



GMMSampling: a new model-based, data difficulty-driven resampling method for multi-class imbalanced data

Iwo Naglik¹ · Mateusz Lango¹

Received: 7 June 2022 / Revised: 22 June 2023 / Accepted: 3 October 2023 /
Published online: 20 November 2023
© The Author(s) 2023

Abstract

Learning from multi-class imbalanced data has still received limited research attention. Most of the proposed methods focus on the global class imbalance ratio only. In contrast, experimental studies demonstrated that the imbalance ratio itself is not the main difficulty in the imbalanced learning. It is the combination of the imbalance ratio with other data difficulty factors, such as class overlapping or minority class decomposition into various subconcepts, that significantly affects the classification performance. This paper presents GMMSampling—a new resampling method that exploits information about data difficulty factors to clear class overlapping regions from majority class instances and to simultaneously oversample each subconcept of the minority class. The experimental evaluation demonstrated that the proposed method achieves better results in terms of G-mean, balanced accuracy, macro-AP, MCC and F-score than other related methods.

Keywords Imbalanced data · Multi-class classification · Resampling methods · Data difficulty factors · Gaussian mixture model

1 Introduction

Learning from class-imbalanced data is an important research topic due to its application in many real-world application domains such as cybersecurity (Sun et al., 2018), sentiment analysis (Lango, 2019) or medicine (Wang et al., 2020). It concerns constructing classifiers from datasets that include at least one underrepresented class, which is typically of key importance from the application point of view (e.g. detecting a rare disease or a cyberattack). Unfortunately, standard classification algorithms learn weak representations of such

Editor: Nuno Moniz, Paula Branco, Luís Torgo, Nathalie Japkowicz, Michal Wozniak, Shuo Wang.

✉ Mateusz Lango
mateusz.lango@cs.put.poznan.pl

Iwo Naglik
inaglik.put.poznan@gmail.com

¹ Institute of Computer Science, Poznan University of Technology, Piotrowo 2, Poznań 61-138, Poland

minority classes, resulting in low predictive performance on their examples and limiting their usefulness in practice.

Due to the big practical importance of the problem, many different learning methods for class-imbalanced data have been proposed in the literature (Branco et al., 2016). They can be roughly divided into three categories: algorithmic-level, data-level and cost-sensitive methods (He & Garcia, 2009). Among them, data-level methods (otherwise known as resampling methods) stand out as effective and universal approaches that can be used with virtually any classification algorithm (Janicka et al., 2019). The resampling methods preprocess the dataset in order to construct a data distribution that is more balanced and therefore more suitable for learning an effective classifier in the imbalanced domain. These methods include undersampling approaches that remove instances of majority classes from the dataset, as well as oversampling approaches that append additional minority instances to the dataset. Such additional instances are usually constructed by duplicating, interpolating, or modifying already existing minority examples. Despite significant research interest and a large number of proposed methods, some issues remain open, leaving room for further improvements.

First, the vast majority of the proposed methods are designed specifically for binary imbalanced classification problems, whereas in practice multi-class imbalanced problems also often emerge. One possible workaround is to decompose a multi-class problem into a series of binary ones and later apply standard methods for binary imbalanced data (Fernández et al., 2013). Nevertheless, such approaches do not take into account the interrelations between classes and ignore the global structure of the multi-class problem, which can result in weaker classification performance.

Second, the recent No Free Lunch theorem for resampling (Moniz & Monteiro, 2021) demonstrated that without any further assumptions about the data distribution, the performances of resampling methods are indistinguishable. Unfortunately, the area of competence of particular resampling methods is often poorly defined, and there are little recommendations for what kind of data a particular method is suitable. There are also some other generally recognized limitations of this type of approaches. For instance, undersampling methods may discard important majority examples from the dataset, causing too weak recognition of majority classes. This issue can be even more critical in the multi-class setting, since to obtain a balanced distribution, one typically needs to undersample all classes to the size of the smallest one. This may result in the undersampling of some minority classes and thus removing scarce minority instances from the dataset. On the other hand, oversampling approaches can make noisy examples more prevalent or overgeneralize the representation of minority classes. Note that the vast majority of the proposed oversampling approaches modify existing instances (e.g. by interpolating them, rotating them around the class centre, or by adding to them some particular noise), but do not generate entirely new examples that follow the class distribution.

Furthermore, almost all data-level approaches for multi-class imbalanced data are primarily focused on dealing with the global imbalance ratio i.e., with the disproportion between class cardinalities. Nevertheless, a recent experimental study on the difficulty of multi-class imbalanced problems (Lango & Stefanowski, 2022) confirmed earlier observations on binary problems (Japkowicz, 2003; Prati et al., 2004; Napierala & Stefanowski, 2012) that the imbalance ratio itself is not the main source of difficulty in imbalanced classification. In particular, for some relatively simple classification problems, increasing the imbalance ratio does not lead to the degeneration of minority class recognition. Only if the global class imbalance occurs together with other so-called *data difficulty factors*, it causes significant deterioration of classification performance. Such data difficulty factors include class overlapping,

occurrence of outlier/rare instances, and the decomposition of minority class into several sub-concepts (Stefanowski, 2013). These subconcepts usually have unequal cardinalities, which leads to yet another data difficulty factor: within-class imbalance, which occurs when some of the minority class subconcepts are even more severely underrepresented in the dataset than the others (He & Garcia, 2009).

According to the aforementioned experimental analysis (Lango & Stefanowski, 2022), these other data difficulty factors seem crucial to handle the issue of class imbalance efficiently, however, they are not considered by most resampling methods already proposed in the literature for multi-class data. One exception is Similarity Oversampling and Undersampling Preprocessing (SOUP) (Janicka et al., 2019) that calculates examples' safe levels to detect data difficulty factors occurring in the dataset and adjust its operation to deal with data regions of different characteristics. Even though experiments demonstrate that the proposed example's safe levels can sufficiently well model class overlapping, detect outliers, and take into account different configurations of class sizes, they do not enable detection of the decomposition of minority class into several subconcepts. Therefore, the methods like SOUP are not able to properly deal with this difficulty factor and to handle the within-class imbalance that possibly occurs for some minority classes.

Based on the critical analysis of the literature, in this paper we propose GMMSampling—a new resampling method for multi-class imbalanced data that:

- Oversamples minority class instances by generating completely new instances that are not mere modifications of existing data points. The new instances are sampled from a class distribution approximated by a universal density approximator (Goodfellow et al., 2016): Gaussian Mixture Model (GMM),
- Uses detected data difficulty factors with examples' safe levels in order to focus the sampling procedure on the most unsafe minority classes regions in the feature space, taking a principled approach towards data difficulty factors like class overlapping or outliers,
- Detects minority class subconcepts by automatically tuning the GMM model and ensuring that each subconcept is sufficiently well represented in the pre-processed training data, thus mitigating the problem of within-class imbalance,
- To obtain a fully balanced distribution, combines oversampling with undersampling that removes the most harmful majority instances according to their safe levels.

The conducted experimental evaluation demonstrated that several standard classification algorithms obtain superior predictive performance in terms of balanced accuracy, MCC, macro-AP, G-mean and F1-score measures while trained on the dataset preprocessed by the proposed approach. The presented ablation study indicated that the main reasons behind obtained high classification performance is better handling of within-class imbalance and usage of the newly proposed example generation schema. Moreover, since the method relies on a well-known probabilistic model of data i.e. GMM, its suitability for a given dataset can be rather easily determined using standard statistical data analysis methods (Wichitchan et al., 2020).

2 Related works

In a given dataset, class imbalance is often measured with the imbalanced ratio (IR) which is defined as a ratio of the biggest class size to the cardinality of the smallest class. Although any dataset with $IR > 1$ can be considered imbalanced, when talking about the

issue of class imbalance we assume that the disproportion between class cardinalities is significant (He & Garcia, 2009).

Since data of such type often occur in practice, a considerable number of methods alleviating the issue of class imbalance has been proposed. Presenting an overview of imbalanced learning is out of scope of this paper. Instead, we will limit this section to discuss the research on data difficulty factors in the multi-class setting and to briefly overview most related methods for multi-class imbalanced data. More detailed reviews on this topic can be found, e.g. in Branco et al. (2016); Fernández et al. (2018).

2.1 Data difficulty factors in multi-class imbalanced problems

Despite the fact that multi-class classification problems are commonly recognized as more difficult than their binary counterparts (Krawczyk, 2016; Zhao et al., 2008; Wang & Yao, 2012), most of the previous research focused on binary imbalanced problems only. Similarly, the sources of difficulty in inducing classifiers from imbalanced data were mostly studied in the binary setting. The studies (Stefanowski, 2013; Japkowicz & Stephen, 2002; Japkowicz, 2003) revealed that other data difficulty factors (e.g. class overlapping, data rarity, partitioning of a class into non-homogeneous subconcepts) occurring together with significant imbalance ratio are much more harmful for the predictive performance than the imbalance ratio itself. Therefore, understanding and properly dealing with these data difficulty factors is considered crucial for the development of better algorithms for imbalanced data (Japkowicz & Stephen, 2002; Japkowicz, 2000; Weiss, 2004; Prati et al., 2004; Santos et al., 2023).

In the context of multi-class imbalanced data, several additional data difficulty factors that do not occur in the binary setting were identified. Wang and Yao (2012) studied the impact of increasing the number of classes on the classification performance, as well as the differences between problems containing many minority classes and one majority class (multi-minority) as well as problems with many majority classes (multi-majority). A more comprehensive recent experimental study (Lango & Stefanowski, 2022) besides multi-majority and multi-minority problems also investigated the problems with gradually growing sizes, i.e. problems that also include classes of intermediate sizes. These studies highlighted that the class size configuration impacts the classification performance and revealed that multi-majority problems are more difficult than multi-minority ones, whereas the difficulty of gradual problems lies somewhere in between.

As in the binary case, the analysis of multi-class problems (Lango & Stefanowski, 2022) found class overlapping very influential for the classification performance of minority classes. However, it also demonstrated that overlapping between different class types poses different levels of difficulty. For instance, the overlap of two minority classes is less harmful than overlapping between a majority and a minority class. Interestingly, the overlapping of an intermediate class with a minority one leads to a fast degradation of minority class recognition without having a major impact on the correct classification of the intermediate class examples (as if the intermediate class was majority one), whereas the overlap of an intermediate class with the majority class makes the performance of intermediate class recognition decrease significantly (as if the intermediate class was a minority one).

In order to design algorithms that take into account the data difficulty factors, methods for detecting them in real datasets are needed. The authors of Lango et al. (2017) proposed calculating *example safe level* for each instance and then average them to assess the

difficulty of a dataset. Example safe level is calculated by finding k -nearest neighbors of an example and average the so-called degrees of similarity between the neighbor's and the example's class. More concretely, the safe level of an example x is defined as:

$$\text{safe_level}(x) = \frac{1}{k} \sum_{c \in C} n_c \mu_{C_x, c} \quad (1)$$

where C is the set of all classes, C_x is the class of example x , n_c is the number of neighbors belonging to the class c and $\mu_{i,j}$ is the degree of similarity between class i and j . Even though the similarity degree $\mu_{i,j} \in [0, 1]$ can be defined in various ways, using the simple heuristic $\mu_{i,j} = \frac{\min\{|C_i|, |C_j|\}}{\max\{|C_i|, |C_j|\}}$ allows for taking into account many useful intuitions, including the previously mentioned double role of intermediate classes as well as handling overlapping between classes of different types.

It is worth mentioning that for binary imbalanced data, many other approaches for detecting data difficulty factors have been proposed. For instance, detecting the difficulty of an example by computing its conditional probabilities by modelling the distribution of each feature with independent Gaussians Zhang et al. (2020). However, to the best of our knowledge, they have not been extended to the multi-class setting.

2.2 Selected methods for multi-class imbalanced data classification

As previously mentioned, the number of methods proposed for multi-class imbalanced data classification problems is still rather small in comparison to the plethora of methods for the binary counterpart. The proposed methods for multi-class imbalanced data can be roughly divided into decomposition methods and specialized approaches (Fernández et al., 2018).

The decomposition methods split the multi-class problem into several binary ones, later applying standard binary imbalanced learning techniques (Fernández et al., 2013). The most well-known decomposition approaches are one-vs-all (OVA) and one-vs-one (OVO). From a dataset with $|C|$ classes, OVA constructs $|C|$ binary problems that concern the detection of a selected class, i.e. except the selected class, the examples of all the other classes are treated as one, common class. On the other hand, OVO constructs a binary classification problem for every pair of classes by filtering out the examples of all the other classes. Both in OVA and OVO, a binary classifier is trained for each problem and their predictions are aggregated in the test phase.

The most basic resampling method for multi-class imbalanced data is Global-CS (Zhou & Liu, 2010), which oversamples all the classes to the size of the biggest one. To achieve this size, all class examples are duplicated iteratively and in the last iteration (when duplicating the whole class would cause exceeding the desired number of instances) a random subset of examples is duplicated.

More informed oversampling procedures include Static-SMOTE (Fernández-Navarro et al., 2011), which is a multi-class extension of a very popular binary oversampling method SMOTE (Chawla et al., 2002). In each iteration, Static-SMOTE selects the smallest class in the dataset and doubles its size through oversampling. The oversampled instances are constructed as random linear interpolations of two examples belonging to the minority class. SMOM (Zhu et al., 2017) also extends SMOTE to multi-class data with special clustering and example weighting. Another form of generating new artificial instances was introduced with Mahalanobis Distance-based Oversampling (MDO) (Abdi & Hashemi, 2015) technique. The method oversamples an instance by

constructing a new instance in a random point within the same Mahalanobis distance from the class center, which is calculated by taking the mean of all class's examples.

There are also some methods that, similarly to the method proposed in this paper, use Expectation-Maximization algorithm and Gaussian Mixture Model through its operation. One of such methods is SMOTE and Clustered Undersampling Technique (SCUT) (Agrawal et al., 2015) that combines minority class oversampling and majority class undersampling. For each majority class, the method fits a GMM model in order to detect clusters of majority class examples. Later, examples from each cluster are randomly undersampled to obtain a dataset in which each cluster is represented by the same number of instances. Additionally, the method oversamples minority class examples with the SMOTE technique. Another similar technique is SGM (Zhang et al., 2020), which also combined oversampling with SMOTE and undersampling based on GMM in the context of training Convolutional Neural Networks (Goodfellow et al., 2016).

The research on data difficulty factors for multi-class imbalanced data motivated Similarity Oversampling and Undersampling Preprocessing (SOUP) (Janicka et al., 2019), which leverages the example safe levels [see Eq. (1)] to concentrate its operation towards particular areas in the feature space. First, the method removes majority examples with the lowest safe levels, clearing the class overlapping area from majority instances. Next, SOUP duplicates the minority examples with the highest safe levels, strengthening the representation of minority class regions. Another, more recent method that also aims to clear overlapping area from majority class examples is Multi-Class Combined Cleaning and Re-sampling (MC-CCR) (Koziański et al., 2020). The method constructs a sphere around each minority example and expands it until a given energy budget is exhausted. Any majority examples that lie within such a sphere are pushed out of it, cleaning the area. Later, additional minority examples are randomly generated within these spheres.

2.2.1 Related methods for binary imbalanced data

Finally, we would like to discuss some methods for *binary* imbalanced data that are related to the proposed method. KDE sampling (Kamalov, 2020) uses Gaussian kernels to estimate the probability distribution of the minority class and sample new artificial minority instances from that distribution. However, the method does not consider data difficulty factors and, in particular, does not detect class subconcepts. Another method that also uses kernel estimation to sample minority instances is PDFOS (Gao et al., 2014). K-means-SMOTE (Douzas et al., 2018) uses a clustering algorithm to partition the dataset into subgroups and later applies SMOTE to oversample clusters with a considerable imbalanced ratio. The number of instances generated from a cluster depends on its estimated sparsity. Again, the method does not detect minority class subconcepts but rather splits the whole dataset into several parts (clusters contain mixed instances of both classes, not subconcepts of one class). It also does not use a generative model to construct new instances, however, it is worth noting that there are some other resampling approaches that do use generative models, but they are mostly proposed for vision tasks and deep learning classifiers (Mullik et al., 2019; Sampath et al., 2021) which are not considered in this work.

3 Gaussian mixture model sampling

3.1 Motivations

In this paper, we present Gaussian Mixture Model Sampling (GMMSampling), which fits a GMM model to each minority class and later samples entirely new instances from the fitted multivariate probability distribution. The method combines under- and over-sampling to clear the class overlapping regions from the most harmful majority instances and to strengthen the representation of minority classes. Contrary to the earlier proposed methods for multi-class imbalanced data, our approach automatically detects minority class subconcepts and ensures that every one of them is sufficiently represented in the resulting dataset. The design of our approach originated from several motivations.

First, the research on the data difficulty factors for multi-class imbalanced data (Lango & Stefanowski, 2022; Wang & Yao, 2012) suggests that effective methods for alleviating class imbalance must not only deal with high imbalance ratio but also focus on other data difficulty factors such as class overlapping, etc. For instance, the experiments presented in Janicka et al. (2019) demonstrated that SOUP preprocessing obtains better results in terms of G-mean measure than, e.g., Static-SMOTE and various decomposition approaches, despite the fact that SOUP does not use any complex example generation strategy. The method simply duplicates already existing examples or removes them; therefore, the whole strength of the method comes from the careful selection of the examples to remove or duplicate. Since the example selection process is guided by the information about data difficulty factors provided by the example safe levels, this result demonstrates the importance of taking data difficulty factors into account while designing new methods.

Even though the research on binary imbalanced data demonstrated that the minority class decomposition into several subconcepts is an important data difficulty factor (Jo & Japkowicz, 2004), the current research on multi-class imbalanced data did not pay much attention to properly handle it. When this difficulty factor occurs in a dataset, the examples of a given minority class, rather than forming a single compact group, are split into many potentially distant groups in the feature space. Since the minority class is underrepresented and their subconcepts can have different densities, some subconcepts could be even more severely underrepresented in the dataset. Therefore, the classifier during training will focus its operation on more prevalent subconcepts, possibly making some of the smaller ones entirely not represented in the model and limiting its classification performance on the minority class. Even though the earlier presented method for detecting data difficulty factors based on the example safe levels will probably assign lower values to examples in such subconcepts, the methods like SOUP will not distinguish them from other examples in the class overlapping regions. They will also fail to exploit the information that these examples constitute a group, and consecutively can not assure that every subconcept is well represented in the oversampled dataset, handling within-class imbalance.

Therefore, following earlier methods, GMMSampling utilizes the example safe levels to properly handle issues like, e.g., class overlapping, but also uses an additional method to detect minority class subconcepts. More specifically, it employs a well-known Expectation-Maximization (EM) algorithm to estimate a Gaussian Mixture Model for each minority class. Such a method of subconcept detection was adopted for several reasons. First, GMM is a universal density approximator (Goodfellow et al., 2016) which means that with a sufficient number of components, any smooth data density distribution can be approximated by it with any nonzero error. Second, since GMM is a statistical model trained by

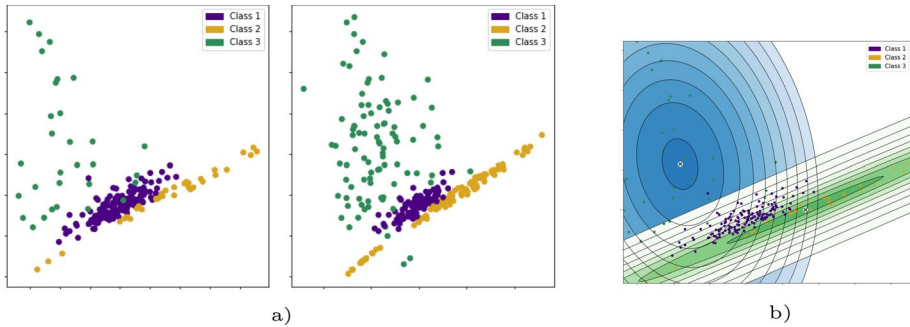


Fig. 1 Visualization of GMMSampling for new-thyroid dataset after reducing the dimensionality with PCA. **a** Demonstrates the dataset before and after preprocessing. **b** Shows fitted GMM for each minority class. Note that Class 1 (violet) is the majority class (Color figure online)

maximum likelihood estimation, the number of subconcepts (components) need not be specified in advance by a user or domain expert but can be detected automatically. Lastly, GMM can be estimated by EM in a computationally efficient way.

GMM model also enables a new interesting and statistically motivated way of oversampling minority classes. Most of the previously proposed methods for oversampling multi-class imbalanced data generate artificial minority instances by altering already existing instances. Some methods simply duplicate instances or construct random linear interpolations of them (Chawla et al., 2002). Other approaches apply more complex heuristics like rotating examples around the class center (Abdi & Hashemi, 2015) or adding to them random noise in a given sphere (Kozarski et al., 2020). Contrary to them, GMMSampling generates completely new minority instances by exploiting the fitted probability distribution of the dataset. Each component of the GMM model is a multivariate Normal distribution, so to generate an instance from a given detected class subconcept, it is sufficient to just take a random draw from it.

Note that even though some of the related methods do use GMM for resampling multi-class data, they rather use it to undersample majority data and not to generate new minority instances. These methods also do not detect minority class subconcepts nor deal with this source of data difficulty.

Finally, GMMSampling combines undersampling with oversampling, following the earlier observations (Agrawal et al., 2015) that both types of methods have their limitations. Oversampling can exacerbate the data noise or overgeneralize the minority class by producing too many artificial examples. On the other hand, too strong majority class undersampling can remove useful information from the dataset. Therefore, GMMSampling undersamples majority classes and oversamples minority ones to the common, average class size. This alleviates the aforementioned problems and at the same time combines the advantages of both types of approaches.

3.2 The proposed method

The pseudocode of Gaussian Mixture Model Sampling is presented in Algorithm 1. The visualisation of an example dataset before and after resampling it with the proposed approach is shown in Fig. 1.

Input: D - imbalanced dataset, C_{min} - set of minority classes, C_{maj} - set of majority classes

```

1:  $balanced\_size \leftarrow mean(\max_{c \in C_{min}} |D_c|, \min_{c \in C_{maj}} |D_c|)$ 
2: for  $c \in C_{min}$  do ▷ Oversampling
3:    $train\_set, dev\_set = split\_data(D_c)$ 
4:    $\ell, \ell_{best}, num\_subconcepts \leftarrow 0$ 
5:   repeat ▷ Detect the number of subconcepts
6:      $\ell_{best} = \ell$ 
7:      $num\_subconcepts \leftarrow num\_subconcepts + 1$ 
8:      $gmm \leftarrow fit\_GMM(train\_set, k = num\_subconcepts)$ 
9:      $\ell \leftarrow gmm.calculate\_likelihood(dev\_test)$ 
10:  until  $\ell_{best} < \ell$ 
11:   $gmm = fit\_GMM(D_c, k = num\_subconcepts)$ 
12:   $n \leftarrow balanced\_size - |D_c|$ 
13:  for  $i \in \{1, 2, \dots, num\_subconcepts\}$  do ▷ Generate new examples
14:     $Q^i \leftarrow$  calculate  $Q^i$  according to Eq. 3
15:     $sample \leftarrow$  random sample from  $i$ -th component (a multivariate
    Normal distribution) of size  $n \cdot Q^i$ 
16:     $D_c \leftarrow D_c \cup sample$ 
17:  end for
18: end for
19: for  $c \in C_{maj}$  do ▷ Undersampling
20:   while  $|D_c| > balanced\_size$  do
21:     $x \leftarrow$  find example with the lowest safe level in  $D_c$ 
22:     $D_c \leftarrow D_c \setminus x$ 
23:   end while
24: end for

```

In the first step, the method computes the desired size of all the classes in the resulting balanced dataset. This size is computed as the average of the biggest minority class size and the smallest majority class size. This ensures that all the minority classes will be oversampled and all the majority classes will be undersampled.

Later, the method iterates over all the minority classes C_{min} , fitting to each a Gaussian Mixture Model in order to detect class subconcepts. The procedure of fitting a GMM includes the selection of a proper number of class subconcepts (lines 3–11), which is determined automatically with an early stopping procedure. GMMSampling temporally divides the minority class data into train and development sets and fits a GMM model for a consecutively increasing number of components. When the value of the likelihood function on the development test begins to decrease, the process is terminated and the

current number of components is selected. The final GMM model is then fitted using all minority class examples¹ and the selected number of components.

Having fitted the probabilistic model of minority class with the estimated number of subconcepts, GMMSampling generates new artificial minority examples from each subconcept (lines 12–17). The goal of oversampling is twofold: to strengthen the representation of the most unsafe minority class subconcepts and to provide a sufficient representation of each minority class subconcept.

GMM is a mixture model that provides a soft clustering of the data. Therefore, instead of assigning an example to one specific group/subconcept, the model assigns the example to various groups but with different probabilities. For each example x , the probability $P(k|x) \propto P(x|k)P(k)$ of belonging to each group k can be calculated. Since the method's goal is to provide a more balanced representation of each minority class subconcept k , GMMSampling will enforce the uniform distribution of $P(k)$ during data resampling. This means that it will ignore subconcept cardinalities, and for each subconcept it will assume the same a priori probability that a random minority example is generated from it. Therefore, after assuming $P(k) = \frac{1}{K}$ where K is the detected number of subconcepts, one can further simplify $P(k|x) \propto P(x|k)P(k) \propto P(x|k)$.

If our goal was only to obtain an equal representation of each subconcept in the pre-processed dataset, we would simply generate the number of instances proportional to $\sum_{x \in X} P(k|x) \propto \sum_{x \in X} P(x|k)$ from each subconcept. Nevertheless, GMMSampling not only handles the issue of minority class decomposition but also takes into account other data difficulty factors such as class overlapping. The method computes a safe level for each minority example as defined in Eq. (1), aggregates them to assess the safeness of the whole class subconcept, and exploits this information to generate more examples from more unsafe subconcepts.

More concretely, the number of oversampled minority instances from each subconcept k is proportional to the value of Q^k defined as follows:

$$\tilde{Q}^k = \sum_{x \in X} (1 - \text{safe_level}(x)) \cdot P(x|k) \quad (2)$$

$$Q^k = \frac{\tilde{Q}^k}{\sum_{k=1}^K \tilde{Q}^k} \quad (3)$$

where K is the number of detected components, X is the set of examples from the considered class, $\text{safe_level}(x)$ is the safe level of example x and $P(x|k)$ is the probability of x from the k -th Gaussian mixture component. One can loosely interpret the value of Q^k as the estimated, averaged number of completely unsafe instances in the subconcept k . Therefore, GMMSampling will generate more instances for minority class subconcepts that contain many unsafe examples and fewer examples for relatively safe minority subconcepts. As mentioned earlier, the generation of new instances is just a random draw from the selected mixture component, i.e., from the multivariate Normal distribution.

The last step of the method is to undersample the majority classes (lines 19–24). GMMSampling computes for all majority instances its safe levels and removes majority instances starting from the most unsafe ones. This step ensures that majority instances

¹ Note that in the correctly designed machine learning experiments, the whole GMMSampling will be executed only on the training set. Therefore, the fitting of a GMM on the whole dataset provided to the method does not lead to the data leakage.

from class overlapping regions are removed, allowing easier learning of minority class sub-concepts in these areas.

The computational complexity of the method is as follows. For each minority class, a GMM model is estimated with the complexity of $O(i|D_c|K)$ where i is the number of iterations of the EM algorithm, K is the number of subconcepts and $|D_c|$ is the size of a given minority class. The computation complexity of safe levels depends on a data structure being used for performing k -nearest neighbors search. For instance, data structures like kd-tree offer the average-case complexity of $O(n \log n)$ and the worst-case complexity of $O(n^2)$. To sum up, the worst-case complexity of GMMSampling is $O(n^2 + nIK)$ where n is the size of the dataset, K is the upper bound for the number of subconcepts, and I is the upper bound for the number of EM iterations.

4 Experimental evaluation

4.1 Experimental setup

The usefulness of GMMSampling was verified on the collection of 15 imbalanced datasets that include eleven real and four artificial datasets that were commonly used in the related studies (Janicka et al., 2019; Lango & Stefanowski, 2018). The selected datasets are diversified, including problems with multi-majority, multi-minority, and gradual class size configurations; with different levels of class overlapping and with different numbers of classes. The collection includes datasets with various levels of class imbalance, with imbalance ratio from 2 to 164 and 27.6 on average. The characteristics of the selected datasets can be found in Table 1. The classification of classes into majority and minority categories was taken from the previous paper on SOUP (Janicka et al., 2019). Generally, two factors were taken into account: (1) the size of classes—majority classes should be significantly larger than minority ones. (2) The analysis of data difficulty factors detected by examples' safe levels - the minority classes should present considerable difficulty in order to be treated as such. A more detailed description of the evaluation of class difficulty can be found in Janicka et al. (2019).

In the experiments, GMMSampling was compared against five related resampling methods for multi-class imbalanced data: a classical oversampling procedure GlobalCS (Zhou & Liu, 2010); two methods that oversample minority classes by altering existing examples in a more complex ways: Static-SMOTE (S-SMOTE) (Fernández-Navarro et al., 2011) and Mahalanobis Distance-based Oversampling (MDO) (Abdi & Hashemi, 2015); as well as with two more recent methods that are motivated by the research on data difficulty factors: Similarity Oversampling and Undersampling Preprocessing (SOUP) (Janicka et al., 2019) and Multi-Class Combined Cleaning and Re-sampling (MC-CCR) (Koziański et al., 2020). Additionally, we included in the comparison five binary imbalanced methods which were extended to handle multiple classes in a one-against-all fashion, i.e. we have applied them iteratively to each minority class treating all the others as a single majority class. These binary methods include classic ones like Random Oversampling (ROS) and Random Undersampling (RUS), as well as approaches that use clustering (K-means-SMOTE (Douzas et al., 2018)), density estimation (KDE sampling (Kamalov, 2020)) and kNN analysis to clear class overlapping regions (All-kNN (Tomek, 1976)). The methods were selected to ensure a diversified set of compared approaches, but also the availability of their implementations was taken into account for practical reasons. For MC-CCR we used the

Table 1 Characteristics of multi-class imbalanced datasets used in the experiments

Dataset	Minority classes					Majority classes			
Cleveland_1	1	2	3	4	0				
	55	36	35	13	164				
Cleveland_2	2	3	4		0+1				
	36	35	13		219				
Contraceptive	2				1	3			
	333				333	511			
Ecoli	pp	imUimS	omomL		cpimL	im			
	52	37	25		145	77			
Glass	vwf	con	tab		bwf	bwnf	Head		
	17	13	9		70	76	29		
Hayes roth	3				1	2			
	31				65	64			
New thyroid	2	3			1				
	35	30			150				
Pageblocks	2	3	4	5	1				
	33	3	8	12	492				
Vehicle	bus	van			opel_saab				
	218	199			429				
Yeast	2	3	5	6	7	1	8	9	10
	20	30	35	44	51	463	168	244	429
Wine quality	7	8				5	6		
	199	81				681	638		
Art1	MIN1	MIN2			MAJ				
	120	240			840				
Art2	MIN1	MIN2			MAJ				
	120	240			840				
Art3	MIN1	MIN2			MAJ				
	120	240			840				
Art4	MIN1	MIN2			MAJ				
	120	240			840				

Names of classes are given in the first row, while their cardinalities in the second row

implementation provided by the authors (Koziarski et al., 2020), for KDE we used our custom implementation, and we used imbalanced-learn package for binary methods. For all the other methods, we used their implementations in the multi-imbalance library² (Gryca et al., 2020). The resampling methods were tested in combination with five classical classifiers: CART decision trees, Naive Bayes, k-nearest neighbors, Random Forest, and Gradient Boosting Trees.

The classification performance was measured with five popular metrics: balanced accuracy (average of class recalls), F-score (macro-averaged binary F-score for all

² The implementation of GMMSampling has been submitted to the multi-imbalance library (develop branch) and will be made available in a future official release of that library: <https://github.com/damian-horna/multi-imbalance>.

classes), G-mean (geometric mean of class recalls), MCC (Matthews correlation coefficient), AP (macro-averaged binary average precision that summarizes Precision-Recall curve). These measures were selected because they are commonly used in the related studies on imbalanced data, and also because there is no clear consensus on the most appropriate metrics for measuring classification performance on imbalanced data. F-score is very often used and recommended in some application areas (Baeza-Yates and Ribeiro-Neto, 1999), but some works argue that Matthews correlation coefficient is a more reliable classification performance indicator while working with imbalanced data (Chicco & Jurman, 2020). On the other hand, a theoretical analysis presented in Brzezinski et al. (2018) revealed that among the investigated measures (including MCC and F-score), only G-mean has the desired theoretical properties. Other works highlight the benefits of using balanced accuracy (Lango & Stefanowski, 2022) or Precision-Recall curves (Saito & Rehmsmeier, 2015).

All the experiments were performed in Python using implementations of classifiers from the sklearn library. The classification performances were evaluated with fivefold stratified cross-validation and averaged over 10 runs.

4.2 Results

The experimental results for the decision tree classifier and balanced accuracy measure are presented in Table 2, while the results of other classification methods and measures are available in the online appendix.³

The proposed method obtained the highest balanced accuracy value averaged over all the datasets. Similarly, the median of balanced accuracy values achieved by the method is the highest among the results obtained by the methods under study. The second-highest average scores were obtained by another method employing example safe levels - SOUP, with the average difference of 2%. On particular datasets, the improvement offered by GMMSampling over the second-best performing method was as large as 7 percentage points. The method provided the highest balanced accuracy value for 7 datasets and second-best for 6 datasets, while RUS, AkNN, KDE, Static-SMOTE, and SOUP were the best methods for one dataset only and MC-CCR obtained the best result for four datasets.

On G-mean, MCC and F-score measures, the proposed method also obtained the highest results averaged over all the datasets. For AP the best average result was obtained by K-means SMOTE but the difference to GMMSampling was lower than 0.001. GMMSampling also obtained a considerably higher median AP score than K-means SMOTE (1.92%).

To better summarize the results, the average ranks for all resampling methods were calculated and presented in Table 3. The approach presented in this paper obtains the lowest (i.e., the best) average rank for decision tree classifier and all investigated measures. It also obtains the best rank for almost all measures while using Naive Bayes classifier, except for G-mean where it ranks third. For kNN classifier GMMSampling obtains the best rank for balanced accuracy and MCC, ranks second for AP and F-score and third for G-mean. Finally, while using ensemble approaches (Random Forest, Gradient Boosting Trees) our method obtains best results on balanced accuracy and G-mean, second-best on F-score and ranks second/third for AP. Still, averaging over all classifiers and metrics, the proposed method obtains the lowest rank with a significant difference to the second-best method

³ <https://www.cs.put.poznan.pl/mlango/publications/GMMSampling.html>.

Table 2 The classification performance of decision tree classifier while combined with different resampling methods, measured with balanced accuracy

Dataset	Ours	RUS	AkNN	ROS	KM-SM	KDE	MDO	GI-CS	S-SM	SOUP	MC-CCR
Clev_1	0.357	0.255	0.253	0.275	<u>0.288</u>	0.283	0.276	0.218	0.273	0.281	0.251
Clev_2	<u>0.394</u>	0.339	0.347	0.314	0.336	0.405	0.357	0.350	0.349	0.387	0.346
Contra	0.513	0.468	0.476	0.478	0.488	0.488	0.485	0.471	0.475	0.487	<u>0.496</u>
Ecoli	0.724	0.671	0.728	0.679	0.697	0.719	<u>0.744</u>	0.681	0.711	0.733	0.766
Glass	<u>0.745</u>	0.586	0.550	0.720	0.581	0.670	0.591	0.661	0.736	0.743	0.747
Hayes-r	<u>0.875</u>	0.834	0.661	0.874	0.874	0.874	0.874	0.874	0.880	0.844	0.880
New t	0.944	0.893	0.899	0.911	0.906	0.921	<u>0.927</u>	0.871	0.921	<u>0.927</u>	0.910
Page	0.758	0.655	0.643	0.594	0.630	0.679	0.697	0.625	<u>0.742</u>	0.692	0.701
Vehicle	0.900	0.888	0.910	0.892	0.904	0.890	0.886	<u>0.905</u>	<u>0.905</u>	0.903	0.899
Wine	0.397	0.359	0.375	0.353	0.380	0.384	<u>0.392</u>	0.359	0.353	0.370	0.369
Yeast	0.475	0.424	0.468	0.431	0.427	0.433	0.456	0.411	0.415	<u>0.473</u>	0.472
Art1	<u>0.960</u>	0.934	0.948	0.949	<u>0.960</u>	0.951	0.957	0.953	0.949	0.953	0.964
Art2	0.753	0.739	<u>0.746</u>	0.703	0.713	0.698	0.730	0.695	0.706	0.713	0.693
Art3	<u>0.538</u>	0.526	0.524	0.503	0.530	0.537	0.533	0.512	0.506	0.552	0.507
Art4	<u>0.823</u>	0.824	0.802	0.771	0.768	0.791	0.793	0.768	0.777	0.798	0.790
Average	0.677	0.626	0.622	0.630	0.632	0.648	0.647	0.624	0.646	<u>0.657</u>	0.653
Median	0.745	0.655	0.643	0.678	0.630	0.680	0.697	0.661	0.712	<u>0.713</u>	0.701

The best results are bolded and the second-best results are underlined

(1.31). The Friedman tests performed for all measures and classifier pairs reject the null hypotheses about the lack of differences between resampling methods with $\alpha = 5\%$.

Furthermore, we also performed paired Wilcoxon tests between GMMSampling and other resampling methods under study. The results are presented in Table 4. The differences between the proposed approach and the baseline methods, i.e. RUS and ROS, are statistically significant for almost every measure and classification algorithm. Similarly, the statistical tests investigating the differences with K-means-SMOTE, KDE, MDO and All-kNN on balanced accuracy and G-mean measures indicate significant improvements for all classifiers. Moreover, the comparison of GMMSampling with another data difficulty-driven method SOUP also demonstrates statistically significant differences in almost all cases for all measures except G-mean. Taking into account all measures and classifiers, the differences were statistically significant for at least half of the considered variants for 9 out of 10 resampling methods (except Static-SMOTE). However, it is worth noting that the differences in average ranks between GMMSampling and Static-SMOTE are often considerable.

4.3 Ablation study

We also performed an ablation study to verify which elements of the proposed method contribute the most to the final result. We wanted to verify the importance of applying the newly proposed GMM-based example generation strategy and check whether the approach truly obtains high results due to better handling of minority class subconcepts. We also wanted to check how the formula for calculating how many examples will be oversampled

Table 3 Average ranks (like in the Friedman test) of various resampling methods for 5 classification measures and 5 classification algorithms: decision tree (DT), Naive Bayes (NB), k-nearest neighbors (kNN), random forest (RF) and gradient boosting trees (GBT)

Meas	Cl	Ours	RUS	AkNN	ROS	KM-SM	KDE	MDO	GI-CS	S-SM	SOUP	MC-CCR
BAcc	DT	2.13	8.20	6.60	8.37	6.27	5.33	4.83	8.27	6.57	<u>4.23</u>	5.20
	NB	4.33	5.50	7.40	4.70	7.90	7.77	6.73	6.73	<u>4.57</u>	4.80	5.57
	kNN	3.87	7.07	7.30	6.63	7.73	5.73	5.37	6.40	5.73	5.73	<u>4.43</u>
	RF	2.33	6.70	8.17	7.40	7.17	<u>4.37</u>	5.90	7.33	7.37	4.73	4.53
	GBT	3.10	6.90	7.93	7.87	6.87	5.53	5.00	8.27	5.60	4.67	<u>4.27</u>
G	DT	2.33	6.27	8.07	8.33	6.87	6.60	6.13	7.73	5.53	<u>3.47</u>	4.67
	NB	4.27	4.97	9.40	4.53	7.37	8.37	7.73	6.27	3.97	<u>4.00</u>	5.13
	kNN	4.60	6.60	8.67	6.10	8.27	6.67	6.20	5.67	5.07	4.00	<u>4.17</u>
	RF	3.13	4.20	8.73	8.33	7.80	5.73	5.87	7.20	6.73	<u>4.00</u>	4.27
	GBT	2.53	5.33	8.60	8.00	7.33	6.20	5.67	8.20	5.80	<u>4.00</u>	4.33
F	DT	4.13	9.73	7.20	6.40	4.70	5.23	4.93	5.93	5.87	7.67	<u>4.20</u>
	NB	2.40	7.23	7.93	6.87	4.97	5.23	5.93	8.13	<u>4.50</u>	6.80	6.00
	kNN	<u>4.07</u>	7.87	6.93	6.83	5.67	3.73	5.40	6.27	4.53	8.60	6.10
	RF	<u>3.93</u>	9.40	8.13	5.93	5.40	4.47	4.80	6.67	5.80	7.67	3.80
	GBT	<u>4.40</u>	9.53	8.13	6.23	4.93	4.80	4.63	7.27	5.27	7.40	3.40
AP	DT	<u>4.07</u>	8.40	6.73	7.73	3.47	4.27	4.33	7.33	6.13	7.20	6.33
	NB	4.13	8.33	5.53	6.87	6.20	4.53	5.67	7.93	<u>4.27</u>	6.80	5.73
	kNN	<u>3.47</u>	6.27	7.13	8.77	4.60	2.53	3.60	8.70	7.40	7.93	5.60
	RF	<u>4.47</u>	7.60	7.53	7.27	5.53	<u>4.47</u>	3.27	6.60	6.40	7.93	4.93
	GBT	3.93	9.53	7.53	7.00	6.27	<u>3.80</u>	3.27	6.67	5.20	7.87	4.93
MCC	DT	4.13	10.07	5.07	6.60	5.10	<u>4.23</u>	5.40	7.40	5.40	7.47	5.13
	NB	2.47	7.37	5.87	7.07	6.03	5.43	5.53	8.07	<u>5.03</u>	7.13	6.00
	kNN	3.67	8.13	6.73	7.70	4.57	<u>3.97</u>	4.33	6.67	5.40	8.00	6.83
	RF	4.40	10.00	8.07	6.73	4.00	<u>3.93</u>	3.87	6.27	6.07	8.47	4.20
	GBT	5.07	9.93	6.73	6.23	4.13	4.27	4.37	7.13	5.40	8.53	<u>4.20</u>
Average		3.65	7.65	7.44	6.98	5.97	5.09	5.15	7.16	5.58	6.36	<u>4.96</u>
Median		3.93	7.60	7.53	6.87	6.03	<u>4.80</u>	5.37	7.20	5.53	7.13	4.93

The best average ranks are bolded, the second-best are underlined and the third-best are in italics

from each subconcept influences the results, and whether both parts of it are necessary. Another research question was about the importance of majority class undersampling.

Therefore, we constructed four simplified versions of GMMSampling:

- GMMSampling without GMM—a method that does not detect class subconcepts and does not generate new instances from a GMM. Instead, the minority classes are oversampled by simply duplicating the examples. However, standard example safe levels are still used to, just like SOUP, guide the oversampling process towards strengthening the safest examples.
- GMMSampling without undersampling—a method that operates just like the original method, but do not perform data difficulty-driven undersampling of the majority class examples. In Algorithm 1, one must omit lines 19–24 to obtain this variant of the method.

Table 4 The results of paired Wilcoxon tests between GMMSampling and other resampling methods for 5 classification measures and 3 classification algorithms: decision trees (DT), Naive Bayes (NB), k-nearest neighbours (kNN), Random Forest (RF) and Gradient Boosting Trees (GBT)

Meas	Cl	RUS	AkNN	ROS	KM-SM	KDE	MDO	GI-CS	S-SM	SOUP	MC-CCR
BAcc	DT	0.000	0.001	0.000	0.000	0.001	0.002	0.000	0.000	0.010	0.026
	kNN	0.001	0.001	0.055	0.002	0.007	0.041	0.121	0.277	0.095	0.303
	NB	0.934	0.000	0.639	0.005	0.002	0.015	0.151	0.561	1.000	0.359
	RF	0.001	0.000	0.002	0.001	0.005	0.001	0.001	0.003	0.008	0.008
	GBT	0.038	0.002	0.007	0.003	0.003	0.002	0.004	0.010	0.055	0.095
G	DT	0.000	0.000	0.000	0.002	0.000	0.000	0.000	0.002	0.303	0.030
	kNN	0.030	0.000	0.151	0.004	0.005	0.008	0.421	0.978	0.847	0.847
	NB	0.804	0.004	0.720	0.020	0.001	0.018	0.359	0.762	0.599	0.978
	RF	0.041	0.000	0.001	0.001	0.002	0.010	0.001	0.003	0.169	0.018
	GBT	0.030	0.000	0.002	0.000	0.000	0.001	0.002	0.001	0.018	0.030
F	DT	0.000	0.018	0.035	0.303	0.330	0.524	0.073	0.208	0.003	0.934
	kNN	0.000	0.010	0.083	0.132	1.000	0.083	0.330	0.720	0.000	0.421
	NB	0.010	0.002	0.001	0.014	0.006	0.015	0.000	0.048	0.015	0.003
	RF	0.000	0.003	0.359	0.229	0.454	0.208	0.048	0.208	0.004	0.599
	GBT	0.000	0.012	0.107	0.303	0.208	0.359	0.055	0.720	0.004	0.978
AP	DT	0.004	0.073	0.022	0.978	0.561	0.524	0.041	0.252	0.002	0.035
	kNN	0.005	0.001	0.000	0.451	0.090	0.804	0.012	0.005	0.000	0.035
	NB	0.004	0.359	0.012	0.079	1.000	0.277	0.026	0.934	0.107	0.330
	RF	0.005	0.012	0.055	0.804	0.890	0.208	0.064	0.121	0.000	0.524
	GBT	0.000	0.041	0.022	0.030	0.277	0.890	0.064	0.252	0.000	0.083
MCC	DT	0.000	0.135	0.048	0.151	0.934	0.018	0.007	0.561	0.001	0.330
	kNN	0.000	0.018	0.008	0.187	0.346	0.303	0.030	0.277	0.000	0.041
	NB	0.002	0.064	0.001	0.002	0.008	0.012	0.000	0.002	0.008	0.001
	RF	0.001	0.005	0.169	0.720	0.454	0.720	0.169	0.252	0.001	0.679
	GBT	0.000	0.151	0.252	0.561	0.890	0.762	0.229	0.679	0.001	0.804

P-values below $\alpha = 5\%$ are bolded. Values are rounded to 3 decimal places, in particular values smaller than 0.0005 are reported as 0.000

- GMMSampling without calculating Q^k —a method that does not use Equations (2) and (3) to compute the number of examples to be oversampled from each class subconcept. Instead, the same number of examples is oversampled from each subconcept. More concretely, line 14 in Algorithm 1 will become $Q^k \leftarrow \frac{1}{K}$ where K is the number of subconcepts.
- GMMSampling without taking $P(x|k)$ into account while calculating Q —a method that does not use the information about probabilistic assignments to clusters. Instead of Eq. (2), a simplified formula is used $\tilde{Q}^k = \sum_{x \in Concept_k} (1 - safe_level(x))$.

The results for the decision tree classifier and G-mean measure are presented in Table 5. The best working method is the full version of GMMSampling algorithm, demonstrating that every part of the presented method is needed to achieve the best results. The worst working variant of the method is the variant that does not use the introduced example generation method and only exploits the information about data difficulty factors provided by

Table 5 Results of the ablation study for decision tree classifier and G-mean measure: average ranks over all datasets and average G-mean values

	Avg. rank	Average G-mean	Diff
GMMSampling	1.86	0.6689	N/A
GMMSampling without GMM	4.26	0.6105	-0.0584
GMMSampling without undersampling	3.80	0.6426	- 0.0263
GMMSampling without calculating Q (oversample equal number of examples from each subconcept)	2.80	0.6605	- 0.0084
GMMSampling without taking $P(x k)$ into account while calculating Q	2.26	0.6661	- 0.0028

Last column presents differences between average G-mean values vs. GMMSampling

example safe levels, i.e. do not take minority class subconcepts into account. This result shows that, indeed, the most significant improvement of classification performance comes from the main idea introduced in this paper i.e. from statistically motivated data generation strategy that handles the within-class imbalance and minority class decomposition.

Omitting the majority class undersampling step makes the second-biggest difference—the variant of the method without it obtained the second-worst average rank and the average value of G-mean dropped of about 2.6 percentage points compared to the full version of the method. Since majority class undersampling is driven by the detection of data difficulty factors such as class overlapping, this result shows its importance.

The proposed formula for calculating the number of examples to be oversampled from each cluster also proved to be useful. Even though the drop in terms of G-mean is not as significant as in the other variants, still the difference in average ranks with the original approach is about 1, meaning that, on average, the method is one place lower in the ranking than the full variant of the method. Finally, simplifying the formula for calculating Q^k made a small difference, but still, these results indicate that both parts of this formula are needed.

5 Summary and future works

In this paper, we introduced Gaussian Mixture Model Sampling for multi-class imbalanced data. The method uses the knowledge about data difficulty factors, in particular about minority class decomposition into several subconcepts, to adjust the resampling process to the difficulties encountered in a given dataset. It automatically detects the number of class subconcepts for each minority class and generates artificial minority examples in a novel and effective way. The performed experiments demonstrated that the proposed method obtains better G-mean, F-score, MCC, AP and balanced accuracy results than other related resampling methods for multi-class imbalanced data in the vast majority of cases.

The presented research can be further extended in several ways. First, the presented method relies on approximating the distribution of minority classes with Gaussian Mixture Models. However, since GMM models only support numerical attributes, such approximation is not very suitable for non-numerical data. Additionally, with the growing data dimensionality, the computational cost of fitting a GMM grows considerably. Both these issues can be solved by adding to the method an additional preprocessing step that would

embed non-numerical values into real coordinate space and reduce the dimensionality. Note that such a feature transformation could also consider data difficulty factors and try to find a feature space with as few data difficulties as possible.

Finally, the presented method fits a probabilistic model for each minority class separately. Therefore, it does not take into account the distributions of the other classes and potentially can fail to model complex scenarios with strongly overlapping minority classes. Even though the experimental analysis of data difficulty factors (Lango & Stefanowski, 2022) demonstrated that overlapping between minority classes is less harmful than other types of overlapping, properly handling such cases could also lead to some gains.

Author contributions Conceptualization, M. Lango; Methodology, I. Naglik and M. Lango; Software, I. Naglik; Investigation, I. Naglik, Formal analysis: M. Lango, Writing—Original Draft Preparation, M. Lango; Writing—Review & Editing, I. Naglik and M. Lango; Visualization, I. Naglik.

Funding The authors were supported by the PUT statutory funds under Grant No. 0311/SBAD/0743.

Availability of data and materials The real datasets used in the experiments are publicly available from UCI repository, the artificial datasets can be found in the repository of multi-imbalance (Grycza et al., 2020) library.

Availability of data and materials The implementation of the proposed method was submitted to multi-imbalance library.

Declarations

Conflict of interest The authors declare that they have no conflict of interest.

Ethical approval Not applicable.

Consent to participate. Not applicable.

Consent for publication Not applicable.

Open Access This article is licensed under a Creative Commons Attribution 4.0 International License, which permits use, sharing, adaptation, distribution and reproduction in any medium or format, as long as you give appropriate credit to the original author(s) and the source, provide a link to the Creative Commons licence, and indicate if changes were made. The images or other third party material in this article are included in the article's Creative Commons licence, unless indicated otherwise in a credit line to the material. If material is not included in the article's Creative Commons licence and your intended use is not permitted by statutory regulation or exceeds the permitted use, you will need to obtain permission directly from the copyright holder. To view a copy of this licence, visit <http://creativecommons.org/licenses/by/4.0/>.

References

- Abdi, L., & Hashemi, S. (2015). To combat multi-class imbalanced problems by means of over-sampling techniques. *IEEE Transactions on Knowledge and Data Engineering*, 28(1), 238–251.
- Agrawal, A., Viktor, H. L., & Paquet, E. (2015). Scut: Multi-class imbalanced data classification using smote and cluster-based undersampling. In: 7th international joint conference on knowledge discovery, knowledge engineering and knowledge management (IC3k) (Vol. 1, pp. 226–234). IEEE.
- Baeza-Yates, R., & Ribeiro-Neto, B. (1999). *Modern information retrieval* (Vol. 463). ACM Press.
- Branco, P., Torgo, L., & Ribeiro, R. P. (2016). A survey of predictive modeling on imbalanced domains. *ACM computing surveys (CSUR)*. <https://doi.org/10.1145/2907070>

- Brzezinski, D., Stefanowski, J., Susmaga, R., & Szczch, I. (2018). Visual-based analysis of classification measures and their properties for class imbalanced problems. *Information Sciences*, 462, 242–261. <https://doi.org/10.1016/j.ins.2018.06.020>
- Chawla, N. V., Bowyer, K. W., Hall, L. O., & Kegelmeyer, W. P. (2002). Smote: Synthetic minority over-sampling technique. *Journal of Artificial Intelligence Research*, 16, 321–357.
- Chicco, D., & Jurman, G. (2020). The advantages of the Matthews correlation coefficient (mcc) over f1 score and accuracy in binary classification evaluation. *BMC Genomics*, 21, 1–13.
- Douzas, G., Bacao, F., & Last, F. (2018). Improving imbalanced learning through a heuristic oversampling method based on k-means and smote. *Information Sciences*, 465, 1–20. <https://doi.org/10.1016/j.ins.2018.06.056>
- Fernández, A., García, S., Galar, M., Prati, R., Krawczyk, B., & Herrera, F. (2018). *Learning from imbalanced data sets*. Springer.
- Fernández, A., López, V., Galar, M., Del Jesus, M. J., & Herrera, F. (2013). Analysing the classification of imbalanced data-sets with multiple classes: Binarization techniques and ad-hoc approaches. *Knowledge-Based Systems*, 42, 97–110.
- Fernández-Navarro, F., Hervás-Martínez, C., & Antonio Gutiérrez, P. (2011). A dynamic over-sampling procedure based on sensitivity for multi-class problems. *Pattern Recognition*, 44(8), 1821–1833.
- Gao, M., Hong, X., Chen, S., Harris, C. J., & Khalaf, E. (2014). Pdfos: Pdf estimation based over-sampling for imbalanced two-class problems. *Neurocomputing*, 138, 248–259.
- Goodfellow, I., Bengio, Y., & Courville, A. (2016). *Deep learning*. MIT Press.
- Grycza, J., Horna, D., Klimczak, H., Lango, M., Plucinski, K., & Stefanowski, J. (2020) multi-imbalance: Open source python toolbox for multi-class imbalanced classification. In *ECML/PKDD*.
- He, H., & Garcia, E. A. (2009). Learning from imbalanced data. *IEEE Transactions on Knowledge and Data Engineering*, 21(9), 1263–1284.
- Janicka, M., Lango, M., & Stefanowski, J. (2019). Using information on class interrelations to improve classification of multiclass imbalanced data: A new resampling algorithm. *International Journal of Applied Mathematics and Computer Science*, 29(4), 769–781.
- Japkowicz N (2000) The class imbalance problem: Significance and strategies. In *Proceedings roc. of the international conference on artificial intelligence* (Vol. 56).
- Japkowicz, N. (2003) Class imbalance: Are we focusing on the right issue? In *Proceedings of the ICML'03 workshop on learning from imbalanced data sets*. ICML '03.
- Japkowicz, N., & Stephen, S. (2002). Class imbalance problem: A systematic study. *Intelligent Data Analysis Journal*, 6(5), 429–450.
- Jo, T., & Japkowicz, N. (2004). Class imbalances versus small disjuncts. *ACM SIGKDD Explorations Newsletter*, 6(1), 40–49.
- Kamalov, F. (2020). Kernel density estimation based sampling for imbalanced class distribution. *Information Sciences*, 512, 1192–1201. <https://doi.org/10.1016/j.ins.2019.10.017>
- Koziarski, M., Woźniak, M., & Krawczyk, B. (2020). Combined cleaning and resampling algorithm for multi-class imbalanced data with label noise. *Knowledge-Based Systems*, 204, 106223. <https://doi.org/10.1016/j.knosys.2020.106223>
- Krawczyk, B. (2016). Learning from imbalanced data: Open challenges and future directions. *Progress in Artificial Intelligence*, 5(4), 221–232.
- Lango, M., & Stefanowski, J. (2022). What makes multi-class imbalanced problems difficult? An experimental study. *Expert Systems with Applications*. <https://doi.org/10.1016/j.eswa.2022.116962>
- Lango, M., Napierala, K., & Stefanowski, J. (2017). Evaluating difficulty of multi-class imbalanced data. In M. Kryszkiewicz, A. Appice, D. Ślęzak, H. Rybinski, A. Skowron, & Z. W. Raś (Eds.), *Foundations of Intelligent Systems* (pp. 312–322). Springer.
- Lango, M. (2019). Tackling the problem of class imbalance in multi-class sentiment classification: An experimental study. *Foundations of Computing and Decision Sciences*, 44(2), 151–178.
- Lango, M., & Stefanowski, J. (2018). Multi-class and feature selection extensions of roughly balanced bagging for imbalanced data. *Journal of Intelligent Information Systems*, 50(1), 97–127.
- Moniz, N., & Monteiro, H. (2021). No free lunch in imbalanced learning. *Knowledge-Based Systems*, 227, 107222. <https://doi.org/10.1016/j.knosys.2021.107222>
- Mullick, S. S., Datta, S., & Das, S. (2019). Generative adversarial minority oversampling. In *The IEEE International Conference on Computer Vision (ICCV)*.
- Napierala, K., & Stefanowski, J. (2012). Identification of different types of minority class examples in imbalanced data. In: 7th international conference on hybrid artificial intelligent systems. Lecture notes in computer science (pp. 139–150). Springer.

- Prati, R. C., Batista, G. E. D. A. P. A., & Monard, M. C. (2004). Class imbalances versus class overlapping: an analysis of a learning system behavior. In *MICAI 2004: Advances in artificial intelligence* (pp. 312–321). Springer.
- Saito, T., & Rehmsmeier, M. (2015). The precision-recall plot is more informative than the roc plot when evaluating binary classifiers on imbalanced datasets. *PLoS One*, *10*(3), 0118432.
- Sampath, V., Maurtua, I., Aguilar Martín, J. J., & Gutierrez, A. (2021). A survey on generative adversarial networks for imbalance problems in computer vision tasks. *Journal of big Data*, *8*, 1–59.
- Santos, M. S., Abreu, P. H., Japkowicz, N., Fernández, A., & Santos, J. (2023). A unifying view of class overlap and imbalance: Key concepts, multi-view panorama, and open avenues for research. *Information Fusion*, *89*, 228–253. <https://doi.org/10.1016/j.inffus.2022.08.017>
- Stefanowski, J. (2013). Overlapping, rare examples and class decomposition in learning classifiers from imbalanced data. In *Emerging paradigms in machine learning* (pp. 277–306). Springer.
- Sun, N., Zhang, J., Rimba, P., Gao, S., Zhang, L. Y., & Xiang, Y. (2018). Data-driven cybersecurity incident prediction: A survey. *IEEE Communications: Surveys & Tutorials*, *21*(2), 1744–1772.
- Tomek, I. (1976). An experiment with the edited nearest-neighbor rule. *IEEE Transactions on Systems, Man, and Cybernetics SMC*, *6*(6), 448–452. <https://doi.org/10.1109/TSMC.1976.4309523>
- Wang, S., & Yao, X. (2012). Multiclass imbalance problems: Analysis and potential solutions. *IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics)*, *42*(4), 1119–1130.
- Wang, L., Lin, Z. Q., & Wong, A. (2020). Covid-net: A tailored deep convolutional neural network design for detection of covid-19 cases from chest x-ray images. *Scientific Reports*, *10*(1), 1–12.
- Weiss, G. M. (2004). Mining with rarity: A unifying framework. *ACM SIGKDD Explorations Newsletter*, *6*(1), 7–19.
- Wichitchan, S., Yao, W., & Yu, C. (2020). A new class of multivariate goodness of fit tests for multivariate normal mixtures. *Communications in Statistics—Simulation and Computation*. <https://doi.org/10.1080/03610918.2020.1808682>
- Zhang, J., Wang, T., Ng, W. W. Y., Pedrycz, W., Zhang, S., & Nugent, C. D. (2020). Minority oversampling using sensitivity. In *International Joint Conference on Neural Networks (IJCNN)*.
- Zhang, H., Huang, L., Wu, C. Q., & Li, Z. (2020). An effective convolutional neural network based on smote and gaussian mixture model for intrusion detection in imbalanced dataset. *Computer Networks*, *177*, 107315.
- Zhao, X.-M., Li, X., Chen, L., & Aihara, K. (2008). Protein classification with imbalanced data. *Proteins: Structure, Function, and Bioinformatics*, *70*(4), 1125–1132.
- Zhou, Z.-H., & Liu, X.-Y. (2010). On multi-class cost-sensitive learning. *Computational Intelligence*, *26*(3), 232–257.
- Zhu, T., Lin, Y., & Liu, Y. (2017). Synthetic minority oversampling technique for multiclass imbalance problems. *Pattern Recognition*, *72*, 327–340. <https://doi.org/10.1016/j.patcog.2017.07.024>

Publisher's Note Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.