Article

Improving Imbalanced Land Cover Classification with K-means SMOTE: Detecting and Oversampling Distinctive Minority Spectral Signatures

Joao Fonseca [1,*], Georgios Douzas [1], Fernando Bacao [1]

[1] NOVA Information Management School (NOVA IMS), Universidade Nova de Lisboa, Campus de Campolide, 1070-312 Lisboa, Portugal; gdouzas@novaims.unl.pt (G.D.); bacao@novaims.unl.pt (F.B.)

* Correspondence: jpfonseca@novaims.unl.pt (J.F.)

Abstract: Land cover maps are a critical tool to support informed policy development, planning, and resource management decisions. The ability to automatically produce Land Use/Land Cover maps, through the use of machine learning methods, will improve the accuracy and timeliness of decision-making. With significant upsides, the automatic production of Land Use/Land Cover maps has been a topic of interest for the remote sensing community for several years, but it is still fraught with technical challenges. One such challenge is the imbalanced nature of most remotely sensed data, where the number of samples of a few classes is significantly greater than the number of samples of the remaining classes. This asymmetric class distribution impacts negatively the performance of classifiers and adds a new source of error to the production of these maps. In this paper, we address the imbalanced learning problem, by using K-Means and SMOTE as an improved oversampling algorithm. K-Means SMOTE improves the quality of newly created artificial data by avoiding the generation of noisy data while effectively overcome data imbalance. The performance of K-Means SMOTE is compared to other popular oversampling methods using seven remote sensing benchmark datasets and a variety of classifiers and evaluation metrics. The results show that the proposed method consistently outperforms the remaining oversamplers and produces higher quality land cover classifications.

Keywords: LULC Classification; Imbalanced Learning; Oversampling; Data Generation; Clustering

Citation: Fonseca, J.; Douzas, G.; Bacao, F. Improving Land Cover Classification using oversampling: A Heuristic Method Based on K-Means and SMOTE . *Journal Not Specified* **2021**, *1*, 0. https://doi.org/

Received:
Accepted:
Published:

1. Introduction

The increasing amount of remote sensing missions granted the access to dense time series (TS) data at a global level and provides up-to-date, accurate land cover information [1]. This information is often materialized through Land Use and Land Cover (LULC) maps. While Land Cover maps define the biophysical cover found on the surface of the earth, Land Use maps define how it is used by humans [2]. Both Land Use and Land Cover maps constitute an essential asset for various purposes, such as land cover change detection, urban planning, environmental monitoring and natural hazard assessment [3]. However, the timely production of accurate and updated LULC maps is still a challenge within the remote sensing community [4]. LULC maps are produced based on two main approaches: photo-interpreted by the human eye, or automatic mapping using remotely sensed data and classification algorithms.

While photo-interpreted LULC maps rely on human operators and can be more reliable, they also present some significant disadvantages. The most important disadvantage is ~~are~~ the cost of production~~production costs~~, in fact photo-interpretation consumes significant resources, both in terms of money and time. Because of that, they are not frequently updated and not suitable for operational mapping over large areas. Finally, there is also the issue of overlooking rare or small-area classes, due to factors such as the minimum mapping unit being used.

Automatic mapping with classification algorithms based on machine-learning (ML) have been extensively researched and used to speed up and reduce the costs of the production process [3,5,6]. Improvements in classification algorithms are sure to have significant impact in the efficiency with which remote sensing imagery is used. Several challenges have been identified in order to improve automatic classification:

1. Improve the ability to handle high-dimensional datasets, in cases such as Multi-spectral TS composites high-dimensionality increases the complexity of the problem and creates a strain on computational power [7].
2. Improve class separability, as the production of an accurate LULC map can be hindered by the existence of classes with similar spectral signatures, making these classes difficult to distinguish [8].
3. Resilience to mislabelled LULC patches, as the use of photo-interpreted training data poses a threat to the quality of any LULC map produced with this strategy, since factors such as the minimum mapping unit tend to cause the overlooking of small-area LULC patches and generates noisy training data that may reduce the prediction accuracy of a classifier [9].
4. Dealing with rare land cover classes, due to the varying levels of area coverage for each class. In this case using a purely random sampling strategy will amount to a dataset with a roughly proportional class distribution as the one on the multi/hyperspectral image ~~landscape~~. On the other hand, the acquisition of training datasets containing balanced class frequencies is often unfeasible. This causes an asymmetry in class distribution, where some classes are frequent in the training dataset, while others have little expression [10,11].

The latter challenge is known, in machine learning, as the imbalanced learning problem [12]. It is defined as a skewed distribution of instances ~~observations~~ found in a dataset among classes in both binary and multi-class problems [13]. This asymmetry in class distribution negatively impacts the performance of classifiers, especially in multi-class problems. The problem comes from the fact that during the learning phase, classifiers are optimized to maximize an objective function, with overall accuracy being the most common one [14]. This means that instances ~~observations~~ belonging to minority classes contribute less to the optimization process, translating into a bias towards majority classes. As an example, a trivial classifier can achieve 99% overall accuracy on a binary dataset where 1% of the instances ~~observations~~ belong to the minority class if it classifies all instances ~~observations~~ as belonging to the majority class. This is an especially significant issue in the automatic classification of LULC maps, as the distribution of the different land-use classes tends to be highly imbalanced. Therefore, improvements in the ability to deal with imbalanced datasets will translate into important progress in the automatic classification of LULC maps.

There are three different types of approaches to deal with the class imbalance problem [6,15]:

1. Cost-sensitive solutions. Introduces a cost matrix to the learning phase with misclassification costs attributed to each class. Minority classes will have a higher cost than majority classes, forcing the algorithm to be more flexible and adapt better to predict minority classes.
2. Algorithmic level solutions. Specific classifiers are modified to reinforce the learning on minority classes. Consists on the creation or adaptation of classifiers.
3. Resampling solutions. Rebalances the dataset's class distribution by removing majority class instances and/or generating artificial minority instances. This can be seen as an external approach, where the intervention occurs before the learning phase, benefitting from versatility and independency from the classifier used.

~~It is important to note that~~ Since resampling strategies represent a set of methods that are detached from classifiers by~~have a significant advantage over the other approaches. By~~ operating at the data level, they~~resampling strategies~~ allow the use of any off the shelf algorithm, without the need for any type of changes or adaptions to the algorithm. This is a significant advantage especially considering that most users in remote sensing are not expert machine learning engineers.

Within resampling approaches there are three subgroups of approaches [6,15,16]:

1. Undersampling methods, which rebalance class distribution by removing instances from the majority classes.
2. Oversampling methods, which rebalance datasets by generating new artificial instances belonging to the minority classes.
3. Hybrid methods, which are a combination of both oversampling and undersampling, resulting in the removal of instances in the majority classes and the generation of artificial instances in the minority classes.

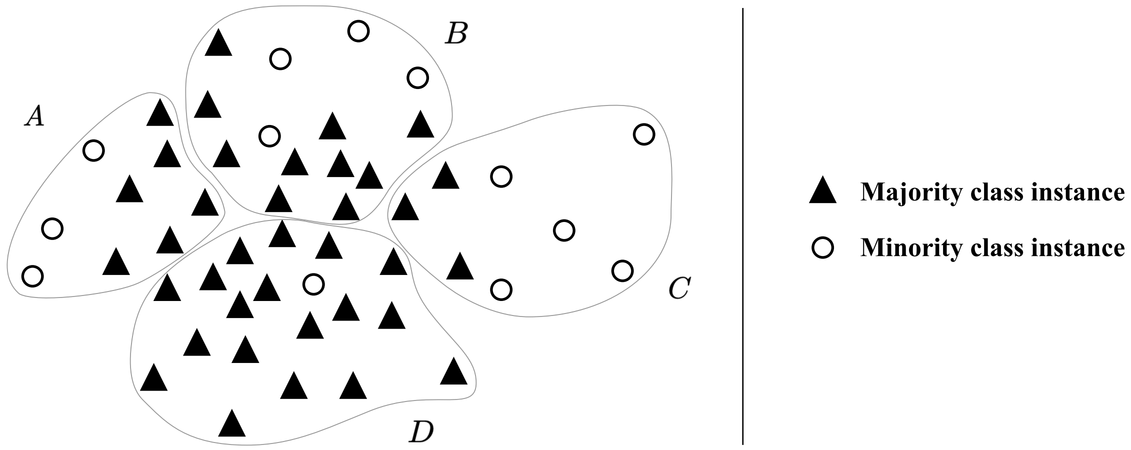
Resampling methods can be further distinguished between non-informed and heuristic (i.e., informed) resampling techniques [15–17]. The former consist of methods that duplicate/remove a random selection of data points to set class distributions to user-specified levels, and are therefore a simpler approach to the problem. The latter consists of more sophisticated approaches that aim to perform over/undersampling based on the points' contextual information within their data space.

The imbalanced learning problem is not new in machine learning but its relevancy has been growing, as attested by [18]. The problem has also been addressed in the context of remote sensing [19]. In this paper, we propose the application of a recent oversampler based on the Synthetic Minority Oversampling Technique (SMOTE) [20], the K-means SMOTE (K-SMOTE) [21] oversampler, to address the imbalanced learning problem in a multiclass context for LULC classification using various remote sensing datasets. Specifically, we use seven benchmark land use datasets that vary among agricultural and urban land use. The K-SMOTE algorithm presents significant advantages over other oversamplers, by coupling two different procedures in the generation of artificial data. The algorithm starts by clustering the instances ~~observations~~; next, the generation of the artificial instances ~~observations~~ is done taking into consideration the distribution of majority/minority cases in each individual cluster. The idea of starting with a clustering procedure before the data generation phase is important in remote sensing because the spectral signature of the different classes can change significantly based on the geographical area in which it is represented. In other words, the spectral signature of a specific class can vary greatly depending on the geography, meaning that often we will be facing within-class imbalance [22].

In fact, we can decompose class imbalance into two different types: between-class imbalance and within-class imbalance [21,23]. While the first refers to the overall asymmetry between majority and minority classes, the second results from the fact that in different areas of the input space there might be different levels of imbalance. Depending on the complexity of the input space, different subclusters of minority and majority instances may be present. In order to achieve a balance between minority and majority instances, these subclusters should be treated separately. Assuming that the role of a classifier is to create rules in such a way that it is able to isolate the different relevant sub-concepts that represent both the majority and minority classes, the classifier will create multiple disjunct rules that describe these concepts. If the input space is simple and the classes' instances are grouped together in a unique cluster, the classifier will only need to create (general) rules that comprise large portions of instances belonging to the same class. To the contrary, if the input space is complex and scatters through multiple small clusters, the classifier will need to learn a more complex set of (specific) rules, which can be seen in Figure 1. It is important to note that small clusters can happen both in the minority and majority class, although they will tend to be more frequent in the minority class due to its underrepresentation.

The efficacy of K-SMOTE is tested using different types of classifiers. To do so, we employ both commonly used and state-of-the-art oversamplers as benchmarking methods: Random oversampling (ROS), ~~Synthetic Minority Oversampling Technique~~

Figure 1. Example of a complex input space. In this example, a classifier would need to separate the minority class' samples across 4 distinguishable clusters (A, B, C and D).



~~(SMOTE) [20]~~ SMOTE and Borderline-SMOTE (B-SMOTE) [24]. Also as a baseline score we include classification results without the use of any resampling method.

This paper is organized in 5 sections: section 2 provides an overview of the state-of-art, section 3 describes the proposed methodology, section 4 covers the results and discussion and section 5 presents the conclusions taken from this study.

2. Imbalanced Learning Approaches

Imbalanced learning has been addressed in three different ways: over/undersampling, cost-sensitive training and changes/adaptations in the learning algorithms [6]. These approaches impact different phases of the learning process, while over/undersampling can be seen as a pre-processing step, cost-sensitive and changes in the algorithm imply a more customized and complex intervention in the algorithms. In this section, we focus on previous work related with resampling methods, while providing a brief explanation of cost-sensitive and algorithmic level solutions.

All of the most common classifiers used for LULC classification tasks [3,5] are sensitive to class imbalance [25]. Algorithm-based approaches typically focus on adaptations based on ensemble classification methods [26] or common non-ensemble based classifiers such as Support Vector Machines [27]. In [28], the reported results show that algorithm-based methods have comparable performance to resampling methods.

Cost-sensitive solutions refer to changes in the importance attributed to each instance through a cost matrix [29–31]. A relevant ~~common~~ cost sensitive solution [29] uses ~~is found in [29]. The authors use~~ the inverse class frequency (i.e., $1/|C_i|$, where C_i refers to the frequency of class i) to give higher weight to minority classes. Cui et al. [30] extended this method by adding a hyperparameter β to class weights as $(1-\beta)/(1-\beta^{|C_i|})$. When $\beta = 0$, no re-weighting is done. When $\beta \to 1$, weights are the inverse of the frequency class matrix. Another method [31] explores adaptations of Cross-entropy classification loss by adding different formulations of class rectification loss.

Resampling (over/undersampling) is the most common approach to imbalanced learning addressing the problem through data resampling in machine learning in general and remote sensing in particular [11]. The generation of artificial instances (i.e., augmenting the dataset), based on rare instances ~~examples~~, is done independently of any other step in the learning process. Once the procedure is applied, any standard machine learning algorithm can be used. Its simplicity makes resampling strategies particularly appealing for any user (especially the non-sophisticated user) interested in applying several classifiers, while maintaining a simple approach. It is also important to notice that over/undersampling methods can also be easily applied to multiclass problems, common in LULC classification tasks.

2.1. Non-informed resampling methods

There are two main non-informed resampling methods. Random Oversampling (ROS) generates artificial instances ~~observations~~ through random duplication of minority class instances. This method is used in remote sensing [32,33] for its simplicity, even though its mechanism makes the classifier prone to overfitting [34]. [33] found that using ROS returned worse results than keeping the original imbalance in their dataset.

A few of the recent remote sensing studies employed Random Undersampling (RUS) [35], which randomly removes instances ~~observations~~ belonging to majority classes. Although it's not as prone to overfitting as ROS, it incurs into information loss by eliminating instances ~~observations~~ from the majority class [11], which can be detrimental to the quality of the results.

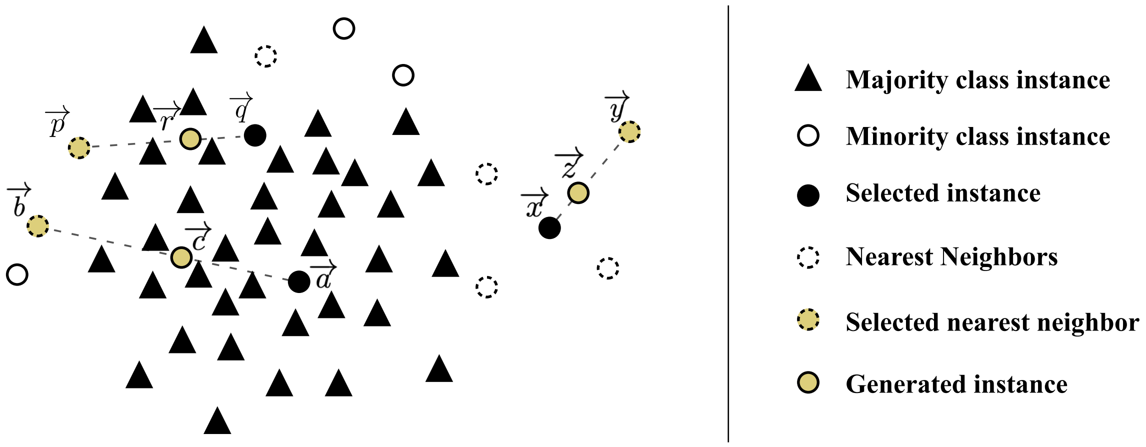
Another disadvantage of non-informed resampling methods is their performance-wise inconsistency across classifiers. ROS' impact on the Indian Pines dataset was found inconsistent between Random Forest Classifiers (RFC) and Support Vector Machines (SVM) and lowered the predictive power of an artificial neural network (ANN) [14]. Similarly, RUS is found to generally lead to a lower overall accuracy due to the associated information loss [14].

2.2. Heuristic methods

The methods presented in this section appear as a means to overcome the insufficiencies found in non-informed resampling. They use either local or global information to generate new, relevant, non-duplicated instances to populate the minority classes and/or remove irrelevant instances from majority classes. In a comparative analysis between over- and undersamplers' performance for LULC classification [36] using the rotation forest ensemble classifier, authors found that oversampling methods consistently outperform undersampling methods. This result led us to exclude undersampling from our study.

SMOTE [20] was the first heuristic oversampling algorithm to be proposed and has been the most popular one since then, likely due to its fair degree of simplicity and quality of generated data. It takes a random minority class sample and introduces synthetic instances ~~examples~~ along the line segment that join a random k minority class nearest neighbor to the selected sample. Specifically, a single synthetic sample $\vec{z}$ is generated within the line segment of a randomly selected minority class instance ~~observation~~ $\vec{x}$ and one of its k nearest neighbors $\vec{y}$ such that $\vec{z} = \alpha\vec{x} + (1-\alpha)\vec{y}$, where α is a random real number~~floating point~~ between 0 and 1, as shown in Figure 2.

Figure 2. Example of SMOTE's data generation process. SMOTE randomly selects instance $\vec{x}$ and one of its nearest neighbors $\vec{y}$ to produce $\vec{z}$. Noisy instance $\vec{r}$ was generated by randomly selecting $\vec{q}$ and nearest neighbor $\vec{p}$ from a different minority class cluster. Noisy instance $\vec{c}$ was generated by randomly selecting the noisy minority class instance $\vec{a}$ and one of its nearest neighbors $\vec{b}$.



A number of studies implement SMOTE within the LULC classification context and reported improvements on the quality of the trained predictors [37,38]. Another study proposes an adaptation of SMOTE on an algorithmic level for deep learning applications [39]. This method combines both typical computer vision data augmentation techniques, such as image rotation, scaling and flipping on the generated instances to populate minority classes. Another algorithmic implementation is the variational semi-supervised learning model [40]. It consists of a generative model that allows learning from both labeled and unlabeled instances while using SMOTE to balance the data.

Despite SMOTE's popularity, its limitations have motivated the development of more sophisticated oversampling algorithms [21,24,41–44]. [41] identify four major weaknesses of the SMOTE algorithm, which can be summarized as:

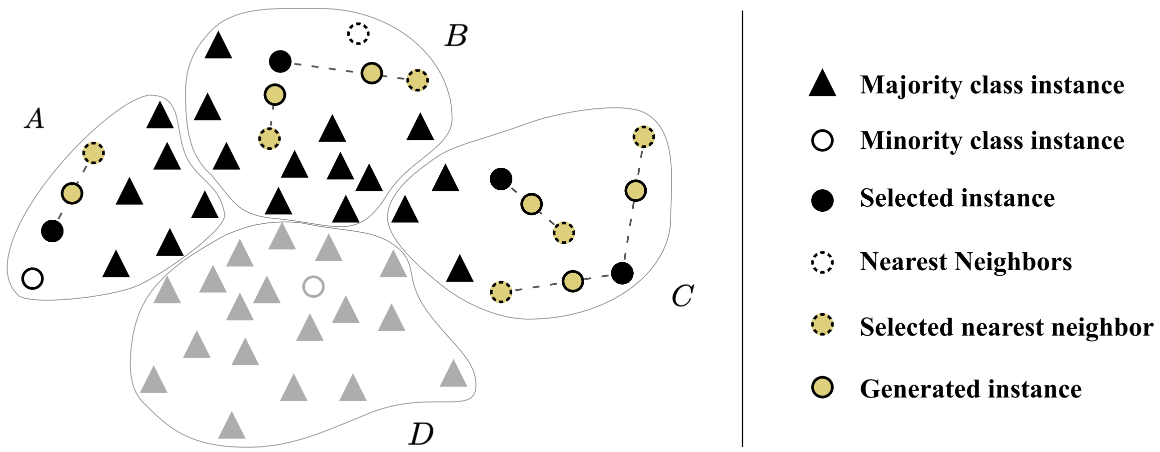
1. Generation of noisy instances due to random selection of a minority instance ~~observation~~ to oversample. The random selection of a minority instance ~~observation~~ makes SMOTE oversampling prone to the amplification of existing noisy data. This has been addressed by variants such as B-SMOTE [24] and ADASYN [44].
2. Generation of noisy instances due to the selection of the k nearest neighbors. In the event an instance ~~observation~~ (or a small number thereof) is not noisy but is isolated from the remaining clusters, known as the "small disjuncts problem" [45], much like sample $\overrightarrow{b}$ from Figure 2, the selection of any nearest neighbor of the same class will have a high likelihood of producing a noisy sample.
3. Generation of nearly duplicated instances. Whenever the linear interpolation is done between two instances ~~observations~~ that are close to each other, the generated instance becomes very similar to its parents and increases the risk of overfitting. G-SMOTE [41] attempts to address both the k nearest neighbor selection mechanism problem as well as the generation of nearly duplicated instances problem.
4. Generation of noisy instances due to the use of instances ~~observations~~ from two different minority class clusters. Although an increased k could potentially avoid the previous problem, it can also lead to the generation of artificial data between different minority clusters, as depicted Figure 2 with the generation of point $\overrightarrow{r}$ using minority class instances ~~observations~~ $\overrightarrow{p}$ and $\overrightarrow{q}$. Cluster-based oversampling methods attempt to address this problem.

This last issue, the generation of noisy instances due to the existence of several minority class clusters, is particularly relevant in remote sensing. It is frequent that instances belonging to the same minority class can have different spectral signatures, meaning that they will be clustered in different parts of the input space. In this context, the use of SMOTE will lead to the generation of noisy instances of the minority class. This problem can be efficiently mitigated through the use of a cluster-based oversampling. According to our literature review cluster-based oversampling approaches have never been applied in the context of remote sensing. On the other hand, while there are several references of the application of cluster-based oversampling in the context of machine learning, there is no such an application for the multiclass case, a fundamental requirement for the application of oversampling in the context of LULC.

Cluster-based oversampling approaches introduce an additional layer to SMOTE's selection mechanism, which is done through the inclusion of a clustering process. This ensures that both between-class data balance and within-class ~~within each class~~ balance is preserved. The self-organizing map oversampling (SOMO) [43] algorithm transforms the dataset into a 2-dimensional input, where the areas with the highest density of minority samples are identified. SMOTE is then used to oversample each of the identified areas separately. ClUstered REsampling SMOTE (CURE-SMOTE) ~~CURE-SMOTE~~ [42] applies a hierarchical clustering algorithm to discard isolated minority instances before applying SMOTE. Although it avoids noise generation problems, it ignores within-class data distribution. Another method [46] uses K-means to cluster the entire input space and applies SMOTE to clusters with the fewest instances ~~observations~~, regardless of their class label. The label of the generated instance ~~observation~~is copied from one of its parents. This method cannot ensure a balanced dataset since class imbalance is not specifically addressed, but rather dataset imbalance.

K-SMOTE [21] avoids noisy data generation by modifying the data selection mechanism. It employs k-means clustering to identify safe areas using cluster-specific Imbalance Ratio (IR, defined by $\frac{count(C_{majority})}{count(C_{minority})}$) and determine the quantity of generated samples per cluster based on a density measure. These samples are finally generated using the SMOTE algorithm. The K-SMOTE's data generation process is depicted in Figure 3. Note that the number of samples generated for each cluster varies according to the sparsity of each cluster (the sparser the cluster is, the more samples will be generated) and a cluster is rejected if the cluster's IR surpasses the threshold. Therefore, this method can be combined with any data generation mechanism, such as G-SMOTE. Also K-SMOTE includes the SMOTE algorithm as a special case when the number of clusters is set to one. Consequently, K-SMOTE is always guaranteed to return results as good as or better than SMOTE.

Figure 3. Example of K-SMOTE's data generation process. Clusters A, B and C are selected for oversampling, whereas cluster D was rejected due to its high imbalance ratio. The oversampling is done using the SMOTE algorithm and the k nearest neighbors selection only considers instances ~~observations~~ within the same cluster.



Although no other study was found to implement cluster-based oversampling, another study [19] compared the performance of SMOTE, ROS, ADASYN, B-SMOTE and G-SMOTE in a highly imbalanced LULC classification dataset. The authors found that G-SMOTE consistently outperformed the remaining oversampling algorithms regardless of the classifier used.

This paper's main contributions are:

- Propose a cluster-based multiclass oversampling method appropriate for LULC classification and compare its performance with the remaining oversamplers in a multiclass context with seven benchmark LULC classification datasets. Allows us to check the oversamplers' performance across benchmark LULC datasets.
- Introducing a cluster-based oversampling algorithm within the remote sensing domain, as well as comparing its performance with the remaining oversamplers in a multiclass context.
- Make available to the remote sensing community the implementation of the algorithm in a Python library and the experiment's source code.

3. Methodology

The purpose of this work is to understand the performance of K-SMOTE as opposed to other popular and/or state-of-the-art oversamplers for LULC classification. This is done using~~To do so, we use~~ 7 ~~LULC~~ datasets with predominantly land use information, along with 3 evaluation metrics and ~~3~~ 5 classifiers to evaluate the performance of oversamplers. In this section we describe the datasets, evaluation metrics, oversamplers, classifiers and software used as well as the procedure developed.

3.1. Datasets

The datasets used were extracted from publicly available hyperspectral scenes. Information regarding each of these scenes is provided in this subsection. The data collection and preprocessing pipeline is shown in Figure 4 and is common to all hyperspectral scenes:~~A similar data preprocessing procedure was used for each scene:~~

1. Data collection of publicly available hyperspectral scenes. The original hyperspectral scenes and ground truth data were collected from a single publicly available data repository available here.
2. Conversion of each hyperspectral scene to a structured dataset and removal of instances with no associated LULC class. This done to reshape the dataset from $(h, w, b + gt)$ into a conventional dataframe of shape $(h * w, b + gt)$, where gt, h, w and b represents the ground truth, height, width and number of bands in the scene, respectively. The pixels without ground truth information are discarded from further analysis
3. Stratified random sampling to maintain similar class proportions on a sample of 10% of each dataset. This is done by computing the relative class frequencies in the original hyperspectral scene (minus the class representing no ground truth availability) and retrieving a sample that ensures the original relative class frequencies remain unchanged.
4. Removal of instances belonging to a class with frequency lower than 20 or higher than 1000. This is done to maintain the datasets to a practicable size due to computational constraints, while conserving the relative LULC class frequencies and data distribution.
5. Data normalization using the MinMax scaler. This ensures all features (i.e., bands) are in the same scale. In this case, the data was rescaled between 0 and 1.

Figure 4. Data collection and preprocessing pipeline.

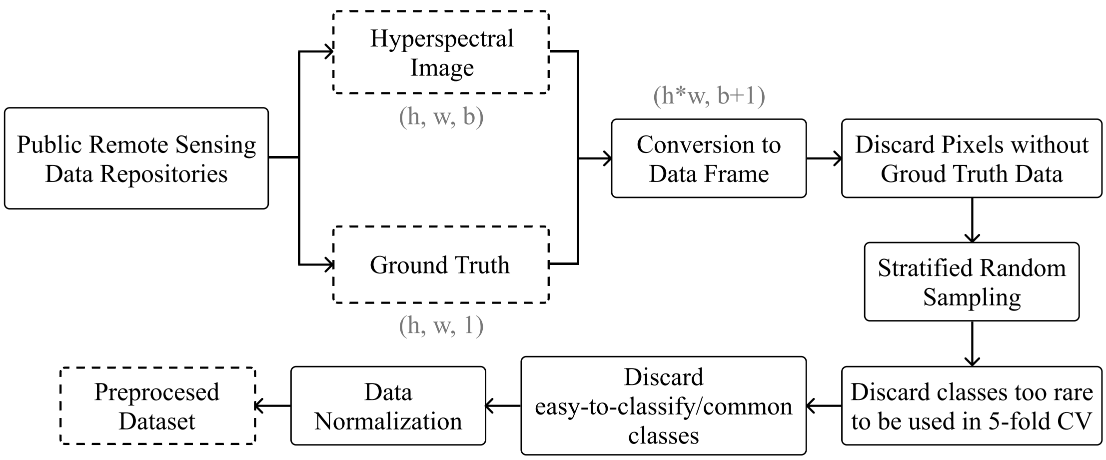


Table 1 provides a description of the final datasets used for this work, sorted according to its IR. Figure 5 shows the original hyperspectral scene out of which the dataset used in this experiment were extracted. In the representation of the ground truth of these scenes, the blue regions in the ground truth of each hyperspectral scene represent unlabeled regions (i.e., no ground truth is available). Particularly, in the Botswana and Kennedy Space Center scenes the truth was photointerpreted in more limited regions of

Table 1: Description of the datasets used for this experiment.

Dataset	Features	Instances	Min. Instances	Maj. Instances	IR	Classes
Botswana	145	288	20	41	2.05	11
Pavia Centre	102	3898	278	879	3.16	7
Kennedy Space Center	176	497	23	80	3.48	11
Salinas A	224	535	37	166	4.49	6
Pavia University	103	2392	89	679	7.63	8
Salinas	224	4236	91	719	7.9	15
Indian Pines	220	984	21	236	11.24	11

the scene. However, the scenes are still represented as they are in order to maintain a standardized analysis over all datasets extracted for the experiment.

Botswana

The Botswana scene was acquired by the Hyperion sensor on the NASA EO-1 satellite over the Okavango Delta, Botswana in 2001-2004 at a 30m spatial resolution. Data preprocessing was performed by the UT Center for Space Research. The scene comprises a 1476 $\times$ 256 pixels with 145 bands and 14 classes regarding land cover types in seasonal and occasional swamps, as well as drier woodlands (see figure 6a).

Pavia Center~~re~~ and University

Both Pavia Center and University scenes were acquired by the ROSIS sensor. These scenes are located in Pavia, northern Italy. Pavia Center~~re~~ is a 1096 $\times$ 1096 pixels image with 102 spectral bands, whereas Pavia University is a 610 $\times$ 610 pixels image with 103 spectral bands. Both images have a geometrical resolution of 1.3m ~~meters~~ and their ground truths are composed of 9 classes each (see Figures 6b and 6c).

Kennedy Space Center

The Kennedy Space Center scene was acquired by the AVIRIS sensor over the Kennedy Space Center, Florida, on March 23, 1996. Out of the original 224 bands, water absorption and low SNR bands were removed and a total of 176 bands at a spatial resolution of 18m are used. The scene is a 512 $\times$ 614 pixel image and contains a total of 16 classes (see figure 6d).

Salinas and Salinas-A

These scenes were collected by the AVIRIS sensor over Salinas Valley, California and contain at-sensor radiance data. Salinas is a 512 $\times$ 217 pixels image with 224 bands and 16 classes regarding vegetables, bare soil and vineyard fields (see Figure 6e). Salinas-A, a subscene of Salinas, comprises 86 $\times$ 83 pixels and contains 6 classes regarding vegetables (see Figure 6f). These scenes have a geometrical resolution of 3.7m ~~meters~~.

Indian Pines

The Indian Pines scene [47] was collected on June 12, 1992 and consists of AVIRIS hyperspectral image data covering the Indian Pine Test Site 3, located in North-western Indiana, USA. As a subset of a larger scene, it is composed of 145 $\times$ 145 pixels (see Figure 6g) and 220 spectral reflectance bands in the wavelength range 400 to 2500 nanometers at a spatial resolution of 20m. Approximately two thirds of this scene is composed by agriculture and the other third is composed of forest and other natural perennial vegetation. Additionally, the scene also contains low density buildup areas.

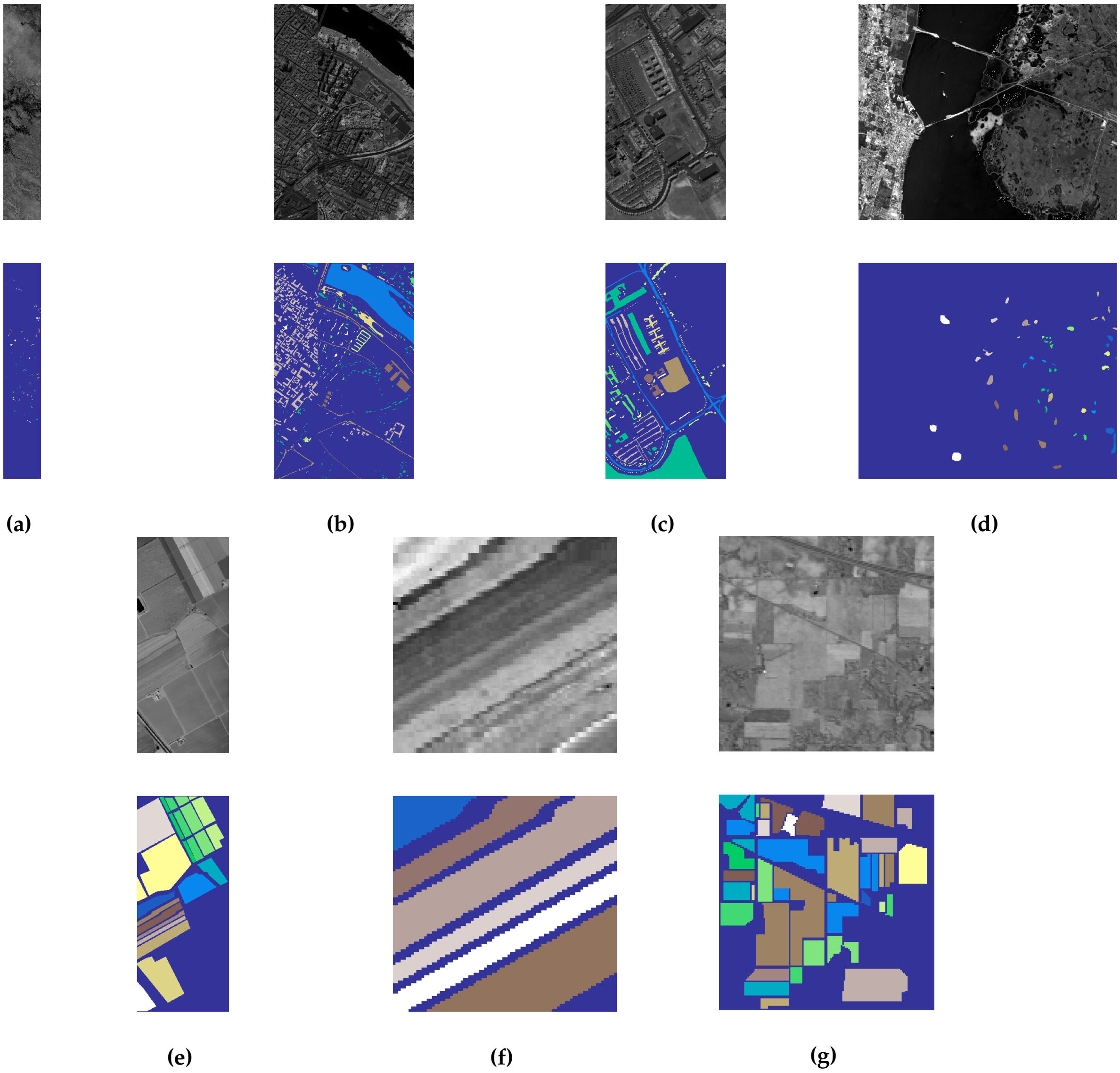


Figure 5. Gray scale visualization of a band (top row) and ground truth (bottom row) of each scene used in this study. (a) Botswana, (b) Pavia Cent~~erre~~, (c) Pavia University, (d) Kennedy Space Center, (e) Salinas, (f) Salinas A, (g) Indian Pines.

3.2. Machine Learning Algorithms

To assess the quality of the K-SMOTE algorithm, three other oversampling algorithms were used for benchmarking. ROS and SMOTE were chosen for their simplicity and popularity. B-SMOTE chosen as a popular variation of the SMOTE algorithm. We also include the classification results of no oversampling (NONE) as a baseline.

To assess the performance of each oversampler, we use the classifiers Logistic Regression (LR) [48], K-Nearest Neighbors (KNN) [49]~~, Decision Tree (DT) [50], Gradient Boosting Classifier (GBC) [51]~~ and Random Forest (RF) [52]. This choice was based on the classifiers' popularity for LULC classification, learning type and training time [5,14].

3.3. Evaluation Metrics

Most of the satellite-based LULC classification studies (nearly 80%) employ *Overall Accuracy* (OA) and the *Kappa Coefficient* [5]. Although, some authors argue that both evaluation metrics, even when used simultaneously, are insufficient to fully address the area estimation and uncertainty information needs [53,54]. Other metrics like User's

Accuracy (or *Precision*) and Producer's Accuracy (or *Recall*) are also common metrics to evaluate per-class prediction power. These metrics consist of ratios employing the True and False Positives (*TP* and *FP*, number of correctly/incorrectly classified instances ~~observations~~ of a given class) and True and False Negatives (*TN* and *FN*, number of correctly/incorrectly classified instances ~~observations~~ as not belonging to a given class). These metrics are formulated as $Precision = \frac{TP}{TP+FP}$ and $Recall = \frac{TP}{TP+FN}$. While metrics like OA and *Kappa Coefficient* are significantly affected by imbalanced class distributions, *F-Score* is less sensitive to data imbalance and a more appropriate choice for performance evaluation [55].

The datasets used present significantly high IRs (see Table 1). Therefore, it is especially important to attribute equal importance to the predictive power of all classes, which does not happen with OA and *Kappa Coefficient*. In this study, we employ 3 evaluation metrics: 1) *G-mean*, since it is not affected by skewed class distributions, 2) *F-Score*, as it proved to be a more appropriate metric for this problem when compared to other commonly used metrics [55], and 3) *Overall Accuracy*, for discussion purposes.

- The *G-mean* consists of the geometric mean of $Specificity = \frac{TN}{TN+FP}$ and *Sensitivity* (also known as *Recall*). For multiclass problems, The *G-mean* is expressed as:

$$\textit{G-mean} = \sqrt{\overline{Sensitivity} \times \overline{Specificity}}$$

- *F-score* is the harmonic mean of *Precision* and *Recall*. The *F-score* for the multi-class case can be calculated using their average per class values [56]:

$$\textit{F-score} = 2\frac{\overline{Precision} \times \overline{Recall}}{\overline{Precision} + \overline{Recall}}$$

- *Overall Accuracy* is the number of correctly classified instances ~~observations~~divided by the total amount of instances ~~observations~~. Having c as the label of the various classes, *Accuracy* is given by the following formula:

$$Accuracy = \frac{\sum_{c} \mathrm{TP}_c}{\sum_{c} (\mathrm{TP}_c + \mathrm{FP}_c)}$$

3.4. Experimental Procedure

The procedure for the experiment started with the definition of a hyperparameter search grid, where a list of possible values for each relevant hyperparameter in both classifiers and oversamplers is stored. Based on this search grid, all possible combinations of oversamplers, classifiers and hyperparameters are formed. Finally, for each dataset, hyperparameter combination and initialization we use the evaluation strategy shown in Figure 7: k-fold cross-validation strategy where $k = 5$ to train each model defined and save the averaged scores of each split.

In the 5-fold cross validation strategy, a ~~each~~ combination of oversampler, classifier and hyperparameters vector is fit 5 times ~~(once for each fold)~~ per dataset. Before the training phase, the ~~Each time, an oversampler will use the~~ training set (containing $\frac{4}{5}$ of the dataset) ~~to generate a set with artificial data, which is appended to the original training set in order to generate a training dataset~~ is oversampled using one of the methods described (except for the baseline method NONE), creating an augmented dataset with the exact same number of instances ~~observations~~ for each class. The newly formed training dataset is used to train the classifier and the test set ($\frac{1}{5}$ of the dataset ~~, the remaining fold~~) is used to evaluate the performance of the classifier. The evaluation scores are then averaged over the 5 times the process is repeated. The range of hyperparameters used are shown in table 3.

Figure 7. Experimental procedure. The performance metrics are averaged over the 5 folds across each of the 3 different initializations of this procedure for a given combination of oversampler, classifier and hyperparameter definition.

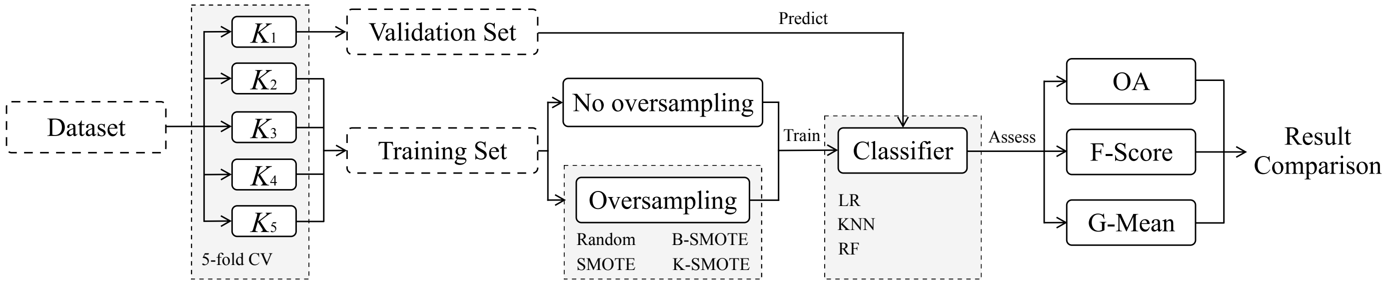


Table 3: Hyper-parameters grid. * One cluster is generated in total, a corner case that mimics the behavior of SMOTE

Classifier	Hyperparameters	Values
LR	maximum iterations	10000
KNN	# neighbors	3, 5, 8
RF	maximum depth	None, 3, 6
	# estimators	50, 100, 200
Oversampler		
K-SMOTE	# neighbors	3, 5
	# clusters (as % of number of instances ~~observations~~)	1*, 0.1, 0.3, 0.5, 0.7, 0.9
	Exponent of mean distance	auto, 2, 5, 7
	IR threshold	auto, 0.5, 0.75, 1.0
SMOTE	# neighbors	3, 5
BORDERLINE SMOTE	# neighbors	3, 5

3.5. Software Implementation

The experiment was implemented using the Python programming language, using the Scikit-Learn [57], Imbalanced-Learn [58], Geometric-SMOTE, Cluster-Over-Sampling and Research-Learn libraries. All functions, algorithms, experiments and results are provided at the GitHub repository of the project.

4. Results & Discussion

When evaluating the performance of an algorithm across multiple datasets, it is generally recommended to avoid direct score comparisons and use classification rankings instead [59]. This is done by assigning a ranking to oversamplers based on the different combinations of classifier, metric and dataset used. These rankings are also used for the statistical analyses presented in Section 4.2.

The rank values are assigned based on the mean validation scores resulting from the experiment described in Section 3. The averaged ranking results are computed over 3 different initialization seeds and a 5 fold cross validation scheme, returning a real number ~~float value~~within the interval $[1,5]$. The mean rankings are presented in Table 4.

4.1. Results

~~The mean ranking results show that K-SMOTE consistently presents the best results for every classifier and performance metric used.~~ The ~~quantitative results of this analysis is~~ mean ranking of oversamplers is presented in Table 4. This ranking was computed by averaging the ranks of the mean cross-validation scores per dataset, oversampler and classifier. K-SMOTE achieves the best mean ranking across datasets with low standard deviation.~~In addition to its superior performance, in the majority of the cases K-SMOTE's mean ranking has a lower standard deviation than any of the remaining methods, and particularly when opposed to SMOTE (the best performing benchmark method).~~

Table 4: Results for mean ranking of oversamplers across datasets.

Classifier	Metric	NONE	ROS	SMOTE	B-SMOTE	K-SMOTE
LR	Accuracy	3.14 ± 0.47	3.00 ± 0.49	3.29 ± 0.42	4.29 ± 0.43	$\mathbf{1.29 \pm 0.18}$
LR	F-score	3.86 ± 0.26	2.71 ± 0.47	3.00 ± 0.44	4.29 ± 0.42	$\mathbf{1.14 \pm 0.14}$
LR	G-mean	4.29 ± 0.18	2.57 ± 0.48	2.57 ± 0.3	4.29 ± 0.42	$\mathbf{1.29 \pm 0.18}$
KNN	Accuracy	2.14 ± 0.55	4.14 ± 0.34	3.43 ± 0.43	3.86 ± 0.34	$\mathbf{1.43 \pm 0.2}$
KNN	F-score	2.86 ± 0.4	4.29 ± 0.29	2.71 ± 0.61	3.86 ± 0.34	$\mathbf{1.29 \pm 0.18}$
KNN	G-mean	3.57 ± 0.48	3.86 ± 0.4	1.86 ± 0.46	4.14 ± 0.26	$\mathbf{1.57 \pm 0.2}$
RF	Accuracy	3.29 ± 0.52	3.14 ± 0.4	3.14 ± 0.51	4.29 ± 0.29	$\mathbf{1.14 \pm 0.14}$
RF	F-score	3.86 ± 0.46	3.14 ± 0.51	3.00 ± 0.44	4.00 ± 0.22	$\mathbf{1.00 \pm 0.0}$
RF	G-mean	4.86 ± 0.14	3.00 ± 0.31	2.57 ± 0.3	3.57 ± 0.37	$\mathbf{1.00 \pm 0.0}$

The mean cross-validation scores are shown in Table 5. As discussed previously in this section,~~Considering~~ the disparity of performance levels~~scores~~ across datasets makes the analysis of these scores less informative.~~, the results presented in this table may not be as informative as the scores for each dataset, presented in Table 6. K-SMOTE's performance is the highest in most classifier/metric combinations and datasets.~~

Table 5: Mean cross-validation scores of oversamplers.

Classifier	Metric	NONE	ROS	SMOTE	B-SMOTE	K-SMOTE
LR	Accuracy	0.906 ± 0.039	0.904 ± 0.04	0.904 ± 0.04	0.901 ± 0.04	$\mathbf{0.909 \pm 0.038}$
LR	F-score	0.891 ± 0.041	0.893 ± 0.042	0.893 ± 0.042	0.890 ± 0.042	$\mathbf{0.898 \pm 0.04}$
LR	G-mean	0.936 ± 0.025	0.940 ± 0.025	0.940 ± 0.025	0.937 ± 0.025	$\mathbf{0.943 \pm 0.024}$
KNN	Accuracy	0.879 ± 0.043	0.865 ± 0.048	0.867 ± 0.05	0.862 ± 0.054	$\mathbf{0.881 \pm 0.045}$
KNN	F-score	0.859 ± 0.05	0.853 ± 0.049	0.861 ± 0.047	0.851 ± 0.053	$\mathbf{0.866 \pm 0.048}$
KNN	G-mean	0.919 ± 0.03	0.920 ± 0.029	0.926 ± 0.027	0.918 ± 0.03	$\mathbf{0.927 \pm 0.027}$
RF	Accuracy	0.898 ± 0.032	0.901 ± 0.031	0.900 ± 0.031	0.898 ± 0.032	$\mathbf{0.905 \pm 0.031}$
RF	F-score	0.879 ± 0.041	0.885 ± 0.037	0.887 ± 0.036	0.883 ± 0.037	$\mathbf{0.891 \pm 0.036}$

Table 5: Mean cross-validation scores of oversamplers.

Classifier	Metric	NONE	ROS	SMOTE	B-SMOTE	K-SMOTE
RF	G-mean	0.930 ± 0.024	0.935 ± 0.022	0.937 ± 0.021	0.935 ± 0.021	**0.939** ± **0.02**

The mean cross-validation scores for each dataset are presented in Table 6. This table allows the direct comparison of the performance metrics being analysed.

~~The performance of both oversamplers and classifiers is generally dependent on the dataset being used. Although both absolute and relative scores between the different oversamplers are dependent on the choice of metric and classifier, K-SMOTE's relative performance is consistent across datasets and generally outperforms the remaining oversampling methods in 56 of the 63 tests.~~

Table 6: Mean cross-validation scores ~~of oversamplers~~ for each dataset. Legend: IP - Indian Pines, KSC - Kennedy Space Center, PC - Pavia Center, PU - Pavia University, SA - Salinas A.

Dataset	Classifier	Metric	NONE	ROS	SMOTE	B-SMOTE	K-SMOTE
Botswana	LR	Accuracy	0.920	0.917	0.920	0.921	**0.927**
Botswana	LR	F-score	0.913	0.909	0.913	0.914	**0.921**
Botswana	LR	G-mean	0.952	0.950	0.952	0.952	**0.956**
Botswana	KNN	Accuracy	0.875	0.862	0.881	0.869	**0.889**
Botswana	KNN	F-score	0.859	0.850	0.873	0.859	**0.879**
Botswana	KNN	G-mean	0.924	0.918	0.930	0.923	**0.933**
Botswana	RF	Accuracy	0.873	0.884	0.877	0.877	**0.890**
Botswana	RF	F-score	0.865	0.877	0.872	0.870	**0.883**
Botswana	RF	G-mean	0.925	0.933	0.929	0.928	**0.936**
IP	LR	Accuracy	0.687	0.681	0.680	0.678	**0.692**
IP	LR	F-score	0.662	0.663	0.659	0.659	**0.674**
IP	LR	G-mean	0.798	0.801	0.798	0.797	**0.807**
IP	KNN	Accuracy	**0.644**	0.602	0.589	0.557	0.632
IP	KNN	F-score	0.593	0.591	0.603	0.560	**0.604**
IP	KNN	G-mean	0.757	0.764	**0.782**	0.751	0.781
IP	RF	Accuracy	0.742	0.747	0.747	0.740	**0.752**
IP	RF	F-score	0.673	0.704	0.713	0.701	**0.714**
IP	RF	G-mean	0.806	0.826	0.835	0.831	**0.838**
KSC	LR	Accuracy	0.904	0.905	0.905	0.899	**0.909**
KSC	LR	F-score	0.868	0.873	0.874	0.862	**0.877**
KSC	LR	G-mean	0.928	0.932	0.932	0.924	**0.934**
KSC	KNN	Accuracy	0.855	0.859	0.862	0.857	**0.865**
KSC	KNN	F-score	0.808	0.819	**0.827**	0.810	0.826
KSC	KNN	G-mean	0.893	0.901	**0.906**	0.895	0.905
KSC	RF	Accuracy	0.860	0.859	0.863	0.859	**0.868**
KSC	RF	F-score	0.817	0.815	0.826	0.816	**0.832**
KSC	RF	G-mean	0.898	0.899	0.905	0.898	**0.907**
PC	LR	Accuracy	0.954	0.955	0.955	0.950	**0.956**
PC	LR	F-score	0.944	0.947	0.947	0.941	**0.948**
PC	LR	G-mean	0.968	0.972	0.972	0.966	**0.973**
PC	KNN	Accuracy	**0.926**	0.920	0.923	0.924	**0.926**
PC	KNN	F-score	**0.915**	0.909	0.913	0.913	**0.915**
PC	KNN	G-mean	0.953	0.955	**0.957**	0.954	**0.957**
PC	RF	Accuracy	0.938	0.941	0.940	0.938	**0.942**
PC	RF	F-score	0.928	0.932	0.931	0.928	**0.933**
PC	RF	G-mean	0.959	0.964	**0.965**	0.961	**0.965**
PU	LR	Accuracy	**0.905**	0.897	0.897	0.891	0.904

Table 6: Mean cross-validation scores ~~of oversamplers~~ for each dataset. Legend: IP - Indian Pines, KSC - Kennedy Space Center, PC - Pavia Center, PU - Pavia University, SA - Salinas A.

Dataset	Classifier	Metric	NONE	ROS	SMOTE	B-SMOTE	K-SMOTE
PU	LR	F-score	0.890	0.894	0.894	0.888	**0.898**
PU	LR	G-mean	0.932	0.947	0.947	0.942	**0.949**
PU	KNN	Accuracy	**0.895**	0.867	0.865	0.873	**0.895**
PU	KNN	F-score	**0.891**	0.868	0.868	0.874	**0.891**
PU	KNN	G-mean	0.940	0.935	0.936	0.936	**0.941**
PU	RF	Accuracy	**0.912**	0.908	0.907	0.908	0.911
PU	RF	F-score	**0.909**	0.906	0.906	0.908	**0.909**
PU	RF	G-mean	0.946	0.946	0.948	0.948	**0.949**
Salinas	LR	Accuracy	**0.990**	**0.990**	0.989	**0.990**	**0.990**
Salinas	LR	F-score	0.985	**0.986**	0.985	0.985	**0.986**
Salinas	LR	G-mean	0.992	**0.993**	0.992	0.992	**0.993**
Salinas	KNN	Accuracy	**0.970**	0.967	0.969	0.967	**0.970**
Salinas	KNN	F-score	0.959	0.957	**0.960**	0.957	**0.960**
Salinas	KNN	G-mean	0.977	0.978	**0.981**	0.976	**0.981**
Salinas	RF	Accuracy	0.984	0.983	0.983	0.983	**0.985**
Salinas	RF	F-score	0.979	0.979	0.977	0.978	**0.980**
Salinas	RF	G-mean	0.989	0.989	0.989	0.989	**0.990**
SA	LR	Accuracy	0.979	0.981	0.983	0.979	**0.984**
SA	LR	F-score	0.976	0.979	**0.982**	0.977	**0.982**
SA	LR	G-mean	0.985	0.988	**0.990**	0.987	0.989
SA	KNN	Accuracy	0.987	0.979	0.982	0.983	**0.988**
SA	KNN	F-score	0.986	0.979	0.981	0.982	**0.987**
SA	KNN	G-mean	0.992	0.989	0.990	0.991	**0.993**
SA	RF	Accuracy	0.980	0.983	0.984	0.979	**0.985**
SA	RF	F-score	0.979	0.982	0.983	0.978	**0.984**
SA	RF	G-mean	0.987	0.988	0.989	0.986	**0.990**

4.2. Statistical Analysis

The experiment's multi-dataset context was used to perform a Friedman test [60]. Table 7 shows the results obtained in the Friedman test performed, where the null hypothesis is rejected in all cases.

Table 7: Results for Friedman test. Statistical significance is tested at a level of $\alpha = 0.05$. The null hypothesis is that there is no difference in the classification outcome across oversamplers.

Classifier	Metric	p-value	Significance
LR	Accuracy	9.8e-03	True
LR	F-score	2.3e-03	True
LR	G-mean	9.8e-04	True
KNN	Accuracy	4.3e-03	True
KNN	F-score	4.3e-03	True
KNN	G-mean	3.0e-03	True
RF	Accuracy	5.5e-03	True
RF	F-score	2.9e-03	True
RF	G-mean	1.8e-04	True

4.3. Discussion

The mean rankings presented in Table 4 show that on average, K-SMOTE produced the best results for every classifier and performance metric used. This is owed to the clustering phase and subsequent selection of data to be considered for oversampling. By successfully clustering and selecting the relevant areas in the data space to oversample, the SMOTE procedure was only used on minority classes within regions that represent well their spectral signature. This implies that this method, although successful in label noise free datasets such as the ones used in this experiment, is particularly effective for training classifiers whenever the ground truth data is both imbalanced and outdated.

As previously discussed, the direct comparison of performance metrics averaged over various datasets is not recommended due to the varying levels of performance of classifiers across datasets [59]. Nonetheless, these results are shown in Table 5 to provide a fuller picture of the results obtained in the experiment. We found that on average K-SMOTE provides increased performance, regardless of the classifier and performance metric used. More importantly, K-SMOTE guaranteed a more consistent performance across datasets and with less variability, which can be attested in Tables 4, 5 and 6.

As discussed in Subsection 3.3, Evaluation Metrics, our results are consistent with the findings in [53,54]. Particularly, we consider the results obtained in our experiment using Overall Accuracy to be less informative than the results obtained with the remaining performance metrics, since this metric is affected by imbalanced class distributions. The class imbalance bias in this metric can be observed in our experiment in Table 4 with the classifiers LR and KNN, where the control method (NONE) is only outperformed by K-SMOTE. This effect is observed with more detail in Table 5, where the benchmark oversamplers are outperformed by the control method in 16 out of 63 tests (approximately 25%). Out of these, 11 refer to tests using overall accuracy, while 9 of these 11 are found among the four datasets with highest IR, showing the overall accuracy's class imbalance bias discussed in [53,54]. Regardless, K-SMOTE is only outperformed by the control method in 3 out of 63 tests (approximately 4.7%), where all of those tests use overall accuracy. This represents an improvement over the benchmark oversamplers, showing that generally K-SMOTE is the best choice even when overall accuracy is used as the main performance metric.

In addition to its superior performance, in the majority of the cases K-SMOTE's mean ranking has a lower standard deviation than any of the remaining methods, and particularly when opposed to SMOTE (the best performing benchmark method). This shows that the non-random selection of data spaces to oversample contributes to the quality of the data generated even on label noise-free data, making it a more informed data generation method in the context of LULC, as label noise and rare classes are particularly frequent in this domain.

The performance of both oversamplers and classifiers is generally dependent on the dataset being used. Although both absolute and relative scores between the different oversamplers are dependent on the choice of metric and classifier, K-SMOTE's relative performance is consistent across datasets and generally outperforms the remaining oversampling methods in 56 of the 63 tests (approximately 89%). The mean cross-validation results found in Table 6 show that performance-wise, K-SMOTE is always better than or as good as SMOTE, with the exception of 4 situations (representing 6% of the tests done), in which cases the percentage point difference is neglectable (≤ 0.1 percentage points). This shows that, in most cases, the usage of a cluster based procedure before employing the SMOTE algorithm (in this case, K-SMOTE) improves the quality of LULC classification when compared to using SMOTE in its original format, which remains the most popular oversampler among the remote sensing community.

5. Conclusion

This research paper was motivated by the challenges faced when classifying rare classes for LULC mapping. Cluster-based oversampling is especially useful in this context because the spectral signature of a given class often varies, depending on its geographical distribution and the time period within which the image was acquired. This induces the representation of minority classes as small clusters in the input space. As a result, training a classifier capable of identifying LULC minority classes in the hyper/multi-spectral scene over different areas or periods becomes particularly challenging. The clustering procedure, performed before the data generation phase, allows for a more accurate generation of minority samples, as it identifies these minority clusters.

A number of existing methods to address the imbalanced learning problem were identified and their limitations discussed. Typically, algorithm-based approaches and cost-sensitive solutions are not only difficult to implement, but they are also context dependent. In this paper we focused on oversampling methods due to their widespread usage, easy implementation and flexibility. Specifically, this paper demonstrated the efficacy of a recent oversampler, K-Means SMOTE, applied in a multi-class context for Land Cover Classification tasks. This was done with sampled data from seven well known and naturally imbalanced benchmark datasets: Indian Pines, Pavia Center~~re~~, Pavia University, Salinas, Salinas A, Botswana and Kennedy Space Center. For each combination of dataset, oversampler and classifier, the results of every classification task was averaged across a 5 fold stratification strategy with 3 different initialization seeds, resulting in a mean validation score of 15 classification tasks. The mean validation score of each combination was then used to perform the analyses presented in this report.

In 56 out of 63 classification tasks (approximately 89%), K-SMOTE led to better results than ROS, SMOTE, B-SMOTE and no oversampling. More importantly, we found that K-Means SMOTE is always better or equal than the second best oversampling method. K-SMOTE's performance was independent from both the classifier and performance metric under analysis. In general, K-SMOTE shows a higher performance among the non tree-based classifiers employed, when compared to the remaining oversamplers. Although these findings are case dependent, they are consistent with the results presented in [21]. The proposed method also had the most consistent results across datasets, since it produced the lowest standard deviations across datasets in 7 out of 9 cases for both analyses, either based on ranking or mean cross-validation scores.

The proposed algorithm is a generalization of the original SMOTE algorithm. In fact, the SMOTE algorithm represents a corner case of K-SMOTE i.e. when the number of clusters equals to 1. Its data selection phase differs from the one used in SMOTE and Borderline SMOTE, providing artificially augmented datasets with less noisy data than the commonly used methods. This allows the training of classifiers with better defined decision boundaries, especially in the most important regions of the data space (the ones populated by a higher percentage of minority class instances).

As stated previously, the usage of this oversampler is technically simple. It can be applied to any classification problem relying on an imbalanced dataset, alongside any classifier. K-SMOTE is available as an open source implementation for the Python programming language (see Subsection 3.5). Consequently, it can be a useful tool for both remote sensing researchers and practitioners.

Author Contributions: Conceptualization, F.B.; Methodology, J.F. and G.D.; Software, J.F. and G.D.; Validation, F.B., G.D.; Formal Analysis, J.F; Writing - Original Draft Preparation, J.F.; Writing - Review & Editing, F.B., G.D., J.F.; Supervision, F.B.; Funding Acquisition, F.B.

Funding: This research was funded by "Fundação para a Ciência e Tecnologia" (Portugal), grants' number PCIF/SSI/0102/2017 - foRESTER and DSAIPA/AI/0100/2018 - IPSTERS.

Data Availability Statement: The data reported in this study is publicly available. It can be retrieved and preprocessed using the Python source code provided at https://github.com/AlgoWit/publications. Alternatively the original data is available at http://www.ehu.eus/ccwintco/index.php?title=Hyperspectral_Remote_Sensing_Scenes.

Conflicts of Interest: The authors declare no conflict of interest. The funders had no role in the design of the study; in the collection, analyses, or interpretation of data; in the writing of the manuscript, or in the decision to publish the results.

References

1. Drusch, M.; Del Bello, U.; Carlier, S.; Colin, O.; Fernandez, V.; Gascon, F.; Hoersch, B.; Isola, C.; Laberinti, P.; Martimort, P.; Meygret, A.; Spoto, F.; Sy, O.; Marchese, F.; Bargellini, P. Sentinel-2: ESA's Optical High-Resolution Mission for GMES Operational Services. *Remote Sensing of Environment* **2012**, *120*, 25–36. doi:10.1016/j.rse.2011.11.026.
2. Fritz, S.; See, L.; Perger, C.; McCallum, I.; Schill, C.; Schepaschenko, D.; Duerauer, M.; Karner, M.; Dresel, C.; Laso-Bayas, J.C.; Lesiv, M.; Moorthy, I.; Salk, C.F.; Danylo, O.; Sturn, T.; Albrecht, F.; You, L.; Kraxner, F.; Obersteiner, M. A global dataset of crowdsourced land cover and land use reference data. *Scientific Data* **2017**, *4*, 1–8. doi:10.1038/sdata.2017.75.
3. Khatami, R.; Mountrakis, G.; Stehman, S.V. A meta-analysis of remote sensing research on supervised pixel-based land-cover image classification processes: General guidelines for practitioners and future research. *Remote Sensing of Environment* **2016**, *177*, 89–100. doi:10.1016/J.RSE.2016.02.028.
4. Wulder, M.A.; Coops, N.C.; Roy, D.P.; White, J.C.; Hermosilla, T. Land cover 2.0. *International Journal of Remote Sensing* **2018**, *39*, 4254–4284. doi:10.1080/01431161.2018.1452075.
5. Gavade, A.B.; Rajpurohit, V.S. Systematic analysis of satellite image-based land cover classification techniques: literature review and challenges. *International Journal of Computers and Applications* **2019**, pp. 1–10. doi:10.1080/1206212x.2019.1573946.
6. Kaur, H.; Pannu, H.S.; Malhi, A.K. A Systematic Review on Imbalanced Data Challenges in Machine Learning: Applications and Solutions. *ACM Comput. Surv.* **2019**, *52*. doi:10.1145/3343440.
7. Stromann, O.; Nascetti, A.; Yousif, O.; Ban, Y. Dimensionality Reduction and Feature Selection for Object-Based Land Cover Classification based on Sentinel-1 and Sentinel-2 Time Series Using Google Earth Engine. *Remote Sensing* **2020**, *12*, 76. doi:10.3390/RS12010076.
8. Alonso-Sarria, F.; Valdivieso-Ros, C.; Gomariz-Castillo, F. Isolation Forests to Evaluate Class Separability and the Representativeness of Training and Validation Areas in Land Cover Classification. *Remote Sensing* **2019**, *11*, 3000. doi:10.3390/rs11243000.
9. Pelletier, C.; Valero, S.; Inglada, J.; Champion, N.; Marais Sicre, C.; Dedieu, G. Effect of Training Class Label Noise on Classification Performances for Land Cover Mapping with Satellite Image Time Series. *Remote Sensing* **2017**, *9*, 173. doi:10.3390/rs9020173.
10. Wang, R.; Zhang, J.; Chen, J.W.; Jiao, L.; Wang, M. Imbalanced Learning-based Automatic SAR Images Change Detection by Morphologically Supervised PCA-Net. *IEEE Geoscience and Remote Sensing Letters* **2019**, [1906.07923].
11. Feng, W.; Huang, W.; Bao, W. Imbalanced Hyperspectral Image Classification With an Adaptive Ensemble Method Based on SMOTE and Rotation Forest With Differentiated Sampling Rates. *IEEE Geoscience and Remote Sensing Letters* **2019**, pp. 1–5. doi:10.1109/LGRS.2019.2913387.
12. Chawla, N.V.; Japkowicz, N.; Kotcz, A. Editorial: special issue on learning from imbalanced data sets. *ACM SIGKDD Explorations Newsletter* **2004**, *6*, 1. doi:10.1145/1007730.1007733.
13. Abdi, L.; Hashemi, S. To Combat Multi-Class Imbalanced Problems by Means of Over-Sampling Techniques. *IEEE Transactions on Knowledge and Data Engineering* **2016**, *28*, 238–251. doi:10.1109/TKDE.2015.2458858.
14. Maxwell, A.E.; Warner, T.A.; Fang, F. Implementation of machine-learning classification in remote sensing: An applied review. *International Journal of Remote Sensing* **2018**, *39*, 2784–2817. doi:10.1080/01431161.2018.1433343.
15. Fernández, A.; López, V.; Galar, M.; del Jesus, M.J.; Herrera, F. Analysing the classification of imbalanced data-sets with multiple classes: Binarization techniques and ad-hoc approaches. *Knowledge-Based Systems* **2013**, *42*, 97–110. doi:10.1016/J.KNOSYS.2013.01.018.
16. Luengo, J.; García-Gil, D.; Ramírez-Gallego, S.; García, S.; Herrera, F. Imbalanced Data Preprocessing for Big Data. In *Big Data Preprocessing*; Springer International Publishing: Cham, 2020; pp. 147–160. doi:10.1007/978-3-030-39105-8_8.
17. García, S.; Ramírez-Gallego, S.; Luengo, J.; Benítez, J.M.; Herrera, F. Big data preprocessing: methods and prospects. *Big Data Analytics* **2016**, *1*, 9. doi:10.1186/s41044-016-0014-0.
18. Haixiang, G.; Yijing, L.; Shang, J.; Mingyun, G.; Yuanyue, H.; Bing, G. Learning from Class-Imbalanced Data. *Expert Syst. Appl.* **2017**, *73*, 220–239. doi:10.1016/j.eswa.2016.12.035.
19. Douzas, G.; Bacao, F.; Fonseca, J.; Khudinyan, M. Imbalanced learning in land cover classification: Improving minority classes' prediction accuracy using the geometric SMOTE algorithm. *Remote Sensing* **2019**, *11*, 3040. doi:10.3390/rs11243040.
20. Chawla, N.V.; Bowyer, K.W.; Hall, L.O.; Kegelmeyer, W.P. SMOTE: Synthetic Minority Over-sampling Technique. *Journal of Artificial Intelligence Research* **2002**, *16*, 321–357. doi:10.1613/jair.953.
21. Douzas, G.; Bacao, F.; Last, F. Improving imbalanced learning through a heuristic oversampling method based on k-means and SMOTE. *Information Sciences* **2018**, *465*, 1–20. doi:10.1016/j.ins.2018.06.056.
22. Japkowicz, N. Concept-learning in the presence of between-class and within-class imbalances. Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics). Springer Verlag, 2001, Vol. 2056, pp. 67–77. doi:10.1007/3-540-45153-6_7.
23. Jo, T.; Japkowicz, N. Class imbalances versus small disjuncts. *ACM SIGKDD Explorations Newsletter* **2004**, *6*, 40–49. doi:10.1145/1007730.1007737.

24. Han, H.; Wang, W.Y.; Mao, B.H. Borderline-SMOTE: A New Over-Sampling Method in Imbalanced Data Sets Learning. International Conference on Intelligent Computing. Springer, Berlin, Heidelberg, 2005, pp. 878–887. doi:10.1007/11538059_91.
25. Blagus, R.; Lusa, L. Class prediction for high-dimensional class-imbalanced data. *BMC bioinformatics* **2010**, *11*, 523. doi:10.1186/1471-2105-11-523.
26. Mellor, A.; Boukir, S.; Haywood, A.; Jones, S. Exploring issues of training data imbalance and mislabelling on random forest performance for large area land cover classification using the ensemble margin. *ISPRS Journal of Photogrammetry and Remote Sensing* **2015**, *105*, 155–168. doi:10.1016/J.ISPRSJPRS.2015.03.014.
27. Shao, Y.H.; Chen, W.J.; Zhang, J.J.; Wang, Z.; Deng, N.Y. An efficient weighted Lagrangian twin support vector machine for imbalanced data classification. *Pattern Recognition* **2014**, *47*, 3158–3167. doi:10.1016/j.patcog.2014.03.008.
28. Lee, T.; Lee, K.B.; Kim, C.O. Performance of Machine Learning Algorithms for Class-Imbalanced Process Fault Detection Problems. *IEEE Transactions on Semiconductor Manufacturing* **2016**, *29*, 436–445. doi:10.1109/TSM.2016.2602226.
29. Huang, C.; Li, Y.; Loy, C.C.; Tang, X. Learning deep representation for imbalanced classification. Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition, 2016, Vol. 2016-Decem, pp. 5375–5384. doi:10.1109/CVPR.2016.580.
30. Cui, Y.; Jia, M.; Lin, T.Y.; Song, Y.; Belongie, S. Class-balanced loss based on effective number of samples. Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition, 2019, Vol. 2019-June, pp. 9260–9269, [1901.05555]. doi:10.1109/CVPR.2019.00949.
31. Dong, Q.; Gong, S.; Zhu, X. Class Rectification Hard Mining for Imbalanced Deep Learning. Proceedings of the IEEE International Conference on Computer Vision, 2017, Vol. 2017-Octob, pp. 1869–1878, [1712.03162]. doi:10.1109/ICCV.2017.205.
32. Sharififar, A.; Sarmadian, F.; Minasny, B. Mapping imbalanced soil classes using Markov chain random fields models treated with data resampling technique. *Computers and Electronics in Agriculture* **2019**, *159*, 110–118. doi:10.1016/j.compag.2019.03.006.
33. Hounkpatin, K.O.; Schmidt, K.; Stumpf, F.; Forkuor, G.; Behrens, T.; Scholten, T.; Amelung, W.; Welp, G. Predicting reference soil groups using legacy data: A data pruning and Random Forest approach for tropical environment (Dano catchment, Burkina Faso). *Scientific Reports* **2018**, *8*, 1–16. doi:10.1038/s41598-018-28244-w.
34. Krawczyk, B. Learning from imbalanced data: open challenges and future directions. *Progress in Artificial Intelligence* **2016**, *5*, 221–232. doi:10.1007/s13748-016-0094-0.
35. Ferreira, M.P.; Wagner, F.H.; Aragão, L.E.; Shimabukuro, Y.E.; de Souza Filho, C.R. Tree species classification in tropical forests using visible to shortwave infrared WorldView-3 images and texture analysis. *ISPRS Journal of Photogrammetry and Remote Sensing* **2019**, *149*, 119–131. doi:10.1016/j.isprsjprs.2019.01.019.
36. Feng, W.; Huang, W.; Ye, H.; Zhao, L. Synthetic minority over-sampling technique based rotation forest for the classification of unbalanced hyperspectral data. International Geoscience and Remote Sensing Symposium (IGARSS). Institute of Electrical and Electronics Engineers Inc., 2018, Vol. 2018-July, pp. 2651–2654. doi:10.1109/IGARSS.2018.8518242.
37. Jozdani, S.E.; Johnson, B.A.; Chen, D. Comparing Deep Neural Networks, Ensemble Classifiers, and Support Vector Machine Algorithms for Object-Based Urban Land Use/Land Cover Classification. *Remote Sensing* **2019**, *11*, 1713. doi:10.3390/rs11141713.
38. Bogner, C.; Seo, B.; Rohner, D.; Reineking, B. Classification of rare land cover types: Distinguishing annual and perennial crops in an agricultural catchment in South Korea. *PLoS ONE* **2018**, *13*. doi:10.1371/journal.pone.0190476.
39. Zhu, M.; Wu, B.; He, Y.N.; He, Y.Q. LAND COVER CLASSIFICATION USING HIGH RESOLUTION SATELLITE IMAGE BASED ON DEEP LEARNING. *ISPRS - International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences* **2020**, *XLII-3/W10*, 685–690. doi:10.5194/isprs-archives-xlii-3-w10-685-2020.
40. Cenggoro, T.W.; Isa, S.M.; Kusuma, G.P.; Pardamean, B. Classification of imbalanced land-use/land-cover data using variational semi-supervised learning. Proceedings - 2017 International Conference on Innovative and Creative Information Technology: Computational Intelligence and IoT, ICITech 2017. IEEE, 2018, Vol. 2018-Janua, pp. 1–6. doi:10.1109/INNOCIT.2017.8319149.
41. Douzas, G.; Bacao, F. Geometric SMOTE a geometrically enhanced drop-in replacement for SMOTE. *Information Sciences* **2019**, *501*, 118–135. doi:10.1016/J.INS.2019.06.007.
42. Ma, L.; Fan, S. CURE-SMOTE algorithm and hybrid algorithm for feature selection and parameter optimization based on random forests. *BMC Bioinformatics* **2017**, *18*, 169. doi:10.1186/s12859-017-1578-z.
43. Douzas, G.; Bacao, F. Self-Organizing Map Oversampling (SOMO) for imbalanced data set learning. *Expert Systems with Applications* **2017**, *82*, 40–52. doi:10.1016/j.eswa.2017.03.073.
44. Haibo He.; Yang Bai.; Garcia, E.A.; Shutao Li. ADASYN: Adaptive synthetic sampling approach for imbalanced learning. 2008 IEEE International Joint Conference on Neural Networks (IEEE World Congress on Computational Intelligence). IEEE, 2008, pp. 1322–1328. doi:10.1109/IJCNN.2008.4633969.
45. Holte, R.C.; Acker, L.; Porter, B.W.; others. Concept Learning and the Problem of Small Disjuncts. IJCAI. Citeseer, 1989, Vol. 89, pp. 813–818.
46. Santos, M.S.; Abreu, P.H.; García-Laencina, P.J.; Simão, A.; Carvalho, A. A new cluster-based oversampling method for improving survival prediction of hepatocellular carcinoma patients. *Journal of Biomedical Informatics* **2015**, *58*, 49–59. doi:10.1016/j.jbi.2015.09.012.
47. Baumgardner, M.F.; Biehl, L.L.; Landgrebe, D.A. 220 Band AVIRIS Hyperspectral Image Data Set: June 12, 1992 Indian Pine Test Site 3. *Purdue University Research Repository* **2015**. doi:10.4231/R7RX991C.

48. Nelder, J.A.; Wedderburn, R.W. Generalized linear models. *Journal of the Royal Statistical Society: Series A (General)* **1972**, *135*, 370–384.
49. Cover, T.; Hart, P. Nearest neighbor pattern classification. *IEEE Transactions on Information Theory* **1967**, *13*, 21–27. doi:10.1109/TIT.1967.1053964.
50. Salzberg, S.L. C4.5: Programs for Machine Learning by J. Ross Quinlan. Morgan Kaufmann Publishers, Inc., 1993. *Machine Learning* **1994**, *16*, 235–240. doi:10.1007/BF00993309.
51. Friedman, J.H. Greedy function approximation: A gradient boosting machine. *Annals of Statistics* **2001**, *29*, 1189–1232, [arXiv:1011.1669v3]. doi:DOI 10.1214/aos/1013203451.
52. Liaw, A.; Wiener, M.; others. Classification and regression by randomForest. *R news* **2002**, *2*, 18–22.
53. Olofsson, P.; Foody, G.M.; Stehman, S.V.; Woodcock, C.E. Making better use of accuracy data in land change studies: Estimating accuracy and area and quantifying uncertainty using stratified estimation. *Remote Sensing of Environment* **2013**, *129*, 122–131. doi:10.1016/j.rse.2012.10.031.
54. Pontius, R.G.; Millones, M. Death to Kappa: Birth of quantity disagreement and allocation disagreement for accuracy assessment, 2011. doi:10.1080/01431161.2011.552923.
55. Jeni, L.A.; Cohn, J.F.; De La Torre, F. Facing imbalanced data - Recommendations for the use of performance metrics. Proceedings - 2013 Humaine Association Conference on Affective Computing and Intelligent Interaction, ACII 2013, 2013, pp. 245–251. doi:10.1109/ACII.2013.47.
56. He, H.; Garcia, E.A. Learning from Imbalanced Data. *IEEE Transactions on Knowledge and Data Engineering* **2009**, *21*, 1263–1284, [arXiv:1011.1669v3]. doi:10.1109/TKDE.2008.239.
57. Pedregosa, F.; Varoquaux, G.; Gramfort, A.; Michel, V.; Thirion, B.; Grisel, O.; Blondel, M.; Prettenhofer, P.; Weiss, R.; Dubourg, V.; Vanderplas, J.; Passos, A.; Cournapeau, D.; Brucher, M.; Perrot, M.; Duchesnay, É. Scikit-learn: Machine Learning in Python. *Journal of Machine Learning Research* **2011**, *12*, 2825–2830.
58. Lemaître, G.; Nogueira, F.; Aridas, C.K. Imbalanced-learn: A Python Toolbox to Tackle the Curse of Imbalanced Datasets in Machine Learning. *Journal of Machine Learning Research* **2017**, *18*, 1–5.
59. Demšar, J. Statistical comparisons of classifiers over multiple data sets. *Journal of Machine learning research* **2006**, *7*, 1–30.
60. Friedman, M. The use of ranks to avoid the assumption of normality implicit in the analysis of variance. *Journal of the american statistical association* **1937**, *32*, 675–701.