Declarative Knowledge Discovery in Databases via Meta–Learning – Towards Advanced Analytics

Dietmar Seipel[1] and Martin Atzmüller[2]

[1] University of Würzburg, Department of Computer Science,
Knowledge–Based Systems Group (KBS)
Am Hubland, 97074 Würzburg, Germany
dietmar.seipel@uni-wuerzburg.de

[2] Osnabrück University, Institute of Computer Science
Semantic Information Systems Group (SIS)
Wachsbleiche 27, 49090 Osnabrück, Germany
martin.atzmueller@uni-osnabrueck.de

Abstract. Knowledge discovery in databases is typically an exploratory and semi–automatic process, with a human–in–the–loop. Declarative approaches can considerably support this process, by formalizing and including domain knowledge as well as declaratively specifying the data processing and analytics process. Then, mediator frameworks provide powerful methods and approaches for orchestrating data processing and data mining methods in order to support the full knowledge discovery cycle.

This paper proposes a novel approach for declarative knowledge discovery in databases enabling advanced analytics via the concept of meta–learning. We specifically present how to declaratively apply meta–learning and how to *orchestrate* knowledge discovery methods via the toolkit Declare. We utilize various methods from knowledge discovery in databases (KDD) for detecting patterns in (relational or XML) databases. For these, we present an example case for analyzing tennis data demonstrating the efficacy of the presented approach.

1 Introduction: Knowledge Discovery in Databases (KDD)

There exists a plethora of approaches and methods for knowledge discovery in databases (KDD) [28, 29] to reveal previously unknown patterns, e. g., [51, 61], including classification as well as association rule mining [1, 34, 48], clustering [65, 77], and subgroup discovery [3, 38, 76] as prominent examples. A current trend is to combine their strengths by so–called *orchestration* via a mediator framework. Orchestration combines different analysis methods/learning algorithms into a pipeline for knowledge discovery, e. g., [41,62]. For the preprocessing of the data (selection and transformation) and the postprocessing of the patterms (evaluation and presentation), declarative logic programming techniques can be very useful. The application of various methods and tools for data mining can be orchestrated and the *selection and transformation* as well as the *evaluation and presentation* of the patterns and the *data analytics* can be supported by declarative methods, e.g. from logic programming. Here, specifically the declarative toolkit Declare can be used as *mediator* to put the building blocks together. A possible KDD architecture is shown in Figure 1.

In knowledge discovery, exploratory data analysis is an important approach for getting first insights into the data. With this, the incremental knowledge discovery process is guided, since typically the goal of the methods is not only an actionable model, but also a human interpretable set of patterns [50]. In such contexts, declarative approaches can be used for formalizing interesting criteria, constraints, as well as for modelling common processes and data analysis patterns in a declarative and orchestrated knowledge discovery approach. Here, the notion of *meta–learning* via declarative programs is an important and flexible concept: While constraints and interestingness criteria can be formalized in a declarative way, these mostly address *static* criteria so far. In this paper, we address *dynamic* orchestration which is implemented via meta–learning in an iterative and incremental process. As shown in Figure 1, in the pattern mining and analysis step we can apply meta–learning in order to apply *dynamic* changes and refinements on the orchestration process. We will demonstrate this approach in an example case for analyzing tennis data.

The rest of the paper is organized as follows: Section 2 introduces background and related work. After that, Section 3 presents our approach. Next, Section 4 discusses our example case. Finally, Section 5 concludes with a summary and interesting directions for future work.

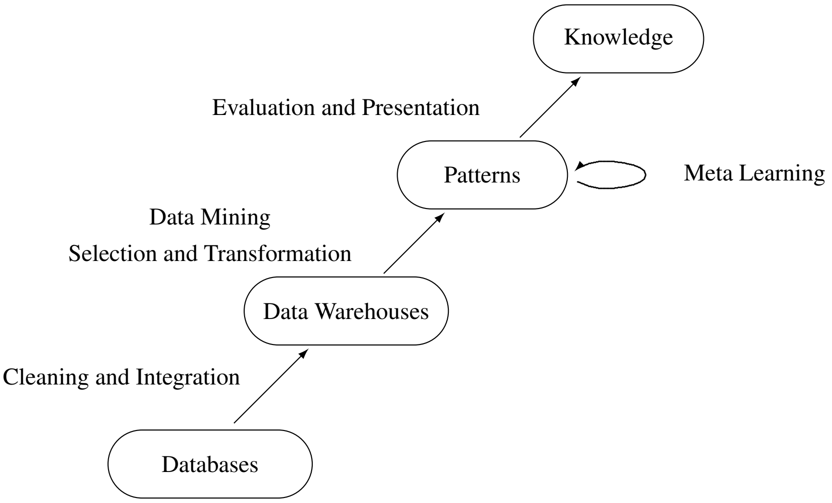


Fig. 1. KDD Architecture and Meta-Learning Integration

2 Background and Related Work

If the user knows which queries to ask to the database, then *online analytical processing* (OLAP) typically uses *aggregation queries* in SQL (SUM, AVG, MIN, MAX) to process spreadsheets (data cubes). But, *frequently we are drowning in data, but starving for knowledge* and the user does not know which patterns to search for in the database. In such cases, exploratory approaches for data mining and knowledge discovery can be applied.

Overall, in such contexts the goal in *relational data mining* is to find highly scalable algorithms, that can also be applied to reveal patterns in very large amounts of data. Furthermore, data mining methods are commonly applied to obtain a set of *novel*, *potentially useful*, and ultimately *interesting* patterns from (large) data sets cf. [27]. This can be achieved e. g., utilizing exploratory data mining techniques like association rule mining or subgroup discovery, as summarized below. Previously unknown patterns in the knowledge can be found without querying by statistical methods. e. g., repeated splitting of a relational table w.r.t. *entropy functions* produces patterns in the form of *decision trees* and classification rules. Association rules can be derived using the well–known *A Priori–Algorithm* for *frequent itemsets*. A classification/association rule r is an uncertain rule over attribute=value pairs $L_i = (A_i = V_i)$ with a support s and a confidence c in the form of an implication $L_1 \wedge \ldots \wedge L_n \overset{s,c}{\Longrightarrow} L_{n+1} \wedge \ldots \wedge L_m$. For larger tables with many attributes or values, usually many rules are derived, and the knowledge engineer has to investigate them.

The *K–Means–Algorithm* [49] clusters groups of similar (logically related) objects, requiring the number of clusters k as a parameter and a distance measure for estimating distances between pairs of objects.

The *SD–Map* algorithm [9,44] for *subgroup discovery* [3,38,76] aims at finding interpretable and interesting patterns, i. e., patterns describing subsets of a dataset that are interesting as estimated by a quality function. That is, in contrast to association rules, subgroups are discovered by such a quality function which can be flexibly defined. In subgroup discovery, a *database* $D = (I, A)$ is given by a set of instances I and a set of attributes A. For nominal attributes, a *basic pattern* $(a_i = v_j)$ is a Boolean function $I \to \{0, 1\}$ that is true if the value of the attribute $a_i \in A$ is equal to v_j for the respective instance. The set of all basic patterns is denoted by Σ. A *subgroup description* or (complex) *pattern* is given by a set $P = \{p_1, \ldots, p_l\}, p_i \in \Sigma, i = 1, \ldots, l$ of basic patterns, interpreted as a conjunction $p_1 \wedge \ldots \wedge p_l$, with $\mathit{length}(P) = l$. A pattern can thus also be interpreted as the *body* of a *rule*. The rule *head* then depends on the property of interest, e. g., for a binary target concept T on a basic pattern $\mathit{sel}_T = \mathsf{t}$. A *subgroup* $S_P = \mathit{ext}(P) = \{\, i \in I | P(i) = \mathsf{t}\,\}$, is the set of all instances that are covered by the subgroup description P. In a top–k setting, a subgroup discovery algorithm returns the top–k subgroups according to a selectable interestingness measure $q: 2^{\Sigma} \to \mathbb{R}$, cf. [3]. For a binary target concept, e. g., the size $n := \mathit{ext}(P)$ of a subgroup described by the pattern P, i. e., its *support*, and the share t_P of the target concept in the subgroup,

i. e., its *confidence*, are combined by the interestingess measures q_S as follows: $q_S(P) = n \cdot (t_P - t_0)$, where t_0 denotes the (default) share of the target concept in the database P, or by the *Lift* quality function $q_L(P) = \frac{t_P}{t_0}$. In general – besides those already mentioned quality functions – many quality functions for a single target feature, e. g., in the binary or numerical case, trade off the size $n = |ext(P)|$ of a subgroup, and the deviation $t_P - t_0$. Thus, standard quality functions are of the form $q_a(P) = n^a \cdot (t_P - t_0),\ a \in [0;1]$. For binary target concepts, this includes, e. g., a *simplified binomial* function $q_a^{0.5}$ for $a = 0.5$, or the *Piatetsky-Shapiro* quality function q_a^1 with $a = 1$, cf. [3].

Exceptional model mining [3, 26] can be seen as a variant of subgroup discovery, focusing on more complex quality functions, i. e., considering complex *target models*, like comparing regression models or graph structures [5]. Essentially, exceptional model mining tries to identify interesting patterns with respect to a local model derived from a set of attributes, cf. [24, 25]. For exceptional model mining, a model consists of a specific *model class* (e. g., a regression or graph–structured model, cf. e. g., [5, 24, 25]), requiring a specific quality function. It applies *model parameters* which depend on the values of the model attributes in the instances of the respective subgroup. The attributes consist of two (usually non–overlapping) sets of describing attributes and model attributes. The goal of exceptional model mining is then to identify patterns, using a specific quality function, for which the model parameters differ significantly from the parameters of the model built from the entire dataset. For a more detailed discussion we refer to [26].

Association rule mining, subgroup discovery and exceptional model mining are prominent methods for local exceptionality detection that can be configured and adapted to various analytical tasks. We can, for example, focus on more complex data such as graphs or sequences, where, e. g., description–oriented community detection using subgroup discovery can be applied, cf. [5]. For providing both structurally valid and interpretable communities we utilize the graph structure as well as additional descriptive features of the graph's nodes. Furthermore, we can focus on sequential patterns [57] as well as temporal episodes [69] which can both be tackled using a combination of pattern mining and declarative techniques.

In addition, we can make use of *condensed* or *concise* representations, which have been first developed in the the field of association rules. This relates to frequent item sets for which condensed representations can be applied in order to reduce the size of the set of association rules that are generated and presented (e. g., [13,20,58,59,64]). These representations are used for the (implicit) redundancy management, since then the condensed patterns also describe the specifically interesting patterns. In this case, the efficiency of the association rule discovery method can also be increased significantly, if the redundancy management is directly incorporated into the algorithm; otherwise, it can typically be applied in a post-processing step. Besides association rules, these techniques can then also be generalized for frequent patterns (cf. [36, 56]). In general, based upon set-theory, condensed representations include *closed-sets*, *free-sets* and *(non-)derivable* sets. We refer to [36, 56] for a detailed description. Furthermore, according adaptations can be applied for subgroup discovery as well [2, 32]. In our context, Declare offers flexible post-processing options on the obtained set of patterns, e. g., using subsumption as discussed below. With the declarative functionality of Declare, flexible post analysis of the obtained patterns and knowledge can be implemented, e. g., [45]. In particular, this allows for implementing declarative and knowledge-driven approaches, for example, including background knowledge for the refinement of the discovered set of patterns, e. g., [4,11,35,52], or for considering causal relations [10,19,23,39,40,53,55]; then, for example, we can identify the subgroups which are causal for the target concept.

In general, local exceptionality detection especially supports the goal of explanation–aware data mining [54] – in line with interpretable and explainable data mining [4, 73] due to its more interpretable results, e. g., for characterizing a set of data, for concept description, for providing regularities and associations between elements in general, and for detecting and characterizing unexpected situations, e. g., events or episodes.

Data warehouses frequently use *declarative query languages*. From knowledge–based information systems, the query languages SQL, XQuery, DATALOG, and SPARQL are well–known: the knowledge engineer does not have to program the instructions for evaluating a query, she/he just specifies the knowledge in an abstract and compact *syntax*. For the logic programming language PROLOG [16, 22] Bob Kowalski has created the slogan ”Algorithm = Logic (what) + Control (how)”. Data warehouses use online analytical processing (OLAP) based on aggregation queries in SQL for computer scientists and graphical user interfaces for domain experts (data cube, spreadsheet), if the user knows which patterns to search for.

3 Method

The declarative toolkit Declare can be used as a mediator to combine meta–learning with declarative orchestration to put all the building blocks together in a flexible declarative way. Between different applications of data mining methods, the basic data can be transformed with Declare, and based on a declarative analysis of the results of previous applications of data mining methods, further data mining methods can be applied (orchestration).

3.1 Meta–Learning and Declarative Knowledge Discovery

In machine learning, the concept of *meta–learning* [17, 43] has been established, e. g., in the area of ensemble learning when improving weak learners. Essentially, *boosting* [30, 31] and bagging [18] focus on the data characteristics together with some evaluation measure of a classifier, for improving the classifier by modifying the originating dataset. In addition, the *selection* of a data mining method, e. g., for classification, association rule mining etc. can also be implemented using meta–learning [14, 71], as well as for formalizing, evaluating and refining a data mining task [7, 12].

In our approach, we extend the concept of meta–learning beyond those proposed in the literature. We extend on those and make use of the full declarative capabilities, e. g., for modfying the data characteristics for data mining and machine learning, for optimizing a given method such as changing a set of input parameters, and by including other important criteria such as interpretability, coverage, redundancy of the given patterns/models etc.

3.2 The Declarative Mediator Declare

The *declarative logic programming toolkit* Declare [66] is an open–source library developed in SWI PROLOG [75] containing a deductive database system DDBASE that can access hybrid databases (relational: SQL, ODBC; XML: XQuery; semantic web; . . .) to produce complex structured answers using a query language DATALOG* for the stratified evaluation of logic programs with embedded PROLOG calls. DDBASE combines PROLOG and DATALOG*. It can process *hybrid knowledge bases* containing relational databases (RDB) and XML documents within the same query using SQL (ODBC) and FNQuery, respectively, see Figure 2. This extends *database programming languages* (DBPL) by XML capabilities.

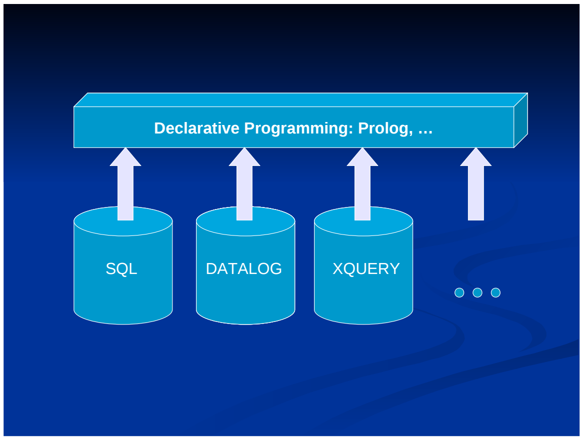


Fig. 2. Declarative Programming for Hybrid Knowledge Bases in DDBASE

Declare has been developed at the Universities of Tübingen and Würzburg; it is availabe publicly and can be installated in a docker container on most common platforms (including Windows, macOS, and Linux). Domain specialists can load data into a hybrid internal *data warehouse* (e.g. in relational or XML format) and then analyse the data using *explainable* techniques [46] from *artificial intelligence* (declararative and online–analytical programming and data mining tools). Declarative logic programming in Declare (DDBASE, DATALOG*, and FNQuery) can be used as a *mediator tool* for managing and connecting hybrid database knowledge.

3.3 Subgroup Discovery with Vikamine

The open–source tool Vikamine[3] [6,8] for subgroup discovery and pattern mining is used in data science, artificial intelligence and complex network analysis. Vikamine is an open environment for intelligent pattern mining and subgroup discovery, which features a variety of state–of–the–art automatic algorithms, visualizations, high extensibility and customization capabilities. It is implemented in Java, but provides various interface solutions, e. g., in an R–implementation[4]; alternatively, it can also be configured and orchestrated via an XML–based interface. Thus, with the latter, Vikamine can be flexibly embedded into data analytics workflows, e. g., in meta–learning and orchestration contexts. This is also the targeted approach in the context of the Declare toolkit.

Vikamine focuses on pattern mining for identifying local exceptional patterns via subgroup discovery; it also supports the variant of exceptional model mining, providing this on complex relational data as well. In this way, Vikamine aims at providing interesting patterns that help to ”make sense” of complex information and knowledge processes. At its core, Vikamine applies subgroup discovery, which aims at identifying subgroups of data instances that are *interesting* with respect to a a specific quality function. A subgroup can be represented by a pattern typically consisting of feature–value pairs, as discussed above. In this way, subgroup discovery provides explicit and *interpretable* patterns, as an instance of interpretable machine learning, e. g., [4,47]. Therefore, this features both interpretability as well as explainability, e. g., [4]. Exemplary applications include e. g., the medical domain [63] as well as industrial applications [70], and the analysis on social interactions and human behavior [21,33,37].

Vikamine, i. e., its core library provides an XML–based interface such that subgroup discovery tasks can be configured using declarative XML specifications for orchestrating Vikamine to be used in larger processing pipelines, e. g., connected to the toolkit Declare.

4 Example Case: Knowledge Discovery in Tennis Data with Declare

For a tennis data warehouse, we have used Declare and the public *data mining* software tools Weka and Vikamine in an extensive *case study* for analysing XML data representing *tennis rallies*. The framework / method proposed in the previous section can be used for knowledge discovery in general, e. g., in projects with relational or XML databases. In the case of the tennis example, tennis trainers could first digitize videos using our tool based on concepts from *neural networks* and *deep learning* utilizing OpenCV [15, 74]; then, trainers can analyse and improve the tactical behaviour of their students using *declarative* data mining concepts. For practical usage, of course, the needs of the users would be to have a stable system running on a common platform such as a Windows PC – or even a smartphone; also, *declarative explanations* and *visualizations* of the obtained results are essential. Our current focus is to provide Declare and the Tennis tool in a *docker container*, as a flexible solution that can be used under different operating systems.

With a graphical user interface, tennis matches are managed and linked with corresponding video sequences. A tactical analysis should reveal promising strategies: The rallies are linked to video sequenes, which we transform to the data format XML using AI techniques (deep learning). Declare has been used for decomposing the tennis court into *tiles*, such that the likelihood that a ball hits a tile becomes positive, whereas it is almost unlikely that the previous numerical values occur more than once, cf. Figure 3. Suitable tilings could be found semi–atomatically by domain experts using the domain knowledge. In a case study, we have analyzed how often a player hit from tile `(i,j)` to `(k,l)` using a *spreadsheet*, cf. Figure 4. The coordinates `x=0/4` and `y=0/3` represent the outfield. The sum of a row `(i,j)` shows how often a player hit from that tile, and the sum of a column `(k,l)` shows how often it was hit to that tile.

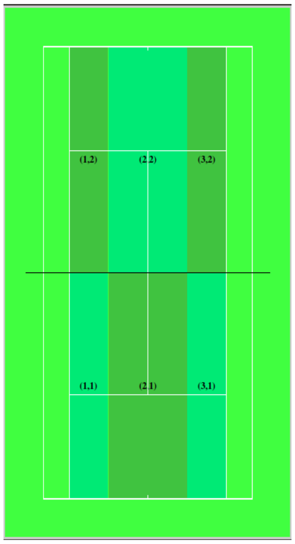


Fig. 3. Tennis Court with Suitable Tiling Computed Using Domain Knowledge

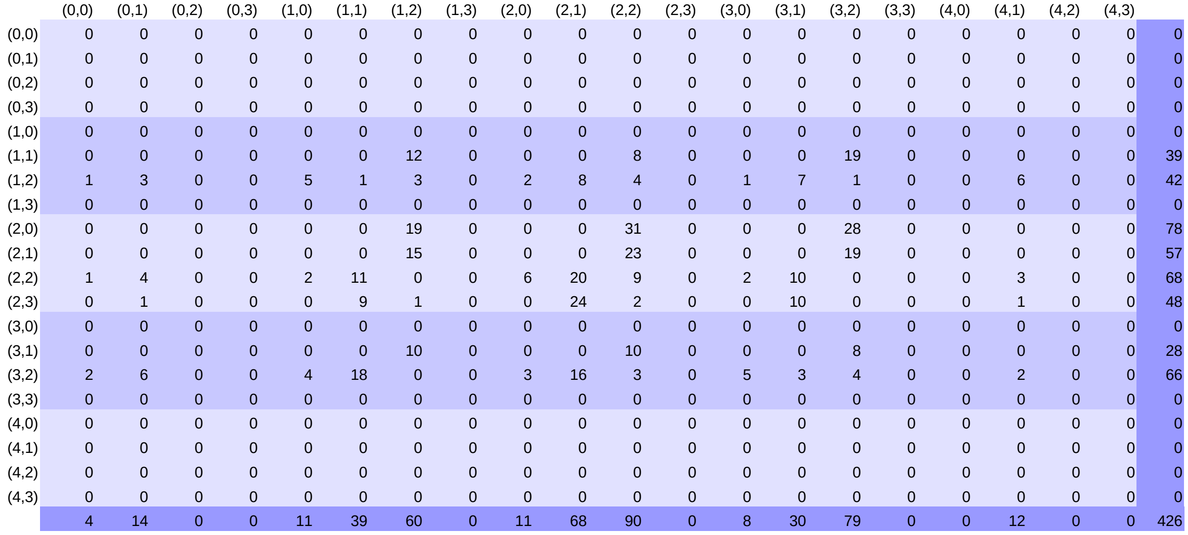
Declare has also been used for orchestrating different data mining methods of Weka and Vikamine. Based on the results of previous association rule minings, other – more or or less refined – tilings of the tennis court have been computed with FNQuery.

[3] http://www.vikamine.org

[4] https://rsubgroup.org)

4.1 OLAP Queries

After selecting and transforming the data using the mediator Declare to derive a suitable spreadsheet, simple OLAP queries could be asked. The spreadsheet of Figure 4 was derived from the tiling, that had been computed using domain knowledge from tennis experts.



	(0,0)	(0,1)	(0,2)	(0,3)	(1,0)	(1,1)	(1,2)	(1,3)	(2,0)	(2,1)	(2,2)	(2,3)	(3,0)	(3,1)	(3,2)	(3,3)	(4,0)	(4,1)	(4,2)	(4,3)	
(0,0)	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
(0,1)	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
(0,2)	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
(0,3)	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
(1,0)	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
(1,1)	0	0	0	0	0	0	12	0	0	0	8	0	0	0	19	0	0	0	0	0	39
(1,2)	1	3	0	0	5	1	3	0	2	8	4	0	1	7	1	0	0	6	0	0	42
(1,3)	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
(2,0)	0	0	0	0	0	0	19	0	0	0	31	0	0	0	28	0	0	0	0	0	78
(2,1)	0	0	0	0	0	0	15	0	0	0	23	0	0	0	19	0	0	0	0	0	57
(2,2)	1	4	0	0	2	11	0	0	6	20	9	0	2	10	0	0	0	3	0	0	68
(2,3)	0	1	0	0	0	9	1	0	0	24	2	0	0	10	0	0	0	1	0	0	48
(3,0)	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
(3,1)	0	0	0	0	0	0	10	0	0	0	10	0	0	0	8	0	0	0	0	0	28
(3,2)	2	6	0	0	4	18	0	0	3	16	3	0	5	3	4	0	0	2	0	0	66
(3,3)	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
(4,0)	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
(4,1)	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
(4,2)	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
(4,3)	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	4	14	0	0	11	39	60	0	11	68	90	0	8	30	79	0	0	12	0	0	426

Fig. 4. A Suitable Spreadsheet for OLAP Queries

To the best of the authors' knowledge, such a spreadsheet analysis for supporting (OLAP) queries in the context of declarative data mining, with adaptations via meta–learning is new. Here, we are currently investigating clustering methods on the speadsheets of different matches, e. g., for finding pairs of players with similar matches.

4.2 Mining Association Rules

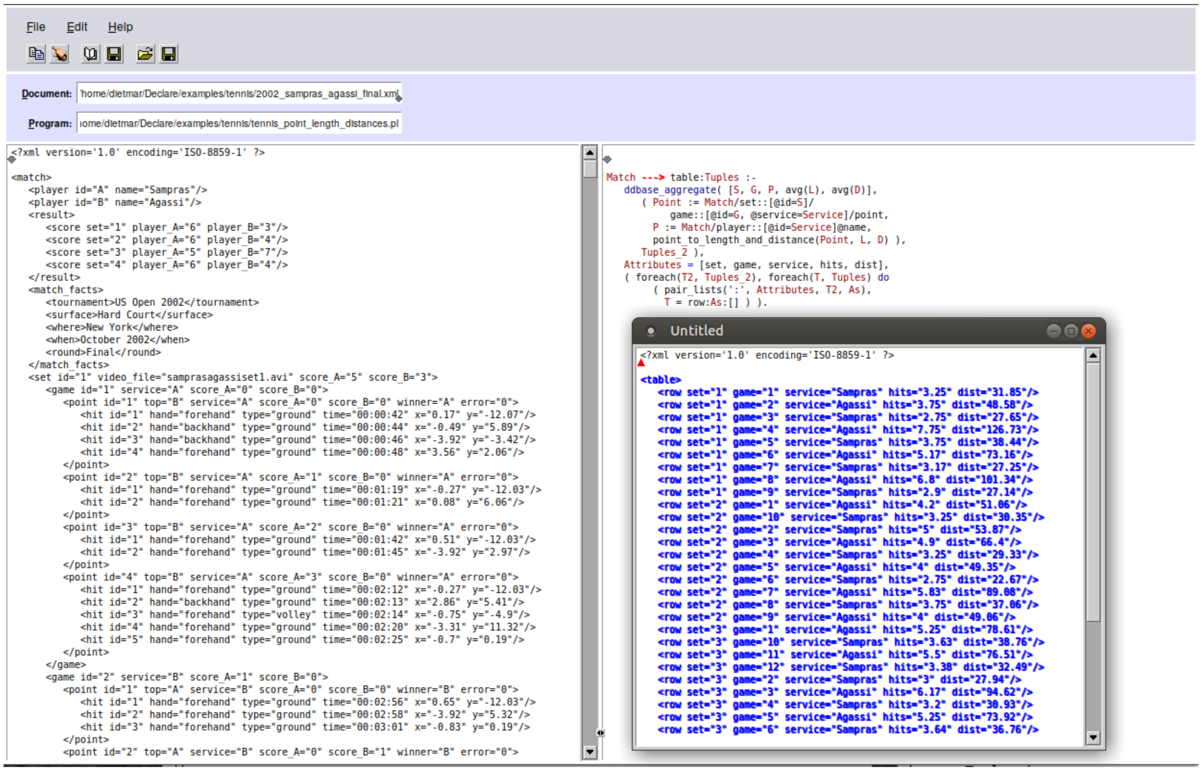
In [74], we have computed *association rules* with conjunctive conditions `x1=i, y1=j` and conjunctive consequences `x2=k, y2=l`. With Declare, it was possible to specify subsumption rules [68] between the association rules to select a small intuitive subset from the very large set of produced patterns (i.e. association rules), that can be presented as knowledge to the tennis expert.

```
Best rules found:
  1. x1=2, y1=0  61  ==>  x2=4, y2=2  44  conf:(0.72)
  2. x1=3, y1=0  64  ==>  x2=2, y2=2  38  conf:(0.59)
 11. x1=3, y1=0  64  ==>  x2=1, y2=2  25  conf:(0.39)

  3. x1=2, y1=3  76  ==>  x2=3, y2=1  38  conf:(0.50)
  5. x1=2, y1=3  76  ==>  x2=4, y2=1  35  conf:(0.46)
  4. x1=3, y1=3  75  ==>  x2=2, y2=1  36  conf:(0.48)
  6. x1=3, y1=3  75  ==>  x2=1, y2=1  34  conf:(0.45)

 14. x1=1, y1=1 106  ==>  x2=4, y2=2  37  conf:(0.35)
 28. x1=1, y1=1 106  ==>  x2=1, y2=2  20  conf:(0.19)
 18. x1=2, y1=1  65  ==>  x2=4, y2=2  20  conf:(0.31)
 ...
```

These interesting association rules have been selected with Declare from a very large number of rules returned by Weka using subsumption rules specified in Declare. The rules 1, 2, and 11 are talking about services from the bottom field, since `y1=0` . The rules 3, 5, 4, and 6 are talking about services hits from the top field, since `y1=3` . Finally, the rules 14, 28, and 18 are talking about ground hits from the bottom field, since `y1=1` .

4.3 Knowledge Representation and Reasoning

The *Tennis Tool* represents tennis matches in a *data warehouse* in XML format. There exists a toolkit FNQuery [67] with a *graphical user interface (GUI)* to process XML data, see Figure 5. After loading an XML document in the left window, we can ask queries with the finger icon or transform the data with the two pages icon. In Figure 5, a query is asked in the left window, and the result is given in the little picture below.


File
Edit
Help
Document: home/dietmar/Declare/examples/tennis/2002_sampras_agassi_final.xml
Program: ome/dietmar/Declare/examples/tennis/tennis_point_length_distances.pl
Untitled

```
<?xml version='1.0' encoding='ISO-8859-1' ?>

<match>
   <player id="A" name="Sampras"/>
   <player id="B" name="Agassi"/>
   <result>
      <score set="1" player_A="6" player_B="3"/>
      <score set="2" player_A="6" player_B="4"/>
      <score set="3" player_A="5" player_B="7"/>
      <score set="4" player_A="6" player_B="4"/>
   </result>
   <match_facts>
      <tournament>US Open 2002</tournament>
      <surface>Hard Court</surface>
      <where>New York</where>
      <when>October 2002</when>
      <round>Final</round>
   </match_facts>
   <set id="1" video_file="samprasagassiset1.avi" score_A="5" score_B="3">
      <game id="1" service="A" score_A="0" score_B="0">
         <point id="1" top="B" service="A" score_A="0" score_B="0" winner="A" error="0">
            <hit id="1" hand="forehand" type="ground" time="00:00:42" x="0.17" y="-12.07"/>
            <hit id="2" hand="backhand" type="ground" time="00:00:44" x="-0.49" y="5.89"/>
            <hit id="3" hand="backhand" type="ground" time="00:00:46" x="-3.92" y="-3.42"/>
            <hit id="4" hand="forehand" type="ground" time="00:00:48" x="3.56" y="2.06"/>
         </point>
         <point id="2" top="B" service="A" score_A="1" score_B="0" winner="A" error="0">
            <hit id="1" hand="forehand" type="ground" time="00:01:19" x="-0.27" y="-12.03"/>
            <hit id="2" hand="forehand" type="ground" time="00:01:21" x="0.08" y="6.06"/>
         </point>
         <point id="3" top="B" service="A" score_A="2" score_B="0" winner="A" error="0">
            <hit id="1" hand="forehand" type="ground" time="00:01:42" x="0.51" y="-12.03"/>
            <hit id="2" hand="forehand" type="ground" time="00:01:45" x="-3.92" y="2.97"/>
         </point>
         <point id="4" top="B" service="A" score_A="3" score_B="0" winner="A" error="0">
            <hit id="1" hand="forehand" type="ground" time="00:02:12" x="-0.27" y="-12.03"/>
            <hit id="2" hand="backhand" type="ground" time="00:02:13" x="2.86" y="5.41"/>
            <hit id="3" hand="forehand" type="volley" time="00:02:14" x="-0.75" y="-4.9"/>
            <hit id="4" hand="forehand" type="ground" time="00:02:20" x="-3.31" y="11.32"/>
            <hit id="5" hand="forehand" type="ground" time="00:02:25" x="-0.7" y="0.19"/>
         </point>
      </game>
      <game id="2" service="B" score_A="1" score_B="0">
         <point id="1" top="A" service="B" score_A="0" score_B="0" winner="B" error="0">
            <hit id="1" hand="forehand" type="ground" time="00:02:56" x="0.65" y="-12.03"/>
            <hit id="2" hand="forehand" type="ground" time="00:02:58" x="-3.92" y="5.32"/>
            <hit id="3" hand="forehand" type="ground" time="00:03:01" x="-0.83" y="0.19"/>
         </point>
         <point id="2" top="A" service="B" score_A="0" score_B="1" winner="B" error="0">
```

```
Match ---> table:Tuples :-
   ddbase_aggregate( [S, G, P, avg(L), avg(D)],
      ( Point := Match/set::[@id=S]/
           game::[@id=G, @service=Service]/point,
        P := Match/player::[@id=Service]@name,
        point_to_length_and_distance(Point, L, D) ),
      Tuples_2 ),
   Attributes = [set, game, service, hits, dist],
   ( foreach(T2, Tuples_2), foreach(T, Tuples) do
        ( pair_lists(':', Attributes, T2, As),
          T = row:As:[] ) ).
```

```
<?xml version='1.0' encoding='ISO-8859-1' ?>

<table>
   <row set="1" game="1" service="Sampras" hits="3.25" dist="31.85"/>
   <row set="1" game="2" service="Agassi" hits="3.75" dist="48.58"/>
   <row set="1" game="3" service="Sampras" hits="2.75" dist="27.65"/>
   <row set="1" game="4" service="Agassi" hits="7.75" dist="126.73"/>
   <row set="1" game="5" service="Sampras" hits="3.75" dist="38.44"/>
   <row set="1" game="6" service="Agassi" hits="5.17" dist="73.16"/>
   <row set="1" game="7" service="Sampras" hits="3.17" dist="27.25"/>
   <row set="1" game="8" service="Agassi" hits="6.8" dist="101.34"/>
   <row set="1" game="9" service="Sampras" hits="2.9" dist="27.14"/>
   <row set="2" game="1" service="Agassi" hits="4.2" dist="51.06"/>
   <row set="2" game="10" service="Sampras" hits="3.25" dist="30.35"/>
   <row set="2" game="2" service="Sampras" hits="5" dist="53.87"/>
   <row set="2" game="3" service="Agassi" hits="4.9" dist="66.4"/>
   <row set="2" game="4" service="Sampras" hits="3.25" dist="29.33"/>
   <row set="2" game="5" service="Agassi" hits="4" dist="49.35"/>
   <row set="2" game="6" service="Sampras" hits="2.75" dist="22.67"/>
   <row set="2" game="7" service="Agassi" hits="5.83" dist="89.08"/>
   <row set="2" game="8" service="Sampras" hits="3.75" dist="37.06"/>
   <row set="2" game="9" service="Agassi" hits="4" dist="49.06"/>
   <row set="3" game="1" service="Agassi" hits="5.25" dist="78.61"/>
   <row set="3" game="10" service="Sampras" hits="3.63" dist="38.76"/>
   <row set="3" game="11" service="Agassi" hits="5.5" dist="76.51"/>
   <row set="3" game="12" service="Sampras" hits="3.38" dist="32.49"/>
   <row set="3" game="2" service="Sampras" hits="3" dist="27.94"/>
   <row set="3" game="3" service="Agassi" hits="6.17" dist="94.62"/>
   <row set="3" game="4" service="Sampras" hits="3.2" dist="30.93"/>
   <row set="3" game="5" service="Agassi" hits="5.25" dist="73.92"/>
   <row set="3" game="6" service="Sampras" hits="3.64" dist="36.76"/>
```

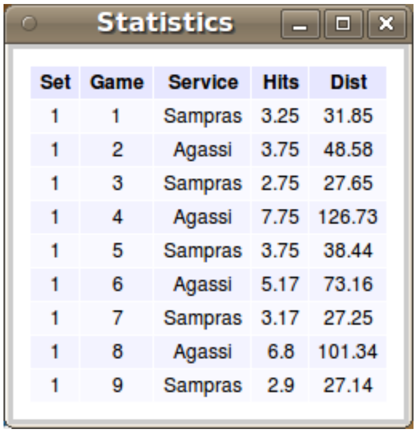
Fig. 5. GUI of FNQuery

Powerful queries with path expressions – similar to XPath – including user–defined aggregation are possible in FNQuery. In the following, we will show two examples which have not been published before.

Example: Length of a Rally. The average length (number of hits and length of the way of the ball) is aggregated by sets (S) and games (G). The serving player (P) is also returned.

```
average_point_distances(Match, Set, Tuples) :-
   ddbase_aggregate( [S, G, P, avg(L), avg(D)],
      ( Point := Match/set::[@id=S]/
           game::[@id=G, @service=Service]/point,
        P := Match/player::[@id=Service]@name,
        point_to_length_and_distance(Point, L, D) ),
      Tuples ).
```

For the first set, the respective statistics with the returned tuples are shown in a table in XPCE, see Figure 6. When Sampras serves, then the rallies are much shorter. As a serve–and–volley player, he runs to the net early to finish the point:



Set	Game	Service	Hits	Dist
1	1	Sampras	3.25	31.85
1	2	Agassi	3.75	48.58
1	3	Sampras	2.75	27.65
1	4	Agassi	7.75	126.73
1	5	Sampras	3.75	38.44
1	6	Agassi	5.17	73.16
1	7	Sampras	3.17	27.25
1	8	Agassi	6.8	101.34
1	9	Sampras	2.9	27.14

Fig. 6. Statistics for the Games of the First Set: Average Number of Hits and Average Point Distances

Example: Cross–Cross–Longline Rally. We are especially interested in rallies where two cross balls are played before a longline (`ccl`).

```
tennis_point_classify(ccl, File, Point) :-
   Point := doc(File)/descendant::point,
   point_to_successive_values(x, Point, [X1, X2, X3, X4]),
   Limit = 1,
   ( X1 >=  Limit, X2 =< -Limit, X3 >=  Limit, X4 >=  Limit
   ; X1 =< -Limit, X2 >=  Limit, X3 =< -Limit, X4 =< -Limit ).

point_to_successive_values(Attribute, Point, Values) :-
   Hits_1 := Point/content::'*',
   n_successive_elements(4, Hits_1, Hits_2),
   ( foreach(H, Hits_2), foreach(V, Values) do
        A := H@Attribute ).
```

The longline starts the decision phase: often, the other player cannot reach the ball, since the longline is unexpected and the way to the other side of the court is long; otherwise, if he reaches the ball, then he has a good chance to win the point himself.

The following Declare query shows the cross–cross–longline rallies and waits for a user input between displaying rallies. Every single rally is visualized in the graphical user interface, see Figure 7, and the corresponding video sequence can be played.

```
?- Type = ccl, File = '2002_sampras_agassi_final.xml',
   forall( tennis_point_classify(Type, File, Point),
      ( tennis_point_xml_display(Point), wait ) ).
```

4.4 Subgroup Discovery with Vikamine

We have applied subgroup discovery using the Vikamine algorithmic kernel component via its XML interface. For this, we applied the tennis dataset which was accordingly preprocessed using Declare, on the level of *points*, and *points and hits* – for a more detailed analysis if not only the attributes that contribute to a point are included for analysis, but also individual hits by the respective players can be analyzed. Using this, we have defined several subgroup discovery tasks, specifying different target concepts.

In our application case, i. e., for tennis data analysis, we can make use of the rich modeling capabilities of the Vikamine kernel component, including extended quality functions which make use of the concepts of subgroup discovery as well as exceptional model mining. We can, for example, focus on patterns that a specific player is the winner of a point. For this simple example, we just need to specify the respective target variable. We implement this by selecting a binary target $t_A = (winner = A)$ or $t_B = (winner = B)$ for the respective players A (Sampras) and B (Agassi), respectively.
Then, we can detect subgroups as follows:

Target t_A (Sampras wins):

- *Sampras* wins in 85% of all cases, if he serves, there are *no errors*, and *Agassi* has a score of 0.
- Another interesting pattern is given by *Sampras* = *top* and *service* = *Sampras* with a share of winning the point of 72%.

Target t_B (Agassi wins):

- If Sampras makes an *error*, then Agassi wins with 100%.
- Also, for example, if *set* = *2* and *error* = *0* and *service* = *Agassi*, then Agassi wins with 79%.

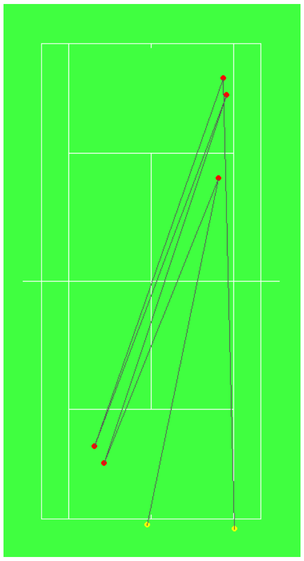


Fig. 7. A Cross–Cross–Longline Rally

Furthermore, such analysis can then also be implemented for numeric target variables; in addition, they can be extended towards more complex modeling structures such as a graph – something where subgroup discovery in particular benefits from its flexibly definable quality function, in contrast to e. g., association rule mining.

5 Conclusions

In general, data mining systems are applied to obtain a set of *novel*, *potentially useful*, and ultimately *interesting* patterns from (large) data sets [27]. While the resulting patterns are typically interpretable, e. g., in pattern mining, the large result sets of potentially interesting patterns that the user needs to assess, require further *exploration* and *interpretation* techniques. Here, specifically declarative approaches, meta–learning and orchestration provide suitable options for the implementation of such approaches, essentially supporting *computational sensemaking* in order to “make sense” in the context of complex information and knowledge processes [4].

In the context of this paper, we focused on a specific application case – given by tennis data mining. Here, in a human-centered knowledge discovery approach, trainers can analyse and potentially improve the tactical behaviour of their students using *declarative* data mining concepts, as outlined in this paper. So far, the orchestration of the methods from knowledge discovery in tennis data warehouses has been done semi–automatically, in particular making use of and based on the transformation capabilities of the mediator toolkit Declare. We are currently working on an extended orchestration prototype for the data mining and knowledge discovery workflow, in order to enable further declarative approaches supported by an integrated architecture.

Furthermore, in [74], we had performed an initial case study for tennis data, where association rule mining was applied for a single tennis match and different tilings of a tennis court. In the future, it will be interesting to built a large *data warehouse* containing tennis matches of many players over several seasons and to orchestrate various methods for data mining. Then we can compare the strategies of different players versus different opponents. In general, we would like to incorporate the usage of *declarative domain knowledge* in the orchestration process.

In the future, we also aim analyze the technique of the different *types of hits* (forehand, backhand, volley) of the players using more sophisticated video analysis.

Acknowledgements

We would like to thank the anonymous reviewers for their useful comments which have improved the paper, and also the domain experts for their valuable input concerning expert knowledge on the tennis application case.

References

1. Rakesh Agrawal, Ramakrishnan Srikant, et al. Fast algorithms for mining association rules. In *Proc. 20th International Conf. on Very Large Data Bases (VLDB 1994)*, volume 1215, pages 487–499. Citeseer, 1994.
2. Martin Atzmueller. *Knowledge-Intensive Subgroup Mining – Techniques for Automatic and Interactive Discovery*, volume 307 of *Dissertations in Artificial Intelligence-Infix (Diski)*. IOS Press, March 2007.
3. Martin Atzmueller. Subgroup Discovery. *WIREs Data Mining and Knowledge Discovery*, 5(1):35–49, 2015.
4. Martin Atzmueller. Declarative Aspects in Explicative Data Mining for Computational Sensemaking. In Dietmar Seipel, Michael Hanus, and Salvador Abreu, editors, *Proc. International Conference on Declarative Programming*, pages 97–114, Heidelberg, Germany, 2018. Springer.
5. Martin Atzmueller, Stephan Doerfel, and Folke Mitzlaff. Description–Oriented Community Detection using Exhaustive Subgroup Discovery. *Information Sciences*, 329:965–984, 2016.
6. Martin Atzmueller and Florian Lemmerich. VIKAMINE – Open–Source Subgroup Discovery, Pattern Mining, and Analytics. In *Proc. ECML/PKDD 2012: European Conference on Machine Learning and Principles and Practice of Knowledge Discovery in Databases*, Heidelberg, Germany, 2012. Springer.
7. Martin Atzmueller, Florian Lemmerich, Jochen Reutelshoefer, and Frank Puppe. Wiki–enabled semantic data mining–task design, evaluation and refinement. In *Proc. International Workshop on Design, Evaluation and Refinement of Intelligent Systems (DERIS 2009), vol. CEUR–WS*, volume 545, 2009.
8. Martin Atzmueller and Frank Puppe. Semi–Automatic Visual Subgroup Mining using VIKAMINE. *Journal of Universal Computer Science*, 11(11):1752–1765, 2005.
9. Martin Atzmueller and Frank Puppe. SD–Map – A Fast Algorithm for Exhaustive Subgroup Discovery. In *Proc. 10th European Conf. on Principles and Practice of Knowledge Discovery in Databases (PKDD 2006)*, pages 6–17, Heidelberg, Germany, 2006. Springer.
10. Martin Atzmueller and Frank Puppe. Causal Subgroup Analysis for Detecting Confounding. In *Proc. 18th International Conference on Applications of Declarative Programming and Knowledge Management (INAP 2007)*, Wuerzburg, Germany, 2007.
11. Martin Atzmueller and Dietmar Seipel. Declarative Specification of Ontological Domain Knowledge for Descriptive Data Mining. In *Proc. International Conference on Applications of Declarative Programming and Knowledge Management (INAP)*, pages 158–170, 2007.
12. Martin Atzmüller. Experience management with task–configurations and task–patterns for descriptive data mining. In Joachim Baumeister and Dietmar Seipel, editors, *Proceedings of the 3rd Workshop on Knowledge Engineering and Software Engineering (KESE 2007) at the 30th German Conference on Artificial Intelligence (KI 2007), Osnabrück, Germany, September 10, 2007*, volume 282 of *CEUR Workshop Proceedings*. CEUR–WS.org, 2007.
13. Yves Bastide, Nicolas Pasquier, Rafik Taouil, Gerd Stumme, and Lotfi Lakhal. Mining Minimal Non-Redundant Association Rules Using Frequent Closed Itemsets. In J. Lloyd and V. Dahl and U. Furbach and M. Kerber and K.–K. Lau and C. Palamidessi and L. and M. Pereira and Y. Sagiv and P. and J. Stuckey, editor, *Computational Logic - CL 2000. Proc. CL '00*, pages 972–986, Heidelberg, Germany, 2000. Springer.
14. Hendrik Blockeel. Declarative data analysis. *International Journal of Data Science and Analytics*, 6(3):217–223, 2018.
15. Gary Bradski and Adrian Kaehler. *Learning OpenCV: Computer vision with the OpenCV library*. " O'Reilly", 2008.
16. Ivan Bratko. *Prolog Programming for Artificial Intelligence*. Addison–Wesley Longman, 4th edition, 2011.
17. Pavel Brazdil, Christophe Giraud Carrier, Carlos Soares, and Ricardo Vilalta. *Metalearning: Applications to Data Mining*. Springer Science & Business Media, 2008.
18. Leo Breiman. Bagging predictors. *Machine learning*, 24(2):123–140, 1996.
19. Kailash Budhathoki, Mario Boley, and Jilles Vreeken. Discovering Reliable Causal Rules. In *Proc. SIAM International Conference on Data Mining (SDM)*, pages 1–9. SIAM, 2021.
20. Toon Calders and Bart Goethals. Mining All Non–Derivable Frequent Itemsets. In Tapio Elomaa, Heikki Mannila, and Hannu Toivonen, editors, *Proc. 6th European Conference on Principles of Data Mining and Knowledge Discovery*, volume 2431 of *Lecture Notes in Computer Science*, pages 74–85. Springer, 2002.
21. Centeio Jorge, Carolina and Atzmueller, Martin and Heravi, Behzad M and Gibson, Jenny L and Rossetti, Rosaldo JF and Rebelo de Sá, Cláudio. "Want to Come Play With Me?" Outlier Subgroup Discovery on Spatio–Temporal Interactions. *Expert Systems*, 2021.
22. William Clocksin and Christopher S. Mellish. *Programming in* PROLOG. Springer Science & Business Media, 2003.
23. Gregory F. Cooper. A Simple Constraint-Based Algorithm for Efficiently Mining Observational Databases for Causal Relationships. *Data Mining and Knowledge Discovery*, 1(2):203–224, 1997.
24. W. Duivesteijn, A. Knobbe, A. Feelders, and M. van Leeuwen. Subgroup Discovery Meets Bayesian Networks–An Exceptional Model Mining Approach. In *Proc. ICDM*. IEEE, 2010.
25. Wouter Duivesteijn, Ad Feelders, and Arno J. Knobbe. Different Slopes for Different Folks: Mining for Exceptional Regression Models with Cook's Distance. In *Proc. KDD*, pages 868–876, 2012.
26. Wouter Duivesteijn, Ad J. Feelders, and Arno Knobbe. Exceptional Model Mining. *Data Mining and Knowledge Discovery*, 30(1):47–98, 2016.

27. Usama M. Fayyad, Gregory Piatetsky-Shapiro, and Padraic Smyth. From Data Mining to Knowledge Discovery: An Overview. In *Advances in Knowledge Discovery and Data Mining*, pages 1–34. AAAI Press, 1996.
28. Fayyad, Usama and Piatetsky–Shapiro, Gregory and Smyth, Padhraic. From data mining to knowledge discovery in databases. *AI magazine*, 17(3):37–37, 1996.
29. Frawley, William J. and Piatetsky–Shapiro, Gregory and Matheus, Christopher J. Knowledge discovery in databases: An overview. *AI magazine*, 13(3):57–57, 1992.
30. Yoav Freund, Robert Schapire, and Naoki Abe. A short introduction to boosting. *Journal Japanese Society For Artificial Intelligence*, 14(771–780):1612, 1999.
31. Yoav Freund and Robert E Schapire. A decision–theoretic generalization of on–line learning and an application to boosting. *Journal of computer and system sciences*, 55(1):119–139, 1997.
32. Gemma C Garriga, Petra Kralj, and Nada Lavrač. Closed Sets for Labeled Data. *Journal of Machine Learning Research*, 9(4), 2008.
33. Andrew Hendrickson, Jason Wang, and Martin Atzmueller. Identifying Exceptional Descriptions of People using Topic Modeling and Subgroup Discovery. In *Proc. 24th International Symposium on Methodologies for Intelligent Systems (ISMIS)*, LNCS, Heidelberg, Germany, 2018. Springer.
34. Jochen Hipp, Ulrich Güntzer, and Gholamreza Nakhaeizadeh. Algorithms for association rule mining – a general survey and comparison. *ACM SIGKDD explorations newsletter*, 2(1):58–64, 2000.
35. Szymon Jaroszewicz and Dan A. Simovici. Interestingness of Frequent Itemsets using Bayesian Networks as Background Knowledge. In *KDD '04: Proc. 10th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 178–186, New York, NY, USA, 2004. ACM Press.
36. Jean–Francois Boulicaut. *Encyclopedia of Data Warehousing and Mining*, chapter Condensed Representations for Data Mining, pages 37–79. Idea Group, 2006.
37. Mark Kibanov, Martin Atzmueller, Jens Illig, Christoph Scholz, Alain Barrat, Ciro Cattuto, and Gerd Stumme. Is Web Content a Good Proxy for Real-Life Interaction? A Case Study Considering Online and Offline Interactions of Computer Scientists. In *Proc. ASONAM*, Boston, MA, USA, 2015. IEEE.
38. Willi Klösgen. Explora: A multipattern and multistrategy discovery assistant. In *Advances in Knowledge Discovery and Data Mining*, pages 249–271. Springer, 1996.
39. Willi Klösgen and Michael May. Census Data Mining - An Application. In D. Malerba and P. Brito, editors, *Proc. Workshop Mining Official Data, 6th European Conference, PKDD 2002*, Helsinki, 2002. Helsinki Univ. Printing House.
40. Willi Klösgen and Michael May. Spatial Subgroup Mining Integrated in an Object-Relational Spatial Database. In T. Elomaa, H. Mannila, and H. Toivonen, editors, *Proc. Principles of Data Mining and Knowledge Discovery. 6th European Conference, PKDD 2002*, volume 2431 of *LNCS*, pages 275–286, Heidelberg, Germany, 2002. Springer.
41. Janez Kranjc, Vid Podpečan, and Nada Lavrač. Clowdflows: A Cloud Based Scientific Workflow Platform. In *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*, pages 816–819. Springer, 2012.
42. Nada Lavrač and Anže Vavpetič. Relational and semantic data mining. In *International Conference on Logic Programming and Nonmonotonic Reasoning*, pages 20–31. Springer, 2015.
43. Christiane Lemke, Marcin Budka, and Bogdan Gabrys. Metalearning: a survey of trends and technologies. *Artificial intelligence review*, 44(1):117–130, 2015.
44. Florian Lemmerich, Martin Atzmueller, and Frank Puppe. Fast Exhaustive Subgroup Discovery with Numerical Target Concepts. *Data Mining and Knowledge Discovery*, 30(3):711–762, 2016.
45. Bing Liu and Wynne Hsu. Post–Analysis of Learned Rules. In *Proc. 13th National Conference on Artificial Intelligence (AAAI-96)*, pages 828–834, Menlo Park, CA, 1996. AAAI Press.
46. Luca Longo, Randy Goebel, Freddy Lecue, Peter Kieseberg, and Andreas Holzinger. Explainable Artificial Intelligence: Concepts, Applications, Research Challenges and Visions. In Andreas Holzinger, Peter Kieseberg, A. Min Tjoa, and Edgar Weippl, editors, *Machine Learning and Knowledge Extraction*, volume 12279 of *LNCS*, pages 1–16. Springer, Cham, 2020.
47. Corentin Lonjarret, Céline Robardet, Marc Plantevit, Roch Auburtin, and Martin Atzmueller. Why Should I Trust This Item? Explaining the Recommendations of any Model. In *Proc. IEEE International Conference on Data Science and Advanced Analytics (DSAA)*, pages 526–535. IEEE, 2020.
48. Bing Liu Wynne Hsu Yiming Ma, Bing Liu, and Yiming Hsu. Integrating classification and association rule mining. In *Proceedings of the fourth International Conference on Knowledge Discovery and Data Mining (KDD 1998)*, pages 80–86, 1998.
49. James MacQueen. Some Methods for Classification and Analysis of Multivariate Observations. In *Proceedings of the Fifth Berkeley Symposium on Mathematical Statistics and Probability*, volume 1, pages 281–297. Oakland, CA, USA, 1967.
50. Heikki Mannila. Theoretical Frameworks for Data Mining. *SIGKDD Explor.*, 1(2):30–32, 2000.
51. Gonzalo Mariscal, Oscar Marban, and Covadonga Fernandez. A survey of data mining and knowledge discovery process models and methodologies. *The Knowledge Engineering Review*, 25(2):137, 2010.
52. Martin Atzmueller and Frank Puppe and Hans–Peter Buscher. Exploiting Background Knowledge for Knowledge–Intensive Subgroup Discovery. In *Proc. 19th International Joint Conference on Artificial Intelligence (IJCAI-05)*, pages 647–652, Edinburgh, Scotland, 2005.

53. Martin Atzmueller and Frank Puppe and Hans–Peter Buscher. A Semi-Automatic Approach for Confounding–Aware Subgroup Discovery. *International Journal on Artificial Intelligence Tools (IJAIT)*, 18(1):1 – 18, 2009.
54. Martin Atzmueller and Thomas Roth–Berghofer. The Mining and Analysis Continuum of Explaining Uncovered. In *Proc. 30th SGAI International Conference on Artificial Intelligence*, 2010.
55. Osman Mian, Alexander Marx, and Jilles Vreeken. Discovering Fully Oriented Causal Networks. In *Proc. AAAI Conference on Artificial Intelligence (AAAI)*. AAAI, 2021.
56. Taneli Mielikäinen. Finding all Occurring Sets of Interest. In Jean–Francois Boulicaut and Saso Dzeroski, editor, *Proc. 2nd International Workshop on Knowledge Discovery in Inductive Databases.*, pages 97–106, 2003.
57. Dennis Mollenhauer and Martin Atzmueller. Sequential exceptional pattern discovery using pattern-growth: An extensible framework for interpretable machine learning on sequential data. In Martin Atzmüller, Tomás Kliegr, and Ute Schmid, editors, *Proceedings of the First International Workshop on Explainable and Interpretable Machine Learning (XI-ML 2020) co-located with the 43rd German Conference on Artificial Intelligence (KI 2020), Bamberg, Germany, September 21, 2020 (Virtual Workshop)*, volume 2796 of *CEUR Workshop Proceedings*. CEUR-WS.org, 2020.
58. Nicolas Pasquier, Yves Bastide, Rafik Taouil, and Lotfi Lakhal. Discovering Frequent Closed Itemsets for Association Rules. In Catriel Beeri and Peter Buneman, editors, *Proc. 7th International Conference on Database Theory (ICDT 99)*, volume 1540 of *Lecture Notes in Computer Science*, pages 398–416. Springer, 1999.
59. Jian Pei, Jiawei Han, and Runying Mao. CLOSET: An Efficient Algorithm for Mining Frequent Closed Itemsets. In *ACM SIGMOD Workshop on Research Issues in Data Mining and Knowledge Discovery*, pages 21–30, 2000.
60. Matic Perovšek, Anže Vavpetič, Janez Kranjc, Bojan Cestnik, and Nada Lavrač. Wordification: Propositionalization by unfolding relational data into bags of words. *Expert Systems with Applications*, 42(17–18):6442–6456, 2015.
61. Piatetsky–Shapiro, Gregory. Knowledge discovery in databases: 10 years after. *ACM SIGKDD Explorations Newsletter*, 1(2):59–61, 2000.
62. Vid Podpečan, Monika Zemenova, and Nada Lavrač. Orange4WS Environment for Service-Oriented Data Mining. *The Computer Journal*, 55(1):82–98, 2012.
63. Puppe, Frank and Atzmueller, Martin and Buscher, Georg and Huettig, Matthias and Lührs, Hardi and Buscher, Hans–Peter. Application and evaluation of a medical knowledge system in sonography (sonoconsult). In *Proceedings of the 2008 conference on ECAI 2008: 18th European Conference on Artificial Intelligence*, pages 683–687, 2008.
64. Jr. Roberto J. Bayardo. Efficiently Mining Long Patterns from Databases. In *SIGMOD '98: Proc. of the 1998 ACM SIGMOD International Conference on Management of data*, pages 85–93, New York, NY, USA, 1998. ACM Press.
65. Lior Rokach. A survey of clustering algorithms. In *Data Mining and Knowledge Discovery Handbook*, pages 269–298. Springer, 2009.
66. Dietmar Seipel. Declare – A Declarative Toolkit for Knowledge–Based Systems and Logic Programming. http://www1.pub.informatik.uni-wuerzburg.de/databases/research.html.
67. Dietmar Seipel. Processing XML–Documents in PROLOG. In *Workshop on Logic Programming (WLP 2002)*, 2002.
68. Dietmar Seipel. *Advanced Databases*, Lecture Notes of a Course at the University of Würzburg, since 2015.
69. Dietmar Seipel, Stefan Köhler, Philipp Neubeck, and Martin Atzmueller. Mining Complex Event Patterns in Computer Networks. In *Post Proceedings of the 1st Workshop on New Frontiers in Mining Complex Patterns (NFMCP 2012*, LNAI. Springer, Heidelberg, Germany, 2013.
70. Eric Sternberg and Martin Atzmueller. Knowledge-Based Mining of Exceptional Patterns in Logistics Data: Approaches and Experiences in an Industry 4.0 Context. In *Proc. 24th International Symposium on Methodologies for Intelligent Systems (ISMIS)*, LNCS, Heidelberg, Germany, 2018. Springer.
71. Thornton, Chris and Hutter, Frank and Hoos, Holger H. and Leyton–Brown, Kevin. Auto–WEKA: Combined selection and hyperparameter optimization of classification algorithms. In *Proceedings of the 19th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 847–855, 2013.
72. Evangelos Triantaphyllou. *Data Mining and Knowledge Discovery via Logic–Based Methods: Theory, Algorithms, and Applications*, volume 43. Springer Science & Business Media, 2010.
73. Simon Vollert, Martin Atzmueller, and Andreas Theissler. Interpretable Machine Learning: A Brief Survey From the Predictive Maintenance Perspective. In *Proc. IEEE International Conference on Emerging Technologies and Factory Automation (ETFA 2021)*. IEEE, 2021.
74. Daniel Weidner, Martin Atzmueller, and Dietmar Seipel. Finding Maximal Non–Redundant Association Rules in Tennis Data. In Petra Hofstedt, Salvador Abreu, Ulrich John, Herbert Kuchen, and Dietmar Seipel, editors, *Declarative Programming and Knowledge Management – Conference on Declarative Programming, DECLARE 2019, Unifying INAP, WLP, and WFLP, Cottbus, Germany, September 9–12, 2019, Revised Selected Papers*, volume 12057 of *Lecture Notes in Computer Science*, pages 59–78. Springer, 2019.
75. Jan Wielemaker. SWI–Prolog Reference Manual 7.6. Technical report, 2017.
76. Stefan Wrobel. An Algorithm for Multi–Relational Discovery of Subgroups. In *Proc. PKDD 1997*, pages 78–87, Heidelberg, Germany, 1997. Springer.
77. Dongkuan Xu and Yingjie Tian. A comprehensive survey of clustering algorithms. *Annals of Data Science*, 2(2):165–193, 2015.