Incremental prediction of configurator input values based on association rules – A case study

Dietmar Jannach and Lukas Kalabis
TU Dortmund
Dortmund, Germany
dietmar.jannach@tu-dortmund.de, lukas.kalabis@tu-dortmund.de

Abstract

In many configurator applications, the user is required to specify a multitude of configuration options interactively. One way of making the configuration process more convenient for the user is to pre-fill the input or selection fields of the user interface with appropriate defaults. Possible strategies to determine default values for example include the selection based on domain knowledge or the usage of statistics.

In this paper we analyze whether or not the dynamic selection of defaults based on automatically determined association rules can help to predict the most probable next input value in an interactive and incremental configuration process. We base our analysis on the data obtained with a real-world configurator application. The value of choosing defaults more intelligently is determined by measuring the number of correctly predicted inputs in the configuration process and by comparing this number to a default strategy based on simple value frequencies.

1 Introduction

In many industrial sectors, the products on the market can be customized to a customer's individual needs in a variety of ways. Accordingly, the interactive preference elicitation and product configuration process can become time-consuming and cumbersome, because the user of the configurator application is often required to enter input or make selections for several dozens of parameters.

Providing suitable default inputs or default selections for the individual options represents one common way to make the configuration process more convenient for the user. How the system chooses the pre-set default value can be based on different strategies. One simple strategy for input fields with a predefined set of options could be simply selecting the first value of the list. Alternatively, one could use the value that was chosen most frequently in the past (also by other users). In some systems – such as the ADVISOR SUITE sales advisory framework [Felfernig *et al.*, 2006] – the selection of the default values is based on domain knowledge. Beside the static definition of defaults, this framework also supports the definition of *rules* that determine the proposed default value for a field based on the inputs already made in the current session.

Our own anecdotal experiences with using predefined default values provided by the domain expert in the domain of interactive advisory systems however showed that such static rules can have different limitations. First, domain experts often define what should be the default selection for an input field merely based on gut feeling and intuition. Second, in some domains, the most appropriate default value changes over time, for example due to technological advances or a changing marketing strategy. The rules determining the defaults in the configurator applications are however not always updated accordingly.

In this work we propose to use association rules [Agrawal and Srikant, 1994], which can be automatically learned from past configuration sessions, to dynamically choose the most appropriate default values. We evaluate the value of applying this self-adapting default selection strategy by counting the number of correct and wrong predictions when replaying past interactive configuration sessions and comparing our strategy to a statistics-based baseline strategy. The analysis of the approach is based on a real-world configuration data base.

2 Mining input value patterns

Figure 1 shows a schematic but typical fragment of a user interface for a configurator, in which the user of the system – in our case a sales representative – incrementally enters the preferences of the customer. In our real-world database from the roofing industry, on average more than two dozens of such parameters have to be entered during the configuration process. The database comprises more than 9,000 past roofing configurations.

The goal of our work is to try to detect patterns co-occurring input values in these past configurations and exploit these patterns to predict the next input values in the interactive configuration process.

Association rules have been traditionally used in data mining scenarios and in particular for shopping basket analysis. With the help of algorithms such as Apriori [Agrawal and Srikant, 1994], rules of the following form can be extracted from past buying transactions:

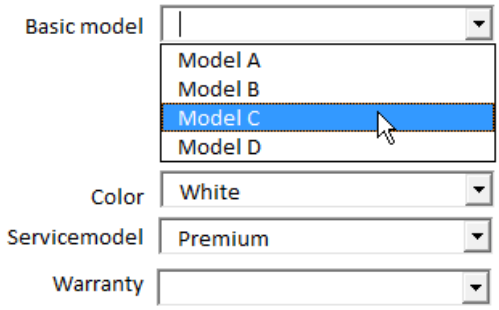

Figure 1: Schematic user interface fragment.

$\{milk, bread\} \Rightarrow \{butter\}$
[support=60%, confidence=80%]

The rule states that whenever customers purchase milk and bread, they also buy butter. Support and confidence are measures for the strength or quality of a rule. The support metric describes in how many of all existing transactions the *itemset* {milk,bread,butter} appeared. The confidence measure corresponds to the percentage of transactions in which milk and bread appeared and where butter (the right hand side of the rule) was also part of the transaction.

In our problem setting, a completed configuration corresponds to a flat list of assignments of values to the more than two dozen input variables, that is, our configuration problem is relatively simple when compared, for example, to classical component port configuration models [Mittal and Frayman, 1989]. Applying association rule mining algorithms such as Apriori (as used in our work) or FP-Growth [Han *et al.*, 2000] is therefore straightforward and the goal of the mining process is to detect patterns such as

$\{Basic\text{-}Model = A,\ Color = White\} \Rightarrow \{Warranty = 3yr.\}$

As an overall result of the mining process, a set of such association rules can be determined. Usually, a minimum support value has to be set in order to only take significant patterns into account.

3 Default selection schemes

In our evaluation, we compared three schemes for determining the default value: (A) take a random value (the first in the list of options); (B) the most frequent value from the past transactions is taken as a default; (C) the selection is based on association rules.

For scheme C, a "sliding window" strategy was applied. Note that we assume in our application that we have a strict order in which the inputs are entered (from top to the bottom). In scheme C, the defaults therefore depend on the previous inputs. If we, for example look for a value for the warranty field, we look for rules that have the previous inputs (such as the color) in the left hand side of the rule. The window size describes how many of the last n inputs we take into account. Taking all previous inputs into account might be impractical as the set of detected association rules might not contain rules that have 20 or more inputs on the left hand side. When multiple rules are in principle applicable (but suggest different right hand side values), we choose the rule with the highest confidence value. In combination with the sliding window strategy, we also apply a relaxation strategy in case we cannot find a matching rule. If, for example, no rule with the left hand side {Basic-Model = A, Color = White} can be found in a window of size 2, we calculate all subsets of the left hand side and take the rule with the highest confidence value.

Overall, in contrast to default selection B where all values can be set at the beginning, in scheme C, the defaults are determined dynamically based on the previous inputs.

4 Evaluation

Metric. As an evaluation metric, we count the number of clicks that are required to configure the customized product variant. Note that we have one full default configuration for scheme A and one for scheme B. To measure the number of required clicks, we iterate through all 9,000 past transactions and check for each transactions how many of the input field values are different from this default configuration. The configuration effort thus corresponds to the average number of values that have to be changed.

For scheme C, we "replay" the configuration process for each of the past configuration sessions and predict the input field value one after the other based on the association rules. In case the prediction of the next input was correct, we move the sliding window forward and predict the next input. In case the prediction was wrong, we increase the counter of required clicks, correct the input value to the one given in the current past transaction and proceed with predicting the next input field.

Results. Due to the fact that the choice of the basic roof model, which has to be done as a first step, considerably influences the available choice for the rest of the configuration process, we have learned a set of association rules for each of these basic models. In addition, we have experimented with different window sizes as well as minimum support values. Figure 2 exemplarily shows the results for one of the basic models.

On average, 27,5 input values were set for a configuration of this model type. Using the simplistic default selection scheme A (pick the first value in the list), on average a bit more than 15 values have to be set (changed) manually by the user[1]. However, if we apply scheme B (pick the most frequent value), a very strong improvement can be observed and only about 6 of the 27 values were not properly predicted, which strongly indicates that there are some configurations options, which are very popular and that there is a long tail of configurations which is very seldom sold.

Regarding the dynamic prediction scheme C, we experimented with different settings and in particular varied the parameters *window size* and *minimum support (MS)*, see Figure 2. With respect to the window size, we can observe that a larger window size, which in turn means that we learn longer rules, helps to improve the predictive accuracy of our rule-based approach. The best results in our experiments were achieved with window sizes 5 and 6. Further tests showed

[1]The median number of input values is 5.

Scheme A
15,83

Scheme B
5,92

Avg. nb of inputs
27,50

Scheme C				
Window size	MS = 10%	MS = 5%	MS = 4%	MS = 3%
1	5,72	5,24	5,45	5,43
2	5,62	5,35	5,24	5,20
3	5,50	5,38	5,09	4,98
4	5,64	5,28	5,07	4,94
5	5,39	4,93	4,74	4,74
6	5,43	4,88	4,73	4,73

Figure 2: Results for one representative basic model.

that beyond this window size no further improvements could be observed. The time required for the (offline) rule mining process however significantly goes up when the maximum length of the rules to be learned is increased.

With respect to the MS value, lower threshold values lead to better results and the best predictions were achieved with MS values at 3% and 4%. Lower MS values mean that also rules for "rare combinations" are learned and included in the rule base. Again, further decreasing the MS value leads to marginal improvements at the price of a much larger rule base. Overall, we can see that the accuracy consistently increases when the MS value is lowered. For the window sizes, in contrast, we could observe that further increasing this parameter does not always lead to better results.

For the particular basic model for which we show the results in Figure 2, we can see an overall improvement from 5,92 to 4,73 required clicks. At first glance, this might not look too impressive. Note however, that the good results that were achieved with the comparably simple scheme B are due to the very unbalanced distribution of the available input values in the past configurations. Figure 3 shows a typical example for a "yes/no" question. For three out of four user input fields, such a lopsided distribution could be observed.

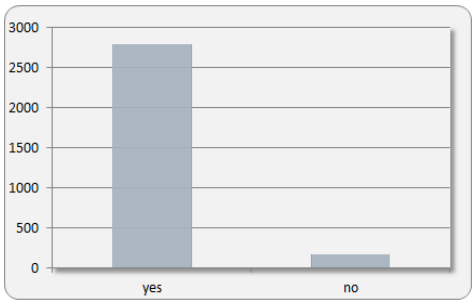

Figure 3: Value distribution for field "standard height (y/n)".

Across all basic models, an average reduction of required clicks of 11.88% was observed. A first analysis showed that the achieved improvements do not so much depend on the number of available training samples but rather on the diversity of the actually configurations.

Running times. The time needed for the generation of input value predictions based on the previously learned association rules can be considered to be suitable for an interactive configuration scenario. In all our experiments, the computation of input values for *all* fields took at most 10 seconds (that is, below 0.5 seconds for each field) even for the largest window sizes considered in the experiments. The times needed for the offline model-building phase strongly depend on the minimum support size. While at the 10% level model-building takes a few seconds, the calculations at the 3% level can take a few hours on a standard desktop PC.

5 Previous works

In [Ardissono *et al.*, 2002], an advanced approach to personalizing the preference elicitation process in a configurator application was proposed. In particular, their system exploits stereotypical user modeling techniques to assess the user's preferences and properties throughout the interaction process. In contrast to our work, in their approach the goal is also to find *personalized* defaults which depend on the individual behavior of the user. Another approach to personalize the interaction process with the goal to reduce the complexity of the overall process for the end user based on user profiles and personalized recommendations was proposed in [Stegmann *et al.*, 2003] and [Stegmann *et al.*, 2006].

Currently, personalization of the default proposal process is beyond the scope of our work but could be relatively easy implemented by learning user-specific rule models for situations, in which enough data is available for personalization.

Recently, in [Tiihonen and Felfernig, 2010] and [Felfernig *et al.*, 2010] a knowledge-based approach to personalizing the user interaction process for configurator applications was presented. Beside the automatic generation of "repair proposals" for situations in which the customer requirements are inconsistent (as also discussed in [Felfernig *et al.*, 2001]), the authors propose different (probabilistic and statistics-based) techniques to determine suitable feature values as proposed earlier already in [Cöster *et al.*, 2002]. The goal of their work is similar to ours, although different techniques are employed and for example metrics for measuring the "distance" between configurations as well as feature weights are exploited. Unfortunately, no empirical evaluation of their proposal was yet done. However, [Felfernig *et al.*, 2010] also consider the question that the proposed feature values have to be consistent with the configuration knowledge base and the current, partial configuration. In our current work, this aspect was not considered yet. One strategy to deal with this aspect could be to systematically try to apply different association rules (ordered by their confidence) and check whether the configuration is still consistent after rule application.

In [Geneste and Ruet, 2001], finally, a Case-Based Reasoning (CBR) approach to reduce the complexity of the in-

teraction process was proposed. The main idea is not to start configurations from scratch, but to reuse and adapt past configurations. The main tasks are therefore to find past similar cases based on a similarity metric and a search algorithm, determine possible adaptations (the adaptation domain) and then guide the user through the adaptation process using constraint satisfaction techniques. Beside the goal of making the interaction process more efficient, one similarity between our work and the work of [Geneste and Ruet, 2001] is that we rely on past configurations to steer the interaction process. However, in our work we assume an incremental process in which configurations are developed and refined step by step. The consideration of the consistency checks before the default proposal process as done in [Geneste and Ruet, 2001] is currently not supported in our approach.

6 Summary and outlook

In this work, we have analyzed how association rules mined from past configurations can be exploited to predict input values for interactive configuration processes and can thus help to make the usage of such systems more comfortable in case the user has to configure a multitude of options. The evaluation on a real-world data set showed that a measurable reduction in required interactions can be achieved when compared to a simpler statistics-based approach or even in cases when we have a market which is dominated by a few very popular configurations.

Among other aspects, our future work includes the analysis of the algorithm on other real-world datasets and the application of other techniques from data mining and machine learning for input value prediction in the configuration domain. Beside that, our goal is to evaluate prediction strategies in which the elements contained in the sliding window not only depend on the chronological order of the inputs but on some relevance weight, assuming that individual inputs are more predictive than others in the configuration process. In addition, future work could also consider the question if there are characteristics of the configuration problem (such as the domain sizes of the variables) which can help us to automatically determine appropriate values for the minimum support threshold.

References

[Agrawal and Srikant, 1994] Rakesh Agrawal and Ramakrishnan Srikant. Fast algorithms for mining association rules in large databases. In *Proc. of 20th Intl. Conference on Very Large Data Bases*, pages 487–499, Santiago de Chile, Chile, 1994.

[Ardissono *et al.*, 2002] L. Ardissono, A. Felfernig, G. Friedrich, A. Goy, D. Jannach, M. Meyer, G. Petrone, R. Schäfer, W. Schütz, and M. Zanker. Personalising on-line configuration of products and services. In *Proc. of the 15th European Conference on Artificial Intelligence*, pages 225–229, Lyon, France, 2002.

[Cöster *et al.*, 2002] Rickard Cöster, Andreas Gustavsson, Tomas Olsson, sa Rudstrm, and Asa Rudstrm. Enhancing web-based configuration with recommendations and cluster-based help. In *In Proceedings of the AH'2002 Workshop on Recommendation and Personalization in eCommerce*, pages 30–39, Malaga, Spain, 2002.

[Felfernig *et al.*, 2001] Alexander Felfernig, Gerhard Friedrich, Dietmar Jannach, and Markus Zanker. Intelligent support for interactive configuration of mass-customized products. In *Proceedings of the 14th International Conference on Industrial and Engineering Applications of Artificial Intelligence and Expert Systems*, pages 746–756, 2001.

[Felfernig *et al.*, 2006] Alexander Felfernig, Gerhard Friedrich, Dietmar Jannach, and Markus Zanker. An integrated environment for the development of knowledge-based recommender applications. *International Journal of Electronic Commerce*, 11:11–34, 2006.

[Felfernig *et al.*, 2010] Alexander Felfernig, Monika Mandl, Juha Tiihonen, Monika Schubert, and Gerhard Leitner. Personalized user interfaces for product configuration. In *Proceedings of the 15th International Conference on Intelligent User Interfaces*, pages 317–320, Hong Kong, China, 2010.

[Geneste and Ruet, 2001] L. Geneste and M. Ruet. Experience based configuration. In *Proceedings of the Configuration Workshop at IJCAI'01*, Seattle, USA, 2001.

[Han *et al.*, 2000] J. Han, J. Pei, and Y. Yin. Mining frequent patterns without candidate generation. In *Proc. 2000 ACM SIGMOD Intl. Conference on Management of Data*, pages 1–12, Dallas, Texas, 2000.

[Mittal and Frayman, 1989] S. Mittal and F. Frayman. Towards a generic model of configuration tasks. In *Proc. 11th Intl. Joint Conference on Artificial Intelligence*, pages 1395–1401, Detroit, Michigan, 1989.

[Stegmann *et al.*, 2003] Rosmary Stegmann, Michael Koch, Martin Lacher, Thomas Leckner, and Volker Renneberg. Generating personalized recommendations in a model-based product configurator system. In *Workshop on Configuration at IJCAI'03*, Acapulco, Mexico, 2003.

[Stegmann *et al.*, 2006] Rosmary Stegmann, Thomas Leckner, Michael Koch, and Johann Schlichter. Customer support for the web-based configuration of individualised products. *International Journal of Mass Customization*, 1(2):195–217, 2006.

[Tiihonen and Felfernig, 2010] Juha Tiihonen and Alexander Felfernig. Towards recommending configurable offerings. *International Journal of Mass Customization*, 3(4):389–406, 2010.