Improving imbalanced predictions via a novel watershed-based oversampling method: Application to sickle cell disease

Yamna Ouchtar, Benjamin Perret, Christian Kassasseya, Pablo Bartolucci, Laurent Najman

Improving imbalanced predictions via a novel watershed-based oversampling method: Application to sickle cell disease.

Yamna Ouchtar[a], Benjamin Perret[a], Christian Kassasseya[b], Pablo Bartolucci[c,1], Laurent Najman[a]

[a] *LIGM, Univ Gustave Eiffel, CNRS, ESIEE Paris, F-77454, Cité Descartes, Marne-la-Vallée, 93160, , France*
[b] *Emergency Department, Hôpital Henri, Mondor, Assistance Publique, Hôpitaux de Paris, Créteil, , France*
[c] *Univ Paris Est Créteil, IMRB, Laboratory of excellence, Créteil, , France*
[d] *Univ Paris Est Créteil, Hôpitaux Universitaires Henri Mondor, APHP, Sickle cell referral center, UMGGR, Créteil, , France*

Abstract

Most of the time, real-world data sets are composed of classes with the same number of samples. But sometimes, for example in the case of fraud detection or rare disease diagnosis, the resulting dataset is composed of asymmetric classes. These datasets are called imbalanced datasets. Classification problems based on imbalanced datasets lead to errors and high variability. Thus, methods to deal with have been developed. In this paper, we propose a novel oversampling method, called WSSMOTE, based on the watershed transformation. We demonstrate that WSSMOTE improves prediction scores in some real-world datasets.

In addition, our main goal is to improve prediction scores of an imbalanced dataset composed of sickle cell disease (SCD) biomarkers. SCD is a serious-inherited disease, and patients with SCD can be affected by vaso-occlusive crises, which are the main cause of hospitalization. During these hospitalizations, acute chest syndrome (ACS) is the leading cause of death. ACS occurs

Email addresses: yamna.ouchtar@esiee.fr (Yamna Ouchtar), benjamin.perret@esiee.fr (Benjamin Perret), christian.kassasseya@outlook.fr (Christian Kassasseya), pablo.bartolucci@aphp.fr (Pablo Bartolucci), laurent.najman@esiee.fr (Laurent Najman)

in approximately 20% of hospitalized patients. During two prospective studies (PRESEV1, 247 patients and PRESEV2, 393 patients) a predictive score for ACS based on clinical and biological data has been developed. The obtained negative predictive value (NPV) is high and relatively similar in both studies (98.9% and 94%, respectively) but, the obtained positive predictive value (PPV) is low and highly variable (44.7% and 27.9%, respectively). Hence, we want to improve prediction performance while reducing its variability. We demonstrate that existing oversampling methods fail to improve the PPV value, whereas WSSMOTE succeed in doing so. With WSSMOTE, the PPV increases from 24.6% to 28.9%, while maintaining a high NPV (96.6%). In addition, the overfitting of the PPV value is reduced from 13.3 % to 1.2.

Keywords: Sickle cell disease, Imbalanced dataset, Oversampling method, Mathematical morphology, Watershed clustering

1. Introduction

Analyzing datasets and solving classification problems can provide essential information for the future. Most of the time, these classification problems are performed on datasets where the two classes are in equilibrium. In other words, if our dataset is binary, i.e., containing two classes A and B, it will have as many samples in class A as in class B. However, there are also many datasets where there is a disequilibrium between classes. For example, in a banking dataset, when the question is whether a transaction is fraudulent or not, 99% of these transactions will be normal and 1% will be considered fraudulent. These sets are called imbalanced datasets. The notion of imbalance is really important to consider, especially in the case of classification. An imbalanced dataset is defined through the notion of majority and minority classes where the ratio of the minority to the majority is usually between 1:1000 and 30:100.

An example of imbalanced dataset is the Predictive Severity Study (PRESEV). PRESEV is a study on Sickle Cell Disease (SCD) biomarkers. SCD was first described in the early 20th century by James Herrick, a Chicago physician [19]. It is now considered the most common monogenic disease in the world and has been recognized as a public health priority by UNESCO, the World Health Organization and the United Nations. SCD is a severe inherited monogenic disease caused by a mutation in the beta-globin gene on

chromosome 11 that results in the production of a pathological haemoglobin called HbS. Under certain conditions, this haemoglobin has the particularity of polymerizing within the red blood cells, which then become sickle-shaped, difficult to deform and obstruct the blood capillaries [29]. These vaso-occlusive crises (VOC) are extremely painful in the bones and represent the first cause of emergency room (ER) visits and hospitalizations of SCD patients [2]. Acute chest syndrome (ACS) is the most feared complication and the leading cause of mortality in patients hospitalized for VOC [33, 27, 35]. It is defined by clinical and/or radiological signs demonstrating the lung damage during a VOC. It appears on average after 2.5 days of evolution and affects about 17% of patients hospitalized for VOC [39]. The PRESEV dataset is imbalanced, as the ACS patients represented around 17:100 of the samples.

Prediction on imbalanced datasets, e.g., the PRESEV study, cannot be achieved using standard machine learning methods. Indeed, standard machine learning focuses on majority class and prediction scores are therefore biased. For this reason, methods, such as oversampling ones, have been developed. In this paper, we will first present an overview of existing oversampling methods; then, we show that these existing methods do not improve the prediction scores on the PRESEV study. Therefore, we have developed a novel pre-processing method called WSSMOTE, based on watersheds. We demonstrate that WSSMOTE improves prediction in some real-world datasets, and notably in the PRESEV case. In addition, WSSMOTE reduces the overfitting and cross-validation variability for ACS predictions.

The outline of this article is as follows: the first section explains the imbalanced issue and methods developed to reduce them. The second one is an in-depth presentation of the PRESEV study case. The third section introduces how watershed structure can improve prediction scores and how WSSMOTE has been designed. The last section shows the improvement in scores after using WSSMOTE in some real-world datasets and in PRESEV.

2. Literature of Imbalanced Datasets

2.1. Issues caused by Imbalanced Datasets

Imbalanced datasets are encountered in several research areas and studies. A repository called KEEL [1] identifies and stores datasets from several domains. Some of them are listed and explained in more details in the table 1.

	Abalone19 [30]	Page-blocks0	Paw	Pima [31]	Segment0 [6]	Vehicle1	Vowel0 [17]	Wisconsin	Yeast1	Haberman	Glass1	Ecoli1 [40]	Subcl35
Area	Biology	Class Text	Artificial Data	Diabetes	Image Segmentation	Transport	Deterding	Breast Cancer	Biology	Breast Cancer	Glass	Proteins	Artificial Data
Number of features	8	10	2	8	19	18	13	9	8	3	9	7	2
Number of samples	4174	5472	600	768	2308	846	988	683	1484	306	214	336	800
Number of minority data points	538	559	100	268	329	217	90	239	37	81	6	77	100
IR	12.9	8.79	5	1.87	6.02	2.9	9.11	34.97	2.46	2.78	35.46	22.94	7

Table 1: Description of Imbalanced Datasets selected from the KEEL repository [1].

Classification problems are solved through scores. Typical scores used for balanced classification are accuracy and error rate. They are defined using the notion of confusion matrix, explained in figure 2.

		Actual	
		Positive	Negative
Predicted	Positive	TP	FP
	Negative	FN	TN

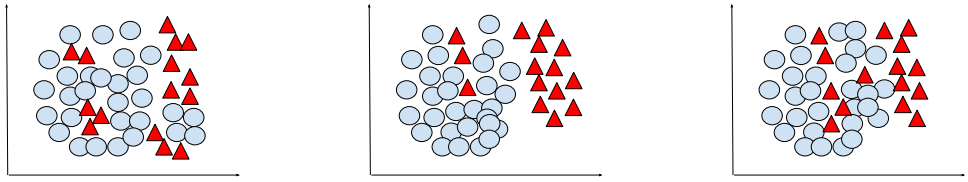
Table 2: Confusion Matrix

The accuracy is thus determined as $Acc = \frac{TP+TN}{TP+TN+FN+FP}$ and the error rate as $error = 1 - Acc$. The accuracy, mixes the number of predicted minority data points and the number of predicted majority ones. In the case of a balanced dataset, where both have the same weight, there is no issue. But, in the case of an imbalanced dataset, due to disproportionate weights, both numbers cannot be combined: the number of predicted minority data points is insignificant compared to the number of predicted majority data points. Other scores, such as the following, are more accurate:

- Precision = Positive Predictive Value (PPV) = $\frac{TP}{TP+FP}$
- Recall = Sensitivity = $\frac{TP}{TP+FN}$
- F1 score = $2 * \frac{Precision*Recall}{Precision+Recall}$
- AUC Score = $\frac{FP}{FP+TN}$
- GMean = $\sqrt{\frac{TP}{TP+FN} * \frac{TN}{TN+FP}}$

Once the score has been defined, many others issues can appear. Figure 1 shows some of them. The most common is the last one, the overlapping. Indeed, as clusters composed of minority data points are smaller than those composed of majority data points, ML algorithms are not able to detect them. This is illustrated in figure 1c, where red triangles have a high probability to

be considered as one cluster even though blue circles separate them. To avoid this issue and improve prediction, the most common strategy is to transform an imbalanced data set into a balanced one using oversampling methods.



(a) Disjunctions (b) Outliers (c) Overlapping

Figure 1: Several types of issues that can occur when the data is imbalanced. The blue circles represent the majority dataset, while the red triangles represent the minority dataset.

2.2. Oversampling Methods

The objective of oversampling methods is to generate new minority data points from the original data points in order to balance the data set. The simplest method is to add new data points randomly by selecting existing ones and generating their exact copies. However, by copying exactly the same data points, noisy data points may become too many and disturb the classification. To avoid this issue, several other methods based on different concepts have been developed. The following list is not exhaustive.

- Ordinary sampling and interpolation concept:
  - SMOTE [9]: It is the most popular oversampling method and many methods are based on it. The SMOTE algorithm can be decomposed into 3 steps. The first one is to construct the list of the k nearest neighbors of each minority data point. In a second time, n_added minority points are randomly chosen and for each minority data point x_1 picked, a random nearest neighbor x_2 of x_1 is chosen. The last step consists in building a new data point x between x_1 and x_2 with a linear interpolation:

$$\mathrm{x} = x_1 + \alpha(x_2 - x_1) \tag{1}$$

    where α is a scalar uniformly sampled in $[0, 1]$.

- Selection based on significance, data distribution, density, or relationships between data points:
  - ProWSyn [4]: For each minority data point, ProWSyn defines a proximity level. This proximity level assesses the relationship and distance between the minority and majority points. This score is then normalized and considered as a weight. Minority points located at the borders, i.e., close to majority points, will have a higher weight than those far from the borders. The minority points are then selected in proportion to their weight, and a linear interpolation (1) is used to create a new point x.
- By defining the space where minority data points can be generated, based on the data distribution and empty spaces. For example, by using clustering:
  - Geometric SMOTE [11]: Geometric SMOTE is a generalization of SMOTE with the objective of reducing the number of minority data points generated in the majority areas. Thus, geometric SMOTE defines an elliptical area around the minority data points and, by deformation and truncation, secures the area where the new data points are generated.
  - DBSMOTE [7]: DBSMOTE relies on the DBSCAN algorithm [37] to construct minority class clusters. This pre-clustering is used to estimate a local distribution of the data and to find the boundaries between the classes. Then, new minority data points are added using the equation (1) in the clusters.

As mentioned, this list is not exhaustive, there are dozens of oversampling methods that have been created over the years. But first, we will look at a concrete example of imbalanced data sets.

3. Case Study: PRESEV

A perfect example of imbalanced dataset is the PRESEV study. Because predictors of the risk of developing ACS during hospitalization have never been studied, a prospective single-center study (PRESEV 1 [3]) was conducted at Henri Mondor Hospital (Créteil, France) to determine whether ACS could be predicted from clinical and laboratory parameters assessed

on arrival at the emergency department with an VOC. This score was then validated in an international prospective study (PRESEV 2 [22]), including patients from Africa and Europe. The PRESEV study aimed to identify clinical and/or biological parameters predicting ACS at ER arrival. PRESEV had two main objectives: to rapidly identify patients at risk of worsening to ACS in order to offer them appropriate monitoring and care upon admission, and to rapidly identify patients who will not develop ACS in order to offer them shorter hospitalizations or even outpatient management. It was therefore necessary first to identify the clinical and/or biological features associated with the occurrence of ACS, and then to use them to define a predictive score for ACS.

3.1. Data Description

As mentioned, PRESEV is composed of two phases: PRESEV1 [3] and PRESEV2 [22]. The objective of PRESEV2 was to validate the model developed on the PRESEV1 dataset. Both studies included only adult patients with severe VOC. Severe VOC was defined as pain or tenderness affecting at least one part of the body, not controlled by grade II analgesics, and requiring opioids. The PRESEV1 study included 244 patients, but only 41 had developed an ACS. The PRESEV2 study included 393 patients, of whom 76 developed ACS. Thus, 16.8% and 19.3% of patients in PRESEV1 and PRESEV2 respectively developed a secondary ACS after admission to the emergency department. Each dataset is composed of several numerical features but only one categorical, a pain score (CPS).

3.2. Previous results

As explained, PRESEV main goal is to identify ACS patients at ER arrival with the lowest possible error. More precisely, we want to select, on the one hand, patients with the highest negative predictive value (NPV) and, on the other hand, patients with the highest positive predictive value (PPV). A high NPV means patients of have a high probability to develop ACS. A high PPV means patients with a low-risk of ACS; those patients can be promoted to ambulatory management, which reduce the length of their hospital stay, and so the cost to patients and communities [32]. Nevertheless, the main priority is to identify ACS patients, so the NPV error need to be less than 5%, i.e., a good NPV is an NPV above 95%.

In the previous studies, a predictive score for ACS on PRESEV1 [3] has been established using the following features: CPS, Leuc, Ret and Hem.

This score has a positive predictive value (PPV) of 44.7% for the high-risk group and a negative predictive value (NPV) of 98.9% for the low-risk group, which represented 39% of the study population. This score was then validated using PRESEV 2 [22], which found a PPV of 27.9% for the high risk score and an NPV of 94% for the low risk score, which represented 12.7% of the total population (results being published). Results are summarized in table 3. Results indicate that PRESEV2 validates the results obtained in PRESEV1. However, there is variability between the training (PRESEV1) and the testing (PRESEV2) part, i.e., the overfitting is around 16.8 % for the PPV value and around 5 for the NPV one.

	PPV (%)	NPV (%)
PRESEV1	44,7	98,9
PRESEV2	27,9	94

Table 3: PPV and NPV obtained by the method developed in articles [3, 22]

3.3. Reproduction of results using usual Machine Learning methods

In order to reduce the overfitting and also to develop a cross variability score, we decided to reproduce the previous score using machine learning (ML) methods. The first step is to select features. The first idea to do it will be to use some dimensional reduction methods, such as Principal Component Analysis (PCA) or an auto-encoder. But, because of the imbalanced criteria of our dataset, those methods will only focus on the majority class and then highlights features important for the majority class without taking care of the minority one. Thus, we select features by trying all the features combinations, as shown in table 4. We therefore decided to put together the PRESEV1 and PRESEV2 datasets into one dataset called PRESEVC. Then, we could define a training and a testing parts and apply on them standard machine learning method. To generate results we select the following usual ML methods: Random forest [5], Adaboost [15], MLP [20], SVM [34] and Logistic Regression [41]. Results leads to the conclusion that the best feature combination is the following one: Hem, Ret, Leuc, LDH, Urea and CRP.

However, the obtained NPV and PPV are not really impressive, probably because of the imbalanced nature of the PRESEV data. Thus, we decided to modify the prediction output into probabilities ones. Results are shown in table 5 and are closed to the one in the previous studies [3, 22]. However, the PPV value decreased and the overfitting is still high, around 13.3% for

Features	ML method	NPV	PPV
Ret, Leuc, Hem, ASAT, LDH, Bili_C, CPS, Urea, CRP	AdaBoost	32.1 ± 30	84 ± 1.3
Ret, Leuc, Hem, ASAT, LDH, Bili_C, CPS, Urea	KNN	37 ± 29	84 ± 2
Ret, Leuc, Hem, ASAT, LDH, Bili_C, CPS, CRP	RF	29.23 ± 18.4	85.4 ± 4.1
Ret, Leuc, Hem, ASAT, LDH, Bili_C, Urea, CRP	RF	33.3 ± 29.7	83.5 ±2.2
Ret, Leuc, Hem, ASAT, LDH, CPS, Urea, CRP	SVM	45.7 ± 30.9	84.5 ± 1.6
Ret, Leuc, Hem, ASAT, Bili_C, CPS, Urea, CRP	RF	37.3 ± 28.3	85.1 ± 2.6
Ret, Leuc, Hem, ASAT, Bili_C, CPS, Urea, CRP	RF	37.3 ± 28.3	85.1 ± 2.6
Ret, Leuc, Hem, LDH, Bili_C, CPS, Urea, CRP	AdaBoost	33.9 ± 29.3	84.4 ± 3.3
Ret, Leuc, Hem, ASAT, LDH, CPS, Urea	KNN	32.5 ± 28.9	83.3 ± 1.9
Ret, Leuc, Hem, ASAT, CPS, Urea, CRP	AdaBoost	44.6 ± 30.6	85.1 ± 1.5
...	...	...	...

Table 4: NPV and PPV values on the testing part obtained using different features.

the PPV and around 3.6% for the NPV value. Overfitting may be due to a lack of data, or also to the imbalanced specification of the dataset.

PRESEVC Hem, Ret, Leuc, LDH, CPS, ASAT, Urea and CRP	PPV	NPV
Train	35,9 ± 25.8	99,4 ± 1.6
Test	22.6 ± 15.8	95.8 ± 2

Table 5: NPV and PPV value obtained using probabilities output after feature selection.

3.4. Results using oversampling methods

To improve the predictions of PRESEV, we need to apply ML methods designed specifically for imbalanced datasets. There are more than 85 oversampling methods. In order to try all the different methods, and to select the best one for our classification problem, we use the smote_variants pipeline designed in [23, 24]. This pipeline finds the best oversampling method that optimizes the user-selected score on a user-provided imbalanced dataset. We apply the pipeline on the PRESEVC dataset, and ranked the oversampling methods with respect to (highest) PPV and NPV values, see table 6. We then reproduced the same strategy as before, by transforming the output of the binary ML algorithm into a probability one. We obtained the results in table 7.

We notice a decrease in cross variability and overfitting, of around 1% for the PPV value and 0.4% for the NPV. But, the overall value of PPV and NPV decreased compare to those reported in the paper literature [3, 22] and available in table 3. Nevertheless, the reduction of the cross variability

Oversampling	Classifier	PPV	NPV
Gaussian_SMOTE [25]	SVM	52 ± 27.7	83.7 ± 2.5
ROSE [28]	SVM	48.7 ± 24.1	85.1 ± 3.1
SMOBD [8]	SVM	48.3 ± 22.8	83.6 ± 2.4

Table 6: Ranking of oversampling methods for PRESEVC dataset on PPV and NPV values.

Gaussian_SMOTE	PPV	NPV
Train	24.6 ± 4.2	96.9 ± 2.9
Test	23.6 ± 3.5	96.5 ± 0.8

Table 7: PPV and NPV values obtained using Gaussian SMOTE [26] oversampling method on PRESEVC dataset.

and of the overfitting are really important for the experiment reproducibility. Therefore, the use of an oversampling method is relevant for better predictions on the PRESEV dataset. Our aim is thus to develop a new oversampling method that better fits the PRESEV c;assification problem.

4. Improve prediction: Materials and Method

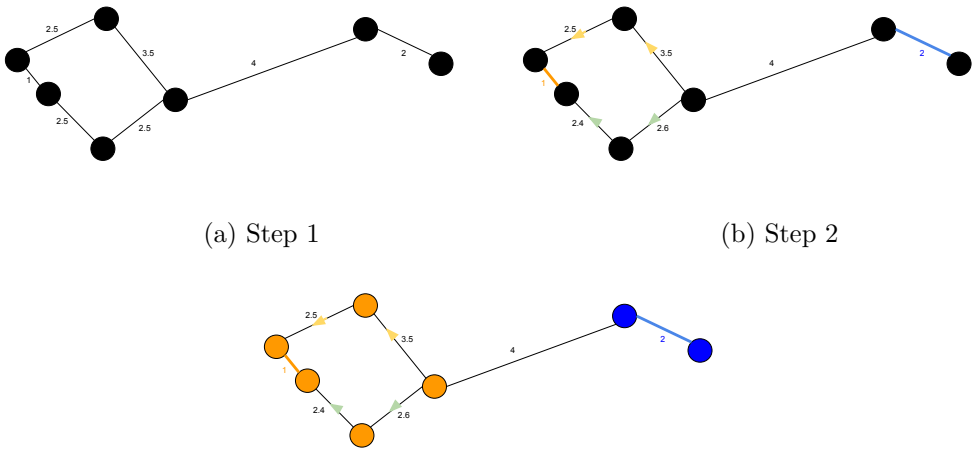
4.1. Watershed

Watershed was originally designed in the context of images, and more specifically to do image segmentation or object detection. But as watershed can be based on edge weighted graphs, the algorithm can be extended to numerical data and can be used for supervised and unsupervised classification problems. For example, in image segmentation or object detection, where the goal is to detect of a small amount of pixels in a background composed of many pixels. If we considered the background as the majority class and the object as the minority one, the idea of developing an oversampling method based on watershed method seemed relevant. Two methods based on watershed have been developed to resolve clustering issues: Watershed Cut [10] and Iterated Watershed [38].

Watershed Cut – A watershed cut of an edge-weighted graph is a way of partitioning the graph vertices based on the "drop of water" principle. In this paradigm, the weight of an edge represents elevation in a topological relief. If you drop a drop of water on a vertex, it will flow towards the adjacent vertex that follows the edge with the lowest weight. This drop will then flow until it finds a local minimum. All the vertices whose flow go into

the same minimum form a catchment basin. A watershed cut is then the partitioning of the graph vertices into such catchment basins.

We illustrate this algorithm using a simple example (See figure 2) of a 2 nearest neighbors graph. This graph is weighted by the Euclidean distances between each point, figure 2a. We follow a first drop on a descending path, until a minimum edge is found (yellow arrows in figure 2b). If this minimum does not belong to any cluster, as in figure 2b, all vertices belonging to this path are marked with the same labels. We then repeat this operation until all vertices belong to a cluster, as in figure 2c.



(a) Step 1

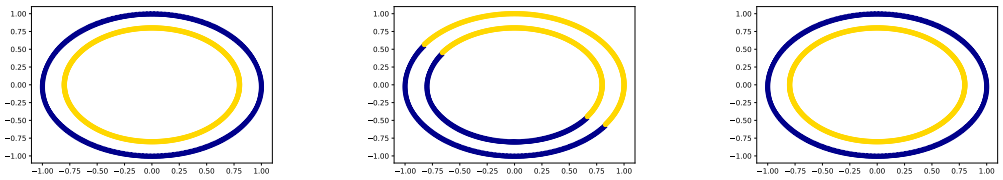
(b) Step 2

(c) Step 3

Figure 2: Obtaining two clusters using the Watershed Cut technique from an edge weighted graph of 2 nearest neighbors. The numbers on the edges represent the distance between two vertices, i.e., between two data points. There are two minima edges on this graph: one at 1 in yellow, and one at 2 in blue. At step 3, the graph is separated in two clusters by the paths descending to either the yellow or the blue minimum (see text).

Iterated Watershed – Watershed cut is not the only clustering method based on watershed. Iterated Watersheds [38] is a method close to KMeans [21] based on watershed. The difference between KMeans and Iterated Watersheds is the preservation of connectivity between data points. In fact, both the KMeans clustering and the Iterated Watershed algorithm can be described in two steps. But first, the user will have to choose a parameter k that corresponds to the number of desired clusters. Once this choice has

been made, the first step in both algorithms is called the maximization step. It consists of assigning each of the data points to one of the k centers. Here is the difference between the two algorithms. Indeed, in the case of KMeans, a point will be connected to its nearest center if the latter is closest to it in terms of distance. Whereas, the Iterated Watershed algorithm will work at path level. A point can only be linked to a center if there is a path between that point and the center and that path is minimal for a function, for example a distance function. In the Iterated Watershed, this step will be carried out by applying a shortest-path algorithm, the IFT [13], on the graph. Thus, the Iterated Watershed algorithm takes into account a connectivity criterion. Once this first step is completed, a partition of the data points is obtained. Then, a second step, the expectation step, is performed. This step consists of calculating the new center for each of the groups obtained by the partition. Thus, k new centers are obtained. We are then able to repeat the maximization and expectation steps, until convergence. Figure 3 compared KMeans and Iterated Clustering on a toy example.



(a) Toy example: two circles.

(b) Clusters obtained using KMeans.

(c) Clusters obtained using Iterated Watershed.

Figure 3: Clusters obtained on a toy example using two clustering methods: KMeans and Iterated Watershed.

4.2. Watershed Clustering vs Usual Clustering methods: Impact on Imbalanced Datasets.

Even if the idea of using image methods seem to be interesting, we need to compare results of these methods with usual ones and study their impact in same imbalanced datasets. For this purpose, we select two standard clustering methods often used in oversampling methods: KMeans [21] and DBSCAN [37] and 6 imbalanced datasets from table 1. Thus, our goal is to compare prediction scores. Since clustering methods are unsupervised methods, we define a procedure to generate predictions using them. First, we

generate clusters on the training part and the testing part using clustering methods. Secondly, we consider that all data points in the same cluster belong to the same class. Thus, unlabeled data points, i.e., data points from the testing part, will be labelled with the majority label of the cluster to which they belong. Using this method, we can compare the different clustering methods. We obtain the results presented in tables 8, 9, 10 and 11.

PPV	Paw	Pima	Segment0	Vowel0	Yeast1	Glass1
Kmeans	63,7 ± 16,5	43,2 ± 5,2	84,2 ± 3,9	87,7 ± 10,1	49,5 ± 11,5	48,3 ± 7,6
DBSCAN	26,6 ± 7	38,1 ± 3,6	41,3 ± 6,5	50,9 ± 14,3	53,2 ± 26,5	59,5 ± 13,7
Watershed Cut	34 ± 4,2	54,5 ± 17	75 ± 1,4	68,4 ± 9,8	42,8 ± 11,4	61 ± 37,3
Iterated Watershed	40,8 ± 10,5	42,1 ± 1,2	83,1 ± 3,8	33,1 ± 5	36,4 ± 15,2	42,7 ± 7,1

Table 8: Comparing precision scores obtained of Watershed Cut, Iterated Watershed and some usual clustering methods on Imbalanced datasets. Highlighted in blue is the best score obtained for a given dataset.

Recall	Paw	Pima	Segment0	Vowel0	Yeast1	Glass1
Kmeans	42,8 ± 7,6	44,8 ± 0	73,2 ± 1,2	71,7 ± 10,1	43,1 ± 11,5	46,3 ± 8,4
DBSCAN	29,2 ± 6,2	26,1 ± 3,7	23,2 ± 3,7	33,6 ± 6,8	14,6 ± 6,4	31,1 ± 9,8
Watershed Cut	58,0 ± 7,8	6,0 ± 1,5	87,2 ± 7,9	89,9 ± 7,6	43,1 ± 9,9	6,3 ± 4,6
Iterated Watershed	44,0 ± 10,0	46,3 ± 6	60,4 ± 4,3	71,8 ± 10,6	25,4 ± 13,8	39,5 ± 12,1

Table 9: Comparing recall scores for Watershed Cut, Iterated Watershed and some usual clustering methods on Imbalanced datasets. Highlighted in blue is the best score obtained for a given dataset.

Gmean	Paw	Pima	Segment0	Vowel0	Yeast1	Glass1
Kmeans	63,8 ± 5,5	54,9 ± 2,8	84,5 ± 0,4	84 ± 6	63,4 ± 8,8	56,4 ± 6,8
DBSCAN	50,4 ± 5,7	44,4 ± 1,2	46,7 ± 3,9	56,6 ± 5,6	35,6 ± 12,9	54 ± 7,1
Watershed Cut	69,4 ± 4	23,9 ± 3,2	91 ± 3,8	92,6 ± 4,2	63 ± 8	16 ± 13,9
Iterated Watershed	62,6 ± 7,4	55,1 ± 2,4	76,9 ± 2,9	78,0 ± 6,3	45,8 ± 18	47,4 ± 6,4

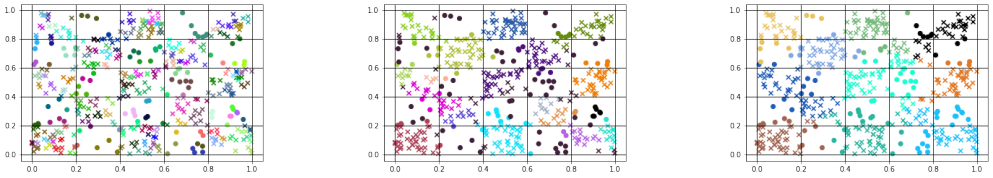
Table 10: Comparing GMean scores for Watershed Cut, Iterated Watershed, and some usual clustering methods on Imbalanced datasets. Highlighted in blue is the best score obtained for a given dataset.

We note and observe that Watershed Cut method improves predictions for some specific scores, e.g., recall or NPV scores. However, the iterated watershed method is less effective. In fact, except in special cases, results obtained using Iterated Watershed method can be compared with those obtained using KMeans. Thus, we decided to consider only the Watershed Cut method as an interesting method to develop a new oversampling method.

NPV	Paw	Pima	Segment0	Vowel0	Yeast1	Glass1
Kmeans	92,2 ± 1	69,4 ± 2,1	95,6 ± 0,2	97,2 ± 1	93,9 ± 1,2	71,2 ± 4,2
DBSCAN	89,7 ± 1	65,8 ± 0,9	88,1 ± 0,6	93,5 ± 0,6	91,3 ± 0,5	71,2 ± 2,8
Watershed Cut	93,3 ± 1	65,9 ± 0,6	97,8 ± 1,3	98,9 ± 0,8	93,8 ± 1,1	65,1 ± 0,9
Iterated Watershed	91,9 ± 1,4	69,7 ± 1,5	93,7 ± 0,7	96,8 ± 1,2	92,2 ± 1,3	64,9 ± 3,6

Table 11: Comparing NPV scores for Watershed Cut, Iterated Watershed, and some usual clustering methods on Imbalanced datasets. Highlighted in blue is the best score obtained for a given dataset.

In addition, we also want to compare the size and aspects of clusters obtained using Watershed Cuts, with those obtained using KMeans and DBSCAN methods. Thus, we made and studied clusters on 2D datasets. We observe (an example is done in figure 4) that Watershed Cut builds very small clusters compared to those obtained with KMeans and DBSCAN. But these clusters are very stable, unlike those obtained by DBSCSAN or KMeans. This could be explained because Watershed Cuts has no parameters related to the data structure.



(a) Watershed Cut Clustering: n_nei:10

(b) DBSCAN Clustering: eps:0.05, min_s:5

(c) KMeans Clustering: nb_clust:50

Figure 4: Imbalanced Chessboard clustered by different methods: Watershed Cut, DBSCAN and KMeans. Crosses correspond to majority data points, whereas circles correspond to minority ones.

4.3. Conclusion

The Watershed Cut seems to be an interesting method to deal with imbalanced datasets because of the typical shape of its clusters, and also due to the good scores obtained directly on the datasets compared to other clustering methods. Thus, we decided to build a new oversampling method based on Watershed Cut. We call this new approach *WSSMOTE*.

4.4. A novel oversampling method: WSSMOTE

WSSMOTE is an oversampling method based on the Watershed cut algorithm. Inspired by other oversampling methods, two different strategies have been developed in WSSMOTE. The first one is based on the ability of the watershed clustering algorithm to obtain small clusters that really match the data distribution. While the second one is based on the article [23], which states that the best oversampling methods based on clustering are those with large clusters, where new data points can thus easily be added far from each other. It therefore seems important to develop a solution where clusters can be concatenated into larger clusters, also called super clusters. We now examine these two strategies in detail.

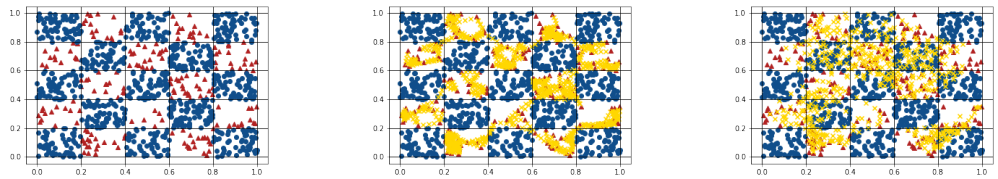
- The small cluster strategy: this strategy was developed because of the problems described in figure 1 and more particularly for disjunction (Fig. 1a) and overlapping issues (Fig. 1c). Indeed, in some datasets, it is really important to define small clusters that really correspond to the data distribution, in order not to add minority data points too far away, and thus to generate them on majority clusters. We therefore use watershed clustering to define small regions, and then generate data points in these regions. Using polynomial fit interpolation [16], minority data points can be added using one of the two following schemes. The first one is to use two random data points from the clusters ("mesh" option). The second one is to use the average data points from the clusters, and another random data point from that cluster ("star" option). These two options lead to different distributions of the final data (initial data points and newly generated minority data points).

- The super-cluster strategy: As explained above, the clusters created by watershed clustering are generally small; as a consequence, they may fail to capture large scale structures in the data. For example, in Figure 1b, each of the three red triangles on the left is considered a cluster using the watershed algorithm. Thus, the watershed clustering strategy will lead to three clusters, and we will therefore copy exactly these red triangles without adding any new information. In contrast, if we first concatenate the three clusters into one and then generate new minority data points, we capture information, and thus improve future prediction. This point was also made in [23]. This is why we developed this second strategy. We first generate our clusters using watershed

clustering, and then obtain a region adjacency graph, also called a rag. Each vertex of this rag represents a cluster, and each edge of the rag corresponds to a link between two clusters. Then we concatenate each cluster to its k nearest clusters. In this way, we obtain larger clusters that preserve well the original data distribution. The final step simply adds data points to the clusters, using the mesh option.

Pseudocode 1 describes the WSSMOTE algorithm. WSSMOTE has two different strategies. The first one, the small cluster one, can be called with the strategy parameter "star" or "mesh". This strategy consists of adding new data points directly inside the watershed clustering. The "star" option adds these data points using a notion of average, while the "mesh" option adds these data points by selecting two random data points within the cluster. The second strategy, the super-cluster one, called with the strategy parameter "concat_k", adds a step before generating new data points. It first concatenates k nearest clusters to create larger ones, and then adds new data points. So, to use this option, we also need to select a parameter k, which, in practice, is between 2 and 50.

The WSSMOTE algorithm is available on the following github url `https://github.com/yamnao/WSSMOTE`. WSSMOTE visualization on imbalanced datasets are shown in figure 5 and figure 6



(a) ChessBoard without oversampling

(b) ChessBoard with WSSMOTE oversampling: parameter "mesh"

(c) ChessBoard with WSSMOTE oversampling: parameter "concat_5"

Figure 5: Application of WSSMOTE on ChessBoard dataset[14]. Blue circles correspond to majority data points, red triangles to minority ones, and yellow crosses to data points added by the WSSMOTE method.

Algorithm 1: WSSMOTE

Data:

- imbalanced data D and its labels L
- nb_add: percentage of data points to be added
- strategy choice: choice between 'star', 'mesh', 'concat_k'
- k: parameter k, number of concatenate clusters

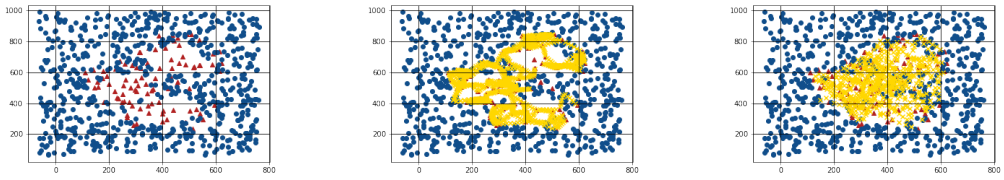
```
nb_to_add: (nb min data pts - nb maj data pts)*nb_add ;
graph, edge_weights: Generate the KNN graph;
Clusters: Watershed Clustering(graph, edge_weights) ;
if strategy choice == 'star' then
    for C in Clusters do
        X_mean: Calculate the mean of all the data pts in C ;
        D_C: Select random data points in C;
        Generate new data points between D_C and X_mean
         equidistantly ;
if strategy choice == 'mesh' then
    for C in Clusters do
        D_C: Select random data points in C;
        Generate new data points between two D_C data points ;
if strategy choice == 'concat_k' then
    ClustersConcat: Concatenate Clusters using Region Adjacency
     Graph and parameter k;
    for C in ClustersConcat do
        D_C: Select random data points in C;
        Generate new data points between two D_C data points ;
```

Result: *Data* D_C

5. Results and Discussion

5.1. Results on some datasets using WSSMOTE

To test our novel oversampling method, we used the datasets described in Table 1. We compared the rank of WSSMOTE with others oversampling methods implemented in smote_variant [24]. Table 12 summarizes the rank of WSSMOTE compared to other 50 oversampling methods. Table 13 summarizes the rank of Gaussian SMOTE method [25] compared to other 50



(a) Paw without oversampling

(b) Paw with WSSMOTE oversampling: “mesh”

(c) Paw with WSSMOTE oversampling: “concat_5”

Figure 6: Application of WSSMOTE on Paw dataset [1]. Blue circles correspond to majority data points, red triangles to minority ones, and yellow crosses to data points added by WSSMOTE method.

oversampling methods (WSSMOTE included).

	accuracy rank	**sensitivity rank**	**specificity rank**	**ppv rank**	**npv rank**	**gacc rank**	**f1 rank**	**auc rank**
yeast1	48	1	50	49	1	16	38	10
ecoli1	34	1	47	46	1	10	16	8
harbeman	27	2	43	24	1	6	11	6
wisconsin	11	3	33	32	4	8	8	12
vehicle1	25	2	46	46	1	19	13	1
glass1	22	2	45	44	2	6	10	11
subcl35	20	29	14	6	26	3	27	5

Table 12: Comparison of WSSMOTE with 50 other oversampling methods, using the smote pipeline. Each number corresponds to the rank of WSSMOTE, for a specific score, and for an imbalanced data set. The best rank is 1, the worst is 50.

	accuracy rank	**sensibility rank**	**specificity rank**	**ppv rank**	**npv rank**	**gacc rank**	**f1 rank**	**auc rank**
yeast1	8	2	9	6	2	43	42	9
ecoli1	3	2	3	2	2	17	5	3
haberman	37	1	7	3	41	12	10	7
wisconsin	42	30	19	27	33	35	42	2
vehicle1	43	1	3	2	33	42	42	47
glass1	31	2	1	14	44	42	42	37
subcl35	2	1	37	8	4	39	2	15

Table 13: Comparison of Gaussian SMOTE with 50 other oversampling methods, using the smote pipeline. Each number corresponds to the rank of Gaussian SMOTE, for a specific score, and for an imbalanced data set. The best rank is 1, the worst is 50.

Results show that WSSMOTE is competitive, especially for NPV and sensitivity scores. In addition, since we want to maximize the value of the

NPV in the prediction of the ACS using PRESEVC, WSSMOTE seems to be an appropriate method.

5.2. Results on the PRESEV dataset

Driven by the good results obtained with WSSMOTE, we tested this method for the prediction of ACS. Our goal, as explained in section 3, is to maximize both the NPV and the PPV scores, in order to identify ACS patients at ER arrival, but also to detect low-risk patients (to offer them shorter hospitalizations). By sorting results with their NPV score, we obtain that the best method is the combination of WSSMOTE and SVM, follow by the combination of Gaussian SMOTE and SVM and then of ROSE and SVM (results are available in the github). To illustrate the complete approach, we thus compare results obtained with the combination of WSSMOTE and SVM with previous results (See table 14).

Prediction Method	PRESEVC	**PPV (%)**	**NPV (%)**
SVM	Train	35,9 ± 25.8	99,4 ± 1.6
	Test	22.6 ± 15.8	95.8 ± 2
Gaussian SMOTE + SVM	Train	23.1 ± 1.1	96.5 ± 0.8
	Test	24.6 ± 4.2	96.9 ± 2.9
WSSMOTE + SVM	Train	27.7 ± 1.9	96.6 ± 0.4
	Test	28.9 ± 3.1	96.6 ± 2.5

Table 14: PPV and NPV scores obtained on PRESEVC using WSSMOTE, and comparison with previous results.

5.3. Discussion

Results demonstrated that WSSMOTE method improves the PPV score by 4.5% compared to the one obtained using Gaussian SMOTE method, which is significant. In addition, the overfitting decreases with WSSMOTE from 13.3% to 1.2% for PPV score and from 3.6% to 0 for the NPV one, compared to the overfitting obtained using usual machine learning method. Furthermore, the cross-validation variability also decreases from around 20.8% without oversampling, to 2.5% with WSSMOTE.

Thus, WSSMOTE improves the prediction of ACS for SCD patients. Better predictions mean a better identification of high-risk patients, and thus, better, more appropriate, monitoring and care. In addition, better prediction also means a better identification of low-risk patients, and thus shorter hospitalization stay. Moreover, as WSSMOTE decreases variability

of results, this method seems robust, and can be applied to new patients with more confidence.

6. Conclusion

WSSMOTE, an oversampling method based on watershed-cuts, improves the prediction of ACS for patients with SCD. PPV increases from around 4%, compared to the one obtained using other oversampling methods. This increase means better identification of ACS patients and thus better care and hospital management. Furthermore, overfitting decreases from 13.3 % to 1.2%, which means better reproducibility for future studies.

WSSMOTE also obtains competitive scores in other real-world datasets. Even if it is unrealistic to hope for a silver-bullet oversampling algorithm [36, 18, 12], this paper shows that improving classification scores on general data can be done thanks to algorithms that were originally designed for image segmentation, especially for imbalanced datasets.

References

[1] Alcalá-Fdez, J., Sanchez, L., Garcia, S., del Jesus, M.J., Ventura, S., Garrell, J.M., Otero, J., Romero, C., Bacardit, J., Rivas, V., et al., 2009. KEEL: a software tool to assess evolutionary algorithms for data mining problems. Soft Computing 13, 307–318.

[2] Ballas, S., Lusardi, M., 2005. Hospital readmission for adult acute sickle cell painful episodes: frequency, etiology and prognostic significance. American Journal of Hematology 79.

[3] Bartolucci, P., Habibi, A., Khellaf, M., Roudot-Thoraval, F., Melica, G., Moutereau, S., Loric, S., Wagner-Ballon, O., Berkenou, J., Santin, A., Michel, M., Renaud, B., Lévy, Y., Galactéros, F., Godeau, B., 2016. Score predicting acute chest syndrome during vaso-occlusive crises in adult sickle-cell disease patients. EBioMedicine 10. URL: `https://linkinghub.elsevier.com/retrieve/pii/S2352396416302961`.

[4] Barua, S., Islam, M., Murase, K., 2013. Prowsyn: Proximity weighted synthetic oversampling technique for imbalanced data set learning. PAKDD Advances in Knowledge Discovery and Data Mining.

[5] Breiman, 2001. Random forests. Journal of Biomedical Science and Engineering 45.

[6] Brodley, C., 2018. Uci image segmentation data. Massachusetts Vision Group URL: https://www.openml.org/d/36.

[7] Bunkhumpornpat, C., Sinapiromsaran, K., Lursinsap, C., 2012. Db-smote: Density-based synthetic minority over-sampling technique. Applied Intelligence 36.

[8] Cao, Q., Wang, S., 2011. Applying over-sampling technique based on data density and cost-sensitive svm to imbalanced learning. 2011 International Conference on Information Management, Innovation Management and Industrial Engineering 2.

[9] Chawla, N.V., Bowyer, K.W., Hall, L.O., Kegelmeyer, W.P., 2002. Smote: Synthetic minority over-sampling technique. Journal of Artificial Intelligence Research 16.

[10] Cousty, J., Bertrand, G., .Najman, L., Couprie, M., 2009. Watershed cuts: Minimum spanning forests and the drop of water principle. IEEE Transactions on Pattern Analysis and Machine Intelligence 31.

[11] Douzas, G., Bacao, F., 2017. Geometric smote: Effective oversampling for imbalanced learning through a geometric extension of smote. Information Sciences 501.

[12] Elrahman, S.A., Abraham, A., 2013. A review of class imbalance problem. Journal of Network and Innovative Computing 1.

[13] Falcao, A.X., Stolfi, J., de Alencar Lotufo, R., 2004. The image foresting transform: Theory, algorithms and applications. IEEE Transactions on Pattern Analysis and Machine Intelligence 26.

[14] Fernandez, A., Garcia, S., Galar, M., et al., 2018. Learning from Imbalanced Data Sets. Springer. doi:10.1007/978-3-319-98074-4.

[15] Freund, Y., Schapire, R., 1995. A decision-theoretic generalization of on-line learning and an application to boosting. Journal of Computer and System Sciences .

[16] Gazzah, S., ben Amara, N.E., 2008. New oversampling approaches based on polynomial fitting for imbalanced data sets. The Eighth IAPR International Workshop on Document Analysis Systems, DAS .

[17] Gorman, R.P., Sejnowski, T.J., 1988. Analysis of hidden units in a layered network trained to classify sonar targets. Neural Networks 1.

[18] Hassanat, A.B., Tarawneh, A.S., Altarawneh, G.A., Almuhaimeed, A., 2022. Stop oversampling for class imbalance learning: A critical review. Digital Object Identifier .

[19] Herrick, J., 1910. Peculiar elongated and sickle-shaped red blood corpuscles in case of severe anemia. The Yale journal of biology and medicine 6.

[20] Hinton, G., 1989. Connectionist learning procedures. Artificial intelligence .

[21] Jin, X., Han, J., 2007. K-means clustering. Encyclopedia of Machine Learning .

[22] Kassasseya, C., Sekou, K., Besse-Hammer, T., Nzouakou, R., Jean-Benoit, A., J.Magnang, L.Affo, Dautheville, S., Ngo, S., Khellaf, M., Diallo, D., Bartolucci, P., 2020. Validation of a predictive score of acute chest syndrome (presev-2 study) in adults. Blood 136. URL: https://ashpublications.org/blood/article/136.

[23] Kovacs, G., 2019a. An empirical comparison and evaluation of minority oversampling techniques on a large number of imbalanced datasets. Applied Soft Computing Journal 83.

[24] Kovacs, G., 2019b. Smote-variants: a python implementation of 85 minority oversampling techniques. Neurocomputing .

[25] Lee, H., Kim, J., Kim, S., 2017a. Gaussian-based smote algorithm for solving skewed class distributions. The International Journal of Fuzzy Logic and Intelligent Systems .

[26] Lee, H., Kim, J., Kim, S., 2017b. Gaussian-based smote algorithm for solving skewed class distributions. Int. J. Fuzzy Logic and Intelligent Systems .

[27] Maitre, B., Habibi, A., Roudot-Thoraval, F., Bachir, D., Belghiti, D.D., Galacteros, F., Godeau, B., 2000. Acute chest syndrome in adults with sickle cell disease. Chest 117.

[28] Menard, 2014. Training and assessing classification rules with imbalanced data. Data Mining and Knowledge Discovery .

[29] Murray, N., May, A., 1988. Painful crises in sickle cell disease-patients' perspectives. British Medical Journal 297.

[30] Nash, W.J., Sellers, T.L., Talbot, S.R., Cawthorn, A.J., Ford, W.B., 1994. The population biology of abalone haliotis species) in tasmania. i. blacklip abalone (h. rubra) from the north coast and the islandsof bass strait. Sea Fisheries Division, Technical Report .

[31] Nnamoko, N., Korkontzelos, I., 2020. Efficient treatment of outliers and class imbalance for diabetes prediction. Artificial Intelligence in Medicine 104.

[32] Pelinski, Y., Mescam, C., Kassasseya, C., Luna, G.D., Guillet, H., Noizat, C., Lejeune, S., Martino, S., Bachir, D., Driss, F., Jebali, A., D'Orengiani, A.P.H.D., Lemonier, N., Pirenne, F., Habibi, A., Bartolucci, P., 2021. Drepadom - establishment of home care services and hospitalizations for sickle cell disease patients as standard care since the covid-19 pandemic. Blood 138.

[33] Perronne, V., Roberts-Harewood, M., Bachir, D., Roudot-Thoraval, F., Delord, J., Thuret, I., Schaeffer, A., Davies, S.C., Galactéros, F., Godeau, B., 2002. Patterns of mortality in sickle cell disease in adults in france and england. Hematol J. 3.

[34] Platt, J.C., 1999. Probabilistic outputs for support vector machines and comparisons to regularized likelihood methods. Advances in Lage margin Classifiers .

[35] Platt, O., Brambilla, D., Rosse, W., Milner, P., Castro, O., Steinberg, M., Klug, P., 1994. Mortality in sickle cell disease. life expectancy and risk factors for early death. New England Journal of Medicine 330.

[36] Santos, B., Wijayanto, H., Notodiputro, K., Sartono, B., 2017. Synthetic over sampling methods for handling class imbalanced problems: A review. IOP Conference Series: Earth and Environmental Science 58.

[37] Schubert, E., Sander, J., Ester, M., Kriegel, H., Xu, X., 2017. DBSCAN revisited, revisited: why and how you should (still) use DBSCAN. ACM Transactions on Database Systems (TODS) 42, 1–21.

[38] Soor, S., Challa, A., Danda, S., Sagar, B.D., Najman, L., 2021. Iterated watersheds, a connected variation of k-means for clustering gis data. IEEE Transactions on Emerging Topics in Computing 9.

[39] Vichinsky, E., Neumayr, L., Earles, A., Williams, R., Lennette, E., Dean, D., Nickerson, B., Orringer, E., Mckie, V., Bellevue, R., Daeschner, C., Manci, E., 2000. Causes and outcomes of the acute chest syndrome on sickle cell disease. national acute chess syndrome study group. New England Journal of Medicine 342.

[40] Yang, Y., Jobin, C., 2014. Microbial imbalance and intestinal pathologies: connections and contributions. Dis Model Mech 7.

[41] Zhu, C., Byrd, R.H., Lu, P., Nocedal, J., 1997. Algorithm 778: L-bfgs-b: Fortran subroutines for large-scale bound-constrained optimization. ACM Transactions on Mathematical Software .