IJS delovno poročilo
DP-15586
2026

Marko Bohanec

Experimental Evaluation of DexLearning Software for Machine Learning and Revision of DEX Models

Jožef Stefan Institute
Ljubljana, Slovenija

DEPARTMENT OF KNOWLEDGE TECHNOLOGIES
Jožef Stefan Institute

Abstract

DexLearning is a research-level open-source software package for machine learning and revision of DEX (Decision EXpert) multi-criteria decision models. It implements three main algorithms: (1) *HINT* (Hierarchy INduction Tool) and (2) *MinError* for developing complete DEX models, including attribute hierarchies and decision rules, from data, and (3) *Model Revision* for adapting the decision rules of existing DEX models to new data. In this study, we experimentally evaluated these algorithms using 34 datasets obtained by converting DEX models into flat decision tables and 16 datasets obtained from public data repositories. The algorithms were assessed in terms of (1) the appropriateness of the discovered DEX model structures, (2) model quality measured by classification accuracy, (3) model size measured by information content in bits, (4) robustness to reductions in the learning data set, assessed using learning curves, and (5) execution time. Learning curves were compared with those of the C4.5 decision-tree learning algorithm.

The results confirm that HINT is a robust algorithm capable of developing accurate and compact models across a wide range of dataset sizes, particularly for noiseless data originating from DEX models. On non-DEX data, HINT performs competitively with C4.5. MinError generally achieves lower classification accuracy than HINT but is better able to handle noisy data. Model Revision substantially improves the classification accuracy of revised models, although some variants require excessive execution times that should be improved in future work. Overall, the experiments reconfirm that DEX model learning and revision are difficult combinatorial and computationally demanding tasks, limiting their applicability to moderate-sized problems with up to approximately 20 basic attributes.

Keywords

Multi-criteria model, method DEX (Decision Expert), machine learning, model revision, software, experimental evaluation.

Contents

1 Introduction

DEX (*Decision EXpert*) (Bohanec, 2022) belongs to the class of *multi-criteria decision modeling methods* (Greco, et al., 2016). They are generally aimed at evaluating and analyzing decision alternatives when there are multiple factors to consider. DEX is a *qualitative* multi-criteria decision analysis method, where all model variables, called *attributes*, are qualitative and represented by words. Attributes are organized *hierarchically*, and the evaluation of decision alternatives is defined by *decision rules*. Usually, DEX models are developed by human decision makers and experts, who manually develop the hierarchical structure of attributes and decision rules. DEX Software (https://dex.ijs.si/) is a collection of computer programs that support this process.

DexLearning is recently developed research-level software for machine learning and revision of DEX models from data (Bohanec, 2026). It is intended for use cases in which data are available to support the development, evaluation, or modification of a DEX model. For example, records of decisions made in the past can provide evidence for developing the model structure and deriving its decision rules. Alternatively, an existing DEX model may need to be adapted in response to new data or other evidence, while preserving as much of the original contents as possible. The software can also assist and validate expert modeling, improve incomplete or inadequate decision rules, and restructure existing model subtrees. The resulting models remain hierarchical and interpretable, while data-driven methods can help identify potentially useful structures and decision rules that may be difficult or time-consuming to derive manually.

DexLearning implements three main DEX-learning algorithms:

- *HINT* (HIerarchy Induction Tool): A machine learning algorithm for developing complete DEX models (attribute hierarchy and decision rules) from data. Originally developed by Zupan (1997), Zupan et al. (1999), and Bohanec and Zupan (2004). It employs function decomposition, which proceeds by breaking large decision tables into a hierarchy of smaller ones, creating new internal attributes in the process.
- *MinError*: A minimum-error variation of HINT, specifically adapted to handling noisy data. Instead for searching for a decomposition that is exact on the training data, MinError aims to find a hierarchy that minimizes a classification error estimate. Proposed by Zupan (1997).
- *Model Revision*: A method for updating decision rules from data, while leaving the model structure intact. Originally developed by Žnidaršič (2007), Žnidaršič and Bohanec (2004), and Žnidaršič et al. (2009).

All these algorithms were originally proposed and implemented by their respective authors, and research studies were conducted to assesses and demonstrate their performance. Over time, however, they were no longer regularly maintained and gradually disappeared from practical use, despite the continuing practical need for them. The development of DexLearning was motivated by the goal of bridging this gap by reimplementing these algorithms and providing research-level tools for skilled decision analysts and researchers. For more information on DexLearning software and involved algorithms, please see Bohanec (2026).

In this report, we present results of an experimental evaluation of DexLearning, carried out using a variety of datasets. The primary goals were to assess the performance of implemented algorithms in terms of:

- appropriateness of discovered DEX model structures,
- quality of developed DEX models in terms of classification accuracy,
- size of developed DEX models in terms of information content (bit size),

- sensitivity of algorithms to reducing the learning data set in terms of learning curves,
- time consumed by the execution.

The side goal was to test software robustness and detect any anomalies that might indicate implementation errors.

DexLearning's results were compared to results published in previous studies. This caused some limitations for the present study:

- All previous studies compared DEX-learning algorithms with C4.5, a classic decision-tree learning algorithm (Quinlan, 1993). Therefore, a basic version of C4.5 (that was presumably available at the time of original studies) was implemented in DexLearning for comparison. Comparison with other machine learning algorithms has not been carried out yet and is a subject of possible further studies.
- Algorithms in DexLearning have been reimplemented from the scratch, considering only the available literature, without using any legacy code. Consequently, some algorithmic steps and control parameters are different from the originals, and they yield different results.
- Additionally, the results of previous studies were reported mainly in the form of charts, and the exact numerical results are no longer available to us. Moreover, many of the experiments relied on random data splits that cannot be reconstructed.
- Consequently, exact numerical comparisons between the results of previous studies and those obtained in this study are not possible. We can only make *qualitative* comparisons and assessments. Nevertheless, we expect the new implementations to produce similar model structures, classification accuracies of a similar magnitude, and learning curves with similar overall behavior.
- The selection of datasets for this study is strictly limited to datasets with properties suitable for DEX: (1) having qualitative, non-numeric input features; (2) a discrete output class; and (3) a sufficiently small decision space to be represented by a DEX decision table.

All the datasets used in this study, as well as corresponding computer programs and supplementary tools, are available on the GitLab repository https://repo.ijs.si/markobohanec/dexlearning. Directories that contain components relevant for this study, are:

- `./Data/`: Data files (`.tab`) and related scale-definition files (`.scl`).
- `./DexiModels/`: DEX models used in experiments (`.dxi`).
- `./Scripts/`: Scripts for running programs DexLearn and Experiments (`.json`).
- `./Executables/`: Compiled programs (`.exe`) and software libraries (`.dll`), prepared for execution.
- `./Results/`: Default directory for writing results. Specified in individual script and command files, and can be overridden.
- `./Documentation/`: Various documentation, including this report (`.pdf`).

2 Datasets used in this study

To best of our knowledge, this study involves the largest collection of datasets tested with DEX so far. Generally, datasets are of two types: (1) generated from original DEX models by "flattening" decision rules into a single decision table, and (2) datasets without underlying DEX models. The latter were obtained from UCI, the UC Irvine Machine Learning Repository (https://archive.ics.uci.edu), and other sources.

2.1 Datasets from DEX models

DEX models, used to generate these datasets, were obtained from a database of DEX models (Bohanec, 2017). It contains various DEX models that have been created since 1979 and were made available to the author. The database is updated yearly and at time of this writing contains 822 models developed in 196 decision-making projects. Models are varied: they are of different sizes, using different languages, some of them are published and some are not. Apart from two exceptions (named CanSimplify and MinMax), all models are "real" in the sense that they were developed by real people with specific decision problems in mind.

We selected 38 DEX models for the purpose of this study. They are listed, together with basic size statistics, in Table 1. The column *Classes* displays the number of output values (scale size of the root attribute). The columns *Attributes*, *Basic* and *Aggregate* display, respectively, the number of all attributes, basic attributes (model inputs) and aggregate attributes (internal attributes). *Depth* is the number of attribute levels in the model tree. *Min. rules* and *Max. rules* are the smallest and largest numbers of decision rules contained in individual decision tables. *Decision space size* is the number of all possible combinations of basic attributes' values. Space size corresponds to the total number of data items of a "flattened" decision model: a dataset consisting of all basic attributes and the root of the model tree (final evaluation). The size is determined by multiplying scale sizes of basic attributes; it increases very quickly with the number of basic attributes.

The selection criteria for these models were threefold: (1) to cover a wide variety of sizes, from small to large in terms of the number of attributes, scale sizes in number of decision rules, (2) that this author is reasonably familiar with them, to be able to comment on the structure and contents of generated models, and (3) that the resulting flat decision table fits the size limits imposed in DexLearning (Bohanec, 2026).

We used DEXiWin software (https://dex.ijs.si/dexisuite/dexiwin.html) to generate experimental datasets from these models. After reading some model into DEXiWin, the corresponding dataset can be produced using the command `File/Edit/Function/Flatten decision tables…`, which writes out a tab-delimited data file in the format suitable for DexLearning.

All these models provide a polygon for testing a variety of inputs for DexLearning programs. In what follows, we shall consider in more detail only a handful of models:

- *CanSimplify*: A model of four input attributes X1, …, X4, suitable for simplification. Decision rules define the function Y = MIN(X1, X2) when X3 is {low, med}, and MAX(X1, X2) otherwise. Attribute X4 is redundant.
- *CAR*: Probably the most known DEX model, aimed at the selection of a family car. Distributed with DEXi and DEXiWIn software, and widely referenced in the literature. The flat dataset is included in the UCI repository.

- *Car_Rev0* and *Car_Rev1*: Even simpler models for evaluating cars, proposed by Žnidaršič and Bohanec (2004) in relation with model revision. Car_Rev1 is an expected result of adapting Car_Rev0 to the corresponding dataset.
- *SkiersBasic* and *SkiersEnhanced*: Two models for predicting injury risk in ski resorts (Bohanec and Delibašić, 2015).
- Models whose names begin with *SKL*: Multiple models developed for the Housing Fund of the Republic of Slovenia to support the allocation of housing loans to citizens. The models correspond to multiple floats of loans carried out in the 1990s (Bohanec et al., 1998).
- *VVO* is the Slovenian acronym for nursery schools; the model was used to support children admission to a nursery school (Olave et al., 1989). The flat dataset is included in the UCI repository, too, under the name *Nursery*.

Table 1: DEX models used in this study.

Model/File	Classes	Attributes	Basic	Aggregate	Depth	Min. rules	Max. rules	Decision space size
CanSimplify	3	5	4	1	1	81	81	81
CAR	4	10	6	4	3	12	36	1728
Car_Rev0	5	7	4	3	2	6	12	54
Car_Rev1	5	7	4	3	2	6	12	54
EMPLOY	4	15	9	6	3	4	45	9600
EmploySmall	4	12	7	5	3	12	27	18000
ENT	5	21	12	9	4	6	50	172800
EVAL1	8	14	10	4	3	6	32	1536
EVAL2	8	17	11	6	4	3	32	6912
EVAL2Link	8	17	9	6	4	3	32	1152
KADRI	4	12	7	5	3	12	27	18000
MinMax	3	4	3	1	1	18	18	18
NEK1	4	12	7	5	3	6	36	972
NEK2	4	16	9	7	4	6	36	5832
PLEZ2	4	19	10	9	5	16	16	1048576
PLEZ3	4	16	10	6	4	16	64	1048576
PROGRAMM	5	18	12	6	3	9	48	2985984
PUB_ENT	4	12	7	5	3	4	20	960
RD8	4	19	12	7	3	9	48	174960
SHUT	2	7	6	1	1	256	256	256
SkiersBasic	3	9	6	3	2	9	27	729
SkiersEnhanced	3	21	14	7	3	9	27	4782969
SKL01	5	12	8	4	3	8	36	1728
SKL03	9	21	13	8	4	6	96	3110400
SKL04	9	18	11	7	3	6	48	145800
SKL05	9	17	11	6	3	8	81	218700
SKL05REK	9	13	8	5	3	8	48	4860
SKL07	9	19	12	7	3	8	81	874800
SKL08	9	19	12	7	3	8	48	194400
SKL08REK	9	15	9	6	3	4	48	4320
SKL11	9	16	9	7	3	2	48	16200
SKL11REK	9	13	8	5	3	6	48	2160
SKL14	5	30	19	11	3	2	60	97200
SKL18	5	30	19	11	3	2	60	97200
SKL20	5	23	15	8	3	4	60	97200
Telfer	3	9	8	1	1	6561	6561	6561
VODA	6	9	5	4	3	10	30	750
VVO	5	13	8	5	3	9	36	12960

Let us also briefly explain the remaining models in Table 1. Models *EMPLOY*, *EmploySmall*, *KADRI* and *PROGRAMM* (for "programmers") assess employees. *ENT*, *EVAL1*, *EVAL2*, *EVAL2Link* and *PUB_ENT* are aimed at the assessment of public enterprises (Rodriguez-Ulloa and Bohanec, 2026). *NEK* refers to the first Slovenian nuclear power plant. *Telfer* is an experimental and unpublished model that involves a huge decision table, a good candidate for decomposition with HINT. *PLEZ* is a shorthand for "climbing" in Slovenian and refers to DEX models used in the sportsmen scouting system Talent (Bohanec et al., 1997). RD8 (coming from Slovenian "rak na dojki") is a breast-cancer risk assessment model. SHUT is a DEX model corresponding to the shuttle-landing problem (Shuttle Landing Control, 1988). Finally, VODA is a model for the assessment of water quality.

2.2 Datasets without underlying DEX models

These datasets do not have known underlying DEX models and were collected by some other means. However, their input features and output classes are all qualitative, and the learning data is small enough to facilitate developing DEX models. The datasets were mainly obtained from UCI, the public UC Irvine Machine Learning Repository (https://archive.ics.uci.edu). Table 2 lists these datasets, displaying their names, the number of input features (basic attributes), the number of distinct class (output attribute) values, and the number of data items contained in the dataset.

Table 2: Non-DEX datasets from the UCI repository and other sources.

Dataset	Input features	Output class values	Data items
Balance	4	3	625
Balloons: AdultAND	4	2	19
Balloons: AdultOR	4	2	20
Balloons: YellowAge	4	2	16
Balloons: YellowSmall	4	2	20
Churn1	6	2	3333
Churn2	6	2	3333
Lenses	4	3	24
Monks1	6	2	124
Monks2	6	2	169
Monks3	6	2	122
NPHA	14	3	696
SHUT	6	2	252
SkiStatsDaily	16	3	605
Tennis	4	2	13
TicTacToe	9	2	958

The datasets in Table 2 are:

- *Balance*: Different conditions of a psychological experiment on balance scale weight and distance (Siegler, 1976).
- *Balloons*: Four small datasets representing different conditions of an experiment (Pazzani, 1991). Rules for assigning the output concept value are known for all four datasets.
- *Churn*: Two datasets on customer churn from a telecommunication provider (Delibašić et al., 2025). The datasets contain two different discretizations of the original numeric dataset (https://github.com/albayraktaroglu/Datasets/blob/master/churn.csv).
- *Lenses*: Database for fitting contact lenses (Cendrowska, 1987).
- *Monks*: A collection of three binary classification problems over a six-attribute discrete domain. Rules for assigning the output concept value are known for all three datasets (Thrun et al., 1991; Wnek, 1993).

- *NPHA*: Results of the National Poll on Healthy Aging to gather insights on the health, healthcare, and health policy issues affecting Americans aged 50 and older (NPHA, 2017).
- *SHUT*: A small dataset for determining the conditions under which an autolanding would be preferable to manual control of the Shuttle spacecraft (Shuttle Landing Control, 1988).
- *SkiStatsDaily*: Skiing data from the Serbian ski resort Mt. Kopaonik: 650 skiing days over six consecutive years (2005 to 2010). Suitable mainly for revision rather than model construction.
- *Tennis*: The classic machine learning dataset of 14 observations of Outlook, Temperature, Humidity, Wind → PlayTennis (Yes/No) (Quinlan, 1986).
- *TicTacToe*: Binary classification task on possible configurations of tic-tac-toe game (Aha, 1991).

3 Methods

This study involved the following methodological steps:

1. *Reconstruction* of DEX models from data with known original DEX models.
2. *Construction* and interpretation of DEX models from data lacking underlying DEX models.
3. Analysis of HINT and MinError model learning *performance* across varying training data sizes, includes a comparison with the C4.5 decision tree learning algorithm.
4. Experimental evaluation of the *model revision* algorithm.

3.1 Reconstruction of DEX models

Reconstruction of DEX models was performed using the HINT algorithm on "flattened DEX model data" (Table 1), consisting of all possible combinations of input attribute values defined by the original model. The reconstruction was carried out by running the `DexLearn` program and executing the `.json` scripts located in the `./Scripts` directory, with one or more scripts using different parameter settings for each model. The original models (in `./DexiModel`) and the reconstructed models (in subdirectories of `./Results`) were then compared on a per-model basis.

When comparing models, it should be considered that DEX learning algorithms cannot fully reconstruct the original models. First and foremost, a learning algorithm has no way of knowing how to name the newly constructed aggregate attributes. Consequently, newly created concepts are assigned names such as C1, C2, etc. ("C" for "concept"). The same applies to the values of newly constructed concepts, which are represented by ordinal numbers rather than by the more usual character strings.

Learned models consist exclusively of binary subtrees. Therefore, original subtrees with more than two subordinate subtrees are represented by multiple, possibly nested, binary subtrees. Nevertheless, the comparison can still reveal whether similar attributes are grouped together into similar subtrees and whether the resulting learned subtrees "make sense" in the given problem domain. It is also useful to compare the sizes of the respective models, in terms of the number of attributes and their bit sizes, since the learning algorithms may identify and eliminate redundant attributes and/or attribute values.

3.2 Construction of DEX models from data

The construction of DEX models from data is similar to reconstruction, except that there is no original model against which the resulting model can be compared. Nevertheless, it is usually possible to assess whether the constructed attribute subtrees represent plausible and valid concepts in the given problem domain.

However, input data rarely contains all possible combinations of input values. Although both the HINT and MinError algorithms can generalize to some extent, missing combinations may still result in undefined decision rules. A particular problem is posed by noisy data, in which identical input configurations are associated with different outcomes. Previous studies (Zupan, 1997, among others) have shown that HINT is particularly sensitive to such situations, whereas MinError handles them comparatively better.

3.3 Performance of machine learning algorithms

The performance of machine learning algorithms HINT, MinError and C4.5 was assessed on a variety of input datasets. The program `Experiment` was used to execute the `.json` scripts while varying:

1. *Sizes*: The percentage of dataset used for training: 100, 90, 80, 70, 60, 50, 40, 30, 20, 10. The remaining items were used to assess the classification accuracy of learned models (except for *Size*=100, where the whole dataset was used for testing, possibly inducing some overfitting).
2. *M-estimate* (relevant only for MinError): 1, 2, and 5.

Ten runs were repeated for each combination of settings. We observed:

- *Classification accuracy* of the learned model, measured on test data;
- *Information contents* of the generated models, in terms of the *number of input attributes* and cumulative *bit size* of decision tables;
- *Elapsed time* for learning each model.

Charts displaying these results were qualitatively compared with charts published in studies Zupan (1997), Zupan et al. (1999), and Bohanec and Zupan (2004).

The experiments were run on mid-range desktop computers, using single-core computation. While the exact elapsed times differ and are not really important, it is important to assess their magnitude and trends while varying the settings.

After it has been observed in early experiments that M-estimates 2 and 5 generally yield worse results than 1, while consuming substantial computing time, only *M-estimate*=1 was tested with the largest datasets.

3.4 Model revision

Model revision requires an already developed DEX model and a dataset that, at least to some extent, disagrees with it. Three cases met these criteria in the collected datasets: *Car_Rev*, *Churn*, and *Skiers* (using *SkiStatsDaily*). The experiments on datasets *Car_Rev* and *Skiers* were performed using the program `DexRevise` with various options: none (default), `/progressive`, `/quick`, `/median`, `/average` and `/monotone`. The original and revised models were compared on a per-model basis, evaluating information content, classification accuracy, the number of revision algorithm iterations, and the appropriateness of the adapted decision rules.

4 Results

4.1 Reconstruction of DEX models

Let us first show an example DEX model reconstruction. Figure 1 shows, side by side, the structure and scales of attributes of the original CAR evaluation model, and the model developed automatically by HINT from data. One can immediately note that the HINT model is "colorless", because good and bad scale-value categories are not known to HINT. Next, all original aggregate attributes are replaced by "nameless" attributes C1, …, C5, whose scale values are represented by ordinal numbers rather than comprehensible names that occur in the original. Nevertheless, it is easy to recognize the similarity of the two structures: the same attributes are grouped together in collocated subtrees, only that the reconstructed ones are binary and therefore nested to deeper levels. Also, what is not shown here, the reconstructed model contains decision rules that represent exactly the same logic as the original model; classification accuracy of the reconstructed model, measured on the full dataset, is 100%. Even though the reconstructed model contains more decision tables (6) than the former (4), they are smaller. Collectively, their information content is lower, too: 125.36 bits instead of 160.00 bits. This indicates that the reconstructed model is actually simpler and, in theory, more comprehensible. In any case, if the original model was not available, it would be relatively easy for a skilled decision analyst to advance the reconstructed model, by naming the aggregate attributes and scale values, to a comprehensible and useful level.

Attribute	Scale
CAR	unacc; acc; good; *v-good*
├─PRICE	v-high; high; med; *low*
│ ├─BUYING	v-high; high; med; *low*
│ └─MAINT	v-high; high; med; *low*
└─TECH	poor; satisf; good; *v-good*
├─COMFORT	bad; acc; good; *v-good*
│ ├─DOORS	2; 3; 4; *5-more*
│ ├─PERSONS	2; 4; *more*
│ └─LUG_BOOT	small; med; *big*
└─SAFETY	low; med; *high*

Attribute	Scale
CAR	unacc; acc; good; v-good
├─C5	1; 2; 3; 4
│ ├─BUYING	v-high; high; med; low
│ └─MAINT	v-high; high; med; low
└─C4	1; 2; 3; 4
├─C3	1; 2; 3
│ ├─C2	1; 2; 3; 4
│ │ ├─C1	1; 2; 3
│ │ │ └─DOORS	2; 3; 4; 5-more
│ │ └─PERSONS	2; 4; more
│ └─LUG_BOOT	small; med; big
└─SAFETY	low; med; high

Figure 1: Structure and scales of the original (left) and reconstructed by HINT (right) CAR model.

In order to compare HINT and MinError, also consider the model developed from the same data by MinError (Figure 2). MinError also succeeded in developing a DEX model, however, it is somewhat inferior to HINT's. The structure does not resemble the original one equally well, and the information content is 140.43 bits. Most importantly, its classification accuracy is 98.03%, meaning that the original model is not accurately reconstructed. This makes MinError less suitable for DEX model reconstruction.

Attribute	Scale
CAR	unacc; acc; good; v-good
├─C4	1; 2; 3; 4; 5; 6; 7
│ ├─C3	1; 2; 3; 4
│ │ ├─C2	1; 2; 3
│ │ │ ├─C1	1; 2
│ │ │ │ ├─DOORS	2; 3; 4; 5-more
│ │ │ │ └─LUG_BOOT	small; med; big
│ │ │ └─PERSONS	2; 4; more
│ │ └─SAFETY	low; med; high
│ └─BUYING	v-high; high; med; low
└─MAINT	v-high; high; med; low

Figure 2: Structure and scales of the CAR model developed by MinError.

In this study, HINT reconstructed DEX models really well. Table 3 displays the properties of best DEX models developed by HINT from the respective full ("flattened") datasets. The same properties are shown as in Table 1, except that the number of *Classes* is not repeated, and the approximate *Time* elapsed by running HINT (in milliseconds) is added. In some cases, HINT generated multiple models with different structures, but the same properties as displayed in the table.

Table 3: DEX models reconstructed by HINT.

Model/File	Attributes	Basic	Aggregate	Depth	Min. rules	Max. rules	Space size	Bit size	Time [ms]
CanSimplify	6	3	3	3	3	15	27	40.71	44.63
CAR	12	6	6	5	4	16	1728	125.36	93.71
Car_Rev0	7	4	3	2	6	12	54	54.13	14.83
Car_Rev1	8	4	4	3	3	12	54	52.37	20.12
EMPLOY	19	9	10	5	4	16	9600	144.23	919.94
EmploySmall	16	7	9	4	5	16	18000	154.42	596.65
ENT	27	12	15	6	3	12	172800	190.58	13688.13
EVAL1	20	10	10	5	3	18	1536	115.28	184.99
EVAL2	18	9	9	4	3	25	1152	152.47	666.41
EVAL2Link	18	9	9	4	3	25	1152	152.47	105.26
KADRI	16	7	9	4	5	16	18000	154.42	592.16
MinMax	5	3	2	2	6	9	18	23.77	115.92
NEK1	9	5	4	3	6	20	162	87.16	45.29
NEK2	14	7	7	5	3	20	972	109.18	180.18
PLEZ2	20	10	10	5	4	16	1048576	276.00	68938.99
PLEZ3	20	10	10	5	4	42	1048576	430.52	67021.43
PROGRAMM	25	12	13	6	3	24	2985984	368.42	286136.60
PUB_ENT	15	7	8	5	3	16	960	111.68	48.91
RD8	25	12	13	5	3	24	174960	227.00	16237.88
SHUT	13	6	7	5	4	9	256	54.79	62.55
SkiersBasic	14	6	8	4	3	12	729	76.13	31.24
SkiersEnhanced	30	14	16	5	3	15	4782969	238.83	484969.70
SKL01	17	8	9	5	3	24	1728	162.01	96.00
SKL03	29	13	16	7	3	44	3110400	605.20	111100.20
SKL04	24	11	13	5	3	36	145800	315.89	4930.44
SKL05	25	11	14	5	3	48	218700	388.63	5473.26
SKL05REK	17	8	9	4	3	36	4860	236.44	109.00
SKL07	28	12	16	6	3	36	874800	343.81	16558.25
SKL08	28	12	16	6	3	30	194400	260.04	5491.04
SKL08REK	20	9	11	4	3	24	4320	178.53	111.34
SKL11	16	7	9	5	3	30	2700	224.19	195.06
SKL11REK	16	8	8	4	4	30	2160	223.53	107.49
SKL14	26	11	15	5	3	40	97200	256.98	2846.53
SKL18	25	11	14	5	3	40	97200	275.51	2751.04
SKL20	25	11	14	5	3	40	97200	275.51	2797.02
Telfer	18	8	10	6	3	25	6561	183.17	332.26
VODA	10	5	5	4	5	25	750	195.10	22.15
VVO	17	8	9	5	3	15	12960	154.34	573.29

For better interpretation of these results, the following chart compare the original model properties (Table 1) with those of the reconstructed models (Table 3). On the horizontal axis, models are sorted by the number of attributes. The thick and thin lines correspond to the original and reconstructed models, respectively.

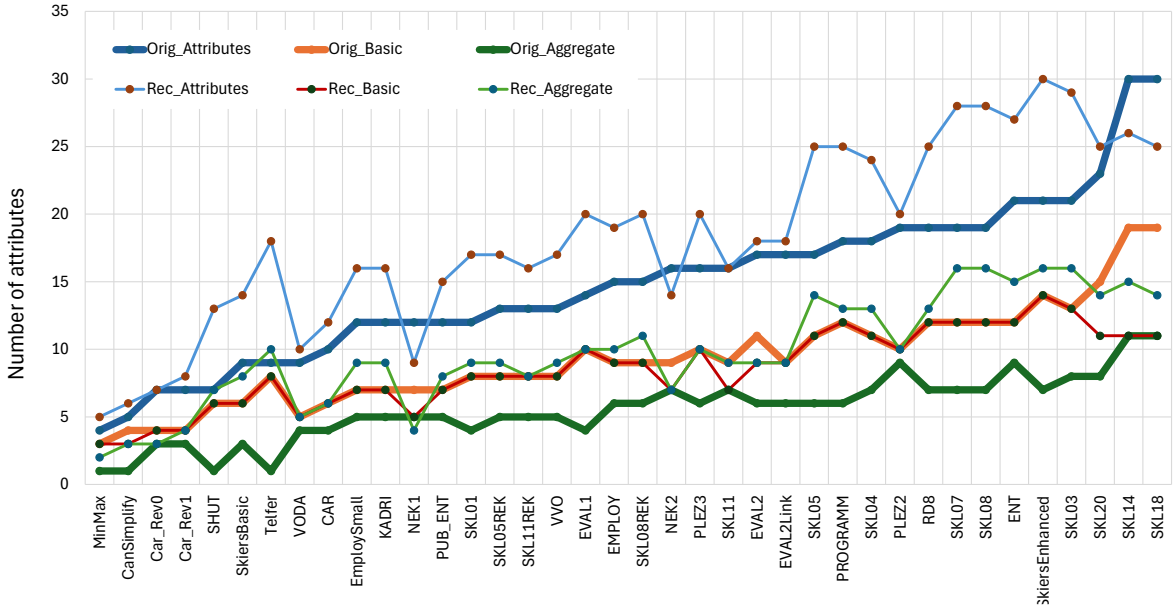

Figure 3: Comparison of the number of attributes (all, basic, aggregate) between original and HINT-developed DEX models.

Figure 3 compares model sizes in terms of the number of all, basic and aggregate attributes. The number of all attributes is typically higher in reconstructed (average 18.37, median 18) than original models (average 15.1, median 15). This is due to the increased number of (binary) subtrees in reconstructed models. Nevertheless, there are five cases (NEK1, NEK2, SKL11, SKL20, SKL14, and SKL18) with a lower number of attributes, which are due to removal of redundant attributes. This is clearly visible in the curves representing the number of basic attributes. The lines showing the number of aggregate attributes behave similarly as those corresponding to all attributes.

Figure 4 compares the depths of models, in the same order as in Figure 3. It is evident that the reconstructed models contain more levels (average 4.64, median 5) than the original ones (average 2.92, median 3). This is due to a deeper nesting of binary subtrees that is needed to accurately represent non-binary original subtrees.

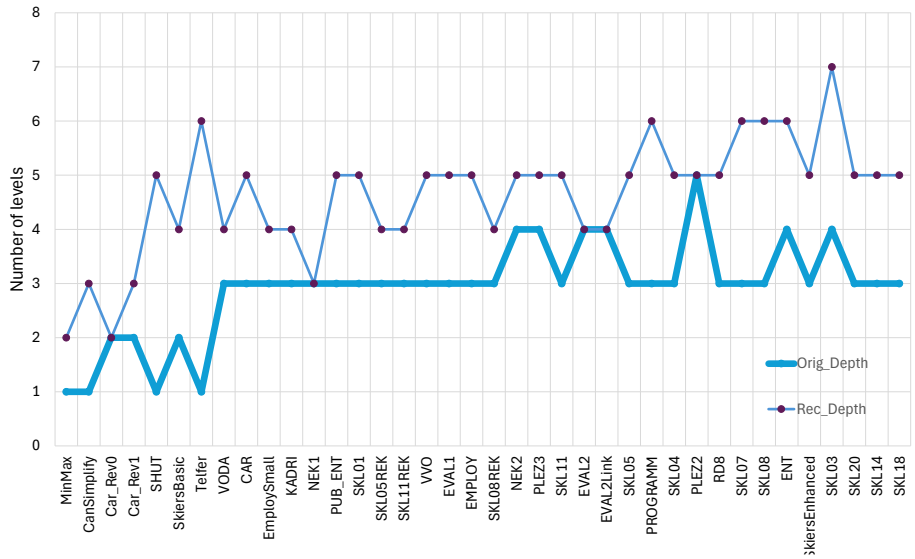

Figure 4: Comparison of model depths of original and reconstructed DEX models.

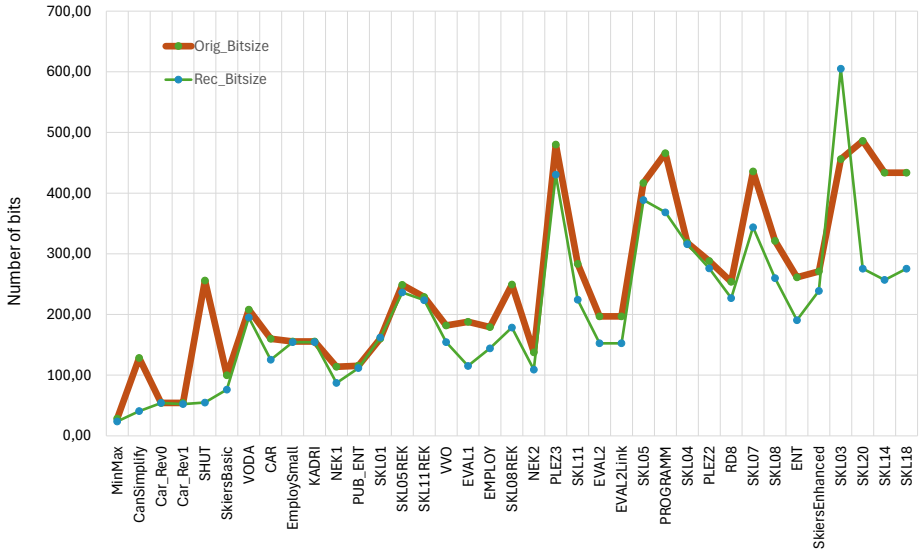

Figure 5: Comparison of information contents of original and reconstructed DEX models.

Figure 5 compares information content of models. The number of bits encoded in reconstructed models is generally smaller (average 200.49, median 180.85) than in the original ones (average 416.73, median 248.95). This indicates that reconstructed models are represented in a more compact way, while still representing the same decision rules. There is only one curious exception (SKL03), in which the number of generated aggregate attributes (16) is twice as many as the original (8), while the generated decision tables are not small enough to substantially diminish the cumulative bit size.

Finally, let us comment on HINT's execution times. It is well known (Zupan, 1997; Zupan et al., 1999) that learning DEX models from data is hard and involves several computational steps that are NP-complete. HINT alleviates this by employing heuristics, which substantially boost the performance, but still have their limits. In this study, DEX models were relatively small (but still realistic), and execution times were perfectly feasible (average 29 seconds, median 453 milliseconds). However, there were some large models (such as SkiersEnhanced) where execution times approached 8 minutes and more. While a faster computer and better optimized code might improve these times to some extent, it is clear that HINT is not really suitable for datasets much larger than those considered in this study. An empirical analysis of execution times in Table 3 reveals that elapsed time T_{ms} is proportional to $0.58\ D^{0.78}$, where D is decision space size. When D grows ×10, time grows roughly ×6–7; when it doubles, time grows roughly ×1.7–1.8. Moreover, D itself grows exponentially with the number of basic attributes, further limiting the scalability of the algorithm.

4.2 Construction of DEX models from data

DEX models were constructed using HINT and MinError from datasets for which no prior DEX model existed; results are shown in Table 4 for HINT and Table 5 for MinError. In addition to properties of the best developed model (number of all, *basic* and *aggregate attributes*, model *depth*, minimum and maximum number of decision *rules* per decision table, size of the covered decision *space*, information contents in *bits*), approximate elapsed execution *time* and achieved classification accuracy are shown.

Let us begin the overview with the *Balloons* and *Monks* datasets. Both are small, consisting of four and three sets of learning examples, respectively, containing about 20 (Balloons) and 150 (Monks) training data items. Importantly, all these datasets are in the UCI repository accompanied with documentation describing the rules that guided the construction of training data. Thus, we can compare these rules with decision tables learned in DEX models.

Table 4: Models constructed from data by HINT.

Dataset	Attributes	Basic	Aggregate	Depth	Min. rules	Max. rules	Space size	Bit size	Time [ms]	Classif. accuracy
Balance	7	4	3	3	25	70	625	424.52	163.74	100.00
Balloons: AdultAND	3	2	1	1	4	4	4	4.00	31.21	100.00
Balloons: AdultOR	3	2	1	1	4	4	4	4.00	15.68	100.00
Balloons: YellowAge	7	4	3	2	4	4	16	12.00	31.30	100.00
Balloons: YellowSmall	3	2	1	1	4	4	4	4.00	31.20	100.00
Churn1	11	6	5	4	4	16	96	82.93	107.82	92.32
Churn2	9	5	4	3	4	9	32	32.94	45.85	88.72
Lenses	7	4	3	3	6	6	24	28.53	46.82	100.00
Monks1	6	3	3	2	4	9	36	17.00	40.03	100.00
Monks2	11	6	5	4	6	12	432	63.06	149.02	100.00
Monks3	11	6	5	4	3	24	432	62.45	72.33	100.00
NPHA	25	13	12	5	4	4200	1728000	32518.47	635 min	93.82
SHUT	13	6	7	5	4	9	256	54.79	93.75	100.00
Tennis	7	4	3	3	4	9	36	21.34	62.46	100.00
TicTacToe	17	9	8	5	6	136	19683	831.32	1337.75	100.00

Table 5: Models constructed from data by MinError.

Dataset	Attributes	Basic	Aggregate	Depth	Min. rules	Max. rules	Space size	Bit size	Time [ms]	Classif. accuracy
Balance	7	4	3	3	15	45	625	158.50	187.50	89.44
Balloons: AdultAND	3	2	1	1	4	4	4	4.00	15.63	100.00
Balloons: AdultOR	3	2	1	1	4	4	4	4.00	31.30	100.00
Balloons: YellowAge	7	4	3	3	4	6	16	16.34	31.29	100.00
Balloons: YellowSmall	3	2	1	1	4	4	4	4.00	31.31	100.00
Churn1	11	6	5	5	4	6	96	26.34	17.00	89.74
Churn2	9	5	4	4	4	6	32	20.34	16.00	88.33
Lenses	7	4	3	3	4	6	24	21.85	31.30	95.83
Monks1	6	3	3	2	4	9	36	17.00	30.00	100.00
Monks2	11	6	5	5	6	12	432	69.96	46.87	100.00
Monks3	11	6	5	5	4	9	432	31.00	46.87	93.44
NPHA	30	13	17	13	3	36	1728000	835.65	6.4 min	75.14
SHUT	13	6	7	6	4	9	256	37.68	31.30	98.02
SkiStatsDaily: Basic	11	6	5	5	6	9	729	61.30	102.06	67.44
SkiStatsDaily: All	27	14	13	13	9	18	4782969	264.69	61 min	72.40
Tennis	7	4	3	3	6	9	36	29.77	15.63	100.00
TicTacToe	17	9	8	8	6	21	19683	213.10	3344.60	89.77

As it turned out, HINT correctly learned all the Balloons' and Monks' rules. As an example, Figure 6 shows the DEX model structure and three decision tables learned from the Monk1 dataset. According to Monk documentation, the underlying rule is `(a1 = a2) or (a5 = 1)`, and the model represents it correctly. MinError also correctly learned these models, except for Monks3, where it achieved classification accuracy 93.44%. It should be noted that Monk3 data contains 5% of added class noise, which makes machine learning more difficult.

Attribute	Scale
Class	1;0
├─C2	1;2
│ ├─a1	1;2;3
│ └─a2	1;2;3
└─C1	1;2
└─a5	3;2;4;1

a1	a2	C2
1	1	1
1	2	2
1	3	2
2	1	2
2	2	1
2	3	2
3	1	2
3	2	2
3	3	1

a5	C1
3	1
2	1
4	1
1	2

C2	C1	Class
1	1	1
1	2	1
2	1	0
2	2	1

Figure 6: DEX model learned by HINT from the Monk1 dataset.

Turning to datasets that are widely addressed in the literature: Lenses, Shuttle, and Tennis. While the latter is a well-known didactic example, the former two address real decision-making tasks in the category that seems very suitable for DEX: having a relatively small number of qualitative attributes, where decision rules are to be formulated or confirmed by domain experts. While we do not know the correct formulation of corresponding DEX models, we can confirm that HINT succeeded to learn 100% accurate models whose attribute structures and decision rules seem very reasonable (Figure 7). This is also indicated by newly constructed concepts (whose names start with "C"): their value scales are very small, containing only two or three distinct values, which speaks in favor that they represent something "real" that "makes sense".

Attribute	Scale
Lenses	3;2;1
├─C2	1;2;3
│ ├─C1	1;2;3
│ │ ├─Age	1;2;3
│ │ └─SpectaclePresc	1;2
│ └─Astigmatic	1;2
└─TearProduction	1;2

Attribute	Scale
DS	1;2
├─C13	1;2
│ ├─C12	1;2;3
│ │ ├─C4	1;2;3
│ │ │ ├─C2	1;2;3
│ │ │ │ └─MAG	1;2;3;4
│ │ │ └─WIND	1;2
│ │ └─C1	1;2;3
│ │ └─ERR	1;2;3;4
│ └─SIGN	1;2
└─C11	1;2;3
├─STAB	1;2
└─VIS	1;2

Attribute	Scale
Tennis	No;Yes
├─C4	1;2;3
│ ├─C2	1;2
│ │ ├─Temperature	Hot;Mild;Cool
│ │ └─Wind	Weak;Strong
│ └─Humidity	High;Normal
└─Outlook	Sunny;Overcast;Rain

Figure 7: DEX models developed from data: Lenses (left), SHUT (middle), Tennis (right).

MinError (Table 5) was also successful with Tennis (100% accuracy), but less successful with Lenses (95.83%) and Shuttle (98.02%).

TicTacToe is already a more difficult problem, even though it addresses a crisp, well-defined game. Again, HINT mastered it with 100% accuracy, while MinError was only 89.77% accurate. HINT developed a model (Figure 8) whose new attributes C6 and C7 have a very high number of values (15 and 17, respectively). While this might be due to a combinatorial nature of the game, it may also indicate an inappropriate formulation of aggregate concepts. Also, because the training data is incomplete, the top-level DEX decision table (corresponding to the attribute Outcome) contains 5.15% of undefined rules.

Attribute	Scale
Outcome	negative;positive
├─C7	1;2;3;4;5;6;7;8;9;10;11;12;13;14;15;16;17
│ ├─C6	1;2;3;4;5;6;7;8;9;10;11;12;13;14;15
│ │ ├─C5	1;2;3;4;5;6
│ │ │ ├─C1	1;2
│ │ │ │ ├─TL	x;o;b
│ │ │ │ └─TM	x;o;b
│ │ │ └─BR	x;o;b
│ │ └─C4	1;2;3;4;5;6;7
│ │ ├─BL	x;o;b
│ │ └─BM	x;o;b
│ └─C3	1;2;3;4;5;6;7
│ ├─MM	x;o;b
│ └─MR	x;o;b
└─C2	1;2;3;4;5;6;7;8
├─TR	x;o;b
└─ML	x;o;b

Figure 8: Structure and value scales of a DEX model learned from the TicTacToe dataset.

The remaining datasets (Balance, Churn, NPHA, Skiers) are increasingly harder to consider, because they address more difficult decision problems; the datasets or their decision spaces are substantially larger. We do not feel competent to interpret the results other than algorithmic and model-related. Balance is the only such dataset that was learned by HINT with 100% accuracy. Classification accuracy of all other models was lower, down to 72.40% with MinError on Skiers. HINT has not been even tried on the SkiStatsDaily dataset, because the data contained too many conflicting items and insufficiently covered the large decision space. Considering execution times, NPHA and Skiers datasets turned out to be particularly difficult, requiring from 6.4 minutes of MinError on NPHA to as much as 10 hours of HINT on NPHA. This reconfirms that HINT and MinError are not really suitable for large, partly incomplete and possibly noisy datasets.

The comparison of models developed by HINT (Table 4) and MinError (Table 5) in terms of their size (number of attributes) shows almost no differences. The only difference occurs with the NPHA dataset, for which HINT developed a model with 25 attributes (12 aggregate) and MinError with 30 attributes (17 aggregate). However, the methods differ in terms of the depth of models and their bit sizes (Figure 9). In general, MinError generates slightly deeper models, but with smaller decision tables. Considering classification accuracy of developed models, measured on the original dataset (Figure 10, left), HINT more often succeeds in developing a 100% accurate model. In general, the algorithms' execution times (Figure 10, right) are similar, with a slight advantage of MinError.

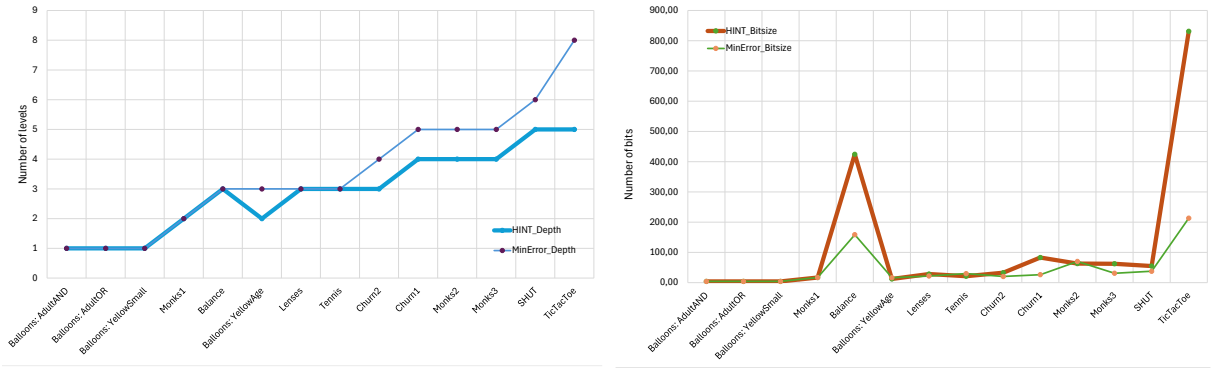

Figure 9: Comparison of models constructed by HINT and MinError: model depth (left) and bit size (right). The datasets are sorted by the number of input features. The largest models (Skiers and NPHA) are outliers and are excluded from these charts.

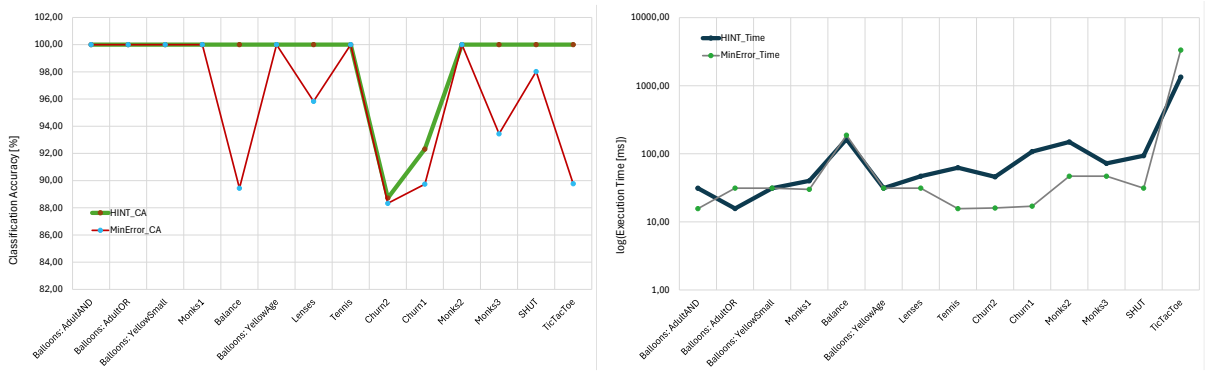

Figure 10: Comparison of models constructed by HINT and MinError: classification accuracy (left) and (logarithm of) execution time (right). The datasets are sorted by the number of input features. The largest models (Skiers and NPHA) are outliers and are excluded from these charts.

4.3 Performance of machine learning algorithms

We assess the performance of DEX machine learning algorithms HINT and MinError depending on the size of the training set, and in comparison with decision tree learning algorithm C4.5. We observe classification accuracy and information content (bit size) of developed models, and the computation time required for their development. Experiments were repeated ten times on random splits of the dataset that contained from 100 to 10 percent (in steps of 10) of all data items. Measurements are displayed using charts (such as Figure 11). Each chart line is surrounded by two bands: the narrower is darker and corresponds to ± standard deviation, while the wider shows the minimum-maximum range of the corresponding quantities.

For data generated from DEX models, the algorithms typically behave as shown in Figure 11 (CAR dataset) and Figure 12 (VVO dataset). HINT learns 100% accurate models for a wide range of data sizes on the left, and only deteriorates when the sizes drop below about 40% on the right. Classification accuracy of MinError drops faster and rarely persists at 100%. The effect of increasing m-estimate (from 1 to 2 and 5) substantially decreases the accuracy; this is the reason for excluding m-estimates 2 and 5 from some of the charts. The accuracy of C4.5 drops gradually from left to right. It might be generally worse (Figure 11) or better (Figure 12) than MinError's with m=1.

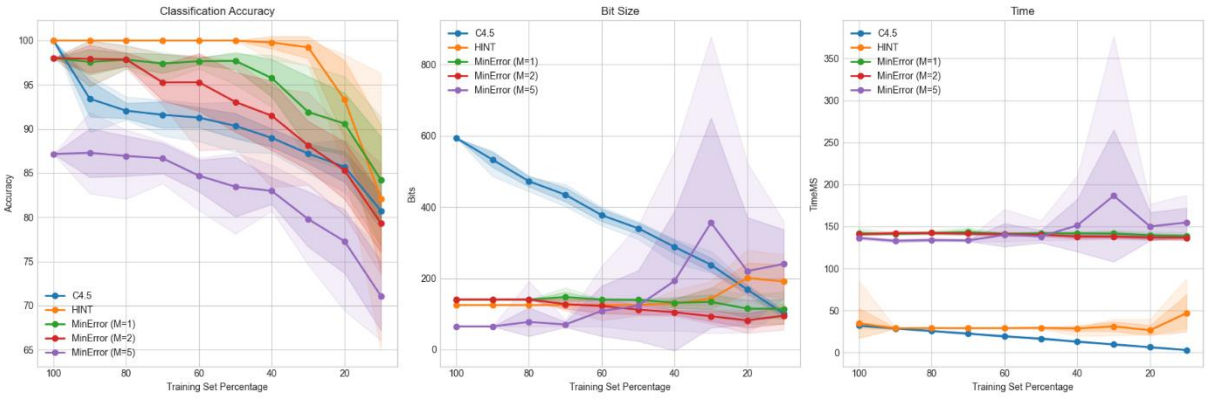

Figure 11: Performance of the algorithms on the CAR dataset.

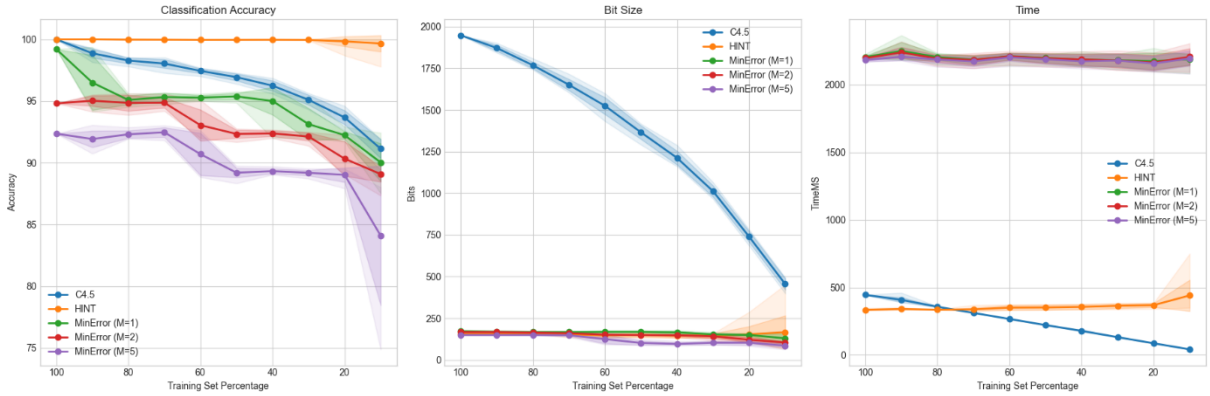

Figure 12: Performance of the algorithms on the VVO dataset.

The bit size of HINT and MinError models is substantially lower than the one of C4.5 decision tree, which seems to indicate that DEX models are indeed appropriate representation formalisms for this type of data. The bit size of decision trees gradually decreases from large to small datasets, which is mainly due to decreased number of tree levels and increased generalization. Execution times of HINT and C4.5 are comparable, but have different trends: C4.5 runs faster on smaller datasets, which cause HINT to struggle more. This is because small datasets induce sparse incompatibility matrices, which require increasing graph-coloring time. MinError consistently consumes more time than the other two algorithms. Let us also add that the CAR and VVO learning curves visually match the curves of CAR and Nursery, published in Zupan (1997).

C4.5 is not always substantially inferior to HINT. In the case of EVAL2 dataset (Figure 13), both algorithms compete well at large dataset sizes. Once more, MinError is worse that the two, particularly with m-estimates 2 and 5.

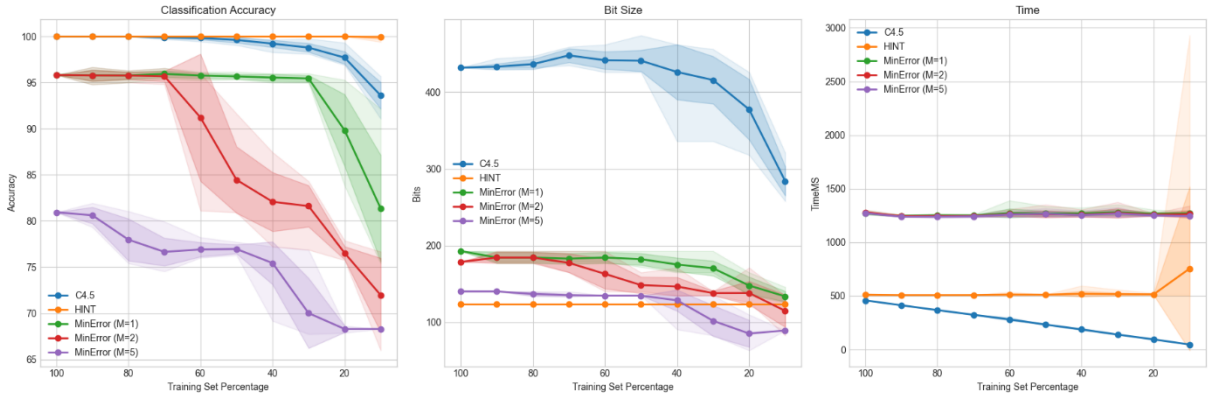

Figure 13: Performance of the algorithms on the EVAL2 dataset.

Algorithms' performance is more varied on non-DEX datasets. For example, on Balance (Figure 14) both HINT and MinError (m=1) achieve similar classification accuracies. Relatively fast deterioration of classification accuracy with decreasing data size may indicate that DEX is not a suitable representation of the underlying principles. Similar observations can be made with the Lenses (Figure 15) and SHUT (Figure 16) datasets.

In summary, we can confirm that the three algorithms implemented in DexLearning are very robust and can generate solid models for a wide variety of training data sizes. HINT performs particularly well when the data matches DEX representational principles (hierarchical attribute structure and decision rules). Otherwise, HINT competes well with C4.5. In most cases, MinError appears inferior to both (in

terms of classification accuracy and execution times). However, it is indispensable in cases when the training data is noisy so that HINT cannot be used, but a DEX model is needed for some reason, either to understand the structure of attributes or to be improved further by model revision.

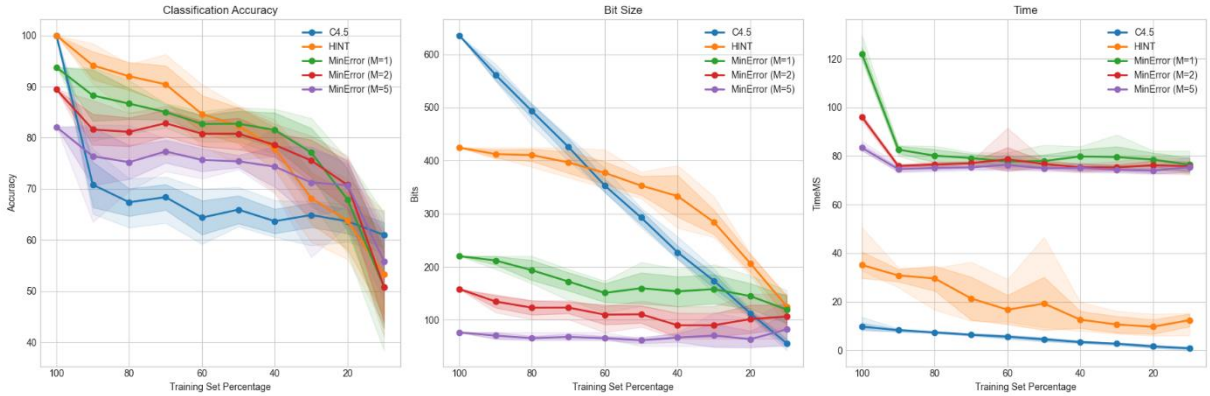

Figure 14: Performance of the algorithms on the Balance dataset.

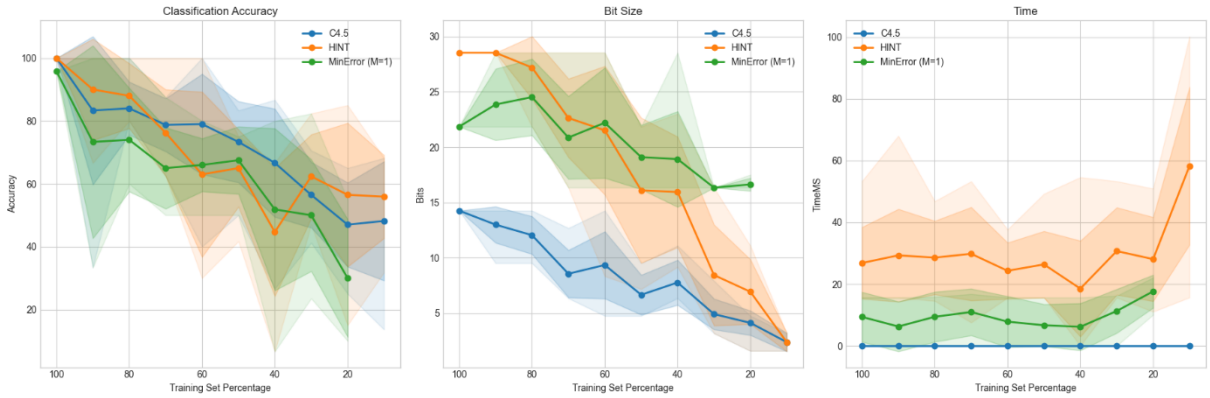

Figure 15: Performance of the algorithms on the Lenses dataset.

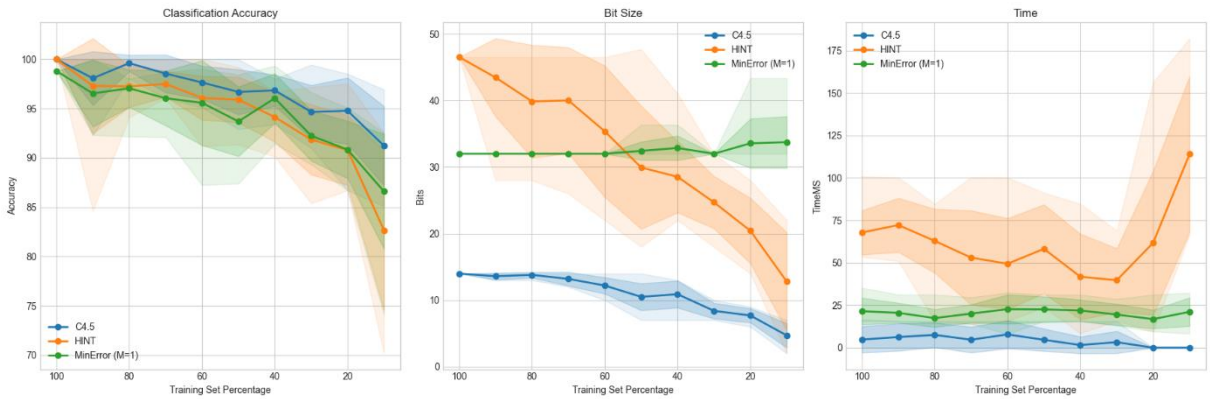

Figure 16: Performance of the algorithms on the SHUT dataset.

4.4 Model revision

Model revision was tested on the *Churn_Rev0* and *Skiers* models. The former use case was designed by Žnidaršič and Bohanec (2004) to test their revision algorithm proposed at that time. The model is aimed at the selection of a family car and consists of two subtrees (Figure 17): (1) "costs", further decomposed in "price" and "maintenance", and (2) "safety", split to "ABS" (Anti-lock Braking System) and "size" (of the car). In addition to the model itself (`Car_Rev0.dxi`), there is a training set

`Car_Rev1.tab` that contains "new evidence" in the sense that some of its data items follow different rules of "car" evaluation than currently encoded in the model. Classification accuracy of `Car_Rev0.dxi`, measured on `Car_Rev1.tab`, is 57.41%. The task is to revise the model's decision rules so as to better adapt it to the new evidence.

Attribute	Scale
car	bad; acceptable; middle; good; *very good*
├─costs	high; middle; *low*
│ ├─price	high; middle; *low*
│ └─maintenance	high; middle; *low*
└─safety	bad; acceptable; good; *very good*
├─ABS	no; *yes*
└─size	small; middle; *big*

Figure 17: Model Car_Rev0: Structure and value scales of attributes.

Running

```
DexRevise Car_Rev0.dxi Car_Rev1.tab /option
```

with different options: none (default), `/progressive`, `/quick`, `/median`, `/average` and `/monotone`,

produces six essentially equal and correct models that all achieved 100% classification accuracy on the training data. Table 1 shows the number of changes made to the model, and elapsed execution time. In all aspects, the reconstruction was successful.

Table 6: Revision of Car_Rev0: Number of changes and execution times for various algorithmic parameters.

Option	Changes	Execution time [ms]
default	7	376.68
`/progressive`	9	29.00
`/quick`	7	21.73
`/median`	7	296.16
`/average`	7	126.91
`/monotone`	7	202.82

While *Car_Rev0* is a small model and seemingly simple for revision, *Skiers* is larger, realistic and fairly difficult. This is especially true for the large version of the model, called *SkiersEnhanced*.

There are two *Skiers* models (Bohanec and Delibašić, 2015): *SkiersBasic* (6 basic and 3 aggregate attributes) and *SkiersEnhanced* (14 basic and 7 aggregate attributes). Both have 3 output values that assess the risk of an injury occurring on a ski slope on a given day. The output values are "L", "M" and "H", standing for Low, Medium, and High. Both models were developed by experts and do not necessarily represent the actual principles that affect skiers' injuries. There is also a dataset *SkiStatsDaily*, collected over the period of six years at the largest Serbian ski resort Mount Kopaonik. The data includes weather data (temperature, wind speed, cloudiness) and data about the utilization of ski lifts and slopes (number of skiers, percentage of ski lift utilization, number of skiers' runs, etc.). This data is connected with the daily number of ski injuries that occurred at the resort. The task of this revision is to change the two DEX models to better correspond to actual events on the ski slopes, as represented in the dataset.

Table 7 shows the classification accuracies, measured on the training data, of the original models, and those revised using various settings (options) of the revision algorithm. For each revised model, the number of changes and approximate execution times are shown.

Table 7: Revision of Skiers models. "Original" denotes the original, expert-developed model.

Model	Option	Classification Accuracy [%]	Changes	Execution Time
Basic	*original*	*45.95*		
	default	67.44	16	5 min
	`/progressive`	67.27	19	12 s
	`/quick`	53.72	2	16 ms
	`/median`	66.45	12	86 s
	`/average`	65.62	10	33 s
	`/monotone`	66.78	13	10 s
Enhanced	*original*	*58.68*		
	default	73.72	26	15.5 hrs
	`/progressive`	72.73	28	88 s
	`/quick`	64.63	1	47 ms
	`/median`	73.22	23	6.6 hrs
	`/average`	70.74	16	6.2 min
	`/monotone`	68.93	14	122 s

In all cases, the classification accuracy of the original models was substantially improved, but only up to a certain limit, indicating that the risk assessment task is inherently difficult. The basic and enhanced models were improved to 67.44% and 73.72% classification accuracy, respectively. The best results were achieved by the default algorithm variant (with no options specified). However, this is also the most time-consuming variant. While the execution times for the basic model were all reasonable due to its small size, some runs on the larger enhanced model took several hours, making this approach impractical in practice. In such cases, it is necessary to use other, more efficient variants of the algorithm, at the cost of somewhat lower classification accuracy.

5 Conclusions

The new DexLearning software package addresses important needs that are essential for many use cases of the DEX methodology, but have remained unmet due to the absence of operational software tools. The package implements three main algorithms: HINT and MinError for learning DEX models from data, and Model Revision for adapting existing DEX models to new data. The availability of DexLearning itself represents an important contribution to the field. All the algorithms were designed so that they produce DEX models in the format of `.dxi` files, which are immediately readable by the DEXi and DEXiWin software, so that they can be inspected and possibly edited by decision makers and decision analysts. This is particularly important in practical applications, because HINT and MinError cannot propose meaningful names for newly created attributes and their values. Consequently, the resulting models must be further verified, interpreted, and completed by the users.

In this study, we experimentally evaluated these algorithms using 34 datasets obtained by converting DEX models into flat decision tables and 16 datasets obtained from public data repositories. The results confirmed that DexLearning is useful for carrying out a range of model construction, reconstruction, and revision tasks across a wide range of dataset sizes. However, the software also has limitations. Both models and data are restricted to qualitative (discrete, non-numeric) variables and values. Furthermore, the combinatorial nature of the algorithms limits their applicability to moderate-sized problems. Based on the results of this study, we estimate that the practical limit is approximately 20 basic attributes, beyond which typical execution times on a desktop computer increase to minutes and even hours.

The algorithms are robust and can generate high-quality DEX models from datasets of varying sizes. HINT performs particularly well on noiseless data that that can be appropriately represented using DEX models. This applies to all datasets generated from DEX models, as well as to some non-DEX datasets. On non-DEX data, HINT competes well with C4.5. HINT-generated models often have relatively low information content (bit size), which is advantageous for model comprehensibility from the decision maker's perspective. Furthermore, visual comparisons of the learning curves show that HINT performs similarly to the results reported in previous studies by Zupan (1997), Zupan et al. (1999), and Bohanec and Zupan (2004), providing additional evidence that the DexLearning implementation is correct.

MinError generally develops models with lower classification accuracy than HINT. Its performance also depends on the setting of the m-estimate, which was found to reduce classification accuracy when increased from 1 to 2 or 5. On the other hand, unlike HINT, MinError can handle datasets containing noise and conflicting outcome assignments. In DexLearning terminology (Bohanec, 2026), it uses `ProbTables` rather than `IntTables` on input. Despite its generally lower classification accuracy, MinError is essential for learning DEX models from noisy data, particularly when the initial loss in accuracy can subsequently be compensated by model revision.

Model revision remains the least tested component of DexLearning. Only two use cases were found to be suitable for this analysis. The smaller one, Car_Rev0, was successfully revised to the expected ideal form. The larger one, Skiers, resulted in a substantial improvement of classification accuracy, but also revealed high execution times for some algorithm variants, which should be addressed in future work. The most pressing challenge is to improve the algorithm's convergence while maintaining a manageable set of promising candidate models at each iteration.

6 References

Aha, D. W. (1991): Incremental constructive induction: An instance-based approach. *Proceedings of the Eighth International Workshop on Machine Learning*, Evanston, ILL: Morgan Kaufmann. 117–121)

Bohanec, M., Kapus, V., Leskošek, B., Rajkovič, V. (eds.) (1997): *Talent: ekspertni sistem za usmerjanje otrok in mladine v športne panoge, uporabniški priročnik*, Ljubljana; Ministrstvo za šolstvo in šport: Zavod Republike Slovenije za šolstvo, 97 str.

Bohanec, M., Cestnik, B., Rajkovič, V. (1998): Evaluation models for housing loan allocation in the context of floats, in *Context sensitive decision support systems* (eds. Berkeley, D., Widmeyer, G.R., Brezillon, P., Rajkovič, V.), 174-189, London: Chapman & Hall.

Bohanec, M., Zupan, B. (2004): A function-decomposition method for development of hierarchical multi-attribute decision models. *Decision Support Systems* 36(3), 215–233. https://doi.org/10.1016/S0167-9236(02)00148-3.

Bohanec, M., Delibašić, B. (2015): Data-mining and expert models for predicting injury risk in ski resorts. In B. Delibašić, J. E. Hernández, J. Papathanasiou, F. Dargam, P. Zaraté, R. Ribeiro, S. Liu, & I. Linden (Eds.), *Decision Support Systems V – Big Data Analytics for Decision Making*, 46–60. Springer International Publishing. https://doi.org/10.1007/978-3-319-18533-0_5.

Bohanec, M. (2017): Multi-criteria DEX models: An overview and analysis. *SOR-2017: 14th International Symposium on Operational Research in Slovenia*, Bled, Slovenia, September 27-29, 2017 (eds. Zadnik Stirn, L., et al.), Ljubljana: Slovenian Society Informatika, Section for Operational Research, 155-160, 2017. https://kt.ijs.si/MarkoBohanec/pub/2017_SOR_DEXmodels.pdf.

Bohanec, M. (2022): DEX (Decision EXpert): A qualitative hierarchical multi-criteria method. *Multiple Criteria Decision Making* (ed. Kulkarni, A.J.), Studies in Systems, Decision and Control 407, Singapore: Springer, 39–78. https://doi.org/10.1007/978-981-16-7414-3

Bohanec, M. (2026): *DexLearning: Software for Learning and Revision of DEX Models from Data*. Report IJS DP-15587, Ljubljana: Jožef Stefan Institute. https://kt.ijs.si/MarkoBohanec/pub/2026_DP15587_DexLearning.pdf.

Cendrowska, J. (1987): PRISM: An algorithm for inducing modular rules. *International Journal of Man-Machine Studies* 27, 349–370.

Delibašić, B., Radovanović, S., Bohanec, M., Suknović, M. (2025): A comparison between DSS and ML models for churn prediction. ICDSST 2025, *11th International Conference on Decision Support System Technology*. Belgrade, Serbia, 43–49. https://kt.ijs.si/MarkoBohanec/pub/2025_ICDSST_Churn.pdf.

Greco, S., Ehrgott, M., Figueira, J. (Eds.) (2016): *Multi Criteria Decision Analysis: State of the art Surveys*. New York: Springer. http://dx.doi.org/10.1007/978-1-4939-3094-4.

Olave, M., Rajkovič, V., Bohanec, M.: (1989): An application for admission in public school systems, in *Expert systems in public administration* (eds. Snellen, I.Th.M., van de Donk, W.B.H.J., Baquiast, J.-P.), 145-160, Elsevier.

Pazzani, M. (1991): The influence of prior knowledge on concept acquisition: Experimental and computational results. *Journal of Experimental Psychology: Learning Memory & Cognition* 17(3), 416–432.

Quinlan, J. R. (1986): *Induction of Decision Trees*. *Machine Learning* 1, 81–106. https://link.springer.com/article/10.1007/BF00116251.

Quinlan, J. R. (1993): *C4.5: Programs for Machine Learning*. San Mateo, CA: Morgan Kaufmann Publishers.

Rodriguez-Ulloa, R., Bohanec, M. (2026): Enhancing the systems approach with artificial intelligence: A Peruvian experience. *Anticipating Socio-Ecological Impacts of Artificial Intelligence*, Proc. UES-EUS 2025, (eds. A. Acevedo-De-los-Ríos et al.), LNNS 1869, Springer Nature Switzerland, 403-418. https://doi.org/10.1007/978-3-032-19495-4_24.

Siegler, R. S. (1976): Three Aspects of Cognitive Development. *Cognitive Psychology* 8, 481–520.

Thrun, S., Bala, J., Bloedorn, E., Bratko, I., Cestnik, B., Cheng, J., de Jong, K., Dzeroski, S., Fisher, D.H., Fahlman, S.E., Hamann, R., Kaufman, K.A., Keller, S., Kononenko, I., Kreuziger, J.S., Michalski, R.S., Mitchell, T.A., Pachowicz, P.W., Vafaie, H., Van de Welde, W., Wenzel, W., Wnek, J., Zhang, J. (1991): The MONK''s Problems-A Performance Comparison of Different Learning Algorithms, CMU-CS-91-197, Sch.

Zupan, B. (1997): *Machine learning based on function decomposition*. Doctoral dissertation. University of Ljubljana, Faculty of Computer and Information Science.

Zupan, B., Bohanec, M., Demšar, J., Bratko, I. (1999): Learning by discovering concept hierarchies. *Artificial Intelligence* 109(1–2), 211–242. https://doi.org/10.1016/S0004-3702(99)00008-9.

Žnidaršič, M. (2007): *Revizija verjetnostnih večparametrskih hierarhičnih modelov*. Doctoral dissertation. University of Ljubljana, Faculty of Computer and Information Science.

Žnidaršič, M., Bohanec, M. (2004): Revision of qualitative multi-attribute decision models. In R. Meredith, G. Shanks, D. Arnott, S. Carlson (Eds.), *DSS2004: IFIP International Conference on Decision Support Systems: Decision Support in an Uncertain and Complex World*, 881–888.

Žnidaršič, M., Bohanec, M., Zupan, B. (2009): Data-driven revision of decision models. In J. Wang (Ed.): *Encyclopedia of Data Warehousing and Mining*, 2nd Edition, 617–623. IGI Global. https://doi.org/10.4018/978-1-60566-010-3.

UCI Datasets

Aha, D. (1991): *Tic-Tac-Toe Endgame* [Dataset]. UCI Machine Learning Repository. https://doi.org/10.24432/C5688J.

Cendrowska, J. (1987): *Lenses* [Dataset]. UCI Machine Learning Repository. https://doi.org/10.24432/C5K88Z.

National Poll on Healthy Aging (NPHA) [Dataset] (2017): UCI Machine Learning Repository. https://doi.org/10.3886/ICPSR37305.v1.

Pazzani, M. (1991): *Balloons* [Dataset]. UCI Machine Learning Repository. https://doi.org/10.24432/C5BP4D.

Shuttle Landing Control [Dataset]. (1988): UCI Machine Learning Repository. https://doi.org/10.24432/C57S34.

Siegler, R. (1976): *Balance Scale* [Dataset]. UCI Machine Learning Repository. https://doi.org/10.24432/C5488X.

Wnek, J. (1993): *MONK's Problems* [Dataset]. UCI Machine Learning Repository. https://doi.org/10.24432/C5R30R.