On improving FOIL Algorithm

Patricia Jiménez, J.L. Arjona, J.L. Álvarez
The Distributed Group - Onuba
Escuela Politécnica Superior.
Crta. Huelva - La Rábida. Palos de la Frontera 21071
Email: {patricia.jimenez, arjona, alvarez}@dti.uhu.es

***Abstract*—FOIL is an Inductive Logic Programming Algorithm to discover first order rules to explain the patterns involved in a domain of knowledge. Domains as Information Retrieval or Information Extraction are handicaps for FOIL due to the huge amount of information it needs manage to devise the rules. Current solutions to problems in these domains are restricted to devising ad hoc domain dependent inductive algorithms that use a less-expressive formalism to code rules.**

We work on optimising FOIL learning process to deal with such complex domain problems while retaining expressiveness. Our hypothesis is that changing the information gain scoring function, used by FOIL to decide how rules are learnt, can reduce the number of steps the algorithm performs. We have analysed 15 scoring functions, normalised them into a common notation and checked a test in which they are computed. The learning process will be evaluated according to its efficiency, and the quality of the rules according to their precision, recall, complexity and specificity. The results reinforce our hypothesis, demonstrating that replacing the information gain can optimise both the FOIL algorithm execution and the learnt rules.

***Index Terms*—FOIL, ILP, scoring functions**

I. Introduction

Machine learning systems aim to automatically learn to recognize complex patterns based on data from some background knowledge and to make intelligent decisions on new data. Many of these systems has their focus on Inductive Logic Programming (ILP), a subfield of machine learning which investigates the construction of first-order logic rules. This kind of systems include FOIL [20], GOLEM [13], PROGOL [12] with Shapiro's program MIS as one of their early predecessors [11].

However, a major problem of ILP systems arises when the set of training data is too large. Obviously, it happens in domains as information retrieval or information extraction; the process of learning the hypothesis that best fits the available knowledge becomes inefficient, or the set of learnt rules has low recall. An alternative choice is propositional logic systems since they result more practical for efficiency reasons, but they produce rules quite less expressive and consequently, they are restricted to be applied to simpler domains problems.

We wish to use FOIL algorithm to deal with such complex domain problems and we try to improve it in order make the learning process more efficient. To carry out this task, we have studied 15 different scoring functions coming from statistics, machine learning, and data mining literature and we propose to use the best one instead of the information gain, which is employed by the original FOIL to select the best candidate rules. We have proved that some of these scoring functions perform better since they find out best rules or find them out faster.

We bet on FOIL algorithm because of the expressiveness of first-order logic rules it is able to devise. First-order rules allow the system to learn relational and recursive concepts that cannot be represented in the attribute-value format assumed by most machine learning algorithms. Furthermore, there have been many authors who have tried to improve FOIL developing successors systems like FOCL [25], AUDREYII [18], mFOIL [17], HYDRA [16], FOSSIL [15], FFOIL [14], FZ-FOIL [27] and FOIDL [10]. Some of these systems proposed to use likelihood ratio, correlation criterion, estimated accuracy and interest measures as alternatives to the information gain. In many cases, their results were better than FOIL ones but restricted to domain dependent tasks. So that the problem has not been solved yet. However, these systems and their results suggest that FOIL can be optimised in many ways.

The paper is organised as follows: first, we introduce an overview of FOIL algorithm and we propose to use new scoring functions in order to solve the mentioned problems. Next section a common notation and a set of scoring functions are explained. Then, we perform a test we show the results obtained. Conclusion section discuss these results and gives some tips for future research.

II. FOIL

In first order learning, training data comprises a target predicate, which is defined by a set of ground literals labelled as positive, if they satisfy the target predicate, and as negative, otherwise. Furthermore, a set of support predicates is defined either extensionally, similarly to what was previously made with the target predicate or intensionally, by means of a set of rules. The goal is to learn a set of logic rules that explain the target predicate in terms of itself and the support predicates.

FOIL is an algorithm of machine learning that induces first order rules. It is based on sequential covering algorithm and uses separate-and-conquer method, attempting to learn one rule at a time to incrementally grow the final set of rules. In order to learn each rule, it follows a top-down approach, starting with the most general rule header, and guided by a greedy search, is adding new unground literals to the rule, until it does not satisfy any negative ground literal belonging to the target predicate. The set of rules is ready when all positive ground literals belonging to the target predicate are satisfied.

Each learnt rule is of the form $H \leftarrow B$ where H is the head and B is the body of the rule. H is an unground literal of the form $R(X_0, X_1, \ldots, X_n)$ where R is the target predicate and $X_0, X_1, \ldots, X_n$ are the variables. Similarly, B is a set of unground literals, for instance $P_1(X_0, X_2), P_2(X_3, X_1), \ldots$, where P_i represents any predicate defined in the knowledge base and $X_0, X_1, \ldots, X_n$ are the variables of the predicate P_i.

To add a new literal to the current rule, a list of unground literals is generated. Each one is added to the current rule giving rise to a new candidate rule. The candidate rules are weighted based on information gain scoring function. It measures how benefits replacing the current rule with a specific candidate rule. To compute this score, the information gain relies on the number of positive and negative ground literals that are satisfied before and after this replacement. The candidate rule with higher score is selected to keep growing.

Let tp be the number of positive ground literals and fp the number of negative ground literals that are satisfied by the current rule. The information conveyed by the knowledge that a ground literal satisfied by the current rule is positive is given by

$$I(H \leftarrow B) = -log \frac{tp}{tp + fp} \tag{1}$$

Similarly, for each new candidate rule $I_k(H \leftarrow B')$ built from adding a new literal generated L_k to the current rule. Being t the number of positive ground literals satisfied by both the current and a new candidate rule, the information gain has a straightforward interpretation in terms of information theory and is given by the formula:

$$I(H \leftarrow B') = t \times (I_k(H \leftarrow B') - I(H \leftarrow B)) \tag{2}$$

In FZFOIL [27] some deficiencies in the information gain have been identified. Presumably, it may be due to the information gain only take into account the number of positive and negative ground literals a new candidate rule satisfies, forgetting other parameters as the number of positive and negative ground literals this candidate rule discards.

For the purpose of improving the learning process, we analyse other scoring functions from the literature trying to solve the information gain problem stated. They weigh the candidate rules up according to the existing correlation between it and the current rule. Therefore, the gain of these scoring functions will measure the amount of correlation gained if the current rule is replaced with a new specific candidate rule.

III. Comparison Framework

The proposed scoring functions will be defined in terms of the well-known contingency table. For evaluating any first order candidate rule $H \leftarrow B'$, we rely on a contingency table as the one below in I.

Actual class are those positive and negative ground literals satisfied by the head of a candidate rule. Predicted class are those ground literals satisfied by the body of the candidate rule being analysed. Thus, tp denotes the number of positive ground literals that are satisfied by the head and the body of the candidate rule and fp denotes the number of negative ground literals satisfied by the head and the body of the candidate rule. Similarly, fn denotes the number of positive ground literals satisfied by the head but not by the body of the candidate rule and tn denotes the number of negative ground literals satisfied by the head but not by its body. N is the total number of positive and negative ground literals.

We have implemented and evaluated a subset of scoring functions proposed in [24] as objective measures and in [19] as measures for predictive and descriptive induction. Furthermore, we have selected other scoring functions for being quite traditional. The set of scoring functions adapted to our notation are showed in table II.

A summary description for each scoring function:

		Predicted Class		
		B	$\neg B$	
Actual Class	H	tp true positives	fn false negatives	
	¬ H	fp false positives	tn true negatives	
				N

TABLE I
Contingency Table

	Scoring Function	Formula
1	Coverage	$\frac{tp+fp}{N}$
2	Laplace Accuracy	$\frac{tp+1}{tp+fp+2}$
3	Leverage	$\frac{tp \cdot tn - fn \cdot fp}{N^2}$
4	ϕ-coefficient	$\frac{tp \cdot tn - fn \cdot fp}{\sqrt{(tp+fn) \cdot (tp+fp) \cdot (fp+tn) \cdot (fn+tn)}}$
5	Support	$\frac{tp}{N}$
6	Confidence	$\frac{tp}{tp+fp}$
7	Satisfaction	$\frac{tp \cdot tn - fp \cdot fn}{(tp+fp) \cdot (tn+fp)}$
8	Confirmation	$\frac{(tp \cdot tn - fp \cdot fn)^2}{N^2 \cdot (tp+fp) \cdot (tn+fp)}$
9	F-measure	$2 \cdot \frac{tp}{2 \cdot tp+fn+fp}$
10	kappa (κ)	$\frac{2 \cdot (tp \cdot tn - fp \cdot fn)}{N^2 - (tp+fn) \cdot (tp+fp) - (fp+tn) \cdot (tn+fn)}$
11	Odds-ratio	$\frac{tp \cdot tn}{fp \cdot fn}$
12	Yule's Q	$\frac{tp \cdot tn - fp \cdot fn}{tp \cdot tn + fp \cdot fn}$
13	Lift (Interest)	$\frac{N \cdot tp}{(tp+fp) \cdot (tp+fn)}$
14	Collective Strength	$\frac{tp+tn}{(tp+fp) \cdot (tp+fn) + (fn+tn) \cdot (fp+tn)} \times \frac{N^2 - (tp+fp) \cdot (tp+fn) - (fn+tn) \cdot (fp+tn)}{N - tp - tn}$
15	Jaccard(ζ)	$\frac{tp}{tp+fp+fn}$

TABLE II
List of Scoring functions

- Coverage is a measure of generality of a rule. If a rule characterizes more information in the data set, it tends to be more interesting.
- Laplace Accuracy [23] is an approximate measure to estimate the expected accuracy directly. General rules tend to be favored.
- Leverage [22] is one of the most frequently measure used in the evaluation of rules. It is also known as Leverage. It trades off generality and relative accuracy.
- $\phi - coefficient$ [21] is a statistical measure analogous to Pearson's product-moment correlation coefficient. It measures the degree of association between two binary variables (e.g., two rules). It is closely related to the χ^2 statistic since $\phi^2 = \frac{\chi^2}{N}$.
- Support [9] is a measure known from association rule learning, also called frequency. It is used for specifying if a rule is observed frequent enough in a data set.
- Confidence is also known as confidence [9]. It is related to the reliability. A rule is reliable if its predictions are highly accurate.
- Satisfaction [19] is similar to confidence e.g., $Sat(H \leftarrow B) = 1$ if $Confidence(H \leftarrow B) = 1$, but, unlike Confidence, it takes the entire contingency table into account and is thus more suited towards knowledge discovery.
- Confirmation [8] is defined in terms of a modified χ^2 statistic. It trades off satisfaction and Leverage measures.
- F-measure is other statistic measure of a test accuracy. It considers both the precision and the recall of the test to compute the score. We compute F1-score which is the harmonic mean of precision and recall.
- kappa (κ) [7] captures the degree of agreement between a pair of variables (e.g., the head and the body of a candidate rule). If both variable are highly agree with each other, then the values for κ will result higher.
- Odds-ratio [5] represents the strength of association or non-independence between two binary data values. Unlike other measures of association for paired binary data, the comparison between the two variables is symmetrical.
- Yule's Q coefficient [6] is a normalized variant of the odds ratio.
- Lift (Interest) [3] is used quite extensively in data mining for measuring deviation from statistical independence. It gives an indication of rule significance or interest.
- Collective Strength [4] is other measure of correlation variant of Lift measure. It compares between actual and expected values.
- Jaccard(ζ) [2] is a statistic used for comparing the similarity and diversity of sample sets. It is used extensively in information retrieval to measure the similarity between documents. We measure the similarity between two rules.

We have defined some measures to help us decide which scoring function is more promising. The set of measures taking into account are:

1) ***Efficiency***. This is defined as the amount of useful work in relation to time and resources used. The resources are memory and space required.
2) ***Precision***. It measures the number of ground literals satisfied by the set of rules correctly against all ground literals satisfied, although so far, we only search for 100% accuracy rules, i.e., we do not allow rules that satisfy any negative ground literal.
3) ***Recall***. It determines if a set of rules is complete, i.e., if it satisfies all positive ground literals belonging to the target predicate.
4) ***Complexity*** of the induced set of rules. It is computed in terms of bits from Minimum Description Length Principle [26].
5) ***Specificity/Generality*** of the induced set of rules. It is general if they are only a few single rules that satisfy most of positive ground literals belonging to the target predicate. They will be too specificity when they are too large and only satisfy a few number of positive ground literals. We prefer general rules rather specific rules.

However, there are measures that can not be estimated objectively because they depend on other measures which we call secondary measures. For instance, secondary measures that may affect efficiency directly may be the number of backtracking performed or the number of candidate rules that were evaluated, which is one of the most expensive step in the algorithm. The number of different predicates used in the induced set of rules could also affect the efficiency of the learning process because it gives an idea about how well the knowledge base was built and therefore, how useful the support predicates defined are.

Secondary measures that may affect to generality/specificity of a rule are the number of the variables used in the set of rules and the deep of the learnt rules measured in terms of the number of unground literals in each rule. If these numbers are small it will mean that the rules are quite general, which is a desirable property.

Intuitively, the total number of induced rules will affect both, efficiency and generality/specificity. Fewer number of rules will make the process more efficient and the final set of rules more general. Note the latter is true as long as the set of rules has a good recall.

IV. On Going Work

The example tested was first showed in [27] and it tries to explain when somebody is sick. To carry out this task we rely on a set of seventeen individuals among which eight are sick, and the rest are not. The target predicate will be sick(X_0), which means the individual X_0 is sick. The support predicates defined to induce a set of rules that explain the target predicate sick(X_0) are:

- bearded(X_i), which means the individual X_i is bearded.
- smoker(X_i), which means the individual X_i is a smoker.
- father(X_i, X_j), which means X_i is X_j's father.
- boss(X_i, X_j), which means X_i is X_j's boss.

In the knowledge base, the positive ground literals belonging to the target predicate are all individuals who are sick. The rest are the negative ground literals and they can be defined explicitly or to be induced by Closed World Assumption. Similarly, we have to define the positive ground literals that satisfy each support predicate but there is no need to define the negative ones explicitly.

Scoring Function	P	R	C	T
Information Gain	1	1	23.75	6257
Coverage	1	0.5	12.17	138275
Laplace Accuracy	1	1	30.66	22317
Leverage	1	1	22.34	5400
Φ-coefficient	1	1	37.09	34200
Support	1	1	26.77	43759
Confidence	1	1	23.08	19017
Satisfaction	1	1	20.17	3512
Confirmation	1	1	20.17	3705
F-Measure	0	0	0	58147
Kappa	1	1	22.34	4983
Odds ratio	1	1	25.92	10470
Yule's Q	1	1	56.85	25701
Lift	1	1	23.08	17249
Collective Strength	1	1	22.34	4745
Jaccard	0	0	0	57480

TABLE III
RESULTS OBTAINED

Our knowledge base would be a Prolog program and the set of learnt rules for this example is showed in IV[1]. The results obtained for the evaluation measures explained previously, are presented in table IV[2]

V. CONCLUSIONS

As well as in other analysis of measures or scoring functions, we can not conclude saying there is a scoring function consistently better than the rest in all applications domains, although we have found some scoring functions that perform better than information gain in our running example.

All rules induced are 100% accurate because we do not allow rules that satisfy any negative ground literal. Therefore the goal is to get the most reduced set of learnt rules with the largest recall in the shortest time possible. The final set of rules will depend largely on the scoring function used. If it is not good enough FOIL might not learn a complete set of rules (i.e, the set do not have a recall of 100%). We wish a balance among precision, recall, efficiency, complexity and specificity/generality to decide which is the most promising set of rules obtained.

Taking of these factors into account, we consider that Collective Strength and Leverage scoring function performed better maintaining the full recall, because they took shorter time to get the rules. Furthermore the set of rules were more general and less complex. Satisfaction, Confirmation and kappa scoring functions are even better than the previous one. They took less time to find out a set of rules and, although Satisfaction and Confirmation scoring functions had one rule more, both had a complexity still lower.

Support scoring function is quite similar to the information gain. It spent more time evaluating many candidate rules and the rules are more complex but more general. It is difficult

Rules
Information Gain
sick(X_0)← smoker(X_0).
sick(X_0)← boss(X_1,X_0), sick(X_1), ¬father(X_0, X_2).
sick(X_0)← boss(X_1,X_0), sick(X_1), father(X_2, X_0), sick(X_2).
Coverage
sick(X_0)← smoker(X_0).
sick(X_0)← boss(X_0,X_1), father(X_0, X_2).
Laplace Accuracy
sick(X_0)← father(X_1,X_0), bearded(X_1), smoker(X_0).
sick(X_0)← boss(X_1,X_0), sick(X_1), father(X_2,X_0), sick(X_2).
sick(X_0)← ¬father(X_0,X_1), boss(X_1,X_2), boss(X_2,X_3), boss(X_4,X_1), father(X_5,X_2), $X_0 \neq X_3$.
Leverage
sick(X_0)← father(X_1,X_0), bearded(X_1), boss(X_0,X_2).
sick(X_0)← ¬father(X_0,X_1), boss(X_1,X_2), ¬boss(X_2,X_0).
ϕ-coefficient
sick(X_0)← ¬boss(X_0,X_1), boss(X_1,X_2), ¬boss(X_2,X_0).
sick(X_0)← father(X_1,X_0), boss(X_0,X_2), ¬father(X_0,X_3).
sick(X_0)← father(X_1,X_0), bearded(X_1), boss(X_0,X_2).
Support
sick(X_0)← boss(X_0,X_1), smoker(X_0).
sick(X_0)← father(X_1,X_0), boss(X_1,X_2), $X_1 \neq X_2$, boss(X_1, X_3), sick(X_1).
Confidence
sick(X_0)← smoker(X_0).
sick(X_0)← boss(X_1,X_0), sick(X_1), father(X_2,X_0), sick(X_2).
sick(X_0)← ¬father(X_0,X_1), boss(X_1,X_2), boss(X_2,X_3), boss(X_4,X_1), father(X_5,X_2), $X_0 \neq X_3$.
Satisfaction
sick(X_0)← smoker(X_0).
sick(X_0)← ¬father(X_0,X_1), ¬bearded(X_0).
sick(X_0)← boss(X_1,X_0), sick(X_1), bearded(X_1).
Confirmation
sick(X_0)← smoker(X_0).
sick(X_0)← ¬father(X_0,X_1), ¬bearded(X_0).
sick(X_0)← boss(X_1, X_0), sick(X_1), bearded(X_1).
F-measure
kappa (κ)
sick(X_0)← father(X_1,X_0), bearded(X_1), boss(X_0,X_2).
sick(X_0)← ¬father(X_0,X_1), boss(X_1,X_2), ¬boss(X_2,X_0).
Odds-ratio
sick(X_0)← smoker(X_0).
sick(X_0)← boss(X_1,X_0), boss(X_0,X_2), sick(X_2), sick(X_1).
sick(X_0)← ¬boss(X_0,X_1), boss(X_1,X_2), ¬bearded(X_0).
Yule's Q
sick(X_0)← ¬bearded(X_0), boss(X_1,X_0), sick(X_1).
sick(X_0)← boss(X_1,X_0), ¬father(X_1,X_2).
sick(X_0)← smoker(X_0).
sick(X_0)← father(X_1,X_0), ¬boss(X_2,X_0).
sick(X_0)← boss(X_0,X_1), boss(X_2,X_0), smoker(X_1).
sick(X_0)← boss(X_0,X_1), boss(X_2,X_0), ¬father(X_0,X_3).
Lift (Interest)
sick(X_0)← smoker(X_0).
sick(X_0)← boss(X_1,X_0), sick(X_1), father(X_2,X_0), sick(X_2).
sick(X_0)← ¬father(X_0,X_1), boss(X_1,X_2), boss(X_2,X_3), boss(X_4,X_1), father(X_5,X_2), $X_0 \neq X_3$.
Collective Strength
sick(X_0)← father(X_1,X_0), bearded(X_1), boss(X_0,X_2).
sick(X_0)← ¬father(X_0,X_1), boss(X_1,X_2), ¬boss(X_2,X_0).
Jaccard(ζ)

TABLE IV
SET OF LEARNT RULES

[1]Note that FOIL relies on predefined predicates which are of the form $X_i = X_j$ or $X_i = c_j$, where X_i and X_j are variables and c_j is a constant (e.g., c_j can be any specific individual).

[2]where: **P**: Precision, **R**: Recall, **C**: complexity (bits), **T**: elapsed time (milliseconds)

to decide which is better. As efficient is a relevant factor, we would opt for information gain. However, Laplace accuracy, ϕ-coefficient, Confidence and Lift behaved worse than information gain, wasting time searching for more specific rules. Note Lift and Laplace accuracy obtained similar results.

Finally, Coverage, FMeasure and Jaccard scoring functions did not find out a set of rules that satisfy all positive ground literals defined in the knowledge base. The last two were unable to find out a single rule and they evaluated a huge amount of candidate rules caused among other factors, by the backtracking performed. Note FMeasure and Jaccard scoring function has very similar formulae so they behaved in an identical way.

We can conclude saying that scoring functions like Leverage, Confirmation, Satisfaction, Kappa and Collective Strength are more promising than the information gain. Anyway, establishing a ranking among the proposed scoring functions is a hard task because we have not identified which are the most relevant evaluation measures yet and there could be more interesting measures to check. The relevance of each one will depend on the domain being studied although we opt for those measures that affect the efficiency directly as more relevant.

The application of this kind of systems is usually better than with any other known approach, so it needs to find more training sets to be tested, to define more additional measures if that would be necessary and to perform an exhaustive evaluation to get a reliable ranking. All this in order to apply FOIL satisfactorily to domains with a huge amount of information.

Acknowledgment

This research is supported by the European Commission (FEDER), the Spanish and the Andalusian R&D&I programmes (grants TIN2007-64119, P07-TIC-2602, P08-TIC-4100, and TIN2008-04718-E).

References

[1] H. Kopka and P. W. Daly, *A Guide to LaTeX*, 3rd ed. Harlow, England: Addison-Wesley, 1999.
[2] C.J. Van Rijsbergen, *Information Retrieval*, Butterworth, 1979.
[3] S. Brin, R. Motwani, C. Silverstein, *Beyond Market Baskets: Generalizing Association Rules to Correlations*, SIGMOD Conference, 1997.
[4] C.C. Aggarwal, P.S. Yu, *A New Framework For Itemset Generation*, Symposium on Principles of Database Systems, 1998.
[5] F. Mosteller, *Association and Estimation in Contingency Tables*, Journal of the American Statistical Association, 1968.
[6] G.U. Yule, *On the Association of Attributes in Statistics*, Philosophical Transactions of the R.Society, Ser. A, 1900.
[7] J. Cohen, *A Coefficient of Agreement for Nominal Scales*, Educational and Psychological Measurement, 1960.
[8] P. Flach, N. Lachiche, *A First-Order Approach to Unsupervised Learning*, 1999.
[9] R. Agrawal, T. Imielinski, A.N. Swami, *Mining Association Rules between Sets of Items in Large Databases*, SIGMOD Conference, 1993.
[10] R.J. Mooney, M.E. Califf, *Induction of First-Order Decision Lists: Results on Learning the Past Tense of English Verbs*, JAIR, 1995.
[11] E.Y. Shapiro, *Algorithmic Program Debudding*, Cambridge, MA: MIT Press, 1983.
[12] A. Srinivasan,S.H. Muggleton, R.D. King, M.J.E Sternberg, *Mutagenesis: ILP experiments in a non-determinate biological domain*, Proceedings of the 4th International Workshop on Inductive Logic Programming, 1994.
[13] S. Muggleton, C. Feng, *Efficient Induction of Logic Programs*, Algorithmic Learning Theory, 1990.
[14] J.R. Quinlan, *Learning First-Order Definitions of Functions*, JAIR, 1996.
[15] J. Fürnkranz, *FOSSIL: A Robust Relational Learner*, European Conference on Machine Learning, 1994.
[16] K.M. Ali, M.J. Pazzani, *HYDRA: A Noise-tolerant Relational Concept Learning Algorithm*, IJCAI, 1993.
[17] N. Lavrac, S. Dzeroski, *Inductive Logic Programming: Techniques and Applications*, Ellis Horwood, 1994.
[18] J. Wogulis, M.J. Pazzani, *A Methodology for Evaluating Theory Revision Systems: Results with Audrey II*, IJCAI, 1993.
[19] N. Lavrac, P.A. Flach, B. Zupan, *Rule Evaluation Measures: A Unifying View*, International Workshop on Inductive Logic Programming. 174–185 (1999)
[20] J.R. Quinlan, *FOIL: A Midterm Report*, European Conference on Machine Learning, 1993.
[21] A. Agresti, *Categorical Data Analysis*, Wiley-Interscience, 1990.
[22] G. Piatetsky-Shapiro, *Discovery, Analysis, and Presentation of Strong Rules*, 1991.
[23] P. Clark, R. Boswell, *Rule Induction with CN2: Some Recent Improvements*, EWSL, 1991.
[24] P. Tan, V. Kumar, J. Srivastava, *Selecting the right objective measure for association analysis*, Inf. Syst., 2004.
[25] M.J. Pazzani, C. Brunk, G. Silverstein, *A Knowledge-intensive Approach to Learning Relational Concepts*, ICML, 1991.
[26] J. Rissanen, *Modeling by Shortest Data Description*, Automatica, 1978.
[27] A.J GÓMEZ, G. FERNANDEZ, *Inducción de definiciones lógicas a partir de relaciones: mejoras en los heurísticos del sistema FOIL*, PRODE, 1992.