Hyperparameter Optimisation for Improving Classification under Class Imbalance

Jiawen Kong*, Wojtek Kowalczyk*, Duc Anh Nguyen*, Stefan Menzel† and Thomas Bäck*
*Leiden Institute of Advanced Computer Science (LIACS)
Leiden University, Leiden, the Netherlands
†Honda Research Institute Europe GmbH, Offenbach, Germany
Email: j.kong@liacs.leidenuniv.nl , w.j.kowalczyk@liacs.leidenuniv.nl, d.a.nguyen@liacs.leidenuniv.nl
Stefan.Menzel@honda-ri.de, T.H.W.Baeck@liacs.leidenuniv.nl

***Abstract*—Although the class-imbalance classification problem has caught a huge amount of attention, hyperparameter optimisation has not been studied in detail in this field. Both classification algorithms and resampling techniques involve some hyperparameters that can be tuned. This paper sets up several experiments and draws the conclusion that, compared to using default hyperparameters, applying hyperparameter optimisation for both classification algorithms and resampling approaches can produce the best results for classifying the imbalanced datasets. Moreover, this paper shows that data complexity, especially the overlap between classes, has a big impact on the potential improvement that can be achieved through hyperparameter optimisation. Results of our experiments also indicate that using resampling techniques cannot improve the performance for some complex datasets, which further emphasizes the importance of analyzing data complexity before dealing with imbalanced datasets.**

***Keywords*—Class Imbalance, Hyperparameter Optimisation, Overlapping Classes**

I. Introduction

The class-imbalance classification problem has caught growing attention from both the academic and the industrial field. Over years of development, many techniques have proven to be efficient in handling imbalanced datasets. These methods can be divided into data-level approaches and algorithmic-level approaches [1], [2], [3], where the data-level approaches aim to produce balanced datasets and the algorithmic-level approaches aim to adjust classical classification algorithms in order to make them appropriate for handling imbalanced datasets.

By far, the most commonly used approach for handling imbalanced data is a combination of resampling techniques and machine learning classification algorithms [4]. Both resampling techniques and machine learning algorithms involve some hyperparameters that are set to some default values and could be tuned. However, hyperparameter optimisation has not been studied yet in detail in the context of learning from imbalanced data, where both components could be tuned simultaneously.

In this paper we explore the potential of applying hyperparameter optimisation for automatic construction of high quality classifiers for imbalanced data. In our research we experiment with a small collection of imbalanced datasets and two classification algorithms: RandomForest and SVM. In each experiment we consider six scenarios for hyperparameter optimisation (see Table I). For classification algorithms, we consider two conditions, algorithms with default hyperparameters (A_d) and algorithms with optimised hyperparameters (A_o). For resampling approaches, we consider three conditions, no resampling applied (R_n), resampling applied with default hyperparameters (R_d) and resampling applied with optimised hyperparameters (R_o).

TABLE I
Six scenarios in our experiments.

Scenario	Classification Algorithms	Resampling Approaches
(1) $A_d + R_n$	Default hyperparameters	No
(2) $A_o + R_n$	Optimised hyperparameters	No
(3) $A_d + R_d$	Default hyperparameters	Default hyperparameters
(4) $A_o + R_d$	Optimised hyperparameters	Default hyperparameters
(5) $A_d + R_o$	Default hyperparameters	Optimised hyperparameters
(6) $A_o + R_o$	Optimised hyperparameters	Optimised hyperparameters

Results of our experiments demonstrate that an improvement can be obtained by applying hyperparameter tuning. In the six scenarios, optimising the hyperparameters for both classification algorithms and resampling approaches gives the best performance for all six datasets. Further study shows that the data complexity of the original data, especially the overlap between classes, influences whether a significant improvement can be achieved through hyperparameter optimisation. Compared to imbalanced datasets with high class overlap, hyperparameter optimisation works more efficiently for imbalanced datasets with low class overlap. In addition, we point out that resampling techniques are not effective for all datasets, and their effectiveness is also affected by data complexity in the original datasets. Hence, we recommend studying the data complexity of imbalanced datasets before resampling the samples and optimising the hyperparameters.

The remainder of this paper is organized as follows. Section II covers the relevant background knowledge on several resampling approaches, hyperparameter optimisation, performance metrics and data complexity measures. Section III presents the research related to our work and shows the necessity of optimising the hyperparameters and studying data complexity

for imbalanced datasets. In Section IV, the experimental setup is introduced in order to understand how the results are generated. Section V gives the results of our experiments. Section VI concludes the paper and outlines further research.

II. Background Knowledge

In this section, we review some background knowledge and start with the brief introduction of several popular resampling techniques (Section II-A) and hyperparameter optimisation (Section II-B). Then, the commonly used performance metric (Section II-C) in the field of imbalanced learning and one data complexity measure (Section II-D) are presented.

A. Resampling Techniques

In the following, the four established resampling techniques SMOTE, ADASYN, SMOTETL and SMOTEENN are introduced.

1) SMOTE: The synthetic minority over-sampling technique (SMOTE), proposed in 2002, is the most popular resampling technique [5]. SMOTE produces balanced data through creating artificial data based on the randomly chosen minority samples and their K-nearest neighbors [5]. A new synthetic sample $\boldsymbol{x_s}$ can be generated according to the following equation [6]

$$\boldsymbol{x_s} = \boldsymbol{x_i} + \delta \cdot (\hat{\boldsymbol{x}}_{\boldsymbol{i}} - \boldsymbol{x_i}), \tag{1}$$

where $\boldsymbol{x_i}$ is the minority sample to oversample, $\hat{\boldsymbol{x}}_{\boldsymbol{i}}$ is a randomly selected neighbor from its K-nearest minority class neighbors and δ is a random number, where $\delta \in [0, 1]$. Figure 1 illustrates how the synthetic samples are created in the SMOTE technique.

SMOTE provides a balanced dataset through introducing synthetic minority samples in order to prevent classification algorithms from overlooking the minority samples, therefore improving their performances.

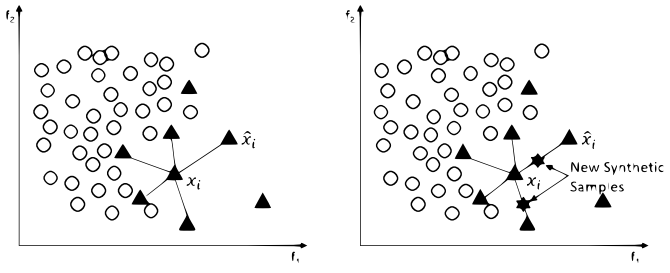

Fig. 1. An illustration of how to generate synthetic samples through SMOTE. Example of K-nearest minority class neighbors for sample $\boldsymbol{x_i}$ (K=5) (left) and new synthetic samples generated through SMOTE (right)

2) ADASYN: The adaptive synthetic (ADASYN) sampling technique is a method that aims to adaptively generate minority samples according to their distributions [7]. The samples which are harder to learn are given higher importance and will be oversampled more often [2]. The key point in ADASYN is to determine a weight ($\hat{r}_i$) for each minority sample and use $\hat{r}_i$ as the sampling importance. Weight $\hat{r}_i$ of a minority sample $\boldsymbol{x_i}$ is defined as [7]

$$\hat{r}_i = \frac{r_i}{\sum_{i=1}^{m_s} r_i}, \quad r_i = \frac{\Delta_i}{K}, \quad i = 1, ..., m_s \ , \tag{2}$$

where m_s is the number of minority samples, Δ_i is the number of neighbors of $\boldsymbol{x_i}$ that belong to majority class. For a specific minority sample, if the value of r_i is close to 1, it indicates a high level of difficulty to learn it. Then, the synthetic samples that will be generated for a minority sample can be calculated by [7]

$$g_i = \hat{r}_i \cdot G, \tag{3}$$

where G is the total number of synthetic minority samples that need to be produced. Figure 2 shows an example of the sampling importance for different minority samples.

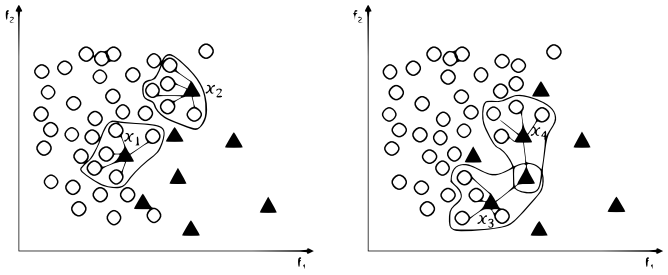

Fig. 2. Example of sampling importance for different minority samples. According to definition, $r_1 = r_2 = 1, r_3 = r_4 = 0.8$ and $\hat{r}_1 = \hat{r}_2 > \hat{r}_3 = \hat{r}_4$, indicating the sampling importance of sample $\boldsymbol{x_1}, \boldsymbol{x_2}$ is higher than $\boldsymbol{x_3}, \boldsymbol{x_4}$ and more synthetic samples will be produced for $\boldsymbol{x_1}$ and $\boldsymbol{x_2}$.

Compared to SMOTE, the only difference in ADASYN oversampling procedure is that more synthetic samples will be generated for harder minority samples. In this way, the ADASYN not only provides less learning bias but puts more focus on the difficulty to learn minority samples.

3) SMOTETL: In a binary classification problem, a Tomek link is defined as a pair of samples from different classes which are the nearest neighbors for each other [8]. In the SMOTETL technique, the first step is to oversample the minority classes using SMOTE and then the Tomek links for the oversampled samples are removed [9]. In other words, the SMOTETL technique provides a more clear decision boundary by removing part of the samples in the overlapping region.

4) SMOTEENN: Similar to SMOTETL, the first step of SMOTEENN is also to oversample the minority class with SMOTE. After that, the Wilson's Edited Nearest Neighbors (ENN) are used to remove the sample who has a different class from at least two of its three nearest neighbors [10]. By removing the noisy samples, SMOTEENN makes the classification algorithm work more efficiently.

B. Hyperparameter Optimisation

The most basic methods used by beginners in the field of imbalanced learning are combining the resampling techniques and machine learning classification algorithms. Compared with

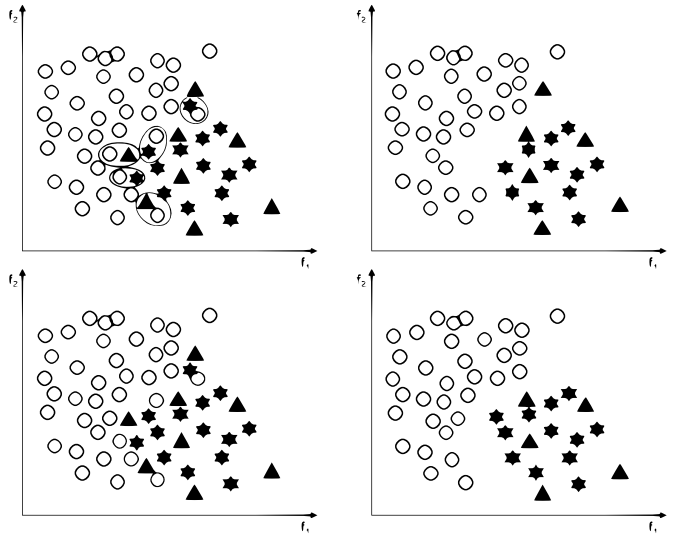

Fig. 3. Example of clearing Tomek links for oversampled samples (upper) and example of removing the noisy samples using ENN (lower). Compared with SMOTETL, SMOTEENN produces a clearer decision boundary.

randomly selecting the hyperparameters in a learning algorithm, choosing a set of optimal hyperparameters can improve the performance of the algorithm.

In this paper, RandomForest and SVM are considered to do the classification and both algorithms involve various hyperparameters, which affect the performance (e.g., prediction accuracy) significantly. For instance, in RandomForest, the choice of the depth of a decision tree and the number of trees in a forest will have an influence on the performance. To determine the best set of hyperparameters for a given problem/dataset naturally leads to the well-established hyperparameter optimisation task. The hyperparameter optimisation problem can be represented by [11]

$$x^* = \arg\min_{x \in \chi} f(x), \tag{4}$$

where x can be any combination of hyperparameters in domain χ and x^* is the set of hyperparameters that achieve the lowest value of objective function $f(x)$. Typically, it is expensive to evaluate $f(x)$ directly.

Bayesian hyperparameter optimisation approaches provide a less expensive way to optimise the hyperparameters. Its strategy keeps tracking previous evaluated results and use the obtained information to form a surrogate probabilistic model of the objective function $M \leftarrow P(y|x)$, where x indicates candidate hyperparameters and y indicates the probability of the corresponding score on the objective function [11], [12]. Compared to the original objective function, this surrogate one is less expensive to optimise, because it chooses the next candidate hyperparameters worth evaluating instead of wasting time on unworthy hyperparameters.

In practical, there are many software packages based on Bayesian hyperparameter optimisation, e.g. Spearmint, SMAC, HyperOpt, SPOT, etc. In this paper, a python library[1], HyperOpt [13], is used to perform the hyperparameter optimisation for classification algorithms.

C. Performance Metrics for Imbalanced Learning

Accuracy is the most commonly used measure for classification problems. In a binary classification problem, the confusion matrix (see Table II) can provide intuitive classification results.

TABLE II
CONFUSION MATRIX FOR A BINARY CLASSIFICATION PROBLEM

	Positive prediction	**Negative prediction**
Positive class	True Positives (TP)	False Negatives (FN)
Negative class	False Positives (FP)	True Negatives (TN)

According to the confusion matrix (see Table II), accuracy (Acc) can be calculated as follows.

$$Acc = \frac{TP + TN}{TP + FN + FP + TN} \tag{5}$$

However, accuracy may give a deceptive evaluation in imbalanced domains. For example, in a binary class-imbalance classification problem, the majority-class and minority-class samples take 95% and 5% of the total samples respectively. Even if the classifier predicts all the samples as majority class, the accuracy is still 95%, which makes the classifier seems extremely efficient but actually it neglects the minority class. That is to say, the accuracy does not reflect the actual effectiveness of an algorithm in imbalanced domains. In imbalance learning domain, the Area Under the ROC Curve (AUC) can be used to evaluate the performance [14], [15] and can be computed by

$$\begin{aligned} AUC &= \frac{1 + TP_{rate} - FP_{rate}}{2}, \\ TP_{rate} &= \frac{TP}{TP + FN}, \\ FP_{rate} &= \frac{FP}{FP + TN}. \end{aligned} \tag{6}$$

where TP_{rate} is the true positives rate, FP_{rate} is the false positives rate.

Apart from the AUC value, there are also some other measures to assess the performance for imbalanced datasets, such as geometric mean (GM) [16] and F-measure (FM) [15]. These measures are given by

$$\begin{aligned} GM &= \sqrt{\frac{TP}{TP + FN} \times \frac{TN}{FP + TN}}, \\ FM &= \frac{(1 + \beta)^2 \times Recall \times Precision}{\beta^2 \times Recall + Precision}, \\ Recall &= \frac{TP}{TP + FN}, \\ Precision &= \frac{TP}{TP + FP}, \end{aligned} \tag{7}$$

where β is a coefficient and normally set to 1.

[1] available at: http://hyperopt.github.io/hyperopt/

D. Data Complexity Measures

For the data complexity measures in binary classification problems, the measures can be divided into *feature overlapping measures*, *measures of the separability of classes* and *geometry, topology and density of manifolds measures* [10], [17]. In this paper, only one of the feature overlapping measures, *maximum Fisher's discriminant ratio* is considered.

The *maximum Fisher's discriminant*, denoted by F1, measures the overlap between the feature values of different classes and is given by [10]:

$$F1 = \max_{i=1}^{m} r_{f_i}, \tag{8}$$

where m is the number of features, r_{f_i} is the discriminant ratio for each feature f_i and can be calculated through the following formula (for a binary classification problem) [18]:

$$r_{f_i} = \frac{\sum_{c=1}^{2} n_c(\mu_c^{f_i} - \mu^{f_i})^2}{\sum_{c=1}^{2}\sum_{j=1}^{n_c}(x_j^c - \mu_c^{f_i})^2}, \tag{9}$$

where n_c is the number of examples in class c, $\mu_c^{f_i}$ is the mean value of feature f_i across class c, μ^{f_i} is the mean value of feature f_i across all classes, and x_j^c represents the value of feature f_i for a sample from class c [18]. F1 measures the highest discriminant ratio among all the features in the dataset and higher discriminant ratio indicates lower complexity [10].

III. Related Works

As mentioned in the Introduction, the combination of resampling techniques and machine learning classification algorithms is the most commonly used approach for handling imbalanced datasets. Research works also focused on these two separate parts, developing new resampling techniques and adjusting machine learning algorithms to be more appropriate for imbalanced datasets. Both resampling techniques and classification algorithms involve some hyperparameters, which might influence the performance significantly. However, no detailed hyperparameter optimisation research has been done in the context of learning from imbalanced data. Previous research has considered the hyperparameters for the classifiers for class-imbalance problems [19], but the hyperparameters in resampling techniques are not included. Apart from developing new techniques to deal with imbalanced datasets, the data complexity in the dataset itself has caught an increasing attention in recent studies of class-imbalance problems. It has been shown that the degradation of machine learning algorithms for imbalanced datasets is not directly caused by class imbalance, but is also related to the degree of class overlapping [20], and the classification algorithms are more sensitive to noise than to class imbalance [15]. It is also concluded that data complexity may influence the choice of resampling methods [2]. Hence, in this paper, we consider the hyperparameter optimisation for both resampling techniques and classification algorithms. Furthermore, the relation between the degree of class overlap and the added value of hyperparameter tuning is investigated.

IV. Experimental Setup

In this section, we first introduce the datasets used our experiment (Section IV-A) and then highlight the importance of cross-validation design (Section IV-B). After that, the procedure to execute the experiment is given (Section IV-C).

A. Datasets

The experiments reported in this paper are based on six imbalanced datasets from the KEEL-collection [21]. Detailed information on the datasets are shown in Table III. The "glass1" *VS* "glass6" and "yeast3" *VS* "yeast4" can be regarded as two comparison groups. IR indicates the imbalance ratio, which is the ratio of the number of majority class samples to the number of minority class samples. The overlap between classes is calculated by Maximum Fisher's Discriminant Ratio (*F1*). Lower *F1* value indicates higher overlap between classes [2].

TABLE III
Information on the datasets.

Dataset	#Attributes	#Examples	#Classes	IR	F1
glass1	9	214	2	1.82	0.92
glass6	9	214	2	6.38	0.53
yeast3	8	1484	2	8.1	0.70
yeast4	8	1484	2	28.1	0.91
ecoli3	7	336	2	8.6	0.84
abalone19	8	4174	2	129.44	0.96

B. Cross-Validation Design

Cross-validation (CV) is an effective technique to assess the classification performance. Recent research has concluded that a poorly designed CV procedure for imbalanced datasets will result in an overoptimism problem [22], [2]. The overoptimism occurs when CV is performed after oversampling. Suppose we first obtain a balanced dataset through oversampling approaches, then perform cross-validation. In this way, since the synthetic samples share similar patterns with the original sample, samples with similar patterns may appear in both training and test set, which will lead to the overoptimism problem [2]. In order to avoid this problem in our experiment, 5-fold stratified CV is first implemented on the dataset and only the training set is oversampled.

C. Design of the Experiments

As mentioned in Section I, we experiment with six imbalanced datasets, two algorithms and four resampling techniques. Thus, in our experiment, we have $6 \cdot 2 \cdot 5 = 60$ settings tested on each data set, with 6 scenarios, 2 classifiers, and 5 resampling approaches (including none).

The hyperparameter optimisation for classification algorithm is done through HyperOpt. Hyperparameters in resampling approaches includes the number of neighbors, imbalance ratio after resampling and etc. In our experiment, hyperparameter optimisation for resampling approaches is done through grid search. Whenever we optimise hyperparameters with "HyperOpt", the AUC loss (1-AUC) is set as the objective

TABLE IV

EXPERIMENTAL RESULTS (AUC) FOR TWO CLASSIFICATION ALGORITHMS REGARDING SIX SCENARIOS.
THE GREY SHADE AND NO SHADE INDICATE THE EXPERIMENTAL RESULTS FOR SVM AND RANDOMFOREST RESPECTIVELY.
P-VALUES INDICATE THE STATISTICAL EVIDENCE OF T-TESTS BETWEEN EXPERIMENTAL RESULTS OF SCENARIO ($A_o + R_o$) AND ($A_d + R_d$).
DATASET WITH * INDICATES THE RESULTS OF SCENARIO ($A_o + R_o$) IS SIGNIFICANTLY HIGHER THAN RESULTS OF SCENARIO ($A_d + R_d$).

		Resampling Approaches (SVM *vs.* RandomForest)									
Scenarios	Dataset	NONE		SMOTE		ADASYN		SMOTETL		SMOTEENN	
$A_d + R_n$		0.6753	0.8301	—	—	—	—	—	—	—	—
$A_o + R_n$		0.8309	0.8345	—	—	—	—	—	—	—	—
$A_d + R_d$		—	—	0.7165	0.8401	0.7253	0.8456	0.7416	0.8420	0.7484	0.8126
$A_o + R_d$	glass1*	—	—	0.8360	0.8537	0.8390	0.8527	0.8423	0.8479	0.8435	0.8278
$A_d + R_o$		—	—	0.7322	0.8599	0.7370	0.8498	0.7437	0.8463	0.7518	0.8216
$A_o + R_o$		—	—	0.8508	0.8649	0.8592	0.8631	0.8659	0.8527	0.8673	0.8379
p-value		—	—	$\ll 0.05$	0.0060	$\ll 0.05$	0.0133	$\ll 0.05$	0.0022	$\ll 0.05$	0.0100
$A_d + R_n$		0.9768	0.9884	—	—	—	—	—	—	—	—
$A_o + R_n$		0.9848	0.9892	—	—	—	—	—	—	—	—
$A_d + R_d$		—	—	0.9749	0.9862	0.9727	0.9849	0.9768	0.9880	0.9761	0.9870
$A_o + R_d$	glass6	—	—	0.9807	0.9893	0.9787	0.9877	0.9832	0.9886	0.9840	0.9883
$A_d + R_o$		—	—	0.9796	0.9888	0.9744	0.9870	0.9805	0.9896	0.9795	0.9905
$A_o + R_o$		—	—	0.9850	0.9897	0.9833	0.9883	0.9861	0.9917	0.9857	0.9910
p-value		—	—	0.0693	0.1633	0.1819	0.1166	0.3067	0.1513	0.0603	0.1279
$A_d + R_n$		0.9688	0.9624	—	—	—	—	—	—	—	—
$A_o + R_n$		0.9712	0.9700	—	—	—	—	—	—	—	—
$A_d + R_d$		—	—	0.9642	0.9662	0.9601	0.9670	0.9659	0.9653	0.9649	0.9693
$A_o + R_d$	yeast3	—	—	0.9663	0.9731	0.9655	0.9727	0.9701	0.9669	0.9689	0.9743
$A_d + R_o$		—	—	0.9671	0.9693	0.9628	0.9696	0.9684	0.9705	0.9668	0.9722
$A_o + R_o$		—	—	0.9704	0.9759	0.9683	0.9756	0.9733	0.9742	0.9716	0.9787
p-value		—	—	0.3890	0.1529	0.1256	0.0567	0.6166	0.0585	0.2084	0.0573
$A_d + R_n$		0.8479	0.9211	—	—	—	—	—	—	—	—
$A_o + R_n$		0.8739	0.9389	—	—	—	—	—	—	—	—
$A_d + R_d$		—	—	0.9025	0.9165	0.8998	0.9123	0.9019	0.9257	0.9079	0.9237
$A_o + R_d$	yeast4*	—	—	0.9132	0.9300	0.9076	0.9293	0.9089	0.9312	0.9093	0.9327
$A_d + R_o$		—	—	0.9098	0.9345	0.9059	0.9319	0.9102	0.9327	0.9122	0.9291
$A_o + R_o$		—	—	0.9178	0.9393	0.9105	0.9346	0.9147	0.9389	0.9201	0.9364
p-value		—	—	$\ll 0.05$	0.0075	0.0133	0.0013	0.0061	0.0036	0.0385	0.0355
$A_d + R_n$		0.9540	0.9359	—	—	—	—	—	—	—	—
$A_o + R_n$		0.9551	0.9535	—	—	—	—	—	—	—	—
$A_d + R_d$		—	—	0.9528	0.9310	0.9505	0.9303	0.9508	0.9300	0.9514	0.9329
$A_o + R_d$	ecoli3	—	—	0.9559	0.9338	0.9519	0.9395	0.9549	0.9384	0.9529	0.9385
$A_d + R_o$		—	—	0.9562	0.9419	0.9528	0.9396	0.9569	0.9417	0.9571	0.9416
$A_o + R_o$		—	—	0.9581	0.9432	0.9543	0.9407	0.9573	0.9444	0.9598	0.9450
p-value		—	—	0.4507	0.1337	0.3408	0.1532	0.4436	0.0773	0.3596	0.0575
$A_d + R_n$		0.7373	0.7239	—	—	—	—	—	—	—	—
$A_o + R_n$		0.7687	0.8077	—	—	—	—	—	—	—	—
$A_d + R_d$		—	—	0.8051	0.7934	0.8053	0.7971	0.8051	0.7946	0.8060	0.8034
$A_o + R_d$	abalone19*	—	—	0.8478	0.8328	0.8484	0.8347	0.8473	0.8331	0.8496	0.8395
$A_d + R_o$		—	—	0.8088	0.8095	0.8097	0.8023	0.8089	0.8077	0.8108	0.8090
$A_o + R_o$		—	—	0.8494	0.8389	0.8503	0.8402	0.8488	0.8391	0.8511	0.8414
p-value		—	—	$\ll 0.05$	$\ll 0.05$	$\ll 0.05$	$\ll 0.05$	$\ll 0.05$	$\ll 0.05$	$\ll 0.05$	$\ll 0.05$

function to minimise and the number of iterations is set to 500. For each experiment, we repeated 30 times with different random seeds. After that, the paired t-tests were performed on each 30 AUC values to test if there is significant difference between the results of each scenario on a 5% significance level.

V. RESULTS AND DISCUSSION

The experimental results are presented in Table IV to investigate the importance of hyperparameter optimisation for imbalanced datasets. For all the six datasets in our experiment, we observe that optimising the hyperparameters for both classifiers and resampling approaches gives the best performance. The statistical hypothesis tests mentioned in Section IV-C are performed on the AUC values of scenario ($A_d + R_d$) and ($A_o + R_o$). The test results indicate that there is enough statistical evidence showing the performance improvements are significant for datasets "glass1", "yeast4" and "abalone19". In other words, applying the hyperparameter optimisation does not bring significant improvement for datasets "glass6", "yeast3" and "ecoli3". Our experimental results demonstrate that significant improvement can be achieved by performing hyperparameter optimisation for datasets with high F_1 values. That is to say, hyperparameter optimisation works efficiently

for datasets with low overlap between classes.

Furthermore, comparing the AUC values of scenario ($A_d + R_n$) and ($A_d + R_d$), for datasets "glass6", "yeast3" and "ecoli3, resampling techniques does not improve the classification performance. Thus, we can conclude that oversampling techniques are not effective for datasets with high overlap. The generated synthetic samples might bring additional noise and make the class overlap even higher. Another point worth mentioning is that, compared to datasets with high overlap, we expected the classification algorithms would perform better on datasets with low overlap. However, the experimental results are contrary to our presupposition. This is because the complexity of a classification problem is not only determined by the overlap between classes but also related to other types of complexity, such as linearity measures.

In the end, we can also observe that there is no specific combination of classifiers and resampling techniques that can provide the best performance for all datasets. For a given dataset, the best combination of classifiers and resampling approaches might depend on the data complexity itself.

VI. Conclusions and Future Work

In this work we considered six scenarios of hyperparameter optimisation for classification algorithms and resampling approaches. Two main conclusions can be derived according to our experimental results:

1). In our experiment, the results of scenario ($A_o + R_o$) outperform the other five scenarios. Especially for imbalanced datasets with low class overlap, applying hyperparameter optimisation for both classification algorithms and resampling approaches can significantly improve the performance. Nevertheless, the time consumption caused by hyperparameter optimisation is not negligible. For example, according to our experimental design, for "glass1" dataset, the time cost of one experiment in scenario ($A_d + R_d$) and ($A_o + R_o$) are respectively 0.0625s and 239.3476s. Therefore, we recommend studying the data complexity and considering the trade-off between time cost and potential improvement before optimising the hyperparameters.

2). Based on our experimental results, we find oversampling techniques does not give performance improvement for imbalanced datasets with high class overlap. This further emphasizes the importance of learning the data complexity before dealing with the imbalanced datasets.

In future work, more data complexity measures will be considered in order to study the relation between hyperparameter optimisation and data complexity in detail. Additionally, more attention should be put on developing techniques which can efficiently handle complex imbalanced datasets. Finally, we observe the best choice of classifiers and oversampling techniques depends on the dataset itself. Therefore, another study worth exploring would be to produce a semi-automatic approach which can help choosing the best combination of resampling approaches, machine learning algorithms and hyperparameter optimisation strategies.

Acknowledgment

This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement number 766186.

References

[1] V. Ganganwar, "An overview of classification algorithms for imbalanced datasets," *International Journal of Emerging Technology and Advanced Engineering*, vol. 2, no. 4, pp. 42–47, 2012.

[2] M. S. Santos, J. P. Soares, P. H. Abreu, H. Araujo, and J. Santos, "Cross-validation for imbalanced datasets: Avoiding overoptimistic and overfitting approaches [research frontier]," *IEEE Computational Intelligence Magazine*, vol. 13, no. 4, pp. 59–76, 2018.

[3] U. Bhowan, M. Johnston, M. Zhang, and X. Yao, "Evolving diverse ensembles using genetic programming for classification with unbalanced data," *IEEE Transactions on Evolutionary Computation*, vol. 17, no. 3, pp. 368–386, 2012.

[4] V. López, A. Fernández, J. G. Moreno-Torres, and F. Herrera, "Analysis of preprocessing vs. cost-sensitive learning for imbalanced classification. open problems on intrinsic data characteristics," *Expert Systems with Applications*, vol. 39, no. 7, pp. 6585–6608, 2012.

[5] N. V. Chawla, K. W. Bowyer, L. O. Hall, and W. P. Kegelmeyer, "Smote: synthetic minority over-sampling technique," *Journal of artificial intelligence research*, vol. 16, pp. 321–357, 2002.

[6] H. He and E. A. Garcia, "Learning from imbalanced data," *IEEE Transactions on Knowledge & Data Engineering*, no. 9, pp. 1263–1284, 2008.

[7] H. He, Y. Bai, E. A. Garcia, and S. Li, "Adasyn: Adaptive synthetic sampling approach for imbalanced learning," in *2008 IEEE International Joint Conference on Neural Networks (IEEE World Congress on Computational Intelligence)*. IEEE, 2008, pp. 1322–1328.

[8] I. Tomek, "Two modifications of cnn," *IEEE Trans. Systems, Man and Cybernetics*, vol. 6, pp. 769–772, 1976.

[9] G. E. Batista, R. C. Prati, and M. C. Monard, "A study of the behavior of several methods for balancing machine learning training data," *ACM SIGKDD explorations newsletter*, vol. 6, no. 1, pp. 20–29, 2004.

[10] A. C. Lorena, L. P. Garcia, J. Lehmann, M. C. Souto, and T. K. Ho, "How complex is your classification problem? a survey on measuring classification complexity," *arXiv preprint arXiv:1808.03591*, 2018.

[11] J. S. Bergstra, R. Bardenet, Y. Bengio, and B. Kégl, "Algorithms for hyper-parameter optimization," in *Advances in Neural Information Processing Systems 24*, J. Shawe-Taylor, R. S. Zemel, P. L. Bartlett, F. Pereira, and K. Q. Weinberger, Eds. Curran Associates, Inc., 2011, pp. 2546–2554. [Online]. Available: http://papers.nips.cc/paper/4443-algorithms-for-hyper-parameter-optimization.pdf

[12] J. Bergstra, D. Yamins, and D. Cox, "Making a science of model search: Hyperparameter optimization in hundreds of dimensions for vision architectures," in *Proceedings of the 30th International Conference on Machine Learning*, ser. Proceedings of Machine Learning Research, S. Dasgupta and D. McAllester, Eds., vol. 28, no. 1. Atlanta, Georgia, USA: PMLR, 17–19 Jun 2013, pp. 115–123. [Online]. Available: http://proceedings.mlr.press/v28/bergstra13.html

[13] J. Bergstra, B. Komer, C. Eliasmith, D. Yamins, and D. Cox, "Hyperopt: A python library for model selection and hyperparameter optimization," *Computational Science & Discovery*, vol. 8, p. 014008, 07 2015.

[14] J. Huang and C. X. Ling, "Using auc and accuracy in evaluating learning algorithms," *IEEE Transactions on knowledge and Data Engineering*, vol. 17, no. 3, pp. 299–310, 2005.

[15] V. López, A. Fernández, S. García, V. Palade, and F. Herrera, "An insight into classification with imbalanced data: Empirical results and current trends on using data intrinsic characteristics," *Information sciences*, vol. 250, pp. 113–141, 2013.

[16] R. Barandela, J. S. Sánchez, V. Garca, and E. Rangel, "Strategies for learning in class imbalance problems," *Pattern Recognition*, vol. 36, no. 3, pp. 849–851, 2003.

[17] T. K. Ho and M. Basu, "Complexity measures of supervised classification problems," *IEEE Transactions on Pattern Analysis & Machine Intelligence*, no. 3, pp. 289–300, 2002.

[18] R. A. Mollineda, J. S. Sánchez, and J. M. Sotoca, "Data characterization for effective prototype selection," in *Iberian Conference on Pattern Recognition and Image Analysis*. Springer, 2005, pp. 27–34.

[19] N. Thai-Nghe, A. Busche, and L. Schmidt-Thieme, "Improving academic performance prediction by dealing with class imbalance," in *2009 Ninth International Conference on Intelligent Systems Design and Applications*. IEEE, 2009, pp. 878–883.

[20] R. C. Prati, G. E. Batista, and M. C. Monard, "Class imbalances versus class overlapping: an analysis of a learning system behavior," in *Mexican international conference on artificial intelligence*. Springer, 2004, pp. 312–321.

[21] J. Alcalá-Fdez, L. Sánchez, S. Garcia, M. J. del Jesus, S. Ventura, J. M. Garrell, J. Otero, C. Romero, J. Bacardit, V. M. Rivas *et al.*, "Keel: a software tool to assess evolutionary algorithms for data mining problems," *Soft Computing*, vol. 13, no. 3, pp. 307–318, 2009.

[22] L. Lusa *et al.*, "Joint use of over-and under-sampling techniques and cross-validation for the development and assessment of prediction models," *BMC bioinformatics*, vol. 16, no. 1, p. 363, 2015.