Lazy Learning of Classification Rules for Complex Structure Data

Yury Kashnitsky

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

Abstract. In this paper, we address machine learning classification problem and classify each test instance with a set of interpretable and accurate rules. We resort to the idea of lazy classification and mathematical apparatus of formal concept analysis to develop an abstract framework for this task. In a set of benchmarking experiments, we compare the proposed strategy with decision tree learning. We discuss the generalization of the proposed framework for the case of complex structure data such as molecular graphs in tasks such as prediction of biological activity of chemical compounds.

Keywords: formal concept analysis, lazy classification, complex structure

1 Introduction

The classification task in machine learning aims to use some historical data (training set) to predict unknown discrete variables in unknown data (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, the 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 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), 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].

Another point of interest in this paper is dealing with complex structure data in classification tasks. While there are various popular techniques for handling time series, sequences, graph data, we discuss how pattern structures as formal data representation and lazy associative classification as a learning paradigm may help to learn succinct classification rules for tasks with complex structure data.

2 Definitions

Here we introduce some notions from Formal Concept Analysis [3] 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 $A \subset G, B \subset M, A' = B$ 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 from [4]. 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 in Fig. 1. 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 [5]. In what follows, we develop the idea of a hypotheses search space represented with a concept lattice.

2.1 Pattern Structures

Pattern structures are natural extension of Formal Concept Analysis to objects with arbitrary partially-ordered descriptions [6]. The order on a set of descriptions D allows one to define a semilattice $(D, \sqcap)$, i.e. for any $d_i, d_j, d_k \in D$:

Table 1. A toy classification problem. 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).

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

$d_i \sqcap d_i = d_i, d_i \sqcap d_j = d_j \sqcap d_i, d_i \sqcap (d_j \sqcap d_k) = (d_i \sqcap d_j) \sqcap d_k$. Please refer to [7] for details.

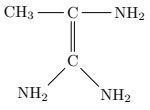
Definition 2. *Let G be a set (of objects), let $(D, \sqcap)$ be a meet-semi-lattice (of all possible object descriptions) and let $\delta : G \to D$ be a mapping between objects and descriptions. Set $\delta(G) := \{\delta(g) | g \in G\}$ generates a complete subsemilattice $(D_\delta, \sqcap)$ of $(D, \sqcap)$, if every subset X of $\delta(G)$ has infimum $\sqcap X$ in $(D, \sqcap)$.* ***Pattern structure*** *is a triple $(G, \underline{D}, \delta)$, where $\underline{D} = (D, \sqcap)$, provided that the set $\delta(G) := \{\delta(g) \mid g \in G\}$ generates a complete subsemilattice $(D_\delta, \sqcap)$ [6,8].*

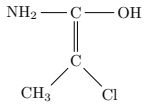
Definition 3. ***Patterns*** *are elements of D. Patterns are naturally ordered by subsumption relation $\sqsubseteq$: given $c,\ d \in D$ one has $c \sqsubseteq d \Leftrightarrow c \sqcap d = c$. Operation $\sqcap$ is also called a* ***similarity*** *operation. A pattern structure $(G, \underline{D}, \delta)$ gives rise to the following* ***derivation operators*** *$(\cdot)^\diamond$:*

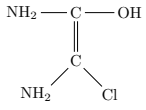
$$A^\diamond = \prod_{g \in A} \delta(g) \qquad \text{for } A \in G,$$
$$d^\diamond = \{g \in G \mid d \sqsubseteq \delta(g)\} \qquad \text{for } d \in (D, \sqcap).$$

Pairs (A, d) satisfying $A \subseteq G,\ d \in \underline{D},\ A^\diamond = d$, and $A = d^\diamond$ are called ***pattern concepts*** *of $(G, \underline{D}, \delta)$.*

Example 2. Closed sets of graphs can be presented with a pattern structure. Let $\{1, 2, 3\}$ be a set of objects, $\{G_1, G_2, G_3\}$ – be a set of their molecular graphs:

G_1 : 

G_2 : 

G_3 : 

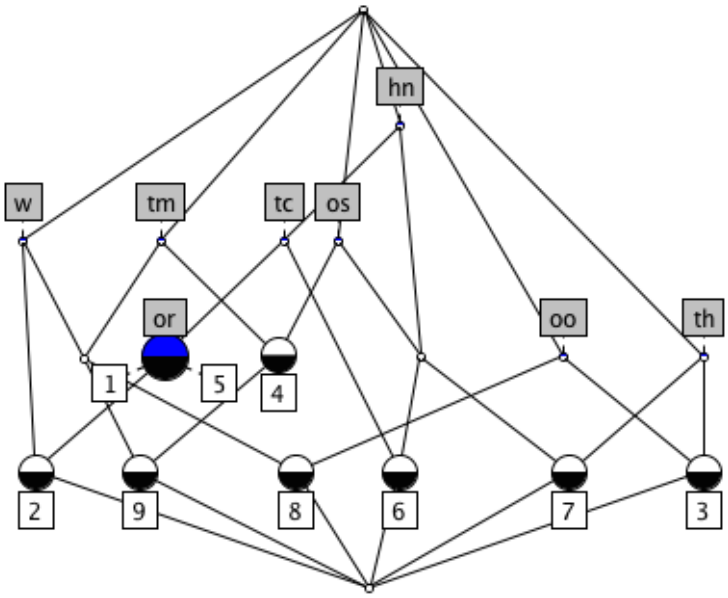


Fig. 1. A concept lattice for the formal context represented with Table 1.

A set of objects $\{1,2,3\}$, their molecular graphs $D = \{G_1, G_2, G_3\}$ ($\delta(i) = G_i, i = 1,\ldots,3$), and a similarity operator $\sqcap$ defined in [9] comprise a pattern structure $(\{1,2,3\}, (D,\sqcap), \delta)$.

Here is the set of all pattern concepts for this pattern structure:

$$\Big\{ \Big(\{1,2,3\}\,,\ \ce{NH2-C(=C)-C}\Big),\ \Big(\{1,2\}\,,\ \ce{CH3-C(=C)-C(-NH2)}\Big),\ \Big(\{1,3\}\,,\ \ce{NH2-C(=C)-C(-NH2)}\Big),$$
$$\Big(\{2,3\}\,,\ \ce{NH2-C(=C(-OH))-C(-Cl)}\Big),\ (1,\{G_1\}),\ (2,\{G_2\}),\ (3,\{G_3\}),\ (\emptyset,\{G_1,G_2,G_3\})\ \Big\}.$$

Please refer to [9] for clarification of this example.

Further, we show how pattern concept lattices help to organize the search space for classification hypotheses.

3 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 [10], 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.

The authors of [11] 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.

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 [12]. 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 [12]. 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.

In [7] and [8] the authors generalize the lazy associative classification framework to operate with complex data descriptions such as intervals, sequences, processes and graphs.

In [13] the authors use 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.

4 The search for classification hypotheses in a concept lattice

4.1 Binary-attribute case

For training and test data represented as binary tables, we propose Algorithm 1.

For each test instance we leave only its attributes in the training set (steps 1-2 in Algorithm 1). We clarify what it means in case of real-valued attributes in subsection 4.2.

Then we utilize a modification of the In-Close algorithm [14,15] to find all formal concepts of a formal context with attributes of a test instance (step 3 in Algorithm 1). We build formal concepts in a top-down manner (increasing the number of attributes) and backtrack when the cardinality of a formal concept intent exceeds k. The parameter k refines the length of any possible hypothesis mined to classify the test instance and is therefore analogous to the depth of a decision tree. We speed up computing closed attribute sets (formal concept intents) by storing them in a separate data structure (set $\mathcal{S}$ in the pseudocode).

While generating formal concepts, we retain the values of the class attributes for all training instances having all corresponding attributes (i.e. for all objects in formal concept extent). We calculate the value of some information criterion (such as Gini impurity, Gini ratio or pairwise mutual information) for each formal concept intent (step 4 in Algorithm 1). Retaining the top n concepts with maximal values of the chosen information criterion, we have a set of rules to classify the current test instance. For each concept we define a classification rule with concept intent as a premise and the most common value of class attribute among the instances of concept extent as a conclusion.

Finally, we predict the value of the class attribute for the current test instance simply via majority rule among n "best" classification rules (step 5 in Algorithm 1). Then the calculated formal concept intents are stored (step 6), and the cycle is repeated for the next test instance.

4.2 Numeric-attribute case

In our approach, we deal with numeric attributes similarly to what is done in C4.5 algorithm [16]. We compute α percentiles $x_1, \ldots, x_\alpha$ for each numeric attribute x and introduce $2{*}\alpha$ new binary attributes in a form "$x \geq x_1$", "$x < x_1$", ..., "$x \geq x_\alpha$", "$x < x_\alpha$". Let us demonstrate steps 1 and 2 of Algorithm 1 in case of binary and numeric attributes with a sample from Kaggle "Titanic: Machine Learning from Disaster" competition dataset.[1]

Example 3. Fig. 2 shows a sample from the Titanic dataset. Let us from a formal context to classify a passenger with attributes "$Pclass = 3, SibSp = 0, Age = 34.5$". We use 25 and 50% percentiles of the *Age* attribute to binarize it. The corresponding binary table is shown in Table 2.

5 Example

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

1. Let us fix Gini impurity as an information criterion of interest and the parameters $k = 2$ and $n = 5$. Thus, we are going to classify a test instance

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

Algorithm 1 The proposed algorithm - binary attribute case

Input: $K_{train} = (G_{train}, M \cup c_{train}, I_{train})$ is a formal context (a training set),
$K_{test} = (G_{test}, M, I_{test})$ is a formal context (a test set);
$CbO(K, k)$ is the algorithm used to find all formal concepts of a formal context K with intent cardinality not exceeding k;
$inf : M \cup c_{train} \to \mathbb{R}$ is an information criterion used to rate classification rules (such as Gini impurity, Gini gain or pairwise mutual information);
k is the maximal cardinality of each classification rule's premise (a parameter);
n is the number of rules to be used for prediction of each test instance's class attribute (a parameter);
Output: c_{test}, predicted values of the class attribute for test instances in K_{test}.

$\mathcal{S} = \emptyset, c_{test} = []$. Initialize a set of formal concept intents (a.k.a. closed itemsets). This set will be used to form classification rules for each test instance from G_{test}. Initialize a list of predicted labels for test instances.

for each test instance $g_t \in G_{test}$ **do**

1. Let $\mathcal{M}_t$ be a set of attributes of a test instance g_t together with the negations of the attributes not in g_t';
2. Build a formal context $K_t = \{G_{train}, \mathcal{M}_t, I_t\}$ where $I_t = I \cap (G \times \mathcal{M}_t)$. Informally, leave only a part of a context K_{train} with attributes from M_t;
3. With the CbO algorithm and a set $\mathcal{S}$ of already computed formal concept intents, find all formal concepts of a formal context K_t with intent cardinality not exceeding the value of the parameter k;
4. Meanwhile, calculate the value of the criterion inf for each concept intent and keep n intents with highest values of the criterion. For each "top-ranked" concept intent B_i determine c_i, the most common class among objects from B_i'. Thus, form $\{B_i \to c_i\},\ i = 1 \dots n$, a set of classification rules for g_t;
5. Predict the value of the class attribute for g_t via a majority rule among $\{B_i \to c_i\},\ i = 1 \dots n$. Add it to c_{test};
6. Add calculated intents to $\mathcal{S}$.

end for

Table 2. A formal context built to classify a test passenger "$Pclass = 3, SibSp = 0, Age = 34.5$."

$Pclass! = 1$	$Pclass == 3$	$SibSp == 0$	$SibSp! = 1$	$Age \geq 26$	$Age < 35$
×	×				
				×	
×	×	×	×	×	×
				×	
×	×	×	×	×	

	Pclass	SibSp	Age	Survived
0	3	1	22	0
1	1	1	38	1
2	3	0	26	1
3	1	1	35	1
4	3	0	35	0

Fig. 2. A sample from the Titanic dataset. Attributes: Pclass - passenger's class, SibSp - the number of passenger's siblings and spouses on board, Age - passenger's age, Survived - whether a passenger survived in the Titanic disaster.

with 5 rules with at most 2 attributes in premise having highest gain in Gini impurity.

2. The case *"Outlook=sunny, Temperature=cool, Humidity=high, Windy=false"* corresponds to a set of attributes $\{os,\ tc\}$ describing the test instance. Or, if we consider the negations of the attributes, such case is described with a set of attributes: $\{\bar{or}, \bar{oo}, os, tc, t\bar{m}, \bar{th}, \bar{hn}, \bar{w}\}$.
3. We build a formal context with objects being the training set instances and attributes of a test instance – $\{\bar{or}, \bar{oo}, os, tc, t\bar{m}, \bar{th}, \bar{hn}, \bar{w}\}$. The corresponding binary table is shown in Table 3.

Table 3. 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).

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

4. The diagram of the for the formal context given by Table 3 is shown in Fig. 3. The horizontal line separates the concepts with intents having at most 2 attributes.
5. 13 formal concepts with intents having at most 2 attributes give rise to 13 classification rules. Top 5 rules having the highest gain in Gini impurity are given in Table 4.

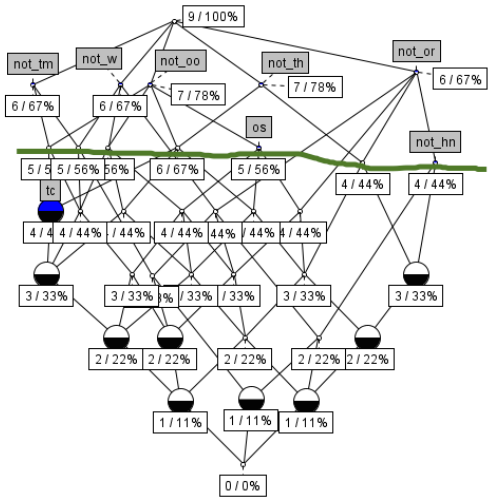


Fig. 3. The concept lattice for the formal context represented with Table 3. The horizontal line separates the concepts with intents having at most 2 attributes.

Table 4. Top 5 rules to classify the test instance *"Outlook=sunny, Temperature=cool, Humidity=high, Windy=false"*

Rule	**Gini gain**
$\{\bar{w}, \bar{tm}\} \rightarrow play$	0.278
$\{\bar{oo}, \bar{th}\} \rightarrow play$	0.111
$\{\bar{oo}, os\} \rightarrow play$	0.044
$\{\bar{oo}, \bar{tm}\} \rightarrow play$	0.044
$\{\bar{or}, \bar{hn}\} \rightarrow play$	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

We compare the proposed classification algorithm ("PCL" for Pattern Concept Lattice based classification) with the results from [12] on several datasets from the UCI machine learning repository.[2]

We used pairwise mutual information as a criterion for rule selection. Parameters $k \in \{3, \ldots 7\}$ and $n \in \{1, \ldots 5\}$ were chosen via 5-fold cross validation.

[2] https://archive.ics.uci.edu/ml/datasets/{Breast+Cancer, Heart+Disease, Hepatitis, Horse+Colic, Ionosphere, Iris, Lymphography, Pima+Indians+Diabetes, Wine, Zoo}

Table 5. Error rates in classification experiments with the UCI machine learning repository datasets.

Dataset	**C4.5**	**LazyDT**	**EAC**	**LAC**	**PCL**
breast	3.9	5.1	3.6	**3.2**	3.3
heart	18.9	17.7	18.1	16.9	**16.5**
hepatitis	22.6	20.3	17.9	17.1	**16.8**
horse	16.3	17.2	15.4	14.5	**14.2**
ionosphere	8.0	8.0	**7.6**	7.8	7.7
iris	5.3	5.3	4.9	**3.2**	3.3
lymph	21.0	20.1	20.2	19.1	**17.9**
pima	27.5	25.9	27.5	22.0	**21.6**
wine	7.9	7.9	7.2	**3.4**	4.1
zoo	7.8	7.8	6.6	**6.5**	7.1
Average	13.92	13.53	12.9	11.37	**11.25**

The described algorithms were implemented in Python 2.7.3 on a dual-core CPU (Core i3-370M, 2.4 GHz) with 3.87 GB RAM.

The algorithm was also tested on a 2001 Predictive Toxicology Challenge (PTC) dataset.[3] Please refer to [9] and [17] for the description of the problem and some notions on pattern structures with descriptions given by labeled graphs. Here we compare the results of the proposed algorithm (Pattern Concept Lattice-based classification) and the previously developed graphlet-based lazy associative classification on the PTC dataset. The results are shown in Table 6.

Table 6. Experimental results for the male rats group of the PTC dataset "GLAC" stands for "Graphlet-based lazy associative classification" [9], "PCL" stands for Pattern concept lattice-based classification (proposed here). For "PCL" we used 5 best rules to classify each test instance.

	K nodes	Accuracy	Precision	Recall	F-score	Time (sec.)
GLAC	2	0.36	0.32	0.33	0.32	5.8
	3	0.68	0.83	0.68	0.75	17.4
	4	0.59	0.57	0.62	0.59	65.7
	5	0.55	0.7	0.62	0.66	196
PCL	2	0.4	0.38	0.33	0.35	15.5
	3	0.69	0.85	0.66	0.74	39
	4	0.62	0.6	0.61	0.6	161.3
	5	0.58	0.74	0.61	0.69	412.4

To clarify, in both algorithms k-graphlet (parameter "K nodes" in Table 6) graph intersections were build. In "GLAC", each test instance is classified via voting among all classification hypotheses. In "PCL", only n best (according to some

[3] http://www.predictive-toxicology.org/ptc/

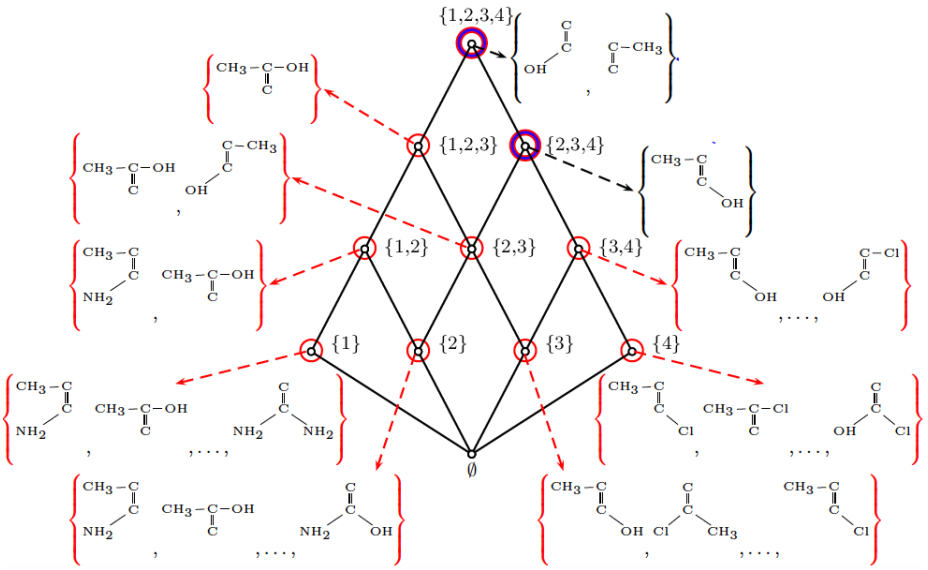


Fig. 4. An example of a pattern concept lattice.

information criterion) closed hypotheses are chosen (here we used n=5). As we can see, "PCL" works slightly better with this dataset suggesting that choosing "best" hypotheses for classification may lead to more accurate classification.

7 Conclusion 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 results while providing succinct classification rules. 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 interpret random forests as a search for an optimal hypothesis in a concept lattice and try to compete with this popular classification technique.

References

1. Grigorios Tsoumakas, Apostolos Papadopoulos, Weining Qian, Stavros Vologiannidis, Alexander D'yakonov, Antti Puurula, Jesse Read, Jan Svec, and Stanislav Semenov, "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.* October 12-14 2014, vol. 8787 of *Lecture Notes in Computer Science*, pp. 541–548, Springer.

2. X. Li and Y. Zhong, "An overview of personal credit scoring: Techniques and future work," *International Journal of Intelligence Science*, vol. 2, no. 4A, pp. 181–189, 2012.
3. Bernhard Ganter and Rudolf Wille, *Formal Concept Analysis: Mathematical Foundations*, Springer-Verlag New York, Inc., Secaucus, NJ, USA, 1st edition, 1997.
4. Thomas M. Mitchell, *Machine Learning*, McGraw-Hill, Inc., New York, NY, USA, 1 edition, 1997.
5. Itamar Hata, Adriano Veloso, and Nivio Ziviani, "Learning accurate and interpretable classifiers using optimal multi-criteria rules.," *JIDM*, vol. 4, no. 3, pp. 204–219, 2013.
6. Bernhard Ganter and Sergei Kuznetsov, "Pattern Structures and Their Projections," in *Conceptual Structures: Broadening the Base*, Harry Delugach and Gerd Stumme, Eds., vol. 2120 of *Lecture Notes in Computer Science*, pp. 129–142. Springer, Berlin/Heidelberg, 2001.
7. Sergei O. Kuznetsov, "Fitting pattern structures to knowledge discovery in big data," in *Formal Concept Analysis: 11th International Conference, ICFCA 2013, Dresden, Germany, May 21-24, 2013. Proceedings*, Peggy Cellier, Felix Distel, and Bernhard Ganter, Eds., Berlin, Heidelberg, 2013, pp. 254–266, Springer Berlin Heidelberg.
8. Sergei O. Kuznetsov, "Scalable Knowledge Discovery in Complex Data with Pattern Structures," in *PReMI*, Pradipta Maji, Ashish Ghosh, M. Narasimha Murty, Kuntal Ghosh, and Sankar K. Pal, Eds. 2013, vol. 8251 of *Lecture Notes in Computer Science*, pp. 30–39, Springer.
9. Yury Kashnitsky and Sergei O. Kuznetsov, "Lazy associative graph classification," in *Proceedings of the 4th International Workshop "What can FCA do for Artificial Intelligence?", FCA4AI 2015, co-located with the International Joint Conference on Artificial Intelligence (IJCAI 2015), Buenos Aires, Argentina, July 25, 2015.*, 2015, pp. 63–74.
10. David W. Aha, Ed., *Lazy Learning*, Kluwer Academic Publishers, Norwell, MA, USA, 1997.
11. Jerome H. Friedman, "Lazy decision trees," in *Proceedings of the Thirteenth National Conference on Artificial Intelligence - Volume 1*. 1996, AAAI'96, pp. 717–724, AAAI Press.
12. Adriano Veloso, Wagner Meira Jr., and Mohammed J. Zaki, "Lazy Associative Classification," in *Proceedings of the Sixth International Conference on Data Mining*, Washington, DC, USA, 2006, ICDM '06, pp. 645–654, IEEE Computer Society.
13. Radim Belohlavek, Bernard De Baets, Jan Outrata, and Vilem Vychodil, "Inducing decision trees via concept lattices," *International Journal of General Systems*, vol. 38, no. 4, pp. 455–467, 2009.
14. Sergei. O. Kuznetsov, "A fast algorithm for computing all intersections of objects from an arbitrary semilattice," *Nauchno-Tekhnicheskaya Informatsiya Seriya 2-Informatsionnye Protsessy I Sistemy*, , no. 1, pp. 17–20, 1993.
15. S. Andrews, "In-close, a fast algorithm for computing formal concepts," in *CEUR Workshop Proceedings*, 2009, vol. 483.
16. J. Ross Quinlan, *C4.5: Programs for Machine Learning*, Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 1993.
17. Sergei O. Kuznetsov and Mikhail V. Samokhin, "Learning Closed Sets of Labeled Graphs for Chemical Applications," in *ILP*, Stefan Kramer and Bernhard Pfahringer, Eds. 2005, vol. 3625 of *Lecture Notes in Computer Science*, pp. 190–208, Springer.