Guided Rule Discovery in XCS for High-Dimensional Classification Problems

Mani Abedini and Michael Kirley

Department of Computer Science and Software Engineering
The University of Melbourne, Australia
{mabedini,mkirley}@csse.unimelb.edu.au

Abstract. XCS is a learning classifier system that combines a reinforcement learning scheme with evolutionary algorithms to evolve a population of classifiers in the form of condition-action rules. In this paper, we investigate the effectiveness of XCS in high-dimensional classification problems where the number of features greatly exceeds the number of data instances – common characteristics of microarray gene expression classification tasks. We introduce a new guided rule discovery mechanisms for XCS, inspired by feature selection techniques commonly used in machine learning. The extracted feature quality information is used to bias the evolutionary operators. The performance of the proposed model is compared with the standard XCS model and a number of well-known machine learning algorithms using benchmark binary classification tasks and gene expression data sets. Experimental results suggests that the guided rule discovery mechanism is computationally efficient, and promotes the evolution of more accurate solutions. The proposed model performs significantly better than comparative algorithms when tackling high-dimensional classification problems.

1 Introduction

Learning classifier systems (LCS) combine machine learning with metaheuristics to build models that learn to solve a particular classification problem (see [8,18] for detailed reviews). The eXtended classifier system (XCS) is a well-known LCS that maintains a population of classifiers [21]. Each classifier consists of a *condition-action-prediction* rule with an associated fitness value, which represents the accuracy of the predicted reward. Through an iterative learning process, the population of classifiers evolves. A key step in this iterative process is the rule discovery component that creates new classifiers that are added to the bounded population pool.

One of the challenges when designing a LCS, is to build flexibility and robustness into the model such that it is capable of handling large scale data mining and classification problems. Consider a prototypical high-dimensional data set, such as a microarrray gene expression data set, that has several thousands genes (features) but only a small number of samples [24]. Standard XCS implementations, and many other machine learning algorithms for that matter, are typically

D. Wang and M. Reynolds (Eds.): AI 2011, LNAI 7106, pp. 1–10, 2011.

less effective in this high-dimensional space. It is difficult to effectively explore the solution space and build an appropriate classification model. In such circumstances, feature selection pre-processing can be used [14]. Such approaches can reduce the negative effect of the irrelevant features on the learning task, and speed up the learning process significantly.

In this paper, a new guided rule discovery mechanisms is proposed for XCS for high-dimensional classification problems. Our model (GRD-XCS), is inspired by feature selection techniques commonly used in machine learning. Typically, filtering techniques assess the relevance of features in the data set, with the low-scoring features subsequently being removed. A subset of the "more important" features is then presented as input to the classification algorithm. However, in our model the filtering process is used to build a probability distribution that biases the evolutionary operators encapsulated in the rule discovery component of XCS. This probability distribution can be thought of as a mask that biases the uniform crossover and mutation operators. This flexible approach is scalable, thus the enhanced XCS can be used to tackle high-dimensional classification tasks without reducing the dimensionality of the data set.

To test the efficacy of the new GRD-XCS, a systematic set of experiments were carried out using benchmark binary classification tasks and a suite of gene expression microarray data sets. The proposed model was compared to XCS and a range of well-known machine learning algorithms. The results show that the new guided rule discovery mechanisms leads to improved accuracy, particularly for high dimensional binary classification problems.

The remainder of this paper is organized as follows: In section 2 we present background material related to XCS and related work. In section 3 we describe the guided rule discovery mechanism in detail. The experiments and results appear in Section 4. We conclude the paper by summarizing the contributions and identifying the possible future directions.

2 Background

2.1 XCS Overview

In this section, we provide a brief overview of the functionality XCS. See [21,8,18] for detailed discussion of LCS in general.

XCS maintains a population of classifiers. Each classifier consists of a *condition-action-prediction* rule, which maps input features to the output signal (or class). A ternary representation of the form 0,1,# (where # is don't care) for the condition and 0,1 for the action can be used. In addition, real encoding can also be used [22].

At each time step, the classifier system receives a problem instance – input in the form of a vector of features – which requires a decision, that is an action to be performed next. A *match set* $[M]$ is created consisting of rules (classifiers) that can be "triggered" by the given data instance. A covering operator is used to create new matching classifiers when $[M]$ is empty. A prediction array $[PA]$ is calculated for $[M]$ that contains an estimation of the corresponding rewards

Algorithm 1. High level overview of XCS

Require: Input data:σ, Population:$[\Delta]$
  repeat
    $\sigma \leftarrow env$
    $[M] \leftarrow GetMatchSet(\sigma,[\Delta])$
    $[PA] \leftarrow CreatePredictionArray([M])$
    $act \leftarrow SelectAnAction([PA])$
    $[A] \leftarrow CreateActionSet([M], act)$
    $R \leftarrow ExecutingActionOnENV(act)$
    $[A] \leftarrow UpdateSet([A], R)$
    $[\Delta] \leftarrow RuleDiscovery([A],[\Delta])$
  until terminating conditions are not met

for each of the possible actions. Based on the values in the prediction array, an action, *act*, is selected. Those classifiers which support the predicted action make up the *Action Set* $[A]$. In response to *act*, the reinforcement mechanism is invoked and the prediction (p), prediction error, accuracy, and fitness of the $[A]$ classifiers are updated. The corresponding numerical reward is distributed to the rules accountable for it so as to improve the estimates of the action values (see algorithm 1).

The rule discovery module is a key component of XCS. During the evolutionary process, fitness-proportionate selection is applied to $[A]$. Standard evolutionary operators, *uniform crossover* and *mutation*, are then applied to the selected individuals. In addition, a second mutation-style operator – the *don't care* operator – is used to randomly modify a condition part of a classifier to the don't care value #. The newly created offspring (classifiers) are then added to the bounded population. A form of niching is then used to determine if the offspring survive in the population and/or which of the old members of the population are deleted to make room for the new classifiers (offspring). A subsumption mechanism combines similar classifiers and a randomized deletion mechanism removes classifiers with a low fitness from the population.

It is important to note that the XCS population consists of a set unique *macro-classifiers* – a set of classifiers that have same condition part and same action part. Every macro-classifier has an associated *numerosity* value, which records how many instances of that specific classifier actually exists in the population.

2.2 Related Work

It is well documented in the evolutionary computation literature that the implementation of the genetic operators can influence the trajectory of the evolving population. However, there has been a paucity of studies focussed specifically on the impact of selected evolutionary operator implementations in LCS. We briefly discuss some of the key studies related to XCS/LCS below.

In one of the first studies focussed on the rule discovery component specifically for XCS, Butz et al. [6] have shown that uniform crossover can ensure successful learning in many tasks. In subsequent work, Butz et al. [5] introduced an informed crossover operator, which extended the usual uniform operator such that exchanges of effective building blocks occurred. This approach helped to avoid the over-generalization phenomena inherent in XCS [13]. In other work, Bacardit et al. [3] customized the GAssist crossover operator to switch between the standard crossover or a new simple crossover, SX, randomly. SX is a heuristic selection approach to take a minimum number of rules from the parents (more than two), which can obtain maximum accuracy. Morales-Ortigosa et al. [16] have also proposed a new XCS crossover operator, BLX, which allowed for the creation of multiple offspring with a diversity parameter to control differences between offspring and parents. Finally, in a more comprehensive overview paper, Morales-Ortigosa et al. [17] present a systematic experimental analysis of the rule discovery component in LCS. Subsequently, they developed crossover operators to enhance the discovery component based on evolution strategies with significant performance improvements.

Other work focussed on biased evolutionary operators in LCS include the work of Luis et al. [12], who introduced a hybridized GA - Tabu Search (GA-TS) method that employed modified mutation and crossover operators. Here, the operator probabilities were tuned by analyzing all the fitness values of individuals during the evolution process. Wang et. al. [20] used *information gain* as the fitness function in a GA. They reported improved results when comparing their model to other machine learning algorithms. Recently, Huerta et al. [4] combined *linear discriminant analysis* with a GA to evaluate the fitness of individuals and associated discriminate coefficients for crossover and mutation operators. Moore et al. [15] argue that the biasing of the initial population, based on expert knowledge preprocessing, should lead to improved performance in LCS. In their approach, a statistical method, Tuned ReliefF, was used to determine the dependencies between features to seed the initial population. A modified fitness function and a new guided mutation operator based on features dependency was also introduced, leading to significantly improved performance.

3 Model

The motivation behind the design and development of the GRD-XCS was to improve classifier performance especially for high-dimensional classification problems. Our goal was to make the overall task computationally faster, without degrading accuracy. To meet this goal, GRD-XCS introduces a probabilistically guided rule discovery mechanism for XCS. Here, two distinct phases are used. In the pre-processing phase, each feature is examined independently of all others and assigned a rank. This rank is then used when generating the probability distribution used to bias the evolutionary operators, which are deployed during the second phase – the generation of classifiers in XCS.

The evolution process is regulated by a rule discovery probability vector, RDP, which controls the bitwise *crossover*, *mutation* and *don't care* operators. Each value in the vector is associated with the corresponding feature, and is allocated a value in the range [0, 1.0]. The RDP values are determined based on a ranked *Information Gain* (IG) measure [11]. The IG measure is defined as *entropy* reduction:

$$IG = H(C) - H(C|f_i) \tag{1}$$

where H represent entropy, $F = \{f_0, f_1, ...f_i, ..., f_n\}$ is the feature set, and C the classes in this context. Entropy is a measure to quantify the information content, it is calculated using the formula:

$$H(C) = \sum_{j \in C} p_j \log_2 p_j \tag{2}$$

where p_j is the probability of having j in C, and the conditional entropy is calculated as:

$$H(C|f_i) = \sum_{j \in C} p_j \log_2 H(C|f_i = j) \tag{3}$$

The actual values in the RDP vector are calculated based on the IG values as described below:

$$RDP_i = \begin{cases} \frac{1-\gamma}{\Lambda} \times (\Lambda - i) + \gamma & \text{if } i \leq \Lambda \\ \\ \xi & \text{otherwise} \end{cases} \tag{4}$$

where i represents the rank index in ascending order for the selected features. The probability values associated with the other features are given a very low value (ξ). Thus, all features have a chance to participate in the rule discovery process. However, the Λ-top ranked features have a greater chance of being selected (see figure 1).

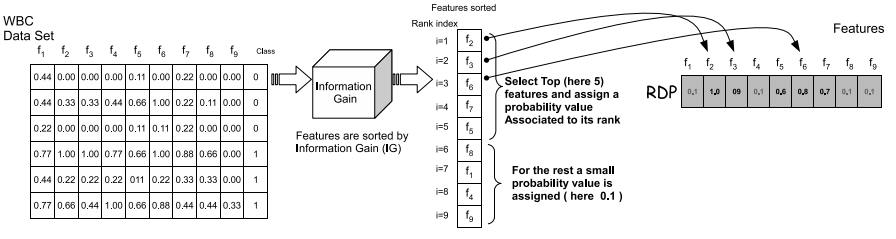


Fig. 1. Information Gain is used to rank the features. The top Λ features (in this example Λ =5) are selected and allocated relatively large probability values $\in [\gamma, 1]$. The RDP vector maintains these values. The highest ranked feature value is set to 1.0. Other features receive smaller values relative to their rank (in this example γ =0.5). Features that are not selected based on information gain, are assigned very small probability values (in this example $\xi = 0.1$).

GRD-XCS uses the probability values recorded in the RDP vector in the preprocessing phase to bias the evolutionary operators used in the rule discovery phase of XCS. The modified algorithms describing the *crossover*, *mutation* and *don't care* operators in GRD-XCS are very similar to standard XCS operators:

GRD-XCS *crossover* operator: The crossover operator is a hybrid uniform /n-point function. Here, an additional check of each feature is carried out before exchange of genetic material. If $rand() < RDP[i]$ then feature i is swapped between the selected parents.

GRD-XCS *mutation* operator: Uses the RDP vector to determine if feature i undergoes a mutation, if the feature was randomly selected to be mutated.

GRD-XCS *don't care* operator: In this special mutation operator, the values in the RDP vector are used in the reverse order. That is, if the feature i has been selected to be mutated and $rand() < (1 - RDP[i])$, then feature i is changed to # (don't care).

The application of the RDP reduces the crossover and mutation probabilities for "uninformative" features. However, it increases the don't care operator probability for the same feature. Therefore, the more informative features (based on the Information Gain measure in this case) should appear in rules more often than the uninformative ones.

4 Experiments

A series of independent experiments were conducted to verify if the guided rule discovery mechanism for XCS was able to find accurate classifiers. In particular, we wished to establish if the proposed model had statistically significantly improved accuracy values when compared to the standard XCS across a suite of benchmark classification problems. All experiments have been conducted with N-fold cross validation over 100 trials. The average accuracy values for specific scenarios have been reported using the Area Under the ROC Curve (AUC) value. Paired t-tests are used for statistical comparisons.

4.1 Data Sets

Table 1 lists the data set characteristics used in the experiments. Two different types of data sets have been used in these experiments: data sets with either a small number of features with many samples (low-dimensional data set) obtained from the UCI [1] machine learning repository; and DNA Microarray Gene Expression data sets with a large number of features with few samples (high-dimensional data set). Gene expression profiles provide important insights into, and further our understanding of, biological processes. As such, they are key tools used in medical diagnosis, treatment and drug design [23].

Table 1. Data set details

Data Set	#Instances	#Features	Cross Validation	Reference
Low-dimensional data sets (UCI examples)				
Pima	768	8	10	[1]
WBC	699	9	10	[1]
Hepatits	155	19	10	[1]
Parkinson	197	23	10	[1]
High-dimensional data sets (Microarray DNA gene expression)				
Breast Cancer	22	3226	3	[10]
Colon Cancer	62	2000	10	[2]
Leukemia Cancer	72	7129	10	[9]
Prostate Cancer	136	12600	10	[19]

4.2 Parameters

Default parameter values as recommended in [7] have been used to configure the underlying XCS model in GRD-XCS. However, in the case of the high-dimensional data sets it was necessary to scale-up the population to 2000 individuals as compared with 1000 individuals for the low-dimensional data sets. The number of iterations was capped at 5000.

The guided rule discovery module relies on the ranking of feature. Here, we have limited the ranking to the top 64 features (Λ= 64) for the gene expression profiles classifications. For the low dimensional data sets, all features were used when building the probability models (see section 3). The limits used in probability values calculations in equation 4 were γ=0.5 and ξ=0.1.

4.3 Results

Tables 2 and 3 lists accuracy results for the low-dimensional data sets and high-dimensional gene expression data sets respectively. Results for GRD-XCS, the standard XCS and a range of machine learning algorithms (using default Weka implementations) are listed for each data set. The bold value in each column indicates the highest mean accuracy value over all trials. The †symbol indicates that the result for the classifier listed in the row was significantly better than the GRD-XCS result based on a paired t-test ($p < 0.05$).

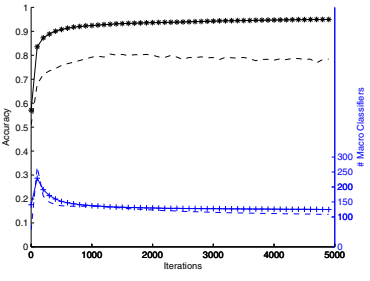
For the low-dimensional data sets considered, the GRD-XCS results were better than the standard XCS, although this difference was not always statistically significant. When compared against the other machine learning algorithms, the GRD-XCS results were somewhat mixed. GRD-XCS performed best for one data set only – the Parkinson data set. In contrast, for the high-dimensional data sets the results for GRD-XCS were significantly better than the other machine learning algorithms based on paired t-tests. A direct comparison between GRD-XCS and the standard XCS clearly illustrates that the guided rule discovery mechanisms leads to improved performance.

Table 2. AUC results for low-dimensional data sets

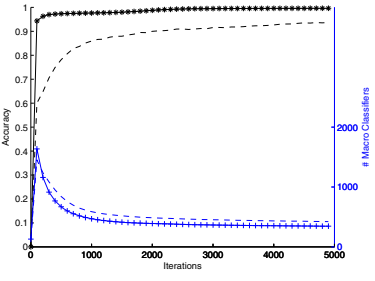
Classifier	Pima	WBC	Hepatit	Parkinson
j48	0.75 ± 0.01 †	0.94 ± 0.01	0.60± 0.04	0.78 ± 0.03
SVM	0.71 ± 0.01 †	0.96± 0.01	0.75 ± 0.02	0.75 ± 0.01
Naive Bayes Classifier	**0.81 ± 0.01** †	**0.98 ± 0.01** †	**0.84± 0.01** †	0.85 ± 0.01
NBTree	0.80 ± 0.01 †	**0.98 ± 0.01** †	0.76± 0.03	0.88 ± 0.02
One Rule	0.65 ± 0.01	0.90 ± 0.01	0.56± 0.02	0.77 ± 0.01
Random Forest	0.79 ± 0.01 †	**0.98 ± 0.01** †	0.81± 0.02 †	**0.94 ± 0.01** †
XCS	0.70± 0.03	**0.98 ± 0.01** †	0.81 ± 0.11 †	0.93 ± 0.08
GRD-XCS	0.72± 0.03	**0.98 ± 0.01**	0.82 ± 0.13	**0.94 ± 0.07**

Table 3. AUC results for high-dimensional data sets

Classifier	Breast Cancer	Colon Cancer	Leukemia	Prostate Cancer
j48	0.43 ± 0.09	0.76 ± 0.04	0.79 ± 0.03	0.79 ± 0.02
SVM	0.63 ± 0.06	0.81 ± 0.03	0.97 ± 0.01	0.91 ± 0.01
Naive Bayes	0.55 ± 0.02	0.64 ± 0.02	0.98 ± 0.01	0.58 ± 0.01
NBTree	0.66 ± 0.03	0.75 ± 0.05	0.97 ± 0.01	0.90 ± 0.01
One Rule	0.42 ± 0.05	0.66 ± 0.04	0.82 ± 0.02	0.81 ± 0.02
Random Forest	0.67 ± 0.09	0.82 ± 0.03	0.92 ± 0.02	0.88 ± 0.01
XCS	0.66 ± 0.12	0.74 ± 0.18	0.93 ± 0.11	0.83 ± 0.09
GRD-XCS	**0.74 ± 0.19**	**0.86 ± 0.14**	**0.99 ± 0.01**	**0.93 ± 0.05**



(a) Parkinson data set



(b) Leukemia data set

Fig. 2. Accuracy and the number of macro-classifier versus the number of iteration comparisons for the base line XCS model and GRD-XCS for representative low-dimensional (dotted lines) and high-dimensional (solid lines) data sets

To further explore the efficacy of the proposed guided rule discovery enhancements, figure 2 plots time series values for overall accuracy and the number of macro-classifiers in the evolving population for both the GRD-XCS and the standard XCS for a representative low-dimensional dat set and a high-dimensional gene expression data set. Space constraints preclude the inclusion of plots for all data sets, however, the general trends for other data sets is qualitatively similar. There is a correlation between the accuracy of the model and the number of

macro-classifiers in the population for high-dimensional classification problems examined. As expected, the number of unique classifiers (individuals) in the population for both XCS and GRD-XCS decreases over time. However, GRD-XCS typically maintains a smaller number of macro-classifiers.

5 Conclusions

In this paper, we have introduced a guided rule discovery component designed specifically for XCS when tackling high-dimensional classification problems. Here, a filtering or feature ranking process is used to build a probabilistic model of feature importance in a pre-processing phase. This probability distribution is then used to bias the evolutionary operators in the underlying XCS model. Comprehensive numerical simulations have shown that the guided rule discovery mechanism improves the performance of XCS in terms of accuracy and more generally in terms of classifier diversity in the population, particularly for high-dimensional classification problems.

We have limited the feature ranking process in this study to simple entropy analysis. In future work, we will explore the use of alternative metrics to rank the features. In the case of microarray data, there is scope to incorporate domain specific knowledge when building the probabilistic rule discovery mask. A second research direction that we will consider will focus on designing a distributed and parallel deployment of the scalable model.

References

1. UCI Machine Learning Repository, `http://archive.ics.uci.edu/ml/`
2. Alon, U., Barkai, N., Notterman, D.A., Gishdagger, K., Ybarradagger, S., Mackdagger, D., Levine, A.J.: Broad patterns of gene expression revealed by clustering analysis of tumor and normal colon tissues probed by oligonucleotide arrays. Proc. of the National Academy of Sciences of the USA 96, 6745–6750 (1999)
3. Bacardit, J., Krasnogor, N.: Smart crossover operator with multiple parents for a Pittsburgh learning classifier system. In: Proceedings of the 8th Conference on GECCO, pp. 1441–1448. ACM (2006)
4. Bonilla Huerta, E., Hernández Hernández, J.C., Hernández Montiel, L.A.: A New Combined Filter-Wrapper Framework for Gene Subset Selection with Specialized Genetic Operators. In: Martínez-Trinidad, J.F., Carrasco-Ochoa, J.A., Kittler, J. (eds.) MCPR 2010. LNCS, vol. 6256, pp. 250–259. Springer, Heidelberg (2010), `http://dx.doi.org/10.1007/978-3-642-15992-3_27`
5. Butz, M., Pelikan, M., Lloral, X., Goldberg, D.E.: Automated global structure extraction for effective local building block processing in XCS. Evolutionary Computation 14(3), 345–380 (2006)
6. Butz, M.V., Goldberg, D.E., Tharakunnel, K.: Analysis and improvement of fitness exploitation in XCS: bounding models, tournament selection, and bilateral accuracy. Evol. Comput. 11, 239–277 (2003)
7. Butz, M.V., Wilson, S.W.: An Algorithmic Description of XCS. In: Lanzi, P.L., Stolzmann, W., Wilson, S.W. (eds.) IWLCS 2000. LNCS (LNAI), vol. 1996, pp. 253–274. Springer, Heidelberg (2001)

8. Fernandndez, A., Garcianda, S., Luengo, J., Bernado-Mansilla, E., Herrera, F.: Genetics-based machine learning for rule induction: State of the art, taxonomy, and comparative study. IEEE Transactions on Evolutionary Computation 14(6), 913–941 (2010)
9. Golub, T.R., Slonim, D.K., Tamayo, P., Huard, C., Gaasenbeek, M., Mesirov, J.P., Coller, H., Loh, M.L., Downing, J.R., Caligiuri, M.A., Bloomfield, C.D.: Molecular classification of cancer: class discovery and class prediction by gene expression monitoring. Science 286, 531–537 (1999)
10. Hedenfalk, I., Duggan, D., Chen, Y., Radmacher, M., Bittner, M., Simon, R., Meltzer, P., Gusterson, B., Esteller, M., Kallioniemi, O.P., Wilfond, B., Borg, A., Trent, J.: Gene-Expression profiles in hereditary breast cancer. N. Engl. J. Med. 344(8), 539–548 (2001)
11. Isabelle Guyon, M.N., Gunn, S., Zadeh, L. (eds.): Feature Extraction, Foundations and Applications. Springer, Heidelberg (2006)
12. Jose-Revuelta, L.M.S.: A Hybrid GA-TS Technique with Dynamic Operators and its Application to Channel Equalization and Fiber Tracking. I-Tech Education and Publishing (2008)
13. Lanzi, P.L.: A Study of the Generalization Capabilities of XCS. In: Bäck, T. (ed.) Proceedings of the 7th International Conference on Genetic Algorithms, pp. 418–425. Morgan Kaufmann (1997)
14. Liu, H., Motoda, H.: Computational Methods of Feature Selection. Data Mining and Knowledge Discovery Series. Chapman & Hall/CRC (2007)
15. Moore, J.H., White, B.C.: Exploiting Expert Knowledge in Genetic Programming for Genome-Wide Genetic Analysis. In: Runarsson, T.P., Beyer, H.-G., Burke, E.K., Merelo-Guervós, J.J., Whitley, L.D., Yao, X. (eds.) PPSN 2006. LNCS, vol. 4193, pp. 969–977. Springer, Heidelberg (2006)
16. Morales-Ortigosa, S., Orriols-Puig, A., Bernadó-Mansilla, E.: New Crossover Operator for Evolutionary Rule Discovery in XCS. In: 8th International Conference on Hybrid Intelligent Systems, pp. 867–872. IEEE Computer Society (2008)
17. Morales-Ortigosa, S., Orriols-Puig, A., Bernadó-Mansilla, E.: Analysis and improvement of the genetic discovery component of XCS. In: International Joint Conference on Hybrid Intelligent Systems, vol. 6, pp. 81–95 (April 2009)
18. Orriols-Puig, A., Casillas, J., Bernadó-Mansilla, E.: Genetic-based machine learning systems are competitive for pattern recognition. Evolutionary Intelligence 1, 209–232 (2065), doi:10.1007/s12065-008-0013-9
19. Singh, D., Febbo, P.G., Ross, K., Jackson, D.G., Manola, J., Ladd, C., Tamayo, P., Renshaw, A.A.: Gene expression correlates of clinical prostate cancer behavior. Cancer Cell 1, 203–209 (2002)
20. Wang, P., Weise, T., Chiong, R.: Novel evolutionary algorithms for supervised classification problems: an experimental study. Evolutionary Intelligence 4(1), 3–16 (2011)
21. Wilson, S.W.: Classifier Fitness Based on Accuracy. Evolutionary Computation 3(2), 149–175 (1995), `http://prediction-dynamics.com/`
22. Wilson, S.W.: Get Real! XCS with Continuous-Valued Inputs. In: Lanzi, P.L., Stolzmann, W., Wilson, S.W. (eds.) IWLCS 1999. LNCS (LNAI), vol. 1813, pp. 209–222. Springer, Heidelberg (2000)
23. Wu, F.-X., Zhang, W., Kusalik, A.: On Determination of Minimum Sample Size for Discovery of Temporal Gene Expression Patterns. In: First International Multi-Symposiums on Computer and Computational Sciences, pp. 96–103 (2006)
24. Zhang, Y., Rajapakse, J.C.: Machine Learning in Bioinformatics, 1st edn. Wiley Series in Bioinformatics (2008)