Stochastic Semantic-Based Multi-objective Genetic Programming Optimisation for Classification of Imbalanced Data

Edgar Galván-López[1(✉)], Lucia Vázquez-Mendoza[2], and Leonardo Trujillo[3]

[1] Department of Computer Science, National University of Ireland Maynooth, Maynooth, Ireland
edgar.galvan@nuim.ie

[2] School of Social Sciences and Philosophy, Trinity College Dublin, Dublin, Ireland
lucyvaz@gmail.com

[3] Posgrado en Ciencias de la Ingeniería, Instituto Tecnológico de Tijuana, Tijuana, Mexico
leonardo.trujillo@tectijuana.edu.mx

Abstract. Data sets with imbalanced class distribution pose serious challenges to well-established classifiers. In this work, we propose a stochastic multi-objective genetic programming based on semantics. We tested this approach on imbalanced binary classification data sets, where the proposed approach is able to achieve, in some cases, higher recall, precision and F-measure values on the minority class compared to C4.5, Naive Bayes and Support Vector Machine, without significantly decreasing these values on the majority class.

1 Introduction

Classification is one of the most studied and challenging problems in machine learning and data mining. The task consists in predicting a value of an attribute, so-called class, based on the values of other attributes. The importance of this research problem can be understood by the fact that many real-world problems have been stated as classification problems, including image recognition [24] and medical diagnosis [1].

Multiple classification algorithms have been proposed and successfully used on classifications tasks, including decision trees, neural networks, support vector machines, Bayesian networks and nearest neighbour classifiers. However, data sets with an imbalanced class distribution (i.e., when the learning examples from at least one class are very uncommon) have posed challenges to well-established classifiers that work under the assumption of a relatively well-balanced class distribution [3,20].

E. Galván-López—Research conducted during Galván's stay at TAO, INRIA and LRI, CNRS & U. Paris-Sud, Université Paris-Saclay, France.

DOI: 10.1007/978-3-319-62428-0_22

In binary classification, the class with the smaller number of examples is called the positive or minority class, whereas the other class is referred as the negative or majority class. Good accuracy on the positive class can in many cases be more important than the accuracy on the negative class. However, it has been reported that very often when trying to increase the accuracy on one class, the accuracy on the other tends to decrease [2]. This is because these two objectives are usually in conflict.

A natural form to deal with this scenario is to use evolutionary multi-objective optimisation [5]. Evolutionary Algorithms (EAs) [8], also known as Evolutionary Computation systems, are influenced by the theory of evolution by natural selection. These algorithms have been with us for some decades and are very popular because they have proven competitive in the face of challenging problem features, such as deceptiveness and multiple local optima, among other characteristics [7]. They are also popular because EAs have been successfully used in many different problems, ranging from the automated optimisation of game controllers [12,19] to the automated design of circuits [10,17].

The idea behind EAs is to automatically generate (nearly) optimal solutions by "evolving" potential solutions (individuals forming a population) over time (generations) by using bio-inspired operators (e.g., crossover, mutation). Briefly, the evolutionary process includes the initialisation of the population $P(0)$ at generation $g = 0$. The population consists of a number of individuals which represent potential solutions to the problem. At each iteration or generation (g), every individual within the population ($P(g)$) is evaluated using a *fitness function* that determines its fitness (i.e., how good or bad an individual is). Then, a selection mechanism takes place to stochastically pick the fittest individuals from the population. Some of the selected individuals are modified by genetic operators and the new population $P(g+1)$ at generation $g+1$ is created. The process stops when some halting condition is satisfied. Further details on how these stochastic optimisation algorithms work can be found in [8].

The goal of this work is to use a Evolutionary Multi-objective Optimisation (EMO) method [4], in specific a MO Genetic Programming (MOGP) paradigm to *automatically* design classifiers that can correctly classify imbalanced data. Moreover, MOGP has the potential to yield programs (classifiers) that are readable to a human expert. It also has the advantage that from a single run, the approach is able to produce multiple results allowing the user to select the most convenient for the final application (e.g., a high accuracy on the positive class regardless of the accuracy achieved on the negative class). The novelty of this work is the use of semantics in the proposed MOGP. The motivation for doing so is because the specialised EA literature indicates that the adoption of semantics in a canonical Genetic Programming (GP) [22] system yields better results compared to a GP system without semantics. We apply the proposed approach, described in detail in Sect. 2, on a number of imbalanced classification problems of different nature and complexity, and we used these problems as 'they are', that is, we do not try to balance the classes with a sampling method.

The rest of this paper is organised as follows. In Sect. 2, we introduce our proposed approach. Section 3 provides details on the experimental setup used in this research. Section 4 shows and discusses the results found in this study, and finally, conclusions are drawn in Sect. 5.

2 Proposed Approach

2.1 Evolutionary Multi-objective Optimisation

Multi-objective optimisation (MO) aims to simultaneously optimise several objectives. No single solution exists when these objectives are in conflict and the optimal trade-offs between these must be sought. This idea is captured in the Pareto dominance relation: a solution will dominate another one if it is at least as good as the other solution on all the objectives and better on at least one. Similarly, solutions are non-dominated if they are not dominated by any solution in the set of candidate solutions.

Evolutionary multi-objective optimisation (EMO) [5] is based on the following: by replacing the single-objective selection steps, based on the comparison of fitness values, by some Pareto-based comparison, one turns a single-objective evolutionary optimisation algorithm into a multi-objective evolutionary optimisation algorithm, but because Pareto dominance is not a total order, some additional criterion must be used so as to allow the comparison of any pair of points of the search space.

In NSGA-II [6], the Pareto-based comparison uses the non-dominated sorting procedure: all non-dominated individuals in the population are assigned Rank 1 and removed from the population, the remaining non-dominated individuals are assigned Rank 2, and so on. The secondary criterion is the *crowding distance* that promotes diversity among the individuals having the same Pareto rank: in objective space, for each objective, the individuals in the population are ordered, and the partial crowding distance for each of them is the difference in fitness between its two immediate neighbours. The crowding distance is the sum over all objectives of these partial crowding distances. Intuitively, it can be seen as the Manhattan distance between the extremal vertices of the largest hypercube containing the point at hand and no other point of the population. Selecting points with the largest crowding distance amounts to favour the low-density regions of the objective space, thus favouring diversity.

In general, NSGA-II proceeds as follows. From a given population of size N, N offspring are created using standard variation operators (crossover and mutation). Parents and offspring are merged and the resulting population, of size $2N$, is ordered using non-dominated sorting and the crowding distance as a secondary criterion. The best N individuals according to this ranking are selected to survive at the next generation.

2.2 Semantics in Genetic Programming

GP has been successfully used in various different challenging problems (see Koza's article on human competitive results for a comprehensive review [23]).

Despite its proven success, it also suffers from some limitations and researchers have been interested in making GP more robust or reliable by studying various elements of the search process (e.g., neutrality [16,18,25], locality [13,14]). One of these elements that has relatively recently attracted the attention of researchers is the study of semantics in GP, resulting in a dramatic increase in the number of related publications (e.g., [11,15,26,27]).

Semantics is a broad concept that has been studied in different fields (e.g., natural language, psychology), making it hard to give a precise definition of the concept. Moreover, the way semantics have been adopted in canonical GP varies significantly (see [27] for a summary of works carried out on semantics in GP).

In this work we adopted the popular use of semantics in GP from recent related works [26], where researchers have defined it as the output a program produces when it is evaluated on the complete set of training instances, as adopted in various previous works [11,26].

Thus, two individuals (in this case, two expressions/classifiers) are regarded as semantically different if their output vectors are different, they are considered semantically similar, otherwise. Similarly, a semantic distance is defined as the number of different outputs between two individuals. Commonly, when computing the semantic distance, two outputs are considered different if their absolute difference is greater than a given threshold [11,26]. In this work, we set the threshold at 0.5. The speciliased literature indicates that promoting semantically different individuals tends to yield better results.

2.3 Incorporating Semantics into MOGP

In this work, we incorporated semantics into the core of the NSGA-II algorithm. This algorithm, as briefly described before, relies on two main elements: a ranking system and a crowding distance.

We incorporated semantics by replacing this last element, the crowing distance, with a semantic-based indicator called Semantic-based Crowding Distance (SCD). This is computed the following way: a *pivot* is chosen, being the individual from the first Pareto front (Rank 1) that is the furthest away from the other individuals of this front using the crowding distance. For each point, its semantic distance with the pivot is computed. Similarly to the crowding distance, the SCD is computed as the average of the semantic distance differences with its closest neighbours in each direction. The higher values of this SCD are favored during the selection step of NSGA-II. This allows us to have a set of individuals that are spread in semantic space, thereby promoting semantic diversity, the same way NSGA-II promotes diversity ('spreadness') in objective space. Hereafter, this variant of NSGA-II will be called Distance based on Semantics (DBS).

3 Experimental Design

We used binary imbalanced classification problems taken from the well-known UCI Machine Learning Repository [1]. These problems are of different nature

Table 1. Binary imbalanced classification data sets used in our research. Table adapted from [2].

Data set	Classes	Number of examples			Imb.	Features	
	Positive/negative (brief description)	Total	Positive	Negative	Ratio	No.	Type
Ion	Good/bad (ionsphere radar signal)	351	126 (35.8%)	225 (64.2%)	1:3	34	Real
Spect	Abnormal/normal (cardiac tom. scan)	267	55 (20.6%)	212 (79.4%)	1:4	22	Binary
Yeast_1	mit/other (protein sequence)	1482	244 (16.5%)	1238 (83.5%)	1:6	8	Real
Yeast_2	me3/other (protein sequence)	1482	163 (10.9%)	1319 (89.1%)	1:9	8	Real
Derma	Pityriasis/other (dermatology diseases)	358	20 (5.5%)	338 (94.5%)	1:17	34	Integer
Balanced	Balanced/unbalanced (balance scale)	625	49 (7.8%)	576 (92.2%)	1:12	4	Integer
Abalone_1	9/18 (biology of abalone)	731	42 (5.75%)	689 (94.25%)	1:17	8	Real
Abalone_2	19/other (biology of abalone)	4177	32 (0.77%)	4145 (99.23%)	1:130	8	Real

Table 2. Confusion matrix.

	Predicted object	Predicted non-object
Actual object	True Positive (TP)	False Negative (FN)
Actual non-object	False Positive (FP)	True Negative (TN)

and complexity (e.g., from a few number of features up to dozens of them, and these include real, binary and integer-value features, from low to relatively high imbalanced data). Table 1 shows the sizes of both negative (majority) and positive (minority) classes for each of the problems used in this work along with some details[1]. We used the data 'as is' (e.g., we did not try to balance the classes with sampling techniques). For each data set, half of the data (with the same class balance than in the whole dataset) was used as a training set and the rest as a test set. All reported results, discussed in the following section, are on the latter.

To automatically generate classifiers via MOGP, we defined our terminal and function in the same manner as in [2]. The terminal set consists of problem features from each data set. The function set consists of a conditional IF function and the typical four standard arithmetic operators, and so, the function set is defined by $F = \{if, +, -, *, /\}$, where the latter operator is protected division, which returns the numerator if the denominator is zero. The IF function takes three arguments: if the first is negative, the second argument is returned, otherwise the last argument is returned. These functions are used to build a classifier (e.g., mathematical expression) that returns a single value for a given input (data example to be classified). This number is mapped onto a set of class labels using zero as the class threshold. In our studies, an example is assigned to the positive class if the output of the classifier is greater or equal to zero. It is assigned to the negative class, otherwise.

The most common way to measure fitness in classification is the overall classification accuracy: using the four outcomes for binary classification shown in Table 2 and the minority class being the positive class, we have that the accuracy

[1] For the abalone data sets, we substituted F, M and I by 1, 2 and 3, respectively.

Table 3. Total number of independent runs executed for our experiments.

Description	Value
Number of independent runs	50
Number of data sets	8
Number of algorithms (NSGA-II and NSGA-II DBS)	2
Total number of runs	800*

*800 = 50 * 8 * 2.

is given by $Acc = \frac{TP+TN}{TP+TN+FP+FN}$. The drawback with Acc is that it can bias the evolutionary search towards the majority class as pointed out in [28] via [2]. Also, as pointed out in [2], a better approach is to treat each objective (class) 'separately' using MOGP. To this end, we used in the fitness function of our MOGP the true positive rate ($TPR = \frac{TP}{TP+FN}$) and the true negative rate ($TNR = \frac{TN}{TN+FP}$) to measure the accuracy for the positive and negative class, respectively.

The experiments were conducted using a steady state approach with tournament selection of size 7. To initialise our population, we used the ramped half-and-half method (initial and final depth set at 1 and 5, respectively). To control bloat (a dramatic increase of programs' length without observing a corresponding improvement in fitness) a maximum depth of 8 was specified (where root was considered of depth 0) or a maximum number of 800 nodes was used. Crossover and mutation rates were 60% and 40%, respectively. Elitism for NSGA-II is not required given the nature of the algorithm. We compare our proposed approached, NSGA-II DBS with canonical NSGA-II, as well with three well-known machine learning algorithms (i.e., Naive Bayes, C4.5 and SVMs).

Because of the stochasticity associated to MOGP, we performed extensive independent runs (we executed 800 runs in total, see Table 3 for details). Runs were stopped when the maximum number of generations was reached.

4 Experimental Results

4.1 Precision, Recall and F-Measure

We compared our results against three very popular machine learning algorithms: Naive Bayes (NB), C4.5 and SVMs. To this end, we use the well-known weka tool [21]. The comparison of results is conducted by computing the F-measure which reflects the performance on both precision and recall. Recall is defined by

$$R = TP_{rate} = \frac{TP}{TP + FN} \tag{1}$$

and precision is defined by

$$P = \frac{TP}{TP + FP}. \tag{2}$$

Table 4. F-measure comparison on the positive (minority) class.

Data set	NSGA-II		NSGA-II DBS		NB		C4.5		SVM	
	Positive	Negative	Positive	Negative	Positive	Negative	Positive	Negative	Positive	Negative
Precision										
Ion	0.838 ± 0.097	0.645 ± 0.025	0.873 ± 0.075	0.657 ± 0.018	0.689	0.881	0.939	0.865	0.930	0.826
Spect	0.560 ± 0.021	0.491 ± 0.052	0.563 ± 0.020	0.496 ± 0.052	0.614	0.763	0.63	0.759	0.644	0.797
Yeast$_1$	0.452 ± 0.055	0.832 ± 0.016	0.455 ± 0.055	0.833 ± 0.019	0.685	0.918	0.685	0.888	0.737	0.845
Yeast$_2$	0.546 ± 0.064	0.879 ± 0.006	0.541 ± 0.047	0.879 ± 0.006	0.827	0.964	0.817	0.951	0.917	0.890
Derma	0.963 ± 0.073	0.955 ± 0.002	0.970 ± 0.060	0.955 ± 0.002	1	0.917	1	0.917	1	1
Balanced	0.636 ± 0.346	0.893 ± 0.021	0.643 ± 0.322	0.899 ± 0.020	0	0.917	0	0.917	0	0.917
Abalone$_1$	0.199 ± 0.097	0.941 ± 0.018	0.29 ± 0.074	0.945 ± 0.012	0.186	0.974	0.333	0.953	0	0.948
Abalone$_2$	0.017 ± 0.006	0.992 ± 0.004	0.014 ± 0.005	0.979 ± 0.084	0.019	0.995	0	0.993	0	0.993
Recall										
Ion	0.832 ± 0.045	0.896 ± 0.076	0.818 ± 0.052	0.926 ± 0.050	0.81	0.795	0.730	0.973	0.635	0.973
Spect	0.716 ± 0.077	0.550 ± 0.062	0.711 ± 0.078	0.558 ± 0.060	0.66	0.725	0.642	0.750	0.717	0.738
Yeast$_1$	0.753 ± 0.033	0.799 ± 0.057	0.750 ± 0.039	0.801 ± 0.061	0.587	0.945	0.49	0.956	0.111	0.992
Yeast$_2$	0.925 ± 0.036	0.894 ± 0.027	0.928 ± 0.033	0.893 ± 0.024	0.736	0.978	0.637	0.980	0.121	0.998
Derma	0.995 ± 0.035	0.998 ± 0.005	0.995 ± 0.035	0.998 ± 0.003	1	0.994	1	0.994	1	1
Balanced	0.922 ± 0.940	0.868 ± 0.156	0.893 ± 0.177	0.889 ± 0.127	0	1	0	1	0	1
Abalone$_1$	0.835 ± 0.900	0.760 ± 0.910	0.836 ± 0.111	0.800 ± 0.096	0.579	0.861	0.105	0.988	0	1
Abalone$_2$	0.688 ± 0.186	0.660 ± 0.156	0.692 ± 0.150	0.69 ± 0.154	0.333	0.873	0	1	0	1
F-measure										
Ion	0.831 ± 0.053	0.749 ± 0.043	**0.843 ± 0.050**	0.768 ± 0.027	0.745	0.836	0.821	0.916	0.755	0.893
Spect	0.627 ± 0.032	0.518 ± 0.056	0.626 ± 0.034	0.525 ± 0.055	0.636	0.744	0.636	0.755	**0.679**	0.766
Yeast$_1$	0.562 ± 0.037	0.815 ± 0.038	0.562 ± 0.037	0.816 ± 0.042	**0.632**	0.931	0.507	0.921	0.193	0.913
Yeast$_2$	0.684 ± 0.048	0.887 ± 0.016	0.681 ± 0.037	0.886 ± 0.014	**0.779**	0.971	0.716	0.965	0.214	0.941
Derma	0.977 ± 0.049	0.976 ± 0.002	0.981 ± 0.041	0.976 ± 0.002	0.957	0.997	0.957	0.997	**1**	1
Balanced	**0.703 ± 0.285**	0.873 ± 0.095	0.700 ± 0.271	0.890 ± 0.074	0	0.957	0	0.957	0	0.957
Abalone$_1$	0.303 ± 0.099	0.835 ± 0.093	**0.330 ± 0.089**	0.863 ± 0.063	0.282	0.914	0.160	0.970	0	0.973
Abalone$_2$	0.032 ± 0.011	0.782 ± 0.118	0.028 ± 0.009	0.744 ± 0.146	**0.035**	0.930	0	0.996	0	0.996

The F-measure represents a harmonic mean between these two elements. Hence, a high value on the F-measure denotes that both recall and precision are high. The F-measure is defined by

$$F - measure = \frac{2}{1/R + 1/P}. \tag{3}$$

We compare precision, recall and F-measure results on the five different learning methods (NSGA-II, NSGA-II DBS, NB, C4.5 and SVM) for each of the eight data sets used (Table 1). For the Pareto front (PF) found by every independent run of the NSGA-II and NSGA-II DBS, we took the best pair of true positive rate and true negative rate measured in terms of the Euclidean distance between a point in the PF against the optimal solution (100% accuracy). These results are shown in Table 4.

Due to space constraints, we focus our attention on the F-measure values on the positive (minority) class. The best (higher) values are highlighted in boldface. The MOGP approaches (NSGA-II and NSGA-II DBS) are better compared to the other three algorithms in three problems (Ion, Balanced and Abalone$_1$). In two of these problems (Ion and Abalone$_1$), NSGA-II DBS is better than NSGA-II (Balanced). When comparing these two algorithms on the latter problem, we can see that they are very close to each other (the difference is only 0.003) but NSGA-II DBS is better when one takes into account also the negative (majority) class.

When we focus our attention on the results of the other learning algorithms we can see that the worst performance is achieved by C4.5. On the other hand,

MOGP Approach	Evolved Classifier
NSGA-II	`IF + IF IF * f3 f1 / f1 f2 * f4 f1 * * f2 f2 IF f4 f2 f1 / - f1 f3 * f2 f4 - IF IF f1 f1 f2 * * f2 f2 IF f4 f2 f1 / f3 f2 * / f4 f2 * f3 f2 - * * / f3 f2 * f1 f4 - * / f3 f2 - f1 f1 IF * f1 f1 / f2 f3 * f1 f4 / IF IF f1 f1 f2 * / f4 f2 * f3 f2 / f3 f2 f2 IF + / * f2 * f2 f3 - f1 f4 / / - f1 f3 - f2 f1 + f4 f2 * / f2 f2 / f1 f4 + * IF * f1 f1 / f2 f3 * f1 f4 / / - f1 * f2 f3 - f2 f1 + f4 f2 * * f2 f3 - f1 f1`
NSGA-II DBS	`IF - * - / f1 IF f2 f2 f2 f3 * - f3 f3 / / + - f3 f3 - f1 f3 IF - f2 f4 - f3 f3 + f1 f1 f1 - / + f2 f4 f2 / + f3 f1 IF f2 f3 f1 - IF IF f4 f1 f1 - - f2 f4 + f2 f1 - - f4 f2 / + f2 f4 f2 / + f3 f1 IF f4 f1 f1 * - + + f3 f4 / + f2 f4 f2 IF + f4 f4 * f1 f2 / f1 f1 - / + f2 f4 f2 / + f3 f1 IF f4 f1 f1`

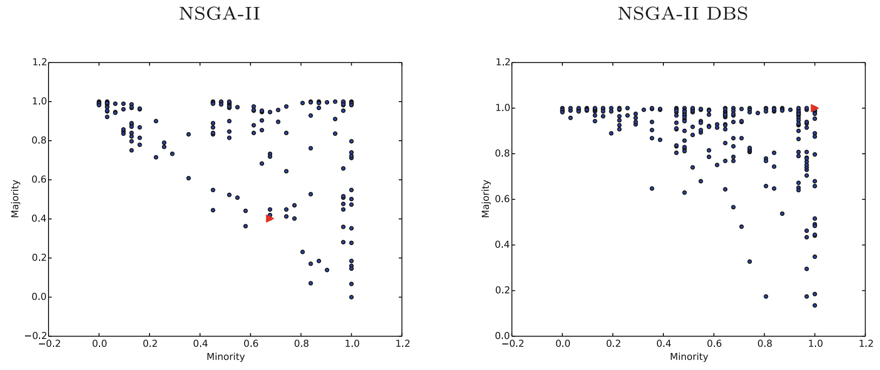

Fig. 1. Accuracy of all unique evolved solutions on the balanced data set found by NSGA-II and NSGA-II DBS shown in the left and right-hand side of the figure, respectively. The red triangles indicate the evolved classifiers found by these approaches, shown at the top of the figure. `f` denotes a feature, where `f1` denotes the first feature of the data set. (Color figure online)

Naive Bayes achive good results on the Yeast and Abalone$_2$ problems, and SVMs achive good results on the Spect and Derma problems.

Overall, our MOGP approaches are very competitive to well-established machine learning classifiers. In fact, they show excellent performance on a specific problem, discussed next.

4.2 MOGP Classifiers

We now focus our attention on the results produced by NSGA-II and NSGA-II DBS. To do so, we analyse the balanced data set, where the results obtained by these two approaches on the positive (minority) are impressive without decreasing significantly their precision, recall, F-measure on the negative (majority) call. F-measure values on the positive class are 0.703 ± 0.285 and 0.700 ± 0.271, when using NSGA-II and NSGA-II DBS, respectively, against the other algorithms where they failed completely (i.e., F-measure on the positive class is 0).

Figure 1 shows all the unique solutions found by NSGA-II and NSGA-II DBS on this data set over all 50 independent runs (notice that a single run yields

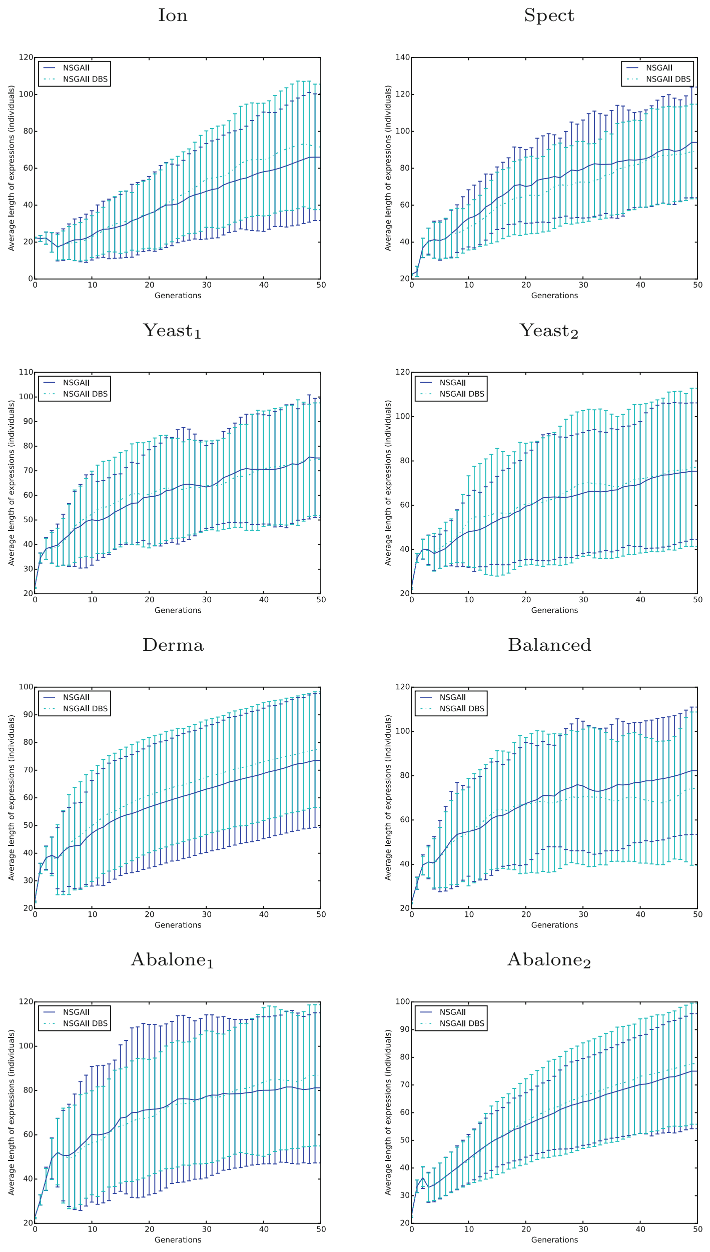

Fig. 2. Average length of evolved solutions, indicated in the y-axis, over 50 generations, indicated in the x-axis, over 50 independent runs using NSGA-II (blue line) and NSGA-II DSB (cyan line). (Color figure online)

multiple results). Some elements are worth mentioning. Our proposed approach, NSGA-II DBS, is able to produce more unique solutions compared to the NSGA-II as noticed by the number of scatter points in the figure. Furthermore, there are more solutions close to the optimal solutions (100% accuracy) compared to NSGA-II. The evolved classifiers, shown at the top of Fig. 1, correspond to those marked by the red triangles shown in the scatter plots (bottom of the figure). These classifiers were selected based on the shortest solutions found by these two algorithms.

4.3 Length of Evolved Classifiers

Very often it is necessary to account for a classifier that can be interpreted. One element that helps towards this is the size of the classifier [9]. Figure 2 shows the average size of the classifier, along with the standard deviation, found by both MOGP approaches for each of eight data sets used in this work.

There are two elements worth discussing: the length of the evolved classifier gets larger as evolution continues (i.e., in all cases the classifiers are significantly larger at the end of the evolutionary search compare to when search starts). The other element is that both algorithms tend to produce evolved classifier of similar length. This means that there is no extra cost when using our proposed approach, NSGA-II DBS, in terms of analysing the evolved classifer.

5 Conclusions

In this work, we proposed a stochastic multi-objective genetic programming approach based on the semantics of programs, for classification of imbalanced data. The results are highly encouraging: the proposed approach (NSGA-II DBS) is able to classify data from the positive class without significantly decreasing the recall, precision and F-measure values on the negative class.

The MOGP approaches used in this work have some advantages: a single run offers multiple solutions and the final user could decide which is the best suited according to his/her needs (e.g., a 100% accuracy on the positive class could be good enough regardless of the accuracy obtained on the negative class). We have also learnt that the proposed approach is able to produce more solutions compared to the traditional NSGA-II. Morever, it is possible to interpret the evolved classifiers thanks to the nature of the representation (encoding of a solution) used by the MOGP approaches. Finally, we also have learnt that the NSGA-II DBS produces classifiers that are of similar length compared to the well-known NSGA-II.

Acknowledgements. EGL's research is funded by an ELEVATE Fellowship, the Irish Research Council's Career Development Fellowship co-funded by Marie Curie Actions. EGL would like to thank the TAO group at INRIA Saclay France for hosting him during the outgoing phase of the fellowship. LVM thanks the SSSP for hosting her during her research visit at TCD. The authors would like to thank the reviewers for their comments that helped us to improve our work. EGL would also like to thank E. Mezura-Montes, O. Ait Elhara and M. Schoenauer for their earlier involvement in this work.

References

1. Asuncion, A., Newman, D.: UCI machine learning repository (2007)
2. Bhowan, U., Johnston, M., Zhang, M., Yao, X.: Reusing genetic programming for ensemble selection in classification of unbalanced data. IEEE Trans. Evol. Comput. **18**(6), 893–908 (2014)
3. Chawla, N.V., Japkowicz, N., Kotcz, A.: Editorial: special issue on learning from imbalanced data sets. SIGKDD Explor. Newsl. **6**(1), 1–6 (2004)
4. Coello, C.A.C.: Evolutionary multi-objective optimization: a historical view of the field. IEEE Comput. Intell. Mag. **1**(1), 28–36 (2006)
5. Deb, K., Kalyanmoy, D.: Multi-Objective Optimization Using Evolutionary Algorithms. Wiley, New York (2001)
6. Deb, K., Pratap, A., Agarwal, S., Meyarivan, T.: A fast and elitist multiobjective genetic algorithm: NSGA-II. IEEE Trans. Evol. Comput. **6**, 182–197 (2002)
7. Eiben, A.E., Smith, J.: From evolutionary computation to the evolution of things. Nature **521**, 476–482 (2015)
8. Eiben, A.E., Smith, J.E.: Introduction to Evolutionary Computing. Springer, Berlin (2003). doi:10.1007/978-3-662-05094-1
9. Freitas, A.A.: Data Mining and Knowledge Discovery with Evolutionary Algorithms, 1st edn. Springer, Berlin (2002). doi:10.1007/978-3-662-04923-5
10. Galván-López, E.: Efficient graph-based genetic programming representation with multiple outputs. Int. J. Autom. Comput. **5**(1), 81–89 (2008)
11. Galván-López, E., Cody-Kenny, B., Trujillo, L., Kattan, A.: Using semantics in the selection mechanism in genetic programming: a simple method for promoting semantic diversity. In: 2013 IEEE Congress on Evolutionary Computation, pp. 2972–2979, June 2013
12. Galván-López, E., Fagan, D., Murphy, E., Swafford, J., Agapitos, A., O'Neill, M., Brabazon, A.: Comparing the performance of the evolvable π grammatical evolution genotype-phenotype map to grammatical evolution in the dynamic Ms. Pac-Man environment. In: 2010 IEEE Congress on Evolutionary Computation (CEC), pp. 1–8, July 2010
13. Galván-López, E., McDermott, J., O'Neill, M., Brabazon, A.: Defining locality in genetic programming to predict performance. In: IEEE Congress on Evolutionary Computation, pp. 1–8. IEEE (2010)
14. Galván-López, E., McDermott, J., O'Neill, M., Brabazon, A.: Towards an understanding of locality in genetic programming. In: Proceedings of the 12th Annual Conference on Genetic and Evolutionary Computation, GECCO 2010, NY, USA, pp. 901–908. ACM (2010)
15. Galván-López, E., Mezura-Montes, E., Ait ElHara, O., Schoenauer, M.: On the use of semantics in multi-objective genetic programming. In: Handl, J., Hart, E., Lewis, P.R., López-Ibáñez, M., Ochoa, G., Paechter, B. (eds.) PPSN 2016. LNCS, vol. 9921, pp. 353–363. Springer, Cham (2016). doi:10.1007/978-3-319-45823-6_33
16. Galván-López, E., Poli, R.: Some steps towards understanding how neutrality affects evolutionary search. In: Runarsson, T.P., Beyer, H.-G., Burke, E., Merelo-Guervós, J.J., Whitley, L.D., Yao, X. (eds.) PPSN 2006. LNCS, vol. 4193, pp. 778–787. Springer, Heidelberg (2006). doi:10.1007/11844297_79
17. López, E.G., Poli, R., Coello, C.A.C.: Reusing code in genetic programming. In: Keijzer, M., O'Reilly, U.-M., Lucas, S., Costa, E., Soule, T. (eds.) EuroGP 2004. LNCS, vol. 3003, pp. 359–368. Springer, Heidelberg (2004). doi:10.1007/978-3-540-24650-3_34

18. Galván-López, E., Poli, R., Kattan, A., O'Neill, M., Brabazon, A.: Neutrality in evolutionary algorithms.. What do we know? Evol. Syst. **2**(3), 145–163 (2011)
19. Galván-López, E., Swafford, J.M., O'Neill, M., Brabazon, A.: Evolving a Ms. PacMan controller using grammatical evolution. In: Di Chio, C., et al. (eds.) EvoApplications 2010. LNCS, vol. 6024, pp. 161–170. Springer, Heidelberg (2010). doi:10.1007/978-3-642-12239-2_17
20. Guo, X., Yin, Y., Dong, C., Yang, G., Zhou, G.: On the class imbalance problem. In: 2008 Fourth International Conference on Natural Computation, vol. 4, pp. 192–201, October 2008
21. Hall, M., Frank, E., Holmes, G., Pfahringer, B., Reutemann, P., Witten, I.H.: The weka data mining software: an update. SIGKDD Explor. Newsl. **11**(1), 10–18 (2009)
22. Koza, J.R.: Genetic Programming: On the Programming of Computers by Means of Natural Selection. The MIT Press, Cambridge (1992)
23. Koza, J.R.: Human-competitive results produced by genetic programming. Genet. Program. Evolvable Mach. **11**(3–4), 251–284 (2010)
24. Kubat, M., Holte, R.C., Matwin, S.: Machine learning for the detection of oil spills in satellite radar images. Mach. Learn. **30**(2), 195–215 (1998)
25. Poli, R., Galván-López, E.: The effects of constant and bit-wise neutrality on problem hardness, fitness distance correlation and phenotypic mutation rates. IEEE Trans. Evol. Comput. **16**(2), 279–300 (2012)
26. Uy, N.Q., Hoai, N.X., O'Neill, M., McKay, R.I., Galván-López, E.: Semantically-based crossover in genetic programming: application to real-valued symbolic regression. Genet. Program. Evolvable Mach. **12**(2), 91–119 (2011)
27. Vanneschi, L., Castelli, M., Silva, S.: A survey of semantic methods in genetic programming. Genet. Program. Evolvable Mach. **15**(2), 195–214 (2014)
28. Weiss, G.M., Provost, F.: Learning when training data are costly: The effect of class distribution on tree induction. J. Artif. Int. Res. **19**(1), 315–354 (2003)