Post-processing association rules using networks and transductive learning.

http://www.producao.usp.br/handle/BDPI/48799

Downloaded from: Biblioteca Digital da Produção Intelectual - BDPI, Universidade de São Paulo
Abstract—Association is widely used to find relations among items in a given database. However, finding the interesting patterns is a challenging task due to the large number of rules that are generated. Traditionally, this task is done by post-processing approaches that explore and direct the user to the interesting rules of the domain. Some of these approaches use the user’s knowledge to guide the exploration according to what is defined (thought) as interesting by the user. However, this definition is done before the process starts. Therefore, the user must know what may be and what may not be interesting to him/her. This work proposes a general association rule post-processing approach that extracts the user’s knowledge during the post-processing phase. That way, the user does not need to have a prior knowledge in the database. For that, the proposed approach models the association rules in a network, uses its measures to suggest rules to be classified by the user and, then, propagates these classifications to the entire network using transductive learning algorithms. Therefore, this approach treats the post-processing problem as a classification task. Experiments were carried out to demonstrate that the proposed approach reduces the number of rules to be explored by the user and directs him/her to the potentially interesting rules of the domain.

Keywords—Association Rules; Pruning; Post-Processing; Label Propagation; Networks

I. INTRODUCTION

Association is widely used in data mining due its simplicity and comprehensibility. This task aims to extract the correlations among items in a given database [1]. The problem is that a large number of patterns are obtained - even a small dataset can generate a large number of rules, depending on the configuration parameters. Therefore, it is infeasible to manually explore all these rules. Generally, the number of interesting rules is very small compared to the total generated and, in most of the times, the user must search many rules to look for the rules that are considered interesting to him/her.

Aiming to reduce the number of rules to be explored, some approaches were proposed to post-process these rules. Some approaches use objective measures to create a ranking of the most interesting patterns ([16]); others use clustering to partition the domain ([5], [3]); others the user’s knowledge, previously obtained, to select the most interesting patterns ([7], [8]); etc. (more details in Section II). Regarding the approaches that do not use the user’s knowledge to explore the domain, the exploration is done using some automatic mechanisms that analyse the ruleset to direct the user to the interesting rules of the domain. On the other hand, regarding the approaches that use the user’s knowledge, his/her knowledge is informed before the exploration starts, which means that the user needs to know, beforehand, what he/she thought to be of interesting. However, in some cases, the user does not know what may be interesting and what he/she wants to discover and/or explore. Besides, the knowledge needs to be informed in some determined formalism and, so, the user needs to know how to define his/her knowledge in some pre-defined format.

Based on the exposed, this paper presents a general post-processing approach that iteratively extracts the user’s knowledge during the exploration, i.e., that interacts with the user during the post-processing process. This is done by selecting a set of rules to be classified by the user at each iteration, that way the knowledge obtained in the previous iteration is kept. The proposed approach works in three phases: first, it uses a network to model the rules. In the second phase, it selects the rules to be classified by the user by creating a ranking using network measures; the user classifies these rules as “Interesting” or “Not-Interesting”. In the last phase, transductive learning algorithms are applied to propagate the user’s classifications among the rules in the network and a stopping criterion is tested: if the criterion is met the process ends and the rules classified as “Interesting” are shown to the user; else the process starts again from phase two, using the knowledge previously obtained. Therefore, the proposed approach treats the post-processing problem as a classification task.

The first phase (modeling phase) uses network’s configurations to model the rules. To do so, it is necessary to define a similarity measure to calculate a weight among the connections of the rules. The approach takes as input a set of association rules and a similarity measure and generates a network of association rules. Networks have the capability to represent the knowledge and the relations among the rules without losing information.

In the second phase (interactive phase) the rules modeled in the network are processed, using a network measure to create a ranking among them. This ranking is used to suggest some rules to be classified by the user in two classes: “Interesting” or “Not-Interesting”. Creating the ranking using network measures favors the rules that have more connections and, conse-
sequently, the rules that have a greater impact in the network. The ranking is created so a small number of rules are classified by the user, reducing his/her effort. Also, the user’s classification is important because he/she has the knowledge that cannot be obtained by automatic post-processing approaches.

The last phase (classification phase) uses transductive learning algorithms to propagate the classifications performed by the user to the entire rule set. Transductive learning algorithms classify a dataset using a small number of classified data based on the existing similarity among the instances [12]. The transductive algorithms used by the proposed approach are the network based ones, applied over the network modeled on the first phase.

In summary, the main contribution of this paper is a new association rule post-processing approach that iteratively includes the user’s knowledge during the post-processing phase, excluding the need of the user to have prior knowledge in the domain. Besides, this paper also presents a different way to post-process the rules, considering the problem as a classification task. This is proposed because the user must define, or classify, some rules as “Interesting” or “Not-Interesting”; therefore, it is possible to classify the entire rule set based on these classified rules. The paper presents a general approach to post-process the association rules and presents some experiments that were carried out with six datasets. The results show that the approach is capable of reducing the number of rules explored by the user and also show that the approach is capable of finding the rules that are considered interesting.

This paper is organized as follows. Section II describes related researches and basic concepts. Section III presents the proposed approach and its motivation. Section IV describes some experiments that were carried out to analyze the approach. Section V discusses the results obtained in the experiments. Finally, conclusion is given in Section VI.

II. BACKGROUND

A network can be characterized by a set of elements and the relations among them. Formally, a network can be represented by \( N = (V, E, W) \) where \( V \) is the set of vertices (elements), \( E \) the set of links between the vertices and \( W \) the weight of the links [9]. That representation allows a large variety of exploration. One way to explore the network, searching for information, is by applying transductive learning algorithms. Transductive learning algorithms classify a dataset based on few classified examples (the training set is composed of labeled and unlabeled instances). In this case, the algorithm classifies the unlabeled instances without creating a classification model. In this paper, the network based transductive learning (label propagation) algorithms are used. For that, the classes are propagated neighbor to neighbor, reducing a loss function [12].

The Gaussian Fields and Harmonic Function (GFHF) classifier, proposed by [11] and used in this paper, is a network based transductive classifier that reduces a quadratic loss function. The label propagation is made aiming to reduce Equation 1,

\[
F = \infty \sum_{i \in L} (f_i - y_i)^2 + \frac{1}{2} \sum_{i,j \in V} w_{ij} (f_i - f_j)^2 \tag{1}
\]

where \( L \) is the set of labeled (classified) elements, \( f \) is the function that returns the class of an element according to the classifier, \( y \) is the true class of an element and \( w \) is the matrix containing all the similarities among the elements in the network. The first sum is responsible to ensure that the elements that have a true class keep their class correct. This is done by multiplying the sum of the differences among the classes (the true and the predicted) by infinite. That way, if an element is erroneously classified the equation returns infinite and the propagation will be made again (the algorithm iterates until it converges). The second sum is responsible to analyze the classification of the elements that do not have a true class, i.e., the elements that the correct class is not known. This is done by comparing the elements that already have a class (a true class or a class given by the classifier) with the elements that do not have a class (classes are adjusted iteratively). The elements that already have a class will propagate their class to the elements that do not have. The class to be assigned to a non-classified element will be the same class of a classified element that presents a high similarity with it. That way, the loss function will be minimized. After each iteration, the classifier will check all the assigned classes applying Equation 1; if there is no change, the process finishes.

The Learning with Local and Global Consistency (LLGC) classifier, also used in this paper, proposed by [10], optimizes a different loss function that decreases the strength among the connections with high degree, shown in Equation 2,

\[
F(t + 1) = (\alpha \ S \ F(t)) + ((1 - \alpha) \ Y), \quad S = \frac{w}{\sqrt{D \sqrt{D}}} \tag{2}
\]

where \( D \) in \( S \) is the vector containing the output degree of each element, \( F(t) \) is the value obtained in the previous iteration, \( Y \) is the vector containing the true classes assigned to the elements and \( \alpha \) is the weight given to each side of the equation (classifier parameter). In this equation, \( S \) is used instead of \( w \) so the propagation strength of the objects with higher degree can be decreased. This is done so the objects with low degrees can be taken in account during the propagation. This algorithm, which also works iteratively, propagates the classes based on the weight among the elements in the network. The propagation is made considering the classified elements and their neighbors.

Association rules post-processing approaches aim to reduce the number of rules that are going to be explored by the final user, directing him/her to the rules that are considered interesting. The approaches can be divided in six different groups: Filtering by Constraints, Evaluation Measures, Summarization, Grouping, Pruning, and Hybrids. In the Filtering by Constraints approaches the rules are explored through constraints informed by the user before the post-processing starts. These constraints are informed through a defined formalism, such as templates and/or schemas (examples in [7] and [8]). Evaluation Measures, used in this paper as a baseline to analyze the proposed approach, rank the rules, according to their relevance,
considering measures with different objectives. These measures are usually classified as objective or subjective. While objective measures take into account the data structure to measure the rule’s relevance, subjective measures take into account the user’s needs/interests (examples in [6]). Summarization approaches aim to find more general rules, reducing the number of rules that needs to be explored. The process can be done in a wide range of ways (examples in [2] and [4]), some of them including the use of ontologies. Grouping approaches come from the clustering area. These approaches apply clustering algorithms over association rules aiming to structure the domain, putting similar rules in the same group (examples in [5] and [3]). In the Pruning approaches the rules that are not considered interesting are removed so the interesting ones can remain to be explored (examples in [7], [8] and [3]). Finally, Hybrid approaches combine two or more approaches previously explained. In this case, more than one approach is sequentially applied so the processing can be done using more than one bias (examples in [5], [7], [8] and [3]).

III. TRANSDUCTIVE POST-PROCESSING APPROACH

Some post-processing approaches extracts the user’s knowledge before exploring the rules, forcing the user to have an idea of what he/she wants to explore. The transductive post-processing approach, proposed in this paper, comes up with a different way to extract and use the user’s knowledge. The approach selects some rules to be classified by the user, directing the user’s efforts to the rules considered to have most impact in the network, according to some network measure. These classified rules are used by the networks’ based transductive learning algorithms to propagate the classes among all the rules that are not classified in the network. The transductive approach is used due to the low number of classified elements (in this cases, rules) needed to classify the entire dataset (in this case, ruleset). The general idea of the transductive post-processing approach is structured in Algorithm 1. The modeling phase encompasses lines 1 and 2; the interactive phase lines 4, 5, 6 and 7; the classification phase lines 8, 9 and 10.

In line 1, the first phase starts by modeling the association rules set \( R \) in a network. To do so, it is necessary that a similarity measure \( SM \) and a network type \( NT \) be defined. The \( SM \) is applied to calculate the existing similarities among the rules to use these values as the weights of the connections between the rules. Moreover, the \( NT \) defines the way the network is going to be built (\( Knn \), for instance). In line 2, the association rules, modeled as a network, are prepared to be processed in the next phase. At this point, all the rules in \( SI \) are considered unlabeled, i.e., without a known class.

After modeling, the second phase starts. In line 4, the approach explores the network of association rules to select some of them, stored in \( SI \), to be classified by the user. To do so, it is necessary to define a network measure \( NM \) and the number of rules \( Nr \) to be classified. First, the network is analyzed using \( NM \) and a ranking is created. This ranking is used to select the \( Nr \) rules to be classified by the user. However, for a broader exploration of the rules, the approach selects the top \( Nr \) and the last \( Nr \) rules. That way, the exploration is made in two directions: one starting with the rules with the best scores and one with the rules with the worst scores. In line 5, the user classifies the selected rules as “Interesting” or “Not-Interesting”, according to his/her experience, to direct the exploration of the ruleset. These classifications will form the labeled set to be used by the transductive algorithm in the last phase. Finally, the network is prepared to be classified in lines 6 and 7. The line 6 is responsible to build a labeled set, which contains all the rules already labeled by the user, and line 7 is responsible to build the set of rules to be classified by the algorithm, i.e., the set of rules that do not have a known class.

The last phase of the approach starts at line 8. The unlabeled association rules are classified using a network based transductive learning algorithm \( C \), considering as the training set the rules already classified by the user. The classifier will propagate the classes among the rules in the network, classifying them as “Interesting” or “Not-Interesting”. It is important to distinguish the classifications made by the user from the ones made by the classifier: The user’s classifications cannot be changed, i.e., over the iterations these classifications will be maintained and used in the training set over all the iterations; on the other hand, the classifier’s classifications can change over the iterations, based on the user’s classifications through the iterations. After the classification process, the rules considered “Interesting” by the classifier are stored in \( SI \) in line 9. This means that from the second iteration onwards, only the rules classified “Interesting” by the classifier are considered as candidates to be selected in line 4. That way, the user is directed to the rules considered interesting according to his/her knowledge.

In the end, the stopping criterion is checked in line 10. At this point, it will be decided if the process is over or if it will be another iteration. If the stopping criterion is met, then the rules classified as “Interesting” by the user and by the classifier, in the last iteration, are shown to the user as a result set (line 11). On the other hand, if the stopping criterion is not met, then the process goes back to the second phase, in line 4.

Algorithm 1: The transductive post-processing approach.

\[
\begin{align*}
\text{Input} & : \text{A set of association rules } R; \text{ number of rules } Nr; \text{ network measure } NM; \text{ similarity measure } SM; \text{ network type } NT; \text{ classifier } C. \\
\text{Output} & : \text{The set of interesting association rules in } R. \\
1 & \text{NAR} = \text{ModelRulesAsNetwork}(R, SM, NT); \\
2 & \text{SI} = \text{NAR}; \\
3 & \text{repeat} \\
4 & \text{SubSetRules} = \text{SelectSubSetRules}(\text{SI}, NM, Nr); \\
5 & \text{SubSetLblRules} = \text{UserLabelsRule}(\text{SubSetRules}); \\
6 & \text{LabeledRules} = \text{LabeledRules} + \text{SubSetLblRules}; \\
7 & \text{UnlabeledRules} = \text{NAR} - \text{LabeledRules}; \\
8 & \text{CR} = \text{ClassifyRules}(C, \text{LabeledRules}, \text{UnlabeledRules}); \\
9 & \text{SI} = \text{SelectInterestingRules}(\text{CR}); \\
10 & \text{until StoppingCriterionMet}(); \\
11 & \text{return SI + SelectInterestingRules}(% \text{LabeledRules}); \\
\end{align*}
\]

Note that the proposed approach is generic, i.e., allows a large diversity of configurations. This means that each step shown in Algorithm 1 can be instantiated in different manners, making the exploration possibilities wider.
Different similarities measures SM and different types of networks NT, such as Knn and Gaussian, can be used to model the association rules R (line 1). The way the network is built guides the exploration with different objectives through its connections.

To select the subset of potentially interesting rules, “SelectSubSetRules” (line 4), different network measures NM can be used. These measures aim to find the rules that are considered most relevant to the network. Therefore, several measures can be used, such as centrality measures and network exploration algorithms.

To classify the rules (line 8), all the network based transductive algorithms can be used, setting a wide range of bias in the exploration of the rules.

In “StoppingCriterionMet()” (line 10), the stopping criterion can have different objectives. The criterion can be a maximum number of iterations, total execution time, a number of explored rules, etc. This criterion can stop the process according to the final user’s needs, improving the user’s experience with the approach.

IV. Experiments

In order to demonstrate the feasibility of the proposed approach, experiments were carried out. Since the approach is generic, allowing a large diversity of possibilities, different configurations were used to explore different interesting results.

As shown in Algorithm 1, the approach starts the exploration by modeling the association rules R in a network. So, first of all, it was necessary to select a set of association rules R. In the experiments, the selected datasets, used to obtain R, can be divided in two groups: relational and transactional. In all of them the rules were extracted using an Apriori\(^1\) implementation with a minimum of two items and a maximum of five items per rule. The minimum of two items was selected so the rules had, at least, one item in the antecedent and one item in the consequent. The maximum of five items was selected so the rules did not get too long, differentiating the user's classification process.

The relational datasets were Weather-Nominal (5;14), Contact-Lenses (5;24), Balloons (5;76) and Hayes-Roth (5;132). The numbers in parentheses indicate, respectively, number of attributes and number of instances. The first two are available in Weka\(^2\); the other two in the UCI Repository\(^3\). Before extracting the rules, these datasets were converted to a transactional format, where each transaction was composed by pairs of the form "attribute=value". Besides, in order to produce a suitable number of rules a maximum support (min-sup) of 0.0% and a minimum confidence (min-conf) of 0.0% were used in Weather-Nominal, Contact-Lenses and Balloons – in fact, all possible combinations were generated in each one. The values of min-sup=2.5% and min-conf=0.5% were used to Hayes-Roth set for the same reasons. 722 rules were obtained for Weather-Nominal, 890 for Contact-Lenses, 772 for Balloons and 889 for Hayes-Roth.

The transactional datasets were Groceries (9835;169) and Sup (1716;1939). In this case, the numbers in parentheses indicate, respectively, number of transactions and number of distinct items. The first one is available in the R Project for Statistical Computing through the package "arules"\(^4\). The last one was donated by a supermarket located in São Carlos city, Brazil. With the Groceries dataset 1092 rules were generated using a min-sup of 0.7% and a min-conf of 0.5% and with Sup 1149 rules considering a min-sup of 1.25% and a min-conf of 0.5%. As before, the support and confidence low values were selected to generate a suitable number of rules.

As presented in Algorithm 1, the first step (line 1) is responsible to model the rules in a network. The network was modeled using a modification of the Jaccard (Jacc) measure as SM, a measure mostly used in this context to compute the similarity between itemsets. The measure can be seen in Equation 3, where \(LHS(r)\) returns the Left-Hand side of a rule \(r\) and \(RHS(r)\) the Right-Hand side. In this work, the measure considers the two sides of a rule separately, so the rule implication is accounted and the items occurring in different sides are not considered shared items.

\[
Jacc(r_1,r_2) = \frac{JaccA(r_1,r_2) + JaccC(r_1,r_2)}{2}, \text{ where}
\]

\[
JaccA(r_1,r_2) = \frac{LHS(r_1) \cap LHS(r_2)}{LHS(r_1) \cup LHS(r_2)},
\]

\[
JaccC(r_1,r_2) = \frac{RHS(r_1) \cap RHS(r_2)}{RHS(r_1) \cup RHS(r_2)}
\]

Using Jacc as SM, three different NT types were accounted to build the network: one that limits the number of connections (Knn); one that benefits higher similarities values (Gaussian); one that considers the rules’ similarities without any restriction or change (Similarity). In Knn each rule is connected to its K most similar rules, using the Jacc values as weights – in this case, it is important to have in mind that the connections are not mutual, i.e., a connection between \(r_1\) and \(r_2\) does not imply in a connection between \(r_2\) and \(r_1\). In Gaussian each rule is connected to all the other rules and the weights among the rules are computed by applying the Jacc values to a Gaussian function \(e^{-\frac{d^2}{2\sigma^2}}\), being \(\sigma\) a parameter. Finally, in Similarity each rule is also connected to all the other rules and, in this case, the weights of the connections are the Jacc values without any further processing. These NT were selected so the impact of the number of connections could be studied. In Knn, the values selected for \(K\) were 1, 7, 17, 37 and 57, in order to analyze its variation in the final results. In Gaussian, the values selected for \(\sigma\) were 0.05, 0.2, 0.35, 0.5 and 0.75, also in order to analyze its variation in the results.

Many NM network measures, such as centralities measures, can be used to select the rules that are going to be classified by the user (line 4). In the experiments, the output

---

1 Developed by Christian Borgelt: http://www.borgelt.net/apriori.html.
2 http://www.cs.waikato.ac.nz/ml/weka/.
3 http://archive.ics.uci.edu/ml/. In Balloons the four available sets were joined. In Hayes-Roth, the first attribute, regarding an ID number, was not considered.
4 http://cran.r-project.org/web/packages/arules/index.html.
degree \((OD)\) was used as \(NM\). This \(NM\) sums all the weights a rule \(r_i\) has to all the other rules in the network (it considers the sum of the weights of its connections). This step (line 4) selects the \(N_R\) best rules, according to the \(NM\) measure, as the \(N_R\) worst rules. In the experiments, \(N_R\) was set to 5; therefore, in each iteration 10 rules are evaluated by the user. This number was selected so the number of rules to be evaluated by the user would not be so large, making the classification process more difficult, and not so small, to lack information per iteration. It is important to remember that in the first iteration the whole ruleset is considered; after that, only the rules that are considered “Interesting” by the classifier are considered in the process of selecting the rules.

In line 5 of Algorithm 1, the user must classify the selected rules manually. Once it is difficult to obtain a user’s evaluation in many different rulesets, a user simulation was used. To simulate the user’s classification, an objective set \(OS\) was created. The idea was to consider some rules in \(R\) as the interesting ones of the domain. For that, in the relational datasets, since all the instances had a defined class, a tree based classifier was applied (C4.5). The rules obtained by the classifier were considered the ones that best explain the datasets and, therefore, were used to construct these \(OS\) sets. Although the rules in these \(OS\) sets contain as consequent the classes of their datasets, the rules in \(R\) contain any pair “attribute=value”, since all possible relations were extracted. On the other hand, since the instances in the transactional datasets do not have a defined class, these \(OS\) sets were built by randomly selecting some rules in \(R\). The number of rules in the \(OS\) sets were: Weather-Nominal \(|OS| = 5\) (0.69%), Contact-Lenses \(|OS| = 4\) (0.45%), Balloons \(|OS| = 7\) (0.91%), Hayes-Roth \(|OS| = 12\) (1.35%), Groceries \(|OS| = 7\) (0.64%) and Sup \(|OS| = 9\) (0.78%).

Based on the built \(OS\) sets, the simulation of the user’s classification was based on a threshold \(t\) value. For each rule \(r\) to be classified, a value \(v\) was computed. Two different ways of calculating \(v\) were tested: Sim.Mean (Equation 5) and Sim.Clos (Equation 6) (in both of the equations \(|OS|\) represents the number of rules in \(OS\) and \(OS_i\) the i-th rule in \(OS\)). In Sim.Mean each rule \(r\) to be classified receives a value \(v\) computed by considering its similarity in relation to all the rules in \(OS\). This strategy favors \(OS\) sets that present rules with high similarities among them. In Sim.Clos each rule \(r\) to be classified receives a value \(v\) computed by considering its similarity with the most similar rule in \(OS\). This strategy favors \(OS\) sets that present rules with small similarities among them. Based on this information, it is checked if the computed value \(v\) is ≥ the threshold \(t\) value, i.e., if the similarity of the current rule \(r\) in relation to the rules in \(OS\) is high. If so, the rule is classified as “Interesting”; otherwise as “Not-Interesting”. In the experiments, the threshold \(t\) value was calculated on demand according to the first \(2N_R\) selected rules (first iteration of line 4). This \(t\) value was computed, using Equation 4, considering the similarity among the rules to be classified \(R_i\) and the rules in \(OS\) set (in this case, \(Sim.S\) can be Equation 5 or 6, depending on the equation used to compute \(v\)).

\[
t = \frac{\sum_{i=1}^{2N_R} Sim.S(R_{t_i}, OS)}{2N_R} \tag{4}
\]

\[
Sim.Mean(r, OS) = \frac{\sum_{i=1}^{\text{|OS|}} Jacc(r, OS)}{\text{|OS|}} \tag{5}
\]

\[
Sim.Clos(r, OS) = \max_{\forall i \in |OS|} (Jacc(r, OS_i)) \tag{6}
\]

To classify the rules (line 8), two classifiers were used: GFHF and LLGC. These two classifiers present great results in the literature, as seen in [11] and [10]. Besides the good results, these two classifiers were selected to analyze how a parameter variation impacts on the final results and how many good results a classifier without parameter can obtain. The LLGC classifier requires the configuration of the \(\alpha\) parameter (see Section II). In the experiments, the values selected for \(\alpha\) were 0.1, 0.3, 0.5, 0.7 and 0.9.

Finally, it is necessary to define the stopping criterion (line 10). This criterion is responsible to stop the looping. In the experiments, the process continued until all the rules contained in \(OS\) were found. Therefore, the stopping criterion ensured that all the rules contained in \(OS\) were found, allowing the approach to be compared with others in the literature.

A traditional post-processing approach was used as a baseline to compare the results obtained by the proposed approach. Using the same rulesets \(R\), experiments were carried out using objective measures to find all the rules contained in the same \(OS\) sets. Therefore, the stopping criterion was the same used in the proposed approach, i.e., the process did not stop until all the rules in \(OS\) were found. To explore the rules, a ranking was created using 18 different objective measures: in this case, these 18 measures were calculated for each rule \(r\) in \(R\) and, then, their rank mean was used to rank the rules. Based on this rank, a search was carried out in \(R\) until all the rules in \(OS\) were found. The evaluation measures approach are briefly explained in Section II and the post-processing baseline, as used here, in [5].

V. RESULTS AND DISCUSSION

The best results obtained using each classifier, combined with each user simulation measure (Equations 5 and 6), are shown in Table I. The first column presents the analyzed dataset. The second column the network type \(NT\) used to model the rules \((k=\text{Knn, } \sigma=\text{Gaussian, Sim=Similarity})\). The third column presents the classifier that was used to propagate the labels. “Objective” is used to represent the results of the approach that uses objective measures to explore the rules, as previously explained. The fourth column shows the number of iterations needed to find all the rules in \(OS\). The fifth column presents the number of rules explored by the user to find all the rules in \(OS\). This column contains the number of rules evaluated by the user in each iteration (10 rules per iteration) plus the number of rules the transductive algorithm classified as “Interesting” in the last iteration. The rules classified by the user are considered because the user analyzed them. The rules classified as “Interesting” by the classifier are considered too because these rules are the final result of the approach and some rules contained in \(OS\) are inside this set. The sixth column presents the percentage of reduction in the exploration space, i.e., the percentage of the ruleset that will not be
As can be observed, the proposed approach presented better results than the objective measures approach in 5 of 6 datasets (the best results regarding each dataset are boldface). In the relational datasets the distances between the two approaches, regarding the exploration space reduction, were smaller than the transactional ones, although the values were significant (in Balloon 9.58% (79.92%-70.34%) (1 iteration, 155 explored rules), in Contact-Lenses 17.41% (81.57%-64.16%) (1 iteration, 164 explored rules), in Hayes-Roth 28.57% (78.74% - 50.17%) (3 iterations, 189 explored rules)). In the transactional datasets the distances were greater than 50% (in Groceries 64.56% (71.15%-6.59%) (9 iterations, 315 explored rules), in Sup 56.14% (56.40%-0.26%) (17 iterations, 501 explored rules)). In the case where the proposed approach lost (Weather-Nominal), the difference obtained between the approaches was 3.05% (83.38%-80.33%).

The results seems to be promising in both relational and transactional datasets compared to the objective measures approach. The $k$-nn network presented the best results compared to the other two networks — the best configuration in 5 of 6 datasets. Besides, in 4 of 5 the $k = 17$ showed better results. The Gaussian network showed good results in 2 datasets, but not as the best result. The Similarity network won only in one dataset. This result indicates that restricting the number of connections can be good for the classification processes over the association rules. It can also be seen that the LLGC classifier showed better results in 5 of 6 datasets, using, in the majority of the cases, the parameter $\alpha = 0.5$.

VI. CONCLUSION

This work presented an association rule post-processing approach that uses networks and transductive learning. The paper discussed the main ideas of the approach. Experiments were carried out aiming to find a set of objective rules (OS) among the rules to be explored. To analyze the results, a comparison with the objective measures approach was made. As presented in Section V, the proposed approach shows better results than the objective measures approach, presenting a great potential in post-processing association rules, since it reduces the exploration space and directs the user to the knowledge considered interesting.

The proposed approach opens a wide area of research, since many possible configurations and measures can be explored. Aiming to decrease the number of rules to be explored, a rank will be created on the output of the process to direct the user to what he/she thinks is most important. Therefore, the user will not need to check all the outputted rules to find the ones that are considered interesting to him/her. Besides, it is necessary to explore other similarity measures, network measures and network types to understand the different biases that can be used. Also, different classifiers need to be tested to demonstrate how different classifiers behave in the considered context. Finally, a case study will be carried out where the proposed approach will be evaluated in a real application.

ACKNOWLEDGMENT


REFERENCES


