Global Optimization in Learning with Important Data: an FCA-Based Approach

Yury Kashnitsky, Sergei O. Kuznetsov

National Research University Higher School of Economics, Moscow, Russia
{ykashnitsky, skuznetsov}@hse.ru

Abstract. Nowadays decision tree learning is one of the most popular classification and regression techniques. Though decision trees are not accurate on their own, they make very good base learners for advanced tree-based methods such as random forests and gradient boosted trees. However, applying ensembles of trees deteriorates interpretability of the final model. Another problem is that decision tree learning can be seen as a greedy search for a good classification hypothesis in terms of some information-based criterion such as Gini impurity or information gain. But in case of small data sets the global search might be possible.
In this paper, we propose an FCA-based lazy classification technique where each test instance is classified with a set of the best (in terms of some information-based criterion) rules. In a set of benchmarking experiments, the proposed strategy is compared with decision tree and nearest neighbor learning.

Keywords: Formal Concept Analysis, lazy learning, global optimization

1 Introduction

The classification task in machine learning aims to use some historical data (a training set) to predict unknown discrete variables in unknown data (a test set). While there are dozens of popular methods for solving the classification problem, usually there is an accuracy-interpretability trade-off when choosing a method for a particular task. Neural networks, random forests and ensemble techniques (boosting, bagging, stacking etc.) are known to outperform simple methods in difficult tasks. Kaggle competitions also bear testimony for that – usually, winners resort to ensemble techniques, mainly to gradient boosting [1]. The mentioned algorithms are widely spread in those application scenarios where classification performance is the main objective. In Optical Character Recognition, voice recognition, information retrieval and many other tasks typically we are satisfied with a trained model if it has a low generalization error.

However, in lots of applications we need a model to be interpretable as well as accurate. Some classification rules, built from data and examined by experts, may be justified or proved. In medical diagnostics, when making highly responsible decisions, e.g., predicting whether a patient has cancer (i.e., dealing with "important data"), experts prefer to extract readable rules from a machine learning model in order to "understand" it and justify the decision. In credit scoring,

for instance, applying ensemble techniques can be very effective, but the model is often obliged to have "sound business logic", that is, to be interpretable [2].

2 Related work

Eager (non-lazy) algorithms construct classifiers that contain an explicit hypothesis mapping unlabelled test instances to their predicted labels. A decision tree classifier, for example, uses a stored model to classify instances by tracing the instance through the tests at the interior nodes until a leaf containing the label is reached. In eager algorithms, the main work is done at the phase of building a classifier.

In lazy classification paradigm [3], however, no explicit model is constructed, and the inductive process is done by a classifier which maps each test instance to a label using a training set.

2.1 Lazy decision trees

The authors of [4] point the following problem with decision tree learning: while entropy measures used in C4.5 and ID3 are guaranteed to decrease on average, the entropy of a specific child may not change or may increase. In other words, a single decision tree may find a locally optimal hypothesis in terms of entropy measure such as Gini impurity or pairwise mutual information. But using a single tree may lead to many irrelevant splits for a given test instance. A decision tree built for each test instance individually can avoid splits on attributes that are irrelevant for the specific instance. Thus, such "customized" decision trees (actually classification paths) built for a specific test instance may be much shorter and hence may provide a short explanation for the classification.

2.2 Lazy associative classification

Associative classifiers build a classifier using association rules mined from training data. Such rules have the class attribute as a conclusion. This approach was shown to yield improved accuracy over decision trees as they perform a global search for rules satisfying some quality constraints [5]. Decision trees, on the contrary, perform greedy search for rules by selecting the most promising attributes.

Unfortunately, associative classifiers tend to output too many rules while many of them even might not be used for classification of a test instance. Lazy associative classification algorithm overcomes these problems of associative classifiers by generating only the rules with premises being subsets of test instance attributes [5]. Thus, in lazy associative classification paradigm only those rules are generated that might be used in classification of a test instance. This leads to a reduced set of classification rules for each test instance.

2.3 Decision trees in terms of Formal Concept Analysis

In [6] the authors utilize concept lattices to represent each concept intent (a closed set of attributes) as a decision tree node and a concept lattice itself – as a set of overlapping decision trees. The construction of a decision tree is thus reduced to selecting one of the downward paths in a concept lattice via some information criterion.

2.4 Lazy classification for complex structure data

The modification of the lazy classification algorithm capable of handling complex structure data was first proposed in [7]. The main difference from the Lazy Associative Classification algorithm is that the method is designed to analyze arbitrary objects with complex descriptions (intervals, sequences, graphs etc.). This setting was implemented for interval credit scoring data [8] and for graphs in a toxicology prediction task [9].

3 Definitions

Here we introduce some notions from Formal Concept Analysis [10] which help us to organize the search space for classification hypotheses.

Definition 1. *A formal context in FCA is a triple $K = (G, M, I)$ where G is a set of objects, M is a set of attributes, and the binary relation $I \subseteq G \times M$ shows which object possesses which attribute. gIm denotes that object g has attribute m. For subsets of objects and attributes $A \subseteq G$ and $B \subseteq M$ Galois operators are defined as follows:*

$$A' = \{m \in M \mid gIm \ \forall g \in A\},$$
$$B' = \{g \in G \mid gIm \ \forall m \in B\}.$$

A pair (A, B) such that $A \subseteq G, B \subseteq M, A' = B$ and $B' = A$, is called a formal concept of a context K. The sets A and B are closed and called the extent and the intent of a formal concept (A, B) respectively.

Example 1. Let us consider a "classical" toy example of a classification task. The training set is represented in Table 1. All categorical attributes are binarized into "dummy" attributes. The table shows a formal context $K = (G, M, I)$ with $G = \{1, \ldots, 10\}$, $M = \{or, oo, os, tc, tm, th, hn, w\}$ (let us omit a class attribute "play") and I – a binary relation defined on $G \times M$ where an element of a relation is represented with a cross ($\times$) in a corresponding cell of a table.

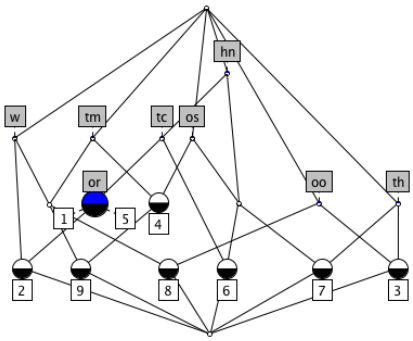
A concept lattice for this formal context is depicted to the right from1. It should be read as follows: for a given element (formal concept) of the lattice its intent (closed set of attributes) is given by all attributes which labels can be reached in ascending lattice traversal. Similarly, the extent (a closed set of objects) of a certain lattice element (formal concept) can be traced in a downward

lattice traversal from a given point. For instance, a big blue-and-black circle depicts a formal concept $(\{1, 2, 5\}, \{or, tc, hn\})$.

Such concept lattice is a concise way of representing all closed itemsets (formal concepts' intents) of a formal context. Closed itemsets, further, can serve as a condensed representation of classification rules [11]. In what follows, we develop the idea of a hypotheses search space represented with a concept lattice.

Table 1. A toy classification problem and a concept lattice of the corresponding formal context. Attributes: *or* – outlook = rainy, *oo* – outlook = overcast, *os* – outlook = sunny, *tc* – temperature = cold, *tm* – temperature = mild, *th* – temperature = high, *hn* – humidity = normal, *w* – windy, *play* – whether to play tennis or not (class attribute).

№	or	oo	os	tc	tm	th	hn	w	play
1	×			×			×		×
2	×			×			×	×	
3		×				×			×
4			×		×				
5	×			×			×		×
6			×	×			×		×
7			×			×	×		×
8		×			×			×	×
9			×		×			×	
10			×	×					?



4 Concept lattice a hypothesis search space

Further we describe and illustrate the proposed approach in binary- and numeric-attribute cases when dealing with binary classification. The approach is naturally extended to multiclass case with the corresponding adjustments to information criteria formulas.

4.1 Binary-attribute case

In case of training and test data represented as binary tables, the proposed algorithm is described as Algorithm 1.

Let $K_{train} = (G_{train}, M_0 \cup \overline{M}_0 \cup c_{train}, I_{train})$ and $K_{test} = (G_{train}, M_0 \cup \overline{M}_0, I_{test})$ be formal contexts representing a training set and a test set correspondingly. We state clearly that the set of attributes is dichotomized:
$M = M_0 \cup \overline{M}_0$ where $\forall g \in G_{train}, m \in M_0 \; \exists \; \overline{m} \in \overline{M}_0 : gI_{train}m \rightarrow \neg gI_{train}\overline{m}$.
Let $CbO(K, min\_supp)$ be the algorithm used to find all formal concepts of a

formal context K with support greater or equal to min_supp (by default we use a modification of the InClose-2 program implementation [12] of the CloseByOne algorithm [13]). Let $inf : M \cup c_{train} \to \mathbb{R}$ be an information criterion used to rate classification rules (we use Gini impurity by default). Finally, let min_supp and n_rules be the parameters of the algorithm (the minimal support of each classification rule's premise and the number of rules to be used for prediction of each test instance's class attribute).

With these designations, the main steps of the proposed algorithm for each test instance are the following:

1. For each test object we leave only its attributes in the training set (step 1 in Algorithm 1). Or, formally, we build a new formal context $K_t = \{G_{train}, g_t', I_{train}\}$ with the same objects G_{train} as in the training context K_{train} and with attributes of a test object $g_t' \cup c_{train}$. We clarify what it means in case of real-valued attributes in subsection 4.2.
2. With $CbO(K, min\_supp)$, find all formal concepts of a formal context K_t satisfying the constraint on minimal support. We build formal concepts in a top-down manner (increasing the number of attributes) and backtrack when the support of a formal concept intent is less than min_supp. The parameter min_supp refines the support of any possible hypothesis mined to classify the test object and is therefore analogous to the parameter $min\_samples\_leaf$ of a decision tree. While generating formal concepts, we keep track of the values of the class attributes for all training objects having all corresponding attributes (i.e. for all objects in formal concept extent). We calculate the value of an information criterion inf (we use Gini impurity by default) for each formal concept intent.
3. Then the mined formal concepts are sorted by the value of the criterion inf from the "best" to the "worse".
4. Retaining first n concepts with the best values of the chosen information criterion, we have a set of rules to classify the current test object. For each concept we define a classification rule with concept intent as an antecedent and the most common value of class attribute among the objects of concept extent as a consequent.
5. Finally, we predict the value of the class attribute for current test object simply via majority rule among n "best" classification rules' antecedents. We also save the rules for each test object in a dictionary r_{test}.

4.2 Numeric-attribute case

In our approach, we deal with numeric attributes similarly to what is done in the CART algorithm [14]. We sort the values of a numeric attribute and identify the thresholds to binarize numeric attributes where the target attribute changes. Let us demonstrate step 1 of Algorithm 1 in case of binary and numeric attributes with a sample from Kaggle "Titanic: Machine Learning from Disaster" competition dataset.[1]

[1] https://www.kaggle.com/c/titanic

Algorithm 1 Lazy Lattice-based Optimization (LLO)

Input: $K_{train} = (G_{train}, M_0 \cup \overline{M}_0 \cup c_{train}, I_{train})$
$K_{test} = (G_{test}, M_0 \cup \overline{M}_0, I_{test})$
$min\_supp \in \mathbb{R}^+,\ n_{rules} \in \mathbb{N}$;
$CbO(K, min\_supp) : K \to \mathcal{S}$;
$sort(\mathcal{S},\ inf) : \mathcal{S} \to \mathcal{S}$
$inf : M \cup c_{train} \to \mathbb{R}$;
Output: c_{test}, r_{test}

$c_{test} = \emptyset, r_{test} = \emptyset$
for $g_t \in G_{test}$ **do**
1. $K_t = \{G_{train}, g_t', I_{train}\}$
2. $\mathcal{S}_t = \{(A, B) \mid A \subseteq G_{train}, B \subseteq g_t', A' = B, B' = A, \frac{|A|}{G_{train}} \geq min\_supp\} = CbO(K_t, min\_supp)$
3. $\mathcal{S}_t = sort(\mathcal{S}_t,\ inf)$
4. $\{B_i\}_{i \in [1, n_{rules}]} = \{B_j \mid (A_j, B_j) \in \mathcal{S}_t\}, j \in [1, n\_rules]$
5. $c_i = argmax(\{count(c_{train_j}) \mid j \in B_i'\})$
6. $r_{test}[i] = \{B_i \to c_i\}, i = 1, \ldots, n_{rules}$
7. $c_{test}[i] = argmax(\{count(c_j) \mid j = 1, \ldots, n_{rules})\}$

end for

Example 2. Table 2 shows a sample from the Titanic dataset. Let us build a formal context to classify passenger no. 7 with attributes *Pclass=2, Age=28, City=C*. If we sort the data by age in ascending order we see where the target attribute "Survived" switches from 0 to 1 or vice versa.

Age	16	18	30	39	42	62
Survived	1	0	0	1	0	1

Thus we have a set of thresholds to discretize the attribute "Age": $T = \{17, 34.5, 40.5, 52\}$. The formal context K_7 (corresponding to K_t for $t = 7$ in Algorithm 1) is presented in Table 3.

4.3 Complexity

The algorithm is based on the CloseByOne lattice-building algorithm with time complexity shown [15] to be equal to $O(|G||M|^2|\mathcal{L}|)$ for a formal context (G, M, I) and a corresponding lattice $\mathcal{L}$. To put it simply, the complexity is linear in the number of objects, quadratic in the number of attributes and linear in the number of built formal concepts.

In the proposed algorithm CloseByOne is run for each test object (step 3 in Algorithm 1), and for each formal concept information criterion values are calculated. Calculating entropy or Gini index is linear in the number of objects as it requires calculating supports of attribute sets. This is done "on-the-go" while building a lattice (step 4 in Algorithm 1).

Table 2. A sample from the Titanic dataset. Attributes: "Pclass" – passenger's class, "City" – boarding city (here Cherburg or Southhampton), "Age" – passenger's age, "Survived" – whether a passenger survived in the Titanic disaster.

Id	**Pclass**	**Age**	**City**	**Survived**
1	3	39	S	1
2	3	16	S	1
3	1	62	C	1
4	3	42	S	0
5	2	30	C	0
6	2	18	C	0
7	2	28	C	?
8	1	47	C	?

Table 3. A formal context built to classify a test passenger no. 7.

Id	$Pclass! = 1$	$Pclass == 2$	$Pclass! = 3$	$Age \geq 17$	$Age \leq 34.5$	$Age \leq 40.5$	$Age \leq 52$	$City == C$	**Survived**
1	×			×		×	×		1
2	×				×	×	×		1
3			×	×				×	1
4	×			×			×		0
5	×	×	×	×	×	×	×	×	0
6	×	×	×	×	×	×	×	×	0

Therefore, the time complexity of classifying $|G_t|$ test instances with the proposed algorithm based on a training formal context (G, M, I) is approximately $O(|G_t||G||M|^2|\bar{\mathcal{L}}|)$ where $|\bar{\mathcal{L}}|$ is an average lattice size for formal contexts described in step 2 in Algorithm 1.

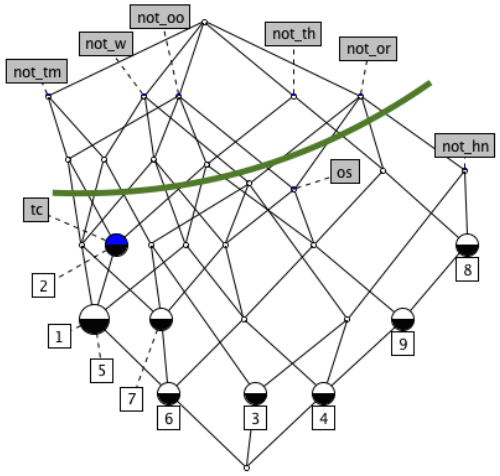
5 Example

Let us illustrate the proposed algorithm with a toy example from Table 1. To classify the object no. 10, we do the following steps according to Algorithm 1:

1. Let us fix Gini impurity as an information criterion of interest and the parameters $min\_supp = 0.5$ and $n = 3$. Thus, we are going to classify a test instance with 3 rules supporting at least 5 objects and having highest gain in Gini impurity.
2. The case *Outlook=sunny, Temperature=cool, Humidity=high, Windy=false* corresponds to a set of attributes $\{os,\ tc,\ hh, w\}$ describing the test instance. Or, if we consider the negations of the attributes, such case is described with a set of attributes: $\{\bar{or}, \bar{oo}, os, tc, \bar{tm}, \bar{th}, \bar{hn}, \bar{w}\}$
3. We build a formal context with objects being the training set instances and attributes of a test instance – $\{\bar{or}, \bar{oo}, os, tc, \bar{tm}, \bar{th}, \bar{hn}, \bar{w}\}$. The corresponding binary table is shown in Table 4.

Table 4. The training set instances with attributes of a test instance *Outlook=sunny, Temperature=cool, Humidity=high, Windy=false*. Attributes: $\bar{or}$ – outlook is not rainy, $\bar{oo}$ – outlook is not overcast, *os* – outlook = sunny, *tc* – temperature = cool, $t\bar{m}$ – temperature is not mild, $\bar{th}$ – temperature is not high, $\bar{hn}$ – humidity is not normal, $\bar{w}$ – not windy, *play* – whether to play tennis or not (class attribute). A concept lattice on the right-hand side is build with the corresponding formal context. The horizontal line separates the concepts with extents comprised of at least 5 objects.

№	$\bar{or}$	$\bar{oo}$	**os**	**tc**	$t\bar{m}$	$\bar{th}$	$\bar{hn}$	$\bar{w}$	**play**
1		×		×	×	×		×	×
2		×		×	×	×			
3	×				×		×	×	×
4	×	×	×			×	×	×	
5		×		×	×	×		×	×
6	×	×	×	×	×	×		×	×
7	×	×	×		×			×	×
8	×					×	×		×
9	×	×	×			×	×		



4. A concept lattice, organizing all formal concepts for a formal context is shown to the right from Table 4. The horizontal line separates the concepts with extents having at least 5 objects (above, $min\_supp \geq 0.5$).
5. 9 formal concepts satisfying $min\_supp \geq 0.5$ give rise to 9 classification rules. Top 3 rules having the highest gain in Gini impurity are given in Table 5.

Table 5. Top 3 rules to classify the test instance *Outlook=sunny, Temperature=cool, Humidity=high, Windy=false*

Rule	**Gini gain**
$\{not\ windy, temperature\ not\ mild\} \rightarrow play$	0.278
$\{outlook\ not\ overcast, temperature\ not\ high\} \rightarrow play$	0.111
$\{outlook\ not\ overcast, temperature\ not\ mild\} \rightarrow play$	0.044

6. The "best" rules mined in the previous step unanimously classify the test instance *Outlook=sunny, Temperature=cool, Humidity=high, Windy=false* as appropriate for playing tennis.

6 Experiments

As we have stated, in this paper we deal with "important data" problems, those where accurate and interpretable results are needed. We compare the proposed classification algorithm (denoted as LLO for "Lazy Lattice-based Optimization") with Scikit-learn [16] implementations of CART [14] and kNN on several datasets from the UCI machine learning repository.[2]

We used pairwise mutual information as a criterion for rule selection. CART and kNN parameters were chosen in stratified 5-fold cross-validation and are given in Table 7.

Parameter min_supp for LLO was taken equal to CART min_sample_leaf for each dataset divided by the number of objects. We used $n = 5$ classification rules to vote for a test instance label.

As it can be seen, the proposed approach performs better than CART on most of the datasets while kNN is often better when the number of attributes is not high. Obviously, the running times of LLO are far from perfect. That is due to the computationally demanding nature of the algorithm.

Conclusions and further work

In this paper, we have shown how searching for classification hypotheses in a formal concept lattice for each test instance individually may yield accurate

[2] http://repository.seasr.org/Datasets/UCI/csv/

Table 6. Accuracy and F1-score in classification experiments with the UCI machine learning datasets. "CART acc" stands for "5-fold cross-validation accuracy of the CART algorithm", ... , "LLO F1" stands for "5-fold cross-validation F1 score of the Lazy Lattice Optimization algorithm".

dataset	CART acc	kNN acc	LLO acc	CART F1	kNN F1	LLO F1
audiology	0.743	0.442	**0.758**	0.725	0.336	**0.736**
breast-cancer	0.738	0.727	**0.769**	0.477	0.66	**0.694**
breast-w	0.936	0.773	**0.942**	0.909	0.734	**0.921**
colic	0.647	0.644	**0.653**	0.619	0.569	**0.664**
heart-h	0.782	**0.837**	0.791	0.664	**0.831**	0.787
heart-statlog	0.804	**0.848**	0.816	0.761	**0.846**	0.823
hepatitis	**0.794**	**0.794**	0.782	**0.867**	0.702	0.755
hypothyroid	**0.975**	0.923	0.968	**0.974**	0.886	0.948
ionosphere	0.9	0.783	**0.924**	0.923	0.757	**0.938**
kr-vs-kp	**0.98**	0.761	**0.98**	0.981	0.756	**0.984**
letter	0.769	0.711	**0.774**	0.769	0.645	**0.771**
lymph	0.818	**0.831**	0.82	0.806	0.813	**0.85**
primary-tumor	0.425	**0.469**	0.457	0.376	**0.418**	0.409
segment	0.938	0.872	**0.947**	**0.938**	0.869	0.928
sonar	0.697	0.663	**0.73**	0.665	0.658	**0.718**
soybean	0.877	**0.89**	0.88	0.868	0.883	**0.879**
splice	0.943	0.833	**0.956**	0.943	0.832	**0.948**
vehicle	**0.708**	0.677	0.692	**0.708**	0.667	0.62
vote	0.956	0.929	**0.968**	0.946	0.929	**0.955**
vowel	0.436	0.405	**0.442**	**0.428**	0.387	0.406
waveform-5000	0.761	**0.834**	0.783	0.761	0.583	**0.774**

Table 7. Parameters and runtimes in classification experiments with the UCI machine learning datasets. "CART msl" stands for for the minimal required number of objects in each node of a CART tree ($min\_samples\_leaf$), "kNN k" is the number of neighbors used by the kNN algorithm.

dataset	**# objects**	**# attr**	**CART msl**	**kNN k**	**CART time**	**kNN time**	**LLO time**
audiology	226	94	1	10	0.31	0.52	9.97
breast-cancer	286	39	4	20	0.29	0.52	5.35
breast-w	699	89	6	10	0.29	0.83	258.37
colic	368	59	1	30	0.3	0.52	6.41
heart-h	294	24	2	20	0.3	0.52	0.89
heart-statlog	270	13	5	45	0.3	0.53	3.76
hepatitis	155	285	2	10	0.29	0.55	62.9
hypothyroid	3772	126	7	15	0.63	1.39	298.84
ionosphere	351	34	4	10	0.41	0.54	2.03
kr-vs-kp	3196	38	1	50	0.4	2.03	23.15
letter	20000	256	1	71	38.04	67.85	6607.2
lymph	148	50	1	10	0.29	0.52	4.7
primary-tumor	339	26	4	15	0.3	0.52	1.59
segment	2310	19	1	10	1.05	0.83	4.17
sonar	208	60	3	15	0.41	0.53	3.79
soybean	683	98	1	10	0.3	0.73	32.6
splice	3190	287	6	65	1.3	11.29	1302.57
vehicle	846	18	4	10	0.62	0.63	1.34
vote	435	32	2	10	0.31	0.53	2.65
vowel	990	26	2	35	0.63	0.63	3.29
waveform-5000	5000	40	5	82	3.79	1.34	40.2

results while keeping the classification model interpretable. The proposed strategy is computationally demanding but may be used for "small data" problems where prediction delay is not as important as classification accuracy and interpretability.

Further we plan to implement the idea of searching for classification hypotheses in a concept lattice for complex structure data such as molecular graphs We plan to implement the same strategy of lazy classification by searching for succinct classification rules in a pattern concept lattice. The designed framework might help to learn sets of rules for tasks such as biological activity (toxicology, mutagenicity, etc.) prediction. We are also going to interpret random forests as a search for an optimal hypothesis in a concept lattice and try to compete with this popular classification method.

References

1. Tsoumakas, G., Papadopoulos, A., Qian, W., Vologiannidis, S., D'yakonov, A., Puurula, A., Read, J., Svec, J., Semenov, S.: Wise 2014 challenge: Multi-label classification of print media articles to topics. In: 15th International Conference on Web Information Systems Engineering (WISE 2014). Proceedings Part II. Volume 8787 of Lecture Notes in Computer Science., Springer (October 12-14 2014) 541–548
2. Li, X., Zhong, Y.: An overview of personal credit scoring: Techniques and future work. International Journal of Intelligence Science **2**(4A) (2012) 181–189
3. Aha, D.W., ed.: Lazy Learning. Kluwer Academic Publishers, Norwell, MA, USA (1997)
4. Friedman, J.H.: Lazy decision trees. In: Proceedings of the Thirteenth National Conference on Artificial Intelligence - Volume 1. AAAI'96, AAAI Press (1996) 717–724
5. Veloso, A., Meira Jr., W., Zaki, M.J.: Lazy Associative Classification. In: Proceedings of the Sixth International Conference on Data Mining. ICDM '06, Washington, DC, USA, IEEE Computer Society (2006) 645–654
6. Belohlavek, R., De Baets, B., Outrata, J., Vychodil, V.: Inducing decision trees via concept lattices. International Journal of General Systems **38**(4) (2009) 455–467
7. Kuznetsov, S.O.: Scalable knowledge discovery in complex data with pattern structures. In Maji, P., Ghosh, A., Murty, M.N., Ghosh, K., Pal, S.K., eds.: PReMI. Volume 8251 of Lecture Notes in Computer Science., Springer (2013) 30–39
8. Masyutin, A., Kashnitsky, Y., Kuznetsov, S.O.: Lazy classification with interval pattern structures: Application to credit scoring. In: CEUR Workshop Proceedings. Volume 1430. (2015) 43–54
9. Kashnitsky, Y., Kuznetsov, S.O.: Lazy associative graph classification. In: CEUR Workshop Proceedings. Volume 1430. (2015) 63–74
10. Ganter, B., Wille, R.: Formal Concept Analysis: Mathematical Foundations. 1st edn. Springer-Verlag New York, Inc., Secaucus, NJ, USA (1997)
11. Hata, I., Veloso, A., Ziviani, N.: Learning accurate and interpretable classifiers using optimal multi-criteria rules. JIDM **4**(3) (2013) 204–219
12. Andrews, S.: In-close2, a high performance formal concept miner. In: Conceptual Structures for Discovering Knowledge - 19th International Conference on Conceptual Structures, ICCS 2011, Derby, UK. Proceedings. (2011) 50–62

13. Kuznetsov, S.O.: A fast algorithm for computing all intersections of objects from an arbitrary semilattice. Nauchno-Tekhnicheskaya Informatsiya Seriya 2 – Informatsionnye protsessy i sistemy (1) (1993) 17–20
14. Breiman, L., Friedman, J., Stone, C., Olshen, R.: Classification and Regression Trees. The Wadsworth and Brooks-Cole statistics-probability series. Taylor & Francis (1984)
15. Kuznetsov, S.O., Obiedkov, S.A.: Comparing performance of algorithms for generating concept lattices. Journal of Experimental & Theoretical Artificial Intelligence **14**(2-3) (2002) 189–216
16. Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., Blondel, M., Prettenhofer, P., Weiss, R., Dubourg, V., Vanderplas, J., Passos, A., Cournapeau, D., Brucher, M., Perrot, M., Duchesnay, E.: Scikit-learn: Machine Learning in Python. Journal of Machine Learning Research **12** (2011) 2825–2830