Soft Comput (2009) 13:959–977
DOI 10.1007/s00500-008-0392-y

ORIGINAL PAPER

A study of statistical techniques and performance measures for genetics-based machine learning: accuracy and interpretability

S. García · A. Fernández · J. Luengo ·
F. Herrera

Published online: 20 December 2008

Abstract The experimental analysis on the performance of a proposed method is a crucial and necessary task to carry out in a research. This paper is focused on the statistical analysis of the results in the field of genetics-based machine Learning. It presents a study involving a set of techniques which can be used for doing a rigorous comparison among algorithms, in terms of obtaining successful classification models. Two accuracy measures for multi-class problems have been employed: classification rate and Cohen's kappa. Furthermore, two interpretability measures have been employed: size of the rule set and number of antecedents. We have studied whether the samples of results obtained by genetics-based classifiers, using the performance measures cited above, check the necessary conditions for being analysed by means of parametrical tests. The results obtained state that the fulfillment of these conditions are problem-dependent and indefinite, which supports the use of non-parametric statistics in the experimental analysis. In addition, non-parametric tests can be satisfactorily employed for comparing generic classifiers over various data-sets considering any performance measure. According to these facts, we propose the use of the most powerful non-parametric statistical tests to carry out multiple comparisons. However, the statistical analysis conducted on interpretability must be carefully considered.

Keywords Genetics-based machine learning · Genetic algorithms · Statistical tests · Non-parametric tests · Cohen's kappa · Interpretability · Classification

1 Introduction

In general terms, the classification problem can be covered by numerous techniques and algorithms, which belong to different paradigms of machine learning (ML). The new developed methods for ML must be analysed against previous approaches following a rigorous criterion, since in any empirical comparison the results are dependent on the choice of the cases for studying, the configuration of the experimentation and the measurements of performance. Nowadays, the statistical validation of published results is a necessity in order to establish a certain conclusion on an experimental analysis (Demšar 2006).

Evolutionary rule-based systems (Freitas 2002) is a kind of Gen-etics-Based Machine Learning (GBML) that uses sets of rules as knowledge representation (Grefenstette 1993). Many approaches have been proposed in GBMLs based on offering some advantages with respect to other existing ML techniques; such as the production of interpretable models, no assumption of prior relationships among attributes and the possibility of obtaining compact and precise rule sets. Some examples of proposed GBMLs are: GABIL (De Jong et al. 1993), SIA (Venturini 1993), XCS (Wilson 1995), DOGMA and JoinGA (Hekanaho 1998), G-Net (Anglano and Botta 2002), UCS (Bernadó-

S. García (✉)
Department of Computer Science,
University of Jaén, 23071 Jaén, Spain
e-mail: sglopez@ujaen.es

A. Fernández · J. Luengo · F. Herrera
Department of Computer Science and Artificial Intelligence,
University of Granada, 18071 Granada, Spain
e-mail: alberto@decsai.ugr.es

J. Luengo
e-mail: julianlm@decsai.ugr.es

F. Herrera
e-mail: herrera@decsai.ugr.es

Mansilla and Garrell 2003), GASSIST (Bacardit 2004), OCEC (Jiao et al. 2006) and HIDER (Aguilar-Ruiz et al. 2000).

Recently, statistical analysis is highly demanded in any research work and thus, we can find recent studies that propose some methods for conducting comparisons among various approaches (Demšar 2006; Markatou et al. 2005). Statistics allows us to determine whether the obtained results are significant with respect to the choices taken and whether the conclusions achieved are supported by the experimentation that we have carried out. On the other hand, the performance of classifiers is not only given by their classification rate and there is a growing interest in proposing or adapting new accuracy measures (Ben-David 2007; Drummond and Holte 2006). Most of the accuracy measures are proposed for two-class problems and their adaptation to multi-class problems is not intuitive (Landgrebe and Duin 2008). Only two accuracy measures have been used for multi-class problems with successful results: the classical classification rate and the Cohen's kappa measure. The main difference between them is the scoring of the true classifications rates. Classification rate scores all the successes over all classes, whereas Cohen's kappa scores the successes independently for each class and aggregates them. The second way of scoring is less sensitive to randomness caused by different number of examples in each class, which causes a bias in the learner towards the obtention of data-dependent models.

In GBMLs, the interpretability of the rule sets obtained is very important, due to the fact that very large sets of rules or very complex rules are rather lacking in interest.

The use of parametric statistical techniques over the sample of results is only adequate when they fulfill three necessary conditions: independency, normality and homoscedasticity (Sheskin 2006, Zar 1999). This paper shows that these conditions are usually not verified when analysing GBML algorithms. Under these assumptions, a statistical analysis conducted by means of parametric tests may not be safe with respect to the achieved results and hence, the conclusions about an experimental study could be incorrect.

In this paper, we are interested in the study of the most appropriate statistical techniques and performance measures for analysing the experimentation of GBML algorithms. We mainly focus on five topics:

- To study the fulfillment of the necessary conditions for a safe usage of parametric tests.
- To emphasize the existing differences between a pairwise comparison statistical procedure and a multiple comparison statistical procedure, pointing out the advantages of using the second ones.
- To notice that the use of different performance measures may yield different conclusions in the statistical study, due to the fact that they have different purposes in the evaluation of the algorithms.
- To show the generality for comparing GBML algorithm with other ML approaches, in spite of the non-stochasticity of the latter methods. For this purpose, we will include the CN2 algorithm (Clark and Niblett 1989) when conducting the non-parametric statistical analysis.
- Making an analysis based on interpretability is not trivial. We give some concerns in this paper and we justify why the available interpretability metrics have to be treated with "a grain of salt".

In order to do that, the paper is organized as follows. Section 2 presents the GBML algorithms used. The description of the multi-class performance measures together with the experimental framework and the results obtained are given in Sect. 3. We introduce the statistical analysis and we carry out the study of the necessary conditions for a safe use of parametric tests in Sect. 4. Section 5 describes the procedures for doing pairwise comparisons with non-parametric statistics. In the case of multiple comparisons tests, we present and use them in Sect. 6. We present the analysis based on interpretability and we give our concerns in Sect. 7. Finally, the conclusions are summarized in Sect. 8. An appendix is included containing an extended description of the GBML methods used in our study.

2 Genetics-based machine learning algorithms for classification

In this paper we use GBML methods in order to perform classification tasks. Specifically, we have chosen four Genetics Interval Rule Based Algorithms, such as Pittsburgh Genetics Interval Rule Learning Algorithm (Pitts-GIRLA), XCS, Genetic Algorithm based Classifier System (GASSIST-ADI) and Hierarchical Decision Rules (HIDER). These algorithms are provided by the KEEL software (Alcalá-Fdez et al. 2009), which includes updated versions of these GBML methods.

In the following we will give a brief description of the different approaches that we have employed in our work. A wider explanation about the methods exposed here can be found in the appendix of this work.

1. *Pitts-GIRLA Algorithm.*
   The Pittsburgh Genetic Interval Rule Learning Algorithm (Pitts-GIRLA) (Corcoran and Sen 1994) is a GBML method which makes use of the Pittsburgh approach in order to perform a classification task. The

main structure of this algorithm is a generational Genetic Algorithm (GA) in which, for each generation, the steps of selection, crossover, mutation and replacement are applied.

All chromosomes are initialized at random, with values between the range of each variable. The selection mechanism consists in choosing two individuals at random among all the chromosomes of the population.

The fitness of a particular chromosome is simply the percentage of instances correctly classified by the chromosome's rule set (classification rate).

The best chromosome of the population is always maintained as in the elitist scheme.

2. *XCS Algorithm.*

   XCS (Wilson 1995) is a Learning Classifier System (LCS) (Sigaud and Wilson 2007) that evolves online a set of rules that describe the feature space accurately. It inherits part of its behavior from ZCS (Wilson 1994), and differs in several ways from more traditional LCSs. Firstly, the classifier fitness is based on the payoff prediction instead of the prediction itself. Secondly, XCS has no message list. Finally, the GA is applied over niches instead of the whole population. The set of rules has a fixed maximum size N and it is initially built by generalizing some of the input examples.

3. *GASSIST Algorithm.*

   Genetic Algorithms based claSSIfier sySTem (GASSIST) (Bacardit and Garrell 2007) is a Pittsburgh-style LCS originally inspired in GABIL (De Jong et al. 1993) from where it has taken the semantically correct crossover operator.

   The core of the system consists of a GA which evolve individuals formed by a set of production rules. The individuals are evaluated according to the proportion of correct classified training examples.

   In GASSIST-ADI, the representation for real-valued attributes is through Adaptive Discretization Intervals Rule Representation (Bacardit and Garrell 2003, 2004).

4. *HIDER Algorithm.*

   HIerarchical DEcision Rules (HIDER) (Aguilar-Ruiz et al. 2000), produces a hierarchical set of rules, that is, the rules are sequentially obtained and must be, therefore, tried in order until one, whose conditions are satisfied, is found.

   In order to extract the rule-list, a real-coded GA is employed in the search process. Two genes define the lower and upper bounds of the rule attribute. One rule is extracted in each iteration of the GA and all the examples covered by that rule are deleted. A parameter called "examples pruning factor" defines a percentage of examples that can remain uncovered. Thus, the termination criterion is reached when there are no more examples to cover, depending on the "examples pruning factor".

   The GA main operators are defined in the following:

   a. *Crossover*: Where the offspring takes values between the upper and lower bounds of the parents.
   b. *Mutation*: Where a small value is subtracted or added in the case of lower and upper bound respectively.
   c. Fitness Function: The fitness function considers a two-objective optimization, trying to maximize the number of correctly classified examples and to minimize the number of errors.

3 Performance measures and experimental results

In this section, we describe the accuracy measures for multi-class problems and the interpretability metrics used in this paper. Regarding the first ones, in the specialized literature we observe that most of them are designed for binary-class problems (Sokolova et al. 2006). Well-known accuracy measures for binary-class problems are: classification rate, precision, sensitivity, specificity, G-mean (Barandela et al. 2003), F-score, AUC (Huang and Ling 2005), Youden's index γ (Youden 1950) and Cohen's Kappa (Ben-David 2007).

Some of the two-class accuracy measures have been adapted for multi-class problems. For example, in a recent paper (Landgrebe and Duin 2008), the authors propose an approximating multi-class ROC analysis, which is theoretically possible but its computation is still restrictive. Only two measures are widely used because of their simplicity and successful application when the number of classes is large enough. We refer to classification rate and Cohen's kappa measures, which will be explained in Sect. 3.1. The two interpretability metrics will be described in Sect. 3.2. Finally, Sect. 3.3 presents the experimental framework of this paper and shows the average results obtained for each GBML algorithm employed.

3.1 Accuracy measures for multi-class problems

The analysis of the four GBML approaches described previously will be carried out by means of the following accuracy measures:

– *Classification rate*: is the number of successful hits relative to the total number of classifications. It is by far the most commonly used metric for assessing the performance of classifiers for years (Alpaydin 2004; Lim et al. 2000; Witten and Frank 2005).

- *Cohen's kappa*: is an alternative to *classification rate*, a method, known for decades, that compensates for random hits (Cohen 1960). Its original purpose was to measure the degree of agreement or disagreement between two people observing the same phenomenon. Cohen's kappa can be adapted to classification tasks and it is recommended to be employed because it takes random successes into consideration as a standard, in the same way as the AUC measure (Ben-David 2007). Also, it is used in some well-known software packages, such as WEKA (Witten and Frank 2005), SAS, SPSS, etc.

An easy way of computing Cohen's kappa is to make use of the resulting confusion matrix in a classification task. Specifically, the Cohen's kappa measure can be obtained using the following expression:

$$\text{kappa} = \frac{n\sum_{i=1}^{C} x_{ii} - \sum_{i=1}^{C} x_{i.}x_{.i}}{n^2 - \sum_{i=1}^{C} x_{i.}x_{.i}}, \tag{1}$$

where x_{ii} is the cell count in the main diagonal, n is the number of examples, C is the number of class values, and $x_{.i}$, $x_{i.}$ are the columns and rows total counts, respectively.

Cohen's kappa ranges from -1 (total disagreement) through 0 (random classification) to 1 (perfect agreement). Being a scalar, it is less expressive than ROC curves when applied to binary-classification. However, for multi-class problems, kappa is a very useful, yet simple, meter for measuring the accuracy of the classifier while compensating for random successes.

The main difference between classification rate and Cohen's kappa is the scoring of the correct classifications. Classification rate scores all the successes over all classes, whereas Cohen's kappa scores the successes independently for each class and aggregates them. The second way of scoring is less sensitive to randomness caused by different number of examples in each class, which causes a bias in the learner towards the obtention of data-dependent models.

3.2 Interpretability measures

The analysis of the four GBML approaches described in the paper will be carried out by means of two interpretability measures:

- *Size*: it is a measure that considers the number of rules which compose the model (see expression 2). Reducing the size of the model increases the interpretability by the user.

$$Size = n_R, \tag{2}$$

- *Number of antecedents* (*ANT*): Let R_i being a rule in the form *Cond* →*Class*, and *Cond* composed by $(Antecedent_1 \wedge Antecedent_2 \wedge \ldots \wedge Antecedent_k)$, this measure is defined as the following expression:

$$Ant(R_i) = k. \tag{3}$$

The average number of antecedents in the rule is described in the expression:

$$ANT = \frac{1}{n_R}\sum_{i=1}^{n_R} Ant(R_i). \tag{4}$$

3.3 Experimental results

We have selected 14 data-sets from UCI repository (Asuncion and Newman 2007). Table 1 summarizes the properties of these data-sets. It shows, for each data-set, the number of examples (#Ex.), the number of attributes (#Atts.) and the number of classes (#C.). In the case of presenting missing values (*cleveland* and *wisconsin*) we have removed the instances with any missing value before partitioning. We also add in the last columns some of the Pitts-GIRLA parameters (number of rules #R and number of generations #Gen) which we have made problem-dependent in order to increase the performance of the algorithm. The rest of the parameters are common for all problems and they are shown in Table 2.

We have used tenfold cross validation (10 fcv) and we have repeated 5 times the experiments using the GBML algorithms with different random seeds. Thus, we have obtained samples composed by 50 results in each of the measures considered. CN2 is a deterministic algorithm and has been run only once, obtaining 10 results per data-set.

Tables 3 and 4 show the results obtained for the GBML approaches studied in this paper and for CN2 over all

Table 1 Data-sets summary descriptions and Pitts-GIRLA problem-dependent parameters

Data-set Description				Pitts-GIRLA	
Data-set	#Ex.	#Atts.	#C.	#R	#Gen
bupa (bup)	345	6	2	30	5,000
cleveland (cle)	297	13	5	40	5,000
ecoli (eco)	336	7	8	40	5,000
glass (gla)	214	9	7	20	10,000
haberman (hab)	306	3	2	10	5,000
iris (iri)	150	4	3	20	5,000
monk-2 (mon)	432	6	2	20	5,000
new-Thyroid (new)	215	5	3	20	10,000
pima (pim)	768	8	2	10	5,000
vehicle (veh)	846	18	4	20	10,000
vowel (vow)	988	13	11	20	10,000
wine (win)	178	13	3	20	10,000
wisconsin (wis)	683	9	2	50	5,000
yeast (yea)	1484	8	10	20	10,000

Table 2 Parameter specification for the algorithms employed in the experimentation

Algorithm	Parameters
Pitts-GIRLA	Number of rules: “problem-dependent”, Number of generations: “problem-dependent”
	Population size: 61 chromosomes, Crossover Probability: 0.7, Mutation Probability: 0.5.
XCS	Number of explores = 100,000, population size = 6,400,$\alpha = 0.1$, $\beta = 0.2$, $\delta = 0.1$, $v = 10.0$, $\theta_{\mathrm{mna}} = 2$, $\theta_{\mathrm{del}} = 50.0$,$\theta_{\mathrm{sub}} = 50.0$, $\varepsilon_0 = 1$, do Action Set Subsumption = false, fitness reduction = 0.1, $p_I = 10.0$, $F_I = 0.01$, $\varepsilon_I = 0.0$, $\gamma = 0.25$, $\chi = 0.8$, $\mu = 0.04$, $\theta_{\mathrm{GA}} = 50.0$, doGASubsumption = true, type of selection = RWS, type of mutation = free, type of crossover = 2 point, $P_{\#} = 0.33$, $r_0 = 1.0$, $m_0 = 0.1$, $l_0 = 0.1$, doSpecify = false, nSpecify = 20.0 pSpecify = 0.5.
GASSIST-ADI	Threshold in hierarchical selection = 0
	Iteration of activation for rule deletion operator = 5
	Iteration of activation for hierarchical selection = 24
	Minimum number of rules before disabling the deletion operator = 12
	Minimum number of rules before disabling the size penalty operator = 4
	Number of iterations = 750, initial number of rules = 20, population size = 400
	Crossover probability = 0.6, probability of individual mutation = 0.6
	Probability of value 1 in initialization = 0.90, tournament size = 3
	Possible size in *micro-intervals* of an attribute = {4, 5, 6, 7, 8, 10, 15, 20, 25}
	Maximum number of intervals per attribute = 5, $p_{\mathrm{split}} = 0.05$, $p_{\mathrm{merge}} = 0.05$
	Probability of reinitialize begin = 0.03, probability of reinitialize end = 0
	Use MDL = true, iteration MDL = 25
	Initial theory length ratio = 0.075, weight relaxation factor = 0.90
	Class initialization method = cwinit, default class = auto
HIDER	Population size = 100, number of generations = 100, mutation probability = 0.5
	Percentage of Crossing = 80, Extreme Mutation Probability =0.05,
	Prune ExamplesFactor = 0.05, Penalty Factor = 1, Error Coefficient = 0.
CN2	Percentage of examples to cover = 95%
	Star size = 5, Use disjunct selectors = No

data-sets, considering the *classification rate* and *kappa* measures in test data, respectively. The column titled *Mean* shows the average classification rate achieved and the column titled *SD* shows the associated standard deviation. We stress the best result for each data-set and the average one in boldface.

Using the same data-sets and configuration of the algorithms, Table 5 shows the results obtained for the GBML approaches studied in this paper over all data-sets, considering *size* and *ANT* measures. The column titled Mean shows the average size/ANT achieved and the column titled *SD* shows the associated standard deviation. We also stress the best result for each data-set and the average one in boldface.

4 Study on the initial conditions for parametric tests using genetics-based machine learning

In this paper, we discuss on the use of statistical techniques for the analysis of GBML methods. Firstly, we distinguish between two types of analysis: *single data-set* analysis and *multiple data-set* analysis. A single data-set analysis is carried out when the results of two or more algorithms are compared considering an unique problem or data-set. A multiple data-set analysis is given when our interest lies in comparing two or more approaches over multiple problems or data-sets simultaneously, in the way of obtaining generalizable conclusions on an experimental study.

Table 3 Average classification rate offered by the algorithms

	Pitts-GIRLA		XCS		GASSIST-ADI		HIDER		CN2	
	Mean	SD	Mean	SD	Mean	SD	Mean	SD	Mean	SD
bup	0.5922	0.0641	**0.6568**	0.0764	0.6306	0.0932	0.6186	0.0986	0.5715	0.0740
cle	0.5583	0.0376	**0.5650**	0.0540	0.5613	0.0693	0.5545	0.0723	0.5412	0.0457
eco	0.7367	0.0850	0.8105	0.0680	0.7985	0.0703	**0.8422**	0.0597	0.8101	0.0618
gla	0.6247	0.1104	**0.7181**	0.1279	0.6472	0.1035	0.6962	0.1331	0.6998	0.0963
hab	0.6997	0.1245	0.7284	0.0484	0.7121	0.0676	**0.7485**	0.0449	0.7349	0.0444
iri	0.9493	0.0514	0.9493	0.0477	**0.9653**	0.0409	0.9640	0.0409	0.9400	0.0492
mon	0.6236	0.1165	**0.6728**	0.0238	0.6673	0.0407	0.6719	0.0206	0.6719	0.0215
new	0.9140	0.0499	**0.9449**	0.0545	0.9269	0.0511	0.9382	0.0660	0.9446	0.0472
pim	0.6485	0.1161	**0.7520**	0.0581	0.7425	0.0437	0.7473	0.0497	0.7122	0.0393
veh	0.4594	0.1095	**0.7359**	0.0446	0.6783	0.0421	0.6593	0.0502	0.6191	0.0839
vow	0.2467	0.0548	0.5438	0.0682	0.4020	0.0356	**0.7248**	0.0482	0.6212	0.0632
win	0.7039	0.2199	**0.9584**	0.0477	0.9056	0.0744	0.9476	0.0792	0.9268	0.0648
wis	0.7655	0.2269	**0.9666**	0.0189	0.9564	0.0247	0.9653	0.0236	0.9517	0.0218
yea	0.3723	0.0877	0.4960	0.0598	0.5442	0.0327	**0.5781**	0.0376	0.5560	0.0362
AVG	0.6353	0.1039	0.7499	0.0570	0.7242	0.0564	**0.7625**	0.0611	0.7358	0.1529

Table 4 Average kappa offered by the algorithms

	Pitts-GIRLA		XCS		GASSIST-ADI		HIDER		CN2	
	Mean	SD	Mean	SD	Mean	SD	Mean	SD	Mean	SD
bup	0.0916	0.1472	**0.2619**	0.1837	0.2382	0.1842	0.1793	0.1939	0.0444	0.1580
cle	0.1710	0.1192	**0.2995**	0.0949	0.2750	0.0948	0.2387	0.1182	0.1617	0.0586
eco	0.6260	0.1099	0.7345	0.0964	0.7158	0.1000	**0.7761**	0.0827	0.7317	0.0892
gla	0.4663	0.1490	**0.6089**	0.1731	0.5019	0.1416	0.5665	0.1899	0.5765	0.1284
hab	0.0605	0.1156	0.0943	0.1431	0.1272	0.1921	0.1469	0.1719	**0.1826**	0.1900
iri	0.9240	0.0771	0.9240	0.0716	**0.9480**	0.0614	0.9460	0.0613	0.9100	0.0738
mon	0.0067	0.0354	0.0107	0.0536	0.0460	0.1161	**0.1095**	0.1697	0.0000	0.0000
new	0.8171	0.1013	**0.8762**	0.1327	0.8424	0.1077	0.8644	0.1363	0.8742	0.1063
pim	0.1260	0.2047	**0.4321**	0.1404	0.4131	0.1103	0.3794	0.1334	0.2476	0.1182
veh	0.2802	0.1470	**0.6479**	0.0593	0.5714	0.0558	0.5450	0.0669	0.4897	0.1130
vow	0.1726	0.0602	0.4982	0.0751	0.3422	0.0391	**0.6969**	0.0530	0.5833	0.0695
win	0.5125	0.3822	**0.9371**	0.0716	0.8560	0.1135	0.9201	0.1171	0.8870	0.1000
wis	0.5465	0.3683	**0.9271**	0.0411	0.9040	0.0542	0.9222	0.0532	0.8909	0.0501
yea	0.1640	0.1226	0.3279	0.0837	0.3983	0.0453	**0.4481**	0.0505	0.4137	0.0483
AVG	0.3546	0.1528	0.5415	0.1014	0.5128	0.1011	**0.5528**	0.1141	0.4995	0.0931

The Central Limit Theorem suggests that the sum of many independent, identically distributed random variables approaches a normal distribution (Sheskin 2006). This theorem for classification performance is rarely held, it depends on the case of the problem studied and the number of runs of the algorithm. However, an excessive number of runs (the effect size of the samples) affects negatively in the statistical test due to the fact that it makes a statistical score more sensitive to a little difference of results (which would not be detected), by the simple fact of repeating runs. Thus, our intention is to study the necessary conditions for using parametric statistical tests on single data-set analysis by means of the obtaining of large size result samples by running the algorithms several times.

For doing so, we firstly introduce the necessary conditions mentioned above. Then, we present the analysis of these conditions, and finally we show some case studies of the normality property.

Table 5 Average of interpretability measures of GBML algorithms

Data-set	Size								ANT							
	Pitts-GIRLA		XCS		GASSIST-ADI		HIDER		Pitts-GIRLA		XCS		GASSIST-ADI		HIDER	
	Mean	SD	Mean	SD	Mean	SD	Mean	SD	Mean	SD	Mean	SD	Mean	SD	Mean	SD
bup	30.00	0.00	2400.62	198.03	16.84	6.20	**5.56**	1.05	2.96	0.16	**2.31**	0.28	3.53	0.46	5.29	0.36
cle	40.00	0.00	4594.96	109.62	**10.76**	4.54	22.30	2.28	6.09	0.27	4.15	0.17	**3.28**	0.92	5.75	0.24
eco	40.00	0.00	2321.02	147.56	**6.32**	1.45	8.66	1.10	3.52	0.24	2.06	0.17	**1.69**	0.40	5.39	0.39
gla	20.00	0.00	3254.32	155.87	**8.52**	2.51	22.38	2.43	3.96	0.32	2.86	0.23	**2.30**	0.64	8.44	0.20
hab	10.00	0.00	1181.52	360.75	7.92	3.25	**2.26**	0.66	1.53	0.23	**1.31**	0.25	1.81	0.48	1.93	0.25
iri	20.00	0.00	547.08	105.77	4.08	0.27	**3.00**	0.00	1.91	0.19	1.19	0.14	**0.91**	0.21	2.26	0.41
mon	20.00	0.00	283.78	95.72	**5.50**	0.61	6.26	4.15	2.73	0.23	**1.23**	0.12	1.27	0.95	3.12	0.99
new	20.00	0.00	1037.00	133.42	5.42	1.01	**3.34**	0.52	2.17	0.21	1.57	0.16	**1.52**	0.29	4.35	0.41
pim	10.00	0.00	3576.62	150.34	**15.34**	4.61	8.84	2.00	**3.04**	0.42	3.17	0.16	3.50	0.63	7.20	0.38
veh	20.00	0.00	5211.18	56.17	**11.68**	3.92	46.86	4.59	7.77	0.42	5.14	0.14	**3.19**	0.65	17.36	0.16
vow	20.00	0.00	4284.34	141.49	**11.92**	4.44	114.50	4.55	5.60	0.43	**2.04**	0.08	2.15	0.54	9.93	0.11
win	20.00	0.00	4098.70	347.37	**4.30**	0.54	27.50	2.53	5.75	0.41	2.83	0.18	**1.74**	0.35	12.79	0.11
wis	50.00	0.00	708.90	78.20	5.92	1.35	**2.12**	0.33	4.46	0.23	**2.11**	0.14	3.19	0.72	3.46	0.59
yea	20.00	0.00	3608.44	221.66	**8.38**	2.17	46.12	8.27	3.67	0.31	**2.29**	0.21	2.37	0.48	6.07	0.12
AVG	24.29	0.00	2650.61	164.43	**8.78**	2.70	22.84	2.46	3.94	0.29	2.45	0.17	**2.32**	0.55	6.67	0.34

4.1 Conditions for a safe use of parametric tests

In (Sheskin 2006), the distinction done between parametric and non-parametric tests is based on the level of measure represented by the data that will be analysed. In this way, a parametric test uses data with real values belonging to a range.

The latter does not involve that when we always dispose of this type of data, we should use a parametric test. It is possible that one or more initial assumptions for the use of parametric tests may be not fulfilled, making that a statistical analysis loses credibility.

In order to use the parametric tests, it is necessary to check the following conditions (Sheskin 2006; Zar 1999):

- Independence: In statistics, two events are independent when the fact that one occurs does not modify the probability of the other one occurring.
- Normality: An observation is normal when its behaviour follows a normal or Gauss distribution with a certain value of mean μ and variance σ^2. A normality test applied over a sample can indicate the presence or absence of this condition in the observed data. A well-known example of normality test is the Kolmogorov-Smirnov test, which possess a very low power. In this study, we will use more powerful normality tests:
  - Shapiro-Wilk (SW): It analyses the observed data for computing the level of symmetry and kurtosis (shape of the curve) in order to compute the difference with respect to a Gaussian distribution afterwards, obtaining the p value from the sum of the squares of these discrepancies. The power of this test has been shown to be excellent; however, its performance is adversely affected in the common situation where there is tied data.
  - D'Agostino-Pearson (DP): It first computes the skewness and kurtosis to quantify how far from Gaussian the distribution is in terms of asymmetry and shape. Then, it calculates how far differs each one of these values from the expected value with a Gaussian distribution, and computes a single p value from the sum of these discrepancies. The performance of this test is not as good as that of SW's procedure, but it is not as affected by tied data.
- Heteroscedasticity: This property indicates the existence of a violation of the hypothesis of equality of variances. A Levene test is used for checking whether or not k samples present this homogeneity of variances (homoscedasticity). When the observed data does not fulfill the normality condition, it is more reliable the result of using this test than Bartlett test (Zar 1999), which is another test that checks the same property.

With respect to the independence condition, Demšar suggests in (Demšar 2006) that independency is not truly verified in 10 fcv (a portion of samples is used either for training and testing in different partitions). In the

Table 6 Normality condition in classification rate

	bup	cle	eco	gla	hab	iri	mon	new	pim	veh	vow	win	wis	yea
Shapiro-Wilk														
Pitts-GIRLA	* (0.02)	* (0.00)	* (0.00)	(0.73)	* (0.00)	* (0.00)	* (0.00)	* (0.01)	* (0.00)	* (0.00)	* (0.00)	* (0.00)	* (0.00)	* (0.00)
XCS	(0.25)	* (0.03)	(0.23)	* (0.00)	* (0.02)	* (0.00)	* (0.00)	* (0.00)	* (0.03)	(0.17)	(0.30)	* (0.00)	* (0.00)	(0.45)
GASSIST	(0.17)	* (0.01)	(0.22)	(0.31)	(0.08)	* (0.00)	(0.07)	* (0.00)	* (0.01)	(0.96)	(0.32)	* (0.00)	* (0.04)	(0.78)
HIDER	(0.11)	(0.42)	(0.22)	* (0.00)	* (0.01)	* (0.00)	(0.06)	* (0.00)	* (0.00)	(0.25)	(0.15)	* (0.00)	* (0.00)	(0.23)
D'Agostino-Pearson														
Pitts-GIRLA	(0.13)	(0.10)	* (0.00)	(0.69)	* (0.00)	(0.11)	* (0.00)	(0.71)	* (0.00)	* (0.02)	* (0.00)	* (0.00)	* (0.00)	* (0.00)
XCS	(0.44)	(0.09)	(0.61)	(0.06)	(0.22)	(0.06)	* (0.00)	* (0.00)	(0.24)	(0.33)	(0.40)	* (0.00)	* (0.03)	(0.48)
GASSIST	(0.16)	(0.13)	(0.88)	(0.37)	(0.58)	(0.08)	(0.19)	(0.70)	* (0.02)	(0.93)	(0.95)	(0.17)	(0.36)	(0.39)
HIDER	(0.07)	(0.52)	(0.42)	(0.05)	(0.78)	* (0.00)	(0.19)	* (0.00)	* (0.00)	(0.43)	(0.37)	* (0.00)	* (0.02)	(0.18)

following, we show a normality analysis by using SW and DP tests, together with a heteroscedasticity analysis by using a Levene test.

4.2 Analysis of the conditions for a safe use of parametric tests

We apply the two tests of normality (SW and DP) presented above by considering a level of significance $\alpha = 0.05$ (we have employed the statistical software package SPSS). Tables 6 and 7 show the results in *classification rate* and *kappa* measures, respectively. Tables 8 and 9 show the results in size and ANT measures, respectively. The symbol "*" indicates that the normality condition is not satisfied and the value in brackets is the *p* value needed for rejecting the normality hypothesis.

As we can observe in the run of the two tests of normality, we can declare that the conditions needed for the application of parametric tests are not fulfilled in some cases. The normality condition is not always satisfied although the size of the sample of results would be large enough (50 in this case). A main factor that influences this condition seems to be the nature of the problem, since there exist some problems in which it is never satisfied, such as in *wine* and *wisconsin* problems in both *classification rate* and *kappa* measures, and the general trend is not predictable. In addition, the results offered by Pitts-GIRLA are very distant to a normal shape. The measure which yields less rejections of the normality condition is *ANT*.

In relation to the heteroscedasticity study, Table 10 shows the results by applying a Levene test, where the symbol "*" indicates that the variances of the distributions of the different algorithms for a certain function are not homogeneous (the null hypothesis is rejected).

The homoscedasticity property is even more difficult to be fulfilled, since the variances associated to each problem also depend on the algorithm's results, that is, the capacity of the algorithms for offering similar results with random seeds variations. This fact also influences that an analysis of performance of GBML algorithms carried out through parametric statistical treatment could lead to erroneous conclusions.

4.3 Case studies of the normality property

We present two case studies of the normality property considering the sample of results obtained by an GBML method on a data-set. Figs. 1 and 2 show different examples of graphical representations of histograms and Q-Q graphics. A histogram represents a statistical variable by using bars, so that the area of each bar is proportional to the frequency of the represented values. A Q-Q graphic

Table 7 Normality condition in Cohen's kappa

	bup	cle	eco	gla	hab	iri	mon	new	pim	veh	vow	win	wis	yea
Shapiro-Wilk														
Pitts-GIRLA	* (0.00)	* (0.02)	* (0.00)	(0.79)	* (0.00)	* (0.00)	* (0.00)	* (0.04)	* (0.00)	* (0.00)	* (0.00)	* (0.00)	* (0.00)	* (0.00)
XCS	(0.65)	(0.11)	(0.37)	* (0.00)	* (0.01)	* (0.00)	* (0.00)	* (0.00)	* (0.04)	(0.17)	(0.30)	* (0.00)	* (0.00)	(0.51)
GASSIST	(0.30)	* (0.03)	(0.47)	(0.32)	(0.77)	* (0.00)	* (0.00)	* (0.01)	* (0.01)	(0.98)	(0.32)	* (0.00)	(0.07)	(0.14)
HIDER	(0.61)	(0.42)	(0.21)	* (0.00)	* (0.01)	* (0.00)	* (0.00)	* (0.00)	* (0.01)	(0.23)	(0.56)	* (0.00)	* (0.00)	(0.20)
D'Agostino-Pearson														
Pitts-GIRLA	* (0.00)	(0.49)	* (0.00)	(0.58)	* (0.00)	(0.11)	* (0.00)	(0.80)	* (0.00)	* (0.01)	* (0.01)	* (0.00)	* (0.00)	* (0.00)
XCS	(0.54)	(0.41)	(0.72)	(0.06)	* (0.03)	(0.06)	* (0.00)	* (0.00)	(0.27)	(0.32)	(0.40)	* (0.01)	* (0.04)	(0.35)
GASSIST	(0.16)	(0.10)	(0.90)	(0.21)	(0.96)	(0.09)	* (0.00)	(0.66)	* (0.01)	(0.95)	(0.95)	(0.18)	(0.39)	(0.19)
HIDER	(0.33)	(0.45)	(0.43)	(0.05)	(0.21)	* (0.00)	* (0.00)	* (0.02)	* (0.00)	(0.41)	(0.38)	* (0.00)	* (0.01)	(0.20)

Table 8 Normality condition in size

	bup	cle	eco	gla	hab	iri	mon	new	pim	veh	vow	win	wis	yea
Shapiro-Wilk														
Pitts-GIRLA	* (0.00)	* (0.00)	* (0.00)	* (0.00)	* (0.00)	* (0.00)	* (0.00)	* (0.00)	* (0.00)	* (0.00)	* (0.00)	* (0.00)	* (0.00)	* (0.00)
XCS	(0.91)	(0.53)	(0.86)	(0.89)	* (0.00)	(0.75)	(0.26)	(0.74)	* (0.00)	(0.42)	(0.46)	* (0.00)	(0.56)	(0.59)
GASSIST	* (0.00)	* (0.00)	* (0.00)	* (0.00)	* (0.00)	* (0.00)	* (0.00)	* (0.00)	* (0.01)	(0.16)	* (0.00)	* (0.00)	* (0.00)	(0.13)
HIDER	* (0.00)	(0.16)	* (0.00)	* (0.00)	* (0.00)	* (0.00)	* (0.00)	* (0.00)	* (0.04)	(0.46)	(0.84)	(0.21)	* (0.00)	(0.10)
D'Agostino-Pearson														
Pitts-GIRLA	* (0.00)	* (0.00)	* (0.00)	* (0.00)	* (0.00)	* (0.00)	* (0.00)	* (0.00)	* (0.00)	* (0.00)	* (0.00)	* (0.00)	* (0.00)	* (0.00)
XCS	(0.67)	(0.17)	(0.86)	(0.84)	* (0.00)	(0.76)	(0.27)	(0.47)	* (0.00)	(0.38)	(0.22)	* (0.00)	(0.52)	(0.76)
GASSIST	* (0.01)	* (0.00)	* (0.00)	* (0.00)	* (0.00)	* (0.00)	(0.68)	(0.54)	* (0.03)	(0.43)	* (0.04)	* (0.00)	* (0.00)	(0.21)
HIDER	(0.86)	(0.61)	(0.47)	* (0.00)	(0.23)	* (0.00)	(0.10)	* (0.01)	(0.98)	(0.47)	(0.80)	(0.37)	* (0.00)	(0.21)

Table 9 Normality condition in ANT

	bup	cle	eco	gla	hab	iri	mon	new	pim	veh	vow	win	wis	yea
Shapiro-Wilk														
Pitts-GIRLA	(0.83)	(0.13)	(0.10)	(0.50)	(0.26)	(0.27)	* (0.00)	(0.12)	(0.20)	(0.51)	(0.96)	(0.32)	* (0.03)	(0.39)
XCS	(0.15)	(0.67)	(0.18)	(0.10)	(0.55)	(0.64)	(0.86)	(0.23)	(0.73)	(0.67)	(0.43)	(0.46)	(0.68)	(0.17)
GASSIST	(0.85)	* (0.04)	* (0.01)	* (0.00)	(0.19)	* (0.00)	* (0.00)	* (0.01)	* (0.00)	(0.27)	* (0.00)	(0.09)	(0.58)	(0.38)
HIDER	* (0.01)	(0.22)	* (0.00)	* (0.01)	* (0.00)	* (0.01)	* (0.00)	* (0.04)	* (0.01)	(0.26)	(0.05)	(0.74)	* (0.00)	(0.70)
D'Agostino-Pearson														
Pitts-GIRLA	(0.73)	(0.05)	* (0.04)	(0.63)	(0.84)	(0.39)	(0.07)	(0.38)	(0.41)	(0.88)	(0.84)	(0.39)	* (0.00)	(0.33)
XCS	* (0.03)	(0.57)	(0.20)	(0.23)	(0.26)	(0.67)	(0.89)	(0.34)	(0.46)	(0.50)	(0.67)	(0.56)	(0.61)	(0.18)
GASSIST	(0.88)	* (0.00)	* (0.00)	* (0.00)	(0.13)	* (0.00)	* (0.00)	* (0.01)	* (0.01)	(0.05)	* (0.00)	(0.69)	(0.57)	(0.72)
HIDER	* (0.00)	(0.18)	* (0.00)	* (0.00)	(0.76)	* (0.00)	(0.09)	(0.61)	* (0.00)	(0.69)	(0.18)	(0.63)	(0.27)	(0.72)

represents a confrontation between the quartiles from data observed and those from the normal distribution.

In Fig. 1 we observe a typical case of absolute lack of normality. Figure 2 illustrates an example in which the normality hypothesis is accepted as well by the two tests used.

5 Non-parametric tests for comparing two algorithms in multiple data-set analysis

As we introduced previously, the obtention of results in a single data-set analysis when using GBML algorithms is a relatively easy task, due to the fact that new results can be

Table 10 Heteroscedasticity condition by using a Levene test

	bup	cle	eco	gla	hab	iri	mon	new	pim	veh	vow	win	wis	yea
Classification rate	(0.16)	* (0.00)	(0.77)	(0.26)	* (0.01)	(0.53)	* (0.00)	(0.36)	(0.05)	* (0.00)	* (0.00)	* (0.00)	* (0.00)	* (0.00)
Cohen's kappa	(0.53)	* (0.02)	(0.66)	(0.17)	* (0.02)	(0.53)	* (0.00)	(0.36)	* (0.00)	* (0.00)	* (0.00)	* (0.00)	* (0.00)	* (0.00)
Size	* (0.00)	* (0.00)	* (0.00)	* (0.00)	* (0.00)	* (0.00)	* (0.00)	* (0.00)	* (0.00)	* (0.00)	* (0.00)	* (0.00)	* (0.00)	* (0.00)
ANT	* (0.00)	* (0.00)	* (0.00)	* (0.00)	* (0.00)	* (0.00)	* (0.00)	* (0.00)	* (0.00)	* (0.00)	* (0.00)	* (0.00)	* (0.00)	* (0.00)

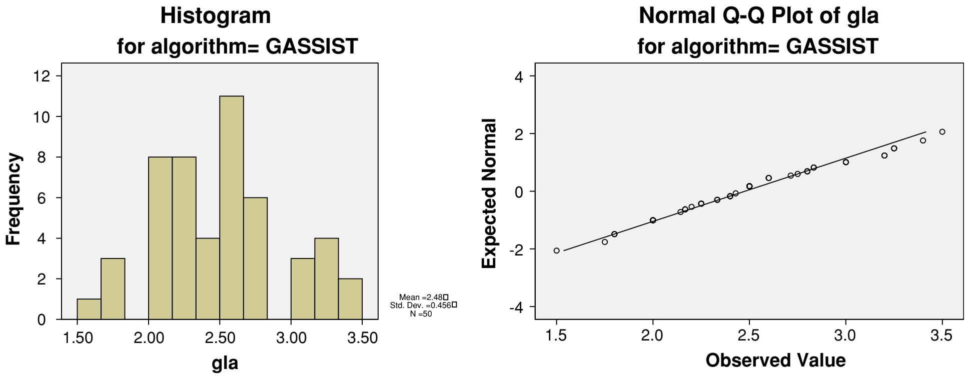

Fig. 1 Cohen's kappa results of XCS over monks data-set: histogram and Q-Q Graphic

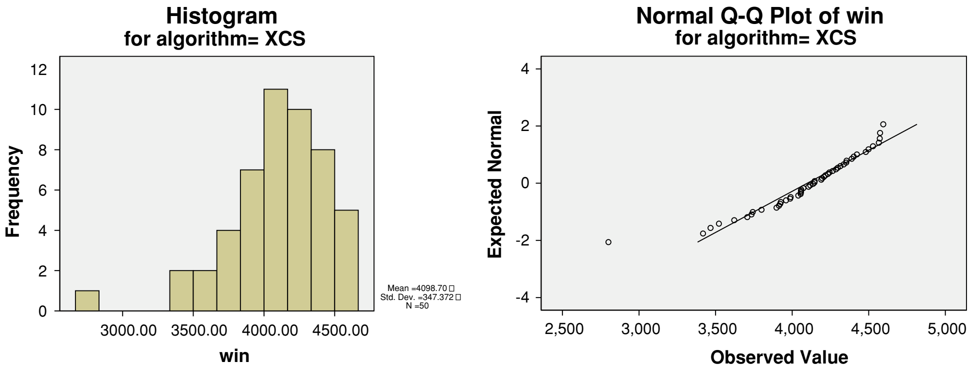

Fig. 2 Classification rate results of GASSIST-ADI over yeast data-set: histogram and Q-Q Graphic

yielded in new runs of the algorithms. In spite of this fact, a sample of 50 results does not always verified the necessary conditions for applying parametric tests, as we could see in the previous section.

On the other hand, other ML approaches are not stochastic and it is not possible to obtain a larger sample of results. This fact makes difficult the comparison between GBML methods and deterministic ML algorithms, given that the sample of results could not be large enough or there is a necessity for using procedures which can operate with samples of different size.

The authors are usually familiarized with parametric and non-parametric tests for pairwise comparisons. GBML approaches have been compared through parametric tests by means of paired *t* tests (Aguilar-Ruiz et al. 2000; Anglano and Botta 2002; Bernadó-Mansilla and Ho 2005; Guan and Zhu 2005). In some cases, the *t* test is accompanied with the non-para-metric Wilcoxon test applied over multiple data-sets (Bernadó-Mansilla and Garrell 2003; Tulai and Oppacher 2004). The use of these types of tests is correct when we are interested in finding the differences between two methods, but they must not be used when we are interested in comparisons that include several methods. In the case of repeating pairwise comparisons, there is an associated error that grows agreeing with the number of comparisons done, called the family-wise error rate (FWER), defined as the probability of at least one error in the family of hypotheses. For solving this problem, some authors use the Bonferroni correction for applying paired t-test in their works (Tan et al. 2006; Bacardit 2004).

Our interest lies in presenting a methodology for analysing the results offered by the algorithms in a certain study of GBML, by using non-parametric tests in a multiple data-set analysis. Furthermore, we want to remark the possibility of comparison with other deterministic ML algorithms. Non-parametric tests could be applied to small sample of data and their effectiveness have been proved in complex experiments. They are preferable to an adjustment of data with transformations or to a discarding of certain extreme observations (outliers) (Koch 1970).

This section is devoted to describing a non-parametric statistical procedure for performing pairwise comparisons between two algorithms, which is the Wilcoxon signed-rank test, Section 5.1; and to show the operation of this test in the presented case study, Section 5.2.

5.1 Wilcoxon signed-ranks test

This is the analogous of the paired t-test in non-parametric statistical procedures; therefore, it is a pairwise test that aims to detect significant differences between two sample means, that is, the behavior of two algorithms. Let d_i be the difference between the performance scores of the two classifiers on ith out of N_{ds} data-sets. The differences are ranked according to their absolute values; average ranks are assigned in case of ties. Let R^+ be the sum of ranks for the data-sets on which the first algorithm outperformed the second, and R^- the sum of ranks for the opposite. Ranks of $d_i = 0$ are split evenly among the sums; if there is an odd number of them, one is ignored:

$$R^+ = \sum_{d_i > 0} \text{rank}(d_i) + \frac{1}{2}\sum_{d_i=0} \text{rank}(d_i)$$

$$R^- = \sum_{d_i < 0} \text{rank}(d_i) + \frac{1}{2}\sum_{d_i=0} \text{rank}(d_i)$$

Let T be the smaller of the sums, $T = \min(R^+, R^-)$. If T is less than or equal to the value of the distribution of Wilcoxon for N_{ds} degrees of freedom (Zar 1999, Table B.12), the null hypothesis of equality of means is rejected.

Wilcoxon signed ranks test is more sensible than the t-test. It assumes commensurability of differences, but only qualitatively: greater differences still count more, which is probably desired, but the absolute magnitudes are ignored. From the statistical point of view, the test is safer since it does not assume normal distributions. Also, the outliers (exceptionally good/bad performances on a few data-sets) have less effect on the Wilcoxon than on the t test. The Wilcoxon test assumes continuous differences d_i, therefore they should not be rounded to one or two decimals, since this would decrease the power of the test due to a high number of ties.

When the assumptions of the paired t-test are met, Wilcoxon signed-ranks test is less powerful than the pai-red t test. On the other hand, when the assumptions are violated, the Wilcoxon test can be even more powerful than the t test. This allows us to apply it over the means obtained by the algorithms in each data-set, without any assumptions about the sample of results obtained.

5.2 A case study in GBML: performing pairwise comparisons

In the following, we will perform the statistical analysis by means of pairwise comparisons by using the results of performance measures obtained by the algorithms described in Sect. 2.

In order to compare the results between two algorithms and to stipulate which one is the best, we can perform a Wilcoxon signed-rank test for detecting differences in both means. This statement must be enclosed by a probability of error, that is the complement of the probability of reporting that two systems are the same, called the p value (Zar 1999). The computation of the p value in the Wilcoxon distribution could be carried out by computing a normal approximation (Sheskin 2006). This test is well known and it is usually included in standard statistics packages (such as SPSS, R, SAS, etc.).

Table 11 Wilcoxon test applied over the all possible comparisons between the five algorithms in classification rate

Comparison	Classification rate		
	R^+	R^-	p value
Pitts-GIRLA-**XCS**	0.5	104.5	0.001
Pitts-GIRLA-**GASSIST-ADI**	0	105	0.001
Pitts-GIRLA-**HIDER**	1	104	0.001
Pitts-GIRLA-**CN2**	6	99	0.004
XCS-GASSIST-ADI	89	16	0.022
XCS-HIDER	53	52	0.975
XCS-CN2	78	27	0.109
GASSIST-ADI-**HIDER**	20	85	0.041
GASSIST-ADI-CN2	52	53	0.975
HIDER-CN2	100	5	0.003

Table 12 Wilcoxon test applied over the all possible comparisons between the five algorithms in kappa

Comparison	Cohen's kappa		
	R^+	R^-	p value
Pitts-GIRLA-**XCS**	0.5	104.5	0.001
Pitts-GIRLA-**GASSIST-ADI**	0	105	0.001
Pitts-GIRLA-**HIDER**	0	105	0.001
Pitts-GIRLA-**CN2**	10	95	0.008
XCS-GASSIST-ADI	74	31	0.177
XCS-HIDER	51	54	0.925
XCS-CN2	78	27	0.109
GASSIST-ADI-HIDER	28	77	0.124
GASSIST-ADI-CN2	60	45	0.638
HIDER-CN2	96	9	0.006

Tables 11 and 12 show the results obtained in all possible comparisons among the five algorithms considered in the study, in classification rate and kappa respectively. We stress in bold the winner algorithm in each row when the p value associated is below 0.05.

The comparisons performed in this study are independent, so they never have to be considered in a whole. If we try to extract from the previous tables a conclusion which involves more than one comparison, we are losing control on the FWER. For instance, the statement: "HIDER algorithm obtains a classification rate better than Pitts-GIRLA and GASSIST-ADI algorithms with a p value lower than 0.05" is incorrect, since we do not prove the control of the FWER. The HIDER algorithm really outperforms Pitts-GIRLA and GASSIST-ADI algorithms considering classification rate in independent comparisons.

The true statistical signification for combining pairwise comparisons is given by expression 5:

$$\begin{aligned} p &= P(Reject\,H_0|H_0\,true) \\ &= 1 - P(Accept\,H_0|H_0\,true) \\ &= 1 - P(Accept\,A_k = A_i, i = 1, \ldots, k-1|H_0\,true) \\ &= 1 - \prod\nolimits_{i=1}^{k-1} P(Accept\_A_k = A_i|H_0\,true) \\ &= 1 - \prod\nolimits_{i=1}^{k-1} [1 - P(Reject\,A_k = A_i|H_0\,true)] \\ &= 1 - \prod\nolimits_{i=1}^{k-1} (1 - p_{H_i}) \end{aligned} \tag{5}$$

Wilcoxon test suggests the following information:

- Regarding classification rate, the best algorithms are XCS and HIDER. In the comparison between them, XCS obtains the most favourable ranking, but its difference with respect to HIDER is rather small, so they are statistically equal in classification rate. Nevertheless, HIDER independently outperforms CN2, whereas XCS does not.
- Regarding kappa, the best algorithms are XCS, HIDER and GASSIST-ADI. The null hypothesis of equality of means is rejected when Pitts-GIRLA takes part in a comparison. In their comparison, HIDER obtains the best ranking and it outperforms CN2 algorithm (XCS and GASSIST-ADI do not).

6 Non-parametric tests for multiple comparisons among more than two algorithms

When a new GBML algorithm proposal is developed, it could be interesting to compare it with previous approaches. Making pairwise comparisons allows us to conduct this analysis, but the experiment wise error can not be previously controlled. Furthermore, a pairwise comparison is not influenced by any external factor, whereas in a multiple comparison, the set of algorithms chosen can determine the results of the analysis.

Multiple comparison procedures are designed for allowing us to fix the FWER before performing the analysis and for taking into account all the influences that can exist within the set of results for each algorithm. Following the same structure as in the previous section, the basic and advanced non-parametrical tests for multiple comparisons are described in Sect. 6.1 and their application on the case study is conducted in Sect. 6.2.

6.1 Friedman test and post-hoc tests

In order to perform a multiple comparison, it is necessary to check whether all the results obtained by the algorithms present any inequality. In the case of finding it, then we can know, by using a post-hoc test, what algorithms partners' average results are dissimilar. In the following, we describe the non-parametric tests used.

- The first one is the Friedman test (Sheskin 2006), which is a non-parametric test equivalent to the repeated-measures ANOVA. Under the null-hypothesis, it states that all the algorithms are equivalent, so a rejection of this hypothesis implies the existence of differences among the performance of all the algorithms studied. After this, a post-hoc test could be used in order to find whether the control or proposed algorithm presents statistical differences with regards to the remaining methods in the comparison. The simplest of them is the Bonferroni-Dunn test, but it is a very conservative procedure and we can use more powerful tests that control the FWER and reject more hypothesis than the Bonferroni-Dunn test; for example the Holm method (Holm 1979).

  The working mode of the Friedman test is described as follows: It ranks the algorithms for each data-set separately, the best performing algorithm getting the rank of 1, the second best rank 2, and so on. In case of ties average ranks are assigned.

  Let ${r^j}_i$ be the rank of the jth of k algorithms on the ith of $N_{\rm ds}$ data-sets. The Friedman test compares the average ranks of algorithms, $R_j = \frac{1}{N_{\rm ds}} \sum_i r_i^j$. Under the null-hypothesis, which states that all the algorithms are equivalent and so their ranks R_j should be equal, the Friedman statistic:

$$\chi_F^2 = \frac{12N_{\rm ds}}{k(k+1)}\left[\sum jR_j^2 - \frac{k(k+1)^2}{4}\right]$$

  is distributed according to χ_F^2 with $k-1$ degrees of freedom, when $N_{\rm ds}$ and k are big enough.
- The second one of them is Iman and Davenport test (Iman and Davenport 1980), which is a non-parametric

test, derived from the Friedman test, less conservative than the Friedman statistic:

$$F_F = \frac{(N_{\mathrm{ds}} - 1)\chi_F^2}{N_{\mathrm{ds}}(K-1) - \chi_F^2}$$

which is distributed according to the F-distribution with $k - 1$ and $(k - 1)(N_{\mathrm{ds}} - 1)$ degrees of freedom. Statistical tables for critical values can be found at (Sheskin 2006; Zar 1999).

- Bonferroni–Dunn test: if the null hypothesis is rejected in any of the previous tests, we can continue with Bonferroni–Dunn procedure. It is similar to Dunnet test for ANOVA and it is used when we want to compare a control algorithm opposite to the remainder. The quality of two algorithms is significantly different if the corresponding average of rankings is at least as great as its critical difference (CD).

$$\mathrm{CD} = q_\alpha \sqrt{\frac{k(k+1)}{6N}}.$$

The value of q_α is the critical value for a multiple non-parametric comparison with a control (Zar 1999, Table B.16).

- Holm test (Holm 1979): it is a multiple comparison procedure that can work with a control algorithm and compares it with the remaining methods. The test statistics for comparing the i-th and j-th method using this procedure is:

$$z = (R_i - R_j) / \sqrt{\frac{k(k+1)}{6N_{\mathrm{ds}}}}$$

The z value is used to find the corresponding probability from the table of normal distribution, which is then compared with an appropriate level of confidence α. In Bonferroni-Dunn comparison, this α value is always $\alpha/(k-1)$, but the Holm test adjusts the value for α in order to compensate for multiple comparison and control the FWER.

The Holm test is a step-up procedure that sequentially tests the hypotheses ordered by their significance. We will denote the ordered p values by p_1, p_2,..., so that $p_1 \leq p_2 \leq \cdots \leq p_{k-1}$. The Holm test compares each p_i with $\alpha/(k - i)$, starting from the most significant p value. If p_1 is below $\alpha/(k - 1)$, the corresponding hypothesis is rejected and we allow to compare p_2 with $\alpha/(k - 2)$. If the second hypothesis is rejected, the test proceeds with the third, and so on. As soon as a certain null hypothesis cannot be rejected, all the remain hypotheses are retained as well.

- Hochberg procedure (Hochberg 1988): It is a step-up procedure that works in the opposite direction to the Holm method, comparing the largest p value with α, the next largest with $\alpha/2$ and so forth until it encounters a hypothesis that it can reject. All hypotheses with smaller p values are then rejected as well.

The post-hoc procedures described above allow us to know whether or not a hypothesis of comparison of means could be rejected at a specified level of significance α. However, it is very interesting to compute the p value associated to each comparison, which represents the lowest level of significance of a hypothesis that results in a rejection. In this manner, we can know whether two algorithms are significantly different and we can also have a metric of how different they are.

Next, we will describe the method used for computing these exact p values for each test procedure, which are called "adjusted p values" (Wright 1992).

- The adjusted p value for the Bonferroni–Dunn test (also known as the Bonferroni correction) is calculated by $p_{\mathrm{Bonf}} = (k - 1)p_i$.
- The adjusted p value for the Holm procedure is computed by $p_{\mathrm{Holm}} = (k - i)p_i$. Once computed all of them for all hypotheses, it is not possible to find an adjusted p value for the hypothesis i lower than for the hypothesis j, $j < i$. In this case, the adjusted p value for hypothesis i is set to the same value as the one associated to hypothesis j.
- The adjusted p value for the Hochberg method is computed with the same formula as in the Holm procedure, and the same restriction is applied in the process, but in the opposite sense, that is, it is not possible to find an adjusted p value for the hypothesis i lower than for the hypothesis j, $j > i$.

6.2 A case study in GBML: performing multiple comparisons

This section presents the study of applying multiple comparisons procedures to the results of the case study

Table 13 Results of the Friedman and Iman–Davenport tests ($\alpha = 0.05$)

	Friedman Value	Value in χ^2	p value	Iman–Davenport Value	Value in F_F	p value
Classification rate	**28.957**	9.487	<0.0001	**13.920**	2.55	<0.0001
Cohen's kappa	**26.729**	9.487	<0.0001	**11.871**	2.55	<0.0001

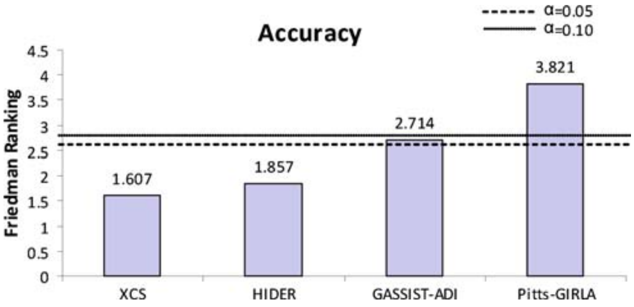

Fig. 3 Bonferroni–Dunn graphic for classification rate

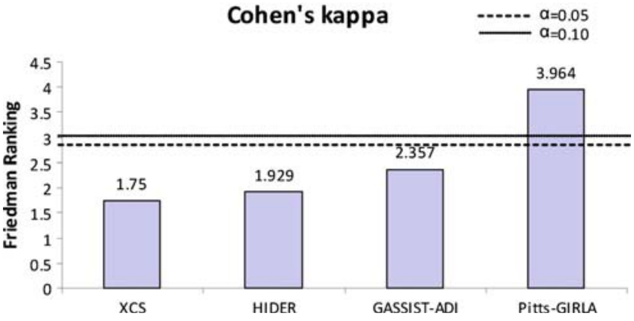

Fig. 4 Bonferroni–Dunn graphic for kappa

described above. We will use the results obtained in the evaluation of the performance measures considered and we will define the control algorithm as the best performing algorithm (which obtains the lowest value of ranking, computed through a Friedman test).

First of all, we have to test whether significant differences exist among all the mean values. Table 13 shows the result of applying a Friedman and Iman–Davenport tests. The table shows the Friedman and Iman–Davenport values, χ^2_F and F_F, respectively, and it relates them with the corresponding critical values for each distribution by using a level of significance $\alpha = 0.05$. The p value obtained is also reported for each test. Given that the statistics of Friedman and Iman–Davenport are clearly greater than their associated critical values, there are significant differences among the observed results with a level of significance $\alpha \leq 0.05$. According to these results, a post-hoc statistical analysis is needed in the two cases.

Then, we will employ a Bonferroni-Dunn test to detect significant differences for the control algorithm in each measure. It obtains the values CD = 1.493 and CD = 1.34 for $\alpha = 0.05$ and $\alpha = 0.10$ respectively in the two measures considered. Figures 3 and 4 summarize the ranking obtained by the Friedman test and draw the threshold of the critical difference of Bonferroni–Dunn' procedure, with the two levels of significance mentioned above. They display a graphical representation composed by bars whose height is proportional to the average ranking obtained for each algorithm in each measure studied. If we choose the smallest of them (which corresponds to the best algorithm), and we sum its height with the critical difference obtained by the Bonferroni method (*CD* value), we represent a cut line that goes through all the graphic. Those bars which are higher than this cut line belong to the algorithms whose performance is significantly worse than that of the control algorithm.

We will apply more powerful procedures, such as Holm and Hochbergs ones, for comparing the control algorithm with the rest of algorithms. Table 14 shows all the adjusted p values for each comparison which involves the control algorithm. The p value is indicated in each comparison and we stress in bold the algorithms which are worse than the control, considering a level of significance $\alpha = 0.05$.

Note that the results offered by the two most powerful procedures, the Holm and Hochberg methods, are the same in this case study. In practice, a Hochberg method is more powerful than the Holm one, but this difference is rather small (Shaffer 1995). In any case, the results here do not coincide exactly with the results obtained with the use of a Wilcoxon test in Sect. 5.2:

Table 14 Adjusted p values for the comparison of the control algorithm in each measure with the remaining algorithms (Holm and Hochberg tests)

i	Algorithm	Unadjusted p	p_{Bonf}	p_{Holm}	p_{Hoch}
Classification rate (XCS is the control)					
1	**Pitts-GIRLA**	1.745×10^{-6}	6.980×10^{-6}	6.980×10^{-6}	6.980×10^{-6}
2	**CN2**	0.01428	0.05711	0.04283	0.04283
3	GASSIST-ADI	0.02702	0.10810	0.05405	0.05405
4	HIDER	0.67571	1.00000	0.67571	0.67571
Cohen's kappa (XCS is the control)					
1	**Pitts-GIRLA**	5.576×10^{-6}	2.230×10^{-5}	2.230×10^{-5}	2.230×10^{-5}
2	CN2	0.01977	0.07908	0.05931	0.05931
3	GASSIST-ADI	0.13517	0.54067	0.27033	0.27033
4	HIDER	0.76509	1.00000	0.76509	0.76509

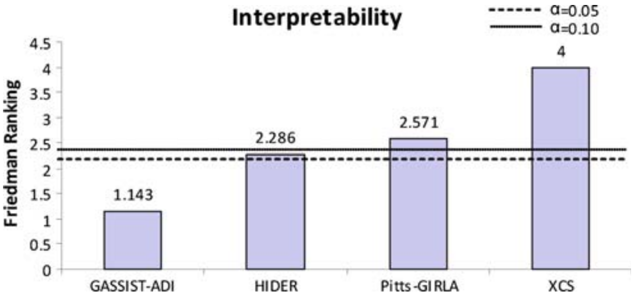

Fig. 5 Bonferroni–Dunn graphic measuring interpretability

- In classification rate, the difference between XCS and HIDER is higher in Holm and Hochberg tests than in the Wilcoxon one. Anyway, no testing procedure is able to distinguish one of them as the best.
- In Cohen's kappa, according to the Holm and Hochberg procedures, the difference between XCS and HIDER is also higher than according to the Wilcoxon test.

After conducting the multiple comparison analysis, we can observe that:

- By using the classification rate measure, XCS is significantly better than Pitts-GIRLA and CN2, but it behaves equally to GASSIST-ADI and HIDER.
- Considering the kappa measure, only Pitts-GIRLA obtains the worst results with respect to the remaining algorithms. The other four GBML algorithms do not differ significantly.
- HIDER loses performance when we evaluate the results with kappa, whereas GASSIST-ADI achieves a better kappa rate (Figs. 3 and 4). The latter seems to be more robust against randomness yielded by the data.

In relation to the sample size (number of data-sets when performing a Wilcoxon or Friedman tests in multiple data-set analysis), there are two main aspects to be determined. Firstly, the minimum sample considered acceptable for each test needs to be stipulated. There is no established agreement about this specification. In our case, the use of a sample as large as possible is preferable, because the power of the statistical tests (defined as the probability that the test will reject a false null hypothesis) will increase. Furthermore, in a multiple data-set analysis, the increase of the sample size depends on the availability of new data-sets. Secondly, we have to study how the results are expected to vary if there was a larger sample size available. In all statistical tests used for comparing two or more samples, the increasing of the sample size benefits the power of the test. As a rule of thumb, the number of data-sets should be greater than 2 k, where k is the number of methods to be compared.

7 Analysing interpretability of models

The interpretability of the rule sets obtained will be evaluated by means of the two measures described in Sect. 3.2, size and ANT. We will aggregate these two measures in one, which will represent the complexity of the rule set. It measures the average complexity of the rule set taking into account the number of rules and the average number of antecedents per rule:

$$complexity = size \cdot ANT.$$

By using the data contained in Sect. 3.3 at Table 5, we can conduct a statistical study of the complexity of the rule

Table 15 Adjusted p values for the comparison of complexity of rules (Holm and Hochberg tests)

Interpretability (GASSIST-ADI is the control)					
i	Algorithm	Unadjusted p	p_{Bonf}	p_{Holm}	p_{Hoch}
1	XCS	2.657×10^{-8}	7.972×10^{-8}	7.972×10^{-8}	7.972×10^{-8}
2	Pitts-GIRLA	0.01283	0.03848	0.02565	0.02565
3	HIDER	0.05704	0.17112	0.05704	0.05704

Table 16 Examples of rules in *iris* data-set

Algorithm	Example of rule
Pitts-GIRLA	IF sepalLength = Don't Care AND sepalWidth = Don't Care AND petalLength = [4.947674287707237,5.965516026050438] AND petalWidth = Don't Care THEN Class = Iris-virginica.
XCS	(normalized) IF sepalLength = [0.0,1.0] AND sepalWidth = [0.0, 1.0] AND petalLength = [0.3641094725703955, 1.0] AND petalWidth = [0.0, 1.0] THEN Class = Iris-setosa
GASSIST-ADI	IF petalLength = [1.0,5.071428571428571] ANDpetalWidth = [0.5363636363636364,1.6272727272727274] THEN Class = Iris-versicolor
HIDER	IF sepalLength = (..., 6.65] AND petalLength = (..., 6.7] AND petalWidth = (..., 0.75] THEN Class = iris-setosa

sets obtained in a multiple data-sets analysis. In this study, only multiple comparison procedures will be used.

Figure 5 shows a Bonferroni-Dunn graphic which compares the complexity of the rule set and Table 15 displays the adjusted p values for all the multiple comparison procedures considered in this study.

As we can see, the two most powerful statistical procedures (Holm and Hochberg ones) are able to distinguish the GASSIST-ADI algorithm as the one whose rule sets are the most interpretable with a $p = 0.05704$ (a level of significance $\alpha = 0.10$ is required).

However, we have to be cautious with respect to the concept of interpretability. GBML algorithms can produce different types of rules or different ways for reading or interpreting the rules. For example, the four algorithms used in this paper produce rule sets with different properties. In Table 16 we show an example of rule for each algorithm, considering the *iris* data-set in the examples):

- Pitts-GIRLA yields a set of conjunctive rules, with possibility of "don't care" values, allowing that the number of antecedents may change in different rules. The classification of a new example implies searching those rules whose antecedent is compatible with it and to determine the class agreeing with the maximal number of rules of the same consequent. If no rules have been found, the example is not classified.
- XCS also uses conjunctive rules, with a generality index in each attribute. If the generality index covers the complete domain of a certain attribute, then it obtains a "don't care" value. In order to classify a new example, the rules that have a positive match with it are chosen and each one of them votes according to their fitness and consequent.
- GASSIST-ADI uses CNF type rules, where disjunctions can coexist with conjunctions. The matching process is done by means of decision lists, in which the rules are evaluated from the top of the list to the bottom, until the antecedent matches the example to be classified. There always is a default rule, so no examples remain unclassified.
- HIDER yields hierarchical rules similar to decision lists in the matching process. Some rules may be included within parent rules and the rules are only formed by conjunctions. The rules allow to define open extremes of real intervals and the rule set usually tends to cover all the space of solutions.

Given the differences among the four algorithms, taking into consideration the characteristics of the rules and the matching techniques, the comparison of interpretability measures must be cautiously taken. Although the results indicate that GASSIST-ADI may produce the most interpretable rule sets, its type of rule could be considered less understandable than the ones yielded by Pitts-GIRLA algorithm. Furthermore, it uses decision lists, so a certain rule (except the first in the list) depends on previous rules. On the other hand, a concept could not be learned because it is being considered in the default rule. With regard to HIDER, although both use the same matching technique, the latter can use open intervals in the rules.

The choice of the most interpretable type of rule or rule set is a relative task because it may depend on the usefulness and purpose of the model. This question is out of the scope of the paper, but we want to point out that a statistical analysis of the interpretability of rule sets could be valid when the circumstances permit so.

8 Conclusions

In this paper we have studied the use of statistical techniques in the analysis of the behaviour of GBML algorithms in classification problems, analysing the use of parametric and non-parametric statistical tests.

We have raised the necessity of applying non-parametric tests in the use of GBML algorithms in classification, due to the fact that the initial conditions that guarantee the reliability of the parametric tests are not satisfied in a single data-set analysis.

Non-parametric tests can be used in multiple data-set analysis and allow the comparison between GBML methods and deterministic algorithms. We have shown how to use a Friedman, Iman–Davenport, Bonferroni–Dunn, Holm, Hochberg, and Wilcoxon tests; which on the whole, are a good tool for the analysis of algorithms' performance. We have employed these procedures to carry out a comparison in a case study composed by an experimentation that involves several data-sets and 4 well-known GBML algorithms.

We have checked that different statistical results are obtained when we consider different accuracy measures, such as classification rate and Cohen's kappa. In interpretability analysis, the results cannot predict what is the algorithm which yields the easiest models, due to the fact that the rule sets are different in structure and there are many ways of representing knowledge.

As main conclusion on the use of non-parametric statistical methods for analysing results, we have emphasized the use of the most appropriate test depending on the circumstances and type of comparison. Specifically, we have recommended the use of the Holm and Hochberg procedures since they are the most powerful statistical techniques for multiple comparisons.

Acknowledgments The study was supported by the Spanish Ministry of Science and Technology under Project TIN-2005-08386-C05-01. J. Luengo holds a FPU scholarship from Spanish Ministry of Education and Science. The authors are very grateful to the anonymous reviewers for their valuable suggestions and comments to improve the quality of this paper. We also are very grateful to Prof. Bacardit, Prof. Bernadó-Mansilla and Prof. Aguilar-Ruiz for providing the KEEL software with the GASSIST-ADI, XCS and HIDER algorithms, respectively.

A genetic algorithms in classification

Here we will give a wider description of all the methods employed in our work, regarding their main components, structure and operation of each one of them.

For more details about the methods explained here, please refer to the corresponding references.

Pitts-GIRLA algorithm. The Pittsburgh genetic interval rule learning algorithm (Pitts-GIRLA) (Corcoran and Sen 1994) is a GBML method which makes use of the Pittsburgh approach in order to perform a classification task. Two real variables indicate the minimum and maximum value of the attribute, where a "don't care" condition may occur if the maximum value is lower than the minimum value.

This algorithm employs three different operators: modified simple (one point) crossover, creep mutation and simple random mutation.

XCS algorithm. XCS (Wilson 1995) is a LCS that evolves online a set of rules that describe the feature space accurately. In the following we will present in detail the different components of this algorithm:

1. **Interaction with the environment**: In keeping with the typical LCS model, the environment provides as input to the system a series of sensory situations $\sigma(t) \in \{0,1\}^L$, where L is the number of bits in each situation. In response the system executes actions $\alpha(t) \in \{a_1,\ldots,a_n\}$ upon the environment. Each action results in a scalar reward $\rho(t)$.
2. **A classifier in XCS**: XCS keeps a population of classifiers which represent its knowledge about the problem. Each classifier is a condition-action-prediction rule having the following parts: the condition $C \in \{0,1,\#\}^L$, the action $A \in \{a_1,\ldots,a_n\}$ and the prediction p. Furthermore, each classifier keeps certain additional parameters such as the prediction error ε, the fitness F, the experience exp, the time stamp ts, the action set size as and the numerosity.
3. **The different sets**: There are four different sets that need to be considered in XCS: the population $[P]$, the match set $[M]$, the action set $[A]$ and the previous action set $[A_{-1}]$.

The result of this algorithm is that the knowledge is represented by a set of rules or classifiers with a certain fitness. When classifying unseen examples, each rule that matches the input votes according its prediction and fitness. The most voted class is chosen to be the output.

GASSIST algorithm. Genetic Algorithms based claSSIfier sySTem (GASSIST) (Bacardit and Garrell 2007) is a Pittsburgh style classifier system based on GABIL (De Jong et al. 1993) from where it has taken the semantically correct crossover operator. The main features of this classifier system are presented as follows:

1. **General operators and policies**
   - *Matching strategy* The matching process follows a "if ... then ... else if ... then ..." structure, usually called decision lists (Rivest 1987).
   - *Mutation operators* When an individual is selected for mutation a random gene is chosen inside its chromosome to be mutated.
2. **Control of the individuals length**: This control is achieved using two different operators:
   - *Rule deletion:* This operator deletes the rules of the individuals that do not match any training example.
   - *Selection bias using the individual size:* Tournament selection is used, where the criterion of the tournament is given by an operator called "hierarchical selection", defined as follows:
     - If $|accuracy_a – accuracy_b| < threshold$ then:
       - If $length_a < length_b$ then a is better than b.
       - If $length_a > length_b$ then b is better than a.
       - If $length_a = length_b$ then we will use the general case.
     - Otherwise, we use the general case: we select the individual with higher fitness.
3. **Knowledge representations**
   - *Rule Representations for symbolic or discrete attributes:* It uses the GABIL (De Jong et al. 1993) representation for this kind of attributes.
   - *Rule Representations for real-valued attributes* For GASSIST-ADI, the representation is based on the Adaptive Discretization Intervals rule representation (Bacardit and Garrell 2003; Bacardit 2004).

HIDER algorithm. HIerarchical DEcision Rules (HIDER) (Aguilar-Ruiz et al. 2000), produces a hierarchical set of rules, which may be viewed as a Decision List. In order to extract the rule-list a real-coded GA is employed in the search process. The elements of this procedure are described below.

1. **Coding**: Each rule is represented by an individual (chromosome), where two genes define the lower and upper bounds of the rule attribute.
2. **Algorithm**: The algorithm is a typical sequential covering GA. It chooses the best individual of the evolutionary process, transforming it into a rule which is used to eliminate data from the training file (Venturini 1993).

   Initially, the set of rules *R* is empty, but in each iteration a rule is included in *R*. In each iteration, the training file is reduced, eliminating those examples that have been covered by the description of the rule *r*, independently of its class.

The GA main operators are defined in the following:

(a) *Initialization*: First, an example is randomly selected from the training file for each individual of the population. Afterwards, an interval to which the example belongs is obtained.

(b) *Crossover*: The crossover works as follows: let $[l_i^j, u_i^j]$ and $[l_i^k, u_i^k]$ be the intervals of two parents, *j* and *k*, for the same attribute *i*. From these parents one child is generated by selecting values that satisfy the expression: $l \in [\min(l_i^j, l_i^k), \max(l_i^j, l_i^k)]$ and $u \in [\min(u_i^j, u_i^k), \max(u_i^j, u_i^k)]$.

(c) *Mutation*: a small value is subtracted or added, depending on whether it is the lower or the upper boundary, respectively.

(d) Fitness function: The fitness function *f* considers a two-objective optimization, trying to maximize the number of correctly classified examples and to minimize the number of errors.

References

Aguilar-Ruiz JS, Giráldez R, Riquelme JC (2000) Natural encoding for evolutionary supervised learning. IEEE Trans Evol Comput 11(4):466–479

Alcalá-Fdez J, Sánchez L, García S, del Jesus MJ, Ventura S, Garrell JM, Otero J, Romero C, Bacardit J, Rivas VM, Fernández JC, Herrera F (2009) KEEL: a software tool to assess evolutionary algorithms to data mining problems. Soft Comput 13(3):307–318

Alpaydin E (2004) Introduction to machine learning, vol 452. MIT Press, Cambridge

Anglano C, Botta M (2002) NOW G-Net: learning classification programs on networks of workstations. IEEE Trans Evol Comput 6(13):463–480

Asuncion A, Newman DJ (2007) UCI Machine Learning Repository. University of California, School of Information and Computer Science, Irvine, CA. http://www.ics.uci.edu/~mlearn/MLRepository.htm

Bacardit J (2004) Pittsburgh genetic-based machine learning in the data mining era: representations, generalization and run-time, Dept. Comput. Sci., University Ramon Llull, Barcelona, Spain

Bacardit J, Garrell JM (2003) Evolving multiple discretizations with adaptive intervals for a pittsburgh rule-based learning classifier system. In: Proceedings of the genetic and evolutionary computation conference (GECCO'03), vol 2724. LNCS, Germany, pp 1818–1831

Bacardit J, Garrell JM (2004) Analysis and improvements of the adaptive discretization intervals knowledge representation. In: Proceedings of the genetic and evolutionary computation conference (GECCO'04), vol 3103. LNCS, Germany, pp 726–738

Bacardit J, Garrell JM (2007) Bloat control and generalization pressure using the minimum description length principle for Pittsburgh approach learning classifier system. In: Kovacs T, Llorá X, Takadama K (eds) Advances at the frontier of learning classifier systems, vol 4399. LNCS, USA, pp 61–80

Barandela R, Sánchez JS, García V, Rangel E (2003) Strategies for learning in class imbalance problems. Pattern Recognit 36(3):849–851

Ben-David A (2007) A lot of randomness is hiding in accuracy. Eng Appl Artif Intell 20:875–885

Bernadó-Mansilla E, Garrell JM (2003) Accuracy-based learning classifier systems: models, analysis and applications to classification tasks. Evol Comput 11(3):209–238

Bernadó-Mansilla E, Ho TK (2005) Domain of competence of XCS classifier system in complexity measurement space. IEEE Trans Evol Comput 9(1):82–104

Clark P, Niblett T (1989) The CN2 induction algorithm. Machine Learn 3(4):261–283

Cohen JA (1960) Coefficient of agreement for nominal scales. Educ Psychol Meas 37–46

Corcoran AL, Sen S (1994) Using real-valued genetic algorithms to evolve rule sets for classification. In: Proceedings of the IEEE conference on evolutionary computation, pp 120–124

De Jong KA, Spears WM, Gordon DF (1993) Using genetic algorithms for concept learning. Machine Learn 13:161–188

Demšar J (2006) Statistical comparisons of classifiers over multiple data sets. J Machine Learn Res 7:1–30

Drummond C, Holte RC (2006) Cost curves: an improved method for visualizing classifier performance. Machine Learn 65(1):95–130

Freitas AA (2002) Data mining and knowledge discovery with evolutionary algorithms, vol 264. Springer, Berlin

Grefenstette JJ (1993) Genetic algorithms for machine learning, vol 176. Kluwer, Norwell

Guan SU, Zhu F (2005) An incremental approach to genetic-algorithms-based classification. IEEE Trans Syst Man Cybern B 35(2):227–239

Hekanaho J (1998) An evolutionary approach to concept learning. Dissertation, Department of Computer Science, Abo akademi University, Abo, Finland

Hochberg Y (2000) A sharper bonferroni procedure for multiple tests of significance. Biometrika 75:800–803

Holm S (1979) A simple sequentially rejective multiple test procedure. Scand J Stat 6:65–70

Huang J, Ling CX (2005) Using AUC and accuracy in evaluating learning algorithms. IEEE Trans Knowl Data Eng 17(3):299–310

Iman RL, Davenport JM (1980) Approximations of the critical region of the Friedman statistic. Commun Stat 18:571–595

Jiao L, Liu J, Zhong W (2006) An organizational coevolutionary algorithm for classification. IEEE Trans Evol Comput 10(1):67–80

Koch GG (1970) The use of non-parametric methods in the statistical analysis of a complex split plot experiment. Biometrics 26(1):105–128

Landgrebe TCW, Duin RPW (2008) Efficient multiclass ROC approximation by decomposition via confusion matrix perturbation analysis. IEEE Trans Pattern Anal Mach Intell 30(5):810–822

Lim T-S, Loh W-Y, Shih Y-S (2000) A comparison of prediction accuracy, complexity, and training time of thirty-three old and new classification algorithms. Machine Learn 40(3):203–228

Markatou M, Tian H, Biswas S, Hripcsak G (2005) Analysis of variance of cross-validation estimators of the generalization error. J Machine Learn Res 6:1127–1168

Rivest RL (1987) Learning decision lists. Machine Learn 2:229–246

Sheskin DJ (2006) Handbook of parametric and nonparametric statistical procedures, vol 1736. Chapman & Hall/CRC, London/West Palm Beach

Shaffer JP (1995) Multiple hypothesis testing. Ann Rev Psychol 46:561–584

Sigaud O, Wilson SW (2007) Learning classifier systems: a survey. Soft Comput 11:1065–1078

Sokolova M, Japkowicz N, Szpakowicz S (2006) Beyond accuracy, F-score and ROC: a family of discriminant measures for performance evaluation. In: Australian conference on artificial intelligence, vol 4304. LNCS, Germany, pp 1015–1021

Tan KC, Yu Q, Ang JH (2006) A coevolutionary algorithm for rules discovery in data mining. Int J Syst Sci 37(12):835–864

Tulai AF, Oppacher F (2004) Multiple species weighted voting - a genetics-based machine learning system. In: Proceedings of the genetic and evolutionary computation conference (GECCO'03), vol 3103. LNCS, Germany, pp 1263–1274

Venturini G (1993) SIA: a supervised inductive algorithm with genetic search for learning attributes based concepts. In: Proceedings of the machine learning ECML'93, vol 667. LNAI, Germany, pp 280–296

Wilson SW (1994) ZCS: a zeroth order classifier system. Evol Comput 2:1–18

Wilson SW (1995) Classifier fitness based on accuracy. Evol Comput 3(2):149–175

Witten IH, Frank E (2005) Data mining: practical machine learning tools and techniques, 2nd edn, vol 525. Morgan Kaufmann, San Francisco

Wright SP (1992) Adjusted *p*-values for simultaneous inference. Biometrics 48:1005–1013

Youden W (1950) Index for rating diagnostic tests. Cancer 3:32–35

Zar JH (1999) Biostatistical analysis, vol 929. Prentice Hall, Englewood Cliffs