An iterated greedy algorithm for improving the generation of synthetic patterns in imbalanced learning*

F.J. Maestre-García[1], C. García-Martínez[1], M. Pérez-Ortiz[2], and P.A. Gutiérrez[1]**

[1] Department of Computer Science and Numerical Analysis, University of Córdoba, Campus de Rabanales, C2 building, 14071 Córdoba, Spain.
{i12magaf,cgarcia,pagutierrez}@uco.es
[2] Department of Quantitative Methods, Universidad Loyola Andalucía, Córdoba, Spain. mariaperez@uloyola.es

Abstract. Real-world classification datasets often present a skewed distribution of patterns, where one or more classes are under-represented with respect to the rest. One of the most successful approaches for alleviating this problem is the generation of synthetic minority samples by convex combination of available ones. Within this framework, adaptive synthetic (ADASYN) sampling is a relatively new method which imposes weights on minority examples according to their learning complexity, in such a way that difficult examples are more prone to be over-sampled. This paper proposes an improvement of the ADASYN method, where the learning complexity of these patterns is also used to decide which sample of the neighbourhood is selected. Moreover, to avoid sub-optimal results when performing the random convex combination, this paper explores the application of an iterative greedy algorithm which refines the synthetic patterns by repeatedly replacing a part of them. For the experiments, six binary datasets and four over-sampling methods are considered. The results show that the new version of ADASYN leads to more robust results and that the application of the iterative greedy metaheuristic significantly improves the quality of the generated patterns, presenting a positive effect on the final classification model.

Keywords: Over-sampling, imbalanced classification, ADASYN, iterative greedy algorithm, metaheuristics

1 Introduction

Learning from imbalanced data represents one of the current challenges in machine learning. In classification domains, imbalanced distributions occur when

* This work has been partially subsidised by the TIN2014-54583-C2-1-R, TIN2015-70308-REDT, and TIN2014-55252-P projects of the Spanish Ministerial Commission of Science and Technology (MINECO, Spain) and FEDER funds (EU).
** Corresponding author

one or more classes have a significantly higher a-priori-probability [16,17]. The difficulty of learning from imbalanced data is that classifiers will often assume that the classes are equally represented in the dataset [26]. Consequently, standard classifiers will be biased towards the majority class, significantly harming the performance of the minority one [8]. Nonetheless, in most cases, rare objects (or minority class samples) will be of great interest and should be the focus of machine learning algorithms [15], e.g. in financial engineering, where it is crucial to detect fraudulent credit card activities from a pool of large transactions [3]. Note however, that an uneven data distribution is not the only factor that hinders the learning in these cases [16,17], the complexity of the data or the size of the training set being also determining factors.

Imbalanced learning is an active field, and there is a wide range of techniques proposed for improving the minority class sensitivity. For a detailed survey of these, we refer the reader to [16]. Two main groups of methods can be emphasized: 1) data preprocessing techniques, where the class priors are changed by under-sampling the majority class, over-sampling the minority one [2,4] or modifying class priors by changing class labels [6]; or 2) specific learners, where the classifier is forced to pay more attention to the minority class [7,29]. Hybrid methods can also be found in the literature and have shown an outstanding performance for this matter. These methods usually combine data and algorithm-level techniques[8,21], e.g. traditional ensemble methods (such as bagging and boosting) [8] which combine resampling with multiple learning models. Although these techniques were not originally proposed to address imbalanced problems, their relatively superior performance [8,22,28,30] enabled ensemble methods to gain attention within this topic. Finally, there have been some recent and successful attempts to tackle class imbalance using ranking algorithms [5].

The analysis made in this paper is mainly contextualised on data approaches, as these present several advantages over algorithmic solutions [11], all basically stemming from the fact that this approach does not rely on the reformulation of a certain classification algorithm. The most straightforward resampling idea would be repetitive over-sampling, which simply replicates existing data points (in a similar way to cost-sensitive learning). However, generative sampling is usually preferred, where the sparse data space is populated with new data points, producing ideally a more dense, smooth and uniformly distributed dataset [21]. Although both over-sampling and under-sampling are widely used, some studies [17] emphasize over-sampling methods for complex and highly imbalanced datasets.

Concerning generative sampling, the most popular method is the Synthetic Minority Over-sampling TEchnique (SMOTE) [4], which is based on a random interpolation between minority class data points (between a randomly chosen pattern and one of its k nearest neighbours). This algorithm presents several important handicaps, all of them related to the omission of the majority class in the over-sampling process, which, in some cases, depending on the data distribution, can result in a set of synthetic patterns that lie in the majority class region and that hinders the learning process. Because of this, there have been

different proposals over the years to improve SMOTE: borderline-SMOTE [13], which focuses on sampling only those minority data points close to the classification decision boundary; cluster-oversampling [18], which considers the so-called 'rare' regions, which are resampled individually; or safe-level SMOTE [2] and LN-SMOTE [24], which generate new synthetic examples in the direction of the regions populated by the minority class, to avoid introducing artificial examples within majority class regions.

ADASYN [15] is also a relatively new approach which refines SMOTE by focusing on those patterns with a higher learning complexity (i.e. those closer to the class boundary). As a result, ADASYN adaptively shifts the focus of the classification model towards more difficult examples, showing promising results [15]. Although more complex examples are resampled with a higher probability, the interpolation process of ADASYN still relies on a random selection of one of the k nearest neighbours of the pattern, which, depending on the dataset and the choice of k, can lead to the generation of minority samples in the majority-class region (see Figure 1), i.e. one the main problems with SMOTE still remains partially unsolved when using ADASYN. As can be checked in this Figure, new synthetic patterns could be generated in regions associated to the majority class, depending on the choice of the k parameter for the nearest neighbour analysis. Different k parameters could be chosen for different patterns, which emphasizes the necessity of using an optimisation algorithm.

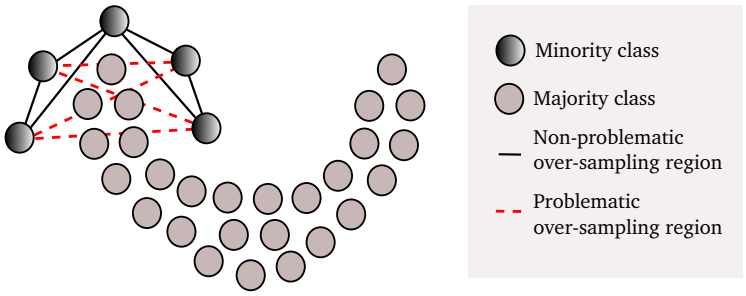


Fig. 1: Imbalanced toy dataset with problems associated to standard oversampling.

In this sense, over-sampling data-level approaches can be formulated as an optimisation problem to further refine the data distribution and avoid the above-mentioned class inconsistencies. In particular, the generation of synthetic samples can be defined as follows: given an imbalanced dataset $D = \{D^m \cup D^M\}$, with D^m being the set of examples of the minority class and D^M the set of examples of the majority class (or examples of other classes; $|D^m| << |D^M|$), the goal is to generate a set of synthetic examples S such that the performance of a classifier trained on $D \cup S$ is maximised. This problem can be effectively

addressed by metaheuristic algorithms. However, to the best of our knowledge, there are very few works in this direction. For example, the work of Ghazikhani et al. [12] presents an evolutionary algorithm that evolves the optimal regions to apply over-sampling. On the other hand, under-sampling can be also approached by using a binary genetic algorithm for selecting which majority patterns are used for learning, and there are different works exploiting this idea [23]. Finally, Wong et al. [32] used an evolutionary algorithm for simultaneously applying under-sampling and over-sampling.

Iterated Greedy (IG) [10,27] is a simple metaheuristic that, in contrast to evolutionary methods, specifically incorporates heuristic information to address the problem at hand. This metaheuristic is based on the application of two operations on the best found solution: *destruction* and *construction*. During the former one, some solution components are removed, generating a partial solution. In the latter, a heuristic greedy procedure completes the partial solution. The process is repeated until a stopping condition is met. This paper proposes an iterated greedy model to improve the solution of over-sampling algorithms in imbalanced datasets. Our method encodes candidate solutions as sets of synthetic samples, where destruction and construction operators apply on. Its heuristic construction operator is an enhanced version of ADASYN that considers the complexity of the patterns also when selecting the neighbour for the convex combination, but could also be applied to other over-sampling strategies. More specifically, from the k nearest neighbours, we select a complex pattern that is, at the same time, close to the pattern to be resampled.

The rest of this paper is structured as follows: Section 2 presents the description of the over-sampling algorithm proposed. Section 3 describes the different experiments considered for evaluating the proposal and the corresponding results. And finally, Section 4 outlines some concluding remarks.

2 Algorithm description

2.1 ANEIGSYN: an Iterated Greedy based Generative Sampler

The model proposed in this paper is an IG that addresses the task of over-sampling as an optimisation problem. Given an imbalanced dataset $D = D^M \cup D^m$, the method searches over the candidate sets of synthetic examples for those from which better classifiers can be induced. The process is divided in four stages:

- *Initialisation:* D is divided into three sets that are subsequently used in their corresponding stages: Tr from which synthetic examples are generated, with 50% of the examples in D, V for guiding the search process by validating the synthetic sets produced, with 30% of the examples in D, and Te for testing the final classifier, with rest of D (20%). Te is never considered for inducing the classifier, but just for evaluation once the search process has finished.
- *Construction:* A candidate set of synthetic examples S is built according to the heuristic described in section 2.2 and using the examples in $Tr \cup V$. This stage generates new synthetic examples, as many as to balance the number

of examples of majority and minority classes in $Tr \cup S$, that are included into S. This stage can also be applied from scratch, i.e. using an empty candidate set S, or over a partial candidate set that is the output of the Destruction stage.

- *Validation:* Any new candidate set S is evaluated according to the performance of the classifier induced from $Tr \cup S$ when predicting the label of the examples in V. The considered performance metric is the geometric mean (see section 3 for more details).
- *Destruction:* The best so far candidate set is partially destroyed by removing 15% randomly chosen examples. Then, this partially destroyed solution becomes the input for the construction stage of the next iteration.

Finally, the process finishes after a maximum number of iterations, 100 in our case, and the best obtained classifier is evaluated on Te.

2.2 Improved version of ADASYN: Adaptive NEighbours Synthetic Sampling (ANESYN)

The heuristic procedure that generates the initial solution and reconstructs every partial one is at the core of every IG algorithm, given that it exploits problem knowledge to produce better solutions than random ones [9].

As stated before, one of the most useful sources of information to generate synthetic examples is the proximity of the solutions to the decision boundary from which synthetic patterns are generated, which can help the learner to approximate it accurately, and how the selected generative solutions are combined, because inappropriate pairings may favour the generation of synthetic examples in non-appropriate regions (see Figure 1). Therefore, our proposed improved version of ADASYN emphasizes these two aspects, producing synthetic solutions from original solutions that are expected to be close to the decision boundary and close to each other. Its main scheme is:

1. The proximity to the decision boundary of each minority example $s_i \in G^m$ is estimated according to the equation ($G = Tr \cup V$):

$$\widehat{r_i} = \frac{r_i}{\sum_j^{|G^m|} r_j}, \text{ where } r_i = |\{s : s \in G^M \text{ and } s \in N_{G \cup S_p}(s_i, k)\}|/k,$$

   where $N_{G \cup S_p}(s_i, k)$ is the set of the k samples from $G \cup S_p$ (i.e. the set of samples used for generation of patterns plus the synthetic patterns already generated but not removed from the solution) closest to s_i.
2. The number of times each minority sample will participate as the main generative sample is computed according to its estimated proximity to the decision boundary, as $g_i = \lfloor \widehat{r_i} \cdot N \rfloor$. Given that we use the floor operator, $\lfloor \cdot \rfloor$, the required number of synthetic patterns is probabilistically completed increasing the number of times minority examples participate according to the remainders ($r_i \cdot N \bmod \sum_j^{|G^m|} r_j$).

3. For each time a minority sample s_i is used for generating a synthetic sample, a secondary original pattern $s_j \in N_G(s_i, k)$ (note that in this case we discard already generated synthetic patterns) is selected in accordance to the probabilities p_j:

$$p_j = \varepsilon_j / \sum_{k}^{|G^m|} \varepsilon_k, \text{ and } \varepsilon_j = \alpha \widehat{r'_j} + (1-\alpha)/d(s_i, s_j),$$

where $\widehat{r'_j}$ is $\widehat{r_j}$ normalised according to the sum of $\widehat{r_j}$ for those patterns in the vicinity $N_G(s_i, k)$, and $d(s_i, s_j)$ is the euclidean distance between the examples, also normalised by the sum of the same quantity for the elements in $N_G(s_i, k)$.

4. Finally, a new synthetic example is generated per pair of generative samples $\{s_i, s_j\}$ as the linear interpolation $s_i + \lambda \cdot (s_j - s_i)$, where λ is a random value in $(0, 1)$.

Note that the proximity of every minority example (Step 2) is computed considering both the original examples and the synthetic ones that have not been removed from the current solution ($G \cup S_p$), so minority examples whose vicinity has been populated with many synthetic ones will be nominated as main generative examples less times than previously.

Additionally, instead of fixing the number of nearest neighbours analysed (see step 1, parameter k) for all datasets, we have considered an adaptive method. The value of k considered for the algorithm is the minimum value for which there is at least one majority sample among the k closest ones for each minority example.

3 Experiments

This section presents the different experiments considered for evaluating the performance of the method proposed. We have tested our algorithm on six real-world machine learning datasets, which can be downloaded from the UCI machine learning repository[3] [20] and Keel repository[4] [1]. The battery of datasets is described in Table 1 and is similar to that considered in the original ADASYN paper [15], but we have also included the `page` dataset. Table 1 includes the imbalance ratio (IR), which is the ratio between the number of patterns of the majority class and that of the minority one.

For the experimental design a holdout procedure is performed 100 times, i.e. 100 training/test partitions are randomly performed, where, as previously discussed, 50% of data is used as training while the remaining 50% is considered for test purposes. All the partitions are stratified, in such a way that the original class distribution of the dataset is approximately maintained in the partitions.

[3] `https://archive.ics.uci.edu/ml/datasets.html`
[4] `http://sci2s.ugr.es/keel/imbalanced.php`

Table 1: Characteristics of the datasets used in this paper

Dataset	# Patterns	# Minority patterns	# Majority patterns	IR	# Attributes
`abalone`	731	42	689	16.40	7
`ionos`	351	126	225	1.79	34
`page`	5472	559	4913	8.79	10
`pima`	768	268	500	1.87	8
`vowel`	990	90	900	10.00	10
`vehicle`	846	199	647	3.25	18

Since the ANEIGSYN algorithm is based on a validation set, the 50% of training data is also partitioned into two stratified sets, as detailed in section 2.1.

All the experiments have been performed using a decision tree as base classifier, given that it is one of the most popular classification methods and has been also considered for evaluating other over-sampling algorithms [13,15]. We have used the implementation included in the Python `scikit-learn` machine learning framework [25], where an optimised version of the CART algorithm is considered for the tree induction. This implementation includes heuristic algorithms, where locally optimal decisions are made at each node. This makes the induction process non deterministic, and different trees can be obtained according to the seed used for random number generation.

Different preprocessing methods have been compared in this paper:

- A decision tree directly obtained from the original dataset, without preprocessing (Original).
- The well known SMOTE algorithm [4].
- The standard ADASYN method, as described in [15], where the difficulty of classifying each minority class pattern is considered for selecting the patterns to be resampled.
- The ANESYN method consists of the application of the improved version of ADASYN, without using the proposed IG algorithm (see section 2.2).
- The application of the improved ADASYN together with the iterative refinement of the generated interpolation (ANEIGSYN, see section 2.1).

The number of nearest neighbours considered is $k = 5$ for all methods, except ANESYN and ANEIGSYN, which automatically select the value of k as explained in section 2.2.

The results have been reported in terms of two metrics, both specially designed to deal with imbalanced data (in both cases, the positive class is the minority one):

- The Geometric Mean of the sensitivities ($GM = \sqrt{S_\mathrm{p} \cdot S_\mathrm{n}}$) [19], where S_p is the sensitivity for the positive class (ratio of correctly classified patterns considering only this class) and S_n is the sensitivity for the negative one.
- The Area Under the ROC curve (AUC) [14]. Although receiver operating characteristic (ROC) graphs are useful tools for analysing the performance

of classifiers, these graphs need to be reduced to single scalar values to ease their comparison. The most common method is to compute the area under the curve (AUC), which has been shown to be equivalent to the probability that a classifier assigns a higher score to a randomly chosen positive pattern than to a negative one.

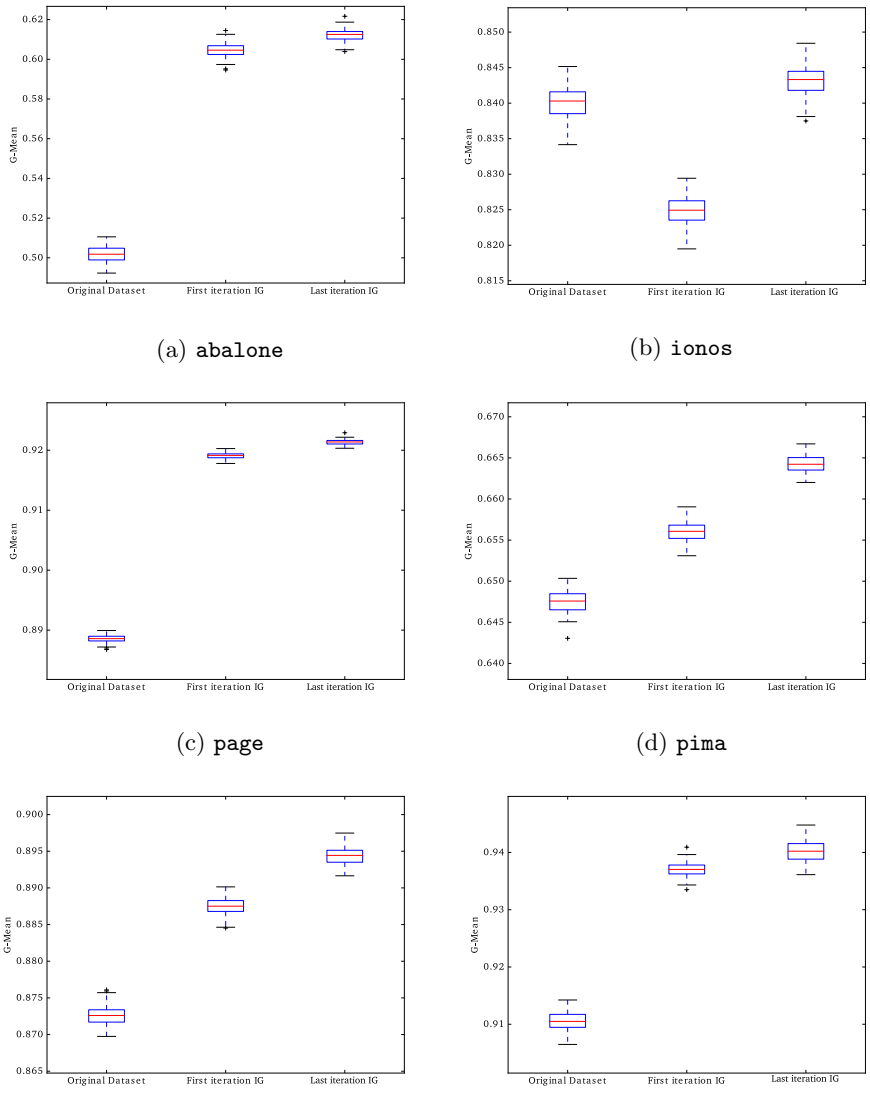
A first experiment was performed to check whether the iterative process of ANEIGSYN resulted in better suited synthetic patterns. Fig. 2 includes different boxplots comparing the test GM results obtained by generating 150 decision trees using the original dataset (Original) and complementing this dataset with the synthetic data generated by ANEIGSYN, before applying the destruction/construction iterative process ('First iteration IG' in the figure), and after the iterative process ('Last iteration IG' in the figure). As can be checked, the generation of synthetic data improves the results obtained by using the original data for all datasets, except for `ionos`, probably due to the relatively low degree of imbalance of this dataset ($IR = 1.79$). However, the IG algorithm improves the results of the trees consistently for all datasets if we compare the first and last iterations. These results confirm that the good performance obtained by the IG algorithm is not due to the lucky chance of the seed considered for the generation of the decision tree but to the quality of the synthetic patterns generated by the method.

Additionally, the results of the different algorithms can be checked in Table 2, where the average test GM and AUC performances are included. We also obtain the ranking of each method in each dataset according to test GM and AUC ($R = 1$ for the best performing method and $R = 5$ for the worst one) and the average rankings are included in Table 2. In order to check if the differences found are significant, a Wilcoxon statistical test [31] has been considered. The corresponding p-values of the tests comparing ANEIGSYN against each of the other methods are also shown in Table 2.

From the results of this Table, we can first conclude that better results are obtained by applying SMOTE than by applying ADASYN. Moreover, the two proposals of this paper (ANESYN and ANEIGSYN) generally improved the performance of ADASYN. Firstly, the improvement of the ADASYN method implemented (ANESYN) results in much better ranking values than standard ADASYN, both for GM and AUC. Indeed, the results of ANESYN are the second best ones from all the methods compared. Secondly, we can clearly conclude that the ANEIGSYN method is the best performing one in GM and AUC, although slightly better results are obtained for GM. Moreover, the statistical tests confirm these results (all differences favouring ANEIGSYN are significant, p-values < 0.05), showing that the differences are not due to random nature of the algorithms.

4 Conclusions

This paper presents two main contributions: 1) a new improved version of the ADASYN method [15], based on a better selection of the secondary pattern from



(a) `abalone` (b) `ionos`

(c) `page` (d) `pima`

(e) `vehicle` (f) `vowel`

Fig. 2: Comparison of the test GM results of 150 decision trees obtained without preprocessing the data (Original) or by including the synthetic data generated by ANEIGSYN for the first and the last iterations. These results are averaged over 100 executions of the algorithm.

Table 2: Average test performance values obtained from the 100 executions and Wilcoxon statistical test results

	Original	SMOTE	ADASYN	ANESYN	ANEIGSYN
			GM		
abalone	0.54222	*0.58094*	0.57606	0.56608	**0.60565**
ionos	0.85164	**0.85273**	0.84114	0.84343	*0.85212*
page	0.89739	0.91108	0.90436	*0.91825*	**0.92186**
pima	0.65718	0.65778	0.65964	*0.66284*	**0.66618**
vehicle	0.89159	0.89003	*0.89741*	0.89516	**0.89772**
vowel	0.93069	0.93156	0.92885	*0.93799*	**0.94233**
$\overline{R}_{GM}$	4.33	3.00	3.67	*2.83*	**1.17**
Wilcoxon p	0.028	0.028	0.046	0.028	–
			AUC		
abalone	0.63602	0.64108	*0.64777*	0.64159	**0.65864**
ionos	*0.85471*	**0.85588**	0.83731	0.84536	0.85396
page	0.90107	0.91145	0.90686	*0.91913*	**0.92230**
pima	0.66566	0.66599	0.66507	*0.66660*	**0.66880**
vehicle	0.89385	0.89228	0.89132	*0.89679*	**0.89896**
vowel	0.93328	0.93341	0.93102	*0.93939*	**0.94342**
$\overline{R}_{AUC}$	3.83	3.00	4.33	*2.50*	**1.33**
Wilcoxon p	0.046	0.046	0.028	0.028	–

The best result is in bold face and the second one in italics

which the synthetic example is generated, and 2) an iterated greedy algorithm for repeatedly destructing and reconstructing the set of synthetic patterns to improve the performance of the over-sampling process. From the results obtained by evaluating six benchmark imbalanced datasets and two baseline over-sampling methods (SMOTE [4] and ADASYN [15]), the improved version of ADASYN is seen to yield superior performance than the original one, and the iterated greedy algorithm is able to improve all the results. As future research lines, we would like to test the use of probabilistic classifiers together with continuous performance metrics for evaluating the quality of the synthetic patterns, which we think could enhance the convergence of the algorithm.

References

1. Alcalá, J., Fernández, A., Luengo, J., Derrac, J., García, S., Sánchez, L., Herrera, F.: Keel data-mining software tool: Data set repository, integration of algorithms and experimental analysis framework. Journal of Multiple-Valued Logic and Soft Computing 17(2-3), 255–287 (2010)
2. Bunkhumpornpat, C., Sinapiromsaran, K., Lursinsap, C.: Safe-Level-SMOTE: Safe-level-synthetic minority over-sampling technique for handling the class imbalanced problem. In: Proceedings of the 13th Pacific-Asia Conference on Advances in Knowledge Discovery and Data Mining. pp. 475–482. PAKDD '09, Springer-Verlag, Berlin, Heidelberg (2009)

3. Chan, P.K., Fan, W., Prodromidis, A.L., Stolfo, S.J.: Distributed data mining in credit card fraud detection. IEEE Intelligent Systems and Their Applications 14(6), 67–74 (1999)
4. Chawla, N.V., Bowyer, K.W., Hall, L.O., Kegelmeyer, W.P.: SMOTE: Synthetic minority over-sampling technique. Journal of Artificial Intelligence Research 16, 321–357 (2002)
5. Cruz, R., Fernandes, K., Cardoso, J.S., Costa, J.F.P.: Tackling class imbalance with ranking. In: Neural Networks (IJCNN), 2016 International Joint Conference on. pp. 2182–2187. IEEE (2016)
6. Domingos, P.: Metacost: A general method for making classifiers cost-sensitive. In: Proceedings of the fifth ACM SIGKDD international conference on Knowledge discovery and data mining. pp. 155–164. ACM (1999)
7. Fernández-Caballero, J.C., Martínez-Estudillo, F.J., Hervás-Martínez, C., Gutiérrez, P.A.: Sensitivity versus accuracy in multiclass problems using memetic pareto evolutionary neural networks. IEEE Transactions on Neural Networks 21(5), 750–770 (2010)
8. Galar, M., Fernández, A., Barrenechea, E., Bustince, H., Herrera, F.: A review on ensembles for the class imbalance problem: Bagging-, boosting-, and hybrid-based approaches. IEEE Transactions on Systems, Man, and Cybernetics, Part C: Applications and Reviews 42(4), 463–484 (2012)
9. García-Martínez, C., Lozano, M., Rodriguez, F.J.: Arbitrary function optimization. no free lunch and real-world problems. Soft Comput. 16(12), 2115–2133 (2012)
10. García-Martínez, C., Rodriguez, F.J., Lozano, M.: Tabu-enhanced iterated greedy algorithm: A case study in the quadratic multiple knapsack problem. European Journal of Operational Research 232, 454–463 (2014)
11. Garcia-Pedrajas, N., Pérez-Rodríguez, J., de Haro-García, A.: OligoIS: scalable instance selection for class-imbalanced data sets. IEEE Transactions on Cybernetics 43(1), 332–346 (2013)
12. Ghazikhani, A., Yazdi, H.S., Monsefi, R.: Class imbalance handling using wrapper-based random oversampling. In: 20th Iranian Conference on Electrical Engineering (ICEE2012). pp. 611–616. IEEE (2012)
13. Han, H., Wang, W., Mao, B.: Borderline-smote: a new over-sampling method in imbalanced data sets learning. In: 2005 International Conference on Intelligent Computing(ICIC05). Lecture Notes on Computer Science, vol. 3644, pp. 878–887. Springer-Verlag (2005)
14. Hanley, J.A., McNeil, B.J.: The meaning and use of the area under a receiver operating characteristic (roc) curve. Radiology 143(1), 29–36 (1982)
15. He, H., Bai, Y., Garcia, E.A., Li, S.: ADASYN: Adaptive synthetic sampling approach for imbalanced learning. In: International Joint Conference on Neural Networks (IJCNN). pp. 1322–1328 (2008)
16. He, H., Garcia, E.A.: Learning from imbalanced data. IEEE Transactions on Knowledge and Data Engineering 21(9), 1263–1284 (2009)
17. Japkowicz, N., Stephen, S.: The class imbalance problem: A systematic study. Intelligent Data Analysis 6(5), 429–449 (Oct 2002)
18. Jo, T., Japkowicz, N.: Class imbalances versus small disjuncts. ACM Sigkdd Explorations Newsletter 6(1), 40–49 (2004)
19. Kubat, M., Matwin, S.: Addressing the curse of imbalanced training sets: One-sided selection. In: Proceedings of the Fourteenth Intern. Conf. on Machine Learning. pp. 179–186. Morgan Kaufmann (1997)
20. Lichman, M.: UCI machine learning repository (2013), `http://archive.ics.uci.edu/ml`

21. Lim, P., Goh, C.K., Tan, K.C.: Evolutionary cluster-based synthetic oversampling ensemble (eco-ensemble) for imbalance learning. IEEE Transactions on Cybernetics 99, 1–12 (2016)
22. Liu, X.Y., Wu, J., Zhou, Z.H.: Exploratory undersampling for class-imbalance learning. IEEE Transactions on Systems, Man, and Cybernetics, Part B 39(2), 539–550 (2009)
23. Luengo, J., Fernández, A., García, S., Herrera, F.: Addressing data complexity for imbalanced data sets: analysis of smote-based oversampling and evolutionary undersampling. Soft Computing 15(10), 1909–1936 (2011)
24. Maciejewski, T., Stefanowski, J.: Local neighbourhood extension of smote for mining imbalanced data. In: Computational Intelligence and Data Mining (CIDM), 2011 IEEE Symposium on. pp. 104–111. IEEE (2011)
25. Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., Blondel, M., Prettenhofer, P., Weiss, R., Dubourg, V., et al.: Scikit-learn: Machine learning in python. Journal of Machine Learning Research 12(Oct), 2825–2830 (2011)
26. Pérez-Ortiz, M., Gutiérrez, P.A., Tino, P., Hervás-Martínez, C.: Oversampling the minority class in the feature space. IEEE Transactions on Neural Networks and Learning Systems 27(9), 1947–1961 (2016)
27. Ruiz, R., Stützle, T.: A simple and effective iterated greedy algorithm for the permutation flowshop scheduling problem. European Journal of Operational Research 177, 2033–2049 (2007)
28. Seiffert, C., Khoshgoftaar, T.M., Van Hulse, J., Napolitano, A.: Rusboost: A hybrid approach to alleviating class imbalance. IEEE Transactions on Systems, Man, and Cybernetics-Part A: Systems and Humans 40(1), 185–197 (2010)
29. Thai-Nghe, N., Gantner, Z., Schmidt-Thieme, L.: Cost-sensitive learning methods for imbalanced data. In: International Joint Conference on Neural Networks (IJCNN). pp. 1–8. IEEE (2010)
30. Wang, S., Minku, L.L., Yao, X.: Resampling-based ensemble methods for online class imbalance learning. IEEE Transactions on Knowledge and Data Engineering 27(5), 1356–1368 (2015)
31. Wilcoxon, F.: Individual comparisons by ranking methods. Biometrics Bulletin 1(6), 80–83 (1945)
32. Wong, G.Y., Leung, F.H., Ling, S.H.: A novel evolutionary preprocessing method based on over-sampling and under-sampling for imbalanced datasets. In: Industrial Electronics Society, IECON 2013-39th Annual Conference of the IEEE. pp. 2354–2359. IEEE (2013)