Preference Miner: A Database Tool for Mining User Preferences

Markus Endres

University of Augsburg
Universitätsstr. 6a
86159 Augsburg, Germany
Email: endres@informatik.uni-augsburg.de

***Abstract*—Advanced personalized e-applications require comprehensive preference knowledge about their users' likes and dislikes in order to provide individual product recommendations, personal customer advice, and custom-tailored product offers. Modeling preferences as strict partial orders with *"A is better than B"* semantics has proven to be very suitable in various e-applications. In this demo, we present the *Preference Miner*, a database tool for detection of strict partial order preferences hidden in the users' log data. With preference mining personalized applications can gain valuable knowledge about their customers' preferences, which can be applied for personalized product recommendations, individual customer service, or one-to-one marketing.**

Keywords–Preference; Personalization; Data Mining; Database.

I. Motivation

In recent years, several techniques have been developed to build user adaptive web sites and personalized web applications [1]. For example, e-commerce applications use link personalization to recommend items based on the customer's buying behavior or some categorization of customers based on ratings and opinions. Research on preference handling systems makes use of quite a variety of different tools, cp. [2]. Some preference elicitation approaches have been proposed in a different manner, e.g., [3], which proposes algorithms for automatic contextual preference elicitation. However, current techniques of automatic personalization lack preference models with limited expressiveness. State-of-the-art approaches either use scores to describe preferences or just distinguish between liked and disliked values. Thus, complex *"I like A more than B"*-relationships, as well as preferences for numeric attributes cannot be expressed in a natural way. Furthermore, these approaches are not able to handle dependencies among preferences, e.g., two preferences are equally important or one preference is preferred to another.

In this demo paper, we present the *Preference Miner*, a database tool for mining user preferences. Preference Mining is a technology for the detection of preferences in the user's previous shopping or browsing behavior recorded in his log data, e.g., click data, browsing data, or explicit feedback. Important applications for preference mining are Internet shops, financial e-services or personal recommender systems where individual customer care plays a significant role [4][5].

The rest of the paper is organized as follows: In Section II we introduce the preference background. Section III describes our demo architecture and Section IV contains our conclusion.

II. Preference Background

A database preference $P = (A, <_P)$ is a strict partial order, where $A = \{A_1, \ldots, A_d\}$ denotes a set of attributes with corresponding domains $\mathrm{dom}(A_i)$. The domain of A is defined as Cartesian product of $\mathrm{dom}(A_i)$, $<_P \subseteq \mathrm{dom}(A) \times \mathrm{dom}(A)$ and $x <_P y$ is interpreted as "y is better than x".

A set of intuitive preference constructors for base and complex preferences is defined in [6]. These definitions of preference constructors have been proven to be appropriate to describe user wishes. On categorical data there are $\mathtt{POS}(A, \text{POS-set})$, $\mathtt{NEG}(A, \text{NEG-set})$, $\mathtt{POS/POS}(A, \text{POS1-set, POS2-set})$, and $\mathtt{POS/NEG}(A, \text{POS-set, NEG-set})$. The POS-set $\subseteq \mathrm{dom}(A)$ of a POS preference defines a set of values that are better than all other values of $\mathrm{dom}(A)$. Analogously, the NEG set describes disliked values. In the POS/POS preference the POS1-set defines the most preferred values, whereas the POS2-set defines the second-preferred values when nothing better is available. The POS/NEG preference defines preferred and non-preferred values. In E-graph of an $\mathtt{EXPLICIT}(A, \text{E-graph})$ preference, a user can specify any better-than relationships. Numerical preference constructors are $\mathtt{AROUND}(A, z)$, $\mathtt{BETWEEN}(A, [low, up])$, $\mathtt{LOWEST}(A)$, and $\mathtt{HIGHEST}(A)$. In AROUND the desired value is z, but if this is not feasible values with the nearest distance from z are best alternatives. BETWEEN prefers values within a $[low, up]$ interval and LOWEST resp. HIGHEST prefer lower and higher values. A Pareto preference $P := P_1 \otimes \ldots \otimes P_m$ treats the underlying preferences as equally important whereas a Prioritization $P := P_1 \& \ldots \& P_m$ treats P_1 more important than P_2, and so on. A more detailed description of the preference model is given in [7].

Example 1. *Assume a dataset called "Notebooks", which contains information about notebooks bought by customers. The data contains different attributes like the size of the hard disk (HDD), the make (Acer, Lenovo, …), or the amount of RAM. The wish for a notebook having a HDD around 1TB and made by Acer (equally important preferences) can be expressed as*

$$\mathtt{AROUND}(\text{HDD, 1TB}) \otimes \mathtt{POS}(\text{Make}, \{\text{Acer}\})$$

We developed a miner for preferences which detects all kinds of base preferences as well as complex preferences like Pareto and Prioritization within log data. For the detection of preferences, our algorithms apply well-established data mining techniques like clustering and density estimation [8]. The preference mining algorithms together with proofs of correctness can be found in [9].

III. Preference Miner Implementation

Figure 1 represents the overall architecture of our Preference Miner implemented in Java 1.8[1]. As input the user or an application program tells the Preference Miner which log-relation to analyze and on which attributes preferences should be detected. Thereby, the log-relations come from a database or any text based file (.txt, .csv, .tsv, etc.).

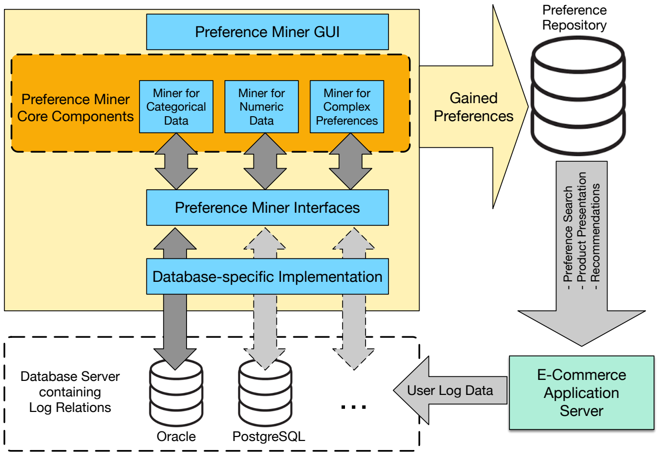
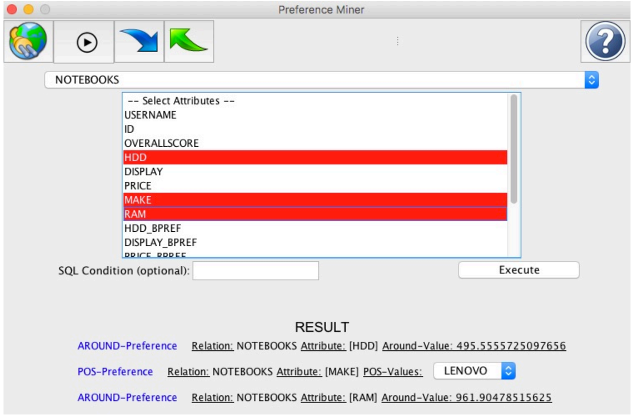

Preference Repository
Preference Miner GUI
Preference Miner Core Components
Miner for Categorical Data
Miner for Numeric Data
Miner for Complex Preferences
Gained Preferences
Preference Miner Interfaces
Database-specific Implementation
- Preference Search
- Product Presentation
- Recommendations
Database Server containing Log Relations
Oracle
PostgreSQL
...
User Log Data
E-Commerce Application Server

Figure 1. Preference Miner architecture.

The Preference Miner implements preference mining algorithms ("Preference Miner Core Components"), where all data intensive operations, such as clustering or density estimation are executed on the database layer for high performance. The core components contain algorithms for mining preferences on categorical and numerical data as well as a miner for complex preferences. To be independent from a specific database system, all database independent operations are implemented as part of the Preference Miner, whereas database specific operations are only specified ("Preference Miner Interfaces").

The graphical user interface (Figure 2) of the Preference Miner allows the comfortable invocation of the preference mining algorithms on the specified attributes and afterwards presents the results. Here, the Preference Miner detected three preferences on the log-relation "Notebooks" mentioned in Example 1. All detected preferences are managed intelligently in an appropriate preference database, the *Preference Repository*, cp. Figure 1. We developed such a Preference Repository [10], which is a storage structure for preferences. A set of access functions allows easy inserts, updates, deletions and selects on the repository. With it the application server can perform dynamic query personalization [1] for preference-based product-search, individualized product presentation, or personalized recommendations.

Since our implementation executes all data-intensive operations on the database layer we can achieve a very good performance behavior. Our tests on a commercial database server has shown that mining numerical preferences is the fastest task. Mining Pareto preferences or Prioritization needs less than a second in the average for 50,000 tuples on a standard computer. Detailed performance measurements for each algorithm can be found in [9]. The good efficiency of our preference mining algorithms allows their usage for *online preference mining*: while interacting with a customer the e-application can check online his preferences and therefore can react flexibly to his wishes during the sales process.



NOTEBOOKS

-- Select Attributes --
USERNAME
ID
OVERALLSCORE
HDD
DISPLAY
PRICE
MAKE
RAM
HDD_BPREF
DISPLAY_BPREF
PRICE_BPREF

SQL Condition (optional):

RESULT

AROUND-Preference Relation: NOTEBOOKS Attribute: [HDD] Around-Value: 495.5555725097656

POS-Preference Relation: NOTEBOOKS Attribute: [MAKE] POS-Values: LENOVO

AROUND-Preference Relation: NOTEBOOKS Attribute: [RAM] Around-Value: 961.90478515625

Figure 2. Preference Miner GUI.

IV. Conclusion

The goal of this work is to provide a tool for mining user preferences form log data. Such preference knowledge can be very useful for personalized applications. Sales advice can be adapted to the customer's individual preferences, e.g., if he likes low prices or a special make. Furthermore, preferences gained with Preference Mining are useful for personalized product recommendations and for the composition of individual product bundles.

Acknowledgements

This work has been partially funded by the German Federal Ministry for Economic Affairs and Energy according to a decision by the German Bundestag, grant no. ZF4034402LF5. We want to thank Stefan Holland for support and providing us with the basic source code of the preference miner algorithms.

References

[1] K. Stefanidis, G. Koutrika, and E. Pitoura, "A Survey on Representation, Composition and Application of Preferences in Database Systems," ACM TODS, vol. 36, no. 3, 2011, pp. 19:1–19:45.

[2] S. Kaci, Working with Preferences: Less Is More, 1st ed. Springer Publishing Company, Incorporated, 2011.

[3] S. de Amo, M. S. Diallo, C. T. Diop, A. Giacometti, D. Li, and A. Soulet, "Contextual Preference Mining for User Profile Construction," Inf. Syst., vol. 49, no. C, 2015, pp. 182–199.

[4] Y. Ioannidis and G. Koutrika, "Personalized Systems: Models and Methods from an IR and DB Perspective," in Proceedings of VLDB '05. ACM, 2005, p. 1365.

[5] B. Satzger, M. Endres, and W. Kießling, "A Preference-Based Recommender System," in Proceedings of EC-Web '06, ser. LNCS, vol. 4082, 2006, pp. 31 – 40.

[6] W. Kießling, "Foundations of Preferences in Database Systems," in Proceedings of VLDB '02, Hong Kong, China, 2002, pp. 311–322.

[7] W. Kießling, M. Endres, and F. Wenzel, "The Preference SQL System - An Overview," Bulletin of the Technical Commitee on Data Engineering, IEEE Computer Society, vol. 34, no. 2, 2011, pp. 11–18.

[8] M. J. A. Berry and G. Linoff, Data Mining Techniques. Wiley, New York, 1997.

[9] S. Holland, M. Ester, and W. Kießling, "Preference Mining: A Novel Approach on Mining User Preferences for Personalized Applications," in Proceedings of PKDD '03, ser. LNCS, vol. 2838. Springer, 2003, conf, pp. 204–216.

[10] S. Holland and W. Kießling, "Situated Preferences and Preference Repositories for Personalized Database Applications," in Proceedings of ER '04, ser. LNCS, vol. 3288. Springer, 2004, conf, pp. 511–523.

[1] The tool is available at https://github.com/endresma/PreferenceMiner.git

ISBN: 978-1-61208-558-6