner on the whole dataset to generate the

initial classifier. When sampling in the next iteration, it gives higher probability to examples misclassified by the current classifier. It then induces a classifier with the learner from the next sample. It combines all classifiers induced so far by the learner to produce the next classifier using weighted. The procedure is repeated for the pre-specified number of iterations. In the presented experiments, J48 with default values of the parameters was used. Bagging and boosting was performed with 10 iterations each, based on the J48 algorithm with default values of the parameters as the base learner.
The input variables consisted of a combination of physical-chemical and eco-toxicological variables. The six physical-chemical inputs were temperature, pH and dissolved oxygen concentration, all three measured in the water column combined with total organic carbon, Kjeldahl-nitrogen and total phosphorus concentrations of the sediment. The eco-toxicological evaluation was based on two acute tests on pore water of the sediments: a 72h growth-inhibition-test with *Selenastrum capricornutum* and a 24h growth-inhibition-test with *Thamnocephalus platyurus*. All physical-chemical variables were continuous, while the two eco-toxicological variables were discrete (false or true). The ouput variables, respectively the two considered taxa *Asellidae* and *Tubificidae,* were also discrete (absent or present). The applied database consisted of measurements from 360 sites in unnavigable watercourses in Flanders (Belgium).

Results

This part will present results on:
1) the comparison of prediction efficiency of moderately present organisms (*Asellidae*) to very frequently present organisms (*Tubificidae*) (as illustrated in Figure 1)
2) the comparison of the different algorithms J4.8, J4.8-based boosting and J4.8-based bagging with regard to prediction error and model complexity (related to the interpretation and generalisation convenience)
3) the effect of tree pruning on the prediction error and the complexity of the models.

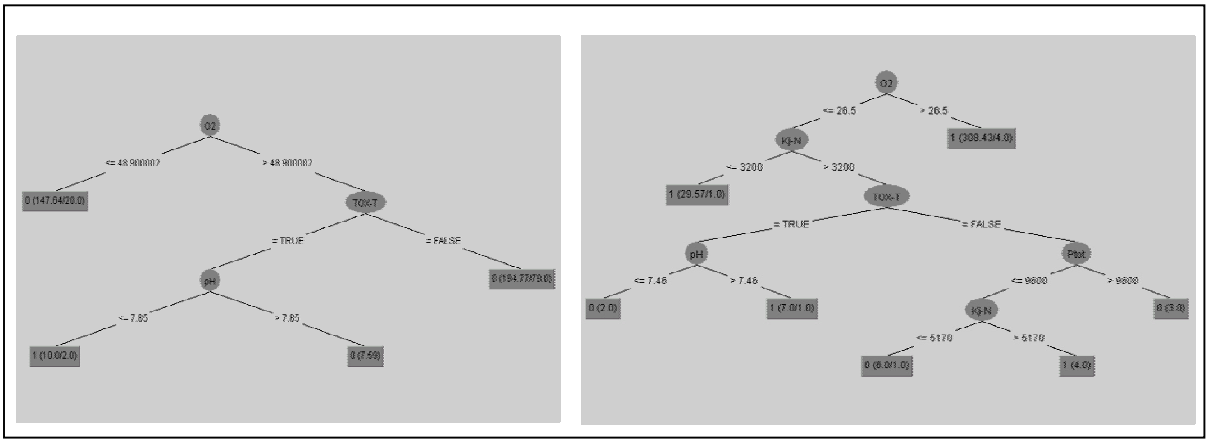

Figure 1 Illustration of two classification trees (left: Asellidae, right: Tubificidae) based on the J4.8 standard algorithm. The nodes represent the input variables that are used to predict the target variables (absence or presence of respectively Asellidae and Tubificidae), while the values on the branches present the cut-levels.

Discussion

In general, the classification trees perform well to predict the two benthic macro-invertebrate taxa, based on the eight input variables. This method does not merely generate results with a low prediction error, but also allows the user to identify associations and general trends in the

data, making it more interesting than complete black box techniques such as artificial neural networks.
When a comparison is made between the prediction of moderately present organisms (*Asellidae*) and very frequently occurring organisms (*Tubificidae*), one can conclude that the prediction of the *Asellidae* with classification trees (69% of the instances are classified well with the standard J4.8 algorithm) is much better than an ordinary probabilistic guess based on the relative absence and presence in the instances. On the other hand, the prediction of *Tubificidae*, which are very frequently occurring organisms (in 96% of the sites present) the prediction is sometimes even worse (only 95% of the instances are classified well with the standard J4.8 algorithm) than an ordinary probabilistic guess based on the average of the set of instances. The latter phenomenon is probably due to the cost of extracting general trends from the data.
Boosting and bagging proved to be very efficient algorithms to decrease the prediction error in all cases and in special for the very frequently occurring *Tubificidae*. The disadvantage of boosting and bagging is the difficulty to find important general trends in the results, due to the increased complexity in the classification trees. Similarly, also the interpretation of the trees is in most cases difficult when boosting and bagging is intensively performed.
Experiments with tree pruning on the other hand, showed that in several cases the tree can be simplified tremendously, without increasing the prediction error too much, leaving a much more convenient result.
Therefore one can conclude that classification trees are interesting grey box prediction techniques, allowing the user to combine a small prediction error with getting some information on general trends in the data. Probably the results can be improved by providing other valuable inputs such as flow velocity, water depth and other structural variables to the system. Experiments with different sets of input variables did not only result in an altered prediction error, but also the complexity of the trees as well as the relative importance of some 'general' trends seemed to get affected. Further research is therefore necessary to get insight in the impact of different input variable sets on the prediction qualities of decision trees.

Acknowledgements

The authors would like to thank the FWO-Flanders for its financial support (project 3G01.02.97). The collection of the data was done within several projects on the establishment of the TRIAD approach for assessing river sediment quality, financed by AMINAL (Division Water) and the Flemish Environmental Agency (VMM).

References

- Breimann, L., J.H. Friedman, R.A. Olshen & C.J. Stone (1984). Classification and regression trees. Pacific Grove: Wadsworth.
- Cestnik, B., I. Kononenko & I. Bratko (1987). ASSISTANT 86: A knowledge elicitation tool for sophisticated users. In: Bratko, I. & Lavrak, N. Progress in machine learning. Wilmslow: Sigma Press.
- Dzeroski S., J. Grbovic & W.J. Walley (1997). Machine learning applications in biological classification of river water quality, p. 429-448. In: Michalski, R.S., I. Bratko & M. Kubat. Machine learning and data mining: methods and applications. New York: John Wiley & Sons Ltd.
- Quinlan, J.R. (1986). Induction of decision trees. Machine Learning 1(1):81-106.
- Quinlan, J.R. (1993). C4.5 : Programs for machine learning. San Francisco: Morgan Kaufmann.

- Witten I.H. & E. Frank (2000). Data mining: practical machine learning tooals and techniques with Java implementations. San Francisco: Morgan Kaufmann Publishers.