Revisiting the Data-Driven Revision of DEX Models

Marko Bohanec†
Dept. of Knowledge Technologies
Jožef Stefan Institute
Ljubljana, Slovenia
marko.bohanec@ijs.si

Martin Žnidaršič
Dept. of Knowledge Technologies
Jožef Stefan Institute
Ljubljana, Slovenia
martin.znidarsic@ijs.si

Sandro Radovanović
Faculty of Organizational Sciences
University of Belgrade
Belgrade, Serbia
sandro.radovanovic@fon.bg.ac.rs

Boris Delibašić
Faculty of Organizational Sciences
University of Belgrade
Belgrade, Serbia
boris.delibasic@fon.bg.ac.rs

Abstract

Model revision is a process of adapting an existing model to new evidence. In this study, we investigate the revision of qualitative multi-criteria DEX (Decision EXpert) models. This work builds on the pioneering research of Žnidaršič et al. from the early 2000s, which we revisit here, extending the revision schema to make use of separate learning and validation data sets, and investigating four variants of the revision algorithm. The new implementation is evaluated experimentally on the *Skiers* use case, which aims to predict skiers' injury risk. The obtained revised models achieve classification accuracies comparable to results previously reported using classical machine learning and DEX modeling.

Keywords

Machine learning, Expert modeling, Data-driven model revision, DEX (Decision EXpert) models, "Skiers" use case

1 Introduction

Many predictive and decision-support models used in practice are initially constructed from expert knowledge rather than learned directly from data [1]. As additional observations become available, these models often require revision in order to improve predictive performance, while preserving their original structure and interpretability [2][3]. *Data-driven model revision* is a process of adapting some initial model to accommodate for the new data. Such model revision combines human expertise with empirical evidence and represents an important bridge between knowledge engineering and machine learning.

The DEX (Decision EXpert) method is a qualitative multi-criteria decision analysis approach [5], where all attributes are qualitative and represented by words, attributes are organized hierarchically, and evaluation 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. However, there is a clear practical need for DEX model revision in application domains where data is available, but is not good or abundant enough for pure data-based modelling, requiring the integration of expert knowledge with available data. The *Skiers* use case [6][7], discussed in Section 4, is one such example. Similar situations arise in customer churn prediction [8] and medical therapy formulation [9], where expert-built DEX models often require revision based on empirical data.

The seminal work on revising DEX models was conducted by Žnidaršič et al. in the early 2000s [10][11]. Their approach addressed the challenge of updating DEX models when new data becomes available. This foundational research established a framework for reconciling expert knowledge with data-driven insights. However, despite its contributions and practical needs, this line of research subsequently lost momentum, leaving open questions regarding algorithmic extensions and the development of a user-oriented software implementation.

This paper revisits the data-driven revision of DEX models with three primary objectives: (1) exploring new algorithmic variants including different search strategies and monotonicity preservation techniques, (2) implementing these algorithms, and (3) experimentally evaluating them using the *Skiers* problem as a test case.

2 Problem Definition

Data-driven revision of DEX models is defined as follows (Figure 1): given an existing DEX model M and a set of data items R, revise the model to match the data items as closely as possible. The process is aimed at constructing the new model M' that maximizes some quality measure $Q(m, R)$, where m is a model whose quality is measured on R. After revision, it is expected that $Q(M', R) > Q(M, R)$.

In this study, we adopt classification accuracy as the quality measure. *Classification accuracy* $CA(m, R)$ is defined as the percentage of data items from R that are correctly classified by m. Other quality measures that are commonly used in machine learning [12], such as precision, recall or F1-Score, could have been used as well.

The data set R must be *compatible* with M: it should contain multiple items of the form $(\boldsymbol{x}, y)$, where $\boldsymbol{x}$ is a vector of values that correspond to all input (basic) attributes of M, and y is one of possible values that can be taken by the output attribute of M.

Even though the revision, as defined above, requires "M' to match R as closely as possible", the process is sensitive to *overfitting*, a phenomenon often encountered in machine learning [12]: it may happen that M' adapts too well to R, achieving very

†Corresponding author.

Permission to make digital or hard copies of part or all of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. Copyrights for third-party components of this work must be honored. For all other uses, contact the owner/author(s).
Information Society 2026, 5–9 October 2026, Ljubljana, Slovenia
© 2025 Copyright held by the owner/author(s).
http://doi.org/10.70314/is.2026.skui.40

high quality $Q(M', R)$, but performing comparatively poorly on some future, so far unseen, data. To alleviate the problem, we adopt the common machine learning solution and introduce two data sets, learning (L) and validation (V) (Figure 2): the former is used to guide the modifications of M, whereas the latter serves for assessing and maximizing the quality $Q(M', V)$. Usually, L and V are obtained by splitting R so that $R = L \cup V$. Conversely, the basic revision schema (Figure 1) is a special case of the extended one (Figure 2), where $R = L = V$.

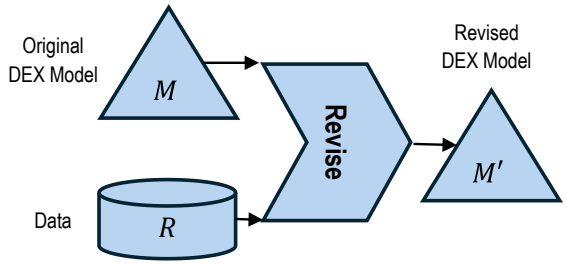


Figure 1: Basic schema of data-driven DEX model revision

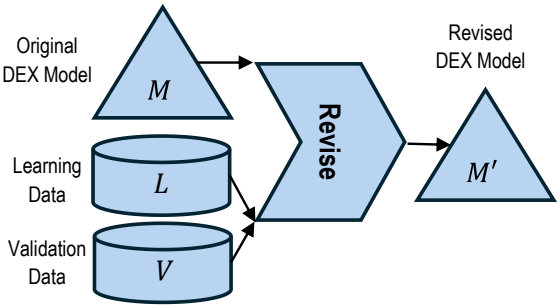


Figure 2: Extended schema of data-driven DEX model revision, using learning (L) and validation (V) data

The transition from M to M' is composed of one or more *admissible transformations*. In DEX models, one may think of modifying the model structure or modifying decision rules. Hereafter we follow the approach of [10] and facilitate only the *modification of decision rules*, while leaving the model structure intact.

Whenever some learning data item $(\boldsymbol{x}, y) \in L$ disagrees with the current model M so that $M(\boldsymbol{x}) \neq y$, it is necessary to identify *transformation candidates* that may improve the quality of M. Let us illustrate the concept using a simple synthetic DEX model in Figure 3. The model has three input attributes $X1, X2$ and $X3$, and one output attribute Y. There is an intermediate (aggregate) attribute $X12$ that combines $X1$ and $X2$. The aggregation is defined by two decision tables. In principle, the model is meant to represent the mapping $Y = \text{MAX}(\text{MIN}(X1, X2), X3)$, but it contains an error to be discovered by revision.

To this end, let us consider the data item $(X1 = \text{med}, X2 = \text{high}, X3 = \text{med}, Y = \text{med})$, which represents a correct instance of the mapping. However, the model evaluates this data item as $Y = \text{high}$, which can be verified in Figure 3 by examining rule 4 in the bottom decision table, which yields $X12 = \text{med}$, and rule 5 in the top decision table, incorrectly yielding $Y = \text{high}$. Therefore, we need to revise the model so as to yield $Y = \text{med}$, as indicated by the data item.

Which are the transformation candidates? One possibility to achieve the final value $Y = \text{med}$ is to change the value of rule 5 in decision table Y from high to med. This is the only option in that decision table. It is also the correct one, but the algorithm does not know that. Another possibility is to investigate candidates recursively in the subordinate decision tables ($X12$ in this case). Rule 2 in table Y indicates an alternative way to achieve $Y = \text{med}$: change $X12$ to low. Looking in table $X12$ and considering that $X1 = \text{med}$ and $X2 = \text{high}$, we can achieve that by setting the value of rule 4 to low. In summary, we have identified two transformation candidates: changing rule 5 of Y to med, or changing rule 4 of $X12$ to low.

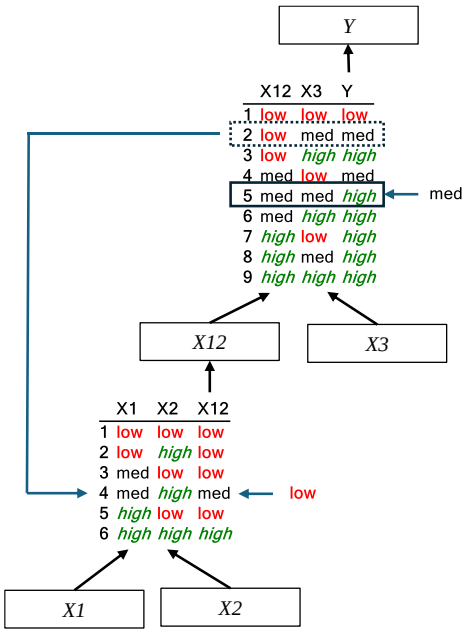


Figure 3: A synthetic DEX model, illustrating the identification of transformation candidates based on data item ($X1 = \text{med}, X2 = \text{high}, X3 = \text{med}, Y = \text{med}$)

Attributes in DEX are usually, but not necessarily, ordered. The ordering implies that the corresponding decision tables are *monotone*: when the value of an ordered attribute increases, the decision table outcome increases, too, or remains constant. In Figure 3, both decision tables are monotone, and both candidate transformations do preserve monotonicity. Monotonicity is considered in a subvariant of the revision algorithm (Section 3), which restricts transformations only to those that preserve the monotonicity of decision rules.

3 DEX Model Revision Algorithm

Having defined the problem and basic concepts, we can now turn to the main revision algorithm. We base our approach on the previous approaches [10][11], but introduce some changes and algorithmic variants. The distinction between sets L and V in Figure 2 is already one of the proposed extensions.

The original approach [10] considered one data item at a time and was sensitive to the incoming order of data items. In the proposed approach, we first traverse the whole data set L, identify candidate transformations for all data items that disagree

with M, and assemble a statistics of candidate value changes for each decision rule: each change receives an associated *weight*, i.e., a count of data items that induced the change. The algorithm is iterative, keeping a set of most prospective models $\mathcal{M}$ in each iteration. In each iteration, one modification of each $M \in \mathcal{M}$ is made, until no further improvement of $Q(M,V)$ is possible. In order to avoid cycles, the algorithm keeps track of already investigated models and prevents reconsidering any of the previous ones.

Algorithm 1: Data-Driven DEX Model Revision

Given:	M: initial DEX model L: learning data set V: validation data set
Output:	$bestM$: best revised DEX model $bestQ = Q(bestM,V)$
Variants:	default, `quick`, `progressive`, `monotone`

```
1  bestM := M
2  bestQ := Q(bestM,V)
3  M := {M} // set of models considered in each iteration
4  while M ≠ ∅:
5  |  nextM := ∅ // candidate models for the next iteration
6  |  iterQ := bestQ // accuracy threshold for the current iteration
7  |  for each m ∈ M:
8  |  |  C := CandidateChanges(m,L)
9  |  |  for each c ∈ C:
10 |  |  |  if ChangeAllowed(m,c):
11 |  |  |  |  newM := ApplyChange(m,c)
12 |  |  |  |  newQ := Q(newM,V)
13 |  |  |  |  if newQ > bestQ: // newM is the best model so far
14 |  |  |  |  |  bestM := newM
15 |  |  |  |  |  bestQ := newQ
16 |  |  |  |  if newQ > iterQ: // newM is a candidate for next iter.
17 |  |  |  |  |  if progressive: iterQ := newQ; nextM := ∅
18 |  |  |  |  |  nextM := nextM ∪ {newM}
19 |  M := nextM
```

The top-level pseudocode is displayed as Algorithm 1. The main loop (steps 4–19) checks the current model candidates $m \in \mathcal{M}$ and keeps track of the best model $bestM$ with best quality $bestQ$ achieved that far, while also preparing candidate models $nextM$ for the next iteration. In step 8, candidate changes of each model M are identified by traversing the set L and using the approach sketched in Section 2 and Figure 3.

The condition $ChangeAllowed(m,c)$ in step 10 determines whether the change c is allowed on model m. The main condition is that c must not revert m to one of the already investigated models. Additionally, $ChangeAllowed$ considers additional conditions that constitute the subvariants of the algorithm (a subvariant is selected at the invocation):

- `quick`: consider only changes $c \in C$ with the maximum *weight* in C;
- `monotone`: change c is allowed only if it preserves the monotonicity of changed rule r with respect to all comparable rules in the corresponding decision table.

By default, the set of models $\mathcal{M}$ in each iteration is kept as large as possible. After setting $iterQ$, the accuracy threshold for the current iteration, all models $newM$ that improve over that measure are included for consideration in the next iteration. This is to ensure that revision paths to obtaining the best final solution are thoroughly investigated. In some cases, however, this may lead to a very extensive and time-consuming search. To speed up the process, the `progressive` subvariant of the algorithm takes place in step 17, progressively raising the threshold $iterQ$ for the current iteration and discarding models already found in that iteration. This generally results in fewer models being included in the next iteration; however, it may miss some good model candidates and, consequently, final solutions.

4 Experimental Evaluation: *Skiers* Use Case

Skiers is a use case aimed at predicting the risk of injuries in ski resorts [6]. The project is based on data 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. In order to support the rescue services, the goal is to predict the risk of ski injuries considering the current situation on the slopes. Several predictive models were developed to this end, using classical machine learning [6], DEX expert modelling [6] and machine learning [7].

a. Basic model

Attribute	Scale
Skiers	H;M;L
├Crowding	H;M;L
├numSkiers	H;M;L
├numPasses	H;M;L
└utilization	H;M;L
└Weather	H;M;L
├tempAvg	L;M;H
├windSpeed	L;M;H
└cloudiness	L;M;H

b. Enhanced model

Attribute	Scale
Skiers	H;M;L
├Crowding	H;M;L
├numSkiers	H;M;L
├numPasses	H;M;L
└utilization	H;M;L
├Weather	H;M;L
├tempAvg	L;M;H
├windSpeed	L;M;H
└cloudiness	L;M;H
└Skiing	H;M;L
├TimeRun	H;M;L
├avgAvgTimeRun	H;M;L
├avgMinTimeRun	H;M;L
└avgMaxTimeRun	H;M;L
├Elevation	H;M;L
├avgAvgElevation	H;M;L
├avgMinElevation	L;M;H
└avgMaxElevation	H;M;L
└SpeedRun	H;M;L
├avgNoRuns	M;L;H
└avgSpeedRun	M;L;H

Figure 4: The structure of the two "Skiers" DEX models

In this experimental evaluation we consider two DEX models from the study [6]. Their structure is shown in Figure 4. Both models were developed manually. The *Basic model* was developed by a domain expert (ski resort safety manager). The model is simple and uses only 6 of the 14 attributes available in the data set. It has a clear and comprehensible structure of two subtrees, consisting of three Crowding and three Weather criteria, respectively. The *Enhanced model* was developed by a decision-support expert. It adds the Skiing subtree (with three subtrees) and uses all the available attributes.

The experiments were carried out by running:

- all the four DEX revision algorithm variations (default, `quick`, `progressive`, `monotone`)
- on both models (*Basic* and *Enhanced*)
- using basic (R, Figure 1) and extended (L+V, Figure 2) data schemas.

The full data set (R) contained 605 daily observations over the period of six years. R was split in the learning set L containing 490 items of the first five years, and validation set V with 115 items of the sixth year.

We observed (Table 1): the achieved classification accuracy (CA) of the best revised model, number of changed decision rules (equal to the number of main iterations of the algorithm), and approximate execution time on a 3.70 GHz desktop computer, single-core computation.

Table 1: Results of experimental evaluation

Model	Variant	Schema	CA [%]	Changes	Time
Basic	*original*	*R*	*45.95*		
	default	*R*	67.44	16	5 min
	`progress.`	*R*	67.27	19	12 s
	`quick`	*R*	53.72	2	16 ms
	`monotone`	*R*	66.78	13	10 s
	original	*L+V*	*46.33*		
	default	*L+V*	69.80	16	3.4 min
	`progress.`	*L+V*	68.78	12	4.5 s
	`quick`	*L+V*	64.69	4	31 ms
	`monotone`	*L+V*	68.57	11	4.8 s
Enhanced	*original*	*R*	*58.68*		
	default	*R*	73.72	26	15.5 hrs
	`progress.`	*R*	72.73	28	88 s
	`quick`	*R*	64.63	1	47 ms
	`monotone`	*R*	68.93	14	122 s
	original	*L+V*	*59.18*		
	default	*L+V*	74.29	22	3.7 hrs
	`progress.`	*L+V*	74.49	28	60 s
	`quick`	*L+V*	65.71	1	31 ms
	`monotone`	*L+V*	71.22	14	63 s

The results indicate profound effects of DEX model revision, which substantially improves the classification accuracy from about 46% to almost 70% (*Basic model*), and 59% to almost 75% (*Enhanced model*). This is comparable to results achieved in [6] by DEX modelling (66.09–75.65%) and classical machine learning (65.22–78.62%).

The default variant of the algorithm performs an extensive search of the solution space and is computationally expensive, often prohibitively so. However, this exhaustive search produced the most accurate models in three of four cases. The `progressive` variant offers a particularly favorable trade-off between efficiency and accuracy: it achieves similar classification accuracy compared to the default variant, while reducing execution time from hours to seconds. In contrast, the `quick` variant sacrifices too much accuracy for speed, terminating the search prematurely and failing to match the best-performing models. The `monotone` variant represents a special case that explores only the subset of monotone solutions. Consequently, it is faster but may produce less accurate models. Its main advantage is that it guarantees the monotonicity of the generated decision rules, a property that is usually highly desirable in practical decision support applications [9].

5 Conclusions

The main goal of this work was to revive the data-driven revision of DEX models and advance it to the level of practical application. The proposed approach is now fully operational, and the initial results indicate that the enhanced algorithms achieve classification performance on a par with other machine learning and DEX learning methods.

This work is still in progress. Although the default variant consistently finds models of high quality, it remains computationally expensive even for moderately sized problems. Future work shall focus on improving its efficiency while preserving its ability to identify high-quality solutions that may be overlooked by the `progressive` variant. We also plan to extend the experimental evaluation to additional application domains, including customer churn prediction [8] and medical therapy formulation [9]. Beyond classification accuracy, we intend to analyze the generated decision rules in greater detail. Finally, we plan to release the algorithm as open-source software.

Acknowledgments

Marko Bohanec and Martin Žnidaršič acknowledge the financial support from the Slovenian Research and Innovation Agency for the programme Knowledge Technologies (research core funding No. P2-0103 and P5-0018). Sandro Radovanović and Boris Delibašić acknowledge the financial support by the University of Belgrade – Faculty of Organizational Sciences and, in part, by the Ministry of Science, Technological Development and Innovation of the Republic of Serbia through institutional funding (grant number: 200151).

References

[1] Wil van der Aalst, 2016: *Process Mining - Data Science in Action*, 2nd Edition, Springer. DOI: https://doi.org/10.1007/978-3-662-49851-4.

[2] Stefan Wrobel, 1996. First-order theory refinement. In L. De Raedt (Ed.), *Advances in Inductive Logic Programming*. IOS Press, 14–33.

[3] Kate Revoredo, 2021. *Process Model Repair Meets Theory Revision - Initial Ideas*. 14th IFIP Working Conference on The Practice of Enterprise Modeling (PoEM), Riga, Latvia, 184–194, DOI: https://doi.org/10.1007/978-3-030-91279-6_13.

[4] Pascal Bercher, Sarath Sreedharan, Mauro Vallati, 2025. A survey on model repair in AI planning. *Proc. 34th International Joint Conference on Artificial Intelligence IJCAI 2026*, 10371–10380. https://doi.org/10.24963/ijcai.2025/1152.

[5] Marko Bohanec, 2022. DEX (Decision EXpert): A qualitative hierarchical multi-criteria method. In A. J. Kulkarni (Ed.), Multiple Criteria Decision Making, 39–78. Springer Nature Singapore. https://doi.org/10.1007/978-981-16-7414-3_3.

[6] Marko Bohanec and Boris Delibašić, 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.

[7] Sandro Radovanović, Marko Bohanec, Boris Delibašić, 2023. Extracting decision models for ski injury prediction from data. *International Transactions in Operational Research* 30(6), 3429–3454. https://doi.org/10.1111/itor.13246.

[8] Boris Delibašić, Sandro Radovanović, Marko Bohanec, Milija Suknović, 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.

[9] Marko Bohanec, 2021. From data and models to decision support systems: Lessons and advice for the future. In J. Papathanasiou, P. Zaraté, J. Freire De Sousa (Eds.), *EURO Working Group on DSS*, 191–211. Springer International Publishing. https://doi.org/10.1007/978-3-030-70377-6_11.

[10] Martin Žnidaršič, Marko Bohanec, 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.

[11] Martin Žnidaršič, Marko Bohanec, Blaž Zupan, 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.

[12] Norman Matloff, 2024. *The Art of Machine Learning: A Hands-On Guide to Machine Learning with R*. San Francisco: No Starch Press. ISBN-13: 978-1-7185-0211-6.