Lazy Classification with Interval Pattern Structures: Application to Credit Scoring

Alexey Masyutin, Yury Kashnitsky, and Sergei Kuznetsov

National Research University Higher School of Economics
Scientific-Educational Laboratory for Intelligent Systems and Structural Analysis
Moscow, Russia
alexey.masyutin@gmail.com,ykashnitsky@hse.ru,skuznetsov@hse.ru

Abstract. Pattern structures allow one to approach the knowledge extraction problem in case of arbitrary object descriptions. They provide the way to apply Formal Concept Analysis (FCA) techniques to non-binary contexts. However, in order to produce classification rules a concept lattice should be built. For non-binary contexts this procedure may take much time and resources. In order to tackle this problem, we introduce a modification of the lazy associative classification algorithm and apply it to credit scoring. The resulting quality of classification is compared to present methods adopted in bank system.

1 Introduction

Banks and credit institutions face classification problem each time they consider a loan application. In most general case, bank is eager to have a tool to discriminate between solvent and potentially delinquent borrowers, i.e. the tool to predict whether the applicant is going to meet his or her obligations or not. Before 1950s such decision making process was expert driven and involved no explicit statistical modeling. The decision whether to grant a loan or not was made upon an interview and after retrieving information about spouse and close relatives [5]. From the 1960s, banks have started to adopt statistical scoring systems that were trained on datasets of applicants, consisting of their socio-demographic factors and loan application features. As far as mathematical models are concerned, they were typically logistic regressions run on selected set of attributes. Apparently, a considerable amount of research was done in the field of alternative machine learning techniques seeking the goal to improve the results of the wide-spread scorecards [8,9,10,11,12].

All mentioned methods can be divided into two groups: the first one provides the result difficult for interpretation, so-called "black box" models, the second group provides interpretable results and clear model structure. The key feature of risk management practice is that, regardless of the model accuracy, it must not be the black box. That is why methods such as neural networks and SVM classifiers did not earn much trust within banking community [5]. The dividing

hyperplane in an artificial high-dimensional space (dependent on the chosen kernel) cannot be easily interpreted in order to claim the reject reason for the client. As far as neural networks are concerned, they also do not provide the user with a set of reasons why a particular loan application has been approved or rejected. In other words, these algorithms do not provide the decision maker with knowledge. The predicted class is generated, but no knowledge is retrieved from data.

On the contrary, alternative methods such as associative rules and decision trees provide the user with easily interpretable rules which can be applied to the loan application. FCA-based algorithms also belong to the second group since they use concepts in order to classify objects. The intent of the concept can be interpreted as a set of rules that is supported by the extent of the concept. However, for non-binary context the computation of the concepts and their relations can be very time-consuming. In case of credit scoring we deal with numerical context, as soon as categorical variables can be transformed into set of dummy variables. Lazy classification [17] seems to be appropriate to use in this case since it provides the decision maker with the set of rules for the loan application and can be easily parallelized. In this paper, we modify lazy classification framework and test it on credit scoring data of a top-10 Russian bank.

The paper is structured as follows: section 2 provides basic definitions. Section 3 argues why the original setting can be inconsistent in case of large numerical context and describes the proposed modification and its parameters. Section 5 describes the data in hand and some experiments with parameters of the algorithm. Finally, section 6 concludes the paper.

2 Main Definitions

First, we recall some standard definitions related to Formal Concept Analysis, see e.g. [1,2].

Let G be a set (of objects), let (D, $\sqcap$) be a meet-semi-lattice (of all possible object descriptions) and let δ: G $\to$ D be a mapping. Then (G, $\underline{D}$,δ), where $\underline{D}$ =(D, $\sqcap$), is called a *pattern structure* [1], provided that the set δ(G) := $\{\delta(g)|g \in G\}$ generates a complete subsemilattice (D_δ, $\sqcap$) of (D, $\sqcap$), i.e., every subset X of $\delta(G)$ has an infimum $\sqcap$X in (D, $\sqcap$). Elements of D are called *patterns* and 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, $\underline{D}$, δ) gives rise to the following *derivation operators* $(\cdot)^\diamond$:

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

These operators form a Galois connection between the powerset of G and $(D, \sqcap)$. The pairs (A, d) satisfying A $\subseteq$ G, d $\in$ D, $A^\diamond$ = d, and A = $d^\diamond$ are called

pattern concepts of (G,$\underline{D}$, δ), with *pattern extent* A and *pattern intent* d. Operator $(\cdot)^{\diamond\diamond}$ is an algebraical closure operator on patterns, since it is idempotent, extensive, and monotone [1].

The concept-based learning model for standard object-attribute representation (i.e., formal contexts) is naturally extended to pattern structures. Suppose we have a set of positive examples G_+ and a set of negative examples G_- w.r.t. a target attribute, $G_+ \cap G_- = \emptyset$, objects from
G_τ = G $\setminus (G_+ \cup G_-)$ are called undetermined examples. A pattern c $\in$ D is an α - weak positive premise (classifier) iff:

$$\frac{||c^\diamond \cap G_-||}{||G_-||} \leq \alpha \; and \; \exists A \subseteq G_+ : c \sqsubseteq A^\diamond$$

A pattern h $\in$ D is an α - weak positive hypothesis iff:

$$\frac{||h^\diamond \cap G_-||}{||G_-||} \leq \alpha \; and \; \exists A \subseteq G_+ : h = A^\diamond$$

In case of credit scoring we work with pattern structures on intervals as soon as a typical object-attribute data table is not binary, but has many-valued attributes. Instead of binarizing (scaling) data, one can directly work with many-valued attributes by applying interval pattern structure. For two intervals $[a_1, b_1]$ and $[a_2, b_2]$, with $a_1, b_1, a_2, b_2 \in \mathbb{R}$ the *meet operation* is defined as [4]:

$[a_1, b_1] \sqcap [a_2, b_2] = [min(a_1, a_2), max(b_1, b_2)]$.

The original setting for lazy classification with pattern structures can be found in [3].

3 Modification of lazy classification algorithm

In credit scoring the object-attribute context is typically numerical. Factors can have arbitrary distributions and take wide range of values. At the same time categorical variables and dummies can be present. With relatively large number of attributes (over 30-40) it produces high-dimensional space of continuous variables. That is when the result of meet operator tends to be very specific, i.e. for almost every $g \in G$ only g and g_n have the descriptin $\delta(g_n) \sqcap \delta(g)$. This happens due to the fact that numerical variables, ratios especially, can have unique values for every object. This results in that for test object g_n the number of positive and negative premises is close to the number of observations in those context correspondingly. In other words, too specific descriptions are usually not falsified (i.e. there are no ojects of opposite class with such description) and almost always form either positive or negative premises. Therefore, the idea of voting scheme for lazy classification in the case of high dimensional numerical context may turn out to be obscure. Thus, it seems reasonable to seek the concepts with larger extent and with not too specific intent. At the same we would like to preserve the advantages of lazy classification, e.g. no need to compute full concept lattice, easy parallelization etc. The way to increase the extent of the generated concepts

is to consider intersection of the test object with more than one element from the positive (negative) context. What is the suitable number of objects to take for intersection? In our modification we consider this as a parameter subsample size and perform grid search. The parameter is expressed as percentage of the observations in the context. As subsample size grows, the resulting intersection $\delta(g_1) \sqcap \ldots \sqcap \delta(g_k) \sqcap \delta(g)$ becomes more generic and it is more frequently falsified by the objects from the opposite context. Strictly speaking, in order to replicate the lazy classification approach, one should consider all possible combinations of the chosen number of objects from the positive (negative) context. Apparently, this is not applicable in the case of large datasets. For example, having 10 000 objects in positive context and having subsample size equal to only two objects will produce almost 50 mln combinations for intersection with the test object. Therefore, we randomly take the chosen number of objects from positive (negative) context as candidates for intersection with the test object. The number of times (number of iterations) we randomly pick a subsample from the context is also tuned through grid search. Intuition says , the higher the value of the parameter the more premises are mined from the data. However, the obvious penalty for increasing the value of this parameter is time and resources required for computing intersections. As we mentioned, the greater the subsample size, the more it is likely that $(\delta(g_1) \sqcap \ldots \sqcap \delta(g_k) \sqcap \delta(g))^{\diamond}$ contains the object of the opposite class. In order to control this issue, we add third parameter which is alpha-threshold. If the percentage of objects from the positive (negative) context that falsify the premise $\delta(g_1) \sqcap \ldots \sqcap \delta(g_k) \sqcap \delta(g)$ is greater than alpha-threshold of this context than the premise will be considered as falsified, otherwise the premise will be supported and used in the classification of the test object.

4 Voting schemes

The final classification of a test object is based on a voting scheme among premises. In most general case voting scheme F is a mapping:

$$F(g_{test}, h_1^+, ..., h_p^+, h_1^-, ..., h_n^-) \to [-1, 1, \emptyset]$$

where g_{test} is the test object with unknown class, h_i^+ is a positive premise $\forall i = \overline{1, p}$ and h_j^- is a negative premise $\forall j = \overline{1, n}$, -1 is a label for negative class, and 1 is a lable for positive class (i.e. defaulters). In other words, F is an aggregating rule that takes premises as input and gives the classification label as an output. Note, that we allow for an empty label. If the label is empty it is said that the voting rule abstains from classification. There may be different approaches to build up aggregating rules. The voting scheme is built upon weighting function $\omega(\cdot)$, aggregation operator $A(\cdot)$ and comparing operator $\otimes$.

$$F(\omega(\cdot), A(\cdot), \otimes) =$$
$$= (A_{i=1}^{p}[\omega(h_i^+)]) \otimes (A_{j=1}^{n}[\omega(h_j^-)])$$

In order to configure a new weighting scheme it is sufficient to define the operators and weighting function. In this paper we use the number of positive

versus negative premises. In this case the rule allows the test object to satisfy both positive and negative premises which decreases the rejects from classification. The weighting function, aggregation operator and comparing operator are defined as follows:

$$A(h) = \sum h$$

$$\omega(h) = \begin{cases} 1, & if\ \delta(g_{test}) \sqsubseteq h \\ 0, & otherwise \end{cases}$$

$$a \otimes b = \begin{cases} sign(b-a), & if\ a \neq b \\ \emptyset, & a = b \end{cases}$$

So the label for a test object g_n is defined by following mapping:

$$F(g_{test}, h_1^+, ..., h_p^+, h_1^-, ..., h_n^-) =$$
$$= (\sum_{i=1}^{p} [\delta(g_{test}) \sqsubseteq h_i^+]) \otimes (\sum_{j=1}^{n} [\delta(g_{test}) \sqsubseteq h_j^-])$$

However, one can think of margin $b - a$ as a measure for discrimination between two classes and consider the decision boundary based on ROC analysis, for instance. This approach is good for decreasing the number of rejects from classification, but it does not account for the support of the premises. Naturally, one would give more weight to the premise with large image (with higher support). Also, if the number of positive and negative premises is equal the rule rejects from classification.

5 Experiments

The data we used for the computation represent the customers and their metrics assessed on the date of loan application. The applications were approved by the bank credit policy and the clients were granted the loans. After that the loans were observed for the fact of delinquency. The dataset is divided into two contexts positive and negative. The positive context is the set of loans where the target attribute is present. The target attribute in credit scoring is typically defined as more than 90 days of delinquency within the first 12 months after the loan origination. So, the positive context is the set of bad borrowers, and the negative context consists of good ones. Each context consists of 1000 objects in order that voting scheme concerned in the second section was applicable. The test dataset consists of 300 objects and is extracted from the same population as the positive and negative contexts. Attributes represent various metrics such as loan amount, term, rate, payment-to-income ratio, age of the borrower, undocumented-to-documented income, credit history metrics etc. The set of attributes used for the lazy classification trials contained 28 numerical attributes. In order to evaluate the accuracy of the classification we calculate the

Gini coefficient for every combination of parameters based on 300 predictions on the test set. Gini coefficient is calculated based on the margin between the number of objects within positive premises and negative ones. In fact, the margin is the analog for the score value in credit scorecards. When the subsample size is low, the intersections of the test object description and the members of positive (negative) context tend to be more specific. That is why, a relatively high number of premises are mined and used for the classification. As subsample size increases, the candidates for premises start being generic and it is likely that there exists certain amount of objects from the opposite context which also satisfy the description. If alpha-threshold is low, the frequency of rejects from classification is high. The dynamics of premise mining is demonstrated on the following graphs:

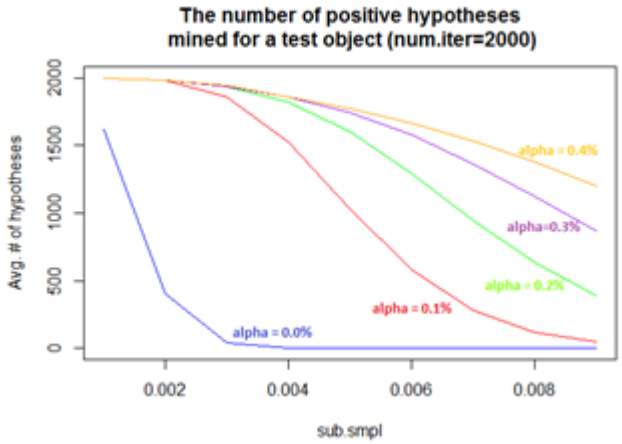

Fig. 1. The dynamics of α - weak positive premises mining

The average number of premises mined for a test object is dropping as expected with the increase in the subsample size and the drop is quicker for higher alpha-thresholds. This supports the idea, that if lazy classification is run in its original setting upon the numerical context (i.e. when subsample size consists of only one object) the number of premises generated is close to the number of objects in the context, so the premises can be considered as too specific. The descriptive graphs above allows one to expect that the proposed parameters of the algorithm can be tuned (grid searched), so as to tackle the trade-off between the high number of premises used for classification and the size of their support. The average number of positive premises tends to fall slightly faster compared to negative premises. Below we present the classification accuracy obtained for different combination of parameters (grid search).

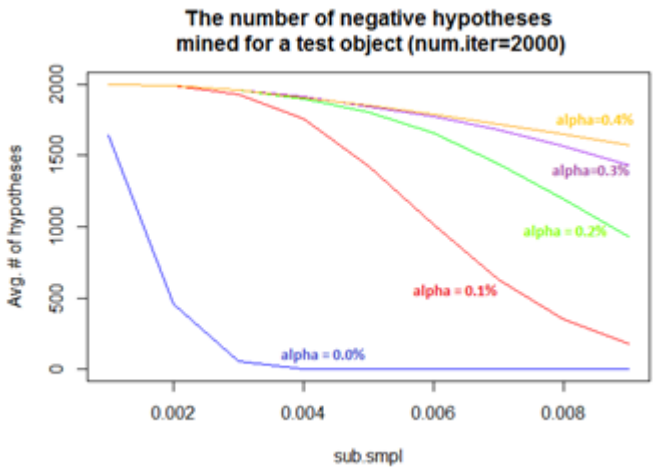

Fig. 2. The dynamics of negative α - weak premises mining

Table 1. Gini coefficients for the parameters grid search

		Subsample size								
Alpha-threshold	Number of iterations	0.1%	0.2%	0.3%	0.4%	0.5%	0.6%	0.7%	0.8%	0.9%
0.0%	100	40%	44%	39%	18%	1%	0%	0%	0%	0%
	150	35%	46%	35%	5%	0%	0%	0%	0%	0%
	200	42%	37%	36%	12%	5%	1%	0%	0%	0%
	500	39%	44%	44%	25%	6%	1%	0%	0%	0%
	1000	44%	47%	44%	41%	11%	3%	0%	0%	0%
	2000	44%	48%	46%	36%	17%	4%	0%	0%	0%
0.1%	100	33%	37%	40%	40%	44%	43%	34%	32%	34%
	150	41%	34%	33%	43%	41%	47%	41%	37%	37%
	200	40%	40%	34%	42%	51%	43%	44%	41%	36%
	500	37%	42%	47%	49%	51%	49%	43%	41%	34%
	1000	37%	42%	46%	48%	49%	48%	43%	43%	37%
	2000	39%	43%	45%	49%	51%	49%	46%	41%	38%
	5000	43%	40%	44%	49%	46%	50%	48%	38%	36%
0.2%	100	29%	38%	42%	32%	43%	37%	46%	43%	37%
	150	27%	42%	41%	41%	36%	47%	48%	45%	41%
	200	32%	40%	43%	42%	42%	49%	46%	47%	48%
	500	39%	46%	46%	48%	47%	48%	51%	48%	51%
	1000	41%	50%	48%	47%	49%	53%	52%	52%	47%
	2000	38%	48%	50%	48%	47%	53%	52%	53%	50%
0.3%	100	35%	38%	39%	42%	39%	45%	34%	45%	39%
	150	27%	43%	44%	42%	42%	39%	37%	40%	46%
	200	34%	46%	47%	45%	49%	47%	45%	45%	52%
	500	31%	45%	49%	50%	49%	46%	50%	51%	47%
	1000	37%	48%	49%	49%	49%	47%	52%	51%	51%
	2000	38%	46%	48%	51%	51%	50%	50%	52%	52%
	5000	40%	47%	46%	51%	52%	51%	49%	51%	53%
	10000	40%	44%	43%	46%	46%	48%	50%	52%	54%
	20000	40%	43%	42%	46%	47%	49%	50%	52%	53%
0.4%	100	28%	39%	44%	48%	43%	50%	53%	42%	49%
	150	34%	42%	43%	42%	43%	52%	50%	45%	47%
	200	33%	46%	43%	47%	51%	49%	49%	42%	45%
	500	37%	50%	50%	49%	49%	49%	51%	47%	48%
	1000	40%	48%	50%	50%	51%	52%	50%	48%	50%
	2000	37%	48%	49%	49%	49%	47%	52%	49%	51%
	5000	39%	42%	42%	43%	45%	47%	49%	52%	49%

We observe the area with zero Gini coefficients where the alpha-threshold is zero and the subsample size is relatively high. That is due to the fact that almost no premises were mined during the lazy classification run. It is quite intuitive because as the subsample size grows, the intersection of the subsample with a test object results in a generic description, which is very likely to be falsified at least by one object from the opposite context. In this case the reject from classification takes place almost for all test objects. The first thing that is quite intuitive is that the more iterations are produced, the higher is the Gini on average:

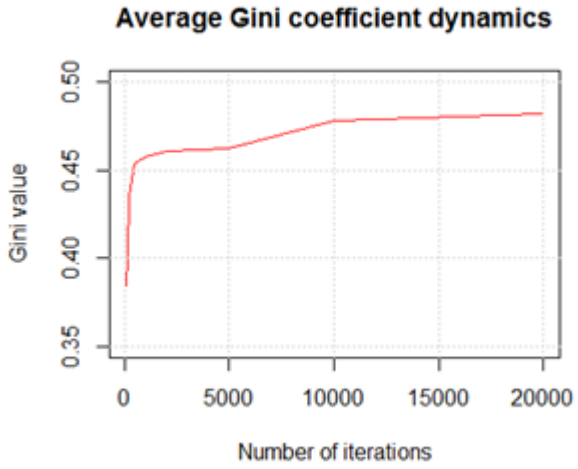

Fig. 3. Average Gini grouped by the different number of iterations (over all other parameter values)

The more times the subsamples are randomly extracted the more knowledge (in terms of premises) is generated. By increasing the number of premises used for classification according to voting scheme, we are likely to capture the structure of the data in more detail. However, the number of iterations is not the only driver of the classification accuracy in our case. We find a range with relatively high Gini in the area of mild alpha-threshold and relatively high subsample size. It also seems natural as soon as the support of a good predictive rule (i.e. premise) is expected to be higher than its support in the opposite context. We elaborate further and run additional grid search in range of parameters providing high Gini coefficient:

Table 2. Gini coefficients for the parameters grid search on specified area

		Subsample size					
Alpha-thresh-old	Number of iterations	1.0%	1.1%	1.2%	1.3%	1.4%	1.5%
0.3%	500	51%	49%	48%	43%	41%	38%
	1000	52%	51%	48%	45%	43%	39%
	2000	54%	53%	49%	47%	46%	38%
	5000	55%	52%	50%	47%	46%	40%
	10000	56%	53%	50%	47%	47%	40%
	20000	55%	53%	51%	46%	48%	41%

According to performed grid search the range with the highest Gini (55%-56%) on the test sample is in range with following parameter values: alpha-threshold = 0,3%, number of iterations = 10000, subsample size = 1,0%. The result was compared to three benchmarks that are traditionally used in the credit scoring within the bank system: logistic regression, scorecard and decision tree. It should be cleared what is implied by the scorecard classifier. Mathematical architecture of the scorecard is based on logistic regression which takes the transformed variables as input. The transformation of the initial variables which is typically used is WOE-transformation [14]. It is wide-spreaded in credit scoring to apply such a transformation to the input variables as soon as it accounts for non-linear dependencies and it also provides certain robustness coping with potential outliers. The aim of the transformation is to divide each variable into not more than k categories. The thresholds are derived so as to maximize the information value of a variable [14]. Having each variable binned into categories, the log-odds ratio is calculated for each category. Finally, instead of initial variables the discrete valued variables are considered as input in logistic regression. The properties of the decision tree were as follows: we ran CART with two possible child nodes from each parent node. The criterion for optimal threshold calculation was the greatest entropy reduction. The number of terminal nodes was not explicitly restricted; however, the minimum size of the terminal node was set to 50. As far as logistic regression is concerned, the variable selection was performed based on stepwise approach [15]. As for scorecard, the variables were initially selected based on their information value after the WOE-transformation. The comparison of the classifiers performance based on test sample of 300 objects is given in Table 3.

Table 3. Modified lazy classification algorithm versus models adopted in the bank

	Gini on test sample
Logistic regression	47.38%
Scorecard (Logistic based on WOE-transformation)	51.89%
CART (minsize= 50)	54.75%
MLCA (s = 1%, a=0.3%, n=10000)	56.30%

6 Conclusion

When dealing with large numerical datasets, lazy classification may be preferable to classification based on explicitly generated calssifiers, since it requires less time and memory resources [3]. However, the original lazy classification setting in case of high dimensional numerical feature space meets certain limitation. The limitation is that, when intersecting descriptions of a test object and every object from the context, one is likely to acquire premises with image consisting only of those two objects. In other words, the premises tend to be very specific for the context and, therefore, the number of positive and negative premises is likely to be equal to the number of the objects in the contexts. The weighting cannot be considered helpful in this case as soon as the premises will have very similar low support. In this paper, we modified the original lazy classification setting by making it, in fact, a stochastic procedure with three parameters: subsample size, number of iterations and alpha-threshold. In effect, the modified algorithm mines the premises with relatively high support that will be used for the classification of the test object. The classification is then carried out upon the predefined voting scheme. We applied the introduced procedure to the retail loan classification problem. The data we used for was provided during the pilot project with one of the top-10 banks in Russia, the details are not provided due to non-disclosure agreement. The positive and negative contexts both had 1000 objects with 28 numerical attributes. The accuracy of the algorithm was evaluated on the test dataset consisting of 300 objects. Gini coefficient was chosen as accuracy metric. We performed the basic grid search by running the modified lazy classification algorithm with different parameter values. The classification accuracy of the algorithm was compared to the conventionally adopted models used in the bank. The benchmark models were logistic regression, scorecard and decision tree. The proposed algorithm outperforms the logistic regression the scorecard with the subsample size parameter around 1%, alpha-threshold equal to 0,3% and with number of iterations over 5000. The performance of the decision tree is at the comparable level with the proposed algorithm, however, the modified lazy classification is slightly better in terms of Gini coefficient. As an area for further research, one can consider and compare accuracy when other voting schemes are used. It is expected that taking into account premises' specificity

one can improve overall accuracy of the classification algorithm or, alternatively, one will reach the same accuracy given less number of iterations, which can save the time resources required for the calculations.

References

1. 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.
2. Ganter, B., Wille, R.: Formal concept analysis: Mathematical foundations. Springer, Berlin, 1999.
3. 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.
4. Mehdi Kaytoue, Sergei O. Kuznetsov; Amedeo Napoli, Sebastien Duplessis, "Mining gene expression data with pattern structures in formal concept analysis", Information Sciences,181,10,1989-2001, Elsevier, 2011
5. Thomas L., Edelman D., Crook J. (2002) Credit Scoring and Its Applications, Monographs on Mathematical Modeling and Computation, SIAM: Pliladelphia, pp. 107–117
6. Bigss, D., Ville, B., and Suen, E. (1991). A Method of Choosing Multiway Partitions for Classification and Decision Trees. Journal of Applied Statistics, 18, 1, 49-62.
7. Naeem Siddiqi, Credit Risk Scorecards: Developing and Implementing Intelligent Credit Scoring, WILEY,ISBN: 978-0-471-75451-0, 2005
8. B Baesens, T Van Gestel, S Viaene, M Stepanova, J Suykens, Benchmarking state-of-the-art classification algorithms for credit scoring, Journal of the Operational Research Society 54 (6), 627-635, 2003
9. Ghodselahi A., A Hybrid Support Vector Machine Ensemble Model for Credit Scoring, International Journal of Computer Applications (0975 – 8887), Volume 17– No.5, March 2011
10. Yu, L., Wang, S. and Lai, K. K. 2009. An intelligent agent-based fuzzy group decision making model for financial multicriteria decision support: the case of credit scoring. European journal of operational research. vol. 195. pp.942-959.
11. Gestel, T. V., Baesens, B., Suykens, J. A., Van den Poel, D., Baestaens, D.-E. and Willekens, B. 2006. Bayesian kernel based classification for financial distress detection. European journal of operational research. vol. 172. pp. 979-1003.
12. P. Ravi Kumar and V. Ravi, "Bankruptcy Prediction in Banks and Firms via Statistical and Intelligent Techniques-A Review," European Journal of Operational Research, Vol. 180, No. 1, 2007, pp. 1-28.
13. 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
14. SAS Institute Inc. (2012), Developing Credit Scorecards Using Credit Scoring for SAS® Enterprise Miner™ 12.1, Cary, NC: SAS Institute Inc.
15. Hocking, R. R. (1976) "The Analysis and Selection of Variables in Linear Regression," Biometrics, 32.

16. Mehdi Kaytoue, Sergei O. Kuznetsov, Amedeo Napoli, and Sebastien Duplessis, "Mining gene expression data with pattern structures in formal concept analysis," Information Sciences, vol. 181, no. 10, pp. 1989–2001, May 2011.
17. Veloso, A. & Jr., W. M. (2011), Demand-Driven Associative Classification., Springer.

Algorithm 1 Lazy Classification by Sub-Samples in Numeric Context

Input: $\{Pos_{data}, Neg_{data}\}$ – positive and negative numerical contexts.
N^+, N^- – number of objects in the contexts. It is preferable that the positive and negative contexts are of the same size.
M – number of attributes.
sub.smpl – percentage of the context randomly used for intersection with the test object (parameter).
num.iter – number of iterations (resamplings) during the premise mining (parameter).
alpha.threshold is the maximum allowable percentage of the opposite context for that the premise is not falsified (parameter).
t – test object.

Output: $margin_t$ – measure that is produced by the voting rule.
y_t – class labels predicted for the test object.

for *iter* from 1 to *num.iter* **do**
S=random.sample(Pos_{data},size=$sub.smpl \cdot N^+$) — mine positive α - weak premises
$descr = \delta(g_1) \sqcap ... \sqcap \delta(g_s) \sqcap \delta(t)$
$Neg_{image} = \{x \in descr^{\diamond} | x \in Neg_{data}\}$
if $||Neg_{image}|| < alpha.threshold \cdot N^-$ **then**
Add *descr* to positive α - weak premises set
else
Do nothing
end if
S=random.sample(Neg_{data},size=$sub.smpl \cdot N^-$) — mine α - weak negative premises
$descr = \delta(g_1) \sqcap ... \sqcap \delta(g_s) \sqcap \delta(t)$
$Pos_{image} = \{x \in descr^{\diamond} | x \in Pos_{data}\}$
if $||Pos_{image}|| < alpha.threshold \cdot N^+$ **then**
Add *descr* to negative α - weak premises set
else
Do nothing
end if
end for
$p = dim$(set of positive α - weak premises)
$n = dim$(set of negative α - weak premises)
Choose voting scheme: $A(\cdot), w(\cdot), \otimes$
$pos.power = A_i^p(w(h_i^+))$
$neg.power = A_j^n(w(h_j^-))$
$margin = pos.power - neg.power$
$y_t = pos.power \otimes neg.power$