Non-metric classification

Up to this time we mainly considered cases in which there existed the natural measure of the distance (or similarity) between the feature vectors.
Nominal (symbolic) data are discreet and (in general) don't have naturally defined similarity and ordering.
These data describe an object in a form of a *list of attributes*.

A typical approach is to select a the constant number of features and to describe an object with a *property n-tuple*.)

The other solution is describing an object with a varying length string of nominal (not necessarily) attributes.

Preparing of these materials was supported by a grant from Liechtenstein, Iceland and Norway through EEA Grants and Norway Grants and co-financed by domestic funds

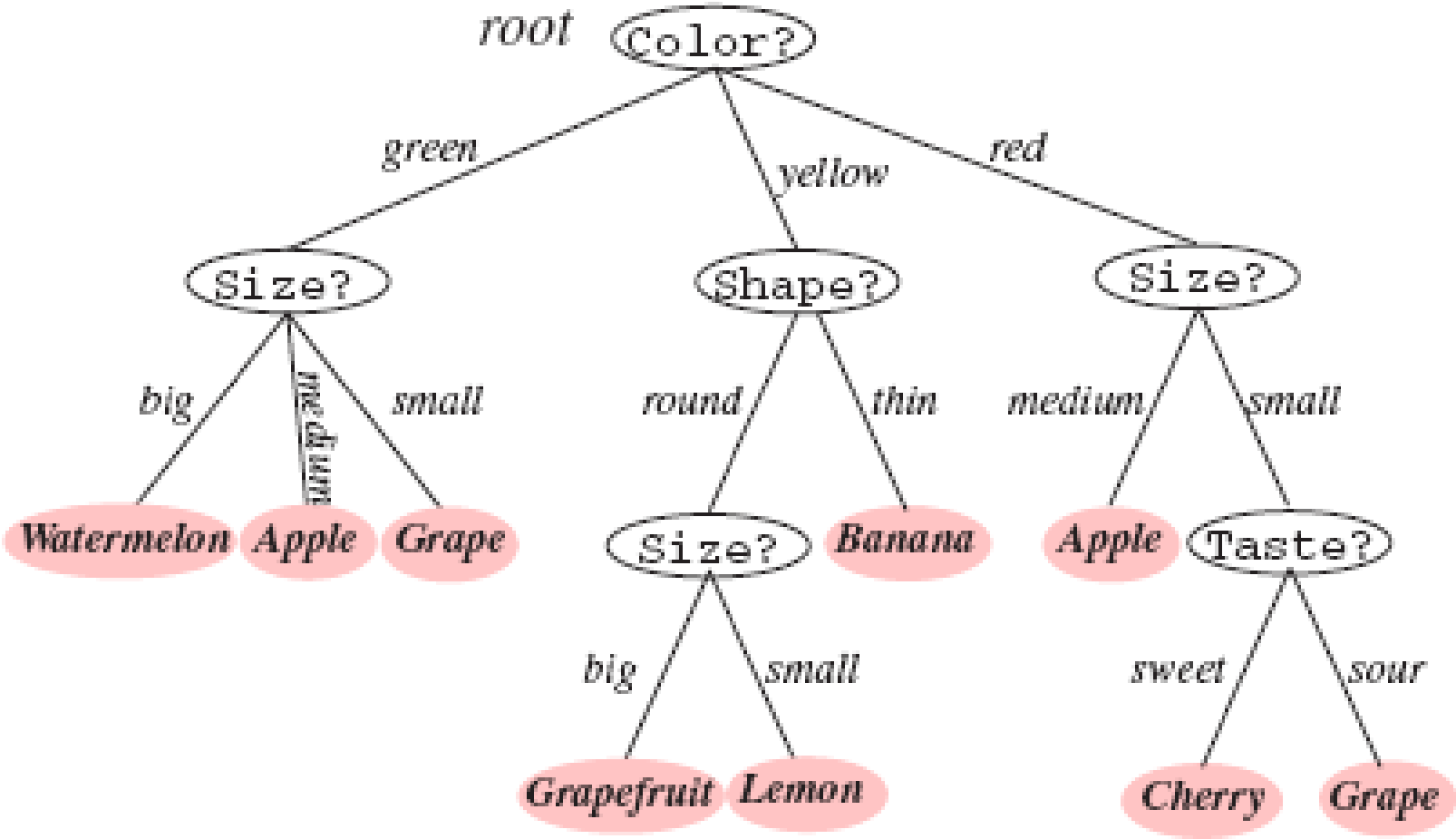
Decision trees

When object is described by a list of attributes, the natural way of classification is the sequence of questions, in which next question depends on the answers given to this point.
A convenient representation of this sequence is the tree (the decision tree).
Nodes of this tree (not counting leaves) are representing the questions – decision criterions.
Branches are answers to the individual questions.
Leaves of this tree contain information about categories (or classes) of the categorized patterns – these are final classifier decisions.

How to grow the decision tree?

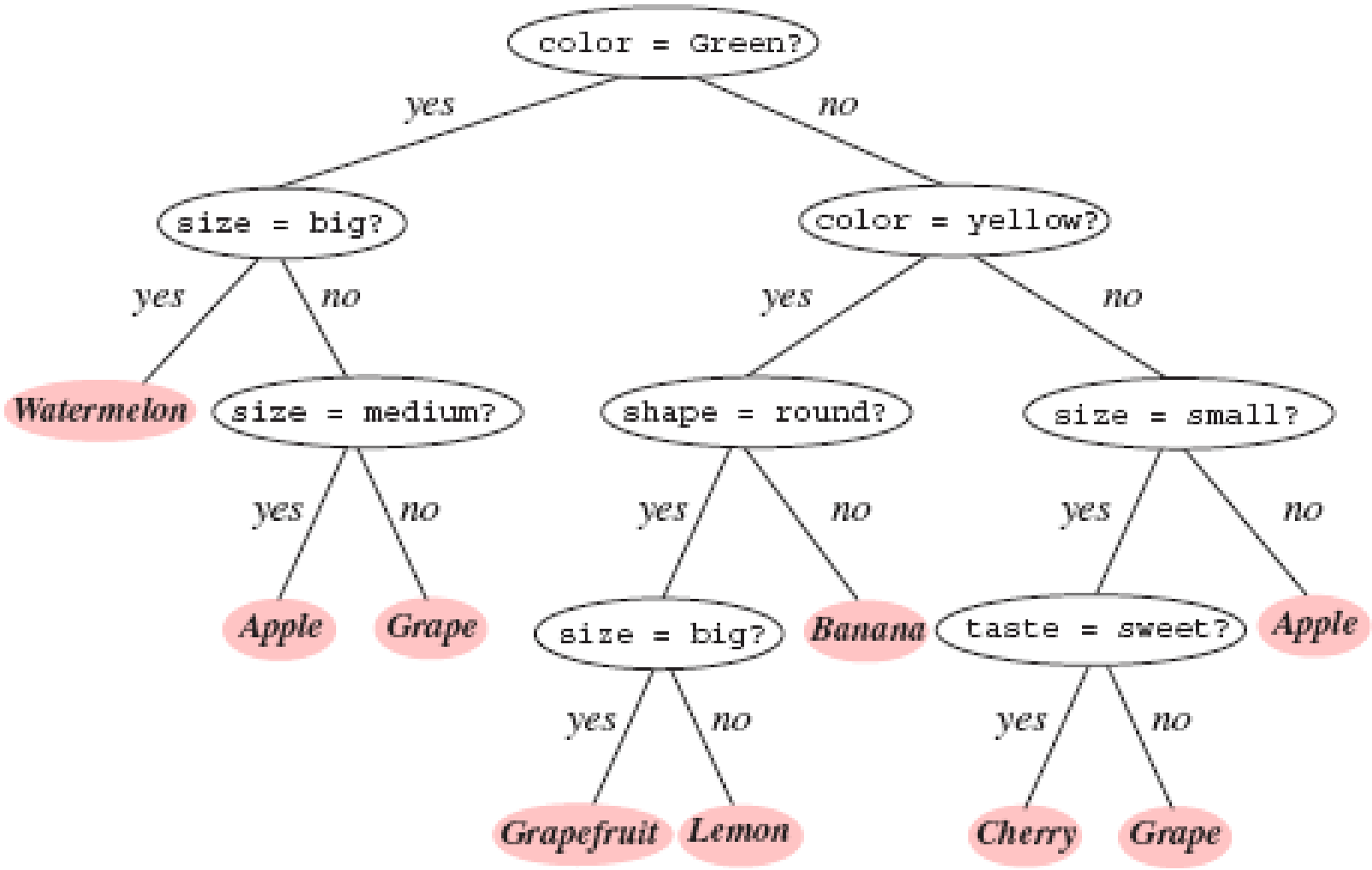
Preparing of these materials was supported by a grant from Liechtenstein, Iceland and Norway through EEA Grants and Norway Grants and co-financed by domestic funds

Example of the decision tree



Preparing of these materials was supported by a grant from Liechtenstein, Iceland and Norway through EEA Grants and Norway Grants and co-financed by domestic funds

Example of the decision tree



Preparing of these materials was supported by a grant from Liechtenstein, Iceland and Norway through EEA Grants and Norway Grants and co-financed by domestic funds

Decision trees

Let's assume that we have training set *D* (the data have class labels, the feature set is determined).
Each *internal* node of the tree splits the set D into two subsets.
If in the descendant node (sub)set is homogenous (i.e. all elements belong to one class), then we mark such node as a leaf.
In the case of non-homogenous set we can:

- Stop splitting, creating the node and computing the proper class label,
- Select another feature and split the set further.

This obvious recursive procedure can be generalized to the classification and regression trees.
(*CART classification and regression tree*)

Preparing of these materials was supported by a grant from Liechtenstein, Iceland and Norway through EEA Grants and Norway Grants and co-financed by domestic funds

CART - Problems

1. Should the properties be restricted to binary-valued or allowed to be multi-valued?
   How many decision outcomes or splits will there be at a node?
2. Which property should be tested at a node?
3. When should a node be declared a leaf?
4. If the tree becomes “too large”, how can it be made smaller and simpler (pruned)?
5. If a leaf node is impure, how should the category label be assigned?
6. How should missing data be handled?

Preparing of these materials was supported by a grant from Liechtenstein, Iceland and Norway through EEA Grants and Norway Grants and co-financed by domestic funds

Number of splits

The number of splits at a node is closely related to question 2, specifying which particular split will be made at a node. In case of binary properties we can have only two descendant nodes. In case of other (non-binary) properties this number is often equal to the cardinality of a property.
What's more, decision criteria can be formulated with the use of more than one property – in general we can produce any number of descendants.
However, every decision (and hence every tree) can be represented by a sequence of binary decisions. (Every tree can be replaced by a binary tree with suitably larger number of nodes).
In practice we use binary trees (almost) always.

Preparing of these materials was supported by a grant from Liechtenstein, Iceland and Norway through EEA Grants and Norway Grants and co-financed by domestic funds

Query selection – node impurity

Which property test or query should be taken at given node?
We prefer decisions that lead to a simple, compact tree with few nodes. We seek a property query *T* at a node *N* that makes the data reaching the immediate descendant as "pure" as possible.
Let *i(N)* denote the impurity of the node *N*.
Entropy impurity (information impurity):

$$i(N) = -\sum_j P(\omega_j)\log_2 P(\omega_j)$$

Gini impurity:

$$i(N) = \sum_{i \neq j} P(\omega_i)P(\omega_j) = \frac{1}{2}\left[1 - \sum_j P^2(\omega_j)\right]$$

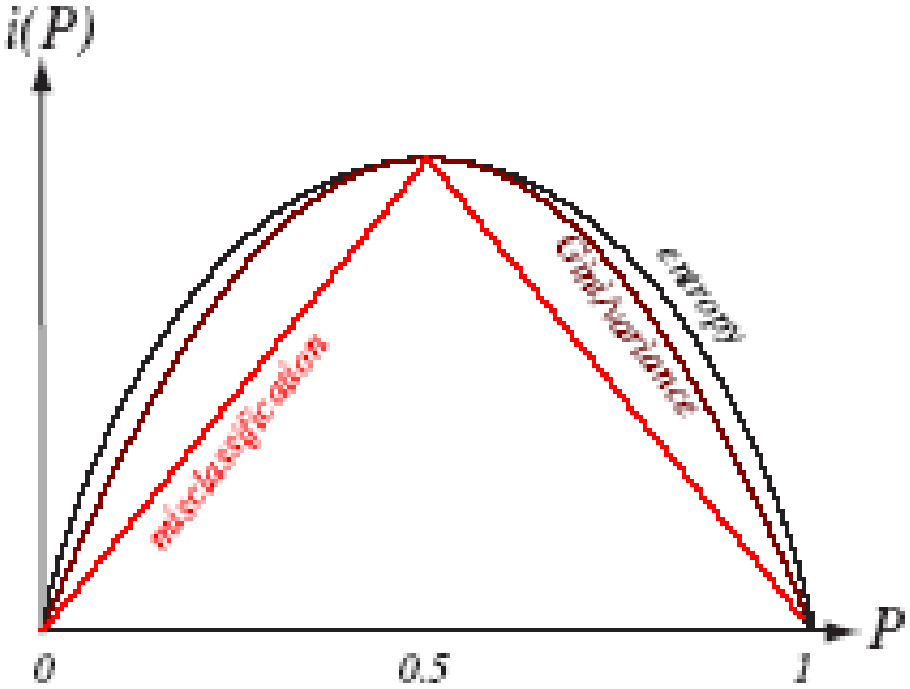
Misclassification impurity:

$$i(N) = 1 - \max_j P(\omega_j)$$

Preparing of these materials was supported by a grant from Liechtenstein, Iceland and Norway through EEA Grants and Norway Grants and co-financed by domestic funds

Node impurity

For the two-category case, the impurity functions peak at equal class frequencies.
(Functions' values on the diagram below are "normalized" for comparison).



Preparing of these materials was supported by a grant from Liechtenstein, Iceland and Norway through EEA Grants and Norway Grants and co-financed by domestic funds

Query selection – node impurity

The decrease in impurity is defined by:

$$\Delta i(N) = i(N) - P_L i(N_L) - P_R i(N_R)$$

The best criterion maximizes $\Delta i(N)$ value.

What is the maximum $\Delta i(N)$ *in the binary tree (two category case)?*

In case of nominal properties we can perform exhaustive search over all possible subsets of the training set.

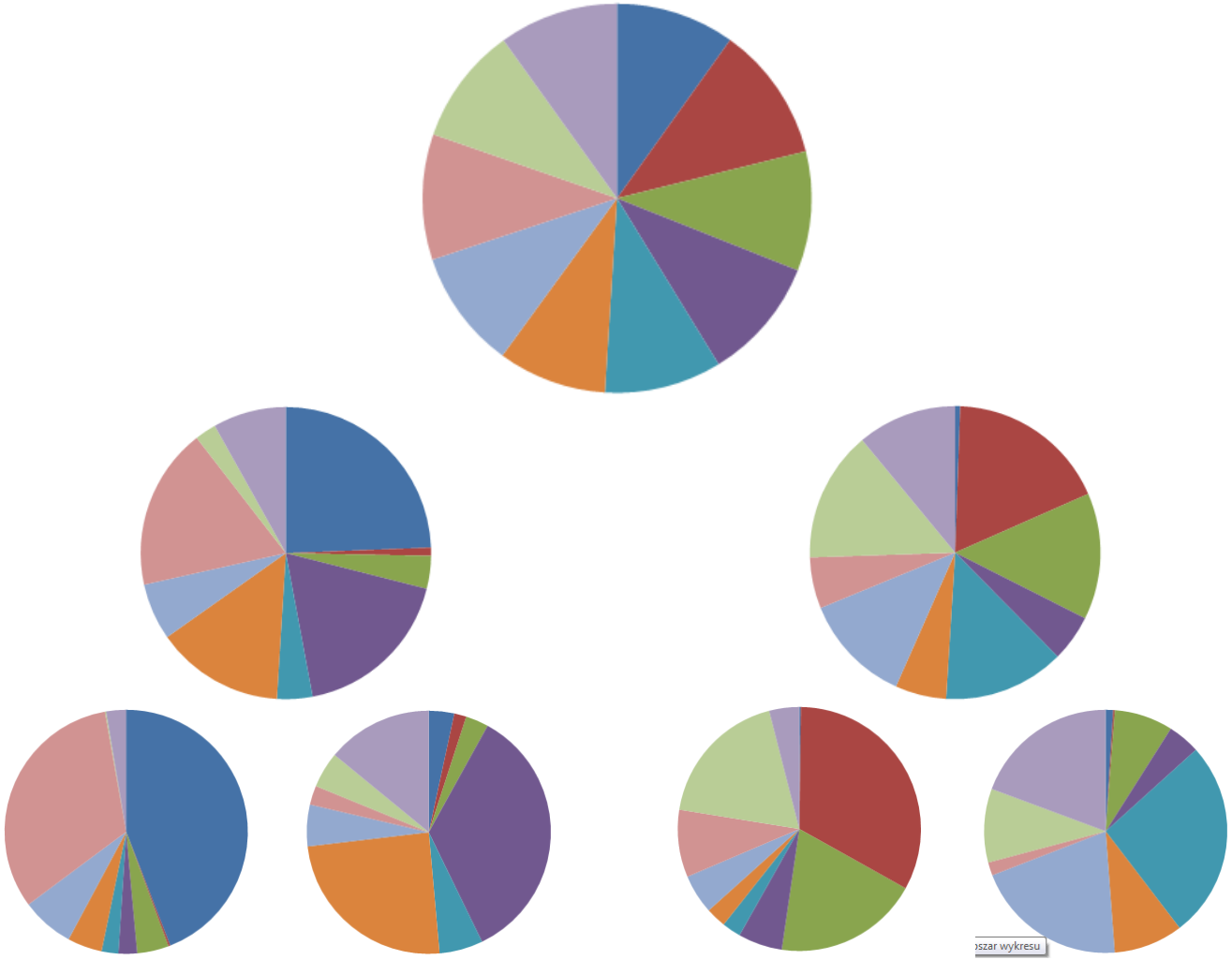
For continuous properties we can use gradient descent algorithm to find splitting hyper-plane.

(If the split is binary the problem is generally one-dimensional optimization problem; when splits are multiple we have two- or more- dimensional problem).

This is a **local** optimization!

Preparing of these materials was supported by a grant from Liechtenstein, Iceland and Norway through EEA Grants and Norway Grants and co-financed by domestic funds

An example of entropy reduction



Preparing of these materials was supported by a grant from Liechtenstein, Iceland and Norway through EEA Grants and Norway Grants and co-financed by domestic funds

When to stop splitting

We select threshold value of the impurity reduction β. If the best of splits yields the impurity reduction lower than our threshold, i.e. $\max_s \Delta i(s) \leq \beta$, we stop splitting.

+ Stop splitting decision is *local*, i.e. leaves can be present on different levels of a tree.
- What value select for β? In general, there is no simple dependency between β value and recognition quality.

We stop splitting when there are no more than *d* elements of the set *D* left at a node *N*.

Preparing of these materials was supported by a grant from Liechtenstein, Iceland and Norway through EEA Grants and Norway Grants and co-financed by domestic funds

When to stop splitting

Let's define *global* tree criterion function: $\alpha \, size + \sum_{leaf} i(N)$

We stop splitting, when this criterion's minimum is reached.
Again we have problem with the parameter value: in this case α .

Statistics approach. During tree construction we estimate the distribution of all the Δi for the current collection of nodes. For any candidate node split, we then determine whether it is "meaningful" – that is, it is significantly different from a random split.
Suppose that at node N we have n patterns, with n_1 in ω_1 and n_2 in ω_2. Suppose a particular candidate split *s* sends *P*n* patterns to the left branch and sends (1-*P*)**n* patterns to the right branch. We test null hypothesis that the split was random, i.e. $P*n_1$ patterns of ω_1, and $P*n_2$ patterns of ω_2 were sent to the left.

Preparing of these materials was supported by a grant from Liechtenstein, Iceland and Norway through EEA Grants and Norway Grants and co-financed by domestic funds

When to stop splitting

We quantify the deviation of the results due to candidate split *s* from the random split by means of the χ^2 (*chi-squared*) statistics, which in two-category case is:

$$\chi^2 = \sum_{i=1}^{2} \frac{(n_{iL} - n_{ie})^2}{n_{ie}}$$

where n_{iL} is the number of patterns in category ω_i sent to the left under decision *s*, and $n_{ie}=Pn_i$ is the number expected by the random rule.

When χ^2 is greater then a critical value, we can reject the null hypothesis because *s* differs “significantly” at some probability or *confidence* level. The critical values depend upon the desired confidence level and the number of degrees of freedom. (In our example this number is 1, because *single* value n_{1L} specifies all other values).

Preparing of these materials was supported by a grant from Liechtenstein, Iceland and Norway through EEA Grants and Norway Grants and co-financed by domestic funds

Pruning

Occasionally, stopped splitting suffers from the lack of sufficient look ahead, a phenomenon called *horizon effect*. The determination of the optimal split at a node *N* is not influenced by decisions at *N*'s descendant nodes.
The principal alternative approach is to fully grow a tree (when leaves have minimum impurity) and then perform pruning of a tree.
All pairs of neighbouring leaf nodes (i.e., ones linked to a common antecedent node, one level above) are considered for elimination. If such *merging* or *joining* yields small increase in impurity it is performed, and an ancestor becomes a leaf...
After such pruning process a tree is unbalanced, with leaf nodes lying in a wide range of levels.
Such a solution for big training sets can be prohibitively costly in processor time and memory.

Preparing of these materials was supported by a grant from Liechtenstein, Iceland and Norway through EEA Grants and Norway Grants and co-financed by domestic funds

Pruning

A conceptually different pruning method is based on *rules*. Each leaf has an associated rule – the conjunction of the individual decisions from the root node, through the tree, to the particular leaf. Thus the full tree can be described by a large list of rules: one for each leaf. Some of these rules can be simplified if a series of decisions is redundant. Eliminating the irrelevant precondition rules simplifies the description, but has no influence on classification.
The basic reason of pruning is to improve accuracy of classification. In this case we eliminate rules so as to improve accuracy on the validation set. One of the benefits of rule pruning is that it allows us to distinguish between the contexts in which any particular node *N* is used. (We can eliminate it for some test pattern x_1 and retain it for another test pattern x_2; in traditional node pruning we must either keep node *N* or prune it away).

Preparing of these materials was supported by a grant from Liechtenstein, Iceland and Norway through EEA Grants and Norway Grants and co-financed by domestic funds