CONSTRUCTING COMPREHENSIBLE AND ACCURATE CLASSIFIERS USING DATA MINING ALGORITHMS

Rok Piltaver

Doctoral Dissertation
Jožef Stefan International Postgraduate School
Ljubljana, Slovenia

Supervisor: Prof. Dr. Matjaž Gams, Jožef Stefan Institute, Ljubljana, Slovenia
Co-Supervisor: Dr. Mitja Luštrek, Jožef Stefan Institute, Ljubljana, Slovenia

Evaluation Board:
Prof. Dr. Marko Bohanec, Chair, Jožef Stefan Institute, Ljubljana, Slovenia
Prof. Dr. Sašo Džeroski, Member, Jožef Stefan Institute, Ljubljana, Slovenia
Prof. Dr. Sanda Martinčić – Ipšić, Member, University of Rijeka, Rijeka, Croatia

MEDNARODNA PODIPLOMSKA ŠOLA JOŽEFA STEFANA
JOŽEF STEFAN INTERNATIONAL POSTGRADUATE SCHOOL



Rok Piltaver

CONSTRUCTING COMPREHENSIBLE AND ACCURATE CLASSIFIERS USING DATA MINING ALGORITHMS

Doctoral Dissertation

GRADNJA RAZUMLJIVIH IN TOČNIH KLASIFIKATORJEV Z ALGORITMI ZA ISKANJE ZAKONITOSTI V PODATKIH

Doktorska disertacija

Supervisor: Prof. Dr. Matjaž Gams

Co-Supervisor: Dr. Mitja Luštrek

Ljubljana, Slovenia, August 2016

To my family, past, present, and future.

Acknowledgments

I would like to thank my supervisor Prof. Dr. Matjaž Gams for giving me the opportunity to work on interesting artificial-intelligence problems in the academic environment provided by the Jožef Stefan Institute – Department of Intelligent Systems, which gave me access to valuable learning resources and support of a team of AI researchers. I thank him for suggesting the dissertation topic and countless discussions in which we confronted our opinions to forge the ideas behind the original contributions of this thesis.

I thank my co-supervisor Dr. Mitja Luštrek who was also my role-model. I thank him for teaching me how to organize and execute research work and how to write research papers and project proposals. His well-argumented criticism turned out to be an indispensable source of knowledge that cannot be found in books.

I thank Prof. Dr. Sanda Martinčić – Ipšić for giving me the opportunity to work at the University of Rijeka – Department of Informatics where I did a big part of research presented in this thesis. I also thank her for motivating me not only with words but also with long weekends we spent in the office analysing research results and polishing papers.

I would also like to thank my family – without them, this thesis would never have been printed. My parents Jože and Majda Piltaver supported me during the graduate study of computer science and stood by my side all along the way. My dear wife Dr. Ivna Kavre Piltaver tried every trick in the book and when they did not work came up with new motivation approaches until she pushed me to enter the last dot of this thesis. I thank her for all the understanding, support and the time I invested in research instead of in our relationship.

I thank psychologists Asst. Prof. Dr. Jana Krivec and Amir Tokić and data-mining experts Prof. Dr. Ljupčo Todorovski and Prof. Dr. Ana Meštrović for discussing survey design and suggesting improvements. I thank colleagues and students at the University of Rijeka, Department of Informatics, and at the Jožef Stefan Institute for answering the lengthy survey.

I thank my colleagues at the Jožef Stefan Institute – Department of Intelligent Systems with whom I worked and learned from. I am grateful to Dr. Domen Marinčič for believing in me, for giving me valuable advices, and for making serious work feel like fun. I thank Prof. Dr. Bogdan Filipič and Dr. Tea Tušar for discussions about multi-objective optimisation. I would also like to thank Dr. Ivna Kavre Piltaver, mag. Eva Černčič and Jernej Zupančič for checking the algorithm correctness proofs and advice on mathematical notation.

Finally, I would like to thank the Jožef Stefan Institute for the PhD scholarship, the Slovene Human Resources and Scholarship Fund for co-funding the international research cooperation that made research about tree comprehensibility possible, and the European Association for Artificial Intelligence for the ECCAI Travel Award that enabled me to present research results at the European Conference on Artificial Intelligence.

Abstract

Classifiers obtained using a machine-learning algorithm can be evaluated by several criteria. This thesis considers two criteria that are usually treated as the most important when deciding which classifier to apply in practice: comprehensibility (i.e., understandability or interpretability) and accuracy. While it is well understood which methods and metrics for estimating a classifier's accuracy should be used in a given application, it is more challenging to measure comprehensibility because the literature on the topic is scarce.

The first part of this thesis is focused on the comprehensibility of classification trees. It systematically investigates the influence of tree-structure parameters (the number of leaves, branching factor, tree depth) and visualisation properties on the tree comprehensibility. In addition, we analyse the influence of the question-depth (the depth of the deepest leaf that is required when answering a question about a classification tree), which turns out to be the most important – while generally overlooked – parameter influencing the comprehensibility. The analysis is based on empiric data obtained using a carefully designed survey with 98 questions answered by 69 respondents grouped according to their data-mining background. We evaluate several tree-comprehensibility metrics and propose two new metrics derived from the structure of a tree and user-provided weights, which enable considering the semantics of the tree in addition to the tree structure itself.

The second problem is that most of the existing machine-learning algorithms are focused primarily on the classifier accuracy, while the ones that take comprehensibility into account as well, usually treat it as less important. Therefore, in many classification domains the user is forced to choose between a comprehensible classifier with relatively low accuracy or an incomprehensible but relatively accurate classifier.

The second part of the thesis deals with the problem of finding classifiers that are accurate and comprehensible at the same time by proposing a multi-objective machine-learning algorithm, which treats both objectives as equally important and returns the set of non-dominated classifiers. We propose a hybrid tree as a classification model, which provides a well-covered range of trade-offs between the two conflicting objectives. A hybrid tree is similar to a classification tree, with an exception that some of its leaves use a black-box classifier, which enables trading comprehensibility for accuracy. A corresponding hybrid-tree comprehensibility metric based on the results from the first part of the thesis is also proposed. The proposed algorithm for multi-objective learning of hybrid classifiers (MOLHC) can examine considerably larger hybrid-trees than a naive exhaustive search algorithm, which faces combinatorial explosion. We show that the MOLHC algorithm is preferable to general multi-objective optimisation algorithms, because it does not require any parameter settings (which are difficult to set for the optimisation algorithm), is formally proven to find the complete set of non-dominated hybrid trees (while the optimisation algorithm only tries to converge to it) and has shorter run-times for reasonably sized trees. A case study in the activity-recognition domain shows that the MOLHC algorithm offers new insights into the classification problem and learns classifiers with a good trade-off between accuracy and comprehensibility.

Povzetek

Klasifikatorje, zgrajene z algoritmi za strojno učenje, ocenjujemo po različnih kriterijih. V tej doktorski disertaciji obravnavamo kriterija, ki sta običajno najpomembnejša pri izbiri klasifikatorja za praktično uporabo: razumljivost in točnost. Čeprav je dobro znano, katere postopke in mere je smiselno uporabiti za ocenjevanje pričakovane točnosti klasifikatorja glede na namen uporabe, je merjenje razumljivosti klasifikatorja dosti zahtevnejše, saj je znanstvena literatura na tem področju pomanjkljiva.

Prvi del disertacije je posvečen razumljivosti klasifikacijskih dreves. V njem smo sistematično analizirali vpliv strukture drevesa (število listov, faktor vejitve in globina drevesa) in načina vizualizacije na njegovo razumljivost. Poleg tega smo analizirali tudi vpliv globine vprašanja (globina najglobljega lista v drevesu, ki ga je potrebno upoštevati, da bi odgovorili na vprašanje o drevesu), ki se je izkazal za dejavnik z največjim vplivom na razumljivost in je v sorodnem delu večinoma spregledan. Analiza temelji na empiričnih podatkih, ki smo jih pridobili na podlagi skrbno zasnovane ankete z 98 vprašanji, na katero je odgovorilo 69 anketirancev, razdeljenih v skupine glede na njihovo znanje s področja odkrivanja zakonitosti v podatkih. Primerjali smo tudi več mer razumljivosti klasifikacijskih dreves in predlagali dve novi meri, ki sta izvedeni iz strukture drevesa in uteži, ki omogočajo upoštevanje pomena informacij v drevesu in ne le njegove strukture.

Druga pomanjkljivost obstoječih algoritmov za strojno učenje je, da se osredotočajo predvsem na točnost, medtem ko algoritmi, ki upoštevajo tudi razumljivost, le-to obravnavajo kot manj pomembno. Zaradi tega so uporabniki v mnogih domenah prisiljeni izbrati med razumljivim klasifikatorjem z relativno nizko točnostjo ali nerazumljivim klasifikatorjem z relativno visoko točnostjo.

Drugi del disertacije rešuje problem iskanja klasifikatorjev, ki so hkrati točni in razumljivi, na podlagi večkriterijskega strojnega učenja, ki obravnava oba kriterija kot enako pomembna in vrne množico nedominiranih klasifikatorjev. V ta namen uvedemo hibridno drevo kot klasifikacijski model, ki ponuja dobro pokrit razpon klasifikatorjev glede na kompromis med nasprotujočima si kriterijema. Hibridno drevo je podobno klasifikacijskemu drevesu, le da v nekaterih listih za klasifikacijo uporablja nerazumljiv klasifikator. To omogoča, da na račun zmanjšane razumljivosti poveča točnost. Za merjenje razumljivosti hibridnih dreves predlagamo mero, ki je zasnovana na podlagi rezultatov iz prvega dela disertacije. Najpomembnejši doprinos je algoritem za večkriterijsko učenje hibridnih klasifikatorjev (MOLHC), ki lahko obravnava bistveno večja drevesa kot naivni preiskovalni algoritem, ki se sooča s kombinatorično eksplozijo. Pokazali smo, da je algoritem MOLHC primernejši od splošnega algoritma za večkriterijsko optimizacijo, ker ne zahteva nastavljanja parametrov (ki jih je za optimizacijski algoritem težko nastaviti), zagotovo vrne celotno množico nedominiranih klasifikatorjev (medtem ko optimizacijski algoritem v najboljšem primeru zagotavlja le konvergenco k tej množici) in ima za smiselno velika drevesa krajši čas izvajanja. Na primeru uporabe v domeni prepoznavanja aktivnosti smo pokazali, da algoritem MOLHC omogoča dodaten vpogled v dani klasifikacijski problem in na izbiro ponuja klasifikatorje z dobrim razmerjem med razumljivostjo in točnostjo.

Contents

List of Figures xv

List of Tables xvii

List of Algorithms xix

Abbreviations xxi

Symbols xxiii

Glossary xxv

1 Introduction 1
- 1.1 Goals and Hypotheses . . . 3
- 1.2 Original Contributions . . . 4
- 1.3 Organization of the Thesis . . . 5

2 Classifier Comprehensibility 7
- 2.1 Introduction . . . 7
- 2.2 Related Work . . . 9
- 2.3 Survey Design and Implementation . . . 11
  - 2.3.1 General Design . . . 11
  - 2.3.2 Preventing Survey Bias . . . 14
  - 2.3.3 Varying the Tree Structure . . . 16
  - 2.3.4 Varying the Tree Visualisation . . . 18
  - 2.3.5 Survey Implementation . . . 20
    - 2.3.5.1 Answering a Survey Question . . . 20
    - 2.3.5.2 Implementation . . . 20
    - 2.3.5.3 Question-Difficulty Scale . . . 22
    - 2.3.5.4 Classification-Tree Comparison Scale . . . 22
    - 2.3.5.5 Classification-Tree Comprehensibility Scale . . . 23
- 2.4 Results and Discussion . . . 23
  - 2.4.1 General Data About the Survey . . . 23
  - 2.4.2 Survey Respondents and the Performance of the Respondent Groups 25
  - 2.4.3 The Influence of Tree-Structure Parameters and Question-Depth . . 28
    - 2.4.3.1 The *Classify Task* . . . 31
    - 2.4.3.2 The *Explain Task* . . . 33
    - 2.4.3.3 The *Validate Task* . . . 35
    - 2.4.3.4 The *Discover Task* . . . 36
  - 2.4.4 The Influence of Tree-Visualisation Properties . . . 36
  - 2.4.5 The Comprehensibility Metrics . . . 39
- 2.5 Conclusions and Future Work . . . 42

3 Learning Comprehensible and Accurate Hybrid Trees 45
3.1 Introduction . . . 45
3.2 Related Work . . . 47
3.3 The Suggested Knowledge Representation and Approach . . . 48
3.4 Multi-Objective Learning of Hybrid Classifiers . . . 51
3.4.1 Preliminaries . . . 51
3.4.2 The Algorithm . . . 52
3.4.3 Data-Mining Process for Multi-Objective Learning . . . 55
3.5 Algorithm Evaluation . . . 58
3.5.1 Run-Time . . . 58
3.5.2 Testing Datasets . . . 58
3.5.3 Comparison with the Baseline Classifiers . . . 59
3.5.4 Comparison with a Multi-Objective Optimisation Algorithm . . . 62
3.6 Application of the MOLHC Algorithm . . . 63
3.7 Analysis of the Algorithm . . . 65
3.7.1 The Number of Hybrid Trees in the Pareto Set . . . 65
3.7.2 Errors in the Estimated Comprehensibility and Accuracy . . . 66
3.8 Conclusion and Future Work . . . 68

4 Conclusions 71
4.1 Summary . . . 71
4.1.1 Classifier Comprehensibility . . . 71
4.1.2 Learning Comprehensible and Accurate Classifiers . . . 72
4.2 Original Contributions . . . 72
4.3 Future Work . . . 74

Appendix A Graphs About Parameters that Influence Tree Comprehensibility 75

Appendix B Survey Question Examples 83

Appendix C The Trees Used in the Survey 91
C.1 Trees with Branching Factor 2 . . . 91
C.2 Trees with Branching Factor 3 . . . 92
C.3 Trees with Branching Factor 4 . . . 93
C.4 Trees Used in the *Discover Task* . . . 94

Appendix D Algorithm Correctness Proofs 97
D.1 Replacing Only Non-Dominated Leaves Produces the Complete Pareto Set . 97
D.2 Some Generated Trees May Be Dominated . . . 103
D.3 Calculating Non-Dominated Leaves . . . 104

References 109

Bibliography 115

Biography 119

List of Figures

Figure 2.1: The basic classification tree . . . 17
Figure 2.2: Tree with branching factor 3 that corresponds to the tree in Figure 2.1 18
Figure 2.3: The tree shows an unusual property for a *bird*: *not airborne* . . . 18
Figure 2.4: Tree visualisation styles . . . 19
Figure 2.5: Subtrees that illustrate the information given in each node . . . 19
Figure 2.6: Two trees with the same structure but different layouts . . . 20
Figure 2.7: An example of a web page with a survey question . . . 21
Figure 2.8: Median time-to-answer vs the mean question-difficulty . . . 25
Figure 2.9: Mean answer-correctness vs the mean question-difficulty . . . 26
Figure 2.10: Mean answer-correctness vs median time-to-answer . . . 27
Figure 2.11: The influence of the question-depth on the time-to-answer (classify) . . 29
Figure 2.12: The influence of the question-depth on the question-difficulty (classify) 32
Figure 2.13: The influence of the question-depth on the answer-correctness (explain) 34
Figure 2.14: The influence of the tree visualisation style on the tree comprehensibility 37
Figure 2.15: The influence of visualised information on the tree comprehensibility . . 37
Figure 2.16: The influence of the tree structure on the tree comprehensibility 38
Figure 2.17: Respondents' subjective ratings of the tree comprehensibility 39
Figure 2.18: Evaluation of comprehensibility metrics . . . 41

Figure 3.1: Data-mining process for multi-objective learning . . . 46
Figure 3.2: An example of a hybrid tree . . . 49
Figure 3.3: Pareto sorting of leaves . . . 53
Figure 3.4: Directly dominated leaves . . . 54
Figure 3.5: Run-times of the MOLHC and naive algorithm . . . 59
Figure 3.6: Pareto fronts for iris and activity-recognition datasets . . . 60
Figure 3.7: Output of the MOLHC algorithm for the activity-recognition domain . 64
Figure 3.8: Number of hybrid trees in a Pareto set . . . 67
Figure 3.9: Error in predicted comprehensibility of hybrid trees . . . 68
Figure 3.10: Error in predicted accuracy of hybrid trees . . . 68

Figure A.1: The influence of the number of leaves on the time-to-answer (classify) . 76
Figure A.2: The influence of the number of leaves on the question-difficulty (classify) 77
Figure A.3: The influence of the branching factor on the time-to-answer (classify) . 78
Figure A.4: The influence of the branching factor on the question-difficulty (classify) 79
Figure A.5: The influence of the question-depth on the time-to-answer (explain) . . 80
Figure A.6: The influence of the question-depth on the question-difficulty (explain) 81

Figure B.1: An example of a question in the *classify task* . . . 84
Figure B.2: An example of a question in the *explain task* . . . 85
Figure B.3: An example of a question in the *validate task* . . . 86
Figure B.4: An example of a question in the *discover task* . . . 87

Figure B.5: An example of a question in the *compare task* 88
Figure B.6: An example of a question in the *rate task* 89

Figure C.1: Trees with 3 or 4 leaves, branching factor 2 and tree depth 2 or 3 . . . 91
Figure C.2: The basic tree with 7 leaves, branching factor 2 and tree depth 5 92
Figure C.3: Trees with 9 or 10 leaves, branching factor 2 and tree depth 7 92
Figure C.4: Tree with 3 leaves, branching factor 3 and tree depth 1 92
Figure C.5: Tree with 7 leaves, branching factor 3 and tree depth 3 93
Figure C.6: Trees with 9 leaves, branching factor 3 and tree depth 2 or 4 93
Figure C.7: Tree with 4 leaves, branching factor 4 and tree depth 1 93
Figure C.8: Tree with 7 leaves, branching factor 4 and tree depth 2 94
Figure C.9: Tree with 10 leaves, branching factor 4 and tree depth 3 94
Figure C.10: Small trees showing an unusual property of *mammals* and *birds* 94
Figure C.11: Medium tree showing unusual properties of *mammals* and *fish* 95
Figure C.12: Big tree showing unusual properties of *mammals*, *insects*, and *reptiles* . 95

Figure D.1: Lemma D.1 proof illustration 97
Figure D.2: Lemma D.2 and D.3 proofs illustration 99
Figure D.3: Lemma D.4 proof illustration 100
Figure D.4: Lemma D.5 proof illustration (part i) 101
Figure D.5: Lemma D.5 proof illustration (part ii) 102
Figure D.6: The MOLHC algorithm may generate some dominated trees 103
Figure D.7: Calculating the set of non-dominated regular leaves 104
Figure D.8: The relations between subsets of regular leaves in a hybrid tree 106

List of Tables

Table 2.1: The investigated tree parameters and measures used for each survey task 14
Table 2.2: The structure of the trees used in the survey. 16
Table 2.3: The statistical data about the size of the survey 24
Table 2.4: The number of respondents by gender, language and experience. 27
Table 2.5: The performance of respondent groups in each task 28
Table 2.6: Summary of the influence of tree structure on comprehensibility 30
Table 2.7: The influence of negated statements on the respondents' performance . . 35
Table 2.8: Evaluation of tree comprehensibility metrics according to the Spearman's ρ 41
Table 2.9: Evaluation of tree comprehensibility metrics according to the Pearson's r 41

Table 3.1: General data about the 21 datasets 61
Table 3.2: Comparison of the MOLHC algorithm with baseline 62
Table 3.3: Relative difference in hypervolume between MOLHC and NSGA-II . . . 63

List of Algorithms

Algorithm 3.1: Multi-objective learning of hybrid classiffers (MOLHC) 57

Abbreviations

ANN	... artificial neural network (Rumelhart, Hinton, & Williams, 1986)
CRISP-DM	... cross-industry standard process for data mining
DM	... data mining
IDA	... intelligent data analysis
IQR	... interquartile range
IT	... Information technology
k-NN	... k-nearest neighbours algorithm (Aha, Kibler, & Albert, 1991)
KDD	... knowledge discovery in databases
MOLHC	... algorithm for multi-objective learning of hybrid classifiers
NSGA-II	... non-dominated sorting genetic algorithm for multi-objective optimization
ROC	... receiver operating characteristic
SVG	... scalable vector graphics – an XML-based vector image format
SVM	... support vector machine algorithm (Platt, 1999)

Symbols

Symbols Used in the Classifier Comprehensibility Chapter

ρ ... Spearman's rank correlation coefficient
B ... the branching factor of a tree
b_l ... branching factor of the leaf l in a classification tree
c_{D} ... weighted sum of the depths d_l over all n leaves in a tree
c_{B} ... weighted sum of the branching factors b_i on the paths from the root of the tree to a leaf l over all n leaves
D ... the depth of a tree
d_l ... depth of the leaf l in a classification tree
L ... the number of leaves in a tree
l ... a leaf in a classification tree
n ... the number of leaves in a classification tree
r ... Pearson product-moment correlation coefficient
w_l ... weight assigned to the leaf l (used for computing the tree comprehensibility)

Symbols Used in the Learning Comprehensible and Accurate Hybrid Trees Chapter

a_i ... the accuracy of a hybrid tree T_i
B_i ... the set of black-box leaves in a hybrid tree T_i
c_i ... the comprehensibility of a hybrid tree T_i
D_l ... the set of leaves that are directly dominated by the leaf l
L_i ... the set of non-dominated leaves in a hybrid tree T_i
L_i^c ... the set of dominated leaves in a hybrid tree T_i
n ... the number of leaves considered for replacing with black-box leaves
non-dom(S) ... the subset of non-dominated objects in the set S
N ... the number of instances in the training set
N_l ... the number of training instances belonging to the leaf l
$N_{j,\mathrm{t}}$ ... the number of correctly classified examples in a regular leaf j
$N_{j,\mathrm{bb}}$ ... the number of correctly classified examples in a black-box leaf j
r_s ... the root node of a subtree S in a hybrid tree
R_i ... the set of regular leaves in a hybrid tree T_i
T_i ... a hybrid tree
T_1 ... the initial classification tree used as input to the MOLHC algorithm
T_{2^n} ... the hybrid tree with the maximal number of black box leaves
$x \preccurlyeq y$ ... the Pareto dominance relation: x dominates y
$\delta_{l,\mathrm{a}}$ ... relative difference in accuracy when replacing the leaf l with a black-box leaf

$\delta_{l,\mathrm{c}}$ ... relative difference in comprehensibility when replacing the leaf l with a black-box leaf

Glossary

answer-correctness	...	the probability of the correct answer for a given survey question
black-box classifier	...	an incomprehensible classifier
black-box leaf	...	a leaf in a hybrid tree that classifies instances using a black-box classifier
comprehensibility	...	the ability to understand the logic behind predictions of a model
dominated solution	...	a solution that is dominated, i.e. worse than at least one other solution according to multiple criteria
hybrid tree	...	a classification model that is a hybrid between a classification tree and a black-box classifier that may be used in some of its leaves in order to classify instances that belong to them
incomparable solutions	...	a pair of solution, one is better in according to one criteria while the other is better in some other criteria
interpretability	...	see comprehensibility
multi-objective learning	...	a sub-field of machine learning that treats multiple learning objectives as equally important and applies multi-objective optimisation to solve the learning problem
multi-objective optimisation	...	an optimisation approach that treats multiple optimisation criteria as equally important and returns a set of non-dominated solutions as result in contrast with the single-objective optimisation that weights the importance of the criteria and returns a single solution
optimisation objective	...	a property of solutions that needs to be optimized in an optimization problem
Pareto optimal	...	non-dominated, see Definitions 3.5 and 3.6
Pareto set	...	the set of non-dominated solutions (e.g. classifiers), see Definition 3.7
Pareto front	...	the image of a Pareto set in the objective space
Pareto dominance	...	relation for comparing the quality of solutions according to multiple criteria, see Definition 3.5
non-dominated solution	...	a solution that is not dominated by any other solution, see Definitions 3.5 and 3.6
question-depth	...	the depth of the deepest leaf that is required to answer a question about a classification tree
question-difficulty	...	respondent's subjective opinion about the difficulty of a survey question
regular leaf	...	a leaf in a hybrid tree that does not use the black-box classifier

single-objective optimisation	...	an optimisation approach that deals with a single optimisation criteria in contrast with multi-objective optimisation that deals with multiple optimisation criteria
time-to-answer	...	the time that is required to answer a survey question
understandability	...	see comprehensibility

Chapter 1

Introduction

As the title of this thesis suggests, it deals with *data mining* – an interdisciplinary field that encompasses statistical, pattern recognition, machine learning and visualisation tools to support the analysis of data and discovery of principles hidden within the data. In some communities data mining (DM) is also referred to as *knowledge discovery in databases* (KDD) or *intelligent data analysis* (IDA) (Kononenko & Kukar, 2007). The overall goal of the data mining is to extract information from large quantities of data and if possible transform it into an understandable structure for further use (Chakrabarti et al., 2016). Data-mining process automatically or semi-automatically analyses the input data and extracts previously unknown and interesting patterns such as groups of similar data records, unusual records, dependencies between them, or functions that model the data. The extracted patterns can be seen as a kind of summary of the input data, which can be used, for example, to discover new knowledge or as a predictive model of the process that generated the input data. The *data-mining process* involves several phases: understanding the data-mining project objectives and requirements to define the data-mining problem; data collection and understanding to identify data quality problems and discover first insights into the data; transformation and cleaning of data to prepare it for modelling tools; applying various modelling techniques and calibrating their parameters; evaluating the constructed models to choose the most appropriate one; and finally deployment which can be as simple as presenting the gained knowledge or as complex as implementing a repeatable data-mining process (Chapman et al., 2000).

The modelling phase of data-mining process employs *machine-learning algorithms* that can learn from and make predictions on data (Kohavi & Provost, 1998). Such algorithms operate by building a model from example inputs in order to make data-driven predictions or decisions. Machine learning has shown tremendous improvements over the last 30 years, which is reflected in numerous commercial and open-source systems for machine learning and their application in industry, medicine, economics, natural and technical sciences, ecology, finance and many others (Kononenko & Kukar, 2007).

Machine-learning tasks are classified into three categories, depending on the nature of the input data (Russell & Norvig, 2003). The first is *supervised learning*: the computer is presented with example inputs and their desired outputs, given by a "teacher", and the goal is to learn a general rule that maps inputs to outputs. The second category is *unsupervised learning*: no labels are given to the learning algorithm, leaving it on its own to find structure in its input. Unsupervised learning can be a goal in itself (discovering hidden patterns in data) or a means towards an end (feature learning). Between supervised and unsupervised learning is *semi-supervised learning*, where the "teacher" gives a training set with some (often many) of the target outputs missing. The third category is *reinforcement learning*: a computer program interacts with a dynamic environment in which it must perform a

certain goal (such as driving a vehicle), without a "teacher" explicitly telling it whether it has come close to its goal. Another example of reinforcement learning is learning to play a game by playing against an opponent.

Machine-learning algorithms can also be categorized with respect to how the obtained (learned) knowledge is used (Kononenko & Kukar, 2007). In *classification*, objects that are assigned to exactly one class from a finite set of possible classes are described with attributes (features, properties) and the goal of the algorithm is to produce a model – termed *classifier* – that determines to which class an object belongs based on the attribute values. The learned classifier can then be used to assign class to previously unseen inputs. This is typically tackled in a supervised way. A typical example of a classification task is assigning email messages to classes "spam" and "not spam". This thesis focuses on classification, but there are also other categories of machine-learning algorithms. For example, *regression* is similar to classification, except that the dependant (target, regression) variable is continuous rather than discrete. Another example is *clustering* where the goal of the learning algorithm is to determine coherent subsets (clusters) of learning examples, which are described only with attributes and no supervising (target) variable is given.

Classifiers extracted using machine-learning algorithms are evaluated according to their *predictive accuracy* estimated by the percent of correctly classified instances, confusion matrix, area under ROC curve, F-score, or some other measure. In addition, there are also other classifier properties that are important for the user: *comprehensibility* (also termed understandability or interpretability), justifiability (Martens & Baesens, 2010), surprisingness (Carvalho, Freitas, & Ebecken, 2003), and others. This thesis considers accuracy and comprehensibility, which are prerequisites for most practical applications. Comprehensible classifiers are especially important in domains such as credit scoring, medicine, churn prediction and bioinformatics because they enable domain experts to explain classifications of individual instances, validate the classifier, confirm hypotheses, discover new knowledge and improve or refine classifiers in collaboration with data-mining experts. It is well understood which methods and measures for estimating classifier's predictive accuracy should be used in a given application. However, the literature on the influences of classifier structure and visualisation properties on the comprehensibility and usability is scarce. Therefore, the first part of this thesis investigates the comprehensibility of classification trees. It systematically investigates the influence of tree-structure parameters (the number of leaves, branching factor, tree depth) and visualisation properties on the tree comprehensibility. In addition, we analyse the influence of the question-depth (the depth of the deepest leaf that is required when answering a question about a classification tree), which turns out to be the most important while generally overlooked parameter influencing the comprehensibility. We also evaluate several tree-comprehensibility metrics and propose two new metrics derived from the structure of a tree and user-provided weights, which enable considering the semantics of the tree in addition to the tree structure itself.

Classification trees, rules and tables are well-known for their comprehensibility, but they often lag behind modern machine-learning algorithms in terms of accuracy. More complex classification models such as support vector machine (SVM), artificial neural network (ANN), random forest, and ensembles achieve higher accuracy in many domains but are not comprehensible. The users are therefore faced with a difficult decision: they can either choose a comprehensible classifier with relatively low accuracy or an incomprehensible classifier with relatively high comprehensibility. In many cases none of the two options is acceptable. The second part of this thesis therefore deals with the problem of learning classifiers that have a good trade-off between accuracy and comprehensibility while treating both objectives as equally important. We propose hybrid tree as a classification model that enables trading accuracy for comprehensibility or vice versa. Hybrid tree is similar to

a classification tree except that some of its leaves may use a black-box classifier to classify instances instead of classifying them in the same class. The inspiration comes from nature: experts in various complex domains can often explain parts of their knowledge with simple rules (formal, codified or explicit knowledge) while they fail to formulate (writing down or verbalize) their knowledge in other parts of domains. Experts rely on tacit knowledge (Polanyi, 1966), i.e. their skills, ideas and experienc, when dealing with more complex domain problems. This shows that their knowledge corresponds neither to comprehensible simple rules, nor to incomprehensible complex models. On the contrary, it is structured so that it provides good trade-off between accuracy and comprehensibility. One of the main contributions of the thesis is the Multi-Objective Learning of Hybrid Classifiers algorithm (MOLHC) that learns the set of non-dominated hybrid trees using the Pareto-based approach and is proven to find the complete set of non-dominated hybrid trees. The goal of the MOLHC algorithm is to find a hybrid tree that classifies simple parts of the domain with regular leaves while it uses black-box leaves to classify more complex parts of the domain. In addition, we propose an interactive iterative data-mining process designed around the MOLHC algorithm, which uses a domain expert in the loop in order to combine expert's knowledge and the power of machine learning. The proposed data-mining process was successfully applied in the activity-recognition domain, where it was shown that it enables the expert to make a well informed decision when selecting a classifier with a good trade-off between the accuracy and comprehensibility, and obtain additional insights into the classification domain.

1.1 Goals and Hypotheses

The purpose of this dissertation is to address the problem of learning accurate and comprehensible classifiers. The specific goals of the dissertation are to:

- Analyse the parameters that influence the comprehensibility of classification trees in order to provide insights that will support development of algorithms for learning comprehensible trees, classification tree comprehensibility metrics reflecting human sense of comprehensibility, and methods for visualisation of classification trees with improved readability.

- Develop an algorithm that will divide classification domain into parts that can be accurately classified with comprehensible classification models and parts that require incomprehensible models in order to achieve high accuracy. The algorithm should return the Pareto set of hybrid classifiers providing diverse trade-offs between accuracy and comprehensibility.

- Compare the performance of the developed algorithm with the state-of-the-art algorithms in various domains in order to confirm its advantages and potential limitations for practical application.

The main hypotheses that are proposed, investigated, and confirmed in this thesis are as follows:

- Simple classification tree complexity measures, such as the number of leaves in a tree, do not take into account all the parameters that influence the classification tree comprehensibility.

- In practice, there are classification domains in which parts can be classified with comprehensible models, while the other parts require incomprehensible models in

order to achieve high accuracy. It is possible to develop an algorithm that will outperform the existing algorithms (at least in such domains) in terms of accuracy and comprehensibility of produced classifiers.

1.2 Original Contributions

The thesis leads to two major original scientific contributions and several related contributions listed below.

1. *Systematic analysis of the influence of tree-structure and visualization parameters on classification tree comprehensibility based on survey respondents' performance and subjective opinions.*

   The analysis introduced the following original contributions:

   - A general design of an in-depth survey for empirical study of the classifier comprehensibility based on best-practices and mistakes from related work, cognitive theory and survey-bias prevention strategies. The survey measures respondents' performance with two objective measures (time to answer a survey question and answer correctness) in four data-mining tasks (classify an instance, explain instance classification, verify statement about the domain, discover new knowledge) and respondents' subjective opinions (question-difficulty and absolute and relative tree comprehensibility). The survey design was validated by psychologists, data-mining experts and a test group of respondents.
   - A rich dataset for empirical study of the classification-tree comprehensibility obtained from the end-user survey with 69 data-mining experts and students.
   - Evaluation of most commonly used tree-comprehensibility metrics and two new metrics (the weighted sum of the depths of leaves and the weighted sum of the branching factors on the paths from the root to the leaves) that consider the semantics of the tree in addition to the tree structure itself.
   - Introduction of question-depth concept (the depth of the deepest leaf that is required when answering a question about a classification tree), which is shown to be the parameter with the highest impact on tree comprehensibility.
   - Validation of the number of leaves as a comprehensibility metrics for large trees and empirical proof that it should be avoided for small trees or when comprehensibility of a single part of the tree is of interest.
   - Empirical confirmation of the fact that classification-tree visualization style and layout significantly influence the comprehensibility.

2. *The algorithm for multi-objective learning of hybrid classifiers (MOLHC). The algorithm is formally proven to find the complete set of non-dominated hybrid trees with regard to their accuracy and comprehensibility.*

   The MOLHC algorithm has the following novel properties:

   - It uses a novel hybrid tree classification model that offers a range of trade-offs between tree comprehensibility and accuracy that is not available using the existing algorithms.
   - It generates the entire Pareto-set of hybrid trees in a single run, which enables the user to better understand modelling of the problem domain.

- Its run-time is substantially shorter than the run-time of the naive exhaustive search algorithm enabling application of the MOLHC algorithm in real-life use-cases.
- In comparison to the multi-objective evolutionary optimization algorithm NSGA-II, the MOLHC algorithm does not require any parameter settings, guarantees finding the complete Pareto-set and is faster for moderate-size trees.
- It evaluates the comprehensibility of generated hybrid-trees based on a novel comprehensibility metric computed as the fraction of examples classified by the comprehensible part of the hybrid tree.

The list of publications related to this thesis is given in the Bibliography section.

1.3 Organization of the Thesis

This thesis is divided into two main parts: Chapter 2 is about classification-tree comprehensibility and Chapter 3 is about learning comprehensible and accurate classifiers. Section 2.1 explains classifier comprehensibility, why it is important and why we decided to investigate it. Section 2.2 gives an overview of related work from the field of comprehensibility and compares it with our work in the field. Section 2.3 describes the design and implementation of the survey that we use to analyse the comprehensibility of classification trees. Section 2.4 analyses and discusses the influence of the classification tree properties on tree comprehensibility. Section 2.5 concludes the chapter about comprehensibility and suggests directions for further research. A detailed overview of Chapter 2 is given in the last paragraph of Section 2.1.

Section 3.1 presents the challenges in learning comprehensible and accurate classifiers and discusses our motivation for developing an algorithm and data-mining process that addresses the challenges. Section 3.2 presents the related work from the field of learning comprehensible classifiers and compares it with our work in the field. Section 3.3 defines the hybrid tree that we propose as a classification model that enables trade-offs between accuracy and comprehensibility. Section 3.4 introduces the MOLHC algorithm that finds the complete set of non-dominated hybrid trees efficiently and the multi-objective data-mining process based on the algorithm. Section 3.5 evaluates the MOLHC algorithm. First, algorithm time complexity is analysed and its run-time is shown to be acceptable for practical use, while a naive search algorithm is too slow. Next, the MOLHC algorithm is compared with pairs of traditional machine-learning algorithms (one focusing on comprehensibility and the other on accuracy) to show that MOLHC learns classifiers with accuracy and comprehensibility that cannot be achieved with the compared algorithms. Finally, MOLHC is compared with a well-known general stochastic multi-objective optimisation algorithm to show that MOLHC is easier to use, is guaranteed to find the optimal solutions and is faster for reasonably sized trees. Section 3.6 presents a case-study of MOLHC application in the activity-recognition domain. Section 3.7 analyses the limitations of the MOLHC algorithm. Section 3.8 concludes the chapter about learning accurate and comprehensible classifiers and suggests directions for further research.

Finally, Chapter 4 summarizes the thesis, recalls its original contributions and suggests ideas for future work.

The Appendix includes four sections. Appendix A includes graphs about parameters that influence tree comprehensibility. An example of a question for each of the six survey tasks is shown in Appendix B. All the trees used in the survey are shown in Appendix C. Appendix D formally proves the correctness of the MOLHC algorithm and the equation

that enables efficient computation of the set of non-dominated leaves. The Author's bibliography and biography are given at the end of the thesis.

Chapter 2

Classifier Comprehensibility

2.1 Introduction

Classifier comprehensibility, which is sometimes referred to as interpretability (Freitas, 2014; Huysmans, Dejaeger, Mues, Vanthienen, & Baesens, 2011; Jin & Sendhoff, 2008; Jin, Sendhoff, & Körner, 2005; Maimon & Rokach, 2005a) or understandability (Allahyari & Lavesson, 2011; Pazzani, 2000; Sommer, 1995), is defined as "the ability to understand the logic behind a prediction of a model" (Martens, Vanthienen, Verbeke, & Baesens, 2011) or "how well humans grasp the induced classifier" (Maimon & Rokach, 2005b). It has been recognised as an important classifier property since the 1980s (Michalski, 1983) and is continuously emphasised (Allahyari & Lavesson, 2011; Freitas, 2014; Huysmans et al., 2011; Martens et al., 2011; Sommer, 1995; Zhou, 2005; Quinlan, 1999). For example, Michie stated that one of the main features of ID3-like algorithms is their ability to generate easy-to-understand decision trees (Michie, 1987). Similarly, Kodratoff (1994) recognises the comprehensibility as a decisive factor when machine learning models are applied in the industry. Comprehensible classifiers are especially important in domains such as credit scoring, medicine, churn prediction and bioinformatics (Freitas, 2014) because they enable domain experts to classify instances without using a computer, explain classifications of individual instances, validate the classifier, confirm hypotheses, discover new knowledge and improve or refine classifiers in collaboration with data-mining experts (Craven & Shavlik, 1995; Zhou, 2005).

Classifier comprehensibility depends on the type of knowledge representation that is employed (Freitas, 2014; Huysmans et al., 2011; Johansson, Niklasson, & König, 2004; Martens et al., 2011; Zhou, 2005). For example, classification trees and rules are considered to be the most comprehensible (Freitas, 2014; Johansson et al., 2004; Martens et al., 2011; Zhou, 2005), while support vector machines, artificial neural networks and ensembles of classifiers are considered to be the least comprehensible (Chorowski, 2012) and, hence, are termed black-box classifiers (Freitas, 2014; Huysmans et al., 2011; Johansson et al., 2004; Zhou, 2005). There are differences in the comprehensibility of the classifiers based on the same type of knowledge representation as well (Martens et al., 2011). For example, a lower classifier complexity corresponds to a higher comprehensibility, therefore classifier complexity is often used as a surrogate metric for classifier comprehensibility (Allahyari & Lavesson, 2011; Freitas, 2003, 2004; Jin & Sendhoff, 2008; Jin et al., 2005; Johansson et al., 2004; Martens et al., 2011). The complexity of a specific classifier can be measured as the number of leaves in a tree (Maimon & Rokach, 2005b), conditions in a classification rule set (Sommer, 1995), or connections in a neural network (Jin & Sendhoff, 2008; Liu & Kadirkamanathan, 1995). Moreover, properties, such as the structure of the model and its visualisation, affect the comprehensibility as well (Göpferich, 2009; Huysmans et al., 2011),

but it is not clear how and to what extent. Therefore, the main problem with regard to most classification algorithms is that they do not explicitly consider the comprehensibility (Huysmans et al., 2011; Johansson et al., 2004), while the ones that do usually simplify it to the classifier complexity (Pazzani, 2000). This approach has several drawbacks (Freitas, 2014) and could lead to over-simplistic models (Elomaa, 1994) that are neither accurate, nor comprehensible. This consideration is the motivation for our systematic empirical study of tree properties that potentially influence the comprehensibility of classifiers, in which we tackle classification trees, which are probably the most commonly used type of comprehensible classifiers.

Comprehensibility is inherently subjective (Huysmans et al., 2011; Martens et al., 2011) and hence impossible to measure directly (Allahyari & Lavesson, 2011). Therefore, we analyse the comprehensibility through the lens of classifier usability, which is actually the property that is important in practice: the easier the classifier is to comprehend, the easier it is to use. Therefore, the classifier comprehensibility and usability can be interchanged in this thesis, although in general, the terms are not exact synonyms (Göpferich, 2009). Our study is based on data about the performance of users while solving four types of tasks that involve classification trees and their opinions on the task difficulty and the tree comprehensibility, which was obtained using a carefully designed survey. We collected the answers to 98 questions from 69 respondents and analysed them with statistically sound methodology; we provide the interpretation of the results as well as several empirically supported guidelines on how to construct more comprehensible classification trees. We focus mainly on the influence of the tree structure properties (the number of leaves, branching factor, tree depth) on the comprehensibility, but we also analyse the influence of several tree visualisation properties. One of our most important contributions is the investigation of the influence of the question-depth, which is equal to the depth of the deepest leaf that is required to answer a question about a classification tree. Another improvement over the related work is the comparison of the performance and opinions about the tree comprehensibility, from novice versus expert data-miners. Finally, we propose two new classification-tree comprehensibility metrics better reflecting the human sense of classifier comprehensibility (the weighted sum of the depths of the leaves and the weighted sum of the branching factors on the paths from the root to the leaves). Comprehensibility metrics are required to act as heuristic functions in learning algorithms (Giraud-Carrier, 1998; Piltaver, Luštrek, Zupančič, Džeroski, & Gams, 2014) and to compare the comprehensibility of the classifiers obtained from various algorithms (Piltaver, Luštrek, Zupančič, et al., 2014; Zhou, 2005).

Section 2.2 gives a review of related work about classifier comprehensibility. Section 2.3 explains the survey design and implementation by listing the general design choices and analysed classification tree properties (Section 2.3.1), survey bias prevention strategies (Section 2.3.2), methods for generating the classification trees used in the survey (Section 2.3.3), analysed classification tree visualisation properties (Section 2.3.4), and an example of a survey question, technical details about the survey implementation and scales used in the survey (Section 2.3.5). Section 2.4 presents and discusses the survey results. First, the general data about the survey is described (Section 2.4.1) followed by a discussion on the performance of different survey respondent groups (Section 2.4.2), the influence of the classification tree parameters on the comprehensibility (Section 2.4.3), and the influence of the classification tree visualisation (Section 2.4.4). Finally, existing tree comprehensibility metrics are analysed and compared with the two new metrics (Sections 2.4.5). This chapter closes with a summary of the most interesting findings and suggested directions for further research (Section 2.5).

2.2 Related Work

Although many papers emphasise the importance of classifier comprehensibility (Freitas, 2014; Kodratoff, 1994; Martens et al., 2011; Michalski, 1983; Michie, 1987; Quinlan, 1999; Sommer, 1995; Zhou, 2005), related work on classifier comprehensibility is relatively scarce (Allahyari & Lavesson, 2011; Pazzani, 2000). The most general related work comes from the field of cognitive science. Cognitive load theory (Sweller, 1988) divides the total amount of mental effort that is used in working memory into three types: the intrinsic cognitive load is inherent to the specific topic and cannot be altered (the complexity of the classification domain); the extraneous cognitive load depends on the way that information or tasks are presented (the classifier representation); and the germane cognitive load (Sweller, Van Merrienboer, & Paas, 1998) refers to the work put into creating a permanent store of knowledge, i.e. the processing and construction of schemas – mental structures that organise the categories of information and their relationships (DiMaggio, 1997) into a permanent store of knowledge. Research in this field has developed several measures of cognitive load (DeLeeuw & Mayer, 2008). For example, Paas and Van Merriënboer (1993) developed a construct (known as relative condition efficiency) for measuring the perceived mental effort (an index of cognitive load). This motivated us to approach the analysis of the classifier comprehensibility with objective measures instead of relying only on subjective opinions. Furthermore, it was shown that experience with a specific task reduces the cognitive load, while the lack of it increases the load (Murphy & Wright, 1984). This concept is addressed in our study by comparing the performance of two groups: data-mining experts (as suggested in Freitas (2014)) and novice data-miners.

Cognitive fit theory (Vessey & Galletta, 1991) proposes that the need to transform the mental representation during problem solving and extraction of information from the problem representation is eliminated if information presentation format matches the tasks. Hence, problem solving with cognitive fit leads to effective and efficient problem-solving. Therefore, we perform empirical evaluation of the respondents' performance in four tasks that have a good cognitive fit and analyse the suitability of classification trees for each task.

More specific studies come from the field of text comprehensibility, where numerous methods for determining comprehensibility have been devised (Göpferich, 2009). Schriver (1989) divides them into three groups and concludes that reader-focused approaches provide advantages over text-focused and expert-judgment-focused approaches. In line with this result, we perform an empirical study that is based on a user survey instead of simply measuring the model complexity, as in (Allahyari & Lavesson, 2011; Freitas, 2003, 2004; Jin & Sendhoff, 2008; Jin et al., 2005; Martens et al., 2011) or using expert-judgements, as in (Freitas, 2014). The Karlsruhe comprehensibility concept (Göpferich, 2009) provides a thorough theoretical background for the evaluation of text comprehensibility based on six text-comprehensibility dimensions, which can be mapped to classifier comprehensibility: concision (less complex classifiers are better), correctness (classifiers that agree with the user's domain knowledge are more comprehensible), motivation (classifiers should attract the user's attention), structure (classifiers should be broken down into adequate components), simplicity (classifiers should use known terminology and avoid ambiguity), and perceptibility (classifier visualisation should support the recognition of the classifiers' structure).

In the information technology (IT) field, there is a considerable number of empirical studies that investigate the comprehensibility (understandability) of conceptual models. Our survey design builds upon the following design issues, which are summarised in a review of experiments from this field (Houy, Fettke, & Loos, 2012): the research design, the number of experiment participants, and the observed dependent variables. In addition, our

study follows the framework for empirical evaluation of model comprehensibility (Aranda, Ernst, Horkoff, & Easterbrook, 2007) in all of the recommendations that can be applied to the classifier comprehensibility.

Finally, a few studies address specifically the classification-tree comprehensibility. Freitas (2014) reviews the case for comprehensible classifier models and discusses the advantages and drawbacks of five types of classification knowledge representations, including classification trees. He also suggests that the distribution of information across the leaves of a tree should be taken into account when analysing the comprehensibility of the tree. Therefore, we study the influence of the question-depth on the tree comprehensibility, which has not been empirically evaluated before.

The work by Allahyari and Lavesson (2011) is probably the first empirical study of classification-model comprehensibility. This study compares the comprehensibility of classifiers that are learned by three classification-tree and three rule-learning algorithms, based on subjective comparisons of classifier pairs by 50 students. We focus on classification trees because they are more comprehensible than classification rules (Allahyari & Lavesson, 2011). Furthermore, one of our survey tasks follows the design of their study, comparing classifier pairs. They also report that the classifier complexity has a negative correlation with the comprehensibility (understandability), and therefore, we extend their work by analysing the influence of several tree structure parameters. We also improve on their work by using objective measures and additional survey tasks, including data-mining experts as survey respondents, analysing larger trees and using a domain that is familiar to the respondents.

The study by Huysmans et al. (2011) empirically evaluates the comprehensibility of decision tables, trees and rules using subjective opinions (answer confidence) and objective measures of the respondent performance (time to answer and answer accuracy) for three tasks: classify an instance, verify whether a classifier agrees with a statement, and compare two classifiers for their equivalence. Our survey includes both objective measures as well as classification and verification tasks. In contrast with their work, we analyse the influence of the tree properties (and not the type of classifier knowledge representation) on the comprehensibility. Moreover, we extend the survey with additional tasks, with questions about the perceived comprehensibility, and by analysing the influence of classifier visualisation and the user's background in data mining.

The related surveys (Allahyari & Lavesson, 2011; Huysmans et al., 2011) consider only the comprehensibility of a classification tree as a whole while we also consider individual parts of a tree. This is important because the parts that classify more instances are used more often, or contain instances whose correct classification and explanation is more important for the user than the rest of the tree.

In our previous work, we designed a survey (Piltaver, Luštrek, Gams, & Martinčić – Ipšić, 2014a) according to experience from related work (Allahyari & Lavesson, 2011; Aranda et al., 2007; Freitas, 2014; Houy et al., 2012; Huysmans et al., 2011); we validated it with a group of data-mining experts and psychologists, implemented it as an online survey and tested it with respondents (Piltaver, Luštrek, Gams, & Martinčić – Ipšić, 2014b). This thesis improves the initial survey design based on the results of the validation and thoroughly discusses the design choices and bias prevention strategies that are crucial for the validity of the results. Finally, we conduct a statistical analysis of the obtained data, propose the question-depth as the novel parameter that influences the comprehensibility and introduce two new tree-comprehensibility metrics (Piltaver, Luštrek, Gams, & Martinčić-Ipšić, 2016).

2.3 Survey Design and Implementation

The following four subsections describe the survey design and implementation. Subsection 2.3.1 presents the general design choices, and Subsection 2.3.2 is dedicated to the prevention of the bias (i.e., avoiding influencing the results of the survey by its design). Subsections 2.3.3 and 2.3.4 present the range of the classification trees that are used in the survey according to the tree structure and visualisation properties. Subsection 2.3.5 gives details about survey implementation and defines rating scales that are used in the survey to collect the respondents subjective opinions.

2.3.1 General Design

Comprehensibility is inherently subjective (Huysmans et al., 2011; Martens et al., 2011) and hence impossible to measure directly (Allahyari & Lavesson, 2011). Therefore, we measure it indirectly by objective measures of the survey respondents' performance. This decision is based on research that argues that the complexity impacts the task performance (Campbell, 1988). We also collect the respondents' subjective opinions about the tree comprehensibility and the question-difficulty.

The first design choice is about the objective measures: the respondents' performance is measured in terms of the time that is required to answer a survey question (hereinafter **time-to-answer**) and the probability of the correct answer (hereinafter **answer-correctness**), which is consistent with the methodology that is used in the related work (Aranda et al., 2007; Houy et al., 2012; Huysmans et al., 2011). In general, the shorter time-to-answer and the higher answer-correctness correspond to easier questions and, therefore, to more comprehensible trees. The survey includes questions about a range of trees that have various structure and visualisation properties in order to objectively quantify the influence of each property on the tree comprehensibility. In addition, the respondents are asked for their subjective opinion about the tree comprehensibility and the difficulty of each question (hereinafter **question-difficulty**). The question-difficulty is rated on the five points scale defined in Section 2.3.5.3. Subjective opinions are used as the gold standard for the evaluation of the tree comprehensibility metrics, to estimate the variability of the respondents' subjective opinions, and to verify the correlations between the subjective opinions and the objective performance measures.

The second choice is about the evaluation: paired statistical tests are used to compare the performance and subjective opinions of each respondent on different trees. Paired tests are required because of the limited number of respondents and the high variability of their performance (see Figure 2.11) and subjective opinions. Hence, each respondent is required to answer all of the questions, and the number of questions must be limited (we set the upper bound to 100) because it is difficult to motivate the respondents to answer a lengthy survey and because answering too many questions could influence the respondent's performance and expressed subjective opinions.

The third design choice involves the set of analysed tree parameters: we consider tree-structure parameters, tree-visualisation properties and the question-depth. The analysed **tree structure parameters** can be computed algorithmically and in turn used for estimation of the tree comprehensibility. We analyse the following parameters, for the following reasons:

- the number of leaves (i.e., nodes without child vertices) – often used as a tree complexity metric that approximates the tree comprehensibility (Freitas, 2003; Martens et al., 2011) and measures concision of information presentation, which influences comprehensibility (Göpferich, 2009);

- the branching factor (i.e., the number of child nodes in the inner vertices) – many tree-learning algorithms enable learning binary trees (Demšar et al., 2013) and several heuristic functions for choosing the node splitting attribute take into account the branching factor (Harris, 2001; Quinlan, 1993);
- the tree depth (the length of the longest path from the root to a leaf) – a well-defined tree complexity measure, which depends on the number of leaves and the branching factor.

We analysed the **tree visualisation properties** because they could influence the comprehensibility (Göpferich, 2009) and to obtain data from which we empirically derived recommendations on visualising classification trees in a comprehensible way. We considered the following properties:

- the visualisation style (plain-text, Weka (Hall et al., 2009) and Orange (Demšar et al., 2013) visualisation styles);
- explicitly visualised information about the tree (text and colour-coded information);
- the tree layout (random and left-to-right order).

Finally, motivated by our previous research (Piltaver, Luštrek, Zupančič, et al., 2014; Piltaver, Luštrek, & Gams, 2014; Piltaver, 2010) and related work (Freitas, 2014; Sommer, 1995), we investigate the influence of the **question-depth**, which is equal to the depth of the deepest leaf that is required to answer a question about a tree.

The survey is divided into two parts. The first part includes four tasks (*classify*, *explain*, *validate*, and *discover*), which are dedicated to quantifying the influence of the tree-structure parameters and the question-depth on the tree comprehensibility based on the objective measurements and respondents' subjective opinions (Table 2.1). Different tasks are used because they can generate different evaluation perspectives (Aranda et al., 2007). A uniform tree visualisation style is used throughout the first part of the survey (as suggested by Allahyari and Lavesson (2011)), while the tree structure parameters and question-depth are varied systematically. The second part includes two tasks (*rate* and *compare*) that are dedicated to the respondents' subjective opinions about the influence of both the tree structure and the visualisation properties (Table 2.1) on the comprehensibility. The following list describes each task (defined in Piltaver, Luštrek, Gams, and Martinčić – Ipšić (2014a, 2014b)) while an example of questions for each task is shown in Section B.

- In the *classify tasks* (Figure B.1), the respondents classify an instance that is represented with an attribute-value table using a classification tree shown on the screen – the same task type was used by Allahyari and Lavesson (2011) and Huysmans et al. (2011). The instruction for the task says: "*Classify the instance described in the table on the left using the classification tree on the right.*" The answer form with the label: "*The instance belongs to class:*" contains a drop-down menu offering the alphabetically ordered names of the seven classes as an answer to the question.
- In the *explain task* (Figure B.2), the respondents are asked which attribute values would a) have to stay the same, b) have to be changed or c) are irrelevant in order to classify an instance into a given class that is different from the class to which the instance currently belongs (according to a classification tree shown on the screen). This corresponds to explaining an individual instance classification. For example, which habits (values of attributes) would a patient with high probability of getting

cancer (class) have to change in order to stay healthy? The instruction for the task says for example: "*What should be the values of the attributes for the example (listed in the table on the left) so that the classification tree would classify it as amphibian? If the value of an attribute is not relevant for the classification into the new class, maintain the default choice (i.e. "irrelevant"); otherwise change it to "unchanged" if the value must remain as it is, or "different" if it needs to be changed.*" The table of attribute-value pairs includes an additional column named "*new value*" with drop-down menus with three choices: *irrelevant* (default value), *different*, and *unchanged*.

- In the *validate task* (Figure B.3), the respondents check whether a classification tree agrees with a statement about the classification domain – this corresponds to validating a part of the classification tree. Similar questions were asked in Huysmans et al. (2011). The instruction for the task says for example: "*Does the classification tree agree with the statement below it: for animals from class amphibian it holds that backbone = yes and breathes = no?*"). The answer form includes a drop-down menu with two possible answers: *yes* or *no*. Each statement is composed of two conditions (for two different attributes), except when knowledge about animal class belonging to a leaf at depth one is verified, e.g. questions about *mammals* for the trees in Figure 2.1. The number of conditions to be validated is limited to two in order to observe the relative difficulty of validating domain knowledge corresponding to leaves at various depths regardless of the number of attributes relevant for classification of the instances belonging to those leaves – this is considered in the *classify* and *explain* tasks.

- In the *discover task* (Figure B.4), the respondents discover which property is unusual for a given class (e.g., *flightless bird*) based on a classification tree that divides the majority of instances that belong to the class into one leaf and the minority with the unusual property into another leaf. The goal of the task is to discover new knowledge from the classification tree. Rather than rediscovering well-known relations between attributes and classes, the questions ask the respondents to find an unusual property for a class of animals – a common property of outliers. The instruction for the task says for example: "*Most animals from class mammal have the same properties; few animals from this class have a rare property. Which is this rare property (select the attribute name and its value for the animals with the rare property)?*" The answer form includes two drop-down menus from which the respondent chooses the name of the attribute and its unusual value.

- In the *compare task* (Figure B.5), whose design is the same as in Allahyari and Lavesson (2011), the respondents are asked to compare the comprehensibility of two trees shown side by side using a scale described in Section 2.3.5.4 and introduced in Piltaver, Luštrek, Gams, and Martinčić – Ipšić (2014b). The visualisation style is fixed and a single tree-structure property is altered or the tree structure is fixed and the visualisation style is altered. In the latter case, one of the trees was shown in the previous five tasks – serving as a known frame of reference – while the other one is a previously unseen tree. The instructions say: "*The following question type measures the subjective opinion about the tree comprehensibility; there are no correct and wrong answers and time needed to answer each question is not important at all. Compare the classification trees in the pictures and choose the answer that best fits your opinion.*" The answer is given by clicking a radio button in front of one of the offered answers. Clicking on a tree opens a new window showing a full-screen picture of the selected tree and a back button. This is needed because the text in nodes of

some of the bigger trees becomes difficult to read when the trees are scaled to half of the screen width.

- In the *rate task* (Figure B.6), the respondents rate how comprehensible a tree is on a five points scale described in Section 2.3.5.5 and introduced in Piltaver, Luštrek, Gams, and Martinčić – Ipšić (2014b). They are asked to rate trees that have a uniform visualisation style and systematically varied tree-structure parameters (note that the question-depth cannot be varied in this task because the comprehensibility of the entire tree is rated). The instruction for the task says: "*How comprehensible is the tree shown below?*" The answer is given by clicking a radio button in front of one of the offered answers.

Table 2.1: The investigated tree parameters and measures used for each survey task. Subjective measures and tree visualisation properties are written in italics, while objective measures and tree structure parameters are in regular font.

Task	**Measures**	**Varied tree parameters**	**Values**
Classify *Explain* *Validate* *Discover*	Time-to-answer, Answer-correctness, *Question-difficulty*	Number of leaves Branching factor Question-depth	[3–11] [2–4] [1–7]
Compare	*Relative comprehensibility*	Number of leaves Branching factor Tree depth *Visualisation style* *Additional visualised information* *Tree layout*	(4 vs 10) (2 vs 3) (2 vs 4) (plain-text, Weka and Orange default) (pie charts, meaningful attribute names) (ordered vs random)
Rate	*Absolute comprehensibility*	Number of leaves Branching factor	[3–10] [2–4]

2.3.2 Preventing Survey Bias

The survey results could be biased by the **classification domain** because the respondents perform better when answering questions about familiar domains compared to unfamiliar domains (Aranda et al., 2007; Martens et al., 2011; Sweller, 1988). Furthermore, interesting domains motivate the respondents (Göpferich, 2009), while dull domains could discourage them or deteriorate their performance. This concern was prevented by posing questions about a single domain that is familiar to all of the respondents and is interesting to the majority (according to the test-group post-survey interviews). The Zoo dataset from the UCI machine-learning repository (Bache & Lichman, 2013) was used because it fits the requirement (i.e., requires only elementary biology knowledge and includes interesting facts about animals) and enables the construction of a large range of trees (see Table 2.2). This dataset consists of 101 instances, each representing an animal (e.g., antelope, crow, frog) that belongs to one of 7 classes: *mammal* (41 instances), *bird* (20), *fish* (13), *mollusc* (10) *insect* (8) *reptile* (5) or *amphibian* (4). Each instance is described with 15 binary (*hair*, *feathers*, *eggs*, *milk*, *airborne*, *aquatic*, *predator*, *toothed*, *backbone*, *breathes*, *venomous*, *fins*,

tail, *domestic*, *cat-sized*) and 1 numeric (*legs*) attribute. The attribute *animal name* is not used because it is a unique identifier of an instance.

The second potential source of bias is the **learning effect** (Vessey & Galletta, 1991): the respondents learn while answering the survey, and therefore they perform better at questions that are posed later in the survey compared with the initial questions. For example, novice data-miners learn how to use classification trees efficiently (e.g., use the mouse pointer to mark the current node while looking up the value of the node attribute). On the other hand, data-mining experts might need a few initial questions to become accustomed to the classification-tree visualisation style and the survey answering form. The learning effect was noticed while testing the initial version of the survey (significantly longer times for answering the first question compared to the subsequent questions), and therefore several precautions were taken:

- The questions are posed in six groups; each corresponds to one of the six tasks. All of the questions in a task are posed in exactly the same way: the phrasing of the question, alphabetical order of the attributes in an attribute-value table that contains exactly 10 attributes, visualisation style and layout of the classification tree, and the answer forms are the same. These precautions shorten the learning period.

- Instructions on how to answer an example question are provided before each task. They include figures and explanations showing how to solve the task step by step. In addition, a test question (in exactly the same format as the questions in the task but about a different domain) must be answered correctly before the respondent is allowed to proceed to the first question (similar to Vessey and Galletta (1991)). This approach verifies that the respondent understands the question and knows how to answer it.

- The tasks are ordered from the simplest to the most difficult, to diminish the learning effect.

- The Latin-square design (Box, Hunter, & Hunter, 2005; Vessey & Galletta, 1991) is used for the question order within each task. This arrangement means that the respondents answer the questions in different orders and that the frequency of posing each question at any given position in the question ordering is constant. This approach distributes the remaining learning effect equally over all of the questions, and therefore, the average results over all of the respondents are not biased by the learning effect.

The respondents' subjective opinions are biased by their mental abilities (Sweller, 1988) and **data-mining experience** (Murphy & Wright, 1984) and could have a high variability due to the subjective interpretations of the comprehensibility rating scales. To prevent this type of bias, the two tasks that ask for subjective opinions are placed at the end of the survey. In this way, novice data-miners obtain experience on which they can soundly base their subjective opinions. Second, the offered answers to the multiple-choice questions are textual (instead of numeric), and their meaning is explained with at least one sentence and one example (Piltaver, Luštrek, Gams, & Martinčić – Ipšić, 2014b). This approach mitigates subjective interpretations of the answers' meaning. For example, the explanation that says: "*I can use the knowledge represented by the classification tree as soon as I read it for the first time; I can easily remember it; I can explain it to another person without looking at the figure.*" leaves fewer interpretation possibilities than a mark 2 on a 1 to 5 scale. Finally, the respondents freely select the survey language (English, Slovenian or Croatian), which prevents decreased performance due to the use of a foreign language.

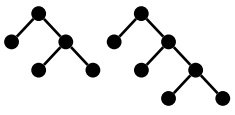
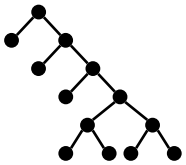
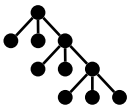
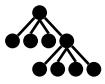
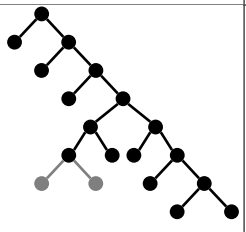
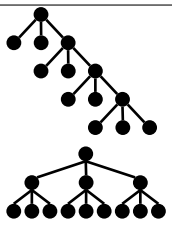
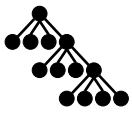
Finally, the survey results can be biased by the **respondent's motivation**: well-motivated respondents perform better and rate questions as easier and trees as more comprehensible compared to non-motivated respondents. Instead of offering monetary awards (Huysmans et al., 2011; Vessey & Galletta, 1991), we used a gamification approach (Zichermann & Cunningham, 2011): a table that compares the performance of the respondent to the best, average and worst respondent is shown after finishing each task. According to the post-survey interviews, the comparison table turned the survey into a competition and motivated the respondents to answer quickly and correctly.

The survey was first tested with a group of test respondents to verify the assumptions on which it is based (Piltaver, Luštrek, Gams, & Martinčić – Ipšić, 2014b): i) the performance of respondent solving the survey tasks depends on the classification tree comprehensibility; ii) the observed classification tree properties influence comprehensibility; iii) the range of classification trees and questions used in the survey is broad enough to measure the influence; iv) the objectively measured respondent performance parameters (time-to-answer, answer-correctness) are related to the subjective perception of the classifier comprehensibility; v) the range and explanations of the scales used to express respondents' subjective opinions are appropriate; and vi) the task difficulty progresses from the easiest to the most difficult.

2.3.3 Varying the Tree Structure

We systematically varied the investigated tree structure parameters, as illustrated in Table 2.2, while keeping the other factors that might influence the respondents' answers unchanged. This approach enables analysing the influence of a single tree structure parameter on the comprehensibility. The size of the trees is limited to assure text readability. Note that this sizing limitation is not a drawback because large trees are rarely used by humans in practice. The classification trees used in the survey (Figures C.1–C.12) are learned and visualized using Orange data mining software (Demšar et al., 2013). The tree construction process (Piltaver, Luštrek, Gams, & Martinčić – Ipšić, 2014b) is described below.

Table 2.2: The structure of the trees (Figures C.1–C.9) used in the survey.

Number of leaves	Branching factor 2	3	4
3 or 4			
7			
9 or 10			

The basic classification tree (Figure 2.1) was learned using the Classification Tree widget in Orange (Demšar et al., 2013) with the following parameters: gini index as the attribute selection criterion, pruning with m-estimate where $m = 2$, and minimum of 4 instances in a leaf. It has 7 leaves, 6 binary attributes (resulting in a uniform branching factor 2), and tree depth 5. Choosing other attribute selection criterion would only change the order of the attributes in the tree; however, the set of the attributes itself would not change.

The larger trees with 9 or 10 leaves (Figure C.3) were obtained by changing the pruning parameters. The smaller trees with 3 or 4 leaves (Figure C.1) were learned using a reduced set of attributes, because pruning resulted in unnatural trees (due to the uneven class distribution in the dataset) that could cause survey bias (Göpferich, 2009; Martens et al., 2011). The sets of attributes used to learn the small trees were chosen so that the leaves of the learned trees correspond to clusters obtained using hierarchical clustering (by Euclidian distance and complete linkage criterion) on the original dataset. Each leaf in the small trees contains animals from a single class or classes of animals with similar properties.

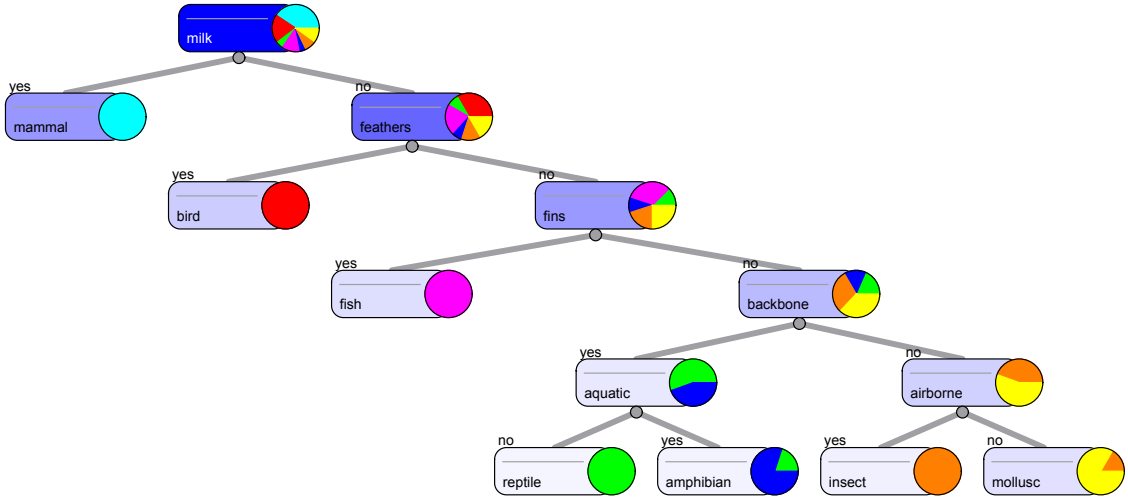


Figure 2.1: The basic classification tree.

To construct trees with branching factor 3 (Figures C.4—C.6) and 4 (Figures C.7—C.9), additional attributes were computed as Cartesian products of the original attributes. For example, combining the attributes *aquatic* and *breathes* produces a ternary attribute with the values: *aquatic-does not breathe* (e.g., *fish*), *aquatic-breathes* (e.g., *amphibian*) and *terrestrial*. As a result, the same or at least very similar structure as in the original binary tree was obtained (e.g., Figure 2.2). Note that two versions of the tree with 9 leaves (Figure C.6) and branching factor 3 were obtained, yet they differ in the maximal depth of the tree.

For the *discover task*, special trees were constructed: they split the majority of instances that belong to a class (e.g., *bird*) into one leaf and the minority with an unusual property (e.g., *not airborne*) into another leaf (Figure 2.3). The outliers in the dataset were first identified using the Outliers widget in Orange (Demšar et al., 2013). After that, the trees that misclassify outliers from one class were constructed using the Interactive Tree Builder widget. Some parts of the tree were selected manually in order to place the attribute that splits the outliers from the normal instances (the attribute expected as the answer) at a desired depth in the tree. The remaining parts of the tree were built using the automatic tree learning function. The limited number of outliers in the dataset resulted in

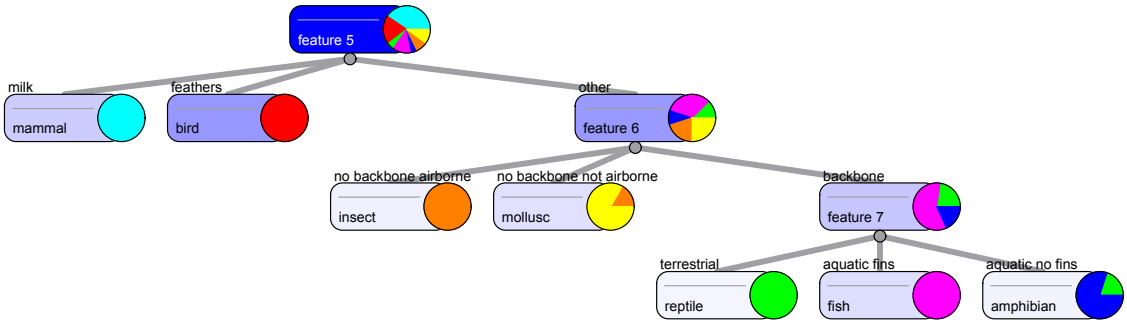


Figure 2.2: Tree with branching factor 3 that corresponds to the tree in Figure 2.1.

8 questions about 4 binary trees with 3 (Figure C.10a), 5 (Figure C.10b), 7 (Figure C.11) and 11 (Figure C.12) leaves. In contrast with the trees used in the other tasks, each of the tree nodes lists the number of instances of a given class belonging to the node – used by respondents to distinguish between the common and the rare properties of animals belonging to the class.

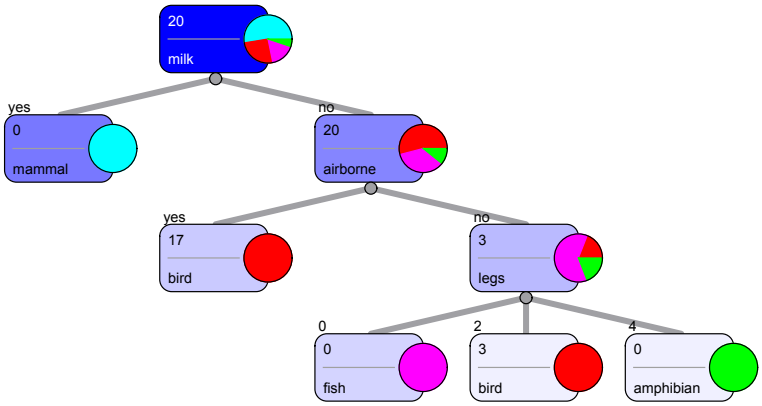


Figure 2.3: The tree shows an unusual property for a *bird*: *not airborne*. The numbers in the nodes correspond to the number of training instances from the class *bird* that belong to each node.

2.3.4 Varying the Tree Visualisation

The influence of the tree visualisation is analysed in the *compare task*. The goal is to determine which tree visualisation parameters influence the comprehensibility and to assess their relative importance; therefore, only a single question per visualisation parameter is posed.

First, the plain-text tree visualisation (Figure 2.4a) and Weka (Hall et al., 2009) default visualisation styles (Figure 2.4b) were compared with Orange default visualisation style (Figures 2.1–2.3). Plain-text visualisation represents each tree node with a line of text as shown in Figure 2.4a and is typically used on text-only displays, e.g., as a part of the classifier output in the Weka Explorer (Hall et al., 2009). Weka also offers a graphical visualisation style illustrated in Figure 2.4b. The structure of the tree is instantly evident in this representation, which makes it easier to follow a path from the root to a leaf and to distinguish leaves from the other nodes. In addition, all the values of an attribute are listed

directly below it, which simplifies comparing the influence of alternative attribute values. Orange visualisation style (Figures 2.1–2.3), which is the richest of the three examined styles and hence used in all the survey questions, is in general similar to the Weka graphical style. However, it supports adding textual information (e.g., majority-class name, target class probability, the number of instances) to the nodes and uses graphical elements and colours to represent additional information (node colour corresponding to the number of instances in a node, class distributions pie charts).

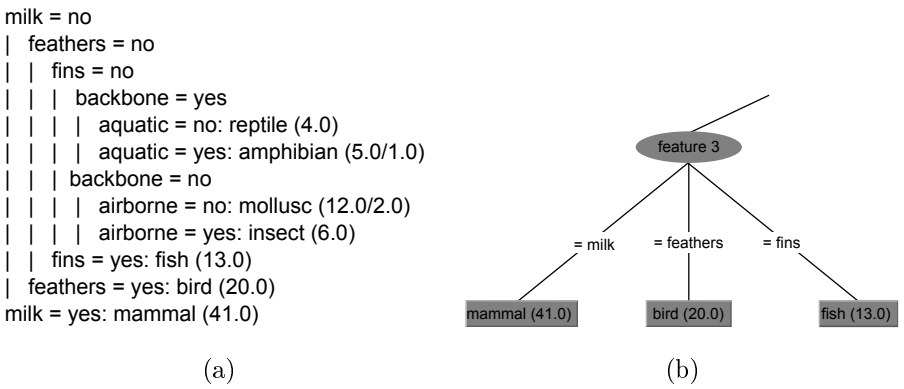


Figure 2.4: Tree visualisation styles: a tree plotted using the plain-text visualisation (a) and a branch of a tree with the Weka visualisation style (b).

Two options with regard to the information shown in the tree visualisation were considered: classification trees with (Figure 2.5a) and without pie charts (Figure 2.5c) and trees with (Figure 2.5a) and without meaningful attribute names, attribute values and class names (Figure 2.5b). The pie charts support the user to perform the following tasks efficiently: finding all the leaves with the same class (same prevalent colour of the pie chart) and estimating the probability of a class in a leaf (the share of the prevalent colour in the pie chart). On the other hand, meaningless attribute and attribute-value names make it more difficult to comprehend a tree, because domain knowledge cannot be used and because remembering numeric attribute names and values is more difficult than remembering known and meaningful semantic names.

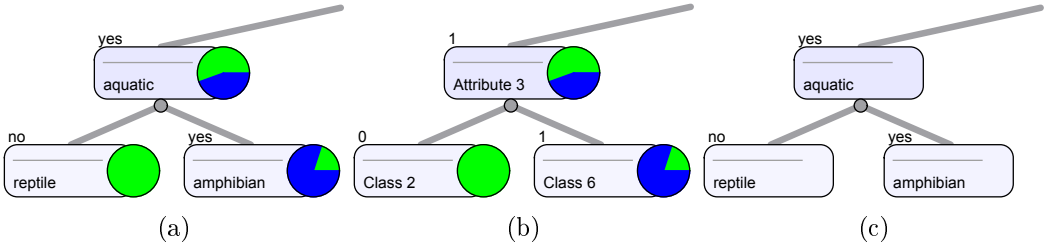


Figure 2.5: Subtrees that illustrate the information given in each node: the default (a), with meaningless names of attributes/class (b) and without pie charts (c).

Finally, the importance of the tree layout was evaluated by comparing the comprehensibility of a tree with a random layout (Figure 2.6b) and a tree with a left-to-right layout (Figure 2.6a). The layout is obtained by placing the shallower subtrees on the left and the deeper subtrees on the right of the parent node. The advantage is that this layout positions

the simpler instances on the left side of the tree figure, which is commonly read first (in all three survey languages) and makes the tree structure systematic. The left-to-right layout was used to visualise the trees in the other tasks.



(a) (b)

Figure 2.6: Two trees with the same structure but different layouts: left-to-right (a) and random (b) order.

2.3.5 Survey Implementation

To conduct the survey, we implemented a custom on-line survey engine (Piltaver, Luštrek, Gams, & Martinčić – Ipšić, 2014b, 2014a), which allows remote participation at a time that is convenient for the respondents. This approach enables accurate measurements of time-to-answer, the recording of all of the respondents' actions (e.g., correcting a wrong answer), displaying data about the respondent's performance (used as a motivation tool), simple translation of the survey into different languages, and real-time progress monitoring, which was especially useful in the testing phase. An example of a question for each of the six tasks is shown in Appendix B. All the trees used in the survey are shown in Appendix C. The survey itself is available at http://dis.ijs.si/classifier-comprehensibility-survey.

2.3.5.1 Answering a Survey Question

Figure 2.7 shows a web page with an example of a survey question. When a web page with a survey question is opened, only the survey question (Figure 2.7a) and a footer of the page are shown. The footer contains the respondent's name, the name of the task and the current question number, a link to a help page, and a support e-mail address. When a respondent clicks the *Start solving* button, the timer starts and the attribute-value table (Figure 2.7b), classification tree (Figure 2.7c) and answer form (Figure 2.7d) are shown. After clicking the *Submit answer* button, the timer stops, the answer form is locked for editing and the question-difficulty form (Figure 2.7e) is shown. When the respondent clicks the *Next question* button, the collected data is stored into a database and the next question appears.

2.3.5.2 Implementation

The online survey is implemented using PHP and JavaScript programming languages, MySQL database, SVG images of classification trees, and CSS for web page formatting. The database includes one table for demographic data and one table for each of the six tasks. Each task table includes question order number, the time of question-answer, and field(s) representing the respondents' answer. Tables for *classify*, *explain*, *validate* and *discover* tasks also include the measured answering time, list of all respondent's clicks, and the indicator of the correct answer.

PHP scripts are used to dynamically generate survey web pages (each corresponding to a survey question) with correct ordering of questions for each respondent (Latin-square design) and storing the answers into the database. One PHP script per survey task is

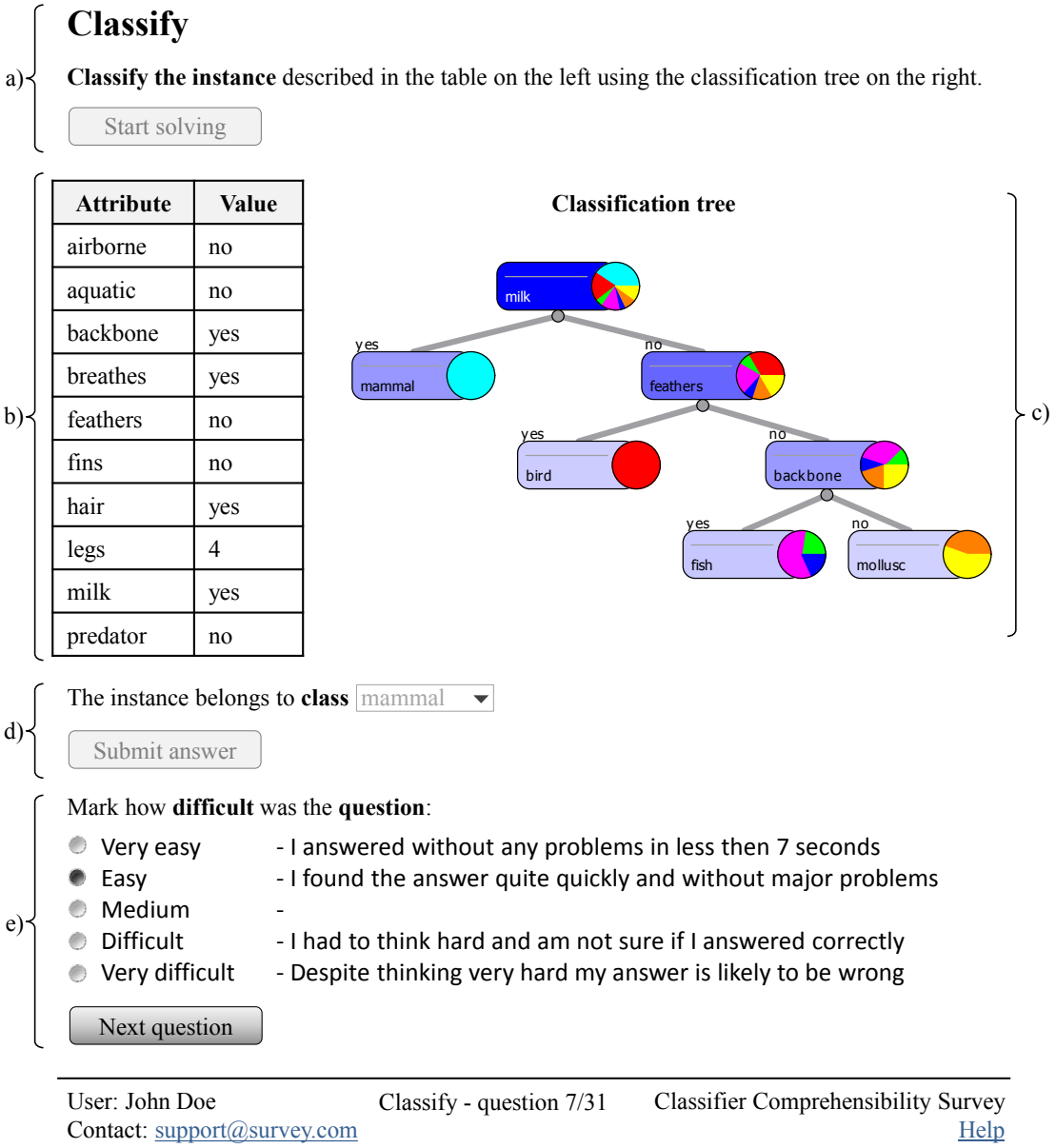


Figure 2.7: An example of a web page with a survey question.

used to generate question web pages based on a question template and a simple data structure defining the questions (a list of records including the question specific text and a link to the classification tree). An additional PHP script includes a library of shared functions and data structures: one represents dataset instances used in the survey and the other represents strings (instructions, attribute names and value, classes, etc.) translated

into English, Slovenian and Croatian languages. Additionally, PHP scripts are used for login and help pages, a respondent home-page providing feedback on survey progress and performance compared to the group, and database operations such as checking correctness of answers and backing-up the database tables.

JavaScript is used to measure the time of answering each question. Clicking on the *Start solving* button calls a JavaScript function that displays a table with attribute-value pairs, image of a tree and answer form (drop-down lists or radio buttons) and starts the timer. Changing a value of an answer form field records the action type and time. When the respondent clicks the *Submit answer* button, the answer fields are disabled, time is calculated, and question-difficulty rating options are displayed. When the *Next question* button is clicked, the collected values are assigned to hidden form fields in order to pass them to the PHP script that stores the data in the database and displays the next question.

Classification trees used in the survey were saved as Scalable Vector Graphics (SVG), which is an XML-based vector image format for two-dimensional graphics. The vector images result in small file size (compared to bitmap images) and enable optimal display quality regardless of the screen size. In addition, SVG format enabled translation of the text in the tree images (exported from Orange (Demšar et al., 2013)) into the three languages using a Java program and the translation table from the PHP library script.

2.3.5.3 Question-Difficulty Scale

Immediately after answering each question in *classify*, *explain*, *validate* and *discover* tasks, respondents were asked to give the subjective judgement of question-difficulty on the five points scale (Figures B.1–B.4). Each label of the scale (Figure 2.7e) was accompanied with an explanation in order to prevent variation due to subjective interpretations of the scale:

1. **Very easy** – I answered without any problems in less than 5 seconds.
2. **Easy** – I found the answer quite quickly and without major problems.
3. **Medium**.
4. **Difficult** – I had to think hard and am not sure if I answered correctly.
5. **Very difficult** – Despite thinking very hard my answer is likely to be wrong.

2.3.5.4 Classification-Tree Comparison Scale

In *compare task*, the respondents compared pairs of trees according to their comprehensibility. The seven points scale defined below had one neutral answer and three answers preferring either the tree shown on the left or the tree shown on the right (Figure B.5). The symmetrical scale was used because the relative positions of the two trees in each question were chosen randomly for each respondent in order to prevent bias due to the order of the compared trees on the page.

0. The trees are **equally comprehensible**: I do not notice any difference in comprehensibility between the two trees.
1. The tree on the **left is slightly more comprehensible**: although one tree is slightly more comprehensible, answering questions similar to the ones in this survey about the other tree would not be more difficult.
2. The tree on the **left is more comprehensible**: answering questions similar to the ones in this survey about the other tree would be more difficult or take more time.

3. The tree on the **left is much more comprehensible**: answering questions similar to the ones in this survey about the other tree would be significantly more difficult and would definitely take more time.

2.3.5.5 Classification-Tree Comprehensibility Scale

In *rate task* (Figure B.6), the respondents expressed their subjective opinion about the classification trees comprehensibility on the five points scale shown below. Each of the scale labels is explained with three statements: the first is about how difficult it is to use the knowledge represented by the tree; the second is about how difficult it is to remember the tree; the third is about how difficult it is to explain the knowledge represented by the tree to another person.

1. **Very easy to comprehend** – I can use the knowledge represented by the classification tree as soon as I read it for the first time; I can easily remember it; I can explain it to another person without looking at the figure.

2. **Easy to comprehend** – I can use the knowledge represented by the classification tree after studying it for some time; I can remember it with some effort; I can explain it to another person without looking at the figure, but I am not sure that the person will fully understand it.

3. **Comprehensible** – Without long study I can use most of the knowledge represented by the classification tree, but need the figure for some details; it would be difficult to remember it; I can explain it to another person if I see the figure while the other person does not see it, but that person is unlikely to fully understand it.

4. **Difficult to comprehend** – Without a long study I can use only some of the knowledge represented by the classification tree, and need the figure for the rest; It would be very difficult to remember it; I can explain the outline to another person if I see the figure while the other person does not see it.

5. **Very difficult to comprehend** – I need the figure to use the knowledge represented by the classification tree; it would be extremely difficult to remember it; I can explain it to another person only if we both see the figure.

2.4 Results and Discussion

This section presents the survey results and discusses the influence of the classification trees' properties on their comprehensibility. General information about the survey questions, tasks, the respondents and their performances are given in the first two subsections, followed by subsections with the results about the influence of the tree structure parameters and the question-depth, the influence of the visualisation properties, and the comprehensibility metrics.

2.4.1 General Data About the Survey

The survey includes 98 questions: 80% of the questions are about the influence of the tree structure parameters on their comprehensibility, and 20% are on subjective opinions about the tree comprehensibility. The data about the size of the survey are presented in Table 2.3. The *classify task* included 30 questions; one question for each possible combination of the tree size, branching factor and question-depth that was available using the trees in

Table 2.2. The test respondents reported that they would not be comfortable answering more questions; therefore, the number of questions in the subsequent tasks was reduced. The questions in the *explain task* required considerably more time (Table 2.3), therefore the number of questions in this task was reduced to 18. The number of questions in the *validate task* was reduced to 23 to optimize the trade-off between the number of questions and the number of possible comparisons for the analysis. The *discover task* included only 8 questions, because it required special trees as explained in Section 2.3.3. The *compare task* included one question for each analysed tree property listed in Table 2.1. The *rate task* included one question for each classification tree illustrated in Table 2.2. On average, the respondents needed about 29 minutes to answer the questions plus 26 minutes to rate the question-difficulty and tree comprehensibility.

Table 2.3: The statistical data about the size of the survey.

Task	*Classify*	*Explain*	*Validate*	*Discover*	*Compare*	*Rate*	Total
Number of questions	30	18	23	8	8	11	98
Number of respondents	69	52	52	52	52	52	/
Number of answers	2,070	936	1,196	416	416	572	5,606
Mean time-to-answer per respondent [s]	17.8	27.3	15.9	25.9	/	/	/
Total time-to-answer for all respondents [h]	10.2	7.1	5.3	3.0	/	/	25.6 + ∼24
Answer-correctness [%]	97.6	87.7	96.0	72.8	/	/	93.0

Figure 2.8 shows the median time-to-answer and the mean question-difficulty (mapped to integers that range from 1 to 5) for each question (represented with a circle). The questions in the *classify task* were rated as the easiest (the mean question-difficulty over all of the questions 1.6) and the answer times are short compared to the other tasks. The most difficult was the *discover task*: the mean question-difficulty over all of the questions is 2.4, and the median time-to-answer for the easiest question is 14.5 s, which is the highest among the four tasks. The *explain task* required more time to answer because the number of clicks needed to answer a question was equal to the depth of the question, while a single mouse click was needed in the other tasks. Figure 2.8 shows a high correlation between the question-difficulty and time-to-answer, which confirms that the time-to-answer is a suitable objective measure and that the difficulty scale was well-designed.

The range of question-difficulty in each task is considerable (Figure 2.8): The question-difficulty ranges for about one point on the difficulty scale within each task. The range is even greater if individual respondents' rates are considered instead of the average rate over all the respondents.

Figures 2.9 and 2.10 show the correlations between the answer-correctness and time-to-answer (or question-difficulty). Answer-correctness is constant for the questions that have a low question-difficulty or time-to-answer. The answer-correctness starts decreasing when the time-to-answer increases above 10 s or when the question-difficulty increases above easy. The answer-correctness is scattered between 91% and 100% in the *classify* task regardless of the time-to-answer or question-difficulty. The questions are so easy that the incorrect answers are probably due to random errors and not due to higher question-difficulty or lower tree comprehensibility. The answer-correctness is approximately 94% and 100% for the easiest third of questions in the *explain* and *validate* tasks, respectively, and starts decreasing towards 75% and 81%, respectively, as the question-difficulty and time-to-answer increase. Again, the incorrect answers are due to random errors, but only for the easiest questions; for more difficult questions the answer-correctness decreases as

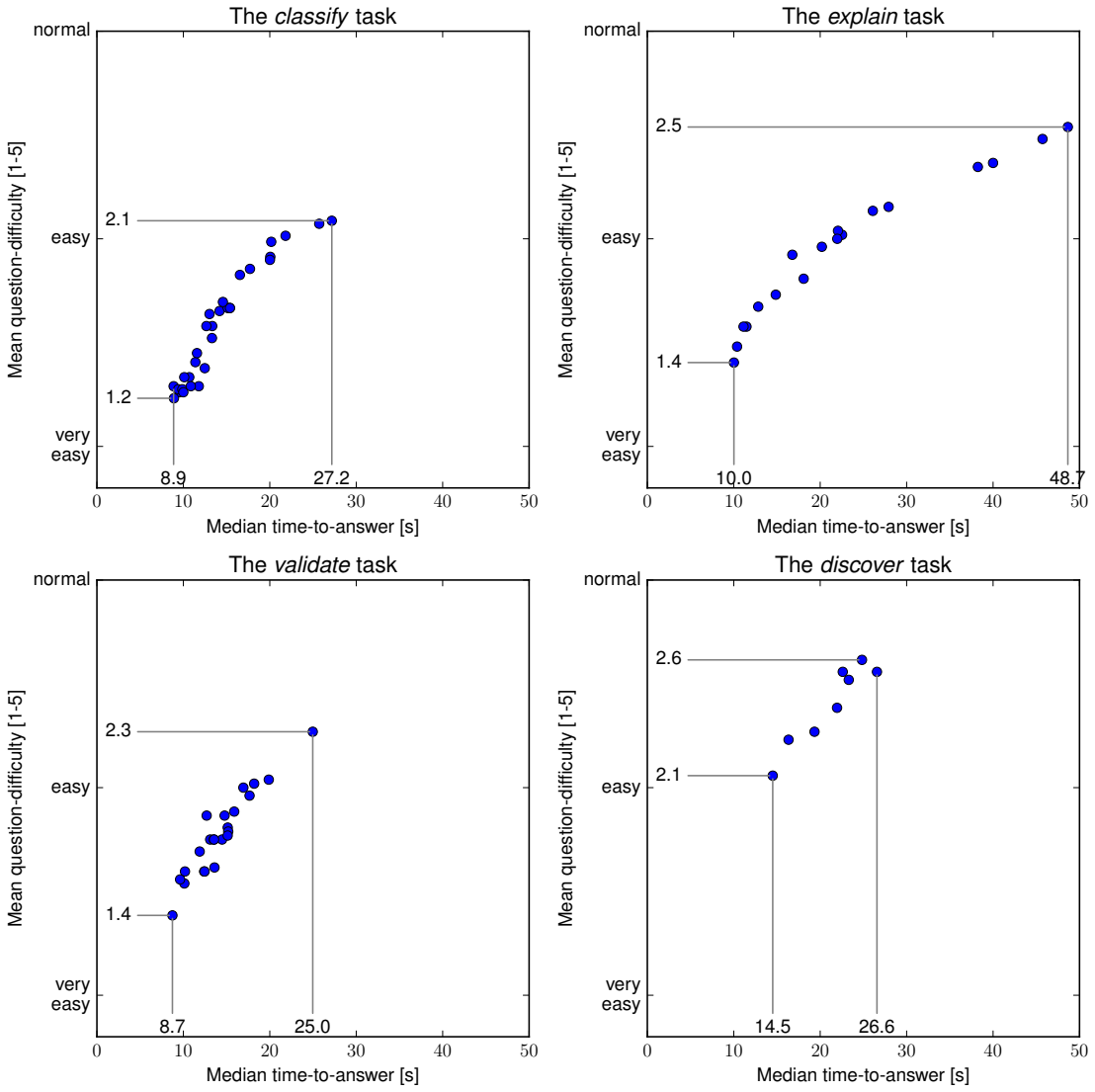


Figure 2.8: Median time-to-answer and mean question-difficulty for the questions in the first part of the survey.

the question-difficulty increases, which is intuitively expected. The answer-correctness decreases from approximately 81% to 67% in the *discover task* as the time-to-answer and question-difficulty increase.

2.4.2 Survey Respondents and the Performance of the Respondent Groups

The survey respondents are data-mining and artificial-intelligence researchers from the Jožef Stefan Institute, Slovenia, and faculty staff and students from the Department of Informatics – University of Rijeka, Croatia. Because previous studies have indicated that past experience, education and individual cognitive abilities can influence the task performance (Aranda et al., 2007; Benbasat & Taylor, 1982; Freitas, 2014; Huysmans et al., 2011; Lee, Cheng, & Cheng, 2007; Murphy & Wright, 1984; Vessey & Galletta, 1991), the respondents are divided into three groups according to their experience with classification trees: data-mining *experts* (PhD in data mining), computer science and informatics

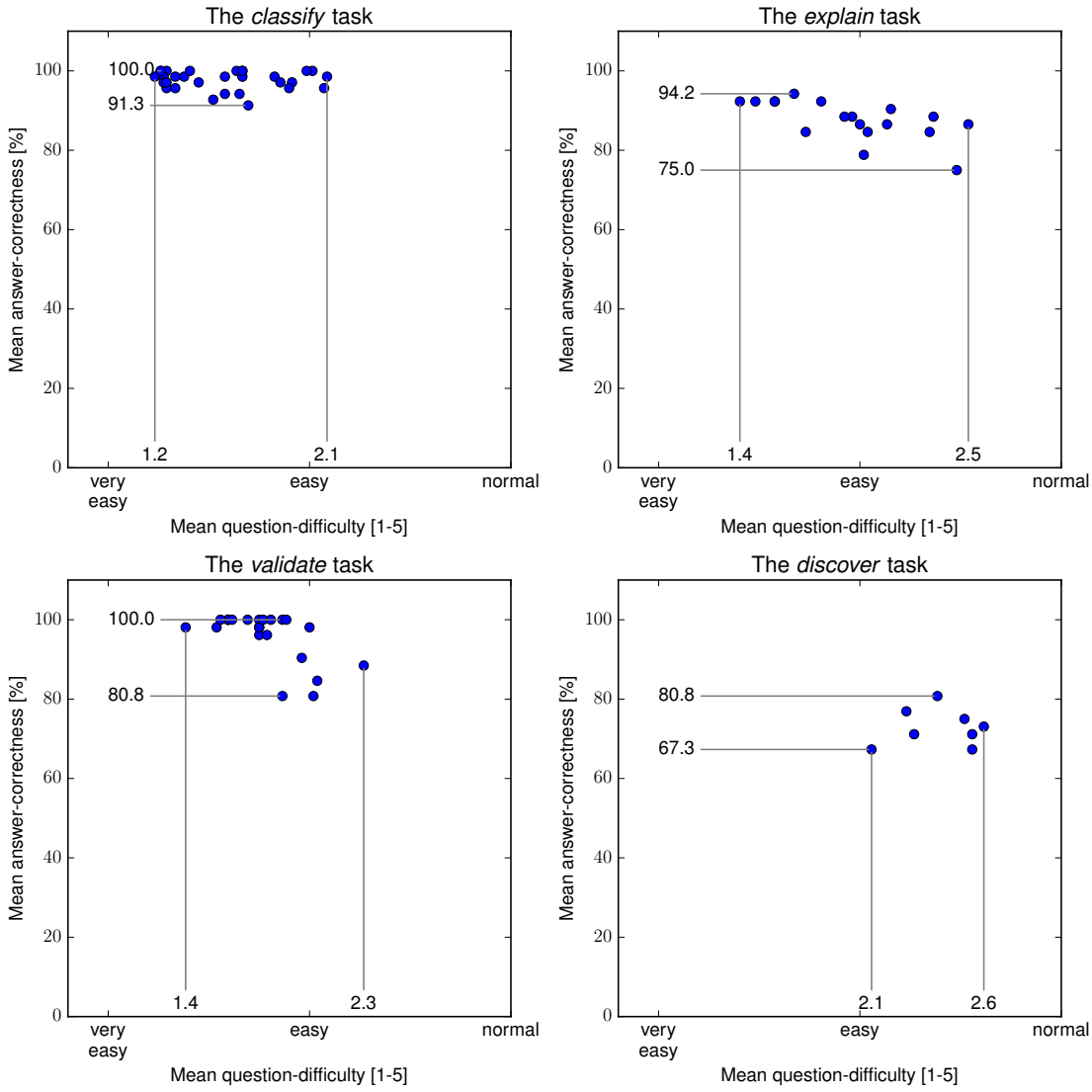


Figure 2.9: Mean answer-correctness and mean question-difficulty for the questions in the first part of the survey.

students (BsC in informatics) who took at least an introductory data-mining course, and other *IT specialists* (MsC or PhD in computer science) with very limited or no experience with classification trees. In total, 52 respondents answered all of the questions: 26 *students*, 19 *experts*, and 7 *IT specialists*, of whom 62% were male. An additional 17 respondents answered only the classify task: 15 *students*, one *expert* and one *IT specialist*. The numbers are comparable with most of the studies that are reviewed in (Houy et al., 2012).

Table 2.4 shows the exact numbers of respondents that answered all the survey questions divided by group, gender and language. The *students* group with 26 respondents represents 50% of all the survey respondents, while the *IT specialists* group containing 7 respondents is the smallest. The respondent sample is diverse according to the data-mining experience and gender. However, a comparison by gender is not possible because the gender groups are not balanced according to data-mining experience. The division by language serves only as an illustration of the respondent diversity.

The respondent sample enables a comparison of the performance of the *experts* and

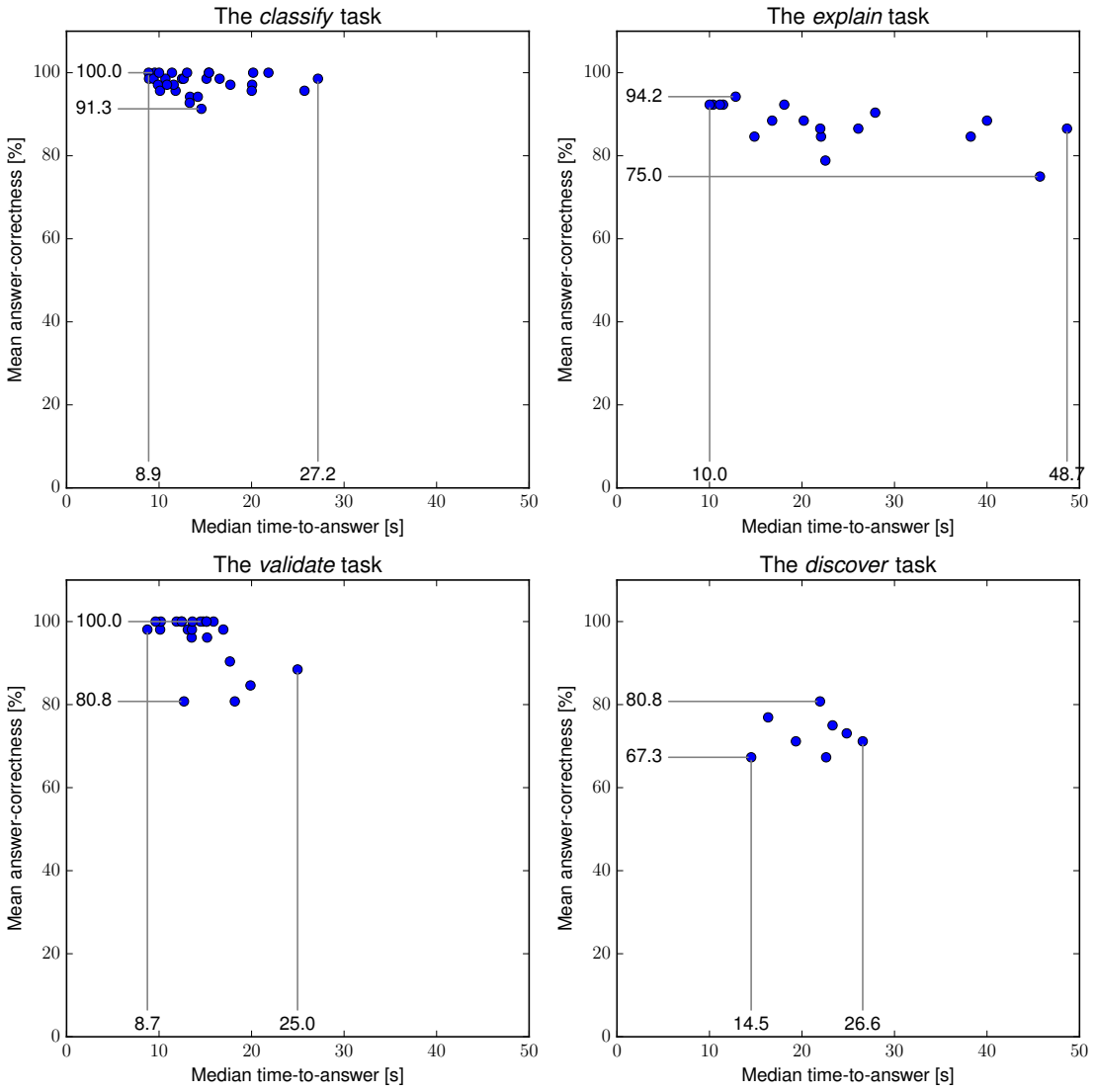


Figure 2.10: Mean answer-correctness and median time-to-answer for the questions in the first part of the survey.

Table 2.4: The number of respondents depending on gender, survey language and data-mining experience.

	32 males (62%)			20 females (38%)			
Group	Croatian	Slovenian	English	Croatian	Slovenian	English	Total
Experts	1	7	6	1	4	0	19 (37%)
Students	9	4	0	13	0	0	26 (50%)
IT specialists	4	1	0	2	0	0	7 (13%)
Total	14	12	6	16	4	0	52

students. The results are given in Table 2.5. In the *classify task*, the *experts* achieve the shortest median time-to-answer, the lowest average question-difficulty and the highest answer-correctness among the three respondent groups. The *students* perform the worst: the probability of an incorrect answer is 3 times higher compared to the *experts*. Nev-

ertheless, the performance of the best *students* is comparable with the performance of *experts*.

Table 2.5: The performance of respondent groups in each task.

Task	Time-to-answer [s]		Answer-correctness [%]		Question-difficulty [1-5]	
	experts	*students*	*experts*	*students*	*experts*	*students*
Classify	13	15	99	97	1.4	1.5
Explain	20	23	93	84	1.8	2.0
Validate	15	15	95	96	1.7	1.6
Discover	20	23	89	64	2.2	2.3

The comparison of the performance between the respondent groups in the *explain task* is the same as in the *classify task*, with two differences: the average answer-correctness for both groups is lower, indicating that this task is more difficult; and the difference between the two groups is much greater in the *explain task*. In the *validate task*, *experts* and *students* perform equally well.

In the *discover task*, the *experts* again perform best. The *students* achieve a slightly longer median time-to-answer but a much lower answer-correctness. The basic knowledge about classification trees does not suffice in the *discover task*; therefore, the *experts* have a larger advantage than in the previous three tasks. The difference in the performance of the respondent groups indicates that the results of the surveys performed only with students could be biased.

2.4.3 The Influence of Tree-Structure Parameters and Question-Depth

Figures showing summary statistics are used to analyse the influence of a single tree structure parameter on the respondents' performance (e.g., Figures 2.11–2.13). Each figure is divided into a grid of subplots. The vertical axis in each subplot corresponds to the analysed performance measure, the horizontal axis to the analysed tree structure parameter, and the two grid dimensions to the two remaining tree structure parameters. Statistical tests are performed between all the value pairs of the analysed parameter (in each subplot) and the Holm-Bonferroni correction for multiple comparisons is applied to check whether the influence of the parameter on the performance is statistically significant. The influence of the tree structure parameters on the respondents' performance was tested using the Wilcoxon signed-rank test for time-to-answer and question-difficulty and McNemar's test for answer-correctness. Paired tests were used to account for the differences in the performances between the respondents. Therefore an increase in the average (or median) value of a performance measure is not enough for the test to be significant. It is only significant if enough respondents perform worse when answering a question about a more complex tree compared to answering a question about a simpler tree. We tested whether increasing a single parameter by one step influences a performance measure while the two remaining parameters were fixed. The Holm-Bonferroni correction for multiple comparisons was applied to check for statistical significance.

Figure 2.11 shows the influence of the question-depth on the time-to-answer in the *classify task* – question-depth is the parameter with the highest influence on comprehensibility. Each subgraph shows a Tukey boxplot for questions with respect to a given branching factor and number of leaves. The tree branching factor is 2 in the left, 3 in the middle and 4 in the right column. The number of leaves is small (3 or 4) in the top row, middle (7) in the middle row and high (9 or 10) in the bottom row. It is not possible to obtain trees that have some combinations of tree structure parameter values (e.g., a tree with 3 leaves and a

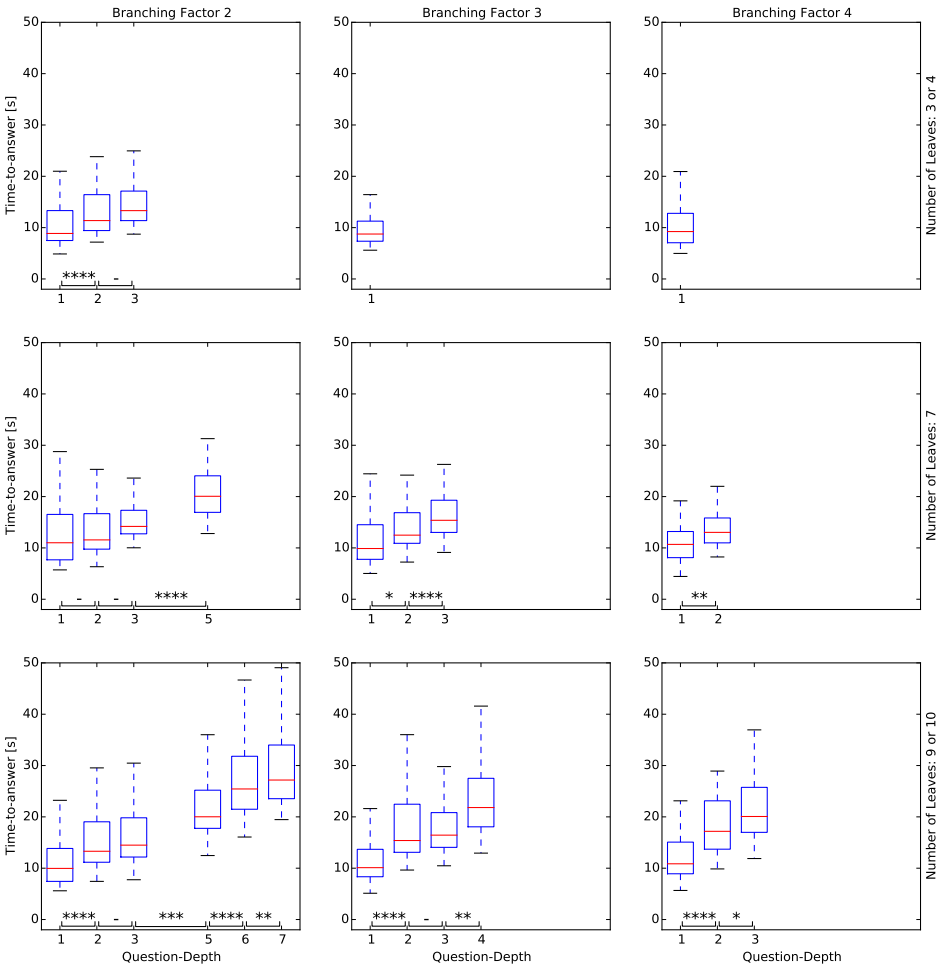


Figure 2.11: The influence of the question-depth on the time-to-answer for a question about a tree that has a given branching factor (column) and number of leaves (row) in the *classify task*. The annotations represent the raw p-values of the Wilcoxon signed-rank tests: a minus sign for a p-value above 0.05 and *, **, *** or **** for p-values below 0.05, 0.01, 0.001 or 0.0001, respectively.

branching factor of 4), and therefore, the trees that have the most similar number of leaves were used instead (4 or 10 leaves instead of 3 or 9, respectively). The horizontal axis in each subplot corresponds to the question-depth, while the vertical axis corresponds to the time-to-answer. The bottom of a box in the boxplots corresponds to the first quartile (i.e., the fastest 25% of the respondents required less time-to-answer), the top corresponds to the third quartile (i.e., the slowest 25% of the respondents required more time-to-answer) and the band inside the box corresponds to the median time-to-answer. The whiskers represent the lowest (typically the fastest answer time) and the highest time-to-answer still within 1.5 interquartile range (IQR) of the lower or the upper quartile, respectively. A time-to-answer above the top whisker is considered an outlier and is not shown in the graph. The time-to-answer has no upper bound, which makes it sensitive to outliers, and

hence, the median and quartiles are observed instead of the mean. The annotations of the boxplots represent the raw p-values of the Wilcoxon signed-rank tests (a non-parametric version of the paired T-test). A minus sign represents a p-value above 0.05 (i.e., not statistically significant) while 1, 2, 3 and 4 stars represents p-values below 0.05, 0.01, 0.001 and 0.0001, respectively. Figure 2.11 shows that increasing the question-depth increases the time-to-answer.

Similar graphs (see Figures 2.11–2.13) were drawn for each combination of the three question/tree parameters, the three respondents' performance measures and the four tasks in the first part of the survey. The results of their analysis are summarised in Table 2.6, while the details are explained in Subsections 2.4.3.1–2.4.3.4). Because there is dense information summarised in the table, an example on how to read the cell in the explain row and question-depth column follows. Increasing the question-depth (d) in the *Explain task* significantly (with very low p-values) increases the time-to-answer in 10 out of 10 tests and significantly (with very low p-values) increases the question-difficulty in 8 out of 10 tests. The answer-correctness decreases with increasing d in 8 out of 10 tests, but the decrease is not statistically significant.

Table 2.6: Summary of the influence of tree structure on comprehensibility. Each cell lists x/y: the number of significant (x) vs. the total number of performed tests (y) for three respondent performance parameters: time-to-answer (⏱), question-difficulty (💡) or answer-correctness (%). Each column shows the results for the studied question parameters: the question-depth (d), the number of leaves (l) and the branching factor (b). Each ratio is accompanied with a comment on the following: the significant tests (i); or the influence of the increased parameter values (j); or the data availability (k).

Task	Question/tree parameter		
	Question-depth (d)	Number of leaves (l)	Branching factor (b)
Classify	⏱**13/18** (72%) ⏱most with a very low p-value[i] 💡**16/18** (89%) 💡some with a very low p-value[i] % no influence[j]	⏱**4/15** (27%) ⏱low p-value for d>1 & b>2[i] 💡**2/15** (13%) 💡only for d>1 and b>2[i] % no influence[j]	⏱**2/12** (17%) ⏱only for d>1[i] 💡**4/12** (33%) 💡only for d>1[i] % no influence[j]
Explain	⏱**10/10** (100%) ⏱most with a very low p-value[i] 💡**8/10** (80%) 💡most with a very low p-value[i] % decreases (8/10) but test are not significant[j]	⏱**1/5** (20%) ⏱only tested for b=2 or d=1[k] 💡**2/5** (40%) 💡tested for b=2 or d=1[k] % no influence[j]	⏱**2/4** (50%) ⏱tested for l=10 or l=4 & d=1[k] 💡**0/4** (0%) 💡tested for b=2 or d=1[k] % no influence[j]
Validate	⏱**6/11** (54%) ⏱most with a very low p-value[i] 💡**6/11** (54%) 💡all with moderate p-value[i] % decreases (9/11) but test are not significant[j]	⏱**3/6** (50%) ⏱significant for b>2 [i] 💡**1/6** (17%) 💡with moderate p-value[i] % no influence[j]	⏱**3/6** (50%) ⏱ for increase of b from 2 to 3[i] 💡**1/6** (17%) 💡tested only for l=10 or d=1[k] % no influence[j]
Discover	⏱💡%trends observed but not tested for significance[j]	⏱💡%trends observed but not tested for significance[j]	⏱💡% no data[k]

Table 2.6 shows that increasing the question-depth most strongly increases the time-to-answer and question-difficulty; it also consistently decreases the answer-correctness (except in the easiest task – *classify*), albeit not at a statistically significant level. Increasing the number of leaves increases the time-to-answer and question-difficulty as well, but only if the questions are not trivial (i.e., a low question-depth and/or branching factor). Furthermore, the p-values are generally higher compared to the corresponding tests for the question-depth. The gap between the influence of the number of leaves and the question-depth is smaller in the *validate* and *discover tasks* than in the *classify* and *explain tasks* because they require analysing a larger part of the tree and not only a single path from the root

to a leaf. The branching factor does not influence the comprehensibility if a question is trivial, but it does have an effect for more complex questions and trees. The p-values for the influence of the branching factor are lower than those for the number of leaves in the *classify* and *explain task*, but comparable in the *validate task*. The analysis for the discover task is limited due to the number of available trees and questions, but the observed trends are in line with the other tasks.

2.4.3.1 The *Classify Task*

Figure 2.11 shows that increasing the question-depth increases the time-to-answer. Each subgraph in the figure shows a Tukey boxplot for a question with a given branching factor and number of leaves. Figure 2.11 also shows that the distribution of the time-to-answer is right-tailed as expected. Despite the limited number of respondents, most of the tests are statistically significant with a low p-value. Therefore we conclude that increasing the question-depth (the depth of the leaf to which an instance corresponds) increases the time-to-answer (the time needed to classify the instance). The result is expected, because the number of attribute values one needs to look up in the attribute-value table is equal to the depth of the leaf in which the instance in question belongs.

Figure 2.12 is drawn with the same methodology as Figure 2.11 except that the vertical axis corresponds to the average question-difficulty and boxplots are replaced with mean values. This simplification is appropriate because the question-difficulty is an integer value between 1 and 5. In contrast, the time-to-answer (shown in Figure 2.11) is a real value without an upper bound, which makes it sensitive to outliers, hence median and quartiles are observed instead of the mean. Figure 2.12 shows that increasing the question-depth increases the question-difficulty. The result is in line with the result about the influence on the time-to-answer, which again confirms that the time-to-answer is a valid objective measure for the analysis of the comprehensibility.

The influence of the question-depth on the answer-correctness is also analysed with the same methodology. However, McNemar's tests are not significant and neither increasing, nor decreasing trends are observed in the graph. The questions in the classify task are so simple (2.4% of the answers were incorrect) that the question-depth does not influence the answer-correctness.

Although it is intuitively expected that increasing the question-depth increases the time-to-answer and question-difficulty, the contribution of the experimental measurements is that they empirically prove this holds even for simple tasks like classification. Therefore we conclude that classification trees are more comprehensible if they correspond to the way humans exchange information: first the important facts and the facts that explain most of the cases should be given with as simple rules as possible, followed by progressively more complicated rules explaining the unusual cases. Leaves that classify many instances should therefore be as high in the tree as possible. This is achieved using the top-down recursive splitting of tree nodes, as well as by using a greedy attribute selection based on the class purity measures applied in most classification-tree learning algorithms. The heuristic bias of such algorithms results in classification trees that try to classify as many instances as possible using as few attributes as possible. Furthermore, the leaves that classify instances corresponding to the class(es) with higher importance should be as high in the tree as possible. This can be achieved using a tree-learning algorithm that takes into account the instance weights and assigning higher weights to the instances belonging to the most important class(es). Note that commonly used tree-learning algorithms are short-sighted (because they are greedy) and use heuristics based on accuracy rather than comprehensibility; therefore the above recommendations should provide a solid basis for improved tree-learning algorithms. Furthermore, the distribution of instances according

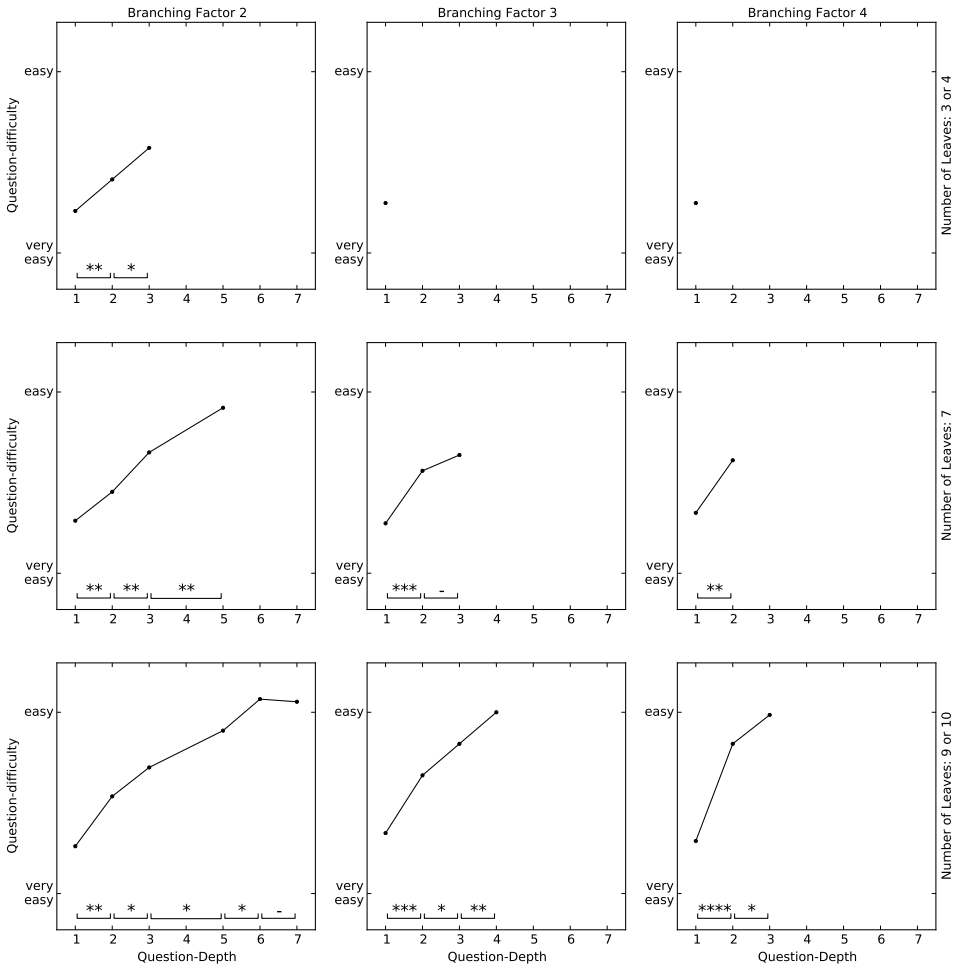


Figure 2.12: The influence of question-depth on the question-difficulty for a question about a tree that has a given branching factor (column) and number of leaves (row) in the *classify task*. The annotations represent the raw p-values of the Wilcoxon signed-rank tests: a minus sign for a p-value above 0.05 and *, **, *** or **** for p-values below 0.05, 0.01, 0.001 or 0.0001, respectively.

to the depth of the leaves to which they correspond should be taken into account by tree comprehensibility metrics as well.

Second, the influence of the number of leaves on the comprehensibility is analysed with the same methodology as before. It does not affect the answer-correctness (in the *classify task*). It also does not influence the time-to-answer (Figure A.1) or question-difficulty (Figure A.2) if the branching factor is 2 or if the question-depth is 1; nevertheless, it increases both of them if the branching factor and/or question-depth is higher. Only the nodes on a path from the root of the tree to the leaf to which the instance in question belongs need to be examined in order to classify the instance. The rest of the tree is not relevant, therefore the number of leaves in the entire tree has a limited effect on the time-to-answer. The result empirically confirms that the number of leaves (which is a tree

complexity measure) is indeed a valid surrogate for comprehensibility, but only if leaves with high enough depth in trees with branching factor greater than 2 are considered, otherwise the number of leaves is not relevant. Whether the number of leaves influences the comprehensibility of binary trees with more than 10 leaves should be analysed in future work, because we observed increasing trends of the question-difficulty and time-to-answer when increasing the number of leaves (but the differences are not statistically significant). Furthermore, according to the p-values, the influence of the question-depth is stronger than the influence of the number of leaves, hence the question-depth can be considered more important. However, this comparison may not be fair since the relative differences between the compared parameter values are expressed in different quantities, which are difficult to compare.

Third, the effect of the tree branching factor on the comprehensibility is analysed. There is no effect on the answer-correctness. Increasing the branching factor increases the time-to-answer (Figure A.3) and question-difficulty (Figure A.4) only if the question-depth is at least 2 and the tree has at least 9 leaves. The p-values of the statistical tests are even lower than for the influence of the number of leaves. In binary trees, attribute-value of the instance in question must be compared only with the value on the left (or right) branch in order to decide to which subtree the instance belongs. In trees with higher branching factor, it needs to be compared with values on several branches until the branch with the matching value is found. Each comparison requires additional time, however the time needed for a comparison is relatively short compared to the time needed to look up a value of an attribute in the attribute-value table. Therefore, the influence of branching factor on time-to-answer is more evident for questions with higher question-depth. We conclude that branching factor does not influence the comprehensibility if a question is trivial, however it does have an effect for more complex questions and trees. Hence binary trees should not be preferred to trees with larger branching factors (3 or 4) for trees with up to 10 leaves. Note that this result should not be generalized to higher branching factors and trees with more leaves, which should be analysed in future work. Furthermore, the attributes with 3 or 4 values were computed as Cartesian products of two binary attributes, which may reduce the influence of the higher branching factors because aggregating related concepts may make information easier to comprehend compared to a more verbose or granulated presentation (Sommer, 1995).

2.4.3.2 The *Explain Task*

The analysis of the *explain task* is performed using the same methodology as for the *classify task* and the results are similar. Increasing the question-depth increases the time-to-answer (Figure A.5) and question-difficulty (Figure A.6); the levels of statistical significance are comparable to the ones in the *classify task*. The differences in time-to-answer and question-difficulty between two questions with consecutive depths are greater compared to the differences in the *classify task*. This can be partially attributed to the fact that the number of clicks needed to answer a question is equal to the depth of the question, while a single mouse click is needed in the other tasks. In contrast with the *classify task*, the answer-correctness decreases with increasing question-depth (Figure 2.13), but the relation is not statistically significant (p-values above 0.1). Increasing the number of respondents in future work will clarify this result.

Increasing the number of leaves slightly increases the median time-to-answer, but the differences are statistically significant (p-value 0.02) only for branching factor 4 and question-depth 1, while the other 4 comparisons are not statistically significant (p-values from 0.057 to 0.60). Note that comparisons for the high branching factor and question-depth were not performed due to the reduced number of questions in the explain task (the

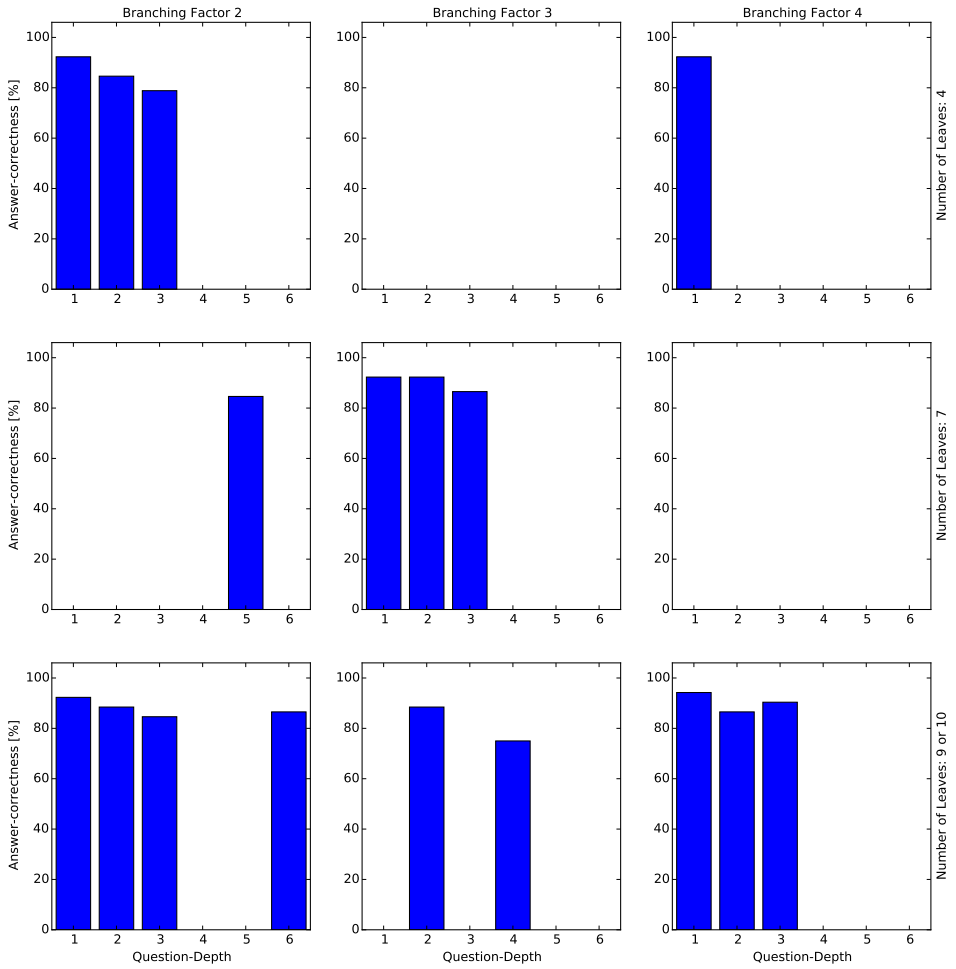


Figure 2.13: The influence of question-depth on the answer-correctness for a question about a tree that has a given branching factor (column) and number of leaves (row) in the *explain task*.

most time demanding task for respondents). This should be checked in future work, since the data suggests that the number of leaves might have a statistically significant effect on the time-to-answer for difficult questions. The influence on the question-difficulty is statistically significant in 2 out of 5 comparisons. The available results agree with the findings in the classify task: increasing the number of leaves from 3 or 4 to 9 or 10 increases the time-to-answer and question-difficulty, while there is no influence on the answer-correctness.

Increasing the branching factor increases the time-to-answer and question-difficulty for questions about big trees (10 leaves). The p-values are comparable with the p-values in the *classify task* for the time-to-answer, but the differences in the question-difficulty are not statistically significant. The relation could not be checked for smaller trees due to the limited number of questions.

2.4.3.3 The *Validate Task*

The analysis of the *validate task* is again done in the same way as for the previous two tasks. The number of questions is higher than in the *explain task*, but a survey-design drawback reduces the number of possible comparisons to a similar number as in the *explain task*: verifying negated statements increases the time-to-answer and question-difficulty, and reduces the answer-correctness compared to verifying positive statements. Table 2.7 shows data about the performance of respondents' asked to check whether a tree agrees with a positive or a negative statement. The first column in the table lists the median time-to-answer, mean question-difficulty and mean answer-correctness for the question with question-depth 2 that asked whether a binary tree with 7 leaves agrees with a negated statement. The following rows give the same statistics for questions about more complex trees or with higher question-depth that ask whether the tree agrees with a positive statement. The question about negated statement is more difficult according to all three performance measures compared to the questions about positive statements. Table 2.7 shows that the effect of negating a statement is greater than the effects of the question-depth, number of leaves and branching factor. Therefore negated statements should be avoided in classifier representation if possible (Huysmans et al., 2011). The result confirms the importance of simplicity as a comprehensibility dimension (Göpferich, 2009). Hence questions with a negated statement are treated as a survey design flaw and are omitted from the analysis. The ability to detect this design drawback verifies that the survey design enables detecting relatively small differences with a reasonably small number of respondents.

Table 2.7: The influence of negated statements on the respondents' performance: the first column gives data about a question that asked the respondents to check whether a tree agrees with a negated statement; the following columns are about more complex trees but asking to check a positive statement.

Negated statement	**yes**	no	no	no
Question-depth	2	**3**	2	2
Branching factor	3	3	**4**	3
Number of leaves	7	7	7	**9**
Time-to-answer [s]	17.8	13.7	12.1	14.7
Question-difficulty [1-5]	2.0	1.8	1.7	1.8
Answer-correctness [%]	82.1	96.4	100	100

Increasing the question-depth increases the time-to-answer and question-difficulty; the result could be checked for big trees (9 or 10 leaves) with the branching factor 2–4 and a small tree (4 leaves) with the branching factor 2, and is statistically significant. The influence is not as strong as in the *classify* and *explain* tasks. Furthermore, the influence is stronger in trees with the branching factor 2 (p-values below 0.01) compared to trees with the branching factor 3 or 4 (p-values close to 0.05). There is no effect on the answer-correctness. Increasing the number of leaves increases the time-to-answer if the branching factor is at least 2 (statistically significant) and it also slightly increases the question-difficulty (not statistically significant). The influence of the branching factor could be checked for big trees (9 or 10 leaves) and question-depths 1–3. Increasing the branching factor increases the time-to-answer; the result is significant for the increase of the branching factor from 2 to 3 but not for the increase from 3 to 4. The influence on the question-difficulty is minor; it is statistically significant (p-value = 0.003) only in 1 out of 6 comparisons.

2.4.3.4 The *Discover Task*

The analysis for the *discover task* is limited by the number of the special trees that enable discovering unusual properties for a given class of animals (i.e., outliers). The survey includes four such trees with 3, 5, 7 and 11 leaves, and branching factor 2, with the exception of two trees that also use the number of legs attribute with 3 possible values. Therefore, the influence of the branching factor cannot be studied. In general, an increase in the time-to-answer and question-difficulty occurs as a result of an increased number of leaves. Since the question-depth cannot remain fixed, only one direct comparison is possible: increasing the number of leaves from 5 to 11 for a question at depth 3 increases the time-to-answer (p-value < 0.01) and slightly increases the question-difficulty (not statistically significant). The number of leaves has a greater influence on the respondents' performance in the *discover task* compared to the previous three tasks because discovering new knowledge from a tree requires at least scanning through if not reading the entire tree, instead of following a single path from the root to the leaf.

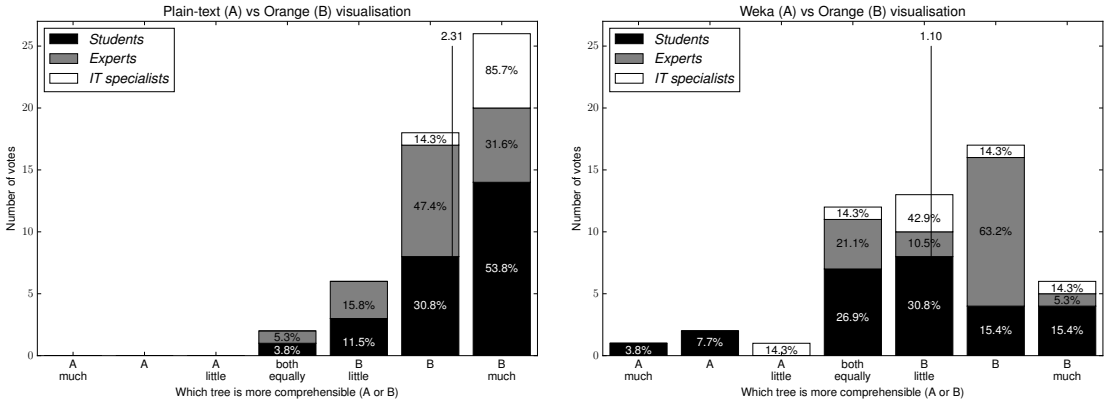
The question-depth in the *discover task* depends on the strategy that a respondent uses to answer the question. It can either be the depth of the node where instances from the class in question are split into two subtrees (top-down strategy using the number of class instances in a node), the depth of the deepest leaf containing instances belonging to the class (bottom-up strategy using class distribution pie charts), or the depth of the tree (exhaustive search strategy). Regardless of the strategy, increasing the question-depth increases the time-to-answer (p-value < 0.05) and question-difficulty (p-value < 0.1), as well as decreases the answer-correctness (p-value < 0.05 only for the comparison between question-depths 3 and 7 in the same tree). The influence of the question-depth is smaller in the *discover task* then in the previous ones.

2.4.4 The Influence of Tree-Visualisation Properties

In the *compare task*, we analyse the influence of the tree visualisation properties on the comprehensibility. The respondents' ratings on the seven-point scale (Section 2.3.5.4) are mapped to integer values between -3 and 3. The rating 3 means that the visualisation B is much more comprehensible than the visualisation A, 2 that B is more comprehensible, 1 that B is slightly more comprehensible, and 0 that they are equally comprehensible. The negative marks represent the inverse relations: A is more comprehensible than B.

The respondents' opinions about the influence of the tree visualisation style on the comprehensibility are shown with histograms in Figure 2.14. The left histogram compares the plain-text visualisation (Figure 2.4a), and the right histogram compares the Weka visualisation (Figure 2.4b) with the Orange visualisation of the same trees. The vertical lines show the average ratings. The respondents find the plain-text visualisation to be considerably less comprehensible than the Orange visualisation: the average rating is 2.31, which is the highest among all of the questions in the *compare task* (Figure 2.14a). Therefore, the plan-text visualisation should be avoided when analysing classification trees. The majority of respondents prefer the Orange over the Weka visualisation (Figure 2.14b), but the average rating of 1.1 is lower than in the comparison with the plain-text visualisation. This finding shows that the additional colour-coded information is beneficial.

Second, the influence of the pie charts (which represent class distributions, see Figures 2.5a and 2.5c) in the Orange visualisation is compared using the histogram shown in Figure 2.15a. The result is similar to the result above, but the average rating is lower, which suggests that pie charts are the main but not the only difference between the Weka and Orange visualisations. The remainder can be attributed to minor differences such as colour-coding the number of examples in the leaves, and the respondents' bias toward the

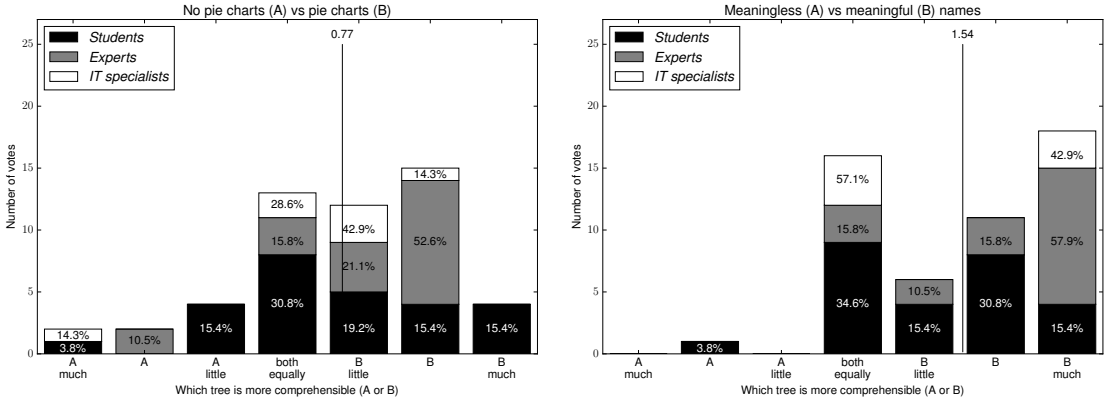


(a) (b)

Figure 2.14: The influence of the tree visualisation style on the tree comprehensibility.

Orange visualisation introduced by the survey design. The result confirms the importance of perceptibility as an important cognitive dimension (Göpferich, 2009).

Third, the importance of the attribute names, attribute values and class names is analysed by comparing simple trees with meaningful (Figure 2.5a) and random (Figure 2.5b) values and names. The results are presented in Figure 2.15b: the average rating (1.54) indicates that meaningful names are more important for comprehensibility than the class-distribution pie charts. This result empirically confirms the importance of the simplicity as a cognitive dimension (Göpferich, 2009) and the fact (used in a related survey design (Allahyari & Lavesson, 2011)) that meaningful concept names make information easier to remember and support faster processing and the triggering of associations with the user's domain knowledge, which facilitates comprehension. Therefore, we suggest using meaningful names in tree visualisations whenever possible, and we warn against using misleading or ambiguous names.



(a) (b)

Figure 2.15: The influence of visualised information on the tree comprehensibility.

Fourth, we analyse the influence of the tree layout on its comprehensibility. A tree shown in Figure C.2 is compared with the same tree when drawn using the same visualisa-

tion style but a random layout – with random relative horizontal positions for the sibling nodes. The layouts of both trees are illustrated in Figure 2.6. Although almost half of the respondents rated the trees as equally comprehensible, 42% preferred the tree with the left-to-right layout (suggested in Section 2.3.4), and less than 10% preferred the random layout. This is in line with the perceptibility cognitive dimension (Göpferich, 2009). The average rating (0.56) is the lowest among all of the results presented in this section; nonetheless, we believe that the importance of the tree layout should be investigated further, especially in larger trees and for users whose native reading direction is not from left to right.

Finally, we analyse the influence of the tree structure parameters on the tree comprehensibility. Over 90% of the respondents found a tree with 4 leaves to be more comprehensible that a tree with 10 leaves (Figure 2.16a), which is in line with the results from the previous tasks and related work (Allahyari & Lavesson, 2011; Huysmans et al., 2011). The average rating was 1.92, which is more than the influence of meaningful names but less than the difference between plain-text and Orange visualisation.

Comparing a tree with 9 leaves and a branching factor of 3 with its binary version (Table 2.2, bottom row left and middle) exhibits an inconclusive but interesting result shown in Figure 2.16b. We cannot say which tree is more comprehensible: while 42% of the respondents think that there is no difference or only a slight difference, the majority think that one or the other tree is more or much more comprehensible. However, the preferences of these respondents are opposing: one half prefers the deeper tree with binary splits, while the other half prefers the shallower tree with a branching factor of 3. This finding could explain why a limited influence of the branching factor is observed in the previous four tasks. The influence of the branching factor on the comprehensibility should be investigated further in domains in which there are multiple values of nominal attributes, which naturally yield trees that have higher branching factors.

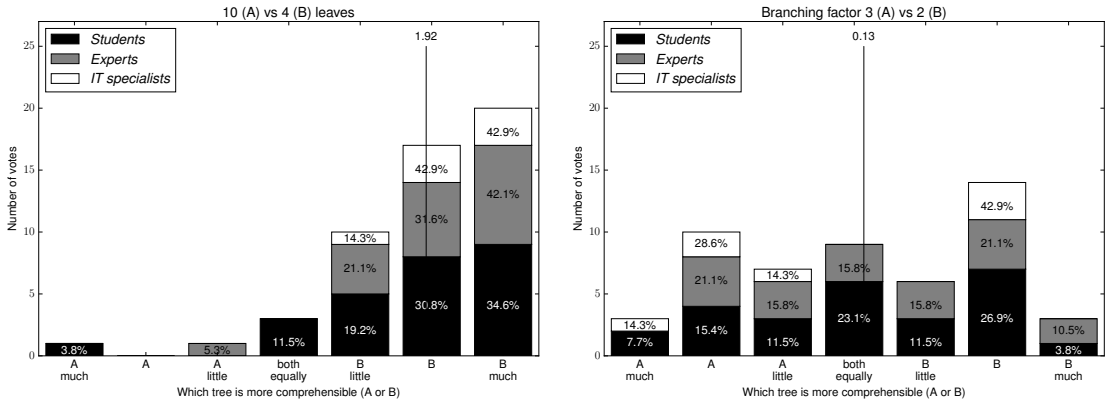


Figure 2.16: The influence of the tree structure on the tree comprehensibility.

Finally, the influence of the tree depth is analysed by comparing trees with 9 leaves and a branching factor of 3: one with a depth of 2 and the other with a depth of 4 (Table 2.2, bottom middle cell). The average rating is 0.15. No difference in the comprehensibility is noticed by 35% of the respondents, and only 29% noticed more than a slight difference: approximately half of them prefer the shallower tree, and the other half prefer the deeper tree. We conclude that the tree depth itself does not have an important influence on the comprehensibility, which is the reason why we decided to omit an in-depth analysis of the tree depth for the first four survey tasks. The other reason is that the number of tree

pairs with different tree depths but no difference in the other tree structure parameters is very limited. Hence, similar trees with different depths would have to be compared, but the small influence of the tree depth on the comprehensibility would then drown in the influence of the other parameters.

2.4.5 The Comprehensibility Metrics

The comprehensibility ratings of the 11 trees (illustrated in Table 2.2) obtained in the *rate task* are used to analyse the match between various tree comprehensibility metrics and the respondents' subjective opinions and to suggest new comprehensibility metrics.

The ratings are shown in Figure 2.17 with a stacked bar chart. Each bar corresponds to a tree described with three parameters: L for the number of leaves, B for the branching factor and D for the depth of the tree. The horizontal axis corresponds to the percent of respondents that gave a certain rating and the colours of the bars correspond to the ratings ranging from *very easy to comprehend* (green) to *very difficult to comprehend* (red). Figure 2.17 shows that respondents' opinions differ but the variability is limited. Almost all the respondents' ratings are within ± 1 of the average answer. This finding suggests that the used rating scale is valid. Furthermore, the opinions show that trees with more leaves are more difficult to comprehend. More leaves in turn causes a higher tree depth, and therefore, increasing the tree depth appears to decrease the tree comprehensibility. However, according to the respondents' subjective ratings, the tree depth and branching factor do not have an obvious (or consistent) effect on the comprehensibility if the number of leaves is fixed. These results agree with the results obtained in the first part of the survey.

Note that the complexity and comprehensibility of the analysed trees is limited: the average ratings of all of the trees are below *difficult to comprehend*. The results are therefore valid for trees that have modest complexity and should not be overgeneralised to more complex trees.

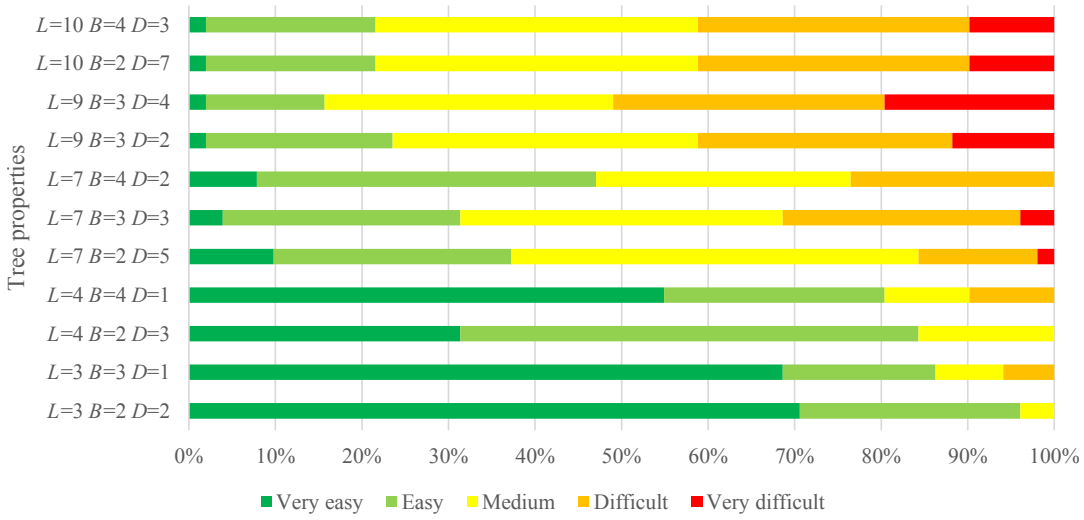


Figure 2.17: Respondents' subjective ratings of the tree comprehensibility for trees with various numbers of leaves (L), branching factors (B) and tree depths (D).

In addition to the average subjective opinion about the comprehensibility of each tree, two additional measures were calculated and used as the gold standards for evaluating tree

comprehensibility metrics. The first one is the sum of the median time-to-answer (over all the respondents) and the second one is the sum of the average question-difficulty (over all the respondents). Both sums are over all the leaves in a tree. The data obtained in the *classify task* was used to calculate the two gold standards because this is the only task in which one question for each possible question-depth in each of the trees was included in the survey. Furthermore, this is the task that was answered by the highest number of the respondents. The advantage of the two gold standard measures based on the respondents permanence over the comprehensibility rates is that they are objective.

The Pearson product-moment correlation coefficient (Pearson's r) between the sum of the median time-to-answer and the sum of the average question-difficulty is 0.987 and the Spearman's rank correlation coefficient (Spearman's ρ) is 1.0. This shows that this two measures are well aligned. The Pearson's r between the average comprehensibility rate and the sum of the median time-to-answer is 0.903 and the Spearman's ρ is 0.936. The Pearson's r between the average comprehensibility rate and the sum of the average question-difficulty is 0.951 and the Spearman's ρ is 0.936. This shows that the average subjective opinions about the tree comprehensibility do not match exactly with the respondents' performance when answering questions about each leaf in the tree. Therefore the two gold standard metrics based on respondents' performance provide an additional view when evaluating the tree comprehensibility metrics.

We evaluated several tree complexity metrics that are often used as surrogates for the comprehensibility metrics based on the Pearson's r and Spearman's ρ. Spearman's ρ measures whether the ranking of trees according to a tree comprehensibility metric is in line with the ranking according to a gold standard. Spearman's ρ with absolute value 1 means that the rankings match perfectly, while value 0 means that there is no correlation between the two rankings. Pearson's r measures the linear correlation between a comprehensibility metric and a gold standard, giving a value between $+1$ and -1 inclusive, where 1 is total positive correlation and 0 is no correlation. The Spearman's ρ values are shown in Table 2.8 and the Pearson's r values are shown in Table 2.9. Each row in the two tables corresponds to one of the three gold standards and each column corresponds to a classification tree metric. The correlations of the tree branching factor were also calculated, but are low (< 0.17) and therefore not shown in the tables.

We propose and evaluate two new tree comprehensibility metrics. The first metric c_{D} (Eq. (2.1)) is the weighted sum of the depths d_l over all n leaves in a tree. The second metric c_{B} (Eq. (2.2)) is the weighted sum of the branching factors b_i on the paths from the root of the tree to a leaf l over all n leaves. The correlation coefficients shown in the tables were obtained using the default weights $w_l = 1$.

$$c_{\mathrm{D}} = \sum_{l=1}^{n} w_l d_l \tag{2.1}$$

$$c_{\mathrm{B}} = \sum_{l=1}^{n} w_l \sum_{i \in path(root,l)} b_i \tag{2.2}$$

Tables 2.8 and 2.9 show that the correlations of the tree depth are the lowest, although not negligible (unlike correlations of the branching factor). The number of leaves and the number of nodes correspond well to all three gold standards (correlations > 0.9), but the number of leaves slightly outperforms the number of nodes. The correlations of metric c_{D} are considerably higher than the correlations of the tree depth, but slightly lower than the correlations of the number of leaves or nodes. The correlations of metric c_{B} are the highest among the evaluated tree comprehensibility metrics, except the Pearson's r with the subjective comprehensibility rates is lower than for the number of leaves. The

Table 2.8: The Spearman's rank correlation coefficient between tree comprehensibility metrics and three gold standards.

Gold standard	Number of leaves	Number of nodes	Tree depth	c_{D}	c_{B}
Comprehensibility rate	0.93	0.90	0.65	0.83	**0.94**
Time-to-answer	0.98	0.97	0.76	0.93	**1.00**
Question-difficulty	0.98	0.97	0.76	0.93	**1.00**

Table 2.9: The Pearson product-moment correlation coefficient between tree comprehensibility metrics and three gold standards.

Gold standard	Number of leaves	Number of nodes	Tree depth	c_{D}	c_{B}
Comprehensibility rate	**0.97**	0.88	0.57	0.74	0.91
Time-to-answer	0.96	0.98	0.77	0.92	**0.99**
Question-difficulty	**0.99**	0.95	0.69	0.86	**0.99**

problem with the number of leaves as a comprehensibility metric is that it assigns the same comprehensibility to the trees with the same number of leaves regardless of the branching factor (Figures 2.18a and 2.18b). In contrast, using c_{B} as a comprehensibility metric correctly ranks trees with the same number of leaves according to their comprehensibility (Figures 2.18c and 2.18d), which reflects in a higher Spearman's ρ.

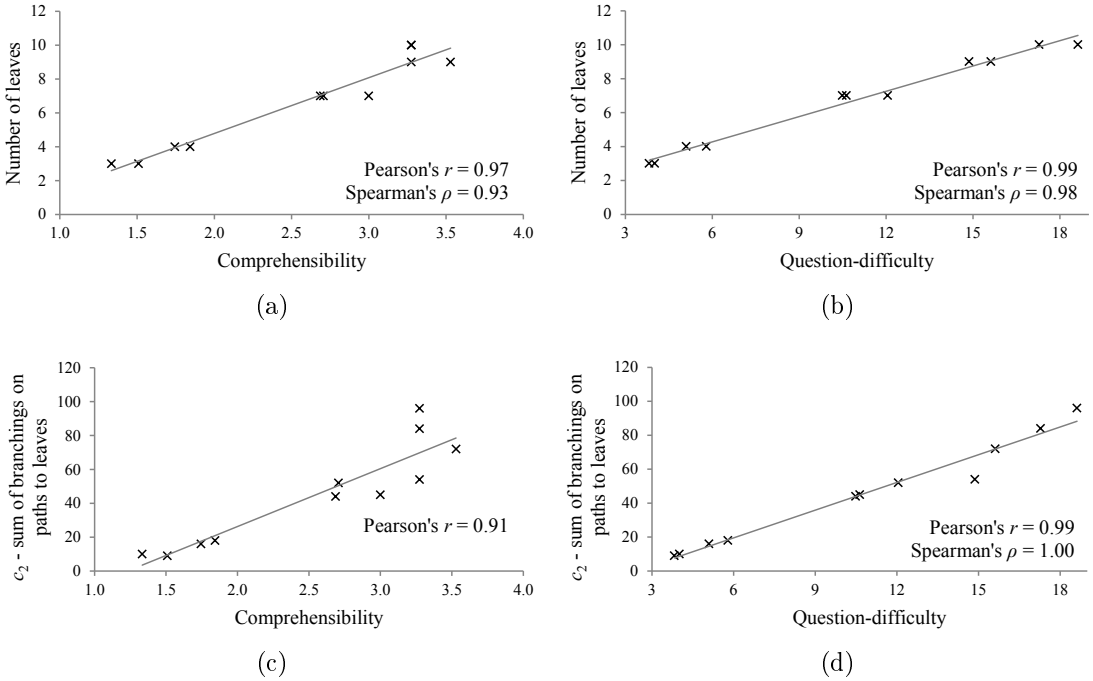


Figure 2.18: Correlations between comprehensibility metrics (the number of leaves in a tree and c_{B}) and gold standards for evaluation: the average respondent ratings (1: very easy to comprehend, 5: very difficult to comprehend) and the sum of question-difficulties over all the leaves in a tree.

Another advantage of the two proposed comprehensibility metrics c_{D} and c_{B} is that they can be weighted by the parameter w_l, which is defined in each leaf l. The weighting is important because it enables specifying user preferences that are related to the semantics of a classification tree, rather than relying solely on the tree structure to estimate its comprehensibility. For example, weighting by the number of instances in a leaf (mentioned by Sommer (1995)) corresponds to preferring the trees that classify the most instances with shallow leaves (i.e., simple classification rules). Another example is weighting by the importance of a class, which corresponds to preferring the trees that classify instances that belong to the important class(es) with short rules. Both comprehensibility metrics account for the question-depth, which is defined per leaf and has the highest influence on the question-difficulty and time-to-answer among all of the analysed tree structure parameters. The Pearson's correlation between metric c_{B} and the average comprehensibility rate increases from 0.91 to 0.94 if the weighting is set by the number of instances in a leaf n_l divided by the number of all instances n ($w_l = n_l/n$).

2.5 Conclusions and Future Work

In this chapter we analyse how tree structure parameters and visualisation style influence the comprehensibility of classification trees. The results are based on empirical data obtained using a carefully designed survey with 98 questions answered by 69 respondents. The survey design allows detecting relatively small differences in the comprehensibility even with a limited number of respondents. Furthermore, the survey implementation enables analysing the effect of a single tree-structure parameter independently of the other parameters. The 11 trees used in the survey have 3 to 11 leaves, 4 to 20 nodes, branching factors between 2 and 4, and tree depths between 2 and 7. Therefore, the conclusions are valid for trees that have modest complexity (commonly used in practice) and should be generalised to large trees with caution. The respondents' performance is measured as the time-to-answer and answer-correctness in four tasks: *classify* an instance, *explain* the classification of an instance, *validate* that the tree agrees with a domain-knowledge rule, and *discover* new knowledge about unusual instances. The survey measures the tree usability directly and objectively, while the tree comprehensibility is measured objectively but indirectly (through its usability). In addition, the respondents' subjective opinions about comprehensibility are collected by asking how difficult each question is in the first four tasks (question-difficulty), how comprehensible the tree is (the *rate task*) and by comparing the comprehensibility of pairs of trees (the *compare task*) that differ in their visualisation properties.

The main contributions of this analysis are the quantification and analysis of the influence of (i) the tree structure parameters, (ii) the question-depth, (iii) the tree visualisation properties, (iv) data-mining experience on the tree comprehensibility, and (v) evaluation of several commonly used and two new classification-tree comprehensibility metrics.

The time-to-answer is highly correlated to the question-difficulty, and therefore, the two measures are well suited for measuring the classification tree comprehensibility in the absence of direct and objective measures. This finding also confirms that the question-difficulty scale is well-designed. The variability of subjective opinions is considerable, which emphasises the subjective nature of comprehensibility. Data mining experts perform better than students, which shows that more experienced classification tree users comprehend the trees faster and better. The difference in performance is the most evident in the *discover task*. The *classify task* is the easiest, and the *discover task* is the hardest (of the four tasks), according to the question-difficulty, answer-correctness and time-to-answer.

Tree-structure properties clearly influence the time-to-answer and question-difficulty,

but they have a limited effect on the answers-correctness. Increasing the **number of leaves** (which in turn increases the number of nodes and the tree depth) increases the time-to-answer and question-difficulty. The result is statistically significant in the *classify* and *explain* tasks for non-trivial questions. The influence on the time-to-answer is also significant in the *validate* task. The same trend is observed in all of the other cases, but the results are not statistically significant. Increasing the number of leaves clearly makes a tree less comprehensible, as confirmed by the *rate* and *compare* tasks. The result empirically supports that the number of leaves is a usable tree comprehensibility metric, but only if leaves with sufficiently high depths and trees with branching factors of greater than 2 are considered. Although the number of leaves and the tree depth correspond well to the comprehensibility of the entire tree, they are not relevant if the needed information is close to the root of the tree or if it is contained in a single path from the root to a leaf of the tree. Finally, the results show that the number of leaves has a greater influence on the tree comprehensibility in the *discover task* compared to the other three tasks because discovering new knowledge from the tree requires at least scanning through if not reading and understanding the entire tree.

Increasing the **branching factor** increases the time-to-answer for non-trivial questions (not analysed in the *discover task*). Increasing the branching factor also increases the question-difficulty; however, the result is statistically significant only in the *classify task*. Higher branching factors and bigger trees should be investigated further because there are two types of respondents that have opposite preferences: half of them prefer a binary tree, while the other half prefer a tree with a branching factor of 3 (according to the *compare task*). This finding could explain why a limited influence of the branching factor is observed in the first part of the survey. In addition, further work should focus on trees that have non-uniform branching factors.

The **tree depth** itself does not have an important influence on the tree comprehensibility and strongly depends on other tree structure parameters (the number of leaves and branching factor); therefore, its influence on the comprehensibility cannot be systematically analysed.

Increasing the **question-depth**, which corresponds to the depth of the deepest leaf that is required to answer a question, increases the time-to-answer and question-difficulty. The influence is statistically significant in almost all of the comparisons. Although this finding is intuitively expected, the contribution of our study is that it empirically proves the result even for a simple task such as classification. In addition, increasing the question-depth decreases the answer-correctness in all of the tasks except for the easiest task (*classify*), although this result is not statistically significant. Increasing the number of respondents would clarify the observed trend. The question-depth has the highest influence on the comprehensibility among all of the observed parameters and is overlooked in the related work.

The results on the question-depth suggest that users prefer trees that follow the following principle: the most important information and information about the most common instances should be given at the top of the tree, i.e., with short rules, while the remaining instances can be classified with deeper leaves. The relative importance and distribution of the instances according to the depth of the leaves to which they correspond should be accounted for by the tree comprehensibility metrics and tree learning heuristics.

Validating **negated statements** increases the time-to-answer and question-difficulty and reduces the answer-correctness compared to validating positive statements. Therefore, negated statements should be avoided in classifier representations whenever possible. The effect of negating a statement is greater than the effects of increasing the question-depth, number of leaves or branching factor.

The **tree visualisation style** has an important influence on the tree comprehensibility and usability. Plain-text visualisation should be avoided while colour-coded information is beneficial. Meaningful attribute names, attribute values and class names should be used in tree visualisation whenever possible, while misleading and ambiguous names should be avoided.

The results also indicate that the **tree layout** is important for many users, and thus, its influence on the comprehensibility should be verified with objective measures. The tree layout algorithm introduced in Section 2.3.4 can be used as a starting point for further research and to improve the visualisation of classification trees.

Finally, we define two new **classification-tree comprehensibility metrics** (Section 2.4.5) that are in line with the survey results: the weighted sum of the depths of the leaves and the weighted sum of the branching factors on the paths from the root to the leaves. Both metrics can be weighted by a parameter that is defined in each leaf, which enables specifying user preferences that are related to the semantics of a classification tree rather than relying solely on the tree structure to estimate its comprehensibility. The two proposed metrics are evaluated and compared with several tree complexity measures that are often used as surrogates for comprehensibility metric.

In **future work**, we plan to study the influence of the number of leaves in larger (binary) trees and the branching factor in trees with higher (and non-uniform) branching factors, which results from nominal attributes with more than two possible values. Studying trees with higher complexity as well as increasing the number of respondents will clarify the results about the influence on the answer-correctness. Another promising direction for future study is analysing the influence of the visualisation properties, especially the tree layout, using the objective measures (time-to-answer, answer-correctness). Including respondents from various cultures (e.g., who differ in the reading direction of their languages) would confirm whether the reported results are culture specific. The results of this study should be confirmed in real-life domains that involve strong numeric components, such as in medical (e.g., stress level and activity monitoring) insurance or bank loans applications. Furthermore, data mining methods could be employed instead of the statistical tests to explain the interplay of various parameters, and derive a formal model of classification-tree comprehensibility. A deeper investigation of the influence of the question-depth could ultimately lead to learning algorithms that will produce more comprehensible trees.

Chapter 3

Learning Comprehensible and Accurate Hybrid Trees

3.1 Introduction

In Chapter 2 we analysed comprehensibility of classification trees. The results evaluate the influence of individual tree properties on comprehensibility and can be used to derive better comprehensibility metrics and develop heuristic functions that will enable learning algorithms to find more comprehensible trees. However, it is not enough for a classifier to be comprehensible; it must also be accurate. Practical data mining applications often deal with complex domains forcing the developers to choose between either incomprehensible but accurate classifiers (e.g., SVM, ANN, ensembles) or comprehensible classifiers with considerably lower accuracy (e.g., classification trees, rules or tables). What is missing are classifiers that sacrifice some accuracy in order to gain comprehensibility or vice versa. The goal of this thesis is thus to learn classifiers that provide a good trade-off between accuracy and comprehensibility, while treating both objectives as equally important. The desired classifiers should resemble the way human experts in various complex domains represent their knowledge. Namely, they can often explain only parts of their knowledge with simple rules. The rest of their knowledge is tacit, so while they can apply it, they cannot easily express it. Hybrid tree is chosen as the classification model to enable the trade-off between the two objectives. The regular leaves of the hybrid tree correspond to comprehensible knowledge because they offer an explanation for the classification of the instances that fall into the regular leaves. On the other hand, instances that fall into the so-called black-box leaves are classified with a black-box classifier, which corresponds to the tacit knowledge. The black-box leaves do not offer classification explanation, but they increase the overall accuracy of the hybrid tree. The goal of the MOLHC algorithm is therefore to find a hybrid tree that classifies the simple parts of the domain with regular leaves and the complex parts of the domain with black-box leaves.

The first difficulty of learning accurate and comprehensible classifiers arises from the fact that the two objectives must be considered at the same time and that they are conflicting (Jin & Sendhoff, 2008) – increasing one decreases the other and vice versa (except if the classifier overfits the training data). The two objectives can be achieved with two general approaches (Jin & Sendhoff, 2008; Freitas, 2004). The traditional weighted-formula approach transforms the multi-objective problem into a single-objective one. The second approach, which is becoming increasingly more popular in the last two decades, is Pareto-based multi-objective approach. Its objective function is not a single scalar value but a vector, which enables treating all the objectives simultaneously and independently. As a consequence, the output of a Pareto-based multi-objective approach is a set

of Pareto-optimal (i.e. non-dominated) classifiers (Deb, 2001) instead of a single classifier. Freitas (2004) analysed the arguments for and against each of the two approaches: the Pareto-based approach is more complex, but it is preferred because it avoids multiple runs of a single-objective optimisation algorithm, does not require the ad-hoc specification of optimisation parameters (i.e. weights), and in addition provides an informative set of Pareto-optimal classifiers (Deb, 2001). Therefore we use the Pareto-based multi-objective approach in order to achieve the goal of this thesis.

The second problem of learning accurate and comprehensible classifiers is choosing an appropriate knowledge representation that offers a whole range of classifiers – from the most comprehensible ones to the most accurate ones. We propose using hybrid trees that classify instances falling into some of their leaves with a black-box classifier. On one hand, hybrid trees offer the comprehensibility of the regular classification trees, and on the other hand, they enable trading comprehensibility for improved classification accuracy in the complex part of the domain. The purpose of this thesis is not to improve classification accuracy of classification trees, but to provide a good trade-off between comprehensibility and accuracy by combining a comprehensible tree and an accurate black-box classifier.

This chapter describes the algorithm for Multi-Objective Learning of Hybrid Classifiers (MOLHC) (Piltaver, 2010; Piltaver, Luštrek, Zupančič, et al., 2014), which learns the set of non-dominated hybrid trees using the Pareto-based approach. It proves that the MOLHC algorithm finds the complete set of non-dominated hybrid trees, empirically evaluates the algorithm results and proposes a data-mining process designed around the MOLHC algorithm. The proposed interactive and iterative data-mining process shown in Figure 3.1 advocates using a domain expert in the loop in order to combine the expert's knowledge and the power of machine learning. This is in a way similar to the classical data mining process except that the expert inspects a set of hybrid classifiers instead of a single classifier in each iteration. The advantage of this process is that it enables the expert to make a well-informed decision when selecting a classifier with a good trade-off between the accuracy and comprehensibility, and to obtain additional insights into the classification domain that would be difficult to obtain using the traditional (i.e. single-objective) data mining algorithms.

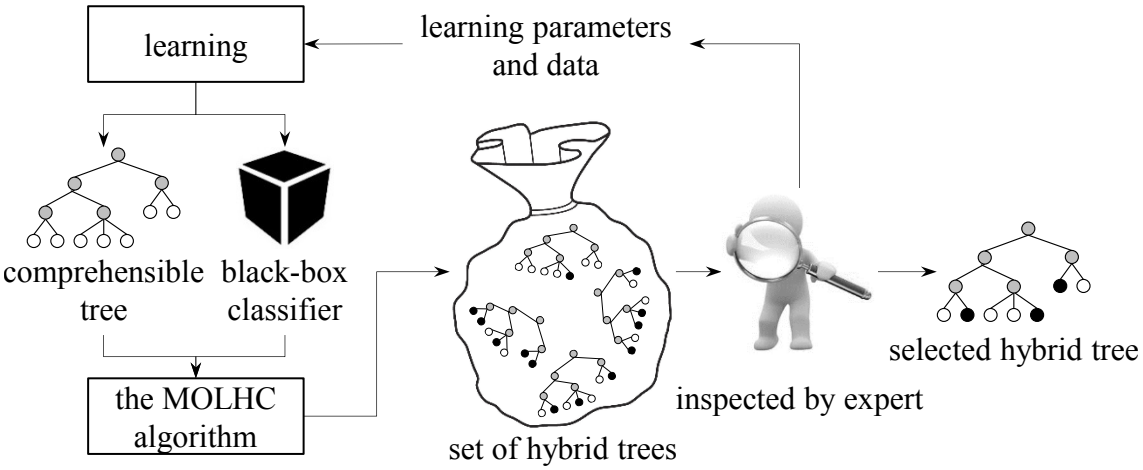


Figure 3.1: The proposed data-mining process uses multi-objective learning and a domain expert in a loop to combine a comprehensible tree and an accurate black-box classifier into a single hybrid tree.

This chapter is organized as follows. A review of literature related to multi-objective learning is given in Section 3.2. Section 3.3 describes the hybrid-tree classification model

and defines a comprehensibility metric associated with it. The pseudo code and explanation of the MOLHC algorithm is given in Section 3.4, while the formal proof that the MOLHC algorithm finds the complete set of non-dominated hybrid trees is in the Appendix D. The algorithm evaluation is presented in Section 3.5. First the MOLHC algorithm is shown to be significantly faster than the naive exhaustive search, which is too slow to be used in practice (Section 3.5.1). Then datasets from the UCI repository (Bache & Lichman, 2013; "Weka: Collections of Datasets," 2016) are described (Section 3.5.2) and used to show that MOLHC outperforms the baseline algorithms (Section 3.5.3) and the well-known multi-objective optimisation algorithm NSGA-II (Section 3.5.4). A case study of MOLHC application in the activity-recognition domain is presented in Section 3.6. Limitations of the MOLHC algorithm are discussed in Section 3.7 and conclusion is given in Section 3.8.

3.2 Related Work

This chapter is related to research from two areas: increasing the comprehensibility of models constructed with machine learning algorithms that apply either single- or multi-objective approach; and constructing hybrid models that combine multiple types of classification models.

Certain established machine learning algorithms already use criteria other than accuracy in the learning process. The RIPPER rule induction algorithm (Cohen, 1995), for example, achieves the compactness of the learned rule set by using the minimum description length principle (Rissanen, 1978). Girosi, Jones, and Poggio (1995) presents an algorithm that includes comprehensibility criteria in the regularization of artificial neural networks. Jin (2000) and Gorzałczany and Rudziński (2012) apply the same idea to construct fuzzy rules with evolutionary optimization methods. The common property of these algorithms is that they combine the accuracy and comprehensibility into a single objective function using a weighted-formula approach – resulting in a single classifier, which was shown to be inferior to the multi-objective approach (Freitas, 2004). In contrast, the MOLHC algorithm uses Pareto-based multi-objective approach to generate a set of classifiers ranging from the most comprehensible to the most accurate. The information provided by the set of suggested classifiers offers additional information enabling the user to take a well informed decision when selecting the final classifier. On the other hand, weighted-formula approach requires the user to search for the appropriate weights defining his preferences iteratively and without any information, which is a considerable drawback.

The work of Kottathra and Attikiouzel (1996), who formulated the training of an artificial neural network as a bi-objective minimization problem with the mean square error and the number of neurons as the conflicting objectives, is one of the earliest examples of multi-objective machine learning. They solved the problem with a branch-and-bound algorithm, while later work mostly uses evolutionary optimization methods. For example Ishibuchi, Nakashima, and Murata (2001) and Pulkkinen (2009) constructed fuzzy rules; Jin, Sendhoff, and Körner (2006) constructed artificial neural networks; Markowska-Kaczmar and Mularczyk (2006) extracted rules from neural networks; Tušar (2007) constructed classification trees; and Clark and Everson (2012) constructed relevance vector machines. The goal of the algorithms mentioned above is to extract a comprehensible classifier from an accurate but incomprehensible classifier. On the other hand, the MOLHC algorithm starts from a comprehensible classifier and improves its accuracy by introducing black-box hybrid leaves. The advantage is that the algorithm can start from an initial comprehensible classifier that is easy to validate and can be constructed either manually by a domain expert or automatically by an existing machine-learning algorithm. In addition, the MOLHC algorithm is guaranteed to find the complete Pareto-optimal set, which is an improvement

over the commonly used stochastic optimization algorithms, e.g., NSGA-II (Deb, Pratap, Agarwal, & Meyarivan, 2002).

Probably the best-known example of hybrid machine-learning models is model tree, which uses linear regression functions in tree leaves (Quinlan, 1992). The resulting models are accurate and compact. In a similar way the NBTrees place Naive Bayes classifiers in the leaves of classification tree (Kohavi, 1996), which retains the comprehensibility of both model types and often improves the accuracy. The VFDTc algorithm for mining data streams similarly combines Naive Bayes with Hoeffding trees, improving the accuracy compared to regular Hoeffding trees (Gama, Rocha, & Medas, 2003). Hybrid SVM-based classification trees (Arun Kumar & Gopal, 2010) use a SVM classifier to classify examples close to the class boundary, and a classification tree for the examples farther away. This results in a significant speed-up over SVM alone, without any compromise in accuracy. The MOLHC algorithm is similar to the ones listed above in that it combines a classification-tree model with another model, however its purpose is not only to improve the classification accuracy of models or to speed up the learning. The goal of replacing some regular leaves in the initial classification tree with hybrid leaves invoking a black-box classifier (achieved by the MOLHC algorithm) is to increase the classification accuracy in the parts of the domain that do not allow accurate classification using a comprehensible model. Hence the resulting hybrid trees mimic a human expert's knowledge: with some comprehensible parts (represented by regular leaves), and some incomprehensible. Such hybrid trees are ideally suited to searching for trade-offs between the accuracy and comprehensibility, because the comprehensibility can be dialled to anywhere between none and complete.

Our research also introduces some further distinctions from the related work. First, most papers discussing the comprehensibility of a classification tree define it as the complexity of the tree (Pazzani, 2000), which is not ideal (as shown in Chapter 2) and is not applicable to hybrid trees. Therefore, we define the comprehensibility as the fraction of examples classified by the comprehensible leaves of the hybrid tree. We also propose an interactive learning process where a human expert in each iteration analyses the set of learned hybrid trees and based on the obtained insights selects the inputs for the next run of the MOLHC algorithm. The proposed process can for instance replace the build model and assess model tasks in the modelling phase of the Cross-Industry Standard Process for Data Mining also known as CRISP-DM (Chapman et al., 2000) – a widely used data-mining process model encompassing the entire life cycle of a data mining project. The analysis of the learned hybrid trees may also provide valuable conclusions about how to perform the tasks such as clean data, construct data, and integrate data in the CRISP-DM data preparation phase. To the best of our knowledge there is no standard data-mining process for multi-objective classification. The MOLHC algorithm can be simply used in most data-mining process models (for instance the ones compared in the survey by Kurgan and Petr (2006)) as well.

3.3 The Suggested Knowledge Representation and Approach

This section defines the notion of a hybrid tree classification model, its comprehensibility and accuracy used in this study. The Pareto dominance relation, used to compare the quality of hybrid trees, and the non-dominated set (i.e. Pareto set), which is the output of the MOLHC algorithm, are defined for the case of hybrid trees.

A hybrid tree is a combination of two classifiers, which are given as an input to the MOLHC algorithm. The first input is a classification tree, which is used as a comprehensible classifier and as a partitioning scheme: examples belonging to a single leaf are treated as a single part of the domain. The second input is a black-box classifier, which is used as

a more accurate, but incomprehensible classifier. The basic idea of the MOLHC algorithm is to replace some leaves of the given comprehensible classification tree with black-box leaves that invoke the given black-box classifier, in order to obtain hybrid trees with better classification accuracy. The third input to the MOLHC algorithm is a learning set of data instances. It is used to compare the classification accuracy of the initial classification tree and the black-box classifier in each part of the domain and to decide which parts of the domain should be classified with each of the two classifiers.

Definition 3.1 (Hybrid tree). Given an initial classification tree and a black-box classifier, we can represent a hybrid tree T_i more compactly with a Boolean vector with n components $(t_{i,1}, t_{i,2}, \ldots, t_{i,n}) \in \{0,1\}^n$, where n is the number of leaves in the initial classification tree in which the black-box classifier achieves higher accuracy than the initial classification tree. The values 0 and 1 for each component of the vector denote the type of corresponding leaf: 0 for regular (i.e. as in the initial tree) and 1 for black-box leaves.

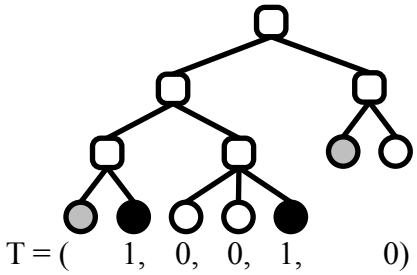


Figure 3.2: An example of a hybrid tree $T = (1, 0, 0, 1, 0)$, where black circles represent black-box leaves, white ones represent regular leaves, and grey ones represent leaves that will always remain regular because their accuracy is at least as good as the accuracy of the black-box classifier.

The initial classification tree is represented as $T_1 = (0, 0, \ldots, 0)$, while the tree with the maximal number of black-box leaves is represented as $T_{2^n} = (1, 1, \ldots, 1)$. The number of ones (zeroes) in a vector corresponds to the number of black-box (regular) leaves. For example, the tree $(1, 0, 0, 1, 0)$ shown in Figure 3.2 is based on the initial tree with seven leaves and has two black-box and three regular leaves plus the two regular leaves that are omitted from the vector representation, because they achieve higher accuracy than the black-box classifier and should therefore not be considered for replacing with black-box leaves.

The comprehensibility of hybrid trees does not depend only on the structure of the tree but also on the number of the examples belonging to the black-box leaves. The MOLHC algorithm considers a classification tree as perfectly comprehensible regardless of its structure and size; however it is only sensible to use this assumption and the MOLHC algorithm with reasonably sized classification trees. We treat trees with more than $\sim$50 leaves as (partially) incomprehensible, because the majority of leaves in such trees (with branching factor 2) have depth greater than 5, while the number of objects an average human can hold in working memory is 7 ± 2 (Miller, 1956). An average person can therefore efficiently think about instances belonging to leaves with depth 5 or less, because each attribute-value pair defining the path from the root to a leaf is an object the user needs to remember when thinking about instances that belong to the leaf.

Definition 3.2 (Comprehensibility of a hybrid tree). The comprehensibility $c_i \in [0, 1]$ of a hybrid tree T_i is defined by Eq. (3.1) as the ratio between the number of examples

N_j classified by regular leaves R_i and the number of all examples N used to evaluate the comprehensibility of the hybrid tree.

$$c_i = (\sum_{j \in R_i} N_j)/N \in [0,1] \tag{3.1}$$

The comprehensibility of a hybrid tree is therefore equal to the probability of classifying an instance with a regular (comprehensible) leaf. By definition, the comprehensibility of the initial classification tree T_1 is 1 and the comprehensibility of the black-box classifier is 0, meaning that it is not comprehensible at all. The comprehensibility of all other hybrid trees is between 0 and 1.

The comprehensibility metric for hybrid trees is based on three assumptions. The first assumption is that the reasonably sized trees are completely comprehensible. This is supported by the results presented in Section 2.4.3, which show that even though there are differences in comprehensibility of trees, they are very limited for the trees with modest size. The complexity of the analysed trees (all with depth ≤ 7) does not significantly influence the answer-correctness (which is close to 100%). The maximal question-difficulty (even for the leaves with depth 7) is below 3, which means that the average respondent answered all the questions *quite quickly and without major problems*. Furthermore, the average comprehensibility ratings of all of the trees in the survey are below *difficult to comprehend*. Therefore, we accept the first assumption as a reasonable simplification. The second assumption is that a black-box classifier is completely incomprehensible, which is true by the definition. The third assumption is that the comprehensibility of a tree can be computed as a sum of terms over its leaves. This is confirmed by almost perfect correlation between the rated tree comprehensibility and the respondents performance computed as the sum of time-to-answer or question-difficulty over the leaves. Furthermore, the tree comprehensibility metric c_{B} that was evaluated as the best is also computed as a sum over the leaves in a tree (see Section 2.4.5). Therefore we accept the third assumption as valid.

Definition 3.3 (Accuracy of a hybrid tree). The classification accuracy a_i of a hybrid tree T_i is defined by Eq. (3.2) as the ratio between the number of correctly classified examples and the number of all examples N used to evaluate the accuracy of the hybrid tree. Note that $a_i \in [0,1]$. The number of correctly classified examples is summed over all the leaves in the hybrid tree partitioned into the set of regular leaves R_i and black-box leaves B_i: $N_{j,\mathrm{t}}$ denotes the number of correctly classified examples in a regular leaf j, while $N_{j,\mathrm{bb}}$ denotes the number of correctly classified examples in a black-box leaf j.

$$a_i = (\sum_{j \in R_i} N_{j,\mathrm{t}} + \sum_{j \in B_i} N_{j,\mathrm{bb}})/N \in [0,1] \tag{3.2}$$

Multi-objective optimisation uses the Pareto dominance relation ($\preccurlyeq$) to compare the quality of solutions (Deb, 2001) and hence returns a set of solutions with different incomparable vector-values of the objective functions as opposed to single-objective optimisation, which returns the solution(s) with the best value of a single scalar objective function.

Definition 3.4 (Quality of a hybrid tree). The quality of a hybrid tree T_i is given by the vector, $(a_i, c_i) \in [0,1] \times [0,1]$, where the comprehensibility c_i and accuracy a_i of the hybrid tree T_i are defined with Eq. (3.1) and Eq. (3.2), respectively.

The two objectives (classifier accuracy and comprehensibility) are conflicting. Therefore, the MOLHC algorithm uses a Pareto-based multi-objective approach that returns a set of non-dominated classifiers (defined by Eq. (3.4)). They provide the optimal trade-off between accuracy (Eq. (3.2)) and comprehensibility (Eq. (3.1)).

Definition 3.5 (Dominance relation on hybrid trees). Eq. (3.3) defines the relation $x \preccurlyeq y$ (hybrid tree x dominates hybrid tree y), where the goal is to maximize the accuracy and comprehensibility: hybrid tree x dominates hybrid tree y if it is better than y in at least one objective and no worse in the other.

$$x \preccurlyeq y \Leftrightarrow (a_x > a_y \wedge c_x \geq c_y) \vee (a_x \geq a_y \wedge c_x > c_y) \tag{3.3}$$

Note that the symbol used to denote the dominance relation ($\preccurlyeq$) is consistent with multi-objective optimisation theory, which treats all problems as minimisation problems (e.g., Deb (2001)), whereas we want to maximize the accuracy and comprehensibility. While our use of the symbol might seem a bit counter intuitive – e.g., the hybrid tree x in Eq. (3.3) is better than the hybrid tree y and not vice versa – we keep this notation for consistency.

Definition 3.6 (Non-dominated hybrid tree). A hybrid tree x is said to be non-dominated (i.e. optimal) in a set of hybrid trees S if no other hybrid tree that dominates x exist in S as stated by Eq. (3.4).

$$x \in \text{non-dom}(S) \Leftrightarrow x \in S, \nexists y \in S : y \preccurlyeq x \tag{3.4}$$

A set of hybrid trees $P' = \text{non-dom}(P) \subseteq P$ is the non-dominated subset of the set of hybrid trees P if it contains exactly the hybrid trees that are not dominated by any member of the set P.

Definition 3.7 (Pareto set). The non-dominated set of the entire solution space is called the Pareto set or globally Pareto-optimal set (Deb, 2001).

In our case, the Pareto set is the set of non-dominated (i.e. optimal) hybrid trees according to their comprehensibility and accuracy in the set of all hybrid trees that can be generated from a given initial tree and black-box classifier. This should be the output of the MOLHC algorithm.

3.4 Multi-Objective Learning of Hybrid Classifiers

3.4.1 Preliminaries

The proposed algorithm for multi-objective learning of hybrid classifiers (MOLHC) takes an initial comprehensible tree and replaces some of its leaves with a black-box classifier in order to produce hybrid trees with higher classification accuracy (and lower comprehensibility) compared to the initial classification tree. The goal of the MOLHC algorithm is to find the Pareto set of hybrid trees (Definition 3.7), from which the user selects one or more hybrid trees. Considering hybrid trees that are not in the Pareto set is not required, because it is guaranteed that all of them are worse in both accuracy and comprehensibility (at the same time) than at least one hybrid tree in the Pareto set. Analysing the set of non-dominated hybrid trees enables the user to extract additional knowledge about the domain and the classification task, which facilitates a well informed decision when choosing a single classifier from the Pareto set.

Any algorithm that learns hybrid trees performs search in the space of (compactly represented) hybrid trees, i.e. the Cartesian product $\{0,1\} \times \{0,1\} \times \cdots \times \{0,1\} = \{0,1\}^n$ containing 2^n hybrid trees. The result of a learning algorithm is therefore a set of Boolean vectors corresponding to a set of hybrid trees, and a set of vectors with two components representing the accuracy and comprehensibility of the hybrid trees.

A naive algorithm that simply calculates the quality of all 2^n possible hybrid trees using Eq. (3.1) and Eq. (3.2) and selects the non-dominated ones faces combinatorial explosion of the search space. Therefore it can be applied only on small initial classification trees (up to ∼20 leaves). In order to enable using considerably larger initial classification trees, the MOLHC algorithm calculates the qualities of hybrid trees using a dynamic programming approach, which has a constant time complexity instead of the $O(n)$ complexity of the naive algorithm (see Eq. (3.1) and Eq. (3.2)). Furthermore, MOLHC avoids generating hybrid trees that can be recognized as dominated even before they are actually generated or their quality is computed. This approach limits the search space, i.e. the number of hybrid trees that need to be generated and evaluated. In other words, MOLHC performs an exhaustive search over a pruned space of all possible combinations of the initial classification tree and the black-box classifier. In contrast with stochastic evolutionary optimization algorithms, which are most often used in multi-objective learning, MOLHC is an exact algorithm. Its output is the complete Pareto set of hybrid trees and not just an approximation of the Pareto set. Nevertheless, the MOLHC algorithm has shorter run-times than for example the well-known multi-objective optimisation algorithm NSGA-II (Deb et al., 2002) when applied to an initial classification trees with modest size – a prerequisite for comprehensibility.

3.4.2 The Algorithm

The MOLHC algorithm (pseudo code given in Algorithm 3.1) starts by identifying the sub-sets of learning examples that belong to each leaf of the initial tree (lines 1–6). The number of correctly classified instances (from the provided learning set), which belong to each leaf l, is computed for the initial tree ($N_{j,\mathrm{t}}$ in line 7) and for the black-box classifier ($N_{j,\mathrm{bb}}$ in line 8). These values are used to compute the relative difference in accuracy $\delta_{l,\mathrm{a}}$ (Eq. (3.5)) and comprehensibility $\delta_{l,\mathrm{c}}$ (Eq. (3.6)) in lines 10–17.

Definition 3.8 (Relative difference in accuracy and comprehensibility). The relative difference in accuracy $\delta_{l,\mathrm{a}}$ and comprehensibility $\delta_{l,\mathrm{c}}$ between a hybrid tree and its variant obtained by replacing the leaf l with a black-box leaf are defined by Eq. (3.5) and Eq. (3.6), respectively.

$$\delta_{l,\mathrm{a}} = \left(N_{l,\mathrm{bb}} - N_{l,\mathrm{t}}\right)/N \tag{3.5}$$

$$\delta_{l,\mathrm{c}} = -N_l/N < 0 \tag{3.6}$$

The relative differences $\delta_{l,\mathrm{a}}$ and $\delta_{l,\mathrm{c}}$ depend only on the black-box leaf l and not on the type of other leaves as proven in Section D.1, Lemma D.2 and Lemma D.3. The relative difference in comprehensibility $\delta_{l,\mathrm{c}}$ is always negative because replacing a regular leaf with a black-box leaf decreases the comprehensibility. On the other hand, the relative difference in accuracy $\delta_{l,\mathrm{a}}$ can be positive, zero or negative. However, only the leaves l with $\delta_{l,\mathrm{a}} > 0$ are considered for replacement with black-box leaves. The other leaves are ignored, because replacing a regular leaf with a black-box leaf that does not increase the accuracy does not make sense – such a replacement decreases the comprehensibility, which results in a dominated tree.

Definition 3.9 (Dominance relation on leaves). Eq. (3.7) defines the relation leaf l dominates leaf k according to the two objectives: relative difference in accuracy $\delta_{l,\mathrm{a}}$ and comprehensibility $\delta_{l,\mathrm{c}}$ where both objectives need to be maximized.

$$l \preccurlyeq k \Leftrightarrow (\delta_{l,\mathrm{a}} > \delta_{k,\mathrm{a}} \wedge \delta_{l,\mathrm{c}} \geq \delta_{k,\mathrm{c}}) \vee (\delta_{l,\mathrm{a}} \geq \delta_{k,\mathrm{a}} \wedge \delta_{l,\mathrm{c}} > \delta_{k,\mathrm{c}}) \tag{3.7}$$

Eq. (3.7) states that the better leaves increase accuracy more and at the same time decrease comprehensibility less than the worse leaves. The quality of leaves therefore depends on the relative difference in accuracy and comprehensibility and is compared using the Pareto dominance relation.

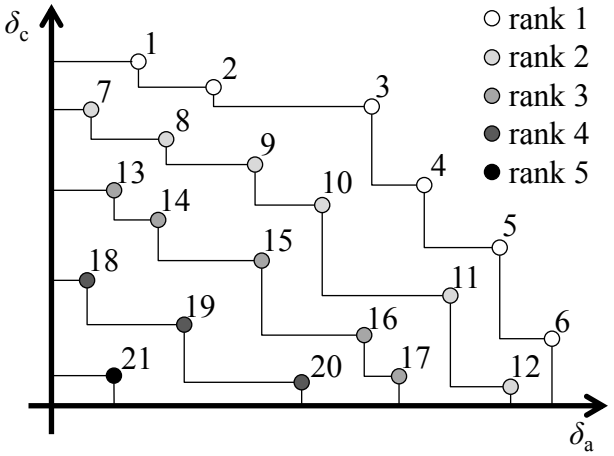


Figure 3.3: An example of Pareto sorting of leaves (rank 1–5) according to the maximal relative difference in accuracy δ_{a} and comprehensibility δ_{c}. The numbers represent indices assigned to the leaves.

The second step of the MOLHC algorithm performs Pareto sorting of leaves, i.e. the leaves are classified into various non-domination levels (line 18, Figure 3.3). This partitions the set of leaves represented in the objective space so that the leaves with non-domination levels i are non-dominated by the leaves with levels $j \leq i$, and so that they dominate all the leaves with levels $k > i$ (Deb, 2001). The leaves with the same non-domination level are sorted according to increasing relative difference in accuracy $\delta_{l,\mathrm{a}}$ (the same as sorting according to the decreasing relative difference in comprehensibility $\delta_{l,\mathrm{c}}$) and assigned ascending integer indices according to the described sorting (line 18, Figure 3.3). In addition, the set of the leaves D_l that are directly dominated by a leaf l (Eq. (3.8)) is calculated for each leaf l (the for loop in lines 19–21). This concludes the initialisation phase of the algorithm.

Definition 3.10 (Directly dominated leaves). The set of leaves D_l that are directly dominated by the leaf l is the set of non-dominated leaves in the set of leaves that are dominated by the leaf l as defined by Eq. (3.8) and shown by an example in Figure 3.4.

$$D_l = \textit{non-dom}(k : l \preccurlyeq k) \tag{3.8}$$

The third step of the MOLHC algorithm uses an iterative search for the hybrid trees (lines 22–43), which proceeds in the order of increasing number of replaced (i.e. black-box) leaves. The MOLHC algorithm starts with the initial classification tree $T_1 = (0, 0, \ldots, 0)$, i.e. the only hybrid tree with 0 black-box leaves (lines 22–28). The accuracy a_1 of the initial hybrid tree T_1 is computed as the sum over all the leaves using Eq. (3.2) (line 25), whereas its comprehensibility c_1 is 1 by Eq. (3.1) (line 26). In each iteration, the algorithm uses the set of hybrid trees *tmp* generated in the previous iteration to generate the set of hybrid trees *tmp*′ with one more black-box leaf: from a hybrid tree $T_i \in$ *tmp* it

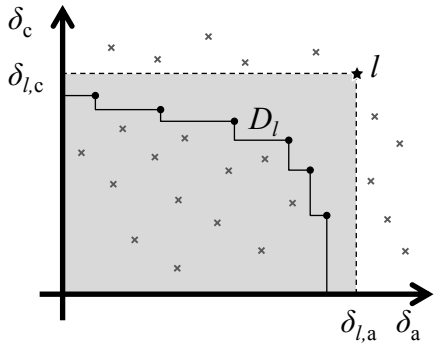


Figure 3.4: The set D_l (circles) of leaves that are directly dominated by the leaf l (star) is the set of non-dominated leaves in the set of leaves that are dominated by the leaf l (gray background).

generates a set of hybrid trees by replacing one of the regular leaves in T_i with a black-box leaf at a time (lines 32–39). MOLHC does calculate the comprehensibility and accuracy of a hybrid tree as the sum over all the leaves (Eq. (3.1) and Eq. (3.2)), because this would require $2 \cdot 2^n(n-1)$ additions. The accuracy a_i and the comprehensibility c_i of a hybrid tree $T_i = (t_{i,1}, t_{i,2}, \ldots, t_{i,n})$ are used instead to compute the accuracy a_j and the comprehensibility c_j of a hybrid tree $T_j = (t_{j,1}, t_{j,2}, \ldots, t_{j,n})$ where $t_{j,l} = 1$ and $t_{j,k} = t_{i,k}$ for each leaf $k \neq l$. The trees T_i and T_j differ only in the fact that leaf l is a regular leaf in T_i while it is a black-box leaf in T_j (line 33). This requires a single addition operation for each objective according to the Eq. (3.9) (lines 35–36). Therefore the number of required additions is reduced from $2 \cdot 2^n(n-1)$ to $2 \cdot ((2^n - 1) + (n-1))$.

$$(a_j, c_j) = (a_i + \delta_{l,a}, c_i + \delta_{l,c}) \tag{3.9}$$

The algorithm does not replace each regular leaf in T_i; it replaces only the non-dominated regular leaves (named L_i) with indices greater than the highest index of a black-box leaf in T_i (line 32). This constraint regarding the indices of leaves prevents generating a hybrid tree more than once — the first optimisation. However, leaf indices must be assigned according to the Pareto sorting (as shown in Figure 3.3) in order to prevent conflicts with the second optimisation of not replacing the dominated leaves. The second optimisation avoids generating most of the dominated hybrid trees (i.e. narrows the search space), which is the most important improvement over the naive algorithm. Nevertheless, the algorithm is still guaranteed to find the complete Pareto set of hybrid trees as proven in Section D.1. The iteration loop stops when all of the leaves are replaced with black-box leaves.

Besides the accuracy and comprehensibility of a hybrid tree T_j, the algorithm also needs to maintain the set of non-dominated leaves L_j for each generated hybrid tree. The set of non-dominated leaves for the initial classification tree T_1 is the set of non-dominated leaves among all the leaves (except the ones that have negative or 0 relative difference in accuracy), i.e. the set of leaves with Pareto dominance rank 1 (line 27). The set of non-dominated regular leaves L_j in the tree T_j generated from the hybrid tree T_i is calculated using Eq. (3.10), which is proven in Section D.3. L_j therefore contains all the non-dominated leaves in the tree T_i (denoted as L_i) except the leaf l (replaced when generating T_j from T_i) and a subset of leaves from D_l (the leaves that are directly dominated by the leaf l) that are not dominated by any leaf from $L_i \setminus \{l\}$. In other words, the leaves that should be replaced in the newly generated hybrid tree T_j are the ones that were considered for replacing in the original tree T_i (i.e. the leaves in the set L_i) except the replaced leaf l

plus leaves that are dominated by the leaf l but are not dominated by the other leaves in L_i. Eq. (3.10) enables efficient computation of the non-dominated set of regular leaves L_i by comparing only the qualities of some leaves and not of all the remaining regular leaves, which would be required by a naive approach. Theorem D.4 given in Section D.3 enables even more efficient computation of L_j from L_i and D_l if the two sets are ordered.

$$L_j = \textit{non-dom}(L_i \setminus l \cup D_l) = L_i \setminus \{l\} \cup \{k : k \in D_l \wedge \nexists m \in L_i \setminus \{l\} : m \preccurlyeq k\} \tag{3.10}$$

In the final step (line 44), the algorithm selects the non-dominated hybrid trees from the set of all generated hybrid trees. As stated by Theorem D.2 in Section D.2, the output of the MOLHC algorithm is the Pareto set of hybrid trees and their corresponding objective values (accuracy and comprehensibility of the hybrid trees), which is used to plot the Pareto front. The user analyses the Pareto front as discussed in Section 3.7.1 and Piltaver, Luštrek, and Gams (2014) and finally selects the most appropriate hybrid tree.

In order to make the MOLHC algorithm implementation efficient, the following subproblems must be solved using appropriate algorithms. First, the Pareto dominance between a set of leaves or a set of hybrid trees is computed using the Two-dimensional Skyline Operator algorithm (Börzsönyi, Kossmann, & Stocker, 2001), which decreases the time complexity $O(n^2)$ of a naive implementation (Deb, 2001) to $O(n \log n)$ – the lowest possible time complexity (Kung, Luccio, & Preparata, 1975). Second, the Pareto sorting of leaves is performed with the non-dominated sorting algorithm proposed by Deb et al. (2002), which has time complexity $O(n^2)$. Using more elaborate algorithms such as the Better Non-dominated Sorting Algorithm (Shi, Chen, & Shi, 2005) theoretically provides time complexity improvement for a constant factor, but is not important for the MOLHC algorithm, which deals with only two objectives and sorts a set containing only a few elements only once. Third, the algorithm that calculates the set of directly dominated leaves for each leaf in the initial tree should take advantage of the Pareto sorting of the leaves. Only the leaves with Pareto dominance rank higher than the rank of a considered leaf l can be dominated by it. Furthermore, the leaves with the same Pareto dominance rank are sorted according to the relative difference in accuracy δ_{a}, therefore only one objective needs to be compared in order to determine that a leaf with a higher Pareto dominance rank is dominated by the leaf l. Finally, the sorting also enables calculating the set of non-dominated leaves in each generated hybrid tree using the dynamic programming algorithm described in Section D.3 and supported by Theorem D.4.

3.4.3 Data-Mining Process for Multi-Objective Learning

The MOLHC algorithm is used within the data-mining process described in this subsection and presented in Figure 3.1. The data-mining process is iterative and includes a domain expert in the loop in order to combine the expert's knowledge and the power of machine learning. Each iteration starts by choosing an initial classification tree and a black-box classifier as the inputs to the MOLHC algorithm. Iteratively improving the inputs to the MOLHC algorithm improves the quality of the produced hybrid trees. The classification tree can be built according to the expert's knowledge or trained using a machine-learning algorithm. In the latter case, an additional training set is used, i.e. not the one used as an input to the MOLHC algorithm. The classification tree must be comprehensible to the expert and as accurate as possible — at least in some of its leaves. An existing accurate classifier can be used as the black-box classifier or it can be trained on the same dataset as the classification tree.

Second, the MOLHC algorithm is executed and the resulting Pareto set of hybrid trees is analysed (Piltaver, Luštrek, & Gams, 2014). If the expert is satisfied with the

accuracy and comprehensibility of some of the hybrid trees, the data mining process is finished. Otherwise, the next iteration of the process begins by selecting new inputs for the MOLHC algorithm. If the difference in accuracy between the most accurate hybrid tree and the initial classification tree is too small, a more accurate black box classifier needs to be learned. The insights obtained by analysing the hybrid trees often enable training a more accurate black-box classifier. In order to do so, new attributes or additional learning instances with less noise are added if parts of the domain cannot be classified accurately using the existing set of attributes, i.e. when both the black-box classifier and the classification tree achieve low accuracy in some leaves of the tree. In addition, an alternative learning algorithm can be used or the parameters of the same learning algorithm can be tuned. If a more accurate black-box classifier cannot be learned, the process is stopped. In this case, the initial classification tree (or a very comprehensible hybrid tree) is chosen as the final classifier.

On the other hand, the expert might not be satisfied with the hybrid trees because of their low comprehensibility. This problem is solved by training a more comprehensible classification tree as an input for the next execution of the MOLHC algorithm. The analysis of the Pareto set of hybrid trees determines which subtrees should be improved. To do so, the relative quality of each leaf in the initial tree is first computed based on the number of hybrid trees from the Pareto set in which the leaf is replaced with the black-box classifier (Piltaver, Luštrek, & Gams, 2014). The leaves that are never replaced with the black-box classifier (e.g., leaf 8 in Figure 3.7) and leaves with high relative quality, i.e. the ones that are infrequently replaced with black-box leaves (e.g., three or more black stars in Figure 3.7), do not need improvement. The leaves with low relative quality (e.g., one or fewer black stars in Figure 3.7), on the other hand, are the ones that should be improved first. Failing to improve the accuracy by replacing such a leaf with a shallow subtree shows that the corresponding part of the domain cannot be classified accurately with a classification tree and therefore a black-box classifier should be used instead or additional attributes need to be added to the training dataset. Note that the relative quality of leaves is complementary information to the classification accuracy in the leaves (Piltaver, Luštrek, & Gams, 2014). If low comprehensibility is caused by poor comprehensibility of attributes in the dataset set, it can be improved by replacing (some of) the attributes with more comprehensible ones. The discussed data-mining process is further explained on a use case in Section 3.6.

Algorithm 3.1: Multi-objective learning of hybrid classiffers (MOLHC)

input : learning data-set learningSet, initial classification tree initialTree, and black-box classifier blackBox
output: set of non-dominated hybrid trees hybridTrees

```
1  N ← 0;
2  foreach leaf l ∈ initialTree do N_l ← 0;
3  foreach instance i ∈ learningSet do
4  |  N ← N + 1;                                  // number of training instances
5  |  l ← initialTree.findCorrespondingLeaf(i);
6  |  N_l ← N_l + 1;                  // number of training instances in leaf l
7  |  if initialTree.classify(i) = i.class() then N_{l,t} ← N_{l,t} + 1;
8  |  if blackBox.classify(i) = i.class() then N_{l,bb} ← N_{l,bb} + 1;
9  end
10 leaves ← {};      // leaves considered for replacing with black-box leaves
11 foreach leaf l ∈ initialTree do
12 |  if N_{l,bb} > N_{l,t} then
13 |  |  leaves ← leaves ∪ {l};
14 |  |  δ_{l,c} ← −N_l/N;
15 |  |  δ_{l,a} ← (N_{l,bb} − N_{l,t})/N;
16 |  end
17 end
18 leaves = sortByAccuracy(paretoSort(leaves));
19 foreach l ∈ leaves do
20 |  D_l ← directlyDominated(l, leaves);
21 end
22 hybridTrees ← {};                          // set of processed hybrid trees
23 T_1 ← (0, 0, ..., 0);                    // initialTree has no black-box leaves
24 m_1 ← −1;                    // max index of black-box leaf in hybrid tree
25 a_1 ← Σ_l N_{l,t};                                // accuracy of the initialTree
26 c_1 = 1;                                // comprehensibility of the initialTree
27 L_1 = non-dom(leaves);              // non-dominated leaves in the initialTree
28 tmp = {T_1};                              // set of unprocessed hybrid trees
29 while tmp ≠ {} do
30 |  tmp' ←= {};
31 |  foreach T_i ∈ tmp do
32 |  |  foreach l ∈ L_i : l > m_i do
33 |  |  |  T_j ← T_i; t_{j,l} ← 1;   // replace leaf l with black-box to obtain T_j
34 |  |  |  m_j ← l;
35 |  |  |  a_j ← a_i + δ_{l,a};
36 |  |  |  c_j ← c_i + δ_{l,c};
37 |  |  |  L_j ← non-dom(L_i \ {l} ∪ D_l);        // non-dominated leaves in L_j
38 |  |  |  tmp' ← tmp' ∪ {T_j};
39 |  |  end
40 |  |  hybridTrees ← hybridTrees ∪ {T_i};
41 |  end
42 |  tmp ← tmp';
43 end
44 hybridTrees ← non-dom(hybridTrees);
45 return hybridTrees;
```

3.5 Algorithm Evaluation

In order to evaluate the suggested MOLHC algorithm three sets of experiments described in the following subsections were performed. First, the run-times of the naive and the MOLHC algorithm are compared in order to evaluate the benefit of the efficient algorithm and assess if the algorithms are fast enough to be used in practice. Second, the Pareto front of hybrid trees is compared with the baseline classifiers on 21 domains in order to find out whether the hybrid trees provide trade-offs between accuracy and comprehensibility that are considerably different from the extremes provided by the traditional learning algorithms, and are therefore useful in practical applications. Third, the run-time of the efficient algorithm is compared with the run-time of a well-known evolutionary multi-objective optimisation algorithm NSGA-II (Deb et al., 2002). Furthermore, the convergence of NSGA-II results towards the Pareto front is analysed for reasonable run-times.

3.5.1 Run-Time

The naive algorithm (i.e. exhaustive search) and the MOLHC algorithm have exponential time complexity $O(2^n)$ in the worst case, where n is the number of leaves in the initial classification tree considered for replacing with black-box leaves – each of the n leaves can be replaced with the black-box or not. Since the problem is obviously at least as difficult as the NP-hard 0/1 knapsack problem, no algorithm that solves the problem in polynomial time is expected to exist.

The exhaustive search generates and evaluates all of the 2^n possible hybrid trees, whereas the MOLHC algorithm avoids generating some of the hybrid trees and is therefore faster. The MOLHC algorithm therefore achieves a considerably lower run-time as shown in Figure 3.5. Each measurements in Figure 3.5 represents an average run-time over 100 randomly generated trees with relative differences in accuracy and comprehensibility of leaves similar to the typical initial trees (10 trees for run-times over 5 min were used). The run-times are approximately 2^{2n} for the exhaustive search algorithm and $2^{0.4n}$ for the MOLHC algorithm (Piltaver, Luštrek, Zupančič, et al., 2014). The average run-time of the exhaustive search algorithm implemented in Java™ for a tree with 18 leaves is around 18 minutes (3.2 min for 17 leaves), while it is only 0.8 ms for the MOLHC algorithm (< 0.5 s for 40 leaves) on a 3 GHz Intel® Core™ 2 Duo computer. This shows that exhaustive search is not usable for initial trees with 20 or more leaves.

Since the goal is to find classifiers comprehensible to humans, only reasonably small initial trees are expected as an input to the algorithm. Experiments show that the efficient implementation of the algorithm finds the complete Pareto set of hybrid trees within seconds, which is acceptable for use in the interactive data mining process described in Section 3.4.3 and illustrated in Figure 3.1.

3.5.2 Testing Datasets

The evaluation of the MOLHC algorithm is based on 21 datasets from the UCI repository (Bache & Lichman, 2013). Evaluation datasets were selected from the set of 94 classification datasets available in ARFF format at the Weka web page ("Weka: Collections of Datasets," 2016). Only datasets with more than 300 instances (49 out of 94) were considered because they had to be split into three parts with at least 100 instances: the first part was used to train the initial tree and the black-box classifier, the second part was used as input to the MOLHC algorithm to find the Pareto set of hybrid trees and predict their accuracy and comprehensibility, and the third part was used to evaluate the set of hybrid trees, initial classification tree and the black-box classifier. The set of selected

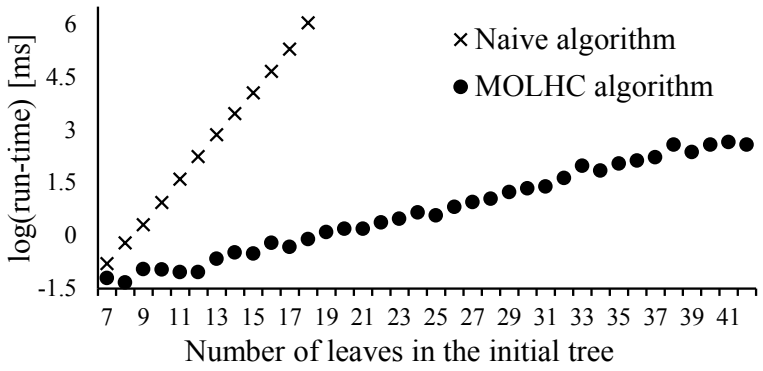


Figure 3.5: Run-times of the MOLHC (circles) and naive algorithm (crosses).

datasets was finally reduced to the datasets in which a black-box classifier achieves at least 10% better classification accuracy than the tree with approximately 20 leaves (21 out of 49 datasets). There is no point in learning hybrid trees if the difference in classification accuracy between the initial tree and the most accurate (black-box) classifier is small. The following algorithms were used to build the black-box classifiers: SVM (Platt, 1999), k-NN (Aha et al., 1991), ANN (Rumelhart et al., 1986), logistic regression (Sumner, Frank, & Hall, 2005), and Naive Bayes (John & Langley, 1995), all implemented in Weka (Hall et al., 2009) and used with the default algorithm parameters. Among them, the classifier with the highest classification accuracy computed using 10-fold cross-validation was chosen as the black-box classifier for each dataset. Finally, 38 experiments with the MOLHC algorithm were conducted: one using a small initial tree (~ 20 leaves) for each of the 21 datasets and another using a bigger initial tree (~ 40 leaves) for the 17 datasets that allowed building a bigger tree. The initial classification trees were learned using the J48 algorithm, which is the Weka implementation of the C4.5 algorithm (Quinlan, 1993). Pruning parameters were set in order to obtain trees with approximately 20 and 40 leaves.

3.5.3 Comparison with the Baseline Classifiers

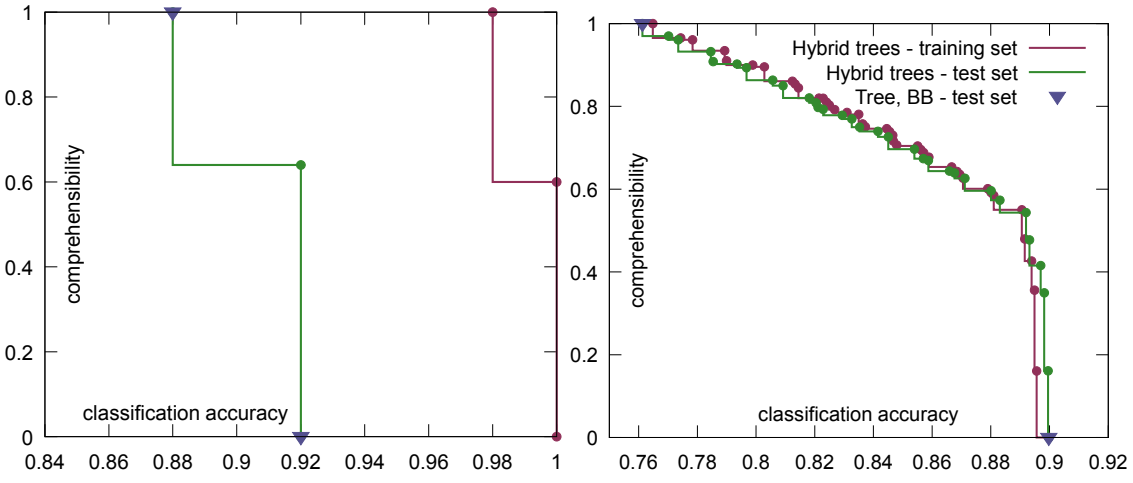
This section compares the Pareto set of hybrid trees with the set of baseline solutions consisting of the initial classification tree and the black-box classifier. In order to evaluate a multi-objective learning algorithm, sets of learned classifiers need to be compared according to multiple objectives as opposed to comparing single classifiers according to a single objective in traditional machine learning. The set of baseline classifiers — used to compare the results of the MOLHC algorithm — includes two classifiers: the initial classification tree as a completely comprehensible classifier and the black-box classifier as an incomprehensible classifier with high classification accuracy.

The hypervolume measure (Zitzler, Thiele, Laumanns, Fonseca, & da Fonseca, 2003) is used to compare the results of MOLHC and the baseline because it is a widely used measure for comparison of multi-objective optimisation algorithms. It measures the volume of the dominated objective space between a reference point and the attainment surface — the envelope marking all the solutions that are sure to be dominated by the set of solutions returned by the algorithm (Deb, 2001). Since only two objectives are considered by the MOLHC algorithm, the hypervolume is in fact the area under the attainment surface represented by a line in Figure 3.6a and 3.6b. The reference point was set to $(-0.2, -0.2)$ following the typically used rule-of-thumb (Beume, Naujoks, & Emmerich, 2007; Knowles, 2006) advocating a space that is a little bit larger than the actual objective

space: in this way the accuracy of a classifier with the comprehensibility 0 contributes to the hypervolume, whereas it would not influence the hypervolume if the reference point was set to $(0, 0)$.

We observe whether the difference in hypervolumes of the baseline and the MOLHC algorithm is small or big. It is small if the hybrid trees produced by the MOLHC algorithm hybrid trees lie close to the attainment surface of the two baseline classifiers. In other words, the Pareto set returned by the MOLHC algorithm is composed of some hybrid trees with high comprehensibility and only slightly higher accuracy than the initial tree, and some hybrid trees with low comprehensibility and approximately the same accuracy as the black-box classifier. This is not a good result, because such hybrid trees are very similar to either the initial classification tree or the black-box classifier. On the other hand, high difference in hypervolume means that hybrid trees produced by the MOLHC algorithm lie further away from the baseline attainment surface, therefore they provide trade-offs between comprehensibility and accuracy that cannot be matched using the traditional machine learning models (i.e. comprehensible trees with relatively low accuracy and incomprehensible black-box classifiers with relatively high classification accuracy).

Figure 3.6 shows the predicted (on the training set) and validated (on the separate test set) results of the MOLHC algorithm and the results of the baseline in the objective space for two datasets. The predicted attainment surface (i.e. Pareto front) obtained by the MOLHC algorithm is shown with a red line, while the validated one is shown in green. The circles represent the accuracy and comprehensibility of the hybrid trees from the Pareto set computed on the training (red) and test dataset (green). Triangles represent the quality of the two baseline classifiers computed on the test dataset.



(a) iris dataset (b) activity-recognition dataset

Figure 3.6: Examples of two Pareto fronts for iris and activity-recognition datasets. Circles represent the accuracy and comprehensibility of the learned hybrid trees, lines represent the Pareto-fronts (green refers to data obtained on the test dataset, red on the training set), triangles refer to the two baseline classifiers.

Table 3.1 shows the data about the 21 datasets used for testing. Table 3.2 shows the comparison of the MOLHC algorithm and the baseline for the 21 datasets. Columns with two numbers delimited by slash sign (in both tables) represent values for a small (on the left) and a big initial classification tree (on the right). The first two columns show the hypervolume for the MOLHC algorithm and the baseline, computed on the test set

consisting of one third of all the instances in a dataset. The column in bold font shows the absolute differences in hypervolume between the MOLHC algorithm and the baseline and the last column shows the relative difference in %. An experiment was also performed using 10-fold cross-validation, obtaining comparable results for the datasets with enough instances and exhibiting high deviations of the results for the small datasets.

Table 3.1: General data about the 21 datasets. Columns with two numbers delimited by slash sign represent values for a small (on the left) and a big classification tree (on the right).

Dataset	Instances	Classes	Attributes	BB accuracy	Leaves	Tree accuracy
anneal	898	6	39	99.67	14/ -	81.7/ -
audiology	226	24	70	100	17/ -	85.5/ -
balance scale	625	3	5	92.82	16/ -	88.0/ -
car	1728	4	7	99.83	11/61	84.4/94.4
cylinder bands	540	2	40	100	10/ -	70.6/ -
dermatology	366	6	35	98.4	14/19	91/96.7
flags	194	8	30	100	6/8	78.5/86.2
heart statlog	270	2	14	87.78	5/9	88.9/96.7
letter	20000	26	17	100	19/40	39.7/57.2
mfeat fourier	2000	10	77	92.05	21/41	82.0/90.7
mfeat karhunen	2000	10	65	99.4	20/41	81.3/91.9
mfeat pixel	2000	10	241	100	19/31	42.7/58.9
mfeat zernike	2000	10	48	89.06	19/46	70.5/83.8
optdigits	5620	10	65	99.84	21/37	80.4/87.9
primary tumor	339	22	18	72.57	11/19	52.2/58.4
soybean	683	19	36	99.56	19/38	64.9/90.4
splice	3190	3	62	97.65	22/41	79.4/84.6
tic-tac-toe	958	2	10	100	15/37	81.3/90
vehicle	846	4	19	99.3	14/40	81.9/97.2
vowel	990	11	14	100	7/56	44.5/87
waveform	5000	3	41	89.14	22/54	84.5/89.4

The best improvement of the MOLHC algorithm over the baseline is 36.6%, the average improvement is 11.17%, and the worst is -1.52%. This shows that the hybrid trees found using the suggested algorithm indeed provide a range of trade-offs between accuracy and comprehensibility that would not be possible using the traditional machine learning algorithms. Therefore hybrid trees provide an important advantage in applications where classifier comprehensibility is important but completely comprehensible classifiers achieve considerably lower accuracy than the incomprehensible classifiers. The negative gain occurred in only one experiment. It is due to a high error in the predicted relative difference of accuracy and comprehensibility that made the tree with the maximum number of replaced leaves seem dominated and hence removed from the Pareto set. This can be avoided by forcing the algorithm to include the black-box classifier in the set of learned hybrid trees, which will guarantee that the gain of the MOLHC algorithm over the baseline will always be equal or greater than 0.

The difference in accuracy between a comprehensible classification tree and an accurate black-box classifier is less than 10% in 28 out of 49 datasets. In such domains there is not much room for a compromise between accuracy and comprehensibility. Nevertheless, we argue that using the data mining process with the MOLHC algorithm still provides valuable insights, because it offers a comprehensive comparison of classifiers. Even if the user finally decides to use the initial classification tree or the black-box classifier instead of the suggested hybrid trees, the user is informed about the difference in accuracy between

Table 3.2: Comparison of the MOLHC algorithm with baseline according to the hypervolume measure. Columns with two numbers delimited by slash sign represent values for a small (on the left) and a big initial classification tree (on the right).

Dataset	MOLHC hypervolume	Baseline hypervolume	Absolute hypervol. difference	Relative hypervol. difference in %
mfeat pixel	1.14/1.22	0.83/0.96	**0.3/0.26**	36.61/27.35
letter	1.11/1.22	0.82/0.98	**0.29/0.24**	35.98/24.84
vowel	1/1.14	0.77/1.02	**0.23/0.12**	29.98/12.22
soybean	1.16/1.23	0.98/1.2	**0.18/0.03**	18.62/2.73
mfeat karhunen	1.31/1.31	1.12/1.15	**0.18/0.16**	16.27/13.74
car	1.38/1.38	1.23/1.28	**0.15/0.1**	11.98/7.58
optdigits	1.34/1.37	1.2/1.26	**0.14/0.11**	11.71/8.87
mfeat zernike	1.15/1.15	1.04/1.03	**0.11/0.12**	10.82/11.94
audiology	1.03/ -	0.93/ -	**0.1**/ -	10.58/ -
tic-tac-toe	1.32/1.32	1.19/1.24	**0.13/0.08**	10.54/6.78
heart statlog	1.21/1.22	1.11/1.08	**0.11/0.14**	9.66/12.65
flags	0.92/0.91	0.85/0.9	**0.07/0.02**	8.24/1.9
splice	1.32/1.34	1.23/1.26	**0.09/0.08**	7.44/6.59
anneal	1.32/ -	1.23/ -	**0.09**/ -	7/ -
mfeat fourier	1.21/1.22	1.13/1.16	**0.08/0.07**	6.93/5.9
vehicle	1.11/1.13	1.05/1.07	**0.05/0.06**	5.14/5.62
waveform	1.22/1.24	1.16/1.16	**0.06/0.08**	4.94/6.88
cylinder bands	1.11/ -	1.06/ -	**0.05**/ -	4.58/ -
dermatology	1.35/1.39	1.31/1.34	**0.05/0.05**	3.52/3.81
primary tumor	0.68/0.67	0.67/0.68	**0.02/-0.01**	2.74/-1.52
balance scale	1.23/ -	1.2/ -	**0.03**/ -	2.38/ -

the two classifiers, the available alternatives (i.e. the hybrid trees), and the relative qualities of the leaves in the initial classification tree, which helps in the validation of the tree.

3.5.4 Comparison with a Multi-Objective Optimisation Algorithm

The MOLHC algorithm was compared with the well-known multi-objective optimization algorithm NSGA-II (Deb et al., 2002). The task of both algorithms was to find the (approximation of) Pareto set of hybrid trees. NSGA-II is a stochastic evolutionary algorithm that tries to converge to the Pareto set. The parameters of the NSGA-II algorithm were set as follows: two point crossover was used as the crossover technique, and bit flip as the mutation operator with the probability of a mutation equal to $1/$*number of leaves* for each bit. Three sizes of population were set to: $6.3 \times$ *number of leaves* $- 15$ (medium population), half that number (small population) and twice that number (big population). The formula linearly approximates the number of hybrid trees in the Pareto set in relation to the number of leaves considered to be marked as hybrid. In a general setting, an appropriate size of the population is not trivial as the number of hybrid trees in the Pareto front is not known in advance. The stopping criterion was defined by setting the NSGA-II execution time limit to 10, 50 and 100 times longer than the average run-time over 10 runs of the MOLHC algorithm. The number of generations was not limited explicitly; however it depended on the run-time limit and the size of the population: longer run-time corresponds to higher number of generations while bigger population size causes the number of generations to decrease for a constant run-time. The DEAP framework (Distributed Evolutionary Algorithms in Python) (Fortin, De Rainville, Gardner, Parizeau, & Gagné, 2012) implementation of NSGA-II was used.

Table 3.3 shows the average relative difference in hypervolume expressed in % between the MOLHC algorithm and the average over five runs of the NSGA-II algorithm. The table shows the average over six domains. The results are similar in all the tested domains, because they depend more on the size of the initial classification tree than on the domain itself. The main motivation of the experiment is to provide empirical data to support our claims about the advantages of the MOLHC algorithm over the general multi-objecttive algorithms, therefore we did not perform the tests for as many domains as in the previous section. Table 3.3 includes the results for the two sizes of the initial classification trees, the three run-time multipliers, and the three sizes of the population used by NSGA-II. The results show that the suggested algorithm outperforms NSGA-II when searching for the trees of comprehensible size; the result is expected because the suggested algorithm is exact (i.e. guaranteed to find the complete Pareto set of hybrid trees) while it is not possible to say whether or not NSGA-II converged close to the Pareto set in a limited run-time. Table 3.3 shows that the difference in hypervolume decreases with decreasing population size (i.e. increasing number of generations) for short run-time limit or for big search space (i.e. initial tree with many leaves). In such cases NSGA-II does not have enough time to converge to the Pareto set well: it converges closer to it if it evaluates more generations, which is only possible when the population size is small. Furthermore, Table 3.3 shows that the difference in hypervolume decreases with decreasing size of the population for a small search space (i.e. few leaves in the initial classification tree) and long run-time. In such cases NSGA-II has enough time to converge to the Pareto set, but if the size of the population is too small it finds only some of the hybrid trees in the Pareto set. The low difference in hypervolume for a big population is caused by the stochastic nature of the NSGA-II: even if NSGA-II has enough time to converge to the Pareto set and big enough population, its stochastic nature prevents it from finding the complete Pareto set of hybrid trees in each run. The MOLHC algorithm is therefore computationally more efficient than NSGA-II for reasonably sized initial trees. In addition, it is also preferred because it does not require any parameter setting and is exact.

Table 3.3: Relative difference in hypervolume between MOLHC and NSGA-II (in %) for three run-times and population sizes of NSGA-II, and two sizes of initial tree.

Population / Run-time	**Small tree** 12.8 leaves			**Big tree** 22 leaves		
	Big	Medium	Small	Big	Medium	Small
$t(MOLHC) \times 10$	3.8	1.8	1.4	5.5	4.9	2.7
$t(MOLHC) \times 50$	0.2	0.1	0.2	2.8	0.6	0.4
$t(MOLHC) \times 100$	0.02	0.03	0.2	1.5	0.3	0.1

NSGA-II may be preferred for large initial trees with more than 100 leaves in order to obtain an approximation of the Pareto set, since run-times of the MOLHC algorithm reach the limits of practicality for such tree sizes. However, big classification trees are considered incomprehensible in most cases, therefore using them as an input to the MOLHC algorithm does not make much sense. Furthermore, the MOLHC algorithm can be stopped at any time to return a subset of the Pareto set; this is an approximation of the Pareto set, which is similar to using a multi-objective optimization algorithm.

3.6 Application of the MOLHC Algorithm

In order to evaluate the usability of the MOLHC algorithm and its results in practice, a case study with a domain expert using the MOLHC algorithm in the activity-recognition

domain is presented. The goal of activity recognition is to recognize the activity a person is performing using sensors and software for processing the sensor data. This section considers learning a classifier that distinguishes between 10 basic activities (listed in the following paragraph) based on attributes extracted from data provided by a single 3-axis accelerometer mounted on the person's chest.

The training and testing dataset was recorded in a laboratory with 9 persons, each performing a given sequence of activities lasting for approximately 1.5 hours: 22% of the time lying, 17% walking, 14% cycling, 10% standing, 7–8% sitting, kneeling, on all fours, and running (each), 4% transitions between activities, and 3% leaning. Two-second time windows of measured accelerations along each of the 3 accelerometer axes were used to compute 61 attributes suggested by the literature, for example: mean value, area under the curve, amplitude, total energy, dominant frequency, mean crossing rate, entropy, variation coefficient, etc. The time windows were overlapping (1 second overlap with the previous window and 1 second overlap with the following window), therefore a total of around 48.000 instances were acquired.

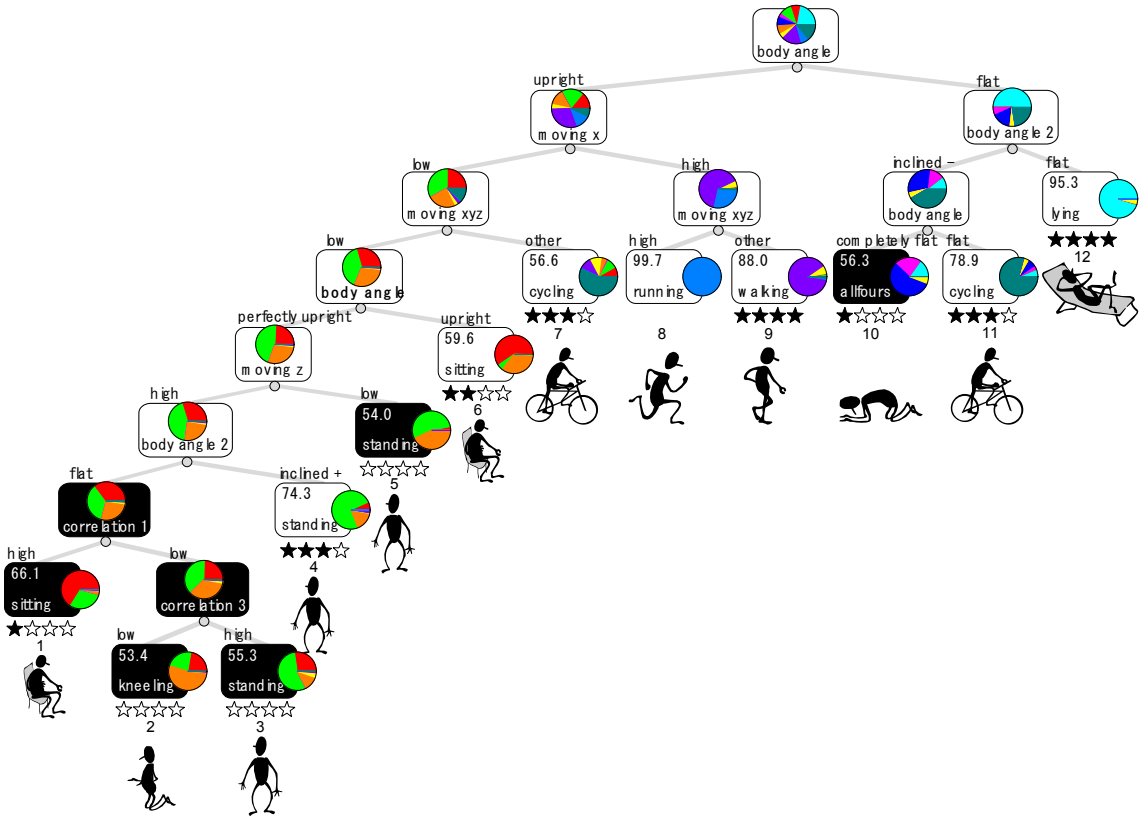


Figure 3.7: Output of the MOLHC algorithm for the activity-recognition domain. The black-box leaves of the chosen hybrid tree are coloured black. More black stars means better relative quality of a leaf. The pie charts represent the class distributions in the nodes and the numbers represent the classification accuracy in the leaves.

The activity-recognition classifier trained on the data was used at the EvAAL live activity-recognition competition (http://evaal.aaloa.org/). A very accurate black-box classifier (90.6% classification accuracy) was learned from the high-quality laboratory data; however, the expert could not validate it. On the other hand, completely comprehensible classifiers performed poorly (77.0% classification accuracy) in comparison. This was thus a problem that called for a hybrid approach using the MOLHC algorithm. A set of hybrid

trees — ranging from the most comprehensible to the most accurate — was generated as a result. The hybrid trees were analysed by an activity-recognition expert to obtain an insight into the activity-recognition problem and learn a hybrid tree that has high enough accuracy and is as comprehensible as possible.

The case study involved an activity-recognition expert following the suggested data-mining process, which included several iterations. The inputs to the MOLHC algorithm (an initial classification tree and a black-box classifier) were selected in each iteration based on the expert's knowledge and the analysis of hybrid trees obtained in the previous iteration. In the first iteration, the activity-recognition expert found the initial classification tree (67.9% accuracy, 8 leaves) constructed using all the 61 attributes as well as the resulting hybrid trees poorly comprehensible, so he selected a subset of 12 attributes to build a new initial tree (67.4% accuracy, 8 leaves) for the second iteration. From the resulting Pareto set, the expert chose the hybrid tree in which three leaves — containing mostly instances of three classes that are difficult to distinguish using simple rules — were merged into a single black-box leaf. Hence, the hybrid tree contained 6 leaves; its accuracy was 80.2% and comprehensibility 0.689. In the third iteration (Pareto front in Figure 3.6b) a larger initial tree (76.1% accuracy, 12 leaves, obtained by changing the values of pruning parameters) was used in order to improve the classification accuracy. The domain expert finally chose the hybrid tree shown in Figure 3.7, which achieves the accuracy 84.1% and comprehensibility 0.72. Therefore the expert decided to sacrifice 6.5% of the accuracy to obtain a classifier that enabled validation of 72% of instances, i.e. 72% of training instances belonged to the regular leaves. The chosen hybrid tree replaced a subtree (containing leaves 1–3 in Figure 3.7) with a single black-box leaf, which increased the overall accuracy by 3.3% and decreased comprehensibility by 0.13. It also replaced leaves 5 and 10, which increased accuracy by additional 4.7% and decreased comprehensibility by additional 0.15.

The domain expert was able to select a useful hybrid tree according to the trade-off between the accuracy and comprehensibility, which would not be possible if only traditional classification models were used. In addition, the domain expert suggested that an additional accelerometer on the thigh should be used in order to distinguish sitting from standing. He also confirmed that the four activities belonging to the leaves with the highest relative quality are easy to classify: an older version of the activity-recognition software (Kozina, Gjoreski, Gams, & Luštrek, 2013) that his team developed actually used hand-crafted rules, similar to the ones in the tree, to recognize the four activities.

3.7 Analysis of the Algorithm

This section discusses two issues related to the application of the MOLHC algorithm: a possible high number of hybrid trees, which does not allow the user to compare all of the learned hybrid trees when choosing the best one, and the errors in the estimated accuracy and comprehensibility of the learned hybrid trees, which may mislead the user when choosing a hybrid tree.

3.7.1 The Number of Hybrid Trees in the Pareto Set

Figure 3.8 shows that the number of hybrid trees and hence the density of the Pareto front depends on the number of leaves in the initial tree that are considered for replacing with the black-box classifier. The number of hybrid trees in the Pareto set also depends on the number of leaves with the same non-domination level, which is not known in advance — this explains the outliers in Figure 3.8. In general, the number of Pareto-optimal hybrid trees is manageable, especially considering that not all of them need to be analysed (Piltaver,

Luštrek, & Gams, 2014). Furthermore, new knowledge about the domain can be extracted by analysing the Pareto front.

The first step in Pareto front analysis is to observe the steepness of the Pareto front, which corresponds to the difficulty of classification task for a decision tree. A steep Pareto front (e.g., Figure 3.6a) represents a domain in which the difference in accuracy between the initial tree and the black-box classifier is small. In such cases user should consider choosing the initial tree because it achieves similar accuracy as the black-box classifier but is completely comprehensible as opposed to the incomprehensible black-box classifier. On the other hand, a gradually decreasing Pareto front (e.g., Figure 3.6b) represents a domain with considerable difference in accuracy between the initial tree and the black-box classifier. In such domains the user should investigate the Pareto front further in order to select a hybrid tree with a desired trade-off between accuracy and comprehensibility. Figure 3.6 shows that Iris dataset is easy to classify using a comprehensible classifier, while a simple classifier does not suffice to classify the activity-recognition datasets with high accuracy.

The second property of the Pareto front to be investigated is the density and spread of hybrid trees along the Pareto front. For instance the Pareto front for Iris dataset (Figure 3.6a) is sparse (includes only two hybrid trees), while the Pareto front for activity-recognition dataset (Figure 3.6b) is dense. If there are few hybrid trees on a Pareto front, the user can inspect and compare all of them, otherwise the user should concentrate on a subset of hybrid trees whose accuracy and comprehensibility are appropriate for the given application. Pareto front may be well spread along the entire range or may be considerably sparser in some parts (e.g., the low comprehensibility range of the Pareto front in Figure 3.6b). Hybrid trees in the sparse part of the Pareto front should be inspected thoroughly, while only a subset of trees needs to be inspected in the dense part because it contains many similar trees. The similarities usually become obvious quickly, however a program with an appropriate graphical user interface could simplify the task.

The third important property of the Pareto front is the presence of knees: parts of the Pareto front with a sudden jump in one of the objectives. Examples of knees are shown in Figure 3.6b; the most obvious one occurs where accuracy approaches 0.9. Knees are important because they limit the set of hybrid trees that need to be examined. Among the hybrid trees near a knee, the ones that have extreme values of an objective are the most interesting. For instance, if a hybrid tree with high accuracy is requested in the activity-recognition domain (Figure 3.6b), then the one with accuracy 0.89 and comprehensibility 0.55 is a good candidate. It has almost the same accuracy as the hybrid trees further down the Pareto front but it has considerably higher accuracy.

3.7.2 Errors in the Estimated Comprehensibility and Accuracy

Two additional important measures for evaluating the MOLHC algorithm are the average errors in the accuracy e_a (Eq. (3.11)) and comprehensibility e_c (Eq. (3.12)) of the Pareto optimal hybrid trees estimated on the training dataset compared to performance on a separate test dataset. This is important since the user selects a hybrid tree based on the Pareto front (Figure 3.6), where the hybrid trees are plotted based on their performance on the training dataset. If these approximations used to draw the Pareto front significantly differ from the results on the dataset on which the selected hybrid tree will be used, the user is misled and may not choose the appropriate hybrid tree. This issue is similar to the problem of estimating the predictive performance of a classifier in single-objective learning. The only difference is that the user will immediately notice a poor classifier choice due to inaccurate estimation of its performance, when using the suggested data-mining process for multi-objective learning. In addition, the user will know that there are other classifiers

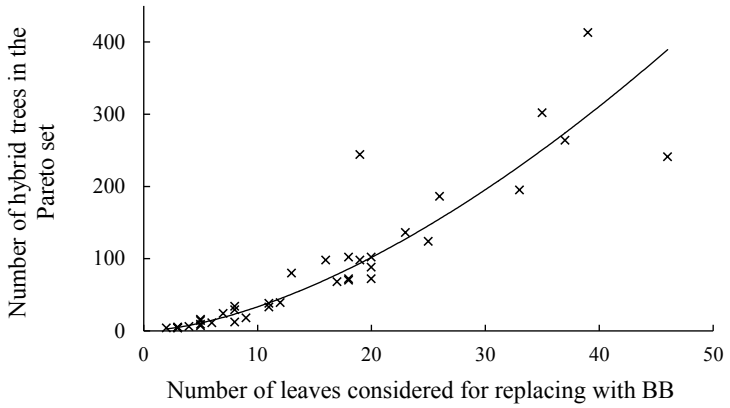


Figure 3.8: Number of hybrid trees in a Pareto set depends on the number of leaves in the initial tree that are considered for replacing with black-box leaves (results obtained from 38 initial trees built on the 21 UCI datasets).

(i.e., hybrid trees) from which he/she can choose. In contrast, selecting between alternative classifiers is done by the learning algorithm itself, hence the user may not notice the poor choice or be aware of the alternative classifiers when using traditional learning algorithms.

Definition 3.11 (Average error). The average errors in the predicted accuracy e_a and comprehensibility e_c are defined as the average absolute errors and are computed using Eq. (3.11) and Eq. (3.12), where S is the set of solutions returned by the algorithm with cardinality $|S|$ and a'_j and c'_j are the classification accuracy and the comprehensibility of the classifier j computed on the test set, respectively.

$$e_a = \sum_{j \in S} |a_j - a'_j| / |S| \tag{3.11}$$

$$e_c = \sum_{j \in S} |c_j - c'_j| / |S| \tag{3.12}$$

The errors depend on the number of training instances as shown in Figures 3.9 and 3.10. Insufficient number of training instances may lead to errors in the estimated comprehensibility and accuracy, which misleads the user when choosing a hybrid tree. Figure 3.6a shows one such example, which occurred because only 50 instances were used for the iris dataset. The problem is only amplified by the fact that the suggested algorithm requires three subsets of a dataset: the first for learning the initial tree and black-box classifier, the second for learning the Pareto set of hybrid trees and usually also the third for evaluating the hybrid trees. In our future work, we plan to improve the MOLHC algorithm using internal n-fold cross-validation. Preliminary results show that this could enable a more reliable performance on small datasets by limiting the errors of the predicted comprehensibility and accuracy of the hybrid trees.

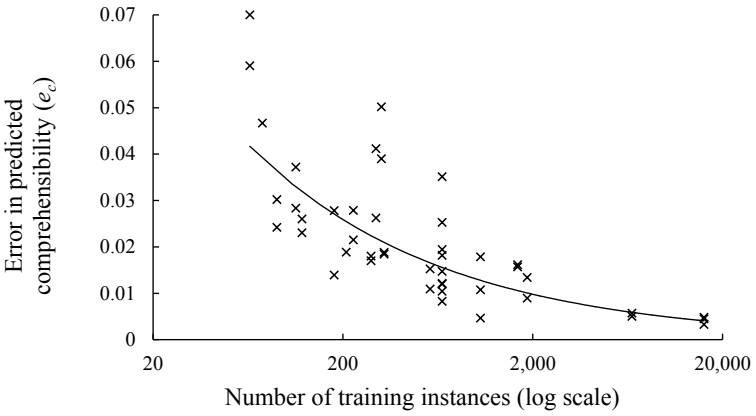


Figure 3.9: Error in predicted comprehensibility of hybrid trees depends on the number of learning examples (results obtained from 38 initial trees built on 21 UCI datasets).

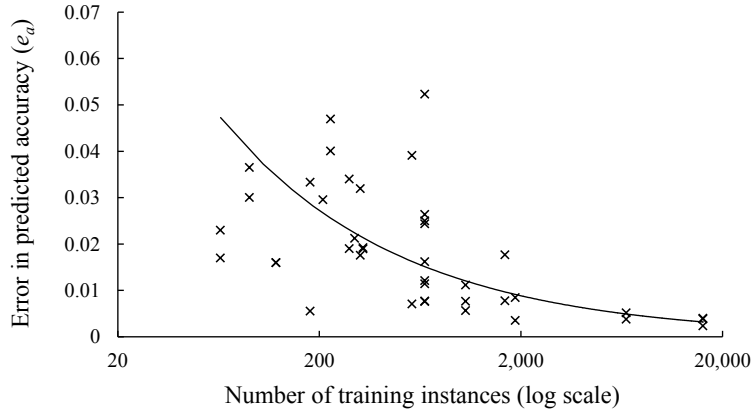


Figure 3.10: Error in predicted accuracy of hybrid trees depends on the number of learning examples (results obtained from 38 initial trees built on 21 UCI datasets).

3.8 Conclusion and Future Work

Most machine-learning research focuses only on improving the accuracy of classifiers, while their comprehensibility is the decisive factor when it comes to their practical applications. This chapter bridges the gap between comprehensible classifiers that often achieve lower accuracy (e.g., classification trees and rules) and more accurate but incomprehensible black-box classifiers (e.g., ANN, SVM, ensembles). Achieving a good trade-off is especially important in practical applications that require a classifier to be comprehensible and accurate at the same time.

We introduce a framework that includes the hybrid tree as a classifier model, the corresponding comprehensibility metric, the MOLHC learning algorithm, and the corresponding data-mining process. The hybrid trees provide trade-offs between the accuracy and comprehensibility that cannot be achieved using the traditional comprehensible or the black-box classifier alone. The proposed MOLHC algorithm returns the set of non-dominated hybrid trees with some regular comprehensible leaves and some more accurate black-box leaves. The dynamic programming approach and search space pruning is used to achieve low runtimes for reasonably sized trees. The proposed algorithm is also faster than the genetic multi-objective optimisation algorithm NSGA-II, guarantees finding the complete set of

non-dominated hybrid trees, and does not require setting the optimisation parameters. This enables using the algorithm within the proposed multi-objective data-mining process that helps the user choose the appropriate algorithm inputs, obtain insights into the classification problem, and finally select a hybrid tree with a good accuracy/comprehensibility trade-off. A case study of using the proposed algorithm for activity recognition shows that the algorithm is applicable in practice. Finally, the algorithm correctness of the proposed MOLHC algorithm is formally proven (Appendix D).

In order to improve the usability of the suggested multi-objective learning framework, several interesting issues for further research should be investigated. The first issue is learning more appropriate initial classification tree that is used as an input to the MOLHC algorithm. Existing algorithms for learning classification tree produce appropriate trees although their heuristic functions explicitly bias them towards trees with high accuracy. In order to improve the initial tree, the learning algorithm should be biased towards distributing the leaves that are easy to classify in one set of leaves and the instances that are difficult to classify in the other set of leaves. We developed prototypes of two algorithms that could be used to learn such classifiers.

The first algorithm starts by measuring how difficult it is to classify each instance in the training set using a comprehensible classifier. In order to do so, it builds a random forest and computes the average depth of the leaf that classifies an instance over all the trees in the forest for each of the instances. A penalty is added if a tree misclassifies an instance. The instances are then assigned weights that are inversely proportional to their average classification depth and a classification tree is built using an algorithm that can take instance weights into account. This increases the importance of the leaves that are easy to classify and decreases the importance of the instances that are difficult to classify, which results in a tree that is more appropriate as an input to the MOLHC algorithm. The preliminary results are encouraging, however we still need to perform a systematic evaluation.

The second algorithm is described in Piltaver (2011). It applies hierarchical clustering using a new distance measure. The distance between clusters is based on the distance between the instances in the attribute-value space and the classification accuracy of a classifier trained on the instances from one cluster applied to the instances from another cluster and vice versa. The MOLHC algorithm is then applied to the clusters by treating instances belonging to each cluster – instated of instances belonging to each leaf of a tree – as parts of the classification domain.

The second issue is the accuracy of the estimated quality of hybrid trees for small datasets. The user chooses the hybrid tree from a Pareto front that is drawn according to the estimated accuracy and comprehensibility. If the estimates have considerable error, the user is misled into choosing a hybrid tree whose accuracy and comprehensibility will be different than expected. Cross-validation technique could be used to mitigate this problem. The first option is to draw some kind of confidence intervals around the hybrid trees shown on the Pareto front. These intervals could be calculated as follows: divide the learning dataset into folds, calculate the classification accuracy and comprehensibility of the hybrid tree on each of the folds, and finally compute the interval widths as the variability of accuracy and comprehensibility over the folds. The second option would be to divide the learning dataset into folds and run the MOLHC algorithm on each of the folds. The accuracy and comprehensibility of the hybrid trees in each of the obtained Pareto sets should then be computed on the part of the dataset that was not used by the MOLHC algorithm in a particular run. Accuracy and comprehensibility of hybrid trees in each Pareto sets should then be shown on a single graph, which will provide information on the reliability of the estimated accuracy and comprehensibility of hybrid trees.

The third issue is selecting a hybrid tree from a potentially large set of learned hybrid trees. We suggest developing a program with a graphical user interface that supports the user when performing this task. A minimum requirement is a graphical user interface that shows the Pareto front of hybrid trees on one side and a visualisation of a currently selected hybrid tree on the other side. Another feature would be automatically changing the colour of the hybrid trees on the Pareto front that are similar to the currently selected hybrid tree (e.g. differ in a single leaf) and the hybrid trees that were already inspected. A more elaborate tool would cluster the hybrid trees in the Pareto set according to the similarity of binary vectors that represent the hybrid trees and similarity in terms of their accuracy and comprehensibility. The graphical user interface would then highlight only one hybrid tree per cluster while greying out the other similar hybrid trees (i.e. the ones from the same cluster). The highlighted hybrid tree could be the one with the highest accuracy, the highest comprehensibility, or the one that is the closest to a knee of the Pareto front.

Additional opportunity for future work are evaluating the MOLHC algorithm when applied to classification rules or tables instead of to classification trees, using multiple black-box classifiers (one per subtree) instead of a single one, and changing the order of generating the hybrid trees from breadth first to depth first. Finally, we intend to implement the MOLHC algorithm as a part of a popular data-mining software package such as Weka or Orange to encourage its usage.

Chapter 4

Conclusions

This final chapter presents the concluding remarks. A summary of the thesis is given in Section 4.1, while Section 4.2 recalls its original contributions. The chapter ends with future work ideas summarized in Section 4.3.

4.1 Summary

4.1.1 Classifier Comprehensibility

The related work on comprehensibility of classifiers is scarce especially considering the importance of comprehensibility in practical applications. The first part of this thesis therefore studies the influence of tree structure parameters, tree visualisation properties and question-depth on the comprehensibility and usability of classification trees. The results are empirically supported by the data about the influence of the investigated properties on the performance of end-user when performing four typical data-mining tasks using classification trees and data about end-users' subjective opinions about the difficulty of the tasks and tree comprehensibility. The data was obtained using a carefully designed survey performed on 69 data-mining experts and students.

The main results are as follows. Increasing the number of leaves increases the time-to-answer and question-difficulty; the effect is more evident in the tasks that require the user to inspect a large part of the tree (e.g., discovering new knowledge from a classification tree) compared to tasks that focus on a single path from the root to a leaf (e.g., classify an instance). Although the number of leaves corresponds well to the comprehensibility of the entire tree, it is not relevant if the needed information is close to the root of the tree or if it is contained in a single path from the root to a leaf of the tree. Increasing the branching factor increases the time-to-answer only for non-trivial questions. There seem to be two types of respondents that have opposite preferences in regard to the branching factor: some prefer deeper trees with binary branching factor while the other prefer shallower trees with a higher branching factor. The influence of the tree depth itself on the comprehensibility is small and could not be analysed systematically because it strongly depends on the other tree structure parameters. Increasing the question-depth, which corresponds to the depth of the deepest leaf that is required to answer a question, significantly increases the time-to-answer and question-difficulty and tends to decrease the answer-correctness. The question-depth has the highest influence on the comprehensibility among all of the observed parameters and is overlooked in the related work, therefore it should be studied further. The results on the question-depth suggest that users prefer trees that contain the most important information at the top of the tree. The relative importance and distribution of the instances according to the depth of the leaves to which they correspond should be

therefore accounted for by the tree comprehensibility metrics and tree learning heuristics.

The tree visualisation style has an important influence on the tree comprehensibility and usability: colour-coded information and meaningful attribute names, attribute values and class names should be used in tree visualisation. The tree layout is also important for many users.

Finally we evaluated several classification-tree comprehensibility metrics and propose two new ones that enable specifying user preferences that are related to the semantics of a classification tree rather than relying solely on the tree structure to estimate its comprehensibility.

The results summarized above confirm the first hypothesis investigated in this thesis stating that simple classification tree complexity measures do not take into account all the parameters that influence the classification tree comprehensibility. We show that comprehensibility depends on several tree structure parameters as well as on the tree visualisation. Furthermore, the high influence of question-depth on classifier comprehensibility shows that the distribution of the information over the leaves of the tree is important for the comprehensibility. Therefore, classification tree comprehensibility metric should take into account the semantic of the tree in addition to its structure.

4.1.2 Learning Comprehensible and Accurate Classifiers

The second problem that is addressed in this thesis is learning classifiers that are both accurate and comprehensible. The main challenge in achieving this goal is that the two objectives are conflicting – increasing one decreases the other (except if the classifier overfits the training data). Therefore most algorithms primarily focus on one of the objectives forcing the user to choose either a comprehensible classifier with relatively low accuracy or an incomprehensible classifier with relatively high accuracy. To solve the problem, we propose learning a set of classifiers with multi-objective learning approach, which enables the user to trade some of accuracy for comprehensibility or vice versa.

We propose hybrid tree as a classification model that enables trade-off between accuracy and comprehensibility and present the MOLHC (Multi-objective learning of hybrid classifiers) algorithm that finds the complete set of non-dominated hybrid trees. We evaluate the comprehensibility of the generated hybrid tree with the proposed measure, which is supported by the results of the survey. We empirically confirm that the MOLHC algorithm has several advantages over general multi-objective optimization algorithms: it does not require any parameter settings, guarantees finding the complete set of non-dominated hybrid trees and has lower run-times for reasonably sized trees. We show that the MOLHC algorithm generates hybrid trees with trade-offs between accuracy and comprehensibility that cannot be achieved using the existing learning algorithms. Finally, we confirm the practical applicability of the MOLHC algorithm by successfully applying it in the activity-recognition domain. This confirms the second hypothesis investigated in this thesis: the MOLHC algorithm outperforms the existing algorithms in terms of accuracy and comprehensibility of the produced classifiers. The set of classification domains that we used to evaluate the MOLHC algorithm represent examples of domains that can only partially be classified with comprehensible classifiers, which confirms the first part of the hypothesis.

4.2 Original Contributions

The thesis leads to two major original scientific contributions and several related contributions listed below.

1. Systematic analysis of the influence of tree-structure and tree-visualization parameters on classification tree comprehensibility based on survey respondents' performance and subjective opinions. The analysis introduced the following original contributions:

   - A general design of an in-depth survey for empirical study of the classifier comprehensibility based on best-practices and mistakes from related work, cognitive theory and survey-bias prevention strategies. The survey measures respondents' performance with two objective measures (time to answer a survey question and answer correctness) in four data-mining tasks (classify an instance, explain instance classification, verify statement about the domain, discover new knowledge) and respondents' subjective opinions (question-difficulty and absolute and relative tree comprehensibility). The survey design was validated by psychologists, data-mining experts and a test group of respondents.
   - A rich dataset for empirical study of the classification-tree comprehensibility obtained from the end-user survey with 69 data-mining experts and students.
   - Evaluation of most commonly used tree-comprehensibility metrics and two new metrics (the weighted sum of the depths of leaves and the weighted sum of the branching factors on the paths from the root to the leaves) that consider the semantics of the tree in addition to the tree structure itself.
   - Introduction of question-depth concept (the depth of the deepest leaf that is required when answering a question about a classification tree), which is shown to be the parameter with the highest impact on tree comprehensibility.
   - Validation of the number of leaves as a comprehensibility metrics for large trees and empirical proof that it should be avoided for small trees or when comprehensibility of a single part of the tree is of interest.
   - Empirical confirmation of the fact that classification-tree visualization style and layout significantly influence the comprehensibility.

2. The algorithm for multi-objective learning of hybrid classifiers (MOLHC), which is formally proven to find the complete set of non-dominated hybrid trees with regard to their accuracy and comprehensibility and has the following novel properties:

   - It uses a novel hybrid tree classification model that offers a range of trade-offs between tree comprehensibility and accuracy that is not available using the existing algorithms.
   - It generates the entire Pareto-set of hybrid trees in a single run, which enables the user to better understand modelling of the problem domain.
   - Its run-time is substantially shorter than the run-time of the naive exhaustive search algorithm enabling application of the MOLHC algorithm in real-life use-cases.
   - In comparison to the multi-objective evolutionary optimization algorithm NSGA-II, the MOLHC algorithm does not require any parameter settings, guarantees finding the complete Pareto-set and is faster for moderate-size trees.
   - It evaluates the comprehensibility of generated hybrid-trees based on a novel comprehensibility metric computed as the fraction of examples classified by the comprehensible part of the hybrid tree.

The list of publications related to this thesis is given in the Bibliography section.

4.3 Future Work

In future work related to the comprehensibility of classification trees, we plan to study the influence of tree structure parameters in larger trees, trees with higher (and non-uniform) branching factors, and real-life domains that involve strong numeric components, such as in medical, insurance, or bank loans applications. Studying trees with higher complexity and increasing the number of survey respondents will clarify the results about the influence on the answer-correctness and the importance of branching factor. Another promising direction for future study is analysing the influence of the visualisation properties, especially the tree layout, using the objective measures such as time-to-answer, answer-correctness. Furthermore, data mining methods could be applied to the collected dataset instead of the statistical tests to explain the interplay of various parameters, and derive a formal model of classification-tree comprehensibility. A deeper investigation of the influence of the question-depth could ultimately lead to learning algorithms that will produce more comprehensible trees, improved classification tree comprehensibility metrics, and methods that will produce more readable tree visualisation. The survey design and experience presented in this thesis could also be used to study the comprehensibility of classification rules or some other classification model.

In order to improve the usability of the MOLHC algorithm, several issues for further research should be investigated. The first issue is learning more appropriate initial classification tree that is used as an input to the MOLHC algorithm. The two algorithm prototypes for partitioning a classification domain into simple and complex parts could be used as a stepping stone in this direction. The second issue is the accuracy of the estimated quality of hybrid trees, which is especially sensitive in small datasets. Cross-validation technique could be used to reduce the error or at least provide information about the reliability of the estimates. The third issue is selecting a hybrid tree from a potentially large set of learned hybrid trees. Programs with appropriate graphical user interface will enable the user to efficiently compare the most interesting hybrid trees in the Pareto set obtain new insights into the classification domain and finally choose the most appropriate classifier. Furthermore, it would be interesting to evaluate the performance and usability of the MOLHC algorithm when applied to classification rules or tables instead of to classification trees. An opportunity for improving the classification accuracy of hybrid trees is using multiple black-box classifiers. Finally, implementation of the MOLHC algorithm as a part of a popular data-mining software package will support its broader adoption and enable practical applications.

Appendix A

Graphs About Parameters that Influence Tree Comprehensibility

The most interesting graphs that show the influence of question-depth, the number of leaves and branching factor on the time-to-answer and question-difficulty are shown below. Each graph shows the influence of a single question parameter on a respondents' performance metric. The remaining two question parameters are fixed in each sub-plot while one increases over the rows and the other over the columns. The annotations in the sub-plots represent the raw p-values of the Wilcoxon signed-rank tests: a minus sign for a p-value above 0.05 and *, **, *** or **** for p-values below 0.05, 0.01, 0.001 or 0.0001, respectively.

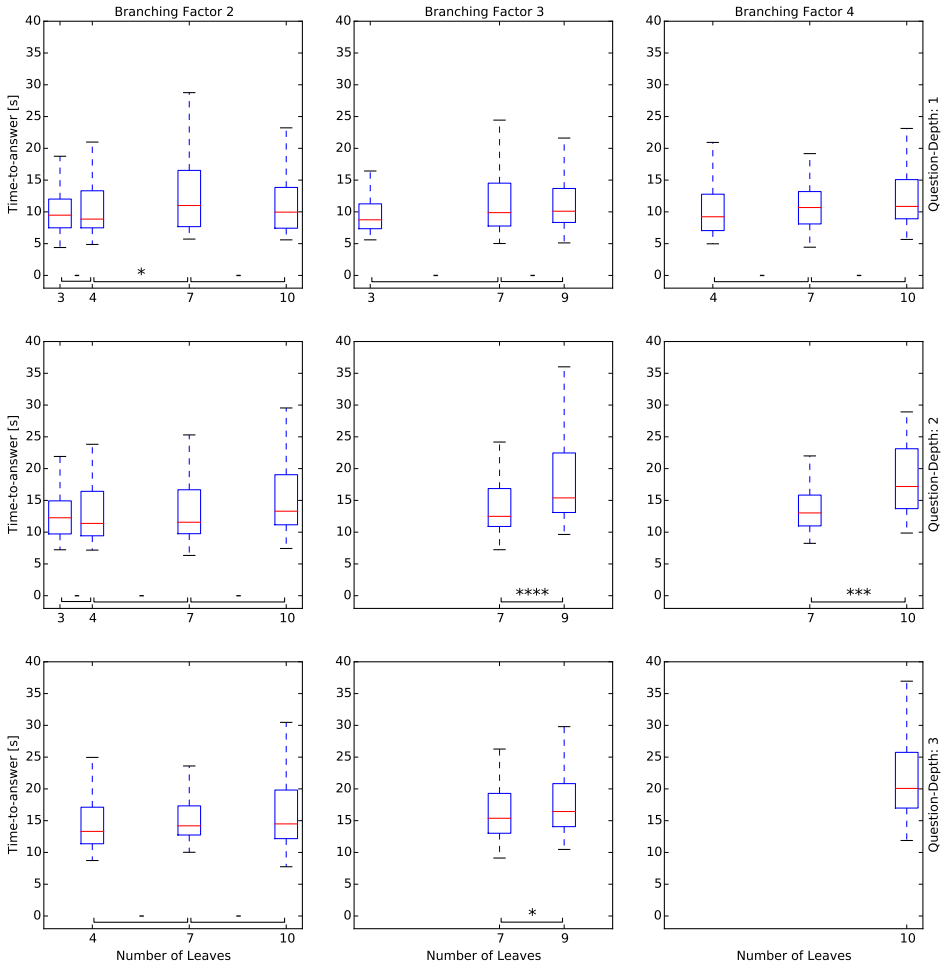


Figure A.1: The influence of the number of leaves on the time-to-answer for a question with a given question-depth (row) about a tree that has a given branching factor (column) in the *classify task*. The annotations in the sub-plots represent the raw p-values of the Wilcoxon signed-rank tests: a minus sign for a p-value above 0.05 and *, **, *** or **** for p-values below 0.05, 0.01, 0.001 or 0.0001, respectively.

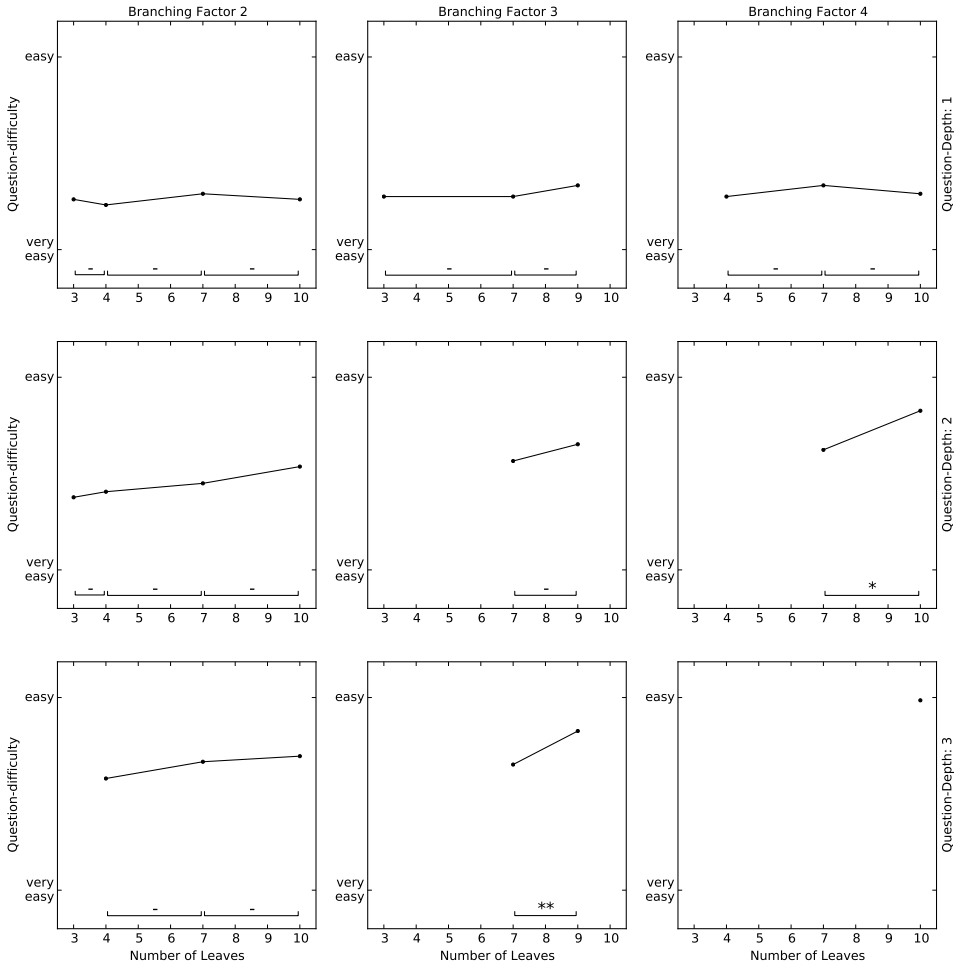


Figure A.2: The influence of the number of leaves on the question-difficulty for a question with a given question-depth (row) about a tree that has a given branching factor (column) in the *classify task*. The annotations in the sub-plots represent the raw p-values of the Wilcoxon signed-rank tests: a minus sign for a p-value above 0.05 and *, **, *** or **** for p-values below 0.05, 0.01, 0.001 or 0.0001, respectively.

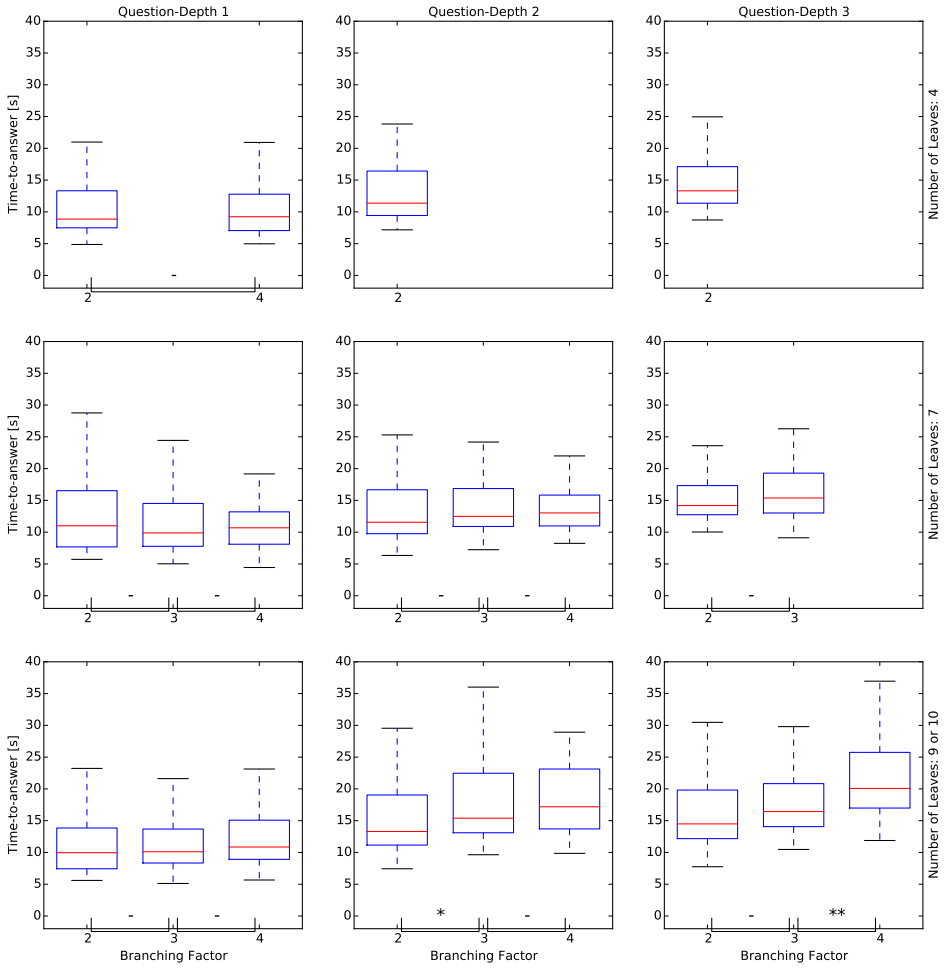


Figure A.3: The influence of the branching factor on the time-to-answer for a question with a given question-depth (column) about a tree that has a given number of leaves (row) in the *classify task*. The annotations in the sub-plots represent the raw p-values of the Wilcoxon signed-rank tests: a minus sign for a p-value above 0.05 and *, **, *** or **** for p-values below 0.05, 0.01, 0.001 or 0.0001, respectively.

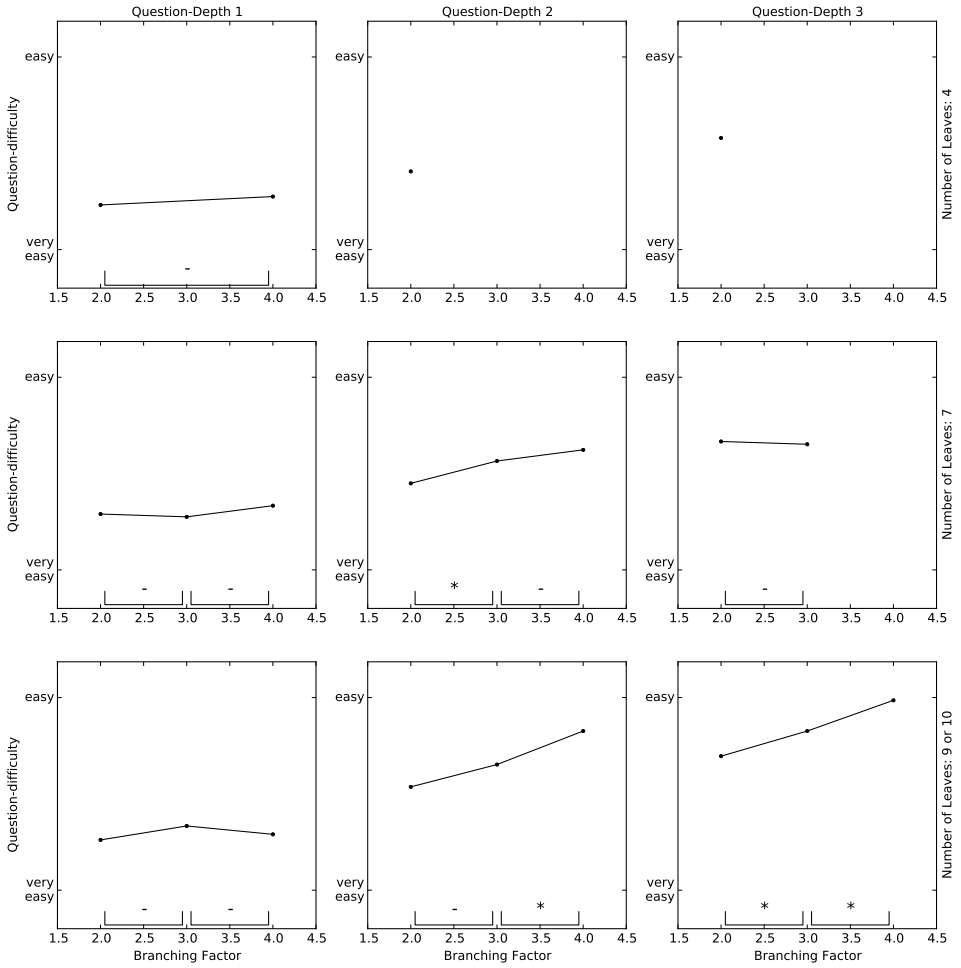


Figure A.4: The influence of the branching factor on the question-difficulty for a question with a given question-depth (column) about a tree that has a given number of leaves (row) in the *classify task*. The annotations in the sub-plots represent the raw p-values of the Wilcoxon signed-rank tests: a minus sign for a p-value above 0.05 and *, **, *** or **** for p-values below 0.05, 0.01, 0.001 or 0.0001, respectively.

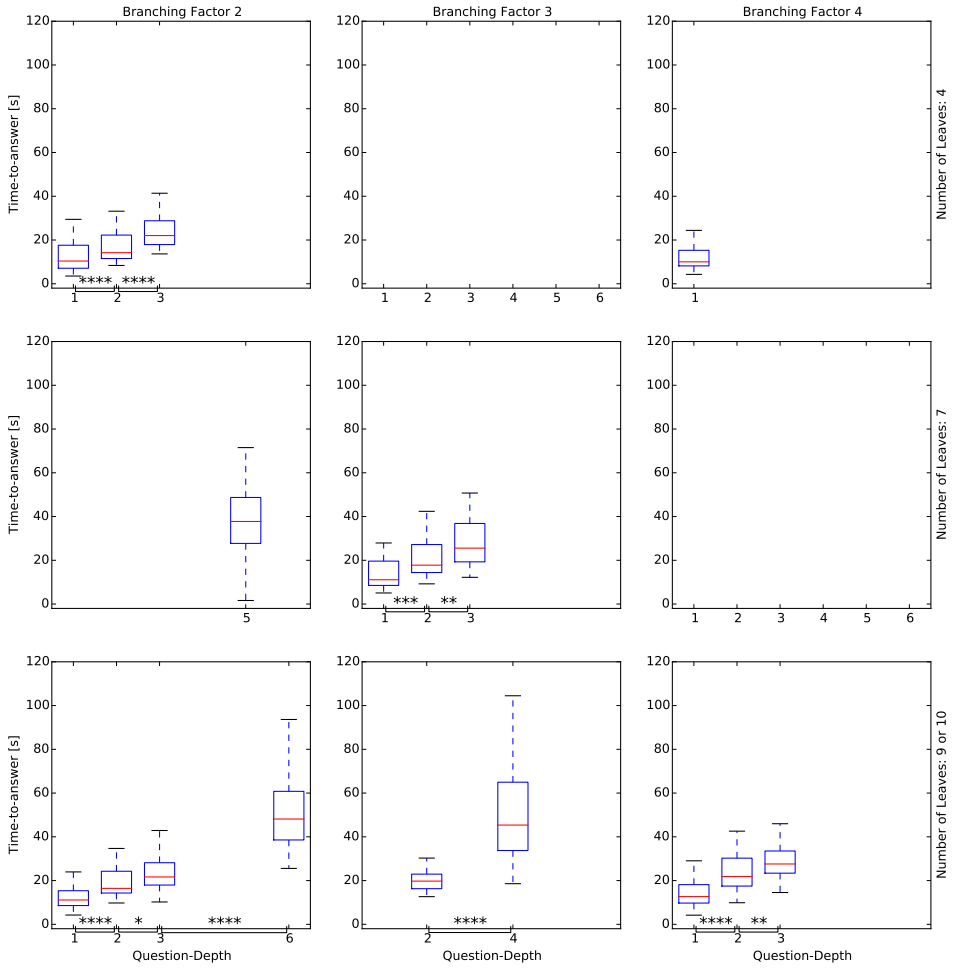


Figure A.5: The influence of the question-depth on the time-to-answer for a question about a tree that has a given branching factor (column) and number of leaves (row) in the *explain task*. The annotations in the sub-plots represent the raw p-values of the Wilcoxon signed-rank tests: a minus sign for a p-value above 0.05 and *, **, *** or **** for p-values below 0.05, 0.01, 0.001 or 0.0001, respectively.

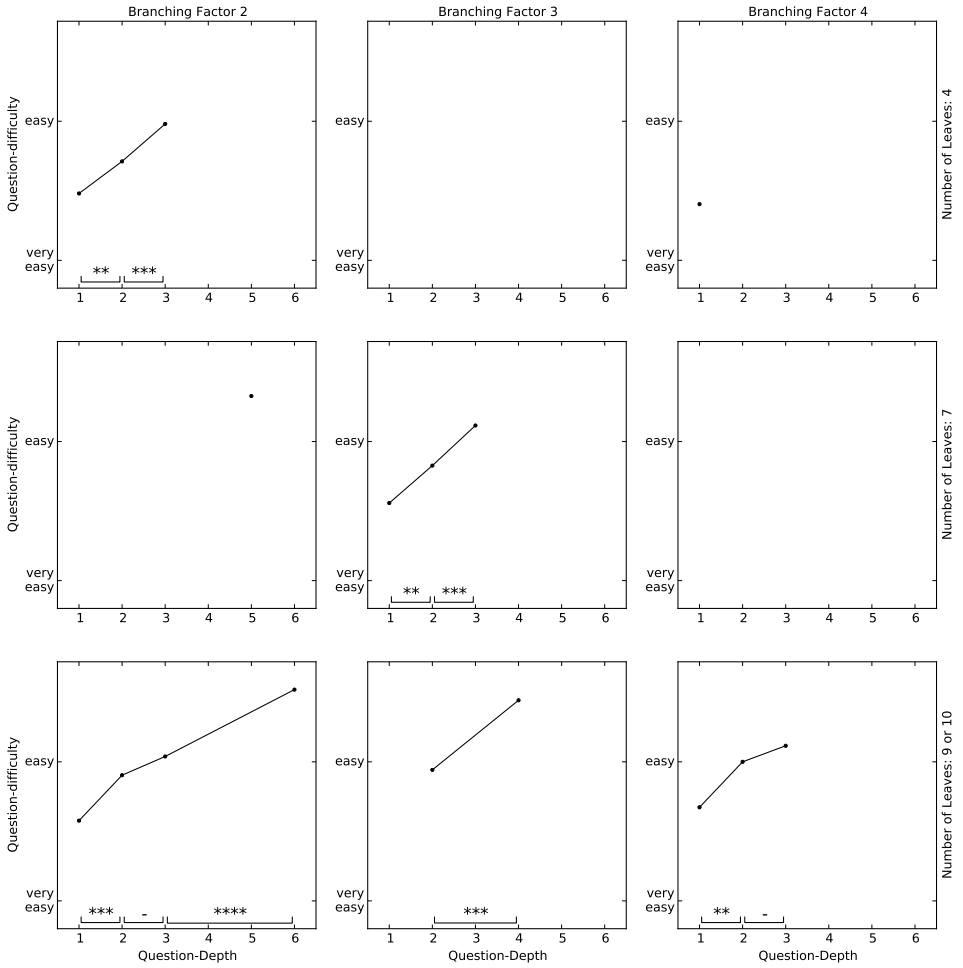


Figure A.6: The influence of question-depth on the question-difficulty for a question about a tree that has a given branching factor (column) and number of leaves (row) in the *explain task*. The annotations in the sub-plots represent the raw p-values of the Wilcoxon signed-rank tests: a minus sign for a p-value above 0.05 and *, **, *** or **** for p-values below 0.05, 0.01, 0.001 or 0.0001, respectively.

Appendix B

Survey Question Examples

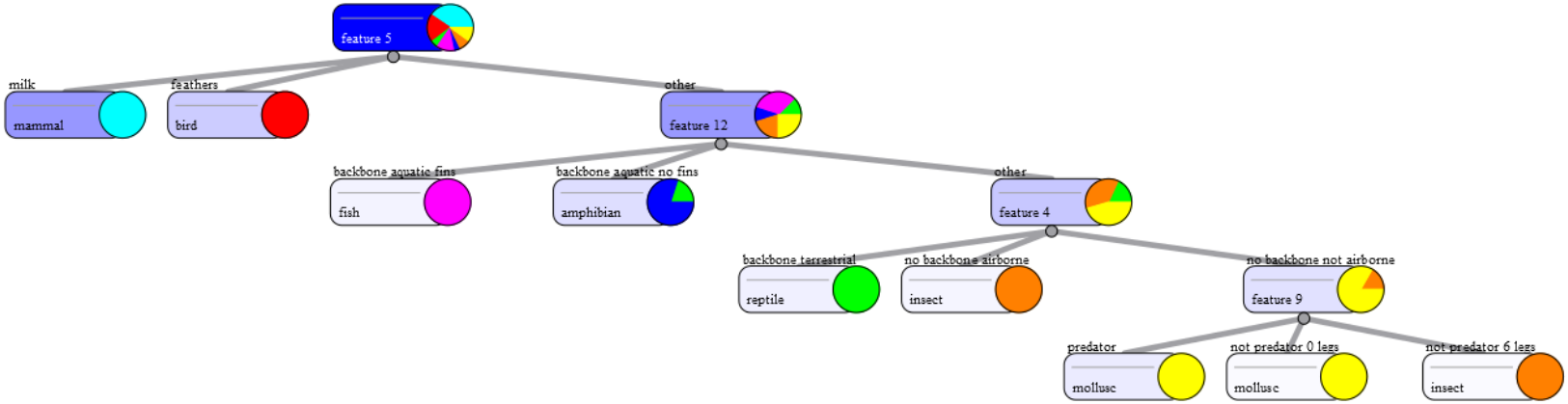
Examples of a question for each of the six tasks are presented below.

Classify Task

Classify the instance described in the table on the left using the classification tree on the right.

Attribute	Value
breathes	yes
eggs	yes
feature 12	backbone aquatic no_fins
feature 4	backbone aquatic
feature 5	other
feature 9	predator
hair	no
tail	no
toothed	yes
venomous	yes

Classification tree



The instance belongs to **class**: amphibian

Mark how **difficult** was the **task**:

- Very easy - I answered without any problems in less then 7 seconds
- Easy - I found the answer quite quickly and without major problems
- Medium -
- Difficult - I had to think hard and am not sure if I answered correctly
- Very difficult - Despite thinking very hard my answer is likely to be wrong

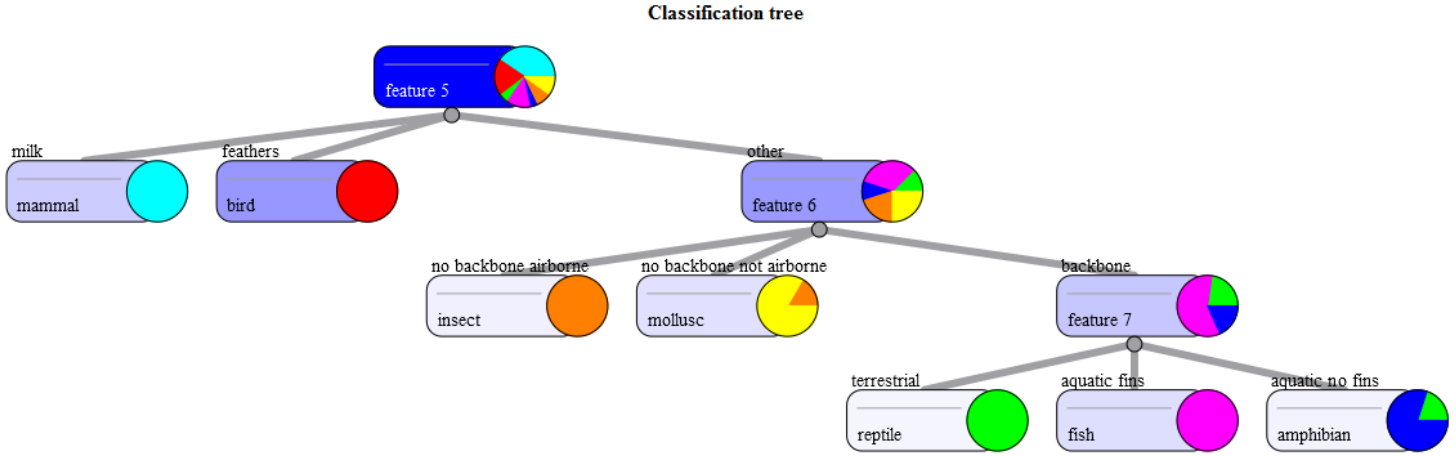
Next question

Figure B.1: An example of a question in the *classify task*

Explain Task

What should be the values of the attributes for the example (listed in the table on the left) so that the classification tree would classify it as **mollusc**?
If the value of an attribute is not relevant for the classification into the new class, maintain the default choice (i.e. "irrelevant"); otherwise change it to "unchanged" if the value must remain as it is, or "different" if it needs to be changed.

Attribute	Value	New value
breathes	yes	irrelevant
eggs	yes	irrelevant
feature 5	other	unchanged
feature 6	no_backbone airborne	different
feature 7	terrestrial	irrelevant
hair	yes	irrelevant
legs	6	irrelevant
predator	no	irrelevant
toothed	no	irrelevant
venomous	yes	irrelevant



Mark how **difficult** was the **task**:

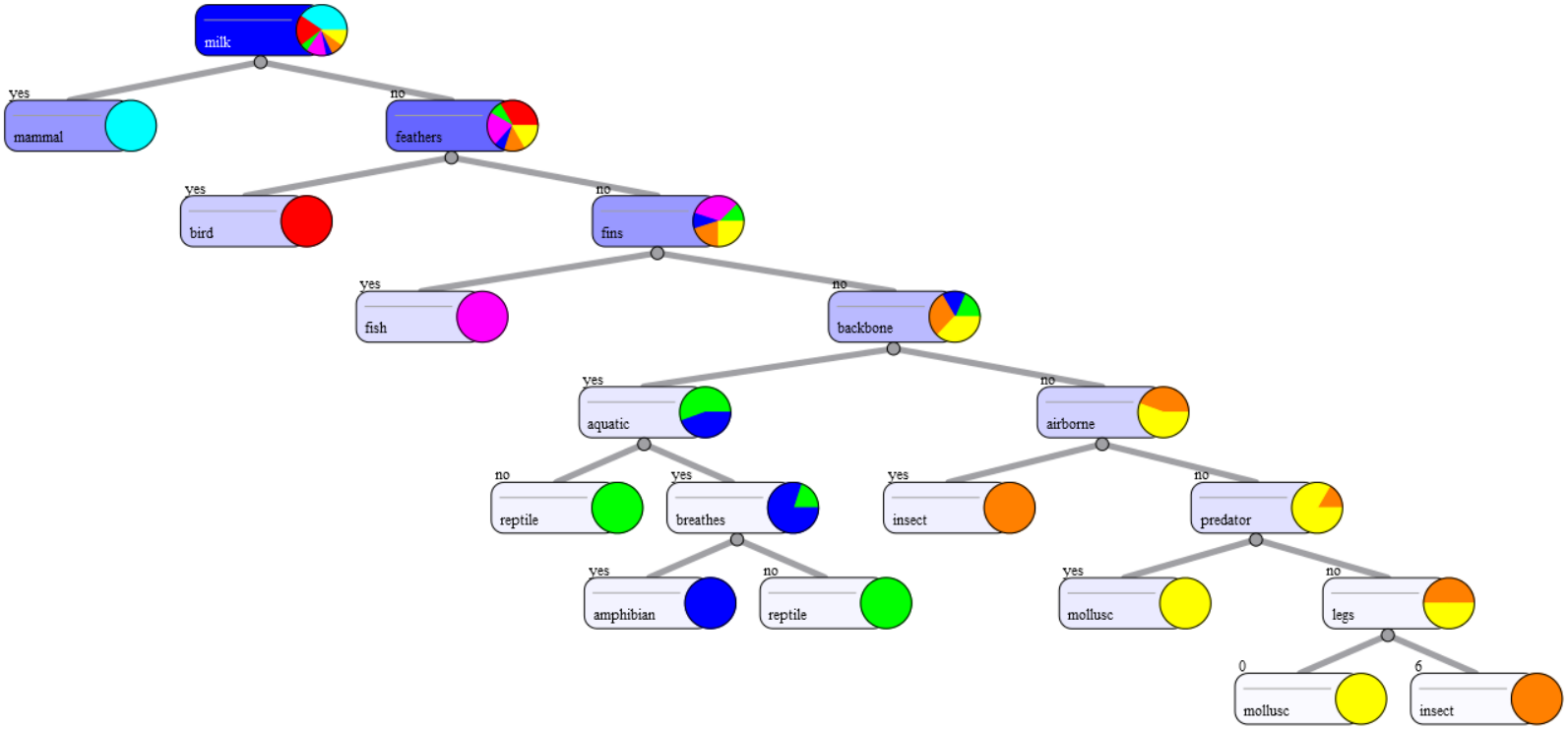
- Very easy - I answered without any problems in less then 7 seconds
- Easy - I found the answer quite quickly and without major problems
- Medium -
- Difficult - I had to think hard and am not sure if I answered correctly
- Very difficult - Despite thinking very hard my answer is likely to be wrong

Next task

Figure B.2: An example of a question in the *explain task*

Validate Task

Does the **classification tree agree with** the **statement** below it:



For animals from class **amphibian** it holds that **backbone=yes** and **breathes=no**.

The above **statement agrees with** the classification **tree**: no

Mark how **difficult** was the **task**:

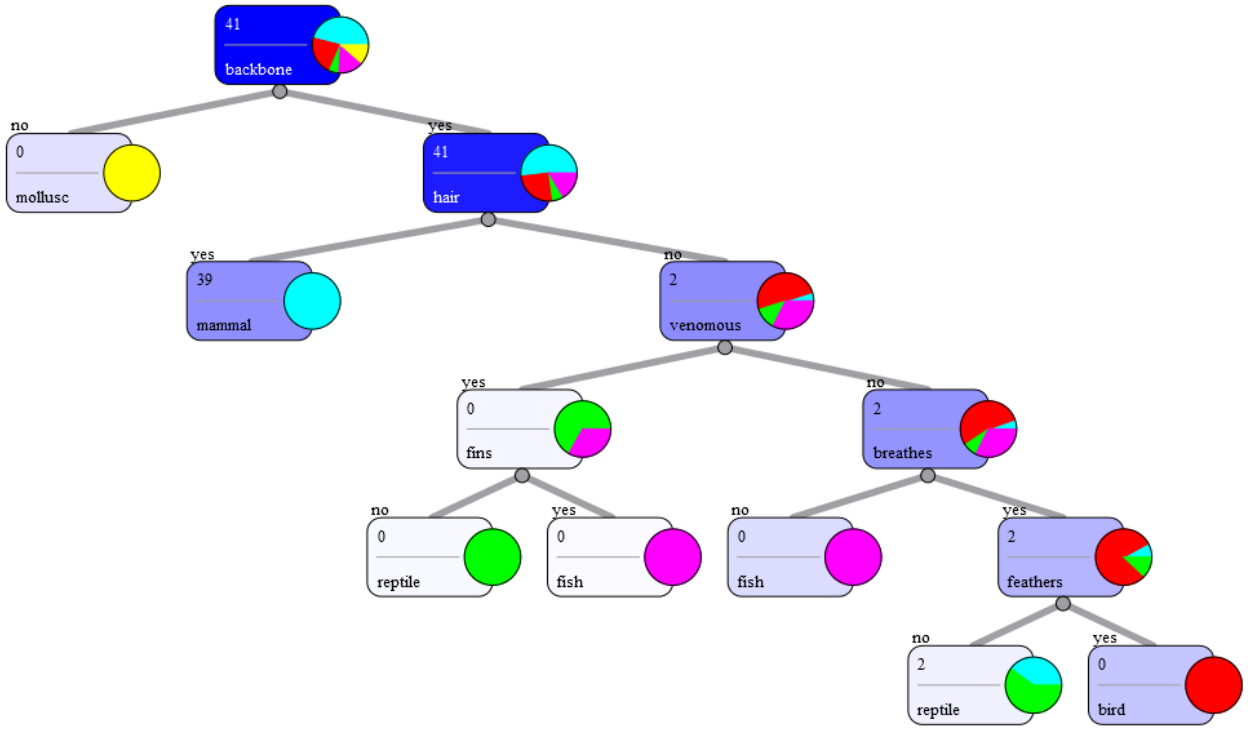
- Very easy - I answered without any problems in less then 7 seconds
- Easy - I found the answer quite quickly and without major problems
- Medium -
- Difficult - I had to think hard and am not sure if I answered correctly
- Very difficult - Despite thinking very hard my answer is likely to be wrong

Next task

Figure B.3: An example of a question in the *validate task*

Discover Task

Most animals from class **mammal** have the same properties; few animals from this class have a rare property. **Which is this rare property** (select the attribute name and its value for the animals with the rare property)? The number of animals from class mammal is given in each node.



The **rare property** of the animals from class **mammal** is: hair = no

Mark how **difficult** was the **task**:

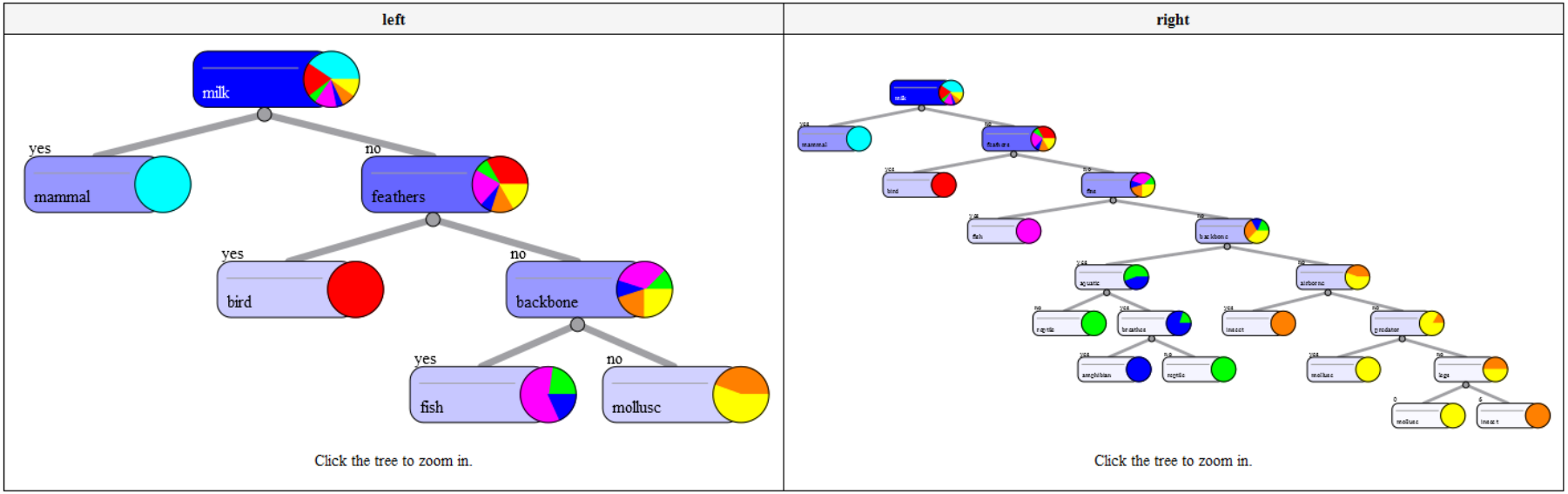
- Very easy - I answered without any problems in less then 7 seconds
- Easy - I found the answer quite quickly and without major problems
- Medium -
- Difficult - I had to think hard and am not sure if I answered correctly
- Very difficult - Despite thinking very hard my answer is likely to be wrong

Next task

Figure B.4: An example of a question in the *discover task*

Compare task

Which tree is **more comprehensible**?



Select statement that best fits your opinion:

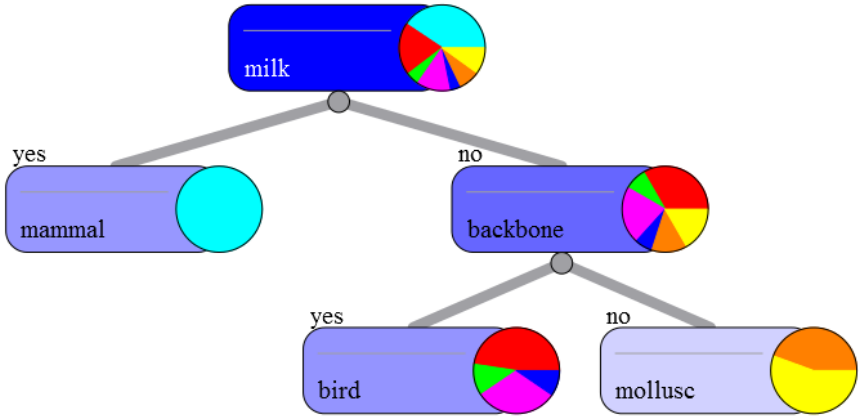
- Tree on the **left is much more comprehensible**: answering questions similar to the ones in this survey about the other tree would be significantly more difficult and would definitely take more time.
- Tree on the **left is more comprehensible**: answering questions similar to the ones in this survey about the other tree would be more difficult or would take more time.
- Tree on the **left is slightly more comprehensible**: although one tree is slightly more comprehensible, answering questions similar to the ones in this survey about the other tree would not be more difficult.
- The trees are equally comprehensible: I don't notice any difference in comprehensibility between the two trees.
- Tree on the **right is slightly more comprehensible**: although one tree is slightly more comprehensible, answering questions similar to the ones in this survey about the other tree would not be more difficult.
- Tree on the **right is more comprehensible**: answering questions similar to the ones in this survey about the other tree would be more difficult or would take more time.
- Tree on the **right is much more comprehensible**: answering questions similar to the ones in this survey about the other tree would be significantly more difficult and would definitely take more time.

Next task

Figure B.5: An example of a question in the *compare task*

Rate task

How comprehensible is the **tree** shown below:



The above **classification tree** is:

- ◉ **Very easily comprehensible**: I can use the knowledge represented by the classification tree as soon as I read it for the first time; I can easily remember it; I can explain it to another person without looking at the figure.
- ○ **Easily comprehensible**: I can use the knowledge represented by the classification tree after studying it for some time; I can remember it with some effort; I can explain it to another person without looking at the figure, but I am not sure that the person will fully understand it.
- ○ **Comprehensible**: Without long study I can use most of the knowledge represented by the classification tree, but need the figure for some details; It would be difficult to remember it; I can explain it to another person if I see the figure while the other person does not see it, but that person is unlikely to fully understand it.
- ○ **Difficult to comprehand**: Without long study I can use only some of the knowledge represented by the classification tree, and need the figure for the rest; It would be very difficult to remember it; I can explain the outline to another person if I see the figure while the other person does not see it.
- ○ **Very difficult to comprehend**: I need the figure to use the knowledge represented by the classification tree; It would be extremely difficult to remember it; I can explain it to another person only if we both see the figure.

Next task

Figure B.6: An example of a question in the *rate task*

Appendix C

The Trees Used in the Survey

All the classification trees used in the *classify*, *explain*, *validate* and *rate* tasks are shown in Figures C.1–C.9. The trees shown in Sections C.1–C.3 were also used in the *compare task* where they were compared with the trees with the same structure but different visualisation properties illustrated in Section 2.3.4.

C.1 Trees with Branching Factor 2

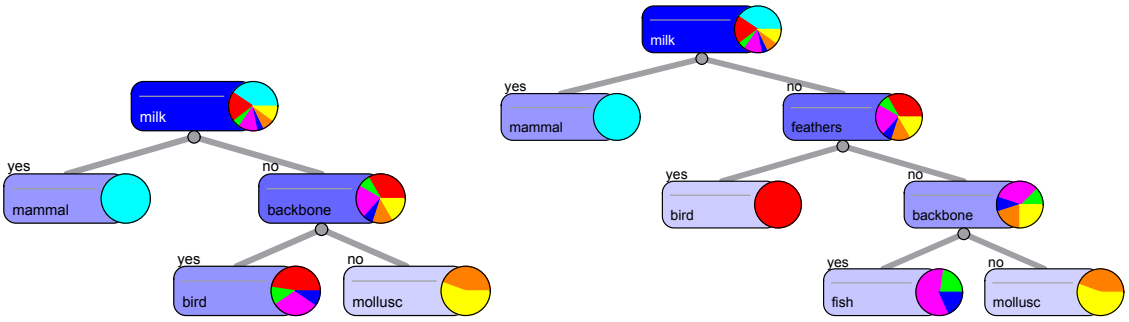


Figure C.1: Trees with 3 (a) or 4 (b) leaves, branching factor 2 and tree depth 2 (a) or 3 (b) – pruned versions of the basic tree (Figure C.2).

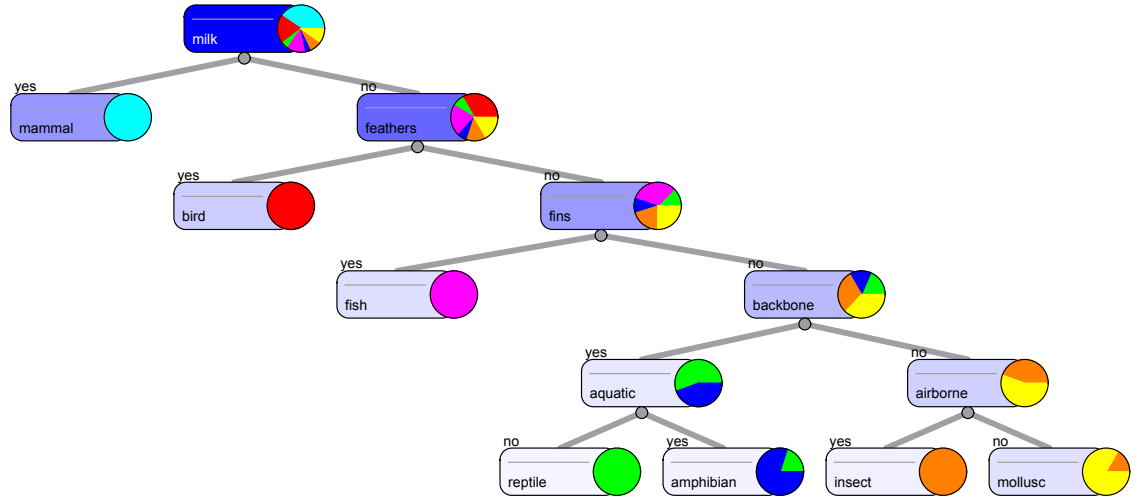


Figure C.2: The basic tree with 7 leaves, branching factor 2 and tree depth 5.

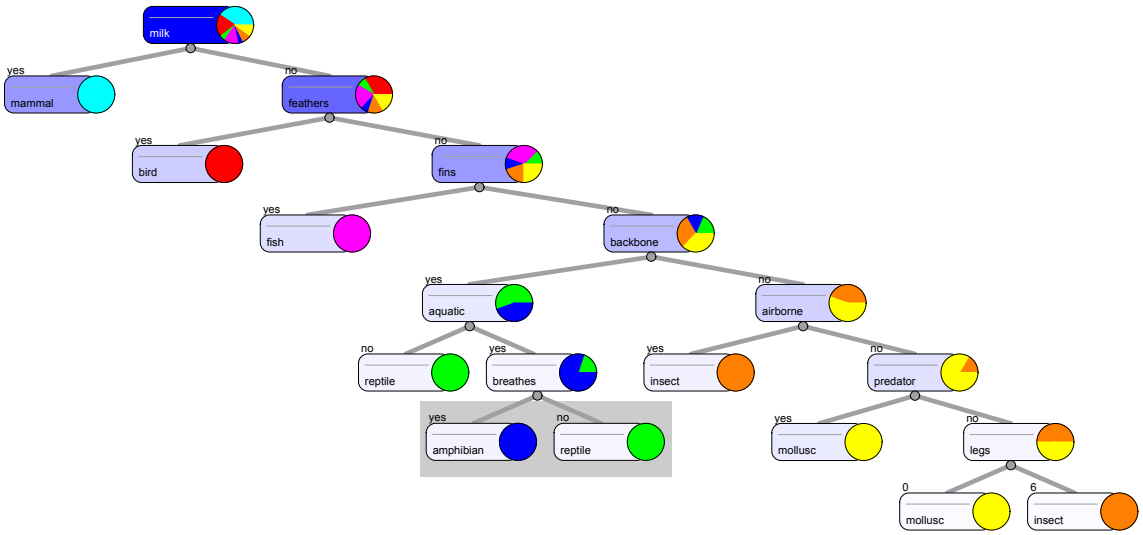


Figure C.3: Trees with 9 (without the subtree marked with grey rectangle) or 10 leaves, branching factor 2 and tree depth 7 – unpruned versions of the basic tree (Figure C.2).

C.2 Trees with Branching Factor 3

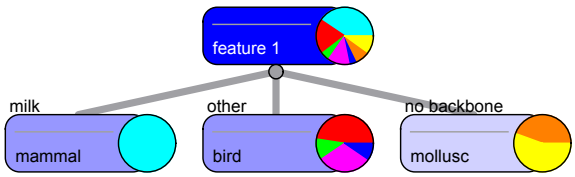


Figure C.4: Tree with 3 leaves, branching factor 3 and tree depth 1 – a version of the tree in Figure C.1a with branching factor 3.

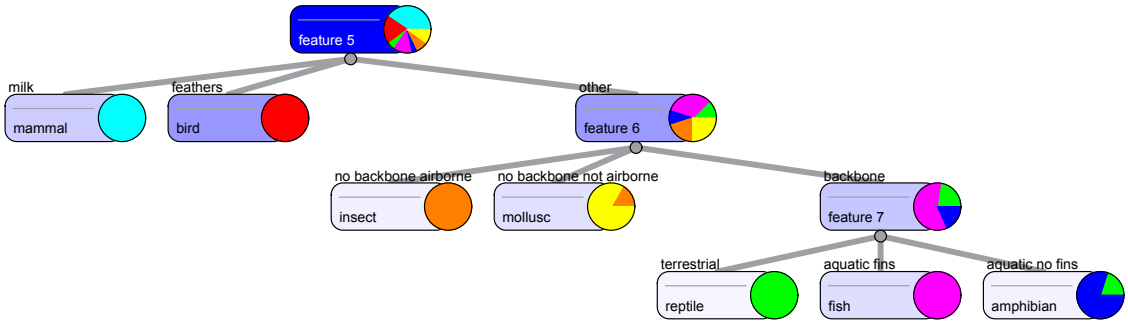


Figure C.5: Tree with 7 leaves, branching factor 3 and tree depth 3 – a version of the tree in Figure C.2 with branching factor 3.

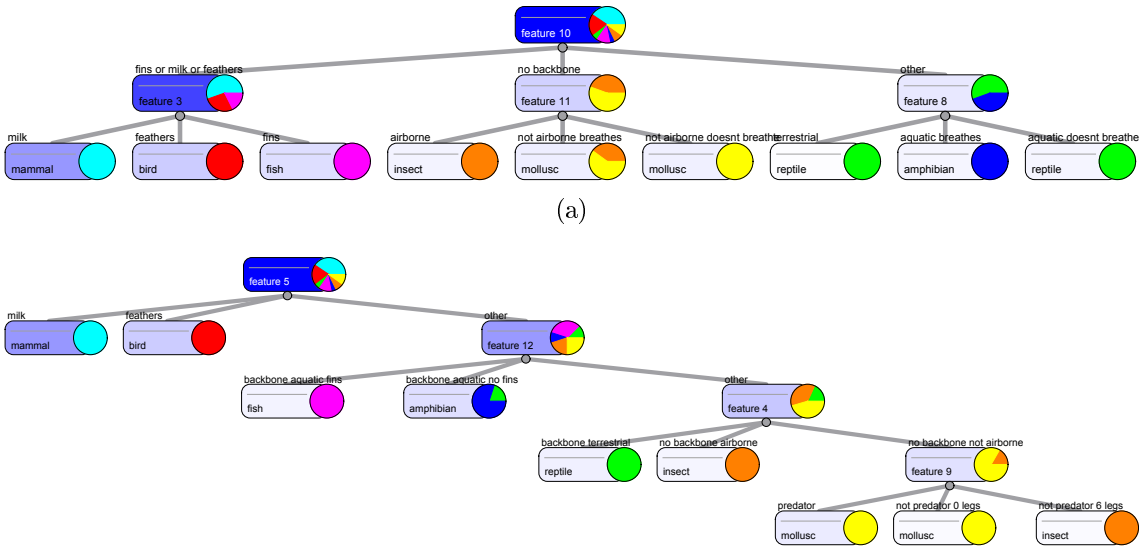


Figure C.6: Trees with 9 leaves, branching factor 3 and tree depth 2 (a) or 4 (b) – two version of the tree in Figure C.3 with branching factor 3.

C.3 Trees with Branching Factor 4

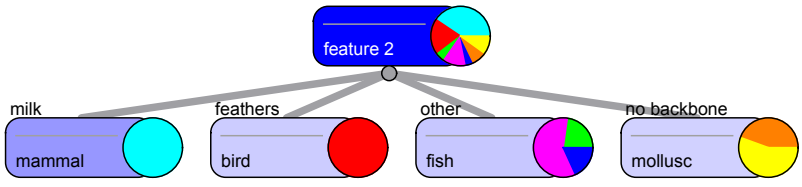


Figure C.7: Tree with 4 leaves, branching factor 4 and tree depth 1 – a version of the tree in Figure C.1b with branching factor 4.

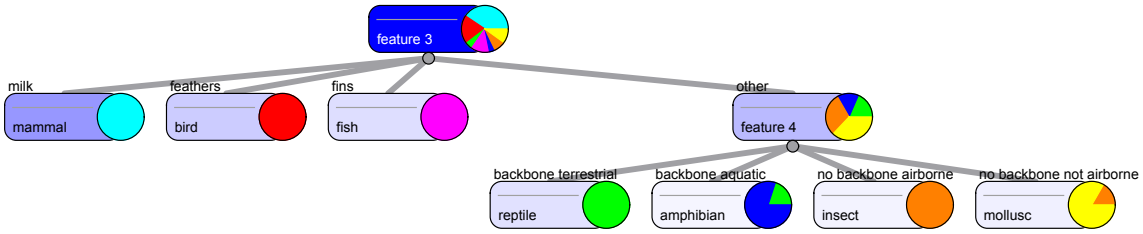


Figure C.8: Tree with 7 leaves, branching factor 4 and tree depth 2 – a version of the basic tree (Figure C.2) with branching factor 4.

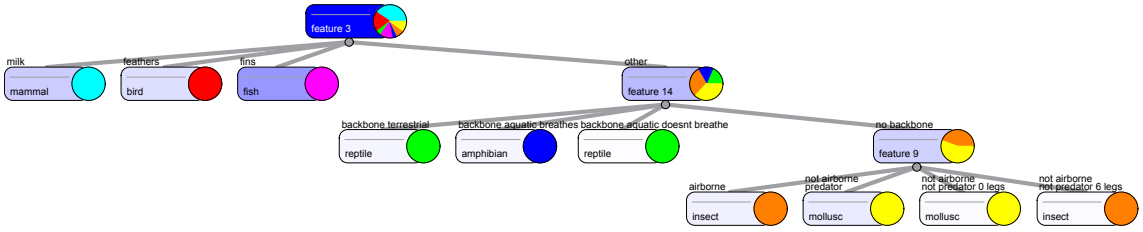


Figure C.9: Tree with 10 leaves, branching factor 4 and tree depth 3 – a version of the tree in Figure C.3 with branching factor 4.

C.4 Trees Used in the *Discover Task*

The *discover task* required special trees (Figures C.10–C.12) described in Section 2.3.3.

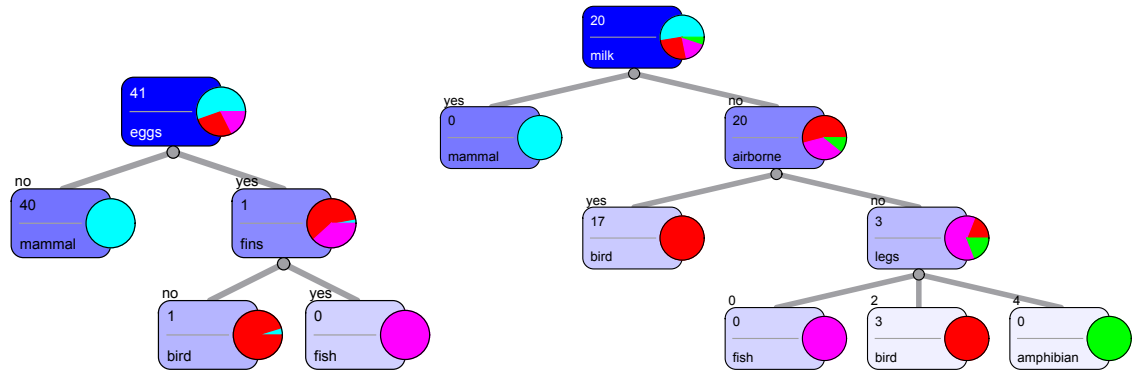


Figure C.10: Small trees showing an unusual property of *mammals* (a) and *birds* (b).

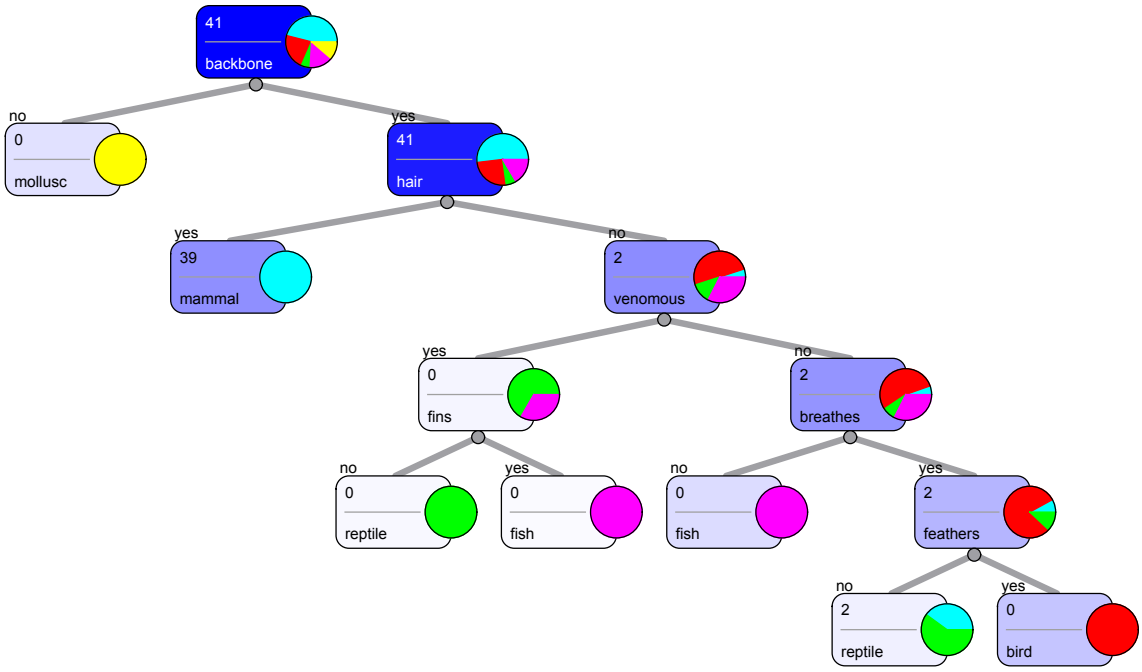


Figure C.11: Medium tree showing unusual properties of *mammals* and *fish* (the numbers in nodes correspond to the number of *mammals* in each node).

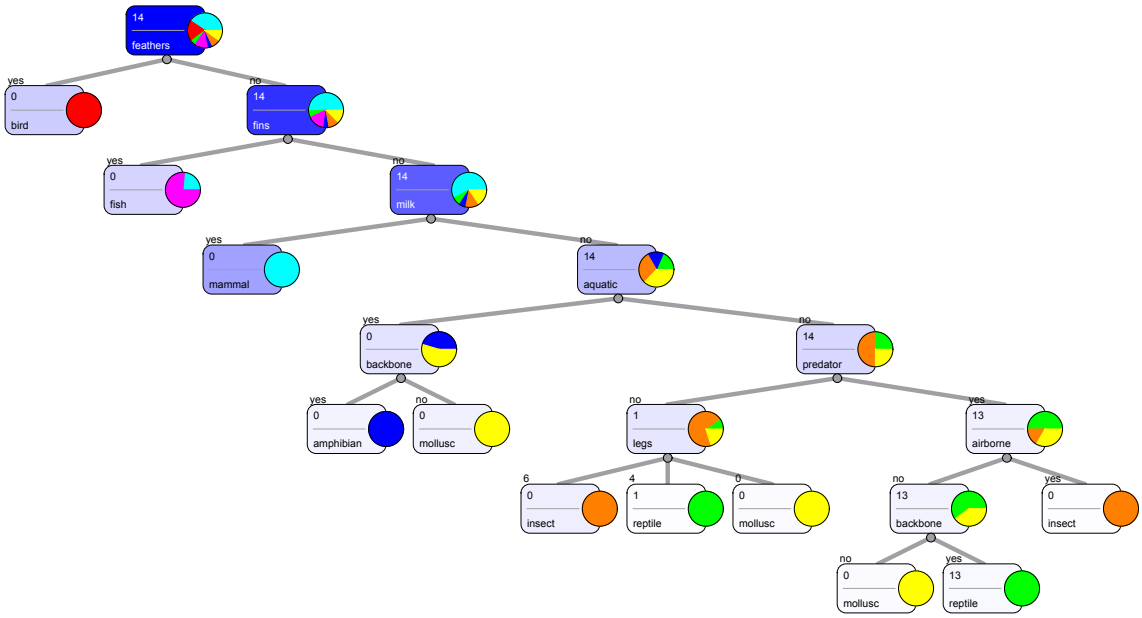


Figure C.12: Big tree showing unusual properties of *mammals*, *insects*, and *reptiles* (numbers in nodes correspond to the number of *reptiles* in each node).

Appendix D

Algorithm Correctness Proofs

D.1 Replacing Only Non-Dominated Leaves Produces the Complete Pareto Set

Theorem D.1 (The MOLHC algorithm returns the complete Pareto set). *The MOLHC algorithm returns the complete Pareto set of hybrid trees even though it investigates only a reduced search space by replacing only the non-dominated regular leaves when generating hybrid trees with an additional black-box leaf.*

Proof. Theorem D.1 can be proven using five lemmas. Lemma D.1 states that investigating all possible combinations of replaced leaves implicitly considers replacing subtrees of the initial tree as well. Lemmas D.2 and D.3 state that it is possible to calculate the accuracy and comprehensibility of a hybrid tree using Eq. (3.9), which enables the dynamic programing approach. Lemma D.4 shows that replacing a single dominated leaf k in a hybrid tree T_i with a black-box leaf produces a dominated hybrid tree T_k. Lemma D.5 shows that replacing additional leaves in the tree T_k produces dominated hybrid trees or trees that will also be generated by replacing the non-dominated leaves in T_i first. Lemmas D.4 and D.5 show that search space can be pruned by not replacing dominated leaves. The combination of the five lemmas proves that the MOLHC algorithm is correct.

Lemma D.1. By considering all possible replacements of leaves in the initial tree with black-box leaves in order to obtain hybrid trees, the replacements of all the subtrees are implicitly considered as well (Figure D.1).

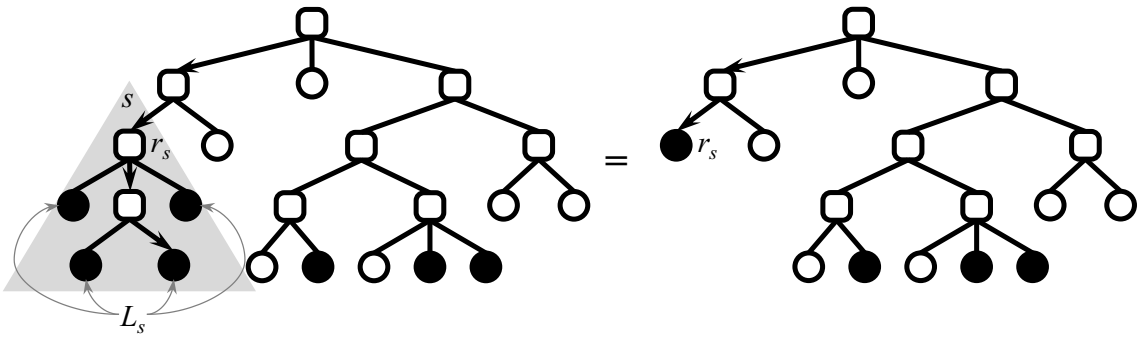


Figure D.1: Lemma D.1 proof illustration: replacing all the leaves L_s of a subtree s with black-box leaves is equivalent to replacing the root of the subtree r_s with a black-box leaf.

Proof. Lemma D.1 holds because replacing all the leaves L_s of a subtree s with black-box leaves is equivalent to replacing the root of the subtree r_s with a black-box leaf (Figure D.1). If an instance belongs to the root r_s of the subtree s, it must belong to one of its leaves L_s; therefore, it will be classified in the same way by a hybrid tree in which the subtree s is replaced with a single black-box leaf or the hybrid tree in which all the leaves L_s of the subtree s are black-box leaves. □

Although the comprehensibility and accuracy of the two trees in Figure D.1 are the same by definition (Eq. (3.1) and Eq. (3.2)), users may prefer the hybrid tree with the subtree s replaced with a black-box leaf because it has a lower number of leaves and nodes. However, this is merely a matter of hybrid tree visualisation and does not affect the correctness of the MOLHC algorithm.

Lemma D.2. The accuracy of a hybrid tree $T_j = (t_{j,1}, t_{j,2}, \ldots, t_{j,n})$ obtained from a hybrid tree $T_i = (t_{i,1}, t_{i,2}, \ldots, t_{i,n})$ by replacing a leaf l with a black-box leaf (i.e. $t_{j,l} = 1$ and $t_{j,k} = t_{i,k}$ for each leaf $k \neq l$) is $a_j = a_i + \delta_{l,\mathrm{a}}$, i.e. equal to the sum of the accuracy a_j of the hybrid tree T_i and the relative difference in accuracy $\delta_{l,\mathrm{a}}$ introduced by replacing the leaf l with the black-box classifier (Figure D.2).

Proof. Define the set of black-box leaves in the hybrid tree T_i as $B_i = \{k : t_{i,k} = 1 \wedge 1 \leq k \leq n\}$ and the set of black-box leaves in the hybrid tree T_j as $B_j = B_i \cup \{l\}$ (Figure D.2). Then it holds:

$$
\begin{gathered}
a_j \equiv \\
(\sum_{k \notin B_j} N_{k,\mathrm{t}} + \sum_{k \in B_j} N_{k,\mathrm{bb}})/N = \\
((\sum_{k \notin B_i} N_{k,\mathrm{t}} - N_{l,\mathrm{t}}) + (\sum_{k \in B_i} N_{k,\mathrm{bb}} + N_{l,\mathrm{bb}}))/N = \\
(\sum_{k \notin B_i} N_{k,\mathrm{t}} + \sum_{k \in B_i} N_{k,\mathrm{bb}})/N + (N_{l,\mathrm{bb}} - N_{l,\mathrm{t}})/N \equiv \\
a_i + \delta_{l,\mathrm{a}}
\end{gathered}
$$

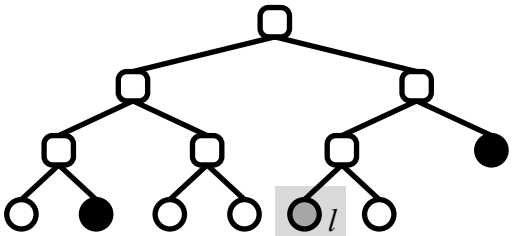
□

Lemma D.3. The comprehensibility of the hybrid tree T_j defined in Lemma D.2 is $c_j = c_i + \delta_{l,\mathrm{c}}$, i.e. equal to the sum of the comprehensibility of the hybrid tree T_i and the relative difference in comprehensibility $\delta_{l,\mathrm{c}}$ introduced by replacing the leaf l with the black-box classifier (Figure D.2).

Proof. Using the sets of black-box leaves B_i and B_j defined in the proof of Lemma D.2 and shown in Figure D.2 it follows:

$$
\begin{gathered}
c_j \equiv \\
(\sum_{k \notin B_j} N_k)/N = \\
(\sum_{k \notin B_i} N_k - N_l)/N = \\
(\sum_{k \notin B_i} N_k)/N + (-N_l/N) \equiv \\
c_i + \delta_{l,\mathrm{c}}
\end{gathered}
$$

□



$$a_i = (N_{1,\mathrm{t}} + N_{2,\mathrm{bb}} + N_{3,\mathrm{t}} + N_{4,\mathrm{t}} + \boldsymbol{N_{l,\mathrm{t}}} + N_{6,\mathrm{t}} + N_{7,\mathrm{bb}}) / N$$
$$a_j = (N_{1,\mathrm{t}} + N_{2,\mathrm{bb}} + N_{3,\mathrm{t}} + N_{4,\mathrm{t}} + \boldsymbol{N_{l,\mathrm{bb}}} + N_{6,\mathrm{t}} + N_{7,\mathrm{bb}}) / N = a_i + \delta_{l,\mathrm{a}}$$
$$c_i = (N_1 + N_3 + N_4 + \boldsymbol{N_{l,\mathrm{t}}} + N_6) / N$$
$$c_j = (N_1 + N_3 + N_4 + N_6) / N = c_i + \delta_{l,\mathrm{c}}$$
$$T_i = (0, 1, 0, 0, \mathbf{0}, 0, 1) \Rightarrow B_i = \{2, 7\}$$
$$T_j = (0, 1, 0, 0, \mathbf{1}, 0, 1) \Rightarrow B_j = \{2, l, 7\}$$

Figure D.2: Lemma D.2 and D.3 proof illustration: Replacing the leaf l with a black-box leaf introduces the same difference in comprehensibility and accuracy regardless of the hybrid tree in which the leaf l was replaced.

The proofs of Lemmas D.2 and D.3 also show that the relative difference in accuracy $\delta_{l,\mathrm{a}}$ and comprehensibility $\delta_{l,\mathrm{c}}$ introduced by replacing the leaf l with a black-box leaf depend only on the total number of instances and the number of correctly classified instances (using either the initial tree or the black-box classifier) in the leaf l and not on the type of other leaves. Therefore, if leaf l is replaced with a black-box leaf it introduces the same difference in comprehensibility and accuracy regardless of the hybrid tree in which the leaf l was replaced.

Lemma D.4. Replacing a single dominated leaf k (among the set of regular leaves in a hybrid tree T_i) in the hybrid tree T_i with a black-box leaf produces a dominated hybrid tree T_k (in the set of all possible hybrid trees obtained from the hybrid tree T_i by replacing a single leaf) as illustrated in Figure D.3.

Proof. Consider a hybrid tree $T_i = (t_{i,1}, t_{i,2}, \ldots, t_{i,n})$ with the accuracy and comprehensibility (a_i, c_i) and the set of regular (not black-box) leaves in the hybrid tree $R_i = \{j : t_{i,j} = 0 \wedge 1 \leq j \leq n\}$. If the set of the regular leaves R_i is empty then all the leaves have already been replaced with black-box leaves. In this case, the MOLHC algorithm will stop because there are no more leaves that can be replaced. This case can thus be omitted from further analysis without any loss of generality, and the set of regular leaves R_i can be assumed not to be empty. It can be partitioned into two sub-sets (with an empty intersection): the set of non-dominated leaves $L_i = \text{non-dom}(R_i)$ and the set of dominated leaves $L_i^c = R_i \setminus L_i$. If the set of dominated leaves L_i^c is empty, no hybrid trees can be obtained by replacing a dominated leaf in Ti. Therefore, this case can also be omitted from further analysis without any loss of generality, and the set of dominated leaves can be assumed not to be empty $L_i^c \neq \{\}$. The set of non-dominated leaves L_i is not empty by definition if the set of regular leaves R_i is not empty (already assumed above). Therefore there must be a non-dominated leaf $l \in L_i$ and a dominated leaf $k \in L_i^c$. By definition leaf l dominates leaf $k : l \preccurlyeq k$. Lemmas D.3 and D.4 state that the quality of hybrid trees T_l and T_k produced by replacing the leaves l and k (respectively) in the hybrid tree T_i are

(see Figure D.3 for an example):

$$(a_l, c_l) = (a_i + \delta_{l,\mathrm{a}}, c_i + \delta_{l,\mathrm{c}})$$
$$(a_k, c_k) = (a_i + \delta_{k,\mathrm{a}}, c_i + \delta_{k,\mathrm{c}})$$

According to the definition of Pareto dominance relation on the set of leaves (Equation (3.7)) there are two cases:

a)

$$\delta_{l,\mathrm{a}} > \delta_{k,\mathrm{a}} \wedge \delta_{l,\mathrm{c}} \geq \delta_{k,\mathrm{c}} \Rightarrow$$
$$a_i + \delta_{l,\mathrm{a}} > a_i + \delta_{k,\mathrm{a}} \wedge c_i + \delta_{l,\mathrm{c}} \geq c_i + \delta_{k,\mathrm{c}} \Rightarrow$$
$$a_l > a_k \wedge c_l \geq c_k \overset{\text{def}}{\Longrightarrow}$$
$$T_l \preccurlyeq T_k$$

b)

$$\delta_{l,\mathrm{a}} \geq \delta_{k,\mathrm{a}} \wedge \delta_{l,\mathrm{c}} > \delta_{k,\mathrm{c}} \Rightarrow$$
$$a_i + \delta_{l,\mathrm{a}} \geq a_i + \delta_{k,\mathrm{a}} \wedge c_i + \delta_{l,\mathrm{c}} > c_i + \delta_{k,\mathrm{c}} \Rightarrow$$
$$a_l \geq a_k \wedge c_l > c_k \overset{\text{def}}{\Longrightarrow}$$
$$T_l \preccurlyeq T_k$$

In both cases the hybrid tree T_k generated by replacing a dominated leaf $k \in L_i^c$ in the hybrid tree T_i is dominated by the hybrid tree T_l generated from the same hybrid tree T_i by replacing a non-dominated leaf $l \in L_i \neq \{\}$. □

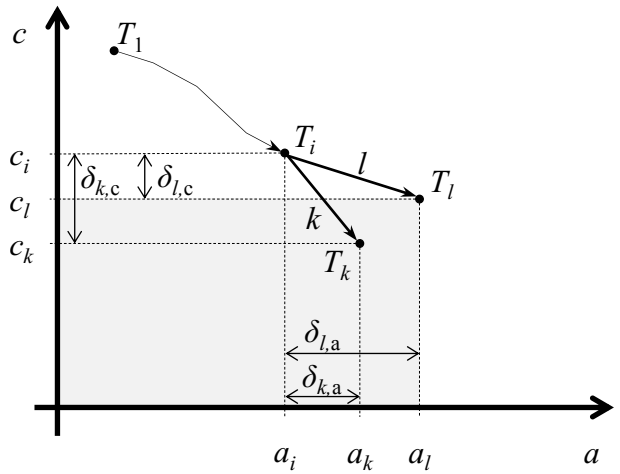


Figure D.3: Lemma D.4 proof illustration: Replacing a dominated leaf k produces a dominated hybrid tree T_k because a hybrid tree T_l that dominates it (grey area) can always be generated from the same initial hybrid tree T_i by replacing a non-dominated leaf l.

Lemma D.5. Replacing additional leaves in the tree T_k (defined in Lemma D.4) produces either i) trees that will also be generated by first replacing the non-dominated leaves in T_i instead of the dominated leaf k or ii) dominated hybrid trees (in the set of all possible hybrid trees obtained from the same initial tree as hybrid tree T_i).

Proof. Take the set of regular leaves R_k in the hybrid tree T_k (i.e. the ones that can be replaced with black-box leaves in the following iterations of the algorithm) defined by Eq. (D.1) and the set of regular leaves R_l in the hybrid tree T_l defined by Eq. (D.2):

$$R_k = \{j : t_{k,j} = 0 \wedge 1 \leq j \leq n\} = R_i \setminus \{k\} \subset R_i \tag{D.1}$$

$$R_l = \{j : t_{l,j} = 0 \wedge 1 \leq j \leq n\} = R_i \setminus \{l\} \subset R_i. \tag{D.2}$$

i) Consider first the case in which the set of the leaves $L \subseteq R_k$ that will be replaced in the dominated hybrid tree T_k to obtain a new hybrid tree T'_k contains the leaf l. Because the order of replacing the leaves from L to produce the new hybrid tree T'_k does not influence the quality of the generated hybrid trees (proven by Lemmas D.2 and D.3), it can be assumed without any loss of generality that the leaf l is the first leaf to be replaced. Notice that the same hybrid tree T'_k can be generated from the hybrid tree T_l by first replacing the leaf k (obtaining the hybrid tree T_{kl}) and then replacing the other leaves from the set $L \setminus \{l\}$ as shown in Figure D.4. This is possible because the same set of leaves is available for replacement with the black-box after replacing the leaf l in the hybrid tree T_k or after replacing the leaf k in the hybrid tree T_l, which follows from Eq. (D.1) and Eq. (D.2): $L \setminus \{l\} \subseteq R_k \setminus \{l\} = R_i \setminus \{k, l\} = R_l \setminus \{k\}$. Any hybrid tree T'_k that can be generated from a dominated hybrid tree T_k by replacing a set of leaves L that contains the leaf l will therefore also be generated from the non-dominated hybrid tree T_l by the MOLHC algorithm — if T'_k is not dominated, otherwise a tree that dominates it will be generated (Lemma D.4).

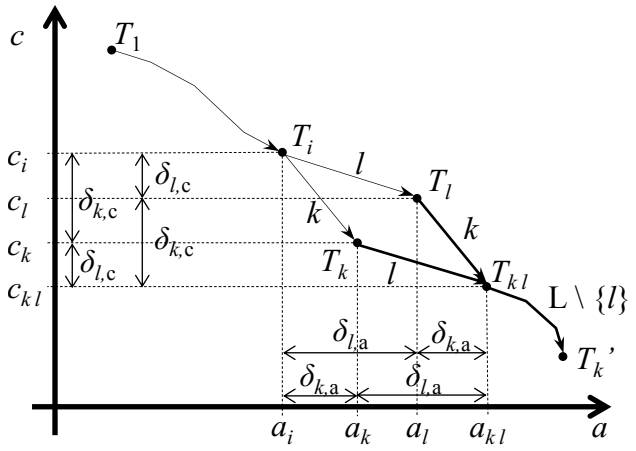


Figure D.4: Lemma D.5 proof illustration (part i): The same hybrid trees can be generated either by replacing a set of leaves L containing the leaf l in a dominated hybrid tree T_k (obtained by replacing a dominated leaf k in T_i) or by replacing the leaf k and the set of leaves $L \setminus \{l\}$ in the non-dominated hybrid tree T_l.

ii) Finally, consider the case where the set of leaves L that will be replaced in the dominated hybrid tree T_k to produce a new hybrid tree T'_k does not contain the leaf l, i.e. $L \subseteq R_k \setminus \{l\}$. Define the total relative difference in accuracy δ_a and the total relative difference in comprehensibility δ_c introduced by replacing the set of leaves L with Eq. (D.3) and Eq. (D.4) respectively:

$$\delta_\mathrm{a} = \sum_{i \in L} \delta_{i,\mathrm{a}} \tag{D.3}$$

$$\delta_{\mathrm{c}} = \sum_{i \in L} \delta_{i,\mathrm{c}} \tag{D.4}$$

Based on Eq. (D.3) and Eq. (D.4) the quality of the hybrid tree T'_k obtained from the hybrid tree T_k by replacing the set of leaves $L \subseteq R_k \setminus \{l\}$ is $(a'_k, c'_k) = (a_k + \delta_{\mathrm{a}}, c_k + \delta_{\mathrm{c}})$

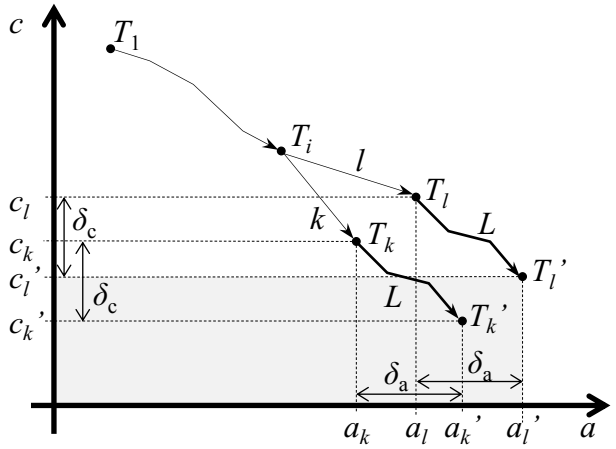


Figure D.5: Lemma D.5 proof illustration (part ii): Replacing a set of leaves L in a dominated hybrid tree T_k (obtained by replacing a dominated leaf k in T_i) produces a hybrid tree T'_k that is dominated (grey area) by the hybrid tree T'_l generated from the non-dominated hybrid tree T_l (obtained by replacing a non-dominated leaf l in T_i) by replacing the same set of leaves L.

Figure D.5 illustrates that the tree T'_l with quality (a'_l, c'_l) can be generated from the non-dominated hybrid tree T_l by replacing the same set of leaves L, which is always possible because L is a subset of regular leaves R_l in the hybrid tree T_l: $L \subseteq R_k \setminus \{l\} = R_i \setminus \{k, l\} \subseteq R_i \setminus \{l\} = R_l$ (following Eq. (D.1) and Eq. (D.2)).

The hybrid tree T'_l dominates the tree T'_k as shown below. According to the definition of Pareto dominance relation on the set of hybrid trees (Eq (3.3)) two cases originating from the fact $T_l \preccurlyeq T_k$ must be considered:

a)

$$\begin{gathered} a_l > a_k \wedge c_l \geq c_k \Rightarrow \\ a_l + \delta_{\mathrm{a}} > a_k + \delta_{\mathrm{a}} \wedge c_l + \delta_{\mathrm{c}} \geq c_k + \delta_{\mathrm{c}} \Rightarrow \\ a'_l > a'_k \wedge c'_l \geq c'_k \overset{\text{def}}{\Longrightarrow} \\ T'_l \preccurlyeq T'_k \end{gathered}$$

b)

$$\begin{gathered} a_l \geq a_k \wedge c_l > c_k \Rightarrow \\ a_l + \delta_{\mathrm{a}} \geq a_k + \delta_{\mathrm{a}} \wedge c_l + \delta_{\mathrm{c}} > c_k + \delta_{\mathrm{c}} \Rightarrow \\ a'_l \geq a'_k \wedge c'_l > c'_k \overset{\text{def}}{\Longrightarrow} \\ T'_l \preccurlyeq T'_k \end{gathered}$$

Therefore replacing any set of leaves in the dominated hybrid tree T_k produces hybrid trees that are dominated (i.e. worse) or at best equal to the (non-dominated) hybrid trees obtained by the MOLHC algorithm from the non-dominated hybrid tree T_l. □

The MOLHC algorithm is therefore guaranteed to find the complete Pareto set of hybrid trees because by replacing only the non-dominated leaves in each step of the algorithm none of the non-dominated hybrid trees (in the set of all possible hybrid trees) will be missed. Hence, the Theorem D.1 is proven. □

D.2 Some Generated Trees May Be Dominated

Theorem D.2 (The MOLHC algorithm may generate some dominates trees). *Replacing only non-dominated leaves in the MOLHC algorithm does not guarantee that all the generated trees will be non-dominated.*

Proof. Theorem D.2 can be proven with an example shown in Figure D.6. Consider a case with three leaves $l_x = (\delta_{x,\mathrm{a}}, \delta_{x,\mathrm{c}})$, $l_y = (\delta_{y,\mathrm{a}}, \delta_{y,\mathrm{c}})$, and $l_z = (\delta_{z,\mathrm{a}}, \delta_{z,\mathrm{c}})$, which are non-dominated and pairwise incomparable (i.e. none of them dominates any other of them), and have the following property: $\delta_{x,\mathrm{a}} + \delta_{y,\mathrm{a}} > \delta_{z,\mathrm{a}} \wedge \delta_{x,\mathrm{c}} + \delta_{y,\mathrm{c}} > \delta_{z,\mathrm{c}}$ Three hybrid trees T_x, T_y, and T_z will be generated by replacing each of the leaves x, y, and z (respectively) with a black-box leaf in the first iteration of the main loop of the MOLHC algorithm. In the second iteration, hybrid tree $T_{x\,y}$ will be generated by replacing leaf x in hybrid tree T_y or leaf y in hybrid tree T_x. Notice that tree $T_{x\,y} = (a_1 + \delta_{x,\mathrm{a}} + \delta_{y,\mathrm{a}}, 1 + \delta_{x,\mathrm{c}} + \delta_{y,\mathrm{c}})$ dominates hybrid tree $T_z = (a_1 + \delta_{z,\mathrm{a}}, 1 + \delta_{z,\mathrm{c}})$ as shown below:

$$\begin{gathered}
\delta_{x,\mathrm{a}} + \delta_{y,\mathrm{a}} > \delta_{z,\mathrm{a}} \wedge \delta_{x,\mathrm{c}} + \delta_{y,\mathrm{c}} > \delta_{z,\mathrm{c}} \Rightarrow \\
(a_1 + \delta_{x,\mathrm{a}} + \delta_{y,\mathrm{a}} > a_1 + \delta_{z,\mathrm{a}}) \wedge (1 + \delta_{x,\mathrm{c}} + \delta_{y,\mathrm{c}} > 1 + \delta_{z,\mathrm{c}}) \Rightarrow \\
a_{x\,y} > a_z \wedge c_{x\,y} > c_z \overset{\text{def}}{\Longrightarrow} \\
T_{x\,y} \preccurlyeq T_z
\end{gathered}$$

□

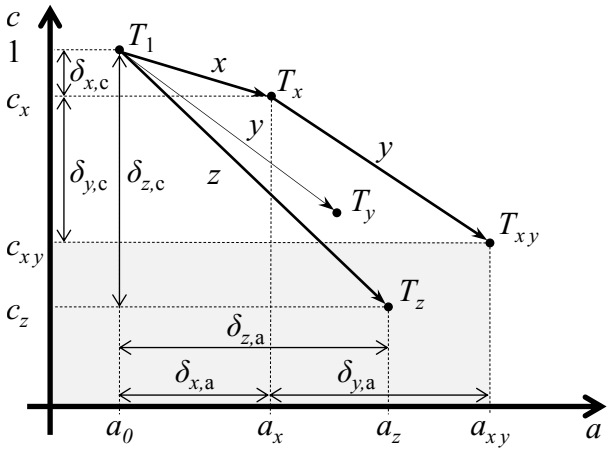


Figure D.6: The MOLHC algorithm may generate some dominated trees, e.g., T_z is dominated by $T_{x\,y}$.

Theorem D.2 states that replacing only non-dominated leaves may generated a set of hybrid trees that contains some dominated trees. Therefore, the MOLHC algorithm must select the non-dominated hybrid trees among the set of all generated trees in the last step in order to find the Pareto set of hybrid trees.

D.3 Calculating Non-Dominated Leaves

Theorem D.3 (Non-dominated leaves can be computed using dynamic algorithm). *Consider the hybrid tree T_j generated from a hybrid tree T_i by replacing a leaf l with the black-box classifier as defined in Lemma D.2. Then the set of non-dominated regular leaves L_j in the hybrid tree T_j can be computed from the set of non-dominated regular leaves L_i in the hybrid tree T_i, and the set D_l (Eq. (3.8)), which consists of the leaves that are directly dominated by the leaf l, using the following equation $L_j = (L_i \setminus \{l\}) \cup \{k : k \in D_l \wedge \nexists m \in L_i \setminus \{l\} : m \preccurlyeq k\}$ as shown in Figure D.7.*

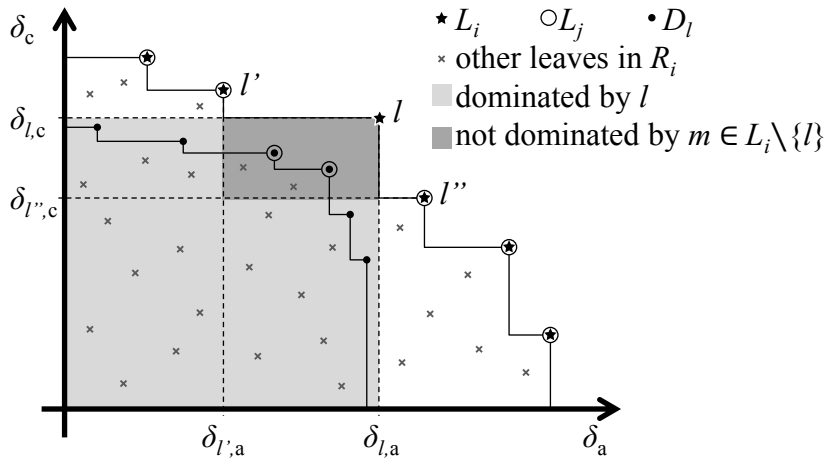


Figure D.7: The set of non-dominated regular leaves L_j (encircled) contains all the leaves from L_i (stars), except the leaf l, and the leaves that are directly dominated by l (black circles) but are not dominated by leaves from $L_i \setminus \{l\}$; those are leaves $k \in D_l$ so that $\delta_{k,a} > \delta_{l',a}$ and $\delta_{k,c} > \delta_{l'',c}$ (dark background).

Proof. Define the set of regular (not black-box) leaves in the hybrid tree T_i as $R_i = \{k : t_{i,k} = 0 \wedge 1 \leq k \leq n\}$ and the set of regular leaves in the hybrid tree T_j as $R_j = R_i \setminus \{l\}$. The proof of Theorem D.3 is based on four lemmas. Lemma D.6 states that the leaf l is not a member of L_j because it belongs to the set of black-box leaves and not to the set of regular leaves R_j. Lemma D.7 states that all the other leaves form the set $L_i \setminus \{l\}$ are members of the set L_j because removing a leaf (e.g., l) from a set cannot cause a non-dominated leaf in the original set to become a dominated leaf in the reduced set. Lemma D.8 states that the only additional leaves from R_j (besides the ones in L_i) that might be in the set L_j are the ones that are dominated by l. Finally, Lemma D.9 states that among those leaves only the ones that are in the set D_l (the set of leaves directly dominated by the leaf l) and that are not dominated by any leaf in $L_i \setminus \{l\}$ belong to the set L_j. The leaves that are dominated by l but are not directly dominated by it are by definition dominated by some leaf from D_l and therefore cannot be non-dominated in $R_j \supseteq D_l$. The leaves that are not dominated by l but are not members of L_i must be dominated by a leaf in $L_i \setminus \{l\}$ and are therefore also dominated in $R_j \supseteq L_i \setminus \{l\}$. The dynamic programming equation $L_j = \text{non-dom}(L_i \cup D_l \setminus \{l\}) = L_i \setminus \{l\} \cup \{k \in D_l \wedge \nexists m \in L_i \setminus \{l\} : m \preccurlyeq k\}$ for calculating the set of non-dominated regular leaves L_j in the hybrid tree T_j thus follows from the four lemmas proven below.

Lemma D.6. The leaf l (defined in Theorem D.3) that is replaced with a black-box leaf in the hybrid tree T_i in order to obtain the hybrid tree T_j is not a member of the set of non-dominated leaves L_j in the hybrid tree T_j, i.e. $l \notin L_j$.

Proof. The leaf l is a black-box leaf in the hybrid tree T_j, therefore it is not the member of R_j (the set of regular leaves in the hybrid tree T_j) and therefore (by definition of non-dominated set) cannot be a member of $L_j \equiv \text{non-dom}(R_j)$. □

Lemma D.7. All the leaves from the set L_i except the leaf l are also members of the set L_j, where L_i, L_j and l are as defined in Theorem D.3, i.e. $n \in L_i \setminus \{l\} \Rightarrow n \in L_j$.

Proof. Lemma D.7 is proven as follows. Any leaf n that is a member of the set $L_i \setminus \{l\}$ must also be a member of $L_i \equiv \text{non-dom}(R_i)$. Therefore, by the definition of non-dominated set (Eq. (3.4)) the following statement holds: $n \in R_i \wedge \nexists m \in R_i : m \preccurlyeq n$. Because leaf n is not the leaf l, the statement can be rewritten as $n \in R_i \setminus \{l\} = R_j \wedge \nexists m \in R_i : m \preccurlyeq n$. If there are no leaves m in the set R_i such that they dominate the leaf n, there certainly can be no such leaves in the set $R_i \setminus \{l\} = R_j$, therefore $n \in R_j \wedge \nexists m \in R_j : m \preccurlyeq n$, which by definition of non-dominated set gives $n \in \text{non-dom}(R_j) \equiv L_j$. □

Lemma D.8. Besides the leaves from L_i, only the regular leaves in hybrid tree T_j that are dominated by l can be members of L_j, where L_i, L_j, T_j and l are as defined in Theorem D.3, i.e. $k \in L_j \setminus L_i \Rightarrow l \preccurlyeq k$.

Proof. Lemma D.8 is proven as follows. Because $k \in L_j \setminus L_i$ we know that $k \in L_j \subseteq R_j \subset R_i$ and that $k \notin L_i$. From $k \in L_j \equiv \text{non-dom}(R_j)$ it follows that $\nexists m \in R_j : m \preccurlyeq k$ (by definition of non-dominated set) and therefore $\nexists m \in L_i \setminus \{l\} : m \preccurlyeq k$ (because $L_i \setminus \{l\} \subseteq L_j \subseteq R_j$). And from $k \notin L_i \equiv \text{non-dom}(R_i)$ it follows that $k \notin R_i \vee \exists m \in R_i : m \preccurlyeq k$ (by definition of non-dominated set), which by the fact that $k \in R_i$ reduces to $\exists m \in R_i : m \preccurlyeq k$. From the transitivity of Pareto dominance relation it follows $\exists m \in \text{non-dom}(R_i) \equiv L_i : m \preccurlyeq k$. The only leaf that satisfies both of the above statements $\nexists m \in L_i \setminus \{l\} : m \preccurlyeq k$ and $\exists m \in L_i : m \preccurlyeq k$ is the leaf l. The leaf l dominates all the leaves from $L_j \setminus L_i$ (as follows from the second statement), i.e. $l \preccurlyeq k$. □

Lemma D.9. Among the leaves that are dominated by the leaf l, only the ones that are directly dominated by l and are not dominated by any member of $L_i \setminus \{l\}$ are in L_j, where L_i, L_j, and l are as defined in Theorem D.3, i.e. $\{k : k \in L_j \wedge l \preccurlyeq k\} = \{k : k \in D_l \wedge \nexists m \in L_i \setminus \{l\} : m \preccurlyeq k\}$.

Proof. To prove Lemma D.9, observe the subsets of R_j shown in Figure D.8 and the properties of the corresponding leaves. The leaf l is not a member of R_j as shown in the proof of Lemma D.6. The set of leaves $\{k : l \preccurlyeq k\}$ that are dominated by the leaf l is a subset of the set of regular leaves R_j in the hybrid tree T_j (i.e. none of the leaves dominated by l is a black-box leaf), because they could not have been replaced with black-box leaves during the execution of the MOLHC algorithm, which replaces only the non-dominated leaves and generated the hybrid tree T_j from the hybrid tree T_i by replacing only the leaf l, which dominates all of the leaves in the set $\{k : l \preccurlyeq k\}$. The set of leaves D_l directly dominated by the leaf l is by Definition 3.10 a subset of $\{k : l \preccurlyeq k\}$. The set of non-dominated regular leaves L_j in the hybrid tree T_j is by definition a subset of regular leaves R_j in the hybrid tree T_j. Lemma D.7 states that $L_i \setminus \{l\}$ is a subset of L_j.

Furthermore, Lemma D.8 states that $L_j \setminus L_i \subseteq \{k : l \preccurlyeq k\}$ and therefore $L_j \setminus L_i \setminus \{k : l \preccurlyeq k\} = \{\}$. Note that among all the leaves dominated by l only the leaves from D_l (the ones that are directly dominated by l) can be in L_j, i.e. $(L_j \cap \{k : l \preccurlyeq k\}) \setminus D_l = \{\}$. This follows from the fact that for every leaf n that is dominated by the leaf l but is not directly dominated by l (i.e. $n \in \{k : l \preccurlyeq k\} \setminus D_l$) there exist a leaf m directly dominated by l such that m dominates n (i.e. $\exists m \in D_l : m \preccurlyeq n$). Because the leaf n is dominated by a leaf from D_l, which is a subset of R_j, it is dominated in the set R_j as well ($\exists m \in R_j : m \preccurlyeq n$) and hence cannot be in L_j, i.e. $n \in \{k : l \preccurlyeq k\}) \setminus D_l \Rightarrow n \notin \text{non-dom}(R_j) \equiv L_j$.

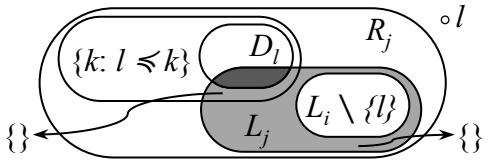


Figure D.8: The relations between subsets of regular leaves in a hybrid tree T_j: light grey subsets are empty, the dark grey subset contains all the leaves that are non-dominated in the new hybrid tree T_j but are not non-dominated in the old hybrid tree T_i.

Note that by definition of the set D_l none of the leaves in the set is dominated by any other leaf in the set (they are pairwise incomparable), therefore it suffices to check whether a leaf from D_l is not dominated by the leaves from $L_i \setminus \{l\}$ in order to check whether it belongs to L_j. None of the leaves from D_l can dominate any leaf from L_i because $n, l \in L_i \wedge k \in D_l \Rightarrow l \not\preccurlyeq n \wedge l \preccurlyeq k \Rightarrow k \not\preccurlyeq n$. Therefore there is no need to check whether leaves from D_l dominate the leaves from $L_i \setminus \{l\}$ — the same follows from Lemma D.7. □

The above proofs show that all the leaves form L_i except the leaf l are in L_j (Lemmas D.6 and D.7) and that any other leaf from the set L_j must also be in the set D_l (Lemmas D.8 and D.9). The Theorem D.3 stating that $L_j = L_i \setminus \{l\} \cup \{k : k \in D_l \wedge \nexists m \in L_i \setminus l : m \preccurlyeq k\}$ is therefore proven by the four lemmas (see Figure D.7 and D.8 for an illustration). □

The proof of Theorem D.3 is valid in general; furthermore an additional property (Theorem D.4) for the specific case of two-objective optimisation where the goal is to maximize both objectives is proven below.

Theorem D.4 (Computing non-dominated leaves efficiently). *The set of non-dominated regular leaves L_j in the hybrid trees T_j (defined in the Theorem D.3) according to the objectives to maximize the accuracy and comprehensibility can be computed using the Eq.* (D.5) *(see an example in Figure D.7):*

$$L_j = L_i \setminus \{l\} \cup \left\{k : k \in D_l \wedge \delta_{k,\mathrm{a}} > \delta_{l',\mathrm{a}} \wedge \delta_{k,\mathrm{c}} > \delta_{l'',\mathrm{c}}\right\} \tag{D.5}$$

Where the indices of the leaves in L_i and D_l (defined in Theorem D.3) are ordered according to the ascending relative difference in accuracy $\delta_{i,\mathrm{a}}$ (which is the same as ordering according to the descending relative difference in comprehensibility $\delta_{i,\mathrm{c}}$) and the leaf l (defined in Theorem D.3) is immediately preceded by the leaf l' and immediately followed by the leaf l'' in the given order (see an example in Figure D.7). If the leaf l' does not exist (i.e. the leaf l is the first one in the order of the leaves in L_i) assign $\delta_{l',\mathrm{a}} = 0$ and similarly if the leaf l'' does not exist (i.e. the leaf l is the last one in the order of the leaves in L_i) assign $\delta_{l'',\mathrm{c}} = -\infty$.

Proof. From the fact that all the leaves in D_l are dominated by the leaf l it follows:

$$k \in D_l \Rightarrow \delta_{k,\mathrm{a}} \leq \delta_{l,\mathrm{a}} \wedge \delta_{k,\mathrm{c}} \leq \delta_{l,\mathrm{c}} \tag{D.6}$$

Because the leaves in L_i are pairwise non-dominated (are incomparable) and ordered, inequalities D.7 and D.8 follow:

$$\delta_{l,\mathrm{a}} \leq \delta_{l'',\mathrm{a}} \tag{D.7}$$

$$\delta_{l,\mathrm{c}} \leq \delta_{l',\mathrm{c}} \tag{D.8}$$

And because all the leaves $k \in L_j \cap D_l$ must be non-dominated and different from l' and l'' it holds that:

$$\delta_{k,\mathrm{a}} > \delta_{l'',\mathrm{a}} \vee \delta_{k,\mathrm{c}} > \delta_{l'',\mathrm{c}} \text{ for } k \in L_j \cap D_l \tag{D.9}$$

$$\delta_{k,\mathrm{a}} > \delta_{l',\mathrm{a}} \vee \delta_{k,\mathrm{c}} > \delta_{l',\mathrm{c}} \text{ for } k \in L_j \cap D_l \tag{D.10}$$

Therefore, based on Eq. D.6–D.10, it follows that:

$$\forall k \in D_l \cap L_j : (\delta_{k,\mathrm{a}} \leq \delta_{l,\mathrm{a}}) \wedge (\delta_{l,\mathrm{a}} \leq \delta_{l'',\mathrm{a}}) \wedge (\delta_{k,\mathrm{a}} > \delta_{l'',\mathrm{a}} \vee \delta_{k,\mathrm{c}} > \delta_{l'',\mathrm{c}}) \Rightarrow \delta_{k,\mathrm{c}} > \delta_{l'',\mathrm{c}} \tag{D.11}$$

$$\forall k \in D_l \cap L_j : (\delta_{k,\mathrm{c}} \leq \delta_{l,\mathrm{c}}) \wedge (\delta_{l,\mathrm{c}} \leq \delta_{l',\mathrm{c}}) \wedge (\delta_{k,\mathrm{a}} > \delta_{l',\mathrm{a}} \vee \delta_{k,\mathrm{c}} > \delta_{l',\mathrm{c}}) \Rightarrow \delta_{k,\mathrm{a}} > \delta_{l',\mathrm{a}} \tag{D.12}$$

The conditions D.11 and D.12 are sufficient to determine if a leaf $k \in D_l$ belongs to L_j because if a leaf k is not dominated by l' and l'' it is not dominated by any other leaf from the set $L_i \setminus \{l\}$. This is proven by partitioning the set $L_i \setminus \{l\}$ into two subsets $P = \{p \in L_i : \delta_{l',\mathrm{a}} \geq \delta_{p,\mathrm{a}}\}$ and $Q = \{q \in L_i : \delta_{l'',\mathrm{c}} \geq \delta_{q,\mathrm{c}}\}$ so that $P \cup \{l', l''\} \cup Q = L_i \setminus \{l\}$ and observing that a leaf $p \in P$ cannot dominate the leaf k because it is not dominated by l' and that a leaf $q \in Q$ cannot dominate the leaf k because it is not dominated by l'':

$$l' \not\preccurlyeq k \wedge p \in P \Rightarrow \delta_{k,\mathrm{a}} > \delta_{l',\mathrm{a}} \wedge \delta_{l',\mathrm{a}} \geq \delta_{p,\mathrm{a}} \Rightarrow \delta_{k,\mathrm{a}} > \delta_{p,\mathrm{a}} \Rightarrow p \not\preccurlyeq k$$
$$l'' \not\preccurlyeq k \wedge q \in Q \Rightarrow \delta_{k,\mathrm{c}} > \delta_{l'',\mathrm{c}} \wedge \delta_{l'',\mathrm{c}} \geq \delta_{q,\mathrm{c}} \Rightarrow \delta_{k,\mathrm{c}} > \delta_{q,\mathrm{c}} \Rightarrow q \not\preccurlyeq k$$

□

Theorem D.4 states that the set L_j contains exactly the leaves from $Li \setminus \{l\}$ and the leaves k from the set D_l with the relative difference in accuracy $\delta_{k,\mathrm{a}} > \delta_{l',\mathrm{a}}$ and relative difference in comprehensibility $\delta_{k,\mathrm{c}} > \delta_{l'',\mathrm{c}}$ if the leaves in L_i are ordered. However, one must also consider the case where the set L_i contains a leaf l_2 with the same relative difference in accuracy and comprehensibility as the leaf l, which makes the ordering of leaves in L_i ill-defined. In such case the set L_j is equal to $L_i \setminus \{l\}$ which trivially follows from Theorem D.3 and the fact that the leaf l_2 dominates exactly the same leaves as the leaf l, so it dominates all the leaves in D_l.

Theorem D.4 enables calculating the set L_j in $\mathrm{O}(m)$ time where m is the number of leaves in D_l, which is less than the number of all the leaves in the initial classification tree that are considered for replacing with black-box leaves, using the following algorithm. Start with an empty set L_j. Take the first leaf k in the ordered set of leaves D_l that are directly dominated by the leaf l. If the relative difference in accuracy $\delta_{k,\mathrm{a}}$ of the leaf k is lower or equal to $\delta_{l',\mathrm{a}}$ mark the leaf k as dominated and consider the next leaf from the set D_l. As soon as a leaf with $\delta_{k,\mathrm{a}} > \delta_{l',\mathrm{a}}$ is found, start comparing the relative difference in comprehensibility $\delta_{k,\mathrm{c}}$ of the considered leaves k from D_l with $\delta_{l'',\mathrm{c}}$ instead of comparing $\delta_{k,\mathrm{a}}$. If $\delta_{k,\mathrm{c}} > \delta_{l'',\mathrm{c}}$ add the leaf k at the end of the ordered set L_j and consider the next leaf form D_l, otherwise stop comparing the leaves from D_l. Make a copy of the ordered set L_i and replace the leaf l with the ordered set L_j; the result is the set L_j of non-dominated leaves in the hybrid tree T_j.

References

Aha, D. W., Kibler, D., & Albert, M. K. (1991, January). Instance-based learning algorithms. *Machine Learning*, *6*(1), 37–66. doi:10.1023/A:1022689900470

Allahyari, H. & Lavesson, N. (2011). User-oriented assessment of classification model understandability. *11th Scandinavian Conference on Artificial Intelligence*.

Aranda, J., Ernst, N., Horkoff, J., & Easterbrook, S. (2007). A framework for empirical evaluation of model comprehensibility. In *Proceedings of the international workshop on modeling in software engineering* (pp. 7–12). MISE '07. Washington, DC, USA: IEEE Computer Society. doi:10.1109/MISE.2007.2

Arun Kumar, M. & Gopal, M. (2010, December). A hybrid SVM based decision tree. *Pattern Recognition*, *43*(12), 3977–3987. doi:10.1016/j.patcog.2010.06.010

Bache, K. & Lichman, M. (2013). UCI machine learning repository. Retrieved from http://archive.ics.uci.edu/ml

Benbasat, I. & Taylor, R. N. (1982, July). Behavioral aspects of information processing for the design of management information systems. *Systems, Man and Cybernetics, IEEE Transactions on*, *12*(4), 439–450. doi:10.1109/TSMC.1982.4308848

Beume, N., Naujoks, B., & Emmerich, M. (2007). SMS-EMOA: Multiobjective selection based on dominated hypervolume. *European Journal of Operational Research*, *181*(3), 1653–1669. doi:10.1016/j.ejor.2006.08.008

Börzsönyi, S., Kossmann, D., & Stocker, K. (2001). The skyline operator. In *Proceedings of the 17th international conference on data engineering* (pp. 421–430). Heidelberg, Germany: IEEE Computer Society.

Box, G. E. P., Hunter, J. S., & Hunter, W. G. (2005). *Statistics for experimenters: design, innovation, and discovery, 2nd edition*. John Wiley & Sons, Inc.

Campbell, D. J. (1988). Task complexity: a review and analysis. *The Academy of Management Review*, *13*(1), 40–52.

Carvalho, D. R., Freitas, A. A., & Ebecken, N. F. F. (2003, December). A critical review of rule surprisingness measures. In N. Ebecken, C. Brebbia, & A. Zanasi (Eds.), *Proceedings of data mining iv - international conference on data mining* (pp. 182–196). WIT Press.

Chakrabarti, S., Ester, M., Fayyad, U., Gehrke, J., Han, J., Morishita, S., ... Wang, W. (2016). Data mining curriculum: a proposal (version 1.0). Retrieved June 27, 2016, from http://www.sigkdd.org/curriculum/CURMay06.pdf

Chapman, P., Clinton, J., Kerber, R., Khabaza, T., Reinartz, T., Shearer, C., & Wirth, R. (2000, August). *Crisp-dm 1.0: step-by-step data mining guide*. SPSS Inc.

Chorowski, J. (2012, December). *Learning understandable classifier models* (Doctoral dissertation, Department of Electrical and Computer Engineering – University of Louisville, Louisville, Kentucky).

Clark, A. R. J. & Everson, R. M. (2012, September). Multi-objective learning of relevance vector machine classifiers with multi-resolution kernels. *Pattern Recognition*, *45*(9), 3535–3543. doi:10.1016/j.patcog.2012.02.025

Cohen, W. W. (1995). Fast effective rule induction. In *Proceedings of the twelfth international conference on machine learning* (pp. 115–123). Tahoe City, California, USA: Morgan Kaufmann.

Craven, M. W. & Shavlik, J. W. (1995). Extracting comprehensible concept representations from trained neural networks. In *In: working notes on the ijcai'95 workshop on comprehensibility in machine learning* (pp. 61–75).

Deb, K. (2001). *Multi-objective optimization using evolutionary algorithms*. New York, NY, USA: John Wiley & Sons, Inc.

Deb, K., Pratap, A., Agarwal, S., & Meyarivan, T. (2002, April). A fast and elitist multiobjective genetic algorithm: nsga-ii. *Evolutionary Computation, IEEE Transactions on*, *6*(2), 182–197. doi:10.1109/4235.996017

DeLeeuw, K. E. & Mayer, R. E. (2008, February). A comparison of three measures of cognitive load: evidence for separable measures of intrinsic, extraneous, and germane load. *Journal of Educational Psychology*, *100*(1), 223–234.

Demšar, J., Curk, T., Erjavec, A., Gorup, Č., Hočevar, T., Milutinovič, M., ... Zupan, B. (2013). Orange: data mining toolbox in python. *Journal of Machine Learning Research*, *14*, 2349–2353.

DiMaggio, P. (1997). Culture and cognition. *Annual review of sociology*, 263–287.

Elomaa, T. (1994, July). In defense of C4.5: notes learning one-level decision trees. In W. W. Cohen & H. Hirsh (Eds.), *Proceedings of the eleventh international conference on machine learning* (pp. 62–69). New Brunswick, NJ, USA: Morgan Kaufmann.

Fortin, F.-A., De Rainville, F.-M., Gardner, M.-A., Parizeau, M., & Gagné, C. (2012, July). DEAP: evolutionary algorithms made easy. *Journal of Machine Learning Research*, *13*, 2171–2175.

Freitas, A. A. (2003). A survey of evolutionary algorithms for data mining and knowledge discovery. In *Advances in evolutionary computing* (pp. 819–845). Springer Berlin Heidelberg.

Freitas, A. A. (2004). A critical review of multi-objective optimization in data mining: a position paper. *ACM SIGKDD Explorations Newsletter*, *6*(2), 77–86.

Freitas, A. A. (2014, March). Comprehensible classification models: a position paper. *SIGKDD SIGKDD Explorations Newsletter*, *15*(1), 1–10. doi:10.1145/2594473.2594475

Gama, J., Rocha, R., & Medas, P. (2003). Accurate decision trees for mining high-speed data streams. In L. Getoor, T. E. Senator, P. Domingos, & C. Faloutsos (Eds.), *Proceedings of the ninth acm sigkdd international conference on knowledge discovery and data mining* (pp. 523–528). KDD '03. Washington, D.C.: ACM. doi:10.1145/956750.956813

Giraud-Carrier, C. (1998). Beyond predictive accuracy: what? In *Ecml'98 workshop notes - upgrading learning to the meta-level: model selection and data transformation* (pp. 78–85). Conference Proceedings/Title of Journal: ECML'98 Workshop Notes - Upgrading Learning to the Meta-Level: Model Selection and Data Transformation. Technical University of Chemnitz.

Girosi, F., Jones, M., & Poggio, T. (1995). Regularization theory and neural networks architectures. *Neural Computation*, *7*, 219–269.

Göpferich, S. (2009). Comprehensibility assessment using the karlsruhe comprehensibility concept. *The Journal of Specialised Translation*, *11*, 31–48.

Gorzałczany, M. B. & Rudziński, F. (2012). Accuracy vs. interpretability of fuzzy rule-based classifiers: an evolutionary approach. In L. Rutkowski, M. Korytkowski, R. Scherer, R. Tadeusiewicz, L. Zadeh, & J. Zurada (Eds.), *Swarm and evolutionary computation* (Vol. 7269, pp. 222–230). Lecture Notes in Computer Science. Springer Berlin Heidelberg. doi:10.1007/978-3-642-29353-5_26

Hall, M., Frank, E., Holmes, G., Pfahringer, B., Reutemann, P., & Witten, I. H. (2009). The weka data mining software: an update. *ACM SIGKDD explorations newsletter*, *11*(1), 10–18.

Harris, E. (2001, October). *Information gain versus gain ratio: a study of split method biases*. The MITRE Corporation.

Houy, C., Fettke, P., & Loos, P. (2012). Understanding understandability of conceptual models - what are we actually talking about? In P. Atzeni, D. Cheung, & S. Ram (Eds.), *Conceptual modeling* (Vol. 7532, pp. 64–77). Lecture Notes in Computer Science. Springer Berlin Heidelberg. doi:10.1007/978-3-642-34002-4_5

Huysmans, J., Dejaeger, K., Mues, C., Vanthienen, J., & Baesens, B. (2011, April). An empirical evaluation of the comprehensibility of decision table, tree and rule based predictive models. *Decision Support Systems*, *51*(1), 141–154. doi:10.1016/j.dss.2010.12.003

Ishibuchi, H., Nakashima, T., & Murata, T. (2001). Three-objective genetics-based machine learning for linguistic rule extraction. *Information Sciences*, *136*(1-4), 109–133. Recent Advances in Genetic Fuzzy Systems. doi:http://dx.doi.org/10.1016/S0020-0255(01)00144-X

Jin, Y. (2000, April). Fuzzy modeling of high-dimensional systems: complexity reduction and interpretability improvement. *Fuzzy Systems, IEEE Transactions on*, *8*(2), 212–221. doi:10.1109/91.842154

Jin, Y. & Sendhoff, B. (2008, May). Pareto-based multiobjective machine learning: an overview and case studies. *Systems, Man, and Cybernetics, Part C: Applications and Reviews, IEEE Transactions on*, *38*(3), 397–415. doi:10.1109/TSMCC.2008.919172

Jin, Y., Sendhoff, B., & Körner, E. (2005). Evolutionary multi-objective optimization for simultaneous generation of signal-type and symbol-type representations. In C. Coello Coello, A. HernĂndez Aguirre, & E. Zitzler (Eds.), *Evolutionary multi-criterion optimization* (Vol. 3410, pp. 752–766). Lecture Notes in Computer Science. Springer Berlin Heidelberg. doi:10.1007/978-3-540-31880-4_52

Jin, Y., Sendhoff, B., & Körner, E. (2006). Simultaneous generation of accurate and interpretable neural network classifiers. In Y. Jin (Ed.), *Multi-objective machine learning* (Vol. 16, pp. 291–312). Studies in Computational Intelligence. Springer Berlin Heidelberg. doi:10.1007/3-540-33019-4_13

Johansson, U., Niklasson, L., & König, R. (2004). Accuracy vs. comprehensibility in data mining models. *Proceedings of the Seventh International Conference on Information Fusion*, *1*, 295–300.

John, G. H. & Langley, P. (1995). Estimating continuous distributions in bayesian classifiers. In P. Besnard & S. Hanks (Eds.), *Proceedings of the eleventh conference on uncertainty in artificial intelligence* (pp. 338–345). UAI'95. Montréal, Qué, Canada: Morgan Kaufmann Publishers Inc.

Knowles, J. (2006, February). Parego: a hybrid algorithm with on-line landscape approximation for expensive multiobjective optimization problems. *Evolutionary Computation, IEEE Transactions on*, *10*(1), 50–66. doi:10.1109/TEVC.2005.851274

Kodratoff, Y. (1994). The comprehensibility manifesto. *KDD Nuggets*, *94*(6).

Kohavi, R. (1996). Scaling up the accuracy of naive-bayes classifiers: a decision-tree hybrid. In *Proceedings of the second international conference on knowledge discovery and data mining* (pp. 202–207). AAAI Press.

Kohavi, R. & Provost, F. (1998, February). Glossary of terms. *Machine Learning*, *30*(2–3), 271–274.

Kononenko, I. & Kukar, M. (2007). *Machine learning and data mining*. Woodhead Publishing.

Kottathra, K. & Attikiouzel, Y. (1996). A novel multicriteria optimization algorithm for the structure determination of multilayer feedforward neural networks. *Journal of Network and Computer Applications*, *19*(2), 135–147. doi:http://dx.doi.org/10.1006/jnca.1996.0011

Kozina, S., Gjoreski, H., Gams, M., & Luštrek, M. (2013). Three-layer activity recognition combining domain knowledge and meta-classification. *Journal of Medical and Biological Engineering*, *33*(4), 406–414.

Kung, H. T., Luccio, F., & Preparata, F. P. (1975, October). On finding the maxima of a set of vectors. *Journal of the ACM*, *22*(4), 469–476. doi:10.1145/321906.321910

Kurgan, L. A. & Petr, M. (2006). A survey of knowledge discovery and data mining process models. *Knowledge Engineering Review*, *21*(1), 1–24.

Lee, C.-C., Cheng, H. K., & Cheng, H.-H. (2007). An empirical study of mobile commerce in insurance industry: task-technology fit and individual differences. *Decision Support Systems*, *43*(1), 95–110. Mobile Commerce: Strategies, Technologies, and ApplicationsDSS on M-Commerce. doi:http://dx.doi.org/10.1016/j.dss.2005.05.008

Liu, G. P. & Kadirkamanathan, V. (1995, June). Learning with multi-objective criteria. In *Artificial neural networks, 1995., fourth international conference on* (pp. 53–58). doi:10.1049/cp:19950528

Maimon, O. & Rokach, L. (2005a). *Data mining and knowledge discovery handbook*. Secaucus, NJ, USA: Springer-Verlag New York, Inc.

Maimon, O. & Rokach, L. (2005b). Decomposition methodology for knowledge discovery and data mining. In O. Maimon & L. Rokach (Eds.), *Data mining and knowledge discovery handbook* (pp. 981–1003). Springer US. doi:10.1007/0-387-25465-X_46

Markowska-Kaczmar, U. & Mularczyk, K. (2006). Ga-based pareto optimization for rule extraction from neural networks. In Y. Jin (Ed.), *Multi-objective machine learning* (Vol. 16, pp. 313–338). Studies in Computational Intelligence. Springer Berlin Heidelberg. doi:10.1007/3-540-33019-4_14

Martens, D. & Baesens, B. (2010). Building acceptable classification models. In R. Stahlbock, F. S. Crone, & S. Lessmann (Eds.), *Data mining: special issue in annals of information systems* (pp. 53–74). Boston, MA: Springer US. doi:10.1007/978-1-4419-1280-0_3

Martens, D., Vanthienen, J., Verbeke, W., & Baesens, B. (2011). Performance of classification models from a user perspective. *Decision Support Systems*, *51*(4), 782–793. Recent Advances in Data, Text, and Media Mining & Information Issues in Supply Chain and in Service System Design. doi:http://dx.doi.org/10.1016/j.dss.2011.01.013

Michalski, R. S. (1983). A theory and methodology of inductive learning. In R. Michalski, J. Carbonell, & T. Mitchell (Eds.), *Machine learning* (pp. 83–134). Symbolic Computation. Springer Berlin Heidelberg. doi:10.1007/978-3-662-12405-5_4

Michie, D. (1987, May). Second european working session on learning. Bled, Slovenia.

Miller, G. A. (1956, March). The magical number seven, plus or minus two: some limits on our capacity for processing information. *The Psychological Review*, *63*(2), 81–97.

Murphy, G. L. & Wright, J. C. (1984). Changes in conceptual structure with expertise: differences between real-world experts and novices. *Journal of Experimental Psychology: Learning, Memory, and Cognition*, *10*(1), 144–155.

Weka: Collections of Datasets. (2016). Retrieved June 10, 2016, from http://www.cs.waikato.ac.nz/ml/weka/datasets.html

Paas, F. G. W. C. & Van Merriënboer, J. J. G. (1993). The efficiency of instructional conditions: an approach to combine mental effort and performance measures. *Human Factors: The Journal of the Human Factors and Ergonomics Society*, *35*(4), 737–743.

Pazzani, M. J. (2000, March). Knowledge discovery from data? *IEEE Intelligent Systems*, *15*(2), 10–13. doi:10.1109/5254.850821

Piltaver, R. (2010, October). Iskanje točnih in razumljivih hibridnih klasifikacijskih dreves. In *Proceedings of 13th international multiconference information society* (pp. 82–85). Ljubljana, Slovenia.

Piltaver, R. (2011, October). Gradnja točnih in razumljivih hibridnih klasifikatorjev na osnovi hierarhičnega gručenja. In *Proceedings of 14th international multiconference information society* (pp. 79–83). Ljubljana, Slovenia.

Piltaver, R., Luštrek, M., & Gams, M. (2014, August). Multi-objective learning of accurate and comprehensible classifiers – a case study. In U. Endriss & J. Leite (Eds.), *Proceedings of 7th european starting AI researcher symposium* (pp. 220–229). Prague, Czech Republic: IOS Press.

Piltaver, R., Luštrek, M., Gams, M., & Martinčić – Ipšić, S. (2014a, October). Comprehensibility of classification trees – survey design. In R. Piltaver & M. Gams (Eds.), *Proceedings of 17th international multiconference information society* (pp. 6–10). Ljubljana, Slovenia.

Piltaver, R., Luštrek, M., Gams, M., & Martinčić – Ipšić, S. (2014b, November). Comprehensibility of classification trees – survey design validation. In *Proceedings of 6th international conference on information technologies and information society* (pp. 5–7). Šmarješke toplice, Slovenia.

Piltaver, R., Luštrek, M., Gams, M., & Martinčić-Ipšić, S. (2016). What makes classification trees comprehensible? *Expert Systems with Applications*, *62*, 333–346. doi:http://dx.doi.org/10.1016/j.eswa.2016.06.009

Piltaver, R., Luštrek, M., Zupančič, J., Džeroski, S., & Gams, M. (2014, August). Multi-objective learning of hybrid classifiers. In T. Schaub, G. Friedrich, & B. O'Sullivan (Eds.), *Proceedings of european conference on artificial intelligence* (pp. 717–722). Prague, Czech Republic: IOS Press.

Platt, J. C. (1999, January). Fast training of support vector machines using sequential minimal optimization. In B. Schölkopf, C. J. C. Burges, & A. J. Smola (Eds.), *Advances in kernel methods — support vector learning* (pp. 185–208). Cambridge, MA, USA: MIT Press.

Polanyi, M. (1966). *The tacit dimension*. Anchor books. Peter Smith.

Pulkkinen, P. (2009). A multiobjective genetic fuzzy system for obtaining compact and accurate fuzzy classifiers with transparent fuzzy partitions. In *Proceedings of the 2009 international conference on machine learning and applications* (pp. 89–94). ICMLA '09. Washington, DC, USA: IEEE Computer Society. doi:10.1109/ICMLA.2009.20

Quinlan, J. R. (1992). Learning with continuous classes. In A. Adams & L. Sterling (Eds.), *Proceedings of the 5th australian joint conference on artificial intelligence* (pp. 343–348). Singapore: World Scientific.

Quinlan, J. R. (1993). *C4.5: programs for machine learning*. San Francisco, CA, USA: Morgan Kaufmann Publishers Inc.

Quinlan, J. R. (1999). Some elements of machine learning. In S. Džeroski & P. Flach (Eds.), *Inductive logic programming* (Vol. 1634, pp. 15–18). Lecture Notes in Computer Science. Springer Berlin Heidelberg. doi:10.1007/3-540-48751-4_3

Rissanen, J. (1978). Modeling by shortest data description. *Automatica*, *14*(5), 465–471. doi:http://dx.doi.org/10.1016/0005-1098(78)90005-5

Rumelhart, D. E., Hinton, G. E., & Williams, R. J. (1986). Parallel distributed processing: explorations in the microstructure of cognition, vol. 1. In D. E. Rumelhart, J. L. McClelland, & C. PDP Research Group (Eds.), (Chap. Learning Internal Representations by Error Propagation, pp. 318–362). Cambridge, MA, USA: MIT Press.

Russell, S. J. & Norvig, P. (2003). *Artificial intelligence: a modern approach*. Prentice Hall series in artificial intelligence. Prentice Hall/Pearson Education.

Schriver, K. et al. (1989). Evaluating text quality: the continuum from text-focused to reader-focused methods. *Professional Communication, IEEE Transactions on*, *32*(4), 238–255.

Shi, C., Chen, M., & Shi, Z. (2005, October). A fast nondominated sorting algorithm. In *Neural networks and brain, 2005. icnn&b '05. international conference on* (Vol. 3, pp. 1605–1610). 13-15 October 2005 ,Beijing, China. doi:10.1109/ICNNB.2005.1614939

Sommer, E. (1995). An approach to quantifying the quality of induced theories. In C. Nedellec (Ed.), *Proceedings of the ijcai workshop on machine learning and comprehensibility*.

Sumner, M., Frank, E., & Hall, M. (2005). Speeding up logistic model tree induction. In A. Jorge, L. Torgo, P. Brazdil, R. Camacho, & J. Gama (Eds.), *Proceedings of the 9th european conference on principles and practice of knowledge discovery in databases* (pp. 675–683). PKDD'05. October 3-7 2005. Porto, Portugal: Springer-Verlag. doi:10.1007/11564126_72

Sweller, J. (1988). Cognitive load during problem solving: effects on learning. *Cognitive Science*, *12*(2), 257–285. doi:10.1207/s15516709cog1202_4

Sweller, J., Van Merrienboer, J. J. G., & Paas, F. (1998). Cognitive architecture and instructional design. *Educational Psychology Review*, *10*(3), 251–296. doi:10.1023/A:1022193728205

Tušar, T. (2007). Optimizing accuracy and size of decision trees. In B. Zajc & A. Trost (Eds.), *Proceedings of the sixteenth international electrotechnical and computer science conference – erk 2007* (Vol. B, pp. 81–84).

Vessey, I. & Galletta, D. (1991). Cognitive fit: an empirical study of information acquisition. *Information systems research*, *2*(1), 63–84.

Zhou, Z.-H. (2005). Encyclopedia of data warehousing and mining. In J. Wang (Ed.), (Chap. Comprehensibility of data mining algorithms, pp. 190–195). Hershey: Idea Group Reference.

Zichermann, G. & Cunningham, C. (2011). *Gamification by design: implementing game mechanics in web and mobile apps* (1st). O'Reilly Media, Inc.

Zitzler, E., Thiele, L., Laumanns, M., Fonseca, C., & da Fonseca, V. (2003, April). Performance assessment of multiobjective optimizers: an analysis and review. *Evolutionary Computation, IEEE Transactions on*, *7*(2), 117–132. doi:10.1109/TEVC.2003.810758

Bibliography

Publications Related to the Thesis

Journal Articles

Piltaver, R., Luštrek, M., Gams, M., & Martinčić-Ipšić, S. (2016). What makes classification trees comprehensible? *Expert Systems with Applications*, *62*, 333–346. doi:http://dx.doi.org/10.1016/j.eswa.2016.06.009

Conference Paper

Piltaver, R. (2010a, October). Iskanje točnih in razumljivih hibridnih klasifikacijskih dreves. In *Proceedings of 13th international multiconference information society* (pp. 82–85). Ljubljana, Slovenia.

Piltaver, R. (2011, October). Gradnja točnih in razumljivih hibridnih klasifikatorjev na osnovi hierarhičnega gručenja. In *Proceedings of 14th international multiconference information society* (pp. 79–83). Ljubljana, Slovenia.

Piltaver, R., Luštrek, M., & Gams, M. (2014, August). Multi-objective learning of accurate and comprehensible classifiers – a case study. In U. Endriss & J. Leite (Eds.), *Proceedings of 7th european starting AI researcher symposium* (pp. 220–229). Prague, Czech Republic: IOS Press.

Piltaver, R., Luštrek, M., Gams, M., & Martinčić – Ipšić, S. (2014a, October). Comprehensibility of classification trees – survey design. In R. Piltaver & M. Gams (Eds.), *Proceedings of 17th international multiconference information society* (pp. 6–10). Ljubljana, Slovenia.

Piltaver, R., Luštrek, M., Gams, M., & Martinčić – Ipšić, S. (2014b, November). Comprehensibility of classification trees – survey design validation. In *Proceedings of 6th international conference on information technologies and information society* (pp. 5–7). Šmarješke toplice, Slovenia.

Piltaver, R., Luštrek, M., Zupančič, J., Džeroski, S., & Gams, M. (2014, August). Multi-objective learning of hybrid classifiers. In T. Schaub, G. Friedrich, & B. O'Sullivan (Eds.), *Proceedings of european conference on artificial intelligence* (pp. 717–722). Prague, Czech Republic: IOS Press.

Other Publications

Journal Articles

Čuk, E., Gams, M., Piltaver, R., Strle, F., Maraspin Čarman, V., & Tasič, J. F. (2015). Intelligent system for diagnosis of erythema migrans. *Applied Artificial Intelligence*, *29*(2), 134–147.

Kramberger, A., Piltaver, R., Nemec, B., Gams, M., & Ude, A. (2016, August). Learning of assembly constraints by demonstration and active exploration. *Industrial Robot: An International Journal*, *43*(5).

Piltaver, R., Cvetković, B., & Kaluža, B. (2015). Denoising human-motion trajectories captured with ultra-wideband real-time location system. *Informatica*, *39*(3), 311–322.

Piltaver, R., Dovgan, E., & Gams, M. (2011, September). An intelligent indoor surveillance system. *Informatica*, *35*(3), 383–390.

Piltaver, R., Luštrek, M., & Gams, M. (2012). The pathology of heuristic search in the 8-puzzle. *Journal of Experimental and Theoretical Artificial Intelligence*, *24*(1), 65–94.

Conference Paper

Cvetković, B., Piltaver, R., Gams, M., & Luštrek, M. (2015, October). Evolution of activity monitoring through various projects. In S. Loshkovska & S. Koceski (Eds.), *Ict innovations conference web proceedings* (pp. 313–322). Skopje, Macedonia: ICT ACT, cop.

Dovgan, E., Piltaver, R., & Gams, M. (2009, October). Inteligentni sistem za nadzor objektov. In M. Bohanec (Ed.), *Proceedings of the 12th international multiconference information society* (Vol. A, pp. 89–92). Ljubljana, Slovenia: Institut Jožef Stefan.

Gjoreski, H., Piltaver, R., & Gams, M. (2015, November). Person identification by analyzing door accelerations in time and frequency domain. In B. de Ruyter (Ed.), *Ambient intelligence: 12th european conference* (Vol. 9425, pp. 60–76). Lecture notes in computer science. Athens, Greece: Springer.

Gjoreski, M., Gjoreski, H., Piltaver, R., & Gams, M. (2013, October). Predicting the arrival and the departure time of an employee. In M. Gams, R. Piltaver, D. Mladenić, M. Grobelnik, F. Novak, B. Blažica, ... A. Brodnik (Eds.), *Proceedings of the 16th international multiconference information society* (Vol. A, pp. 43–46). Ljubljana, Slovenia: Institut Jožef Stefan.

Kozina, S., Piltaver, R., Kužnar, D., & Gams, M. (2013, October). Implementacija virtualnega asistenta v sistem e-vratar. In M. Gams, R. Piltaver, D. Mladenić, M. Grobelnik, F. Novak, B. Blažica, ... A. Brodnik (Eds.), *Proceedings of the 16th international multiconference information society* (Vol. A, pp. 67–70). Ljubljana, Slovenia: Institut Jožef Stefan.

Piltaver, R. (2008, October). Strojno učenje pri načrtovanju algoritmov za razpoznavanje tipov gibanja. In M. Bohanec (Ed.), *Proceedings of the 11th international multiconference information society* (Vol. A, pp. 37–40). Institut Jožef Stefan.

Piltaver, R. (2009, October). Zazvavanje nenavadnega obnašanja s sistemom za lociranje v realnem času in mehko logiko. In M. Bohanec (Ed.), *Proceedings of the 12th international multiconference information society* (Vol. A, pp. 120–123). Ljubljana, Slovenia: Institut Jožef Stefan.

Piltaver, R. (2010b, May). Jezik matematičnih izrazov nad časovnimi vrstami z leksikalnim in sintaktičnim analizatorjem ter evaluatorjem. In B. Kaluža, K. Eleršič, B. Pogorelc, B. Šetina, & M. Vahčič (Eds.), *Proceedings of the 2nd jožef stefan international postgraduate school students conference* (p. 34). Ljubljana, Slovenia: Jožef Stefan International Postgraduate School.

Piltaver, R. (2012, October). Posploševanje visokonivojskih robotskih planov iz primerov. In M. Bohanec (Ed.), *Proceedings of the 15th international multiconference information society* (Vol. A, pp. 138–142). Ljubljana, Slovenia: Institut Jožef Stefan.

Piltaver, R. & Gams, M. (2008, September). Poveljnikova desna roka. In B. Zajc & A. Trost (Eds.), *Proceedings of the 17th international electrotechnical and computer science conference* (Vol. B, pp. 141–144). Portorož, Slovenia: IEEE Region 8, Slovenia Section.

Piltaver, R. & Gams, M. (2009, September). Expert system as a part of intelligent surveillance system. In B. Zajc & A. Trost (Eds.), *Proceedings of the 18th international electrotechnical and computer science* (Vol. B, pp. 191–194). Portorož, Slovenia: IEEE Region 8, Slovenia Section.

Piltaver, R., Gjoreski, H., & Gams, M. (2015, July). Person identification using door accelerations. In *10th anniversary edition workshop on artificial intelligence techniques for ambient intelligence at the twenty-fourth international joint conference on artificial intelligence*. Buenos Aires, Argentina: AAAI Press.

Piltaver, R., Luštrek, M., & Gams, M. (2007, October). Search pathology of 8-puzzle. In M. Bohanec (Ed.), *Proceedings of the 10th international multiconference information society* (Vol. A, pp. 95–98). Ljubljana, Slovenia: Institut Jožef Stefan.

Piltaver, R., Pogorelc, B., & Gams, M. (2011, November). Ambient intelligence for indoor surveillance. In *International joint conference on ambient intelligence* (pp. 5–8). Amsterdam, Netherlands.

Piltaver, R., Tušar, T., Tavčar, A., Ambrožič, N., Šef, T., Gams, M., & Filipič, B. (2014, October). Pametno vodenje sistemov v stavbah s strojnim učenjem in večkriterijsko optimizacijo. In R. Piltaver & M. Gams (Eds.), *Proceedings of the 17th international multiconference information society* (Vol. A, pp. 74–77). Ljubljana, Slovenia: Institut Jožef Stefan.

Piltaver, R., Vodopivec, T., & Gams, M. (2013, October). Identifikacija oseb ob vstopu skozi vrata z uporabo pospeškomera in strojnega učenja. In M. Gams, R. Piltaver, D. Mladenić, M. Grobelnik, F. Novak, B. Blažica, ... A. Brodnik (Eds.), *Proceedings of the 16th international multiconference information society* (Vol. A, pp. 90–93). Ljubljana, Slovenia: Institut Jožef Stefan.

Pušnik, M., Piltaver, R., Viduln, V., & Gams, M. (2013, October). Inteligentni sistem e-vratar. In M. Gams, R. Piltaver, D. Mladenić, M. Grobelnik, F. Novak, B. Blažica, ... A. Brodnik (Eds.), *Proceedings of the 16th international multiconference information society* (Vol. A, pp. 94–97). Ljubljana, Slovenia: Institut Jožef Stefan.

Šef, T., Piltaver, R., & Tušar, T. (2013, October). Projekt opus: optimizacija upravljanja energetsko učinkovitih pametnih stavb. In M. Gams, R. Piltaver, D. Mladenić, M. Grobelnik, F. Novak, B. Blažica, ... A. Brodnik (Eds.), *Proceedings of the 16th international multiconference information society* (Vol. A, pp. 110–113). Ljubljana, Slovenia: Institut Jožef Stefan.

Tavčar, A., Piltaver, R., Zupančič, D., Šef, T., & Gams, M. (2013, October). Modeliranje navad uporabnikov pri vodenju pametnih hiš. In M. Gams, R. Piltaver, D. Mladenić, M. Grobelnik, F. Novak, B. Blažica, ... A. Brodnik (Eds.), *Proceedings of the 16th international multiconference information society* (Vol. A, pp. 114–117). Ljubljana, Slovenia: Institut Jožef Stefan.

Vidulin, V., Piltaver, R., & Gams, M. (2013, October). Pregled inteligentnih algoritmov za procesiranje senzorskih podatkov. In M. Gams, R. Piltaver, D. Mladenić, M. Grobelnik, F. Novak, B. Blažica, ... A. Brodnik (Eds.), *Proceedings of the 16th international multiconference information society* (Vol. A, pp. 126–129). Ljubljana, Slovenia: Institut Jožef Stefan.

Patents

Gams, M., Piltaver, R., Dovgan, E., Planina, A., Pintaric, G., & Pogorelc, B. (2013, February 28). *Intelligent security system and method for detection of unusual behaviour.* SI 23855 A. Slovenia.

Gams, M., Piltaver, R., Gjoreski, H., Moljk, A., Gornik, I., Poje, J., & Virant, M. (2015a, March 26). *Identification method of a person entering a room.* WO 2015/040503 A1. World Intellectual Property Organization.

Gams, M., Piltaver, R., Gjoreski, H., Moljk, A., Gornik, I., Poje, J., & Virant, M. (2015b, March 31). *Process of identifying the person who enters into the space.* SI 24485 A. Slovenia.

Moljk, A., Gornik, I., Poje, J., Virant, M., Gams, M., Piltaver, R., ... Kompara, T. (2014a, September 30). *Interactive door system.* SI 24326 A. Slovenia.

Moljk, A., Gornik, I., Poje, J., Virant, M., Gams, M., Piltaver, R., ... Kompara, T. (2014b, September 4). *Interactive door system.* WO 2014/132108 A1. World Intellectual Property Organization.

As Editor

Gams, M., Piltaver, R., Mladenić, D., Grobelnik, M., Novak, F., Blažica, B., ... Brodnik, A. (Eds.). (2013, October). Proceedings of the 16th international multiconference information society, Ljubljana, Slovenia: Institut Jožef Stefan, *A*.

Piltaver, R. & Gams, M. (Eds.). (2014, October). Proceedings of the 17th international multiconference information society, Ljubljana, Slovenia: Institut Jožef Stefan, *A*.

Piltaver, R. & Gams, M. (Eds.). (2015, October). Proceedings of the 18th international multiconference information society, Ljubljana, Slovenia: Institut Jožef Stefan, *A*.

Biography

Rok Piltaver was born on August 29$^{\text{th}}$, 1982 in Brežice (Slovenia) where he attended elementary school and successfully represented it at mathematics, physics, chemistry and logic competitions. Among other awards, he also won the Gold Jožef Stefan award at the national physics competition. He recieved the Zois schoolarship for gifted students and enrolled in the Gymnasium Brežice where he continued to successfully compete in mathematics and physics. He got introduced to research work in fields of chemistry (gold award at a young researchers forum), biology (best paper award at the 34$^{\text{th}}$ young researchers forum), and sociology (matura seminar paper), and attended Zois scholarship research camps in Kapele (1998) and Velenje (1999).

He concluded the software engineering study program at the Faculty of Computer and Information Science, University of Ljubljana in 2008 with diploma work *The pathology of heuristic search in the 8-puzzle* under the mentorship of Prof. Dr. Ivan Bratko. During the study at the faculty he also attended Music and Ballet High-school Ljubljana in classical and jazz guitar classes. As a student he started with research work at the Department of Intelligent Systems, Jožef Stefan Institute, where he got full-time employment after the graduation.

He published 7 journal papers, more than 25 conference papers, filed 5 patent applications, and edited 3 conference proceedings. He received the Best Slovenian Innovation award (2009) for *Intelligent security system for the surveillance of buildings* and TARAS Award (2011) for the most successful collaboration between research institution and industry for *i-Lab – Intelligent laboratory system for healthcare.* His research interests include artificial intelligence, machine learning and data mining, intelligent systems, ambient intelligence and search pathology. He collaborated on several national and European basic research and application projects: *PDR – surveillance of personnel and equipment movement* (2007–2010), *Confidence – ubiquitous care system to support independent living* (2008–2011), *iLab – intelligent laboratory system for healthcare* (2009–2010), *CHIRON – cyclic and person-centric health management* (2010–2013), *IntellAct – intelligent observation and execution of actions and manipulations* (2011–2014), *E-doorman – intelligent system acting as an human doorman* (2011–2013), *e-Turist – tourist recommender system and trip planner* (2012 – 2013), *OpUS – truly smart home* (2012–2014), *Fit4Work – improving health of older employees* (2014–2017) and others.

He works at Celtra d. o. o. as a software engineer in the analytics department since 2015 where he is responsible for data engineering and analysis supporting a rich-media advertising platform that daily generates tens of gigabytes of data.