Mining Predictive Redescriptions with Trees

Tetiana Zinchenko*
MPI Informatik
Saarbrücken, Germany
zinchenko@mpi-inf.mpg.de

Esther Galbrun†
Boston University
Boston, MA, USA
galbrun@cs.bu.edu

Pauli Miettinen
MPI Informatik
Saarbrücken, Germany
pmiettin@mpi-inf.mpg.de

***Abstract*—In many areas of science, scientists need to find distinct common characterizations of the same objects and, vice versa, identify sets of objects that admit multiple shared descriptions. For example, a biologist might want to find a set of bioclimatic conditions and a set of species, such that this bioclimatic profile adequately characterizes the areas inhabited by these fauna. In data analysis, the task of automatically generating such alternative characterizations is called redescription mining. A number of algorithms have been proposed for mining redescriptions which usually differ on the type of redescriptions they construct. In this paper, we demonstrate the power of tree-based redescriptions and present two new algorithms for mining them. Tree-based redescriptions can have very strong predictive power (i.e. they generalize well to unseen data), but unfortunately they are not always easy to interpret. To alleviate this major drawback, we present an adapted visualization, integrated into an existing interactive mining framework.**

***Keywords*-Redescription mining; Decision trees; Interactive data mining; Visualization**

I. Introduction

Redescription mining is a method that aims at finding alternative descriptions of the same objects, or, in other words, providing information on the same entities from different perspectives. It is a tool for knowledge discovery that helps integrate multiple characterizing data sets of diverse origins and reason across them. Matching the objects across the available datasets—for instance by exploiting information shared between the datasets via entity resolution and matching techniques—is a prerequisite for redescription mining.

Several approaches have been proposed to mine redescriptions. Here, we consider the case of two data sets and we focus on approaches based on decision tree induction [1]. Applying such techniques to redescription mining is not a new idea [2]. Our algorithms, however, differ from the existing ones both in their inner working and in their capabilities. Indeed, we present two algorithms based on decision tree induction which unlike existing tree-based methods can handle non-binary data, and scale well. Importantly, our methods also generalize well to unseen data.

While tree-based redescription mining algorithms can have many desirable properties, they all share an important weakness: expressing the decision trees as Boolean formulae makes them very hard to interpret. To help interpreting the redescriptions, we present an interactive visualization method for tree-based redescriptions.

We have integrated our tree-based algorithms and visualizations into our redescription-mining tool `Siren`. This allows the user to use both tree-based and other algorithms and compare and visualize (and edit) their results in an intuitive and powerfull way.

We briefly explain our tree-based algorithms in Section III, and discuss the visualization of the trees in Section V.

II. Background and Related Work

The input of our methods is a pair of data matrices, where one side contains only Boolean variables while the other side may contain real-valued variables. Throughout, we refer to the two sides as the left and right-hand sides and assume, without loss of generality, that the former is the fully Boolean side. We will represent the data using two matrices D_{L} and D_{R} over two sets of variables, V_{L} and V_{R}, respectively.

The set of entities characterized by the two sides is denoted by E, hence both matrices have $|E|$ rows. The value of $D_{\mathrm{L}}(i,j)$ is the value of variable $v_j \in V_{\mathrm{L}}$ for entity $e_i \in E$.

If $v \in V$ is Boolean, we interpret the column corresponding to it as a truth value assignment for $e \in E$ in a natural way. If $v \in V$ is real-valued, we consider a threshold t and the truth value assignments induced by relations such as $t < v$ or $v \leq t$. These truth assignments and their negations constitute the set of *literals* which can then be combined into *queries* with operators $\wedge$ (and) and $\vee$ (or). A *redescription* R is a pair of queries $(q_{\mathrm{L}}, q_{\mathrm{R}})$ over variables from V_{L} and from V_{R}, respectively.

The support of a query q, $\mathrm{supp}(q)$, is the set of entities for which the query holds true, and the support of a redescription is simply the intersection of supports of its two queries.

The main quality of a redescription is the similarity of the supports of the two queries that it consists of, also called its *accuracy*. The Jaccard similarity coefficient is the measure of choice to evaluate the accuracy of redescriptions, being at once simple, symmetric and intuitive. It is defined as follows:

$$\mathrm{J}(R) = \mathrm{J}(q_{\mathrm{L}}, q_{\mathrm{R}}) = \frac{|\mathrm{supp}(q_{\mathrm{L}}) \cap \mathrm{supp}(q_{\mathrm{R}})|}{|\mathrm{supp}(q_{\mathrm{L}}) \cup \mathrm{supp}(q_{\mathrm{R}})|}.$$

Besides accuracy, several other factors impact the quality of a redescription. For instance, we are not interested in redescriptions which are supported by only a handful of entities or conversely by almost all of them, i.e. redescriptions for which $|\mathrm{supp}(q_{\mathrm{L}}) \cap \mathrm{supp}(q_{\mathrm{R}})|$ is too small or too large.

*Current affiliation: PRISM Informatics Deutschland GmbH, Germany
†Current affiliation: INRIA Nancy Grand-Est, France

Furthermore, redescriptions should also be statistically significant. To evaluate the significance of results, we use p-values as in [3]. Our algorithms incorporate parameters to account for these preferences.

In short, given two data matrices, redescription mining is the task of searching for the best matching pairs of queries, one query over each of the data sets.

Related Work. The problem of mining redescriptions was introduced by Ramakrishnan et al. [2]. They proposed an algorithm called `CARTwheels` based on decision trees, with an alternating approach to grow decision trees from which redescriptions are then extracted [4].

Algorithms employing heuristics—rather than growing decision trees—to produce pairs of queries that are almost equivalent on the given data set were presented in [5]. Theses algorithms rely on different strategies to prune the search space. By means of an efficient on-the-fly discretization, such heuristics were extended to real-valued data, resulting in the `ReReMi` algorithm [3].

The main feature of redescriptions is their ability to describe data from different points of view, i.e. their "multi-view" aspect. A similar approach is taken by some other methods such as Multi-label Classification [6], Emerging Patterns [7], and Subgroup Discovery [8] to name a few (see [3] for more details). The main differences between redescription mining and these methods are that redescription mining aims to find multiple descriptions simultaneously for a subset of entities which is not specified *a priori*, it selects only relevant variables from a potentially large number, and it is symmetric, in the sense that it handles both sides of the data similarly.

III. Algorithms

Our algorithms rely on decision tree induction for mining redescriptions. More specifically, we grow two trees in opposite directions, gradually increasing their depth and matching their leaves. We use *classification and regression trees (CART)* [9] for this purpose, but any other approach to induce decision trees can be exploited as well. What follows is a succinct description of the algorithms; more detailed explanation can be found in [10].

Growing Trees. Our input is a pair of data matrices, where one side (w.l.o.g. the left-hand side) contains Boolean attributes while the other side may contain real-valued attributes.

We use the Boolean variables to initialize our procedure. That is, a column of D_{L} provides the target vector for building the first tree. The vector of predictions output at one step is then used as the target vector when growing a tree over attributes from the other side during the next step, in alternating iterations.

This procedure continues to alternate between growing trees over either side of the data, until any of the following three stopping conditions is met: *i)* no tree can be induced with the given parameters, *ii)* the maximal tree depth chosen by the user is reached, or *iii)* the prediction vector obtained from the new tree is identical to the one previously obtained on that side, i.e. the tree growing procedure has reached a fixed point.

A tunable parameter, $l_{\min}$, controls the minimum number of entities allowed in any leaf of the tree. This helps to combat overfitting and terminate the tree induction earlier.

We present two algorithms which follow the process described above but differ in their strategy for growing trees.

The `SplitT` Algorithm. Our first algorithm grows a new classification tree at each iteration while progressively increasing their depth. As explained above, a column from the Boolean side, D_{L}, is initially used as target vector to induce a tree over the other side, D_{R}. This first iteration produces a tree of depth one with two leaves, which are labelled according to the majority class of the entities they contain, one positive and one negative. The corresponding binary prediction vector for the entities is then used as a target vector to grow a new tree over D_{R}, this time of depth two. In turn, the prediction vector is used as a target to learn from scratch a tree of depth two over D_{R}, and so on.

The `LayeredT` Algorithm. Our second algorithm grows trees layer by layer instead of building a new tree from scratch in each iteration. One layer is added to the current candidate tree by appending a new decision tree of depth one to each of its branches, each learnt independently from the others.

Extracting Redescriptions. At the end of the alternating process we obtain a pair of decision trees, over either sets of variables. Their leaves are labelled as belonging either to the positive or the negative class and matched throught the common entities.

The extraction of a redescription from such a pair of decision treesworks as follows. From either tree we obtain a query over the variables from that side charaterizing the positive class. One literal is constructed for each node of the decision tree using the associated splitting variable and threshold. Then, literals leading to the positive leaves are combined into a query, using conjunctions ($\wedge$) to combine literals within one branch and disjunctions ($\vee$) to combine different branches.

The output of our algorithms consists of the collected redescriptions over all induced tree pairs.

Extending to fully non-Boolean settings. Both of our proposed algorithms can be applied when neither side is fully Boolean. In such cases, a binarization or a clustering routine can be applied to one of the sides to process real-valued and categorical attributes as necessary. However, this might require domain knowledge and some amount of trial and error to determine the most appropriate discretization settings.

IV. Experimental Evaluation

In this section we investigate the predictive power of redescriptions. Extensive experiments on the behaviour of our tree-based algorithm on synthetic and real-world data can be found in [10].

Datasets. In our first dataset, Bio, the entities represent geographic areas of Europe, the left-hand side records the

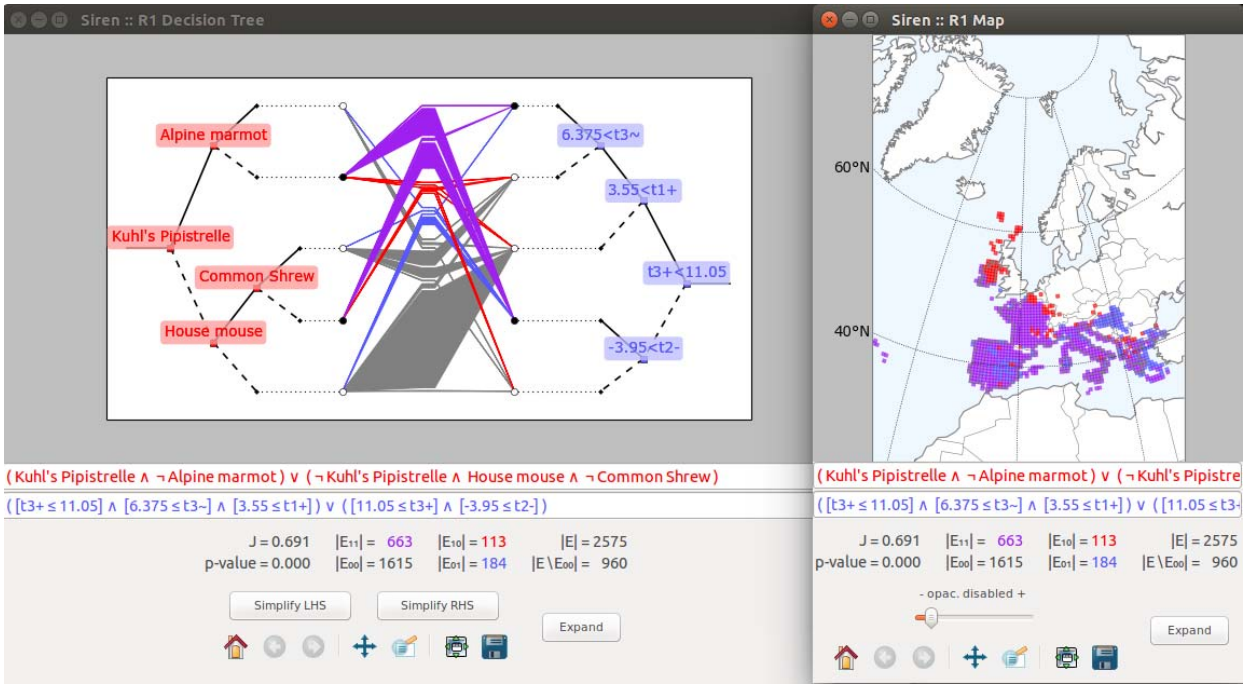

Fig. 1. Visualizations a redescription from Bio, its queries displayed as a pair of classification trees (left) and its support plotted on a map (right).

presence of various mammals species [11] while the right-hand side consists of bioclimatic variables, that is, monthly average rainfall and monthly average, minimum, and maximum temperatures [12] ($|E| = 2575$, $|V_{\mathrm{L}}| = 194$ and $|V_{\mathrm{R}}| = 48$). We extract a subset of this data, which we denote as Bio$_S$, by selecting only areas from Northern Europe, namely areas located at latitudes between 50 and 71° North, and keeping only the average monthly temperatures and rainfall ($|E| = 1271$, $|V_{\mathrm{L}}| = 194$ and $|V_{\mathrm{R}}| = 24$). Our third data set, DBLP, is extracted from the popular computer science bibliography.[1] The entities are researchers and one side records the co-authorship graph while the other side records the number of their publications in each of the major conferences considered ($|E| = 2345$, $|V_{\mathrm{L}}| = 2345$, and $|V_{\mathrm{R}}| = 19$).

Methods. We compared the redescriptions obtained with our proposed algorithms `SplitT` and `LayeredT`, to those returned by the `ReReMi` [3] and `CARTwheels` [2] algorithms (implementations provided by the authors).

To study the ability of the redescription to generalize to unseen data, we selected 80% of the entities, mined redescriptions from this training set then assessed their accuracy on the full original data sets.

For the DBLP data set, entities were split at random between training and hold-out sets. For Bio and Bio$_S$, we had to take into account the North–South trends in climate data: predicting conditions in North with data from South (or vice versa) is unlikely to succeed. Hence we sampled longitudes, and the hold-out set consisted of all point laying (roughly) on the sampled longitudes (in total 20% of entities were held out).

[1] http://www.informatik.uni-trier.de/~ley/db/

With Bio and Bio$_S$ we set the minimum support to 200, the maximum support to 1800 (for DBLP 5 and 1300 respectively) and the maximum p-value to 0.05 for all algorithms.

Results. The `CARTwheels` algorithm was only able to run on the Bio$_S$ data, running out of memory with the others. As `CARTwheels` also requires all-binary data, we binarized the climate data in Bio$_S$. We tested various techniques, and the results we report here are based on segmenting every variable in 4 segments, as this gave the best results with all of the tested methods.

As our goal is to find redescriptions that hold well in the unseen data, we measure the quality of the generalization using the average ratio of the accuracy of the redescriptions in the training data to their accuracy in the full data. If this ratio is significantly below 1, we conclude that the redescriptions did not generalize well.

Statistics for this experiment are reported in Table I. On Bio$_S$, `SplitT` and `CARTwheels` achieve essentially the same quality, both returning high-accuracy redescriptions that generalize well. `CARTwheels`' larger number of redescriptions is due to the fact that it returned multiple almost-identical redescriptions. `LayeredT` also reported redescriptions that generalize well, although their overall accuracy was lower than with the other methods, while `ReReMi` apparently overfitted. On Bio, we observe similar behaviour from `SplitT`, `LayeredT`, and `ReReMi` (`CARTwheels` could not be used on this data) with `SplitT` giving the best overall results.

On the DBLP data, `SplitT` is the only algorithm returning results that are both accurate and generalize well, although most of its redescriptions had lower supports than the redescriptions returned by the other methods: `LayeredT`

TABLE I
ACCURACY OF REDESCRIPTIONS. WE REPORT THE NUMBER OF REDESCRIPTIONS MINED (#), THE AVERAGE ACCURACY (JACCARD COEFFICIENT, J) IN THE TRAINING DATA AND OVERALL AS WELL AS THE AVERAGE OF THE RATIO BETWEEN THE ACCURACY IN THE TRAINING SET AND OVERALL (± STANDARD DEVIATION).

Data	Algorithm	#	J training	J overall	J ratio
Bio_S	SplitT	46	0.87(±0.09)	0.86(±0.09)	0.98(±0.01)
	LayeredT	77	0.61(±0.17)	0.61(±0.17)	0.99(±0.02)
	ReReMi	9	0.88(±0.04)	0.58(±0.22)	0.66(±0.24)
	CARTwheels	88	0.87(±0.07)	0.87(±0.07)	0.99(±0.01)
Bio	SplitT	137	0.82(±0.12)	0.81(±0.12)	0.99(±0.01)
	LayeredT	156	0.49(±0.20)	0.49(±0.20)	1.01(±0.01)
	ReReMi	56	0.92(±0.04)	0.55(±0.27)	0.59(±0.30)
DBLP	SplitT	37	0.75(±0.19)	0.70(±0.17)	0.94(±0.07)
	LayeredT	577	0.13(±0.07)	0.12(±0.07)	0.96(±0.11)
	ReReMi	23	0.36(±0.05)	0.06(±0.04)	0.18(±0.15)

returns redescriptions with very low accuracy, and while ReReMi's accuracy is better in the training data, it again overfits and has very low generalizability.

V. VISUALIZATION

As evidenced by the above empirical investigation, tree-based redescriptions are a powerful way to explain structures in the data. Unfortunately, they can also be very hard to interpret. Consider, for example, the following redescription:

$$\begin{aligned} q_{\text{L}} &= (\text{Kuhl's Pipistrelle} \wedge \neg\text{Alpine marmot}) \\ &\quad \vee (\neg\text{Kuhl's Pipistrelle} \wedge \text{House mouse} \wedge \neg\text{Common Shrew}) \\ q_{\text{R}} &= ([t3+ \leq 11.05] \wedge [6.375 \leq t3 \sim] \wedge [3.55 \leq t1+]) \\ &\quad \vee ([11.05 < t3+] \wedge [-3.95 \leq t2-]) \end{aligned}$$

This redescription encodes a tree where we can take two branches from the root (the root variable appears as itself and in a negated format), and on both branches there are one or two leaves that contribute towards the support. To help the user in understanding the redescriptions better, we propose a visualization technique for the tree-based redescriptions, seen in Figure 1, left. We included this tree-based visualizations, along with the new mining algorithms, into our interactive redescription mining framework, Siren.[2]

Visualizing a decision tree is, of course, a well-known and not too complicated problem. What makes our setting special, is that instead of one decision tree, we must visualize two decision trees that are linked in their leaves. We visualize this linkage by drawing lines between the leaves: a line connects two leaves if the data has an entity for which the corresponding paths in the trees hold true. However, not every path in the tree participates in the query—otherwise we would always cover the whole data—and the color of the line encodes this information. Purple lines correspond to entities that belong to the support of the redescription, i.e. $\text{supp}(q_{\text{L}}) \cap \text{supp}(q_{\text{R}})$; red lines are the ones that belong to $\text{supp}(q_{\text{L}})$ but not to $\text{supp}(q_{\text{R}})$ and vice versa for blue lines; finally the gray lines are the ones that do not belong to $\text{supp}(q_{\text{L}}) \cup \text{supp}(q_{\text{R}})$.

To see the power of our visualization, consider the query q_{R}. The root splits the entities into two sets, depending on whether March's maximum temperature is below or above 11.05. But when it is below (bottom right branch in Figure 1 left), only very few entities contribute to $\text{supp}(q_{\text{L}}) \cap \text{supp}(q_{\text{R}})$, indeed, there are only few purple lines coming out of that leaf. Consequently, the analyst could consider removing the bottom branch, leaving only

$$q_{\text{R}} = [11.05 < t3+] \wedge [-3.95 \leq t2-]$$

as the right-hand side query. Indeed, if he does so, the Jaccard coefficient drops only from 0.691 to 0.665.

VI. CONCLUSIONS

We presented two new decision-tree based algorithms for redescription mining, SplitT and LayeredT. Redescription mining is a powerful data analysis tool, but its usability is significantly reduced unless we can trust that the redescriptions found will also hold on unseen data. The goal of our algorithms is to succeed on this prediction task. Our experiments showed that our algorithms achieve this goal, generalizing much better than the state-of-the-art ReReMi. Compared to the other tree-based algorithm, CARTwheels, our algorithms were able to find more interesting redescriptions, scale to larger data sets, and most importantly, handle non-Boolean data. Combined to a tailored visualization feature integrated in the Siren interface, these algorithms significantly enhance the toolbox of redescription mining.

[2]See also http://siren.mpi-inf.mpg.de/

REFERENCES

[1] J. R. Quinlan, "Induction of decision trees," *Mach. Learn.*, vol. 1, no. 1, pp. 81–106, 1986.

[2] N. Ramakrishnan, D. Kumar, B. Mishra, M. Potts, and R. F. Helm, "Turning CARTwheels: an alternating algorithm for mining redescriptions," in *KDD*, 2004, pp. 266–275.

[3] E. Galbrun and P. Miettinen, "From black and white to full color: Extending redescription mining outside the Boolean world," *Stat. Anal. Data Min.*, vol. 5, no. 4, pp. 284–303, Apr. 2012.

[4] D. Kumar, "Redescription mining: Algorithms and applications in bioinformatics," Ph.D. dissertation, Virginia Polytechnic Institute and State University, 2007.

[5] A. Gallo, P. Miettinen, and H. Mannila, "Finding subgroups having several descriptions: Algorithms for redescription mining." in *SDM*, 2008, pp. 334–345.

[6] G. Tsoumakas, I. Katakis, and I. Vlahavas, "Mining multi-label data," in *Data mining and knowledge discovery handbook*. Springer, 2010, pp. 667–685.

[7] P. K. Novak, N. Lavrač, and G. I. Webb, "Supervised Descriptive Rule Discovery: A Unifying Survey of Contrast Set, Emerging Pattern and Subgroup Mining," *J. Mach. Learn. Res.*, vol. 10, pp. 377–403, 2009.

[8] L. Umek *et al.*, *Subgroup Discovery in Data Sets with Multi–dimensional Responses: A Method and a Case Study in Traumatology*. Springer, 2009.

[9] L. Breiman, J. Friedman, R. Ohlsen, and C. Stone, *Classification and regression trees*. Belmont, CA: Wadsworth International Group, 1984.

[10] T. Zinchenko, "Redescription mining over non-binary data sets using decision trees," Master's thesis, Saarland University, 2015.

[11] A. Mitchell-Jones *et al.*, *The atlas of European mammals*, London, 1999.

[12] R. Hijmans *et al.*, "Very high resolution interpolated climate surfaces for global land areas," *Int. J. Climatol.*, vol. 25, pp. 1965–1978, 2005.