# Improving Imbalanced Learning in Land Cover Classification

## A Heuristic Oversampling Method Based on K-Means and SMOTE

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

[1]NOVA Information Management School, Universidade Nova de Lisboa

*Corresponding Author

Postal Address: NOVA Information Management School, Campus de Campolide, 1070-312 Lisboa, Portugal

Telephone: +351 21 382 8610

Land cover maps are an important resource to make informed policy, development, planning and resource management decisions. Despite their importance, technical skills are still a primary challenge for the development of accurate, timely and automated Land Use/Land Cover maps. Specifically, remotely sensed data is often imbalanced, where a few majority classes dominate over rare classes. This reflects into an asymmetric class distribution, which negatively impacts the performance of classifiers and ultimately adds a new source of inaccuracy to the production of these maps. In this paper, we address the imbalanced learning problem by proposing K-Means SMOTE, a recent oversampling method, as a tool for addressing data imbalance in remote sensing. K-Means SMOTE is an oversampling algorithm that attempts to improve the quality of newly created artificial data by avoiding the generation of noisy data and effectively overcome data imbalance. The performance of K-Means SMOTE is compared to other popular oversampling methods using seven well known 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.

## 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 [Drusch et al., 2012]. This information is often materialized through Land Use/Land Cover (LULC) maps, which constitute an essential asset for various purposes, such as land cover change detection, urban planning, environmental monitoring and natural hazard assessment [Khatami et al., 2016]. However, the production of accurate, updated LULC maps still pose a challenge within the remote sensing community [Wulder et al., 2018]. They can have either one of two sources: Photo-interpreted by the human eye, or Automatic mapping using remotely sensed data and a classification algorithm.

Although photo-interpreted LULC maps rely on human interaction and can be more reliable, they are not without its drawbacks: they are not frequently updated, their production is time and resource consuming, not suitable for operational mapping over large areas and are prone to overlook rare or small-area classes, due to factors such as the minimum mapping unit being used. Concurrently, machine-learning (ML) approaches face different challenges:

1. Mislabelled LULC patches. As mentioned, the usage 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 power of a classifier [Pelletier et al., 2017].

2. High-dimensional datasets. Multi-spectral TS composites are high-dimensional, which increases the complexity of the problem and creates a strain on computational power [Stromann et al., 2020].

3. Class separability. 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 [Alonso-Sarria et al., 2019].

4. Existence of rare land cover classes. Due to the varying levels of area coverage for each class, using a purely random sampling strategy will amount to a dataset with a roughly proportional class distribution as the one on the 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 [Wang et al., 2019, Feng et al., 2019].

The latter challenge is known as the imbalanced learning problem [Chawla et al., 2004]. It is defined as a skewed distribution of observations found in a dataset among classes in both binary and multi-class problems [Abdi and Hashemi, 2016]. This asymmetry in class distribution negatively impacts the performance of classifiers, especially in multi-class problems. During the learning phase, classifiers are optimized to best fit an objective function, being the most common metric the overall accuracy [Maxwell et al., 2018]. This means observations belonging to rare/minority classes contribute less towards the predictive power of the corresponding classes, translating into a bias towards majority classes, as depicted in figure 1a. As an example, a trivial classifier can achieve 99% overall accuracy on a binary dataset where 1% of the observations belong to the minority class if it classifies all observations as belonging to the majority class.

Typical ML algorithms are designed to perform well on relatively balanced datasets. Although, defining a decision boundary on imbalanced datasets is a difficult task since each class' weight in the learning phase is typically as high as its relative number of observations within the training dataset.

There are three different types of approaches to deal with the class imbalance problem [Fernández et al., 2013, Kaur et al., 2019]:

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 (see Figure 1). This is considered an external approach, where the intervention occurs before the learning phase, benefitting from versatility and independency from the classifier used.

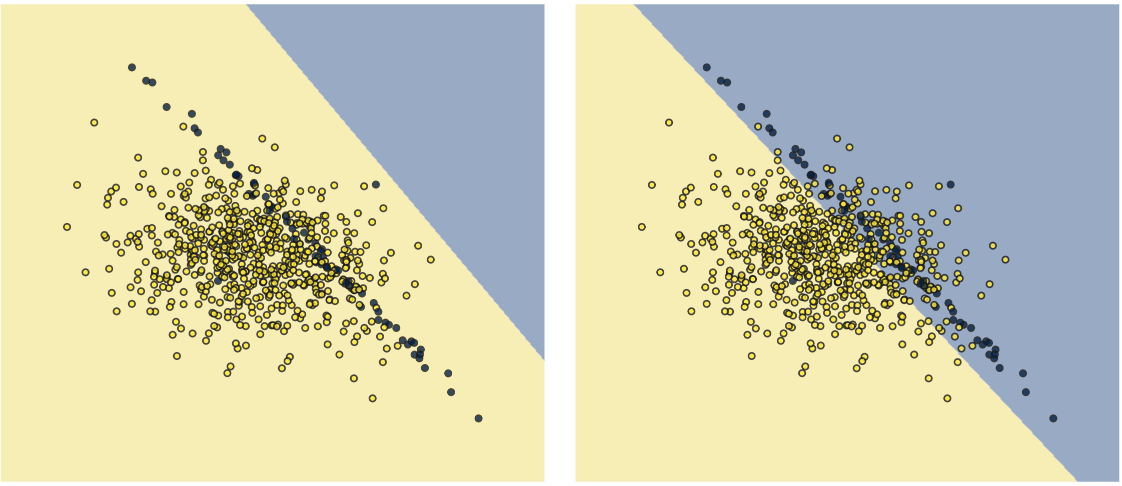
Within resampling approaches there are three subgroups of approaches [Fernández et al., 2013, Kaur et al., 2019, Luengo et al., 2020]:

1. Undersampling methods. They rebalance class distribution by removing instances from the majority classes.

2. Oversampling methods. Dataset is rebalanced by generating new artificial instances belonging to the minority classes.

3. Hybrid methods. 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 [Fernández et al., 2013, Luengo et al., 2020, García et al., 2016]. 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.



(a) (b)

Figure 1: Example of a linear Support Vector Machine's decision function (a) without resampling and (b) with resampling.

In this paper, we propose the K-means SMOTE (K-SMOTE) [Douzas et al., 2018] oversampler to address the imbalanced learning problem in a multiclass context for LULC classification in various reference remote sensing datasets. K-SMOTE's efficacy 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 (SMOTE) [Chawla et al., 2002] and Borderline-SMOTE (B-SMOTE) [Han et al., 2005]. As a baseline we present classification results without the employment 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

Existing methods that address imbalanced learning act on different stages. They can act in the preprocessing step (Over/Undersampling and hybrid approaches), in the learning process (cost-sensitive solutions) or in the algorithm itself (by adapting existing algorithms and/or ensemble methods) [Kaur et al., 2019]. 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 [Khatami et al., 2016, Gavade and Rajpurohit, 2019] are sensitive to class imbalance [Blagus and Lusa, 2010]. Algorithm-based approaches typically focus on adaptations based on ensemble classification methods [Mellor et al., 2015] or common non-ensemble based classifiers such as Support Vector Machines [Shao et al., 2014]. In [Lee et al., 2016], 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 [Huang et al., 2016, Cui et al., 2019, Dong et al., 2017]. A common cost sensitive solution is found in [Huang et al., 2016]. The authors use the inverse class frequency (i.e., $1/|C_i|$) to give higher weight to minority classes. Cui et al. [Cui et al., 2019] extended this method by adding a hyperparameter $\beta$ 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 [Dong et al., 2017] explores adaptations of Cross-entropy classification loss by adding different formulations of class rectification loss.

Imbalanced Learning is most commonly addressed through data resampling in machine learning in general and remote sensing in particular [Feng et al., 2019]. The generation of artificial instances (i.e., augmenting the dataset) based on rare examples is done independently of any other classification and preprocessing step. Once this step is applied, any standard ML procedure can be applied. This simplicity makes resampling strategies particularly appealing for any user interested in applying several classifiers or maintaining a simple approach. Additionally, any of these methods can be naturally applied to multiclass problems and particularly to LULC classification tasks.

## 2.1 Non-informed resampling methods

There are two main non-informed resampling methods. Random Oversampling (ROS) generates artificial observations through random duplication of rare instances. This method is used in remote sensing [Sharififar et al., 2019, Hounkpatin et al., 2018] for its simplicity, even though its mechanism makes the classifier prone to overfitting [Krawczyk, 2016]. Hounkpatin et al. [Hounkpatin et al., 2018] 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) [Ferreira et al., 2019]. This method, on the other hand, randomly removes observations belonging to common classes. Although it's not as prone to overfitting as ROS, it incurs into information loss by eliminating observations from the majority class [Feng et al., 2019].

Another downfall 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) [Maxwell et al., 2018]. Similarly, RUS is found to generally lead to a lower overall accuracy due to the associated information loss [Maxwell et al., 2018].

## 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 [Feng et al., 2018] using the rotation forest ensemble classifier, authors found that oversampling methods consistently outperform undersampling methods. Due to the scope of this study, heuristic undersampling algorithms will not be analysed.

SMOTE [Chawla et al., 2002] 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 examples along the line segments that join any/all of the $k$ minority class nearest neighbors to the selected sample. Specifically, a single synthetic sample $\overrightarrow{z}$ is generated within the line segment of a randomly selected minority class observation $\overrightarrow{x}$ and one of its $k$ nearest neighbors $\overrightarrow{y}$ such that $\overrightarrow{z} = \alpha\overrightarrow{x} + (1-\alpha)\overrightarrow{y}$, where $\alpha$ is a random floating point between 0 and 1, as shown in Figure 2.

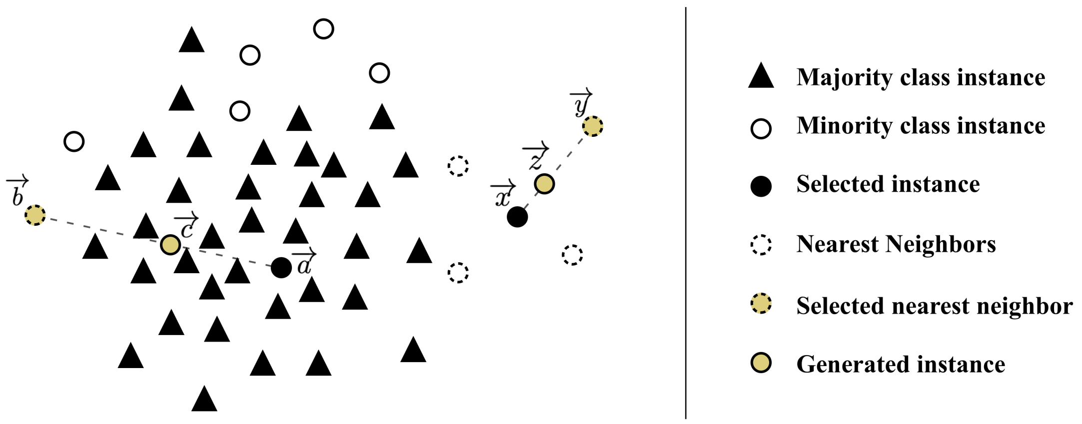



Figure 2: Example of SMOTE's data generation process.

A number of studies implement SMOTE within the LULC classification context and reported improvements on the quality of the trained predictors [Jozdani et al., 2019, Bogner et al., 2018]. Another study proposes an adaptation of SMOTE on an algorithmic level for deep learning applications [Zhu et al., 2020]. 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 [Cenggoro et al., 2018]. It consists of a generative model that allows learning from both labelled and unlabelled instances while using SMOTE to balance the data.

Despite SMOTE's popularity, its drawbacks motivated the development of more sophisticated oversampling algorithms [Douzas and Bacao, 2019]:

1. Generation of noisy instances due to random selection of a minority observation to oversample. The random selection of a minority observation makes SMOTE oversampling prone to the amplification of existing noisy data. In Figure 2 it is possible to observe a minority sample located within a cluster of majority instances. Performing a linear interpolation between the noisy sample $\overrightarrow{a}$ and one of its nearest neighbors $\overrightarrow{b}$ will generate a noisy sample $\overrightarrow{c}$. B-SMOTE [Han et al., 2005] attempts to circumvent the noisy data selection problem by performing a targeted selection of

instances close to the presumed class border, determined by the labels of each sample's $k$ nearest neighbors. Alternatively, a sample will discarded because it was deemed as either noisy, or being far from the class boundary. Another algorithm that addresses this problem is ADASYN [Haibo He et al., 2008]. It calculates a density distribution ratio for each sample based on its $k$-nearest neighbors to determine the number of synthetic observations to generate for each minority class observation using the described SMOTE procedure.

2. Generation of noisy instances due to the selection of the $k$ nearest neighbors. In the event an observation (or a small number thereof) is not noisy but is isolated from the remaining clusters, known as the "small disjuncts problem" [Holte et al., 1989], 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 observations that are close to each other, the generated instance becomes very similar to its parents and increases the risk of overfitting. G-SMOTE [Douzas and Bacao, 2019] attempts to address both the $k$ nearest neighbor selection mechanism problem as well as the generation of nearly duplicated instances problem. It proposes a variation on SMOTE's data generation mechanism by generating data within an oval geometry (instead of a line segment) around the selected observation and the selected nearest neighbor. In its turn, the $k$ nearest neighbors selection can include observations from the remaining classes. To an extent, this algorithm can be considered a generalized version of SMOTE, since under specific hyperparameter definitions it replicates SMOTE's behavior.

4. Generation of noisy instances due to the use of 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. Cluster-based oversampling methods, as well as ADASYN, attempt to address this problem. B-SMOTE [Han et al., 2005] and G-SMOTE also address this problem by allowing the interpolation to be performed with majority class instances.

Although no cluster-based oversampling approach applied within the remote sensing domain was found in the literature, there are numerous methods to consider. Cluster-based oversampling approaches introduce an additional layer to SMOTE's selection mechanism, which is done according to the clustering process. This is done to ensure both between-class data balance, but also ensure that the data distribution within each class is preserved. The self-organizing map oversampling (SOMO) [Douzas and Bacao, 2017] 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. CURE-SMOTE [Ma and Fan, 2017] applies a hierarchical clustering algorithm (CURE) to discard isolated minority instances before applying SMOTE. Although it avoids noise generation problems, it ignores within-class data distribution. Another method [Santos et al., 2015] uses K-means to cluster the entire input space and applies SMOTE to clusters with the fewest observations, regardless of their class label. The label of the generated 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 [Douzas et al., 2018] avoids noisy data generation by modifying the data selection mechanism. It employs $k$-means clustering to identify safe areas using cluster-specific Imbalance Ratio (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 oversamplers focused on data generation, such as G-SMOTE. K-SMOTE can be seen as a generalization of the SMOTE algorithm: when the number of clusters is set to one, it will mimic SMOTE's behavior. Consequently, K-SMOTE is always guaranteed to return results as good as or better than SMOTE.

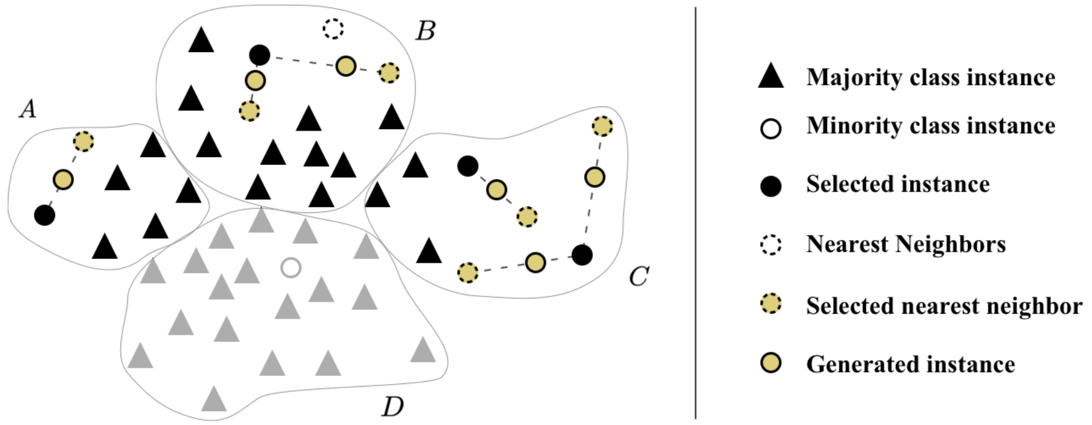



Figure 3: Example of K-SMOTE's data generation process. Clusters $A$, $B$ and $C$ are selected for oversampling, whereas cluster $D$ was rejected due to its high imbalance ratio. The oversampling is done using the SMOTE algorithm and the $k$ nearest neighbors selection only considers observations within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

Although no other study was found to implement cluster-based oversampling, another study [Douzas et al., 2019] 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 main contributions are:

- Testing these oversampling methods in multiple widely used LULC classification datasets. Allows us to check for oversamplers' performance statistical significance across datasets and report K-SMOTE's performance in 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.

# 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. To do so, we employ 7 LULC datasets along with 3 evaluation metrics and 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. A similar data preprocessing procedure was used for each scene: 1) Conversion of each hyperspectral scene to a structured dataset and removal of instances with no associated LULC class, 2) random sampling to maintain similar class proportions on a sample of 10% of each dataset and 3) 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. Table 1 provides a description of the final datasets used for this work.

| Dataset | Features | Instances | Minority instances | Majority 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 |

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

#### Indian Pines

The Indian Pines scene [Baumgardner et al., 2015] 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 4a) and 220 spectral reflectance bands in the wavelength range 400 to 2500 nanometers. 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.

#### Pavia Centre and University

Both Pavia Centre and University scenes were acquired by the ROSIS sensor. These scenes are located in Pavia, northern Italy. Pavia Centre 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 1.3 meters and their ground truths are composed of 9 classes each (see Figures 4b and 4c).

#### 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 4d). Salinas-A, a subscene of Salinas, comprises $86 \times 83$ pixels and contains 6 classes regarding vegetables (see Figure 4e). These scenes have a geometrical resolution of 3.7 meters.

#### 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 4f).

#### 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 4g).

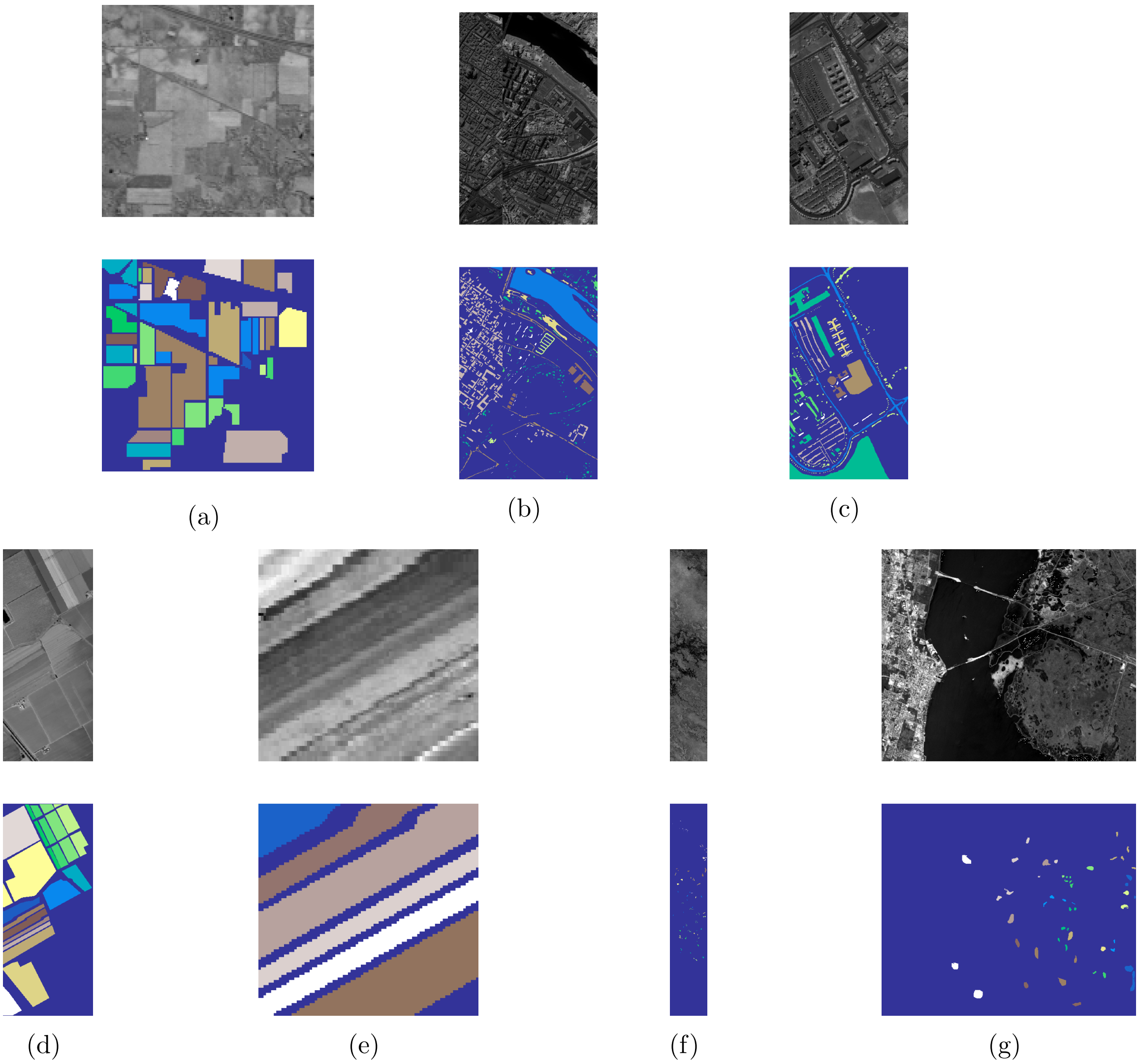



Figure 4: Gray scale visualization of a band (top row) and ground truth (bottom row) of each scene used in this study. (a) Indian Pines, (b) Pavia Centre, (c) Pavia University, (d) Salinas, (e) Salinas A, (f) Botswana, (g) Kennedy Space Center

### 3.2 Evaluation Metrics

Most of the satellite-based LULC classification studies (nearly 80%) employ *Overall Accuracy* (OA) and the *Kappa Coefficient* [Gavade and Rajpurohit, 2019]. Although, some authors argue that both evaluation metrics, even when used simultaneously, are insufficient to fully address the area estimation and uncertainty information needs [Olofsson et al., 2013, Pontius and Millones, 2011]. 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 observations of a given class) and True and False Negatives (*TN* and *FN*, number of correctly/incorrectly classified 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 [Jeni et al., 2013].

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 [Jeni et al., 2013], 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 [He and Garcia, 2009]:

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

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

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

### 3.3 Machine Learning Algorithms

The assess the quality of the K-SMOTE algorithm, five other oversampling algorithms were used for benchmarking. ROS and SMOTE were chosen for their simplicity and popularity. ADASYN and B-SMOTE were chosen for their popularity as outperforming modifications of the SMOTE algorithm. G-SMOTE was chosen for being a state-of-the-art oversampler and was found to outperform all of the other benchmark oversamplers in a past study [Douzas et al., 2019]. In addition to the oversamplers mentioned, we present as the baseline method the classification results without any oversampling method (NONE).

To assess the performance of each oversampler, we use the classifiers Logistic Regression (LR) [McCullagh and Nelder, 1989], K-Nearest Neighbors (KNN) [Cover and Hart, 1967], Decision Tree (DT) [Salzberg, 1994], Gradient Boosting Classifier (GBC) [Friedman, 2001] and Random Forest (RF) [Liaw et al., 2002]. This choice was based on the classifiers' popularity for LULC classification, learning type and training time [Maxwell et al., 2018, Gavade and Rajpurohit, 2019].

### 3.4 Experimental Procedure

The procedure for the experiment reported in this study is similar to the one proposed in [Douzas et al., 2019]. We start by defining a parameter 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 parameters definitions are formed. Finally, for each dataset we employ a $k$-fold cross-validation strategy where $k = 5$ to train each model defined and save the averaged scores of each split.

Each combination of oversampler, classifier and parameters definition is fit 5 times (once for each fold) per dataset. Each time, an oversampler will use the training set (80% 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 with the exact same number of observations for each class. The newly formed training dataset is used to train the classifier and the test set (20% 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 2.

| 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 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 |

Table 2: Hyper-parameters grid. * One cluster is generated in total, a corner case that mimics the behavior of SMOTE

### 3.5 Software Implementation

The experiment was implemented using the Python programming language, using the Scikit-Learn [Pedregosa et al., 2011], Imbalanced-Learn [Lemaître et al., 2017], 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

When evaluating the performance of an algorithm across multiple datasets, it is generally recommended to avoid direct score comparisons and use classification rankings instead [Demšar, 2006]. 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 4.1.

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 float value within the interval $[1, 5]$. The mean rankings are presented in Table 3 and Figure 5.

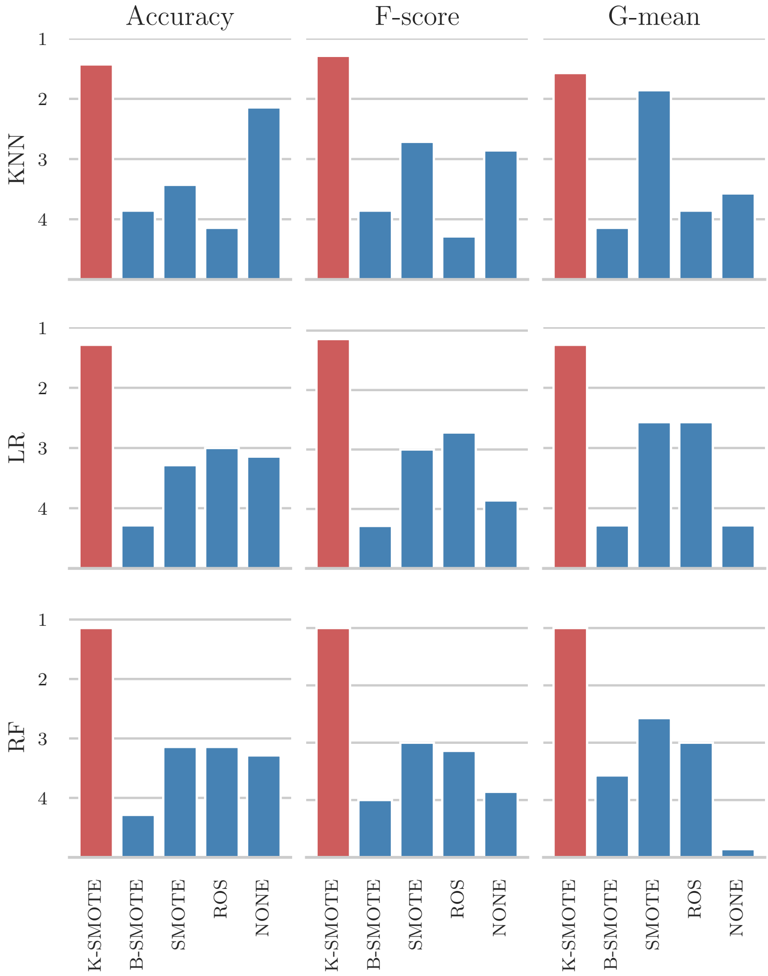



Figure 5: Mean ranking of oversamplers across datasets.

The mean ranking results show that K-SMOTE consistently presents the best results for every classifier and performance metric used. This is visually depicted in Figure 5. The quantitative results of this analysis is presented in Table 3. In addition to its better performance, in most 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).

| 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 | **1.29 ± 0.18** |
| LR | F-score | 3.86 ± 0.26 | 2.71 ± 0.47 | 3.00 ± 0.44 | 4.29 ± 0.42 | **1.14 ± 0.14** |
| LR | G-mean | 4.29 ± 0.18 | 2.57 ± 0.48 | 2.57 ± 0.3 | 4.29 ± 0.42 | **1.29 ± 0.18** |
| KNN | Accuracy | 2.14 ± 0.55 | 4.14 ± 0.34 | 3.43 ± 0.43 | 3.86 ± 0.34 | **1.43 ± 0.2** |
| KNN | F-score | 2.86 ± 0.4 | 4.29 ± 0.29 | 2.71 ± 0.61 | 3.86 ± 0.34 | **1.29 ± 0.18** |
| KNN | G-mean | 3.57 ± 0.48 | 3.86 ± 0.4 | 1.86 ± 0.46 | 4.14 ± 0.26 | **1.57 ± 0.2** |
| RF | Accuracy | 3.29 ± 0.52 | 3.14 ± 0.4 | 3.14 ± 0.51 | 4.29 ± 0.29 | **1.14 ± 0.14** |
| RF | F-score | 3.86 ± 0.46 | 3.14 ± 0.51 | 3.00 ± 0.44 | 4.00 ± 0.22 | **1.00 ± 0.0** |
| RF | G-mean | 4.86 ± 0.14 | 3.00 ± 0.31 | 2.57 ± 0.3 | 3.57 ± 0.37 | **1.00 ± 0.0** |

Table 3: Results for mean ranking of oversamplers across datasets.

The mean percentage difference among K-SMOTE and SMOTE is presented in Figure 6. It is calculated as the score difference among the test (K-SMOTE) and control (SMOTE) oversampler, divided by the control oversampler's score. K-SMOTE's average performance improves classification performance of up to 1.9% and outperforms all other methods, with the exception of two situations when using the G-mean scorer.

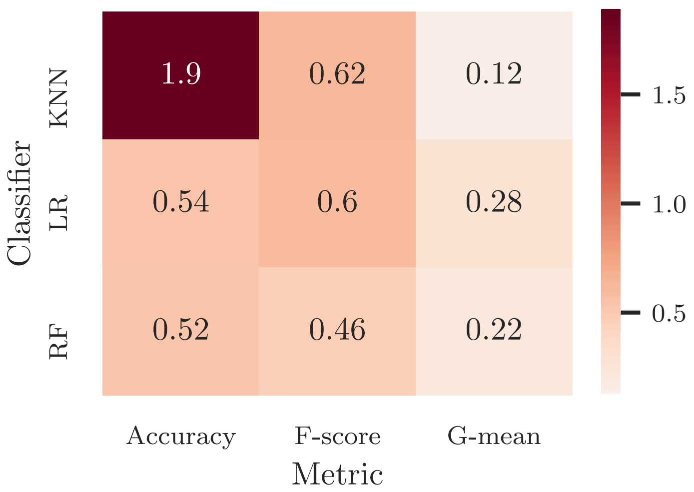



Figure 6: Mean score improvement (percentage difference) of the proposed method versus SMOTE across datasets.

The mean cross-validation scores are shown in Table 4. Considering the disparity of performance scores across datasets, the results presented in this table may not be as informative as the scores for each dataset, presented in Table 5. K-SMOTE's performance is the highest in most classifier/metric combinations and datasets, showing more inconsistency on the Indian Pines and Kennedy Space Center datasets.

| 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 | **0.909 ± 0.038** |
| LR | F-score | 0.891 ± 0.041 | 0.893 ± 0.042 | 0.893 ± 0.042 | 0.890 ± 0.042 | **0.898 ± 0.04** |
| LR | G-mean | 0.936 ± 0.025 | 0.940 ± 0.025 | 0.940 ± 0.025 | 0.937 ± 0.025 | **0.943 ± 0.024** |
| KNN | Accuracy | 0.879 ± 0.043 | 0.865 ± 0.048 | 0.867 ± 0.05 | 0.862 ± 0.054 | **0.881 ± 0.045** |

| Classifier | Metric | NONE | ROS | SMOTE | B-SMOTE | K-SMOTE |
|---|---|---|---|---|---|---|
| KNN | F-score | 0.859 ± 0.05 | 0.853 ± 0.049 | 0.861 ± 0.047 | 0.851 ± 0.053 | **0.866 ± 0.048** |
| KNN | G-mean | 0.919 ± 0.03 | 0.920 ± 0.029 | 0.926 ± 0.027 | 0.918 ± 0.03 | **0.927 ± 0.027** |
| RF | Accuracy | 0.898 ± 0.032 | 0.901 ± 0.031 | 0.900 ± 0.031 | 0.898 ± 0.032 | **0.905 ± 0.031** |
| RF | F-score | 0.879 ± 0.041 | 0.885 ± 0.037 | 0.887 ± 0.036 | 0.883 ± 0.037 | **0.891 ± 0.036** |
| RF | G-mean | 0.930 ± 0.024 | 0.935 ± 0.022 | 0.937 ± 0.021 | 0.935 ± 0.021 | **0.939 ± 0.02** |

Table 4: Mean cross-validation scores of oversamplers.

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.

| 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** |

| Dataset | Classifier | Metric | NONE | ROS | SMOTE | B-SMOTE | K-SMOTE |
|---|---|---|---|---|---|---|---|
| PU | LR | Accuracy | **0.905** | 0.897 | 0.897 | 0.891 | 0.904 |
| 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** |

Table 5: 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.

### 4.1 Statistical Analysis

The experiment's multi-dataset context was used to perform both a Friedman test [Friedman, 1937]. Table 6 shows the results obtained in the Friedman test performed, where the null hypothesis is rejected in all cases. Consequently, the Holm-Bonferroni comparison method (Holm's method) [Holm, 1979] is used for post-hoc analysis.

| 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 |

| Classifier | Metric | p-value | Significance |
|---|---|---|---|
| RF | G-mean | 1.8e-04 | True |

Table 6: 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.

The results of the Holm's method are shown in Table 7. Even though K-SMOTE outperforms the remaining oversamplers, the datasets' inherent high prediction scores make the rejection of this null hypothesis particularly difficult.

| Classifier | Metric | NONE | ROS | SMOTE | B-SMOTE |
|---|---|---|---|---|---|
| LR | Accuracy | 7.0e-02 | 7.0e-02 | 7.0e-02 | **2.6e-02** |
| LR | F-score | **1.5e-02** | 7.7e-02 | 7.7e-02 | **2.2e-02** |
| LR | G-mean | 5.1e-02 | 8.4e-02 | 8.4e-02 | **2.4e-02** |
| KNN | Accuracy | 5.7e-01 | 6.0e-02 | 2.1e-01 | 2.1e-01 |
| KNN | F-score | 1.5e-01 | 5.2e-02 | 1.5e-01 | 9.7e-02 |
| KNN | G-mean | 1.4e-01 | 8.8e-02 | 2.3e-01 | 1.4e-01 |
| RF | Accuracy | **4.4e-02** | **3.5e-02** | **4.4e-02** | **2.4e-02** |
| RF | F-score | 6.9e-02 | 6.9e-02 | 6.9e-02 | **3.6e-02** |
| RF | G-mean | 1.0e-01 | 1.0e-01 | 1.0e-01 | **3.8e-02** |

Table 7: Adjusted p-values using the Holm's method. Bold values are statistically significant at a level of $\alpha = 0.05$. The null hypothesis is that the test method does not perform better than the control method.

## 5 Conclusion

This research paper was motivated by the difficulty posed in classifying rare classes in Land Use/Land Cover tasks. A number of existing methods to address this problem (known as imbalanced learning) was identified and their caveats were exposed. Typically, these methods are not only difficult to implement, they are also context dependent. 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 datasets: Indian Pines, Pavia Centre, Pavia University, Salinas, Salinas A, Botswana and Kennedy Space Center. The experiment comprised a hyper-parameter search in order to tune each algorithm to its specific use case. 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 optimal mean validation score of each combination was then used to perform the analyses presented in this report.

In most cases, classification tasks using K-SMOTE led to better results than using the original, unmodified, imbalanced data. 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 [Douzas et al., 2018]. The proposed method also had the most consistent results across datasets, since it had the lowest standard deviations across datasets in most cases for both analyses, either based on ranking or mean cross-validation scores.

The proposed algorithm is an extension of the original SMOTE algorithm. In fact, the SMOTE algorithm represents a corner case of K-SMOTE: 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.

## References

[Abdi and Hashemi, 2016] Abdi, L. and Hashemi, S. (2016). To Combat Multi-Class Imbalanced Problems by Means of Over-Sampling Techniques. *IEEE Transactions on Knowledge and Data Engineering*, 28(1):238–251.

[Alonso-Sarria et al., 2019] Alonso-Sarria, F., Valdivieso-Ros, C., and Gomariz-Castillo, F. (2019). Isolation Forests to Evaluate Class Separability and the Representativeness of Training and Validation Areas in Land Cover Classification. *Remote Sensing*, 11(24):3000.

[Baumgardner et al., 2015] Baumgardner, M. F., Biehl, L. L., and Landgrebe, D. A. (2015). 220 Band AVIRIS Hyperspectral Image Data Set: June 12, 1992 Indian Pine Test Site 3. *Purdue University Research Repository*.

[Blagus and Lusa, 2010] Blagus, R. and Lusa, L. (2010). Class prediction for high-dimensional class-imbalanced data. *BMC bioinformatics*, 11(1):523.

[Bogner et al., 2018] Bogner, C., Seo, B., Rohner, D., and Reineking, B. (2018). Classification of rare land cover types: Distinguishing annual and perennial crops in an agricultural catchment in South Korea. *PLoS ONE*, 13(1).

[Cenggoro et al., 2018] Cenggoro, T. W., Isa, S. M., Kusuma, G. P., and Pardamean, B. (2018). Classification of imbalanced land-use/land-cover data using variational semi-supervised learning. In *Proceedings - 2017 International Conference on Innovative and Creative Information Technology: Computational Intelligence and IoT, ICITech 2017*, volume 2018-Janua, pages 1–6. IEEE.

[Chawla et al., 2002] Chawla, N. V., Bowyer, K. W., Hall, L. O., and Kegelmeyer, W. P. (2002). SMOTE: Synthetic Minority Over-sampling Technique. *Journal of Artificial Intelligence Research*, 16:321–357.

[Chawla et al., 2004] Chawla, N. V., Japkowicz, N., and Kotcz, A. (2004). Editorial: special issue on learning from imbalanced data sets. *ACM SIGKDD Explorations Newsletter*, 6(1):1.

[Cover and Hart, 1967] Cover, T. and Hart, P. (1967). Nearest neighbor pattern classification. *IEEE Transactions on Information Theory*, 13(1):21–27.

[Cui et al., 2019] Cui, Y., Jia, M., Lin, T. Y., Song, Y., and Belongie, S. (2019). Class-balanced loss based on effective number of samples. In *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, volume 2019-June, pages 9260–9269.

[Demšar, 2006] Demšar, J. (2006). Statistical comparisons of classifiers over multiple data sets. *Journal of Machine learning research*, 7(Jan):1–30.

[Dong et al., 2017] Dong, Q., Gong, S., and Zhu, X. (2017). Class Rectification Hard Mining for Imbalanced Deep Learning. In *Proceedings of the IEEE International Conference on Computer Vision*, volume 2017-Octob, pages 1869–1878.

[Douzas and Bacao, 2017] Douzas, G. and Bacao, F. (2017). Self-Organizing Map Oversampling (SOMO) for imbalanced data set learning. *Expert Systems with Applications*, 82:40–52.

[Douzas and Bacao, 2019] Douzas, G. and Bacao, F. (2019). Geometric SMOTE a geometrically enhanced drop-in replacement for SMOTE. *Information Sciences*, 501:118–135.

[Douzas et al., 2019] Douzas, G., Bacao, F., Fonseca, J., and Khudinyan, M. (2019). Imbalanced learning in land cover classification: Improving minority classes' prediction accuracy using the geometric SMOTE algorithm. *Remote Sensing*, 11(24):3040.

[Douzas et al., 2018] Douzas, G., Bacao, F., and Last, F. (2018). Improving imbalanced learning through a heuristic oversampling method based on k-means and SMOTE. *Information Sciences*, 465:1–20.

[Drusch et al., 2012] 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., and Bargellini, P. (2012). Sentinel-2: ESA's Optical High-Resolution Mission for GMES Operational Services. *Remote Sensing of Environment*, 120:25–36.

[Feng et al., 2019] Feng, W., Huang, W., and Bao, W. (2019). 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*, pages 1–5.

[Feng et al., 2018] Feng, W., Huang, W., Ye, H., and Zhao, L. (2018). Synthetic minority over-sampling technique based rotation forest for the classification of unbalanced hyperspectral data. In *International Geoscience and Remote Sensing Symposium (IGARSS)*, volume 2018-July, pages 2651–2654. Institute of Electrical and Electronics Engineers Inc.

[Fernández et al., 2013] Fernández, A., López, V., Galar, M., del Jesus, M. J., and 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.

[Ferreira et al., 2019] Ferreira, M. P., Wagner, F. H., Aragão, L. E., Shimabukuro, Y. E., and de Souza Filho, C. R. (2019). Tree species classification in tropical forests using visible to shortwave infrared WorldView-3 images and texture analysis. *ISPRS Journal of Photogrammetry and Remote Sensing*, 149:119–131.

[Friedman, 2001] Friedman, J. H. (2001). Greedy function approximation: A gradient boosting machine. *Annals of Statistics*, 29(5):1189–1232.

[Friedman, 1937] Friedman, M. (1937). The use of ranks to avoid the assumption of normality implicit in the analysis of variance. *Journal of the american statistical association*, 32(200):675–701.

[García et al., 2016] García, S., Ramírez-Gallego, S., Luengo, J., Benítez, J. M., and Herrera, F. (2016). Big data preprocessing: methods and prospects. *Big Data Analytics*, 1(1):9.

[Gavade and Rajpurohit, 2019] Gavade, A. B. and Rajpurohit, V. S. (2019). Systematic analysis of satellite image-based land cover classification techniques: literature review and challenges. *International Journal of Computers and Applications*, pages 1–10.

[Haibo He et al., 2008] Haibo He, Yang Bai, Garcia, E. A., and Shutao Li (2008). ADASYN: Adaptive synthetic sampling approach for imbalanced learning. In *2008 IEEE International Joint Conference on Neural Networks (IEEE World Congress on Computational Intelligence)*, pages 1322–1328. IEEE.

[Han et al., 2005] Han, H., Wang, W.-Y., and Mao, B.-H. (2005). Borderline-SMOTE: A New Over-Sampling Method in Imbalanced Data Sets Learning. In *International Conference on Intelligent Computing*, pages 878–887. Springer, Berlin, Heidelberg.

[He and Garcia, 2009] He, H. and Garcia, E. A. (2009). Learning from Imbalanced Data. *IEEE Transactions on Knowledge and Data Engineering*, 21(9):1263–1284.

[Holm, 1979] Holm, S. (1979). A simple sequentially rejective multiple test procedure. *Scandinavian journal of statistics*, pages 65–70.

[Holte et al., 1989] Holte, R. C., Acker, L., Porter, B. W., et al. (1989). Concept learning and the problem of small disjuncts. In *IJCAI*, volume 89, pages 813–818. Citeseer.

[Hounkpatin et al., 2018] Hounkpatin, K. O., Schmidt, K., Stumpf, F., Forkuor, G., Behrens, T., Scholten, T., Amelung, W., and Welp, G. (2018). Predicting reference soil groups using legacy data: A data pruning and Random Forest approach for tropical environment (Dano catchment, Burkina Faso). *Scientific Reports*, 8(1):1–16.

[Huang et al., 2016] Huang, C., Li, Y., Loy, C. C., and Tang, X. (2016). Learning deep representation for imbalanced classification. In *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, volume 2016-Decem, pages 5375–5384.

[Jeni et al., 2013] Jeni, L. A., Cohn, J. F., and De La Torre, F. (2013). Facing imbalanced data - Recommendations for the use of performance metrics. In *Proceedings - 2013 Humaine Association Conference on Affective Computing and Intelligent Interaction, ACII 2013*, pages 245–251.

[Jozdani et al., 2019] Jozdani, S. E., Johnson, B. A., and Chen, D. (2019). Comparing Deep Neural Networks, Ensemble Classifiers, and Support Vector Machine Algorithms for Object-Based Urban Land Use/Land Cover Classification. *Remote Sensing*, 11(14):1713.

[Kaur et al., 2019] Kaur, H., Pannu, H. S., and Malhi, A. K. (2019). A systematic review on imbalanced data challenges in machine learning: Applications and solutions. 52(4):1–36.

[Khatami et al., 2016] Khatami, R., Mountrakis, G., and Stehman, S. V. (2016). 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*, 177:89–100.

[Krawczyk, 2016] Krawczyk, B. (2016). Learning from imbalanced data: open challenges and future directions. 5(4):221–232.

[Lee et al., 2016] Lee, T., Lee, K. B., and Kim, C. O. (2016). Performance of Machine Learning Algorithms for Class-Imbalanced Process Fault Detection Problems. *IEEE Transactions on Semiconductor Manufacturing*, 29(4):436–445.

[Lemaître et al., 2017] Lemaître, G., Nogueira, F., and Aridas, C. K. (2017). Imbalanced-learn: A python toolbox to tackle the curse of imbalanced datasets in machine learning. *Journal of Machine Learning Research*, 18(17):1–5.

[Liaw et al., 2002] Liaw, A., Wiener, M., et al. (2002). Classification and regression by randomforest. *R news*, 2(3):18–22.

[Luengo et al., 2020] Luengo, J., García-Gil, D., Ramírez-Gallego, S., García, S., and Herrera, F. (2020). Imbalanced Data Preprocessing for Big Data. pages 147–160.

[Ma and Fan, 2017] Ma, L. and Fan, S. (2017). CURE-SMOTE algorithm and hybrid algorithm for feature selection and parameter optimization based on random forests. *BMC Bioinformatics*, 18(1):169.

[Maxwell et al., 2018] Maxwell, A. E., Warner, T. A., and Fang, F. (2018). Implementation of machine-learning classification in remote sensing: An applied review. *International Journal of Remote Sensing*, 39(9):2784–2817.

[McCullagh and Nelder, 1989] McCullagh, P. and Nelder, J. (1989). *Generalized Linear Models*.

[Mellor et al., 2015] Mellor, A., Boukir, S., Haywood, A., and Jones, S. (2015). 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*, 105:155–168.

[Olofsson et al., 2013] Olofsson, P., Foody, G. M., Stehman, S. V., and Woodcock, C. E. (2013). Making better use of accuracy data in land change studies: Estimating accuracy and area and quantifying uncertainty using stratified estimation. *Remote Sensing of Environment*, 129:122–131.

[Pedregosa et al., 2011] 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., and Duchesnay, É. (2011). Scikit-learn: Machine Learning in Python. *Journal of Machine Learning Research*, 12(Oct):2825–2830.

[Pelletier et al., 2017] Pelletier, C., Valero, S., Inglada, J., Champion, N., Marais Sicre, C., and Dedieu, G. (2017). Effect of Training Class Label Noise on Classification Performances for Land Cover Mapping with Satellite Image Time Series. *Remote Sensing*, 9(2):173.

[Pontius and Millones, 2011] Pontius, R. G. and Millones, M. (2011). Death to Kappa: Birth of quantity disagreement and allocation disagreement for accuracy assessment.

[Salzberg, 1994] Salzberg, S. L. (1994). C4.5: Programs for machine learning by j. ross quinlan. morgan kaufmann publishers, inc., 1993. *Machine Learning*, 16(3):235–240.

[Santos et al., 2015] Santos, M. S., Abreu, P. H., García-Laencina, P. J., Simão, A., and Carvalho, A. (2015). A new cluster-based oversampling method for improving survival prediction of hepatocellular carcinoma patients. *Journal of Biomedical Informatics*, 58:49–59.

[Shao et al., 2014] Shao, Y. H., Chen, W. J., Zhang, J. J., Wang, Z., and Deng, N. Y. (2014). An efficient weighted Lagrangian twin support vector machine for imbalanced data classification. *Pattern Recognition*, 47(9):3158–3167.

[Sharififar et al., 2019] Sharififar, A., Sarmadian, F., and Minasny, B. (2019). Mapping imbalanced soil classes using Markov chain random fields models treated with data resampling technique. *Computers and Electronics in Agriculture*, 159:110–118.

[Stromann et al., 2020] Stromann, O., Nascetti, A., Yousif, O., and Ban, Y. (2020). 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*, 12(1):76.

[Wang et al., 2019] Wang, R., Zhang, J., Chen, J.-W., Jiao, L., and Wang, M. (2019). Imbalanced Learning-based Automatic SAR Images Change Detection by Morphologically Supervised PCA-Net. *IEEE Geoscience and Remote Sensing Letters.*

[Wulder et al., 2018] Wulder, M. A., Coops, N. C., Roy, D. P., White, J. C., and Hermosilla, T. (2018). Land cover 2.0. *International Journal of Remote Sensing*, 39(12):4254–4284.

[Zhu et al., 2020] Zhu, M., Wu, B., He, Y. N., and He, Y. Q. (2020). LAND COVER CLASSIFICATION USING HIGH RESOLUTION SATELLITE IMAGE BASED ON DEEP LEARNING. *ISPRS - International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences*, XLII-3/W10:685–690.