Towards Explainable Relational Boosting via Propositionalization

Blaž Škrlj[1,2][0000-0002-9916-8756] and Nada Lavrač[1,2]

[1] Jožef Stefan Institute, Jamova 39, Ljubljana, SI
[2] Jožef Stefan International Postgraduate School, Jamova 39, Ljubljana, SI
{blaz.skrlj,nada.lavrac}@ijs.si

Abstract. Many problems can be modeled by using relational structures, such as graphs or relational databases. Propositionalization refers to the process of transforming a relational database into a propositional (tabular) representation, suitable for a wide repertoire of machine learning algorithms, including contemporary black-box classifiers such as deep neural networks or ensemble-based classifiers such as Gradient boosted tree ensembles. Even though such black-box classifiers often outperform symbolic learners, their results are hard to interpret by humans. This paper explores the means to improve black-box classifier interpretability in a relational setting. To this end, we conducted a series of experiments to evaluate how relational features, constructed using Aleph, RSD and Tertius propositionalization algorithms, impact the interpretability and performance of black-box classifiers such as Gradient boosted tree ensembles. We show how improved interpretability can be achieved by combining XGBoost with SHAP, an algorithm that leverages the ideas from coalitional game theory to assign importance scores to the obtained relational features, offering both the state-of-the-art performance as well as insights into the feature impacting individual predictions.

Keywords: Relational learning, boosting, propositionalization

1 Introduction

Relational data mining is widely used in many areas of science, including biology, sociology, and more. The most common task solved by relational learners is entity classification, which is the focus of this work. In relational data classification, two conceptually different approaches are commonly used[3]:

- In the first approach, a learner can learn directly from the given data. Here, a relational learner, e.g., Aleph [20], is given a relational database, with class-labelled training instances as input to the learner. The learner leverages the relational structure of the database to construct a set of interpretable rules, which cover the majority of positive examples and can be used as a classifier (new instances can be classified using the learned rules).

- The second approach implements a two-step process named propositionalization. In the first step, the relational data is transformed into a propositional format, and in the second step, the classification of transformed tabular data is solved by effective propositional learning algorithms, including state-of-the-art deep neural networks [8] or Gradient boosting machines [3].

In this work we focus exclusively on the second group of algorithms, i.e. propositionalization-based learners. Propositionalization algorithms take a relational database as input and transform it into a propositional table, consisting of labeled instances described by features that have been automatically derived from the underlying relational structure. These features correspond to simple conjuncts of the form $f_i := p_1 \wedge p_2 \cdots \wedge p_k$, where f_i denotes the i-th relational feature (a conjunct of elementary relational features) and p_j represents the j-th elementary relational feature. A valid feature could be, for example, $f_i := \text{atomType}(a) \wedge \text{charge}(b)$. Each elementary relational feature p_j describes some relational property of the dataset. For example, if instances correspond to researchers who published papers in conferences, p_j will return value *true* or *false* for a given author. Property p_j can be a rather complex relational query involving multiple relations (as long as that query returns either true or false), or the result of some other aggregation function. For example, the property could be "does author X have a paper published at the ECML/PKDD conference?" or "how many papers has author X published at the ECML/PKDD conference?". While such feature construction could be done manually by a data analyst, we are only interested in *automated* propositionalization methods.[4]

The remainder of this work is structured as follows. First, we discuss some of the related work focusing on the task of *propositionalization*. Next, we discuss the notion of boosting and how it can be applied in combination with propositionalization. We next describe the experimental setting used to benchmark a selection of existing propositionalization approaches alongside a number of widely used propositional learners. Finally, we discuss the results and possible implications of the proposed approach.

2 Background technologies

In the following section, we discuss the related work and techniques relevant to this work.

2.1 Propositionalization

To be able to apply standard machine learning algorithms on multi-relational data, the considered relational database should be transformed into a single *tabular data format*, where each row represents a single data instance, and each

[4] Note that simplest relational features may correspond even to single propositional features of the form *table-attribute-value*, which are evaluated *true* if the given *attribute* has a particular *value* for a given training instance.

column represents a feature (a simple attribute-value, a relational query p_j, or a conjunction of such queries). This transformation into symbolic vector space (i.e. a symbolic data table format), referred to as *propositionalization*, is defined below.

Definition 1 (Propositionalization). *Consider the input of a given data type and format and heterogeneous background knowledge of various data types and formats. Propositionalization corresponds to the process of constructing a tabular (matrix) representation of the data enriched with the background knowledge, where each row represents a single data instance and each column represents a feature in a d-dimensional binary vector space B^d.*

Propositionalization thus transforms a complex data structure such as a relational database to a simpler, binary vector space, where each feature reflects the presence (or absence) of the relational property modelled by the constructed relational feature for the given training instance. Selected propositionalization algorithms used in this study are briefly described below.

RSD [23] is a relational subgroup discovery algorithm composed of two main steps: the propositionalization step and the (optional) subgroup discovery step. The output of the propositionalization step can also be used as input to other propositional learners. RSD effectively produces an exhaustive list of first-order features that comply with the user-defined mode constraints, similar to those of Progol [16] and Aleph [20]. Furthermore, RSD features satisfy the connectivity requirement, which imposes that no feature can be decomposed into a conjunction of two or more features. Mode declarations define the algorithm's syntactic bias, i.e. the space of possible features.

Tertius (Treeliker) [7] is a top-down rule discovery system, incorporating first-order clausal logic. As no particular prediction target is specified beforehand, Tertius can be seen as an ILP system that learns rules in an unsupervised manner. Its relevance lies in the fact that Tertius encompasses 1BC, i.e. relational data is handled through 1BC transformation [6].

Aleph [20] is the most popular ILP algorithm, which is actually an ILP toolkit with many modes of functionality: learning of theories, feature construction, incremental learning, etc. It includes a feature construction functionality, which makes it also act as a propositionalization approach. Aleph uses mode declarations to define the syntactic bias. Input relations are Prolog clauses, defined either extensionally or intensionally.

2.2 Learning by Boosting

In this section, we discuss the notion of boosting machine learning, focusing on the aspects used in this work. We begin with an overview of this methodology.

Boosting refers to a group of learning algorithms. The key idea of boosting revolves around the fact that a series of weak learners can, when joined, form a strong learner [9]. In general, boosting algorithms iteratively construct a strong classifier, where weak learners' predictions are used to weight the following models, emphasizing misclassifications: misclassified input data gain a higher weight and examples that are classified correctly lose weight. Thus, future weak learners focus more on the examples that previous weak learners have *mis-classified*. Boosting algorithms are described, for example, in the AnyBoost framework [12]. Furthermore, it was shown theoretically that boosting performs gradient descent in a function space using a convex cost function. Commonly, boosting-based ensembles are not directly explainable. We next discuss how SHAP [11], a game theory-based approach, can be used to overcome this issue partially.

2.3 Explaining black-box models

Recent trends in machine learning attempt to solve hard problems using black-box, non-interpretable models. Even though such models are not explainable *per se*, individual predictions can be approximated using existing, symbolic learners in order to obtain feature relevances for a given classifier. As the final part of the learning workflow proposed in this work we exploit the recently introduced SHAP tool [11] to identify which features, constructed using propositionalization approaches, were the most relevant for a given classification problem.

Shapley regression values can be interpreted as feature importances for linear models in the presence of *multicolinearity*. The method begins by training the model on all subsets $S \subseteq F$, where F is the set of all features. Each feature is assigned a numeric score denoting the impact of this feature on a given model's performance. This impact is computed by training two models; one with a given feature and one without. Predictions of the two models are compared on a given input. As the effect of withholding a single feature from the model depends on all other features, the performance differences are computed for all possible subsets $S \subseteq F \setminus f_i$; where i denotes f_i-th feature. Shapley values ρ can be thus (for the f_i-th feature) defined as:

$$\rho_i = \sum_{S \subseteq F \setminus f_i} \frac{|S|!(|F| - |S| - 1)!}{|F|!} \big[m_{S \cup f_i}(x_{S \cup f_i}) - m_S(x_S)\big];$$

where $m_{S \cup f_i}(x_{S \cup f_i})$ corresponds to the model's performance when the f_i-th feature is considered and $m_S(x_S)$ when it is not considered. The x_S corresponds to feature values of the feature set S. The used SHAP methodology employs efficient sampling schemes for computing feature relevance. In this work, we use the "tree-explainer" module of SHAP. We refer the interested reader to [11] for a detailed overview of SHAP. The empirical setup is discussed next.

3 Propositionalization and semantic data mining

Recall the definition of propositionalization. (Definition 1) that involves heterogeneous background knowledge of various data types and formats with the aim to construct a tabular (matrix) representation of the data enriched with the background knowledge, where each column represents a feature in a d-dimensional binary vector space B^d. Let τ represent a mapping from a relational database (RDB) to a d-dimensional binary vector space $\mathcal{B}$. Let n represent the number of instances from the target table, for which relational features are constructed. Each propositionalization algorithm thus attempts to identify relational features representing the initial RDB, and can be described as the mapping:

$$\tau : \mathrm{RDB} \rightarrow \mathcal{B}^{n \times d}. \tag{1}$$

An example $b \in \mathcal{B}^{5\times 5}$ can be represented as

$$\begin{array}{ccccc}
p_1 \wedge p_2 & p_3 \wedge p_2 & p_1 \wedge p_3 & p_5 \wedge p_2 & p_4 \wedge p_1 \wedge p_5 \\
\left[\begin{matrix} 1 \\ 0 \\ 0 \\ 0 \\ 1 \end{matrix}\right. & \begin{matrix} 1 \\ 1 \\ 0 \\ 1 \\ 0 \end{matrix} & \begin{matrix} 1 \\ 0 \\ 1 \\ 0 \\ 0 \end{matrix} & \begin{matrix} 1 \\ 0 \\ 0 \\ 0 \\ 0 \end{matrix} & \left.\begin{matrix} 0 \\ 1 \\ 0 \\ 1 \\ 0 \end{matrix}\right]
\end{array}$$

Features p_j describing the data can either be low-level features describing the actual experimental data or higher-level semantic features representing higher-level concepts in **ontologies or taxonomies** encoded as the background knowledge. The latter refers to the *semantic data mining* setting. Higher-level features have higher generalization potential, given that the use of these features ensures higher coverage of induced rules.

4 Proposed approach and experimental setting

In this section we describe the experimental setting used to evaluate the performance of various classifiers trained using some of the mentioned propositionalization approaches. The two-step approach described next is summarized in Figure 1. Here, a relational database is used as input, where only a single entity (table — green circle) is considered for learning, and all others are used to construct features related to this target table. The algorithm yields a propositionalized database of dimension $n \times d$, where n corresponds to the number of examples and d to a number of (selected) relational features. Note that values of feature vectors are boolean $\{0, 1\}$.

4.1 Propositionalization algorithms implementation

We tested the following propositionalization methods, which represent a selection of well-established approaches developed in the Inductive Logic Programming (ILP) community: RSD [23], Tertius [7] and Aleph [20], presented in Section 2.1.

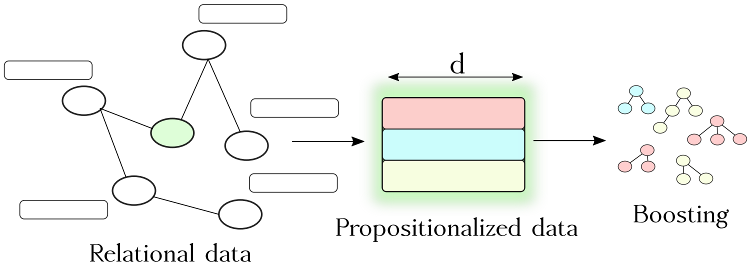

Fig. 1. Schematic representation of propositionalization combined with boosting. A d-dimensional representation of the input relational structure is used as input to a boosting-based learner.

As individual propositionalization algorithms are implemented in different programming languages using different frameworks, executing all of the approaches in the same learning setting can be very challenging. The RSD and Aleph algorithms are written in Prolog, while Tertius is in Java. We built on a recent effort to unify propositionalization approaches under a joint Python framework called PyRDM [22][5], which is freely accessible, and was updated as part of this work.[6] A contribution of this work is also a significant extension of the PyRDM library to include the newest libraries for data pre-processing and manipulation, including Orange 3 [5], Pandas [13], and Scikit-learn [17].

4.2 The classifiers used

In this subsection, we describe the classifiers that learned from the transformed, propositionalized databases. The classifiers were tested on all the propositionalization methods described in the previous section.

Support vector machines. We use the implementation, available as part of the libSVM library [2]. We used $C = 20$ for the regularization value.

Gradient boosting machines. The GBM implementation [17] used is supported in Scikit-learn. We use the default settings for learning. We varied the number of estimators from the range: 10,20,50,100.

Logistic regression. This classifier is also implemented in Scikit-learn [17]. It uses a logistic function to model a binary distributed (dependent) random variable. The regularization parameter was set to 1 (default setting).

Extra-randomized trees. Extremely randomized trees are an algorithm that attempts to capture relevant patterns by constructing larger, random trees. We use the implementation supported in Scikit-learn [17]. We set the number of estimators to 40. We set the number of samples needed for a split to 2. We varied the number of estimators from the range: 10,20,50,100.

[5] `https://github.com/xflows/rdm`

[6] `https://github.com/xflows/rdm`

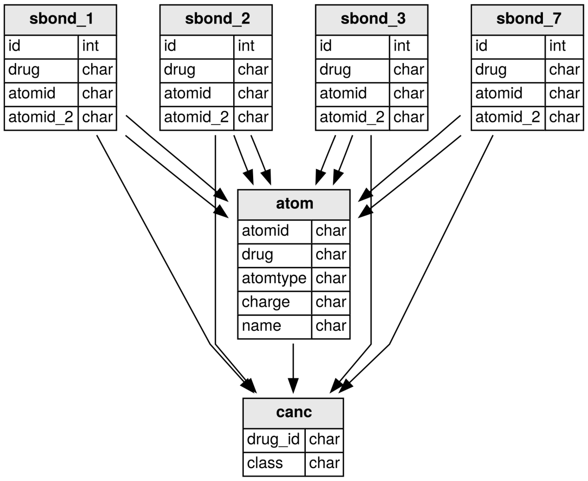

Fig. 2. Carcinogenesis database scheme. Propositionalization algorithms attempt to traverse the database by foreign keys and capture the patterns of relevance to the task.

Extreme Gradient Boosting [3]. The known XGB classifier is one of the state-of-the-art learners commonly used in contests. We varied the number of estimators from the range: 10,20,50,100.

4.3 Data sets used

In this section, we describe the data sets considered for experimental evaluation. The data sets were obtained from the CTU repository [15]. The data sets were selected as they include diverse schemas and come from different domains. An example scheme for the Carcinogenesis data is shown in Figure 2.

Carcinogenesis [21] task is to predict the carcinogenicity of a diverse set of chemical compounds. The data set was obtained by testing different chemicals on rodents, where each trial would take several years and hundreds of animals. The data set consists of 329 compounds, of which One hundred eighty-two are carcinogens.

Mutagenesis [4] task addresses the problem of predicting mutagenicity of aromatic and heteroaromatic nitro compounds. Predicting mutagenicity is an important task as it is very relevant to the prediction of carcinogenesis. The compounds from the data are known to be more structurally heterogeneous than in any other ILP data set of chemical structures. The database contains 230 compounds, of which 138 have positive levels of mutagenicity and are labelled as ‘active’. Others have class value ‘inactive’ and are considered to be negative examples. We took the data sets of the original paper [4], where the data was split into two subsets: 188 compound data set and a smaller data set with 42 compounds.

Trains [14] the task addresses the prediction of a given train's direction based on the cargo and other trains' properties. This is one of the canonical data sets for relational learning.

Facebook [10] the data set, consisting of 10,137 rows and 265 columns, represents Facebook friendships, where the task is to predict the gender of users.

4.4 The scoring scheme

We validated individual propositionalization-learner pairs using 10-fold stratified cross-validation, as implemented in PyRDM. For each fold split, we run first a given propositionalization algorithm, followed by a learner. This way, folds remain consistent; hence the algorithms' results can be compared. We computed Accuracy and AUCs for individual classifiers (all are binary classification problems). If the propositionalization-learner pair was not able to finish in 3 hours, we marked such runs as unsuccessful. We present the results as average classifier ranks, plotted in the same space as vertical lines along a horizontal line corresponding to all possible algorithm combinations. Such representation is sufficient, as we were mainly interested in whether Extreme Gradient Boosting (XGB) performs well when combined with the considered transformation methods, as this relationship was not explored before. Further, good performance of XGB would indicate explanations are sensible, as if the model performed badly, explanations can be less reliable. We use the SHAP algorithm as follows. For a given data set, we train the XGB model on *all* instances. We compute the *Shapley matrix*, which contains feature relevance information. Feature importances are finally visualized for selected (correctly classified) examples.

5 Results

We present the results obtained using the proposed benchmark. We begin by showing bar diagrams where differences between classifiers are shown with respect to a given propositionalization algorithm. Next, we present the critical distance diagrams showing classifier-propositionalization algorithm performance. Finally, we show how XGB's predictions can be interpreted using SHAP.

5.1 Results - classification

We observe the best overall classifier performances were obtained when RSD was used as the propositionalization method. As can be observed in the following results corresponding to individual classifier performances, the Aleph propositionalization performed the worst on average. The average rank diagram showing the performance of individual classifier-propositionalization pairs (or just propositionalization) is shown in Figures 3, 4 and 5. Note that we omit computation of critical distances due to reasons stated in [1].

The classification results further indicate that propositionalization represents the key aspect of the overall classification performance. This observation would

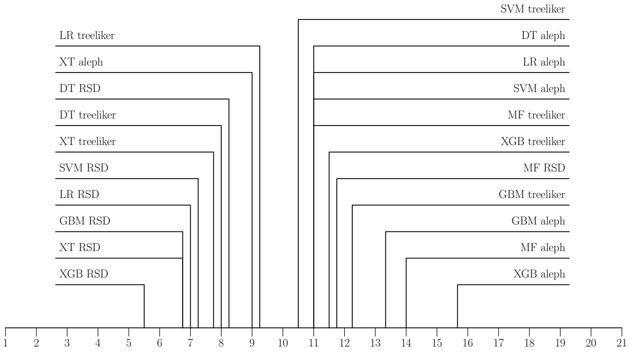

Fig. 3. Propositionalization + learner Accuracy comparison across data sets. Average ranks are shown.

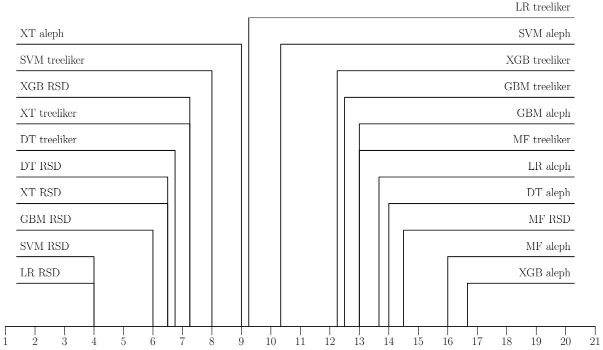

Fig. 4. Propositionalization + learner AUC comparison across data sets. Average ranks are shown.

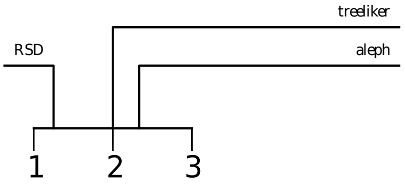

Fig. 5. Comparison of propositionalization approaches (classifier accuracy was averaged).

imply that no matter how well the classifier can learn if the input space is not representative of the modelled problem, the classifier will not learn well, which is indeed a sensible assumption in the relational setting, as the considered propositionalization algorithms consider different aspects of the relational database, and do not output equal representations.

5.2 Results - explanation example

In this section, we look at an example where we analysed feature importances via SHAP. The computed importance-based diagram is shown in Figure 6.

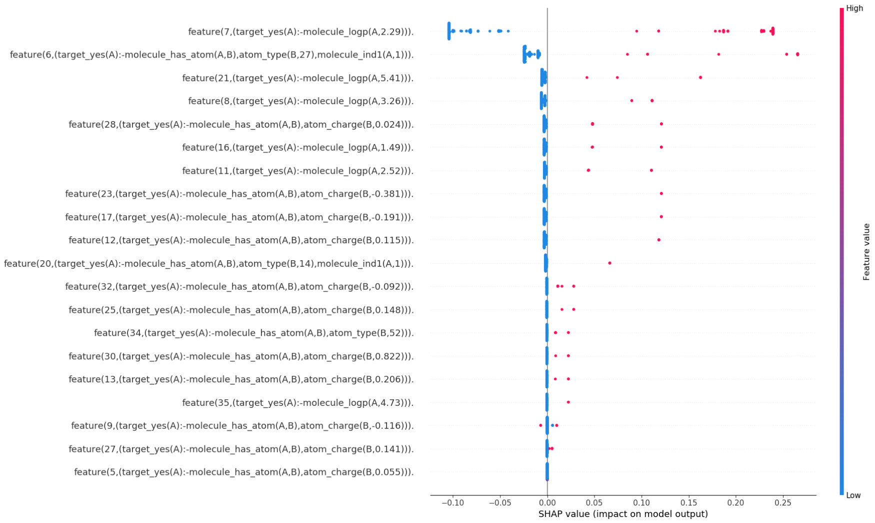

Fig. 6. Feature importances using Shapley values (Mutagenesis). Here, an XGB classifier was trained on the feature space, constructed with RSD.

Conjuncts of atom types and charges emerged as relevant for the classification performance for the mutagenesis (42) data set (Figure 6), indicating that the two attributes are highly relevant for correctly predicting the mutagenicity. Note, however, that the propositionalization is not able to perform discretization, meaning that the explanations are potentially too specific for non-categorical/ordinal variables. Further, it can be observed that there are at least three relational features that impact the model's prediction the most, indicating that diverse relational features were used by the XGBoost learner.

6 Discussion and conclusions

In this section, we discuss the obtained results and present some of the relevant future prospects regarding boosting in a relational setting.

Overall, we tested a set of different learners trained on feature matrices obtained by using three different propositionalization algorithms. We show that boosting based learners outperform many other classifiers. Further, we believe the conducted experiments show a direction for future research, where scalability could be an issue (boosting is very scalable). One of the results of this study also shows that the RSD algorithm is one of the top-performing propositionalization algorithms. Although it performs well, we believe that RSD and other propositionalization algorithms do not scale well, which is an issue that can be addressed using ideas proposed in [18, 19], which will be addressed in future work. To the best of our knowledge, this is one of the first works which utilize SHAP to explain relational features. The obtained results indicate that Shapley values are as such one of the possible ways to explain how the black box learners learn in a relational setting. We have yet to explore the different discretization schemes, which would yield more relevant explanations. We believe the presented results demonstrate the relevance of such approaches and are to our knowledge the first of such kind.

7 Availability

The code is freely available as part of the PyRDM library[7].

References

1. Benavoli, A., Corani, G., Demšar, J., Zaffalon, M.: Time for a change: a tutorial for comparing multiple classifiers through bayesian analysis. The Journal of Machine Learning Research **18**(1), 2653–2688 (2017)
2. Chang, C.C., Lin, C.J.: Libsvm: a library for support vector machines. ACM transactions on intelligent systems and technology (TIST) **2**(3), 27 (2011)
3. Chen, T., He, T., Benesty, M., Khotilovich, V., Tang, Y.: Xgboost: extreme gradient boosting. R package version 0.4-2 pp. 1–4 (2015)
4. Debnath, A.K., Lopez de Compadre, R.L., Debnath, G., Shusterman, A.J., Hansch, C.: Structure-activity relationship of mutagenic aromatic and heteroaromatic nitro compounds. correlation with molecular orbital energies and hydrophobicity. Journal of Medicinal Chemistry **34**(2), 786–797 (1991)
5. Demšar, J., Curk, T., Erjavec, A., Gorup, Č., Hočevar, T., Milutinovič, M., Možina, M., Polajnar, M., Toplak, M., Starič, A., et al.: Orange: data mining toolbox in python. The Journal of Machine Learning Research **14**(1), 2349–2353 (2013)
6. Flach, P., Lachiche, N.: 1BC: A first-order Bayesian classifier. In: Proceedings of the 9th International Workshop on Inductive Logic Programming (ILP-99). pp. 92–103. Springer (1999)
7. Flach, P.A., Lachiche, N.: Confirmation-guided discovery of first-order rules with Tertius. Machine Learning **42**(1/2), 61–95 (2001)
8. Goodfellow, I., Bengio, Y., Courville, A.: Deep Learning. MIT Press (2016), `http://www.deeplearningbook.org`

[7] `https://github.com/xflows/rdm`

9. Kearns, M., Valiant, L.: Cryptographic limitations on learning boolean formulae and finite automata. Journal of the ACM (JACM) **41**(1), 67–95 (1994)
10. Leskovec, J., Mcauley, J.J.: Learning to discover social circles in ego networks. In: Advances in neural information processing systems. pp. 539–547 (2012)
11. Lundberg, S.M., Lee, S.I.: A unified approach to interpreting model predictions. In: Guyon, I., Luxburg, U.V., Bengio, S., Wallach, H., Fergus, R., Vishwanathan, S., Garnett, R. (eds.) Advances in Neural Information Processing Systems 30, pp. 4765–4774. Curran Associates, Inc. (2017), http://papers.nips.cc/paper/7062-a-unified-approach-to-interpreting-model-predictions.pdf
12. Mason, L., Baxter, J., Bartlett, P.L., Frean, M.R.: Boosting algorithms as gradient descent. In: Advances in neural information processing systems. pp. 512–518 (2000)
13. McKinney, W.: pandas: a foundational python library for data analysis and statistics. Python for High Performance and Scientific Computing **14** (2011)
14. Michalski, R.S.: A Theory and Methodology of Inductive Learning. In: Michalski, R.S., Carbonell, J.G., Mitchell, T.M. (eds.) Machine Learning: An artificial intelligence approach, pp. 83–129. Palo Alto: Tioga Publishing Company (1983)
15. Motl, J., Schulte, O.: The ctu prague relational learning repository (2015)
16. Muggleton, S.: Inverse entailment and Progol. New Generation Computing, Special issue on Inductive Logic Programming **13**(3-4), 245–286 (1995)
17. Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., Blondel, M., Prettenhofer, P., Weiss, R., Dubourg, V., et al.: Scikit-learn: Machine learning in python. Journal of machine learning research **12**(Oct), 2825–2830 (2011)
18. Perovšek, M., Vavpetič, A., Cestnik, B., Lavrač, N.: A wordification approach to relational data mining. In: Fürnkranz, J., Hüllermeier, E., Higuchi, T. (eds.) Proceedings of the 16th International Conference on Discovery Science, Singapore, October 6–9, 2013. Lecture Notes in Computer Science, vol. 8140, pp. 141–154. Springer (2013)
19. Perovšek, M., Vavpetič, A., Kranjc, J., Cestnik, B., Lavrač, N.: Wordification: Propositionalization by unfolding relational data into bags of words. Expert Syst. Appl. **42**(17-18), 6442–6456 (2015)
20. Srinivasan, A.: The Aleph manual. http://web.comlab.ox.ac.uk/oucl/research/areas/machlearn/Aleph/ (1999)
21. Srinivasan, A., King, R.D., Muggleton, S., Sternberg, M.J.: Carcinogenesis predictions using ILP. In: Lavrač, N., Džeroski, S. (eds.) ILP-97. Proceedings of the 7th International Workshop on Inductive Logic Programming, Prague, Czech Republic, September 17-20, 1997, Lecture Notes in Computer Science, vol. 1297, pp. 273–287. Springer (1997)
22. Vavpetič, A.: Semantic Subgroup Discovery: Doctoral Dissertation. Ph.D. thesis, A. Vavpetič (2016)
23. Železný, F., Lavrač, N.: Propositionalization-based relational subgroup discovery with RSD. Machine Learning **62**(1-2), 33–63 (2006)