

# Live Logic<sup>TM</sup>: Method for Approximate Knowledge Discovery and Decision Making

Marina Sapir, David Verbel, Angeliki Kotsianti, and Olivier Saidi

Aureon Biosciences, 28 Wells ave Yonkers NY 10701  
marina.sapir@aureon.com

**Abstract.** Live Logic is an integrated approach for support of the learning and decision making in conditions of uncertainty. The approach covers both induction of probabilistic logical hypotheses from known examples and deduction of the plausible solution for an unknown case based on the inducted hypotheses.

The induction method generalizes empirical data, discovering statistical patterns, expressed in logical language. The deduction method uses multidimensional ranking to reconcile contradictory patterns exhibited by a particular case.

The method was applied on clinical data of the patients with prostate cancer who underwent prostatectomy. The goal was to predict biochemical failure based on the pre- and post- operative status of the patient. The patterns found by the method proved to be insightful from the pathologist's point of view. Most of them had been confirmed on the control dataset.

In our experiments, the predictive accuracy of the Live Logic<sup>TM</sup> was also higher than that of other tested methods.

## 1 Introduction

Live Logic<sup>TM</sup> induction method is developed for learning with inconclusive, inconsistent, noisy data. We assume, the dataset of known observations represents only small part of the general population, and the used descriptors are not enough to distinguish the concepts under study completely. The problem is further complicated by requirement for the decision rule to be transparent for the persons who supply the data. This is a very common situation for medical problems, for example.

Traditional approaches building logical rules (such as decision trees [6], Logical analysis of data [13], induction logic programming [7]) build deterministic rules which do not capture the uncertain nature of the problem.

Currently, there are two major ways expressing uncertainty and building the decision systems from inconclusive data: fuzzy systems, and rough sets systems.

Fuzzy systems build rules which infer fuzzy conclusions from fuzzy premises [4]. The main source of uncertainty taken into account in the fuzzy approach is uncertainty of the descriptions of the training data. Yet, in most of applications, data contain crisp numeric features. Fuzzy systems have to use some external knowledge to introduce the membership functions artificially.

The classic rough sets classifiers work with decision space granulated by indiscernibility relationship introduced prior the analysis. In this approach, each example is considered as a decision rule, which infers its class (decision) from the conditions [5]. If all examples with indiscernible conditions have the same class, the rule is considered to be deterministic. Otherwise, the rule is said to describe a borderline case.

Both traditional approaches tend to produce large amount of rules. The finer is the granulation of the decision space, the more rules will be produced, and the smaller is support of each rule, since the size of the available training set is always limited. It leads to poor predictive ability and poor interpretability [4]. It can be seen, for example, in the application of the rough sets to predict metastases in breast cancer patients [18].

Various precision rough sets [19] approach addresses this problem by introducing probabilistic membership function. The concept of tolerance approximation spaces (see [11] and [12]) further extends possibilities of inductive learning by considering similarity relations, more general than indiscernibility relationship. Another way of increasing the power and predictive ability of the decision rules was introduced in [2] with approximate rules, with relaxed requirement on conditional probability of the correct decision.

Several other approaches are proposed to find coarse homogeneous granules in information space based on data itself, to take into account the association between the values of attributes and the decisions. For example, in the same work [2] authors propose methods of discretization for attributes with large number of values. In in [12], some methods for finding similarity relationships in data are introduced. In all the mentioned works, the coarse granules in information space are built upon indiscernibility relationship by combining fine homogeneous granules.

In this paper, we propose an alternative way to build interpretable and statistically justified decision rules with uncertain data. We discover information granules not by combining elementary homogeneous granules but by eliminating external parts of heterogeneous multidimensional blocks in attribute space, until the necessary homogeneity will be achieved.

Live Logic<sup>TM</sup> builds all the most general rules, sufficiently supported by data. The approach allows us to work with continuous attributes, without prior discretization or finding similarity relationship on each variable.

To make the decisions for new instances using these uncertain rules, we propose a novel deduction procedure, which involves preliminary classification of instances by all the rules and multidimensional ranking of these decision vectors.

## 2 Induction Method

### 2.1 Definitions

The induction procedure finds maximal sufficiently homogeneous granules in the feature space as patterns. The theoretical aspects of the proposed induction method are investigated in [10].

The procedure works with data in the form  $[X, Y]$ , where  $X = \{x_{i,j}\}$ ,  $i = 1, \dots, m$ ,  $j = 1, \dots, n$  is matrix with description of  $n$  observations (cases) by  $m$  features;  $Y = \{y_1, \dots, y_n\}$  is a binary outcome vector, assigning a class to each observation. For  $j \in [1, n]$ , if  $y_j = c$ , ( $c \in \{1, 2\}$ ), we say that the  $j$ -th observation  $x_j$  belongs to the class  $c$ . Denote  $C_1, C_2$  all the observations from the classes 1, 2, respectively.

The dataset may have features of various types, such as continuous, nominal or ordinal. For a feature  $p_i$ ,  $i \in [1, m]$ , denote  $[\alpha_i, \beta_i]$  its range in the dataset.

The procedure induces rules of the type:

$$\text{“MostLikely” } I(p_1, a_1, b_1) \& \dots \& I(p_m, a_m, b_m) \Rightarrow (y = c), \quad (1)$$

where, for every  $i \in \{1, \dots, m\}$ ,  $p_i$  is a feature,  $a_i, b_i$  are arbitrary numbers,  $I(p_i, a_i, b_i)$  is an interval predicate of the form  $a_i \leq p_i \leq b_i$ ,  $y$  is the class variable.

If both  $a_i = \alpha_i, b_i = \beta_i$ , the predicate  $I(p_i, a_i, b_i)$  is true for all possible values of the variable  $p_i$ , and it is called *trivial*. The premises in the rule above will be called *clause*. The set  $X(B)$  of the observations satisfying the clause  $B$  is called *block* of the clause  $B$ .

We assume, each feature has its own admissible set of interval predicates. Consider a case of “oriented” [16] feature, which naturally correlates with outcome. For example, the condition “degree of cancer” correlates with the “prognosis”. Then an interpretable rule may associate “small” or “large” values of the feature with the decision, not “middle” values. For such a feature, only predicates

$$I(p_i, \alpha_i, c) = (p \leq c), I(p_i, c, \beta_i) = (p \geq c),$$

including ends of the domain are pragmatically justified and admissible.

For a nominal feature, the only meaningful non-trivial predicates are equalities

$$I(p, c, c) = (p = c).$$

The trivial interval is admissible for each feature.

Also, we may want to restrict the number of the nontrivial predicates in the clauses, because complex patterns are often difficult to understand and use. Denote  $\mathbf{B}(k)$  all clauses with not more than  $k$  non-trivial admissible statements.

Suppose,  $B$  is a clause in  $\mathbf{B}(k)$ , where all predicates are admissible. Clause  $B$  is a  $(h, g)$ -*interesting* in  $\mathbf{B}(k)$  for the class  $C$ , if it satisfies the next two requirements:

1. The block  $X(B)$  is ***h-homogeneous***: The proportion of the cases of the class  $C$ , among all the cases  $X(B)$  is above the threshold  $h$ :

$$\frac{\|X(B) \cap C\|}{\|X(B)\|} \geq h.$$

2. The block  $X(B)$  is ***g-representative*** : The proportion of the all cases of the class  $C$  in  $X$ , which belong to  $X(B)$ , is above the threshold  $g$ :

$$\frac{\|X(B) \cap C\|}{\|C\|} \geq g.$$

The requirements on the interesting clause interpret the quantifier “most likely” in (1).

The goal of the induction step of the Live Logic<sup>TM</sup> is to find all  $(h, g)$ -patterns for given values  $h, g$ .

One may notice here that the concept of  $h$ -homogeneous block may be interpreted as an approximate rule from [2]. However, here it is only auxiliary concept, used to define  $(h, g)$ -interesting rules. Unlike approximate rules, the rules we build here are required to be not only sufficiently consistent, but also representative, describe enough of known examples.

An  $(h, g)$ -interesting clause for the class  $C$  is a ***(h, g)-pattern*** for the class  $C$ , if it is the most general among  $(h, g)$ -interesting clauses. If an observation satisfies the conditions of the pattern we will say that it ***exhibits*** the pattern.

One can notice the parallel and difference between Live Logic<sup>TM</sup> and association rules method (see [1] and [3]), since the definition of interesting clause closely resemble requirements on support and confidence of the interesting association rules.

The most important difference between Live Logic<sup>TM</sup> and association rules method, in our view, is that the goal of learning step in our method is patterns, which are the most general among interesting rules, not all interesting rules. The most general, representative rules are the most robust ones, because they are supported by maximum number of cases; and they are the most parsimonious and understandable, since they do not contain excessive conditions and unnecessary terms.

The discovered patterns describe the maximal sufficiently homogeneous granules in the feature space, which can be used to classify new instances.

## 2.2 The Block Algorithm

The algorithm, mostly, follows one presented in [8] and [9].

Since we have descriptions of only  $n$  cases, every feature has not more than  $n$  different values in the dataset. Therefore, it is sufficient to search patterns only among blocks with limits  $a_i, b_i$  in their interval statements taken among actual values of the feature in the dataset. We further decrease the search space by considering only clauses form  $\mathbf{B}(k)$  with admissible interval statements for each

feature. The algorithm uses “smart” search among all admissible clauses to find the patterns.

First, describe the convenient way of coding clauses. We code the interval statement  $a \leq p_j \leq b$  by the pair  $\langle L_j(a), R_j(b) \rangle$ , where  $L_j(t), R_j(t)$  are the numbers of the values of the feature  $j$ , which are less and higher than  $t$ , respectively. Notice that the trivial interval statement for any feature will be coded by the pair  $\langle 0, 0 \rangle$ . A block  $B = I_1 \& \dots \& I_m$  is coded by the sequence  $\langle d_1, \dots, d_{2m} \rangle$ , where  $d_{2i-1}, d_{2i}$  is a pair, coding  $i$ -th interval statement  $I_i$ .

Let us define the order on the set of all clauses. For clauses  $B_1 = \langle d_1, \dots, d_{2m} \rangle, B_2 = \langle q_1, \dots, q_{2m} \rangle$ ,  $B_1 \prec B_2$  if there exists  $i$  such that  $d_j = q_j$  for all  $j < i$  and  $q_i > d_i$ . The first clause in this order is the clause with all trivial intervals.

For a clause  $B$ , denote  $Nxt(B, k)$  the very next after  $B$  clause in  $\mathbf{B}(k)$  in the order  $\prec$ ;  $NxtOut(B, k)$  the very next after  $B$  block in  $\mathbf{B}(k)$  in the order  $\prec$  which is not included in  $B$ .

The current clause in the algorithm is denoted by  $W$ . The algorithm starts search with the trivial clause  $W = \langle 0, \dots, 0 \rangle$ . Let the set  $M$  be the current set of found patterns, empty at the start of the algorithm. The parameters  $h, g, k$  from the definition of a pattern and class  $C$  are selected upfront.

Let  $R(B, g)$  denote the condition that the clause  $B$  is  $g$ -representative,  $H(B, h)$  the condition that the clause  $B$  is  $h$ -homogeneous. By definition, a pattern is the most general clause  $B \in \mathbf{B}(k)$  satisfying the conditions  $R(B, g) \& H(B, h)$ . Then the algorithm [8] may be presented the next way:

- **Case 1: If  $R(W, g)$  is not true,  $W := NxtOut(W, k)$ ,**
- **Case 2: If  $R(W, g)$  and  $H(W, h)$  are true,**
  1. **if  $M$  does not contain any clause  $C : C \Rightarrow W$ , then  $M := M \cup \{W\}$ ;**
  2.  **$W := NxtOut(W, k)$ ;**
- **Case 3: If  $R(W, g)$  is true, but  $H(W, h)$  is false,  $W := Nxt(W, k)$ .**

In the first case, the current clause is not a pattern, and there are no patterns among the clauses, less general than the current one. Therefore, we skip all less general clauses in the order and find the next one not included in  $W$ .

In the second case, the current clause is a pattern, if it is not less general than any pattern in the set  $M$ . In this case, we put it in the set of the patterns  $M$ . Since no clause less general than the current one can be a pattern, we skip all next less general clauses in the order  $\prec$ .

In the third case, the clause is not a pattern. But some less general clause may be a pattern. Therefore, we proceed to test the very next clause in the order  $\prec$ .

The algorithm repeats this conditional operator in a loop until the procedure  $Nxt(W, k)$  or  $NxtOut(W, k)$  required on the current step is impossible.

The paper [8] contains proof that the algorithm above finds all patterns for the given class.

The algorithm avoids looking over many possible clauses because, when the current clause is not representative or is a pattern, we skip the less general clause, following in the order. The order is designed to maximize this advantage.

### 3 Deduction and Decision Making Apparatus

#### 3.1 Decision Vectors

Below we assume that the dataset has only two classes. By definition, patterns for a certain class describe mostly cases of this class. A pattern of a class  $C$  may be considered as a classifier: if the case exhibits the pattern, it is classified as a case of the class  $C$ . If the case does not exhibit the pattern, the classifier does not give any answer.

If each observation exhibits patterns of only one class, the final conclusion is obvious. Most of time, this is not the case. Therefore, we need a reconciliation procedure.

Suppose, we discovered  $k_1$  patterns of the first class, and  $k_2$  patterns of the second class. Then each case may be coded by a decision vector  $r$  of the length  $k_1 + k_2$ , using the next procedure:

- if the case exhibits the pattern  $i$  of the first class,  $r_i = 1$ , otherwise  $r_i = 0$ ;
- if the case exhibits the pattern  $j$  of the second class,  $r_{k_1+j} = -1$ , otherwise  $r_{k_1+j} = 0$ .

As result of this procedure, we will have dataset  $R$  with binary features, where each entry is a decision vector for an instance in the dataset  $X$ .

#### 3.2 Multidimensional U-Scoring

Having decision vectors, one now needs to compare evidence provided for each instance to belong to one class over the other. Thus, some scoring method and threshold needs to be chosen.

A traditional approach is voting: a score assigned to a vector with values 1, -1 and 0 is the sum of its values. This approach may be generalized as a “weighted voting” system, where the score is a weighted sum of the classifiers’ values. The weights need to take into account mutual correlation of the features and their relative importance for the decision process.

To overcome the need for subjective weights, we use a multidimensional U-score (mU-score) proposed by K. Wittkowski [15]. The dataset  $R$  meets the requirement for mU-scoring: each classifier is “oriented” (positively correlated with the outcome). The mU-scores of decision vectors are built upon a partial order [14] on them. In this order, instances classified by different classifiers are deemed incomparable. The order is defined as follows: for any two vectors  $r_1, r_2 \in R$ ,  $r_1 < r_2$  iff for every coordinate  $i$ ,  $r_{1i} \leq r_{2i}$  and  $r_1 \neq r_2$ .

Then, for each  $d \in R$

$$mUScore(d) = \sum_{r \in R} I(d > r) - \sum_{r \in R} I(r > d),$$

where  $I(a)$  equals 1, if  $a$  is true and is equal 0 otherwise. In another words, the score of the decision vector  $d$  is number of vectors in  $R$  which are smaller than  $d$  minus number of vectors which are larger than  $d$  in the described partial order.

The proper threshold for the separation of the classes by the mU-score may be found by optimization of a chosen criterion for sensitivity and specificity of classification.

## 4 Application of the Method

The method was applied on a dataset from Baylor College of Medicine of the patients with prostate cancer after prostatectomy. The patients were characterized by 15 clinical features. For each patient, either his known time of the biochemical failure or the last observation is present in the data. The goal was to learn to predict biochemical failure within 5 years after the surgery. For this purpose, only the patients with known time of failure and patients with observations after 5 years were selected. The dataset consists of two parts, separated by historical reasons: one part includes 171 cases, out of them 35 are failures. The second dataset contains 282 cases, with 52 failures. Thus, in each dataset, only about 20% of cases belong to the first class. Traditional machine learning methods designed to maximize accuracy of the decision (as SVM, for example) produce the rule, which classifies all or almost all cases as low risk, which is not acceptable. The goal was to develop a decision rule, having high *min(sensitivity, specificity)*, because both sensitivity and specificity are important in this case.

The induction step of the Live Logic<sup>TM</sup> had its own control. First, each pattern, obtained on one dataset, was tested on another dataset. Second, pathologist, specializing in the prostate cancer, analyzed the patterns to evaluate their consistency with current medical knowledge. Generally, the features were coded in such a way, that the higher value of the feature, the higher is the risk. The found patterns, mostly, follow this rule. One exception is the feature “uicc”, which characterizes the degree of the spread of primary tumor over prostate. The patterns for low and high risk contain conditions for high levels of spread only. The reason is that for small tumors cancer aggressiveness is difficult to recognize and the prognosis can not be certain. Below are the examples of the found patterns.

**Table 1.** Examples of patterns of high and low risk of failure

High Risk	Train %	Test %	Low Risk	Train %	Test %
tnm = 5	92	100	uicc < 4	97.3	86.7
gg1 > 3	83	87	ggtot ≤ 7		
tnm > 3			tnm ≤ 3		
ln = 1	92	100	uicc > 4	97.1	83.3
uicc > 4	90	100	gg1 ≤ 3		
tnm > 3			tnm ≤ 4		
prepsa > 11.4	86.7	88.9	uicc > 4	97.3	86.7
gg1 > 3			svi = 0		
tnm > 2			ggtot ≤ 7		
			tnm ≤ 4		

In the tables above, *tnm* stands for “TNM stage”, *ggtot* means “Prostatectomy Gleason Grade”, *gg1* means “Prostatectomy Gleason Score 1”, *svi* means “Seminal Vesical Invasion”, *ln* is “Lymph Node Status”, *prepsa* is “Preoperative PSA”.

In the next table, we compare sensitivity and specificity of the decision rules obtained on one set and tested on another set. For comparison we use SVRc<sup>TM</sup> method which uses some adjustment of the support vector regression method for the censored data [17].

#### Comparing Sensitivity and Specificity of Live Logic<sup>TM</sup> and SVRc<sup>TM</sup>

Datasets		Sensitivity		Specificity	
Training	Testing	LL	SVRc <sup>TM</sup>	LL	SVRc <sup>TM</sup>
Set 1	Set 2	0.79	0.5	0.77	0.87
Set 2	Set 2	0.8	0.65	0.66	0.81

As we see from this table, sensitivity of Live Logic<sup>TM</sup> is consistently higher, while the specificity is lower than that of SVRc<sup>TM</sup>. For the practical purposes, the results of the Live Logic<sup>TM</sup> are preferable, since low sensitivity means that large cohort of the high risk patients will not get a necessary treatment. The results of Live Logic<sup>TM</sup> are preferable from the *min(sensitivity, specificity)* criterion as well.

## 5 Conclusions

The main contribution of this paper is a general approach to the problem of learning under uncertainty, where the resulting rules need to be understandable. The learning and decision making methods for this problem are presented. The learning method finds all strongest and sufficiently consistent decision rules. The decision making method is based on multidimensional partial ordering and calculation of U-score. We demonstrate the advantages of the approach in application to the problem of prognosis of clinical failure for patients with prostate cancer after prostatectomy.

## References

1. Agraval, R., Mannila, H., Srikant, R., Toivonen, H., Verkamo, A.: Fast discovery of association rules. *Advances in Knowledge Discovery and data Mining*. AAAI/MIT Press, Cambridge, MA. (1995)
2. Bazan, J., Nguyen, H.S., Nguyen, S.H., Synak, P., Wróblewski, J.: Rough set algorithms in classification problems. In: Polkowski, L., Lin, T. Y., Tsumoto, S. (eds.): *Rough Set Methods and Applications: New Developments in Knowledge Discovery in Information Systems*. 56 Physica-Verlag, Heidelberg, Germany (2000) 49 – 88.
3. Borgelt, C. and Kruse, R.: Induction of Association Rules: Apriori Implementation. 15th Conference on Computational Statistics (Compstat 2002, Berlin, Germany) Physica Verlag, Heidelberg, Germany (2002).

4. Klose, A. , Nurnberger, A. and Nauck, D.: Some Approaches to Improve the Interpretability of Neuro-Fuzzy Classifiers. Proc. 6th European Congress on Intelligent Techniques and Soft Computing (EUFIT'98), Aachen (1998) 629 – 633.
5. Pawlak, Z.: Some Issues on Rough Sets. Transactions on Rough Sets I. Lecture notes in computer science 3100 Springer-Verlag, Berlin Heidelberg New York (2004) 375 – 391.
6. Quinlan, J.: Induction of decision trees. *Machine Learning*, 1 (1986) 81 – 106.
7. Quinlan, J.: Learning logical definitions from relations. *Machine Learning* 5(3) (1990).
8. Sapir, M.: Constructing plausible hypothesis for diverse attributes. *Automat. Remote control*, No 11 (1993) 134 – 142.
9. Sapir, M., Sherman, S.: A toolkit for automated search for the most general and easily interpretable hypotheses in first order logic systems. *International Conference on Integration of Knowledge Intensive Multi-Agent Systems. KIMAS'03* (2003) 318 – 323.
10. Sapir, M.: Formalization of Induction Logic in Biomedical Research. 4th International Symposium on Robotics and Automation ISRA'2004: 1 – 8.
11. Skowron, A., Stepaniuk, J. Tolerance approximation spaces. *Fundamenta Informaticae*. 27 (1996) 245 – 253.
12. Stefanowski, J.: On rough set based approaches to induction of decision rules. In: Skowron, A., Polkowski L. (eds.): *Rough sets in knowledge discovery*. 1, Physica Verlag, Heidelberg (1998) 500 – 529.
13. Triantaphyllou, E., Kovalerchuk, B., Deshpande, A.: Some recent developments of using logical analysis for inferring a Boolean function with few clauses. In: Barr, R., Helgason, R., Kennington, L. (eds.): *Interfaces in Computer Science and Operations Research Series*, 7, Kluwer (1997) 215 – 236.
14. Wittkowski, K.M.: An extension to Wittkowski. *J Am Statist Assoc* (1992) 87 – 258.
15. Wittkowski, K.M., Lee, E, Nussbaum, R., Chamian, F.N., Krueger, J.G.: Combining several ordinal measures in clinical studies. *Stat Med*, 23 (2004) 1579 – 1592.
16. Wittkowski, K.M.: Novel Methods for Multivariate Ordinal Data applied to Genetic Diplotypes, Genomic Pathways, Risk Profiles, and Pattern Similarity. *Computing Science and Statistics* 35 (2003) 626 – 46
17. Yan, L., Verbel, D., Saidi, O.: Predicting prostate cancer recurrence via maximizing the concordance index. *ACM SIGKDD Conference Proceedings* (2004)
18. Zaluski, J., Szoszkiewicz, R., Krisinski, J., Stefanowski, J.: Rough Set Theory and Decision Rules in Data Analysis of Breast Cancer Patients. Transactions on Rough Sets I. Lecture notes in computer science Vol 3100 Springer-Verlag, Berlin Heidelberg New York (2004) 1 – 58.
19. Ziarko, W.: Variable precision rough sets model. *Journal of Computer and Systems Sciences*. 46 no. 1 (1993) 39 – 59.