Improving KNN classification under Unbalanced Data.
A New Geometric Oversampling Approach

Alexandre Miguel de Carvalho
Center of Math, Computing and Cognition (CMCC)
Universidade Federal do ABC (UFABC)
Santo Andre, Brazil
carvalho.alexandre@ufabc.edu.br

Ronaldo Cristiano Prati
Center of Math, Computing and Cognition (CMCC)
Universidade Federal do ABC (UFABC)
Santo Andre, Brazil
ronaldo.prati@ufabc.edu.br

***Abstract*—Training of classifiers with unbalanced data is one of the main challenges in the field of Machine Learning. Some techniques to try to get around this problem have been proposed, among which the most important is SMOTE. This paper proposes a new approach to the balancing of classifier training data. It is a geometric and spatial approach, using tetrahedralization by Delaunay tessellation. Applying this new method, we can notice an improvement in the classification quality (in terms of AUC) of the KNN classification algorithm when compared with SMOTE.**

***Index Terms*—unbalanced, imbalanced, KNN, classification, machine learning, smote, geometric oversampling, Delaunay, Tetrahedron.**

I. Introduction

In machine learning, the classification process of real data is a big challenge, due to many issues which difficult the tasks of learning the classifiers. One of these issues is class imbalance [15]. This issue is very important as most real problems are naturally unbalanced [21]. Training machine learning with unbalanced data is a recognized problem. The performance of learning algorithms is generally compromised by the higher prevalence of the majority class, and to minimize this problem, some techniques were proposed.

SMOTE [3] is one of the most important algorithms to deal with unbalanced data training. Popular machine learning toolkits like scikit-learn [11], [13], Weka [6] and R [16], [19] have this algorithm implemented, either in their core as a contributed package. Furthermore, some variations of SMOTE have been proposed by focusing the generation of new instances in some areas of the instance space, like borderline SMOTE, and safe-level SMOTE. The main idea of SMOTE is to synthetically generate new instances, aiming to balance the training data distribution prior to apply the learning algorithms.

In spite of this popularity, the SMOTE algorithm and its derivatives suffer from a particular weakness: the generation of new stances is restrict to the line segment joining two instances. In our experiments, we notice that the possible gain by applying SMOTE to deal with imbalanced data sets is modest when we use the k-nearest neighbor algorithm for learning classifiers.

In this work, we propose a new method to generate new instances that can interpolate with a set of points. The approach used the Delaunay tessellation to build a mesh, which completely divides the input space into to polyhedrons[1]. The interpolation process is then performed by considering the barycenter of the polyhedron, thus interpolating over all points which forms the polyhedron. Therefore, in this paper we argue that a superior means of synthesizing instances by interpolating with more than two instances tends to produce better results, in terms of the Area under the ROC curve (AUC), when compared with traditional SMOTE

The remaining of this paper is organized as follows: in section II there are some explanations about SMOTE and its derived methods; in section III we explain our new method. This section is subdivided in four subsections. Subsection III-A describes what is Delaunay Tessellation; subsection III-B describe data transformation; subsection III-C show the new method to generate synthetic data and subsection III-D show experiments procedure. Section IV show and make a discussion about the results found by our new method. Finally, conclusions in section V.

II. Related Work

Haixiang [9], in his work, show history of processes and algorithms to try improve performance of unbalanced data training. Here in this section we talk about some SMOTE derived methods only.

A. Synthetic Minority Over-sampling Technique - SMOTE

SMOTE [3] is an over-sampling technique. Your main goal is generate artificial feature vectors of minority class. Balanced datasets have 50% of class 1 and 50% of class 2. SMOTE generate vectors of minority class to equality this quantity. To do that, SMOTE select one minority class sample, calculate k minority class nearest neighbors and, in a segment formed by

CAPES.

[1]In this particular study, the mesh is generated into a 3D space projection of the data set, so that the mesh is composed by a set of tetrahedrons

selected sample x_i and a random minority neighbor calculated by KNN x_j. In original paper [3], $k = 5$. Basically, new sample are generated by the equation:

$$x_{n+1} = x_i + r(x_j - x_i), \quad 0 \leq r \leq 1 \tag{1}$$

Where:

- x_{n+1} is a new instance vector;
- x_i is a vector feature i;
- x_j is a vector feature j;
- r is a random number.

B. Borderline-SMOTE

Borderline-SMOTE uses the same equation 1 but selected points to generate the new ones are taken from border between two classes, that is, vectors x_j, x_i closest to vectors of majority classes.

C. Safe-Level-SMOTE - SLSMOTE

Safe-Level-SMOTE uses equation 1 with some modifications. This method consider only vectors "safe" of the minority classes that to be used to generate new samples. For this method, definitions of "safe" is measured by *safe level ratio* or slr defined as:

$$slr = \frac{slp}{sln} \tag{2}$$

Where:

- slr, safe level ratio;
- slp, safe level for one x_i;
- sln safe level for one x_j;

K nearest neighbors are used to calculate slp and sln. The safe level is the number of samples of minority class founded by KNN. Criterion to use equation 1 is:

$$if \begin{cases} slr = 1 & \text{then} \quad r \in (0,1) \\ slr > 1 & \text{then} \quad r \in (0,(1/slr)) \\ slr < 1 & \text{then} \quad r \in ((1-slr),1) \end{cases}$$

D. V-synth

V-synth [20] is a oversampling technique that uses Voronoi Diagrams [4] to synthetize new minority classes samples. V-synth is a oversampling technique that uses Voronoi Diagrams to synthetize new minority classes samples. Voronoi Diagrams divide one region based on data points. Voronoi construct one cell around each data sample defining closest region. V-synth, to generate new samples, choose cells based on the following criteria: a cell must be inside a minority class sample and the hull must share an edge with, at least, one cell that inscribe a majority sample data. Figure 1 shows (in red) a Voronoi Diagram and a Delaunay Tesselletion (in black). These two diagrams are dual, one to each other. In future work we will use 3D Voronoi Diagram combined with 3D Delaunay Mesh to identify best regions to synthetize new minority class samples to balance dataset.

III. A New Geometric Approach based on Delaunay Tessellation

The basic idea to create new artificial samples is spatial interpolation. This geometric spatial interpolation allow us to generate new antithetical artificial instances considering more than two points, differently from SMOTE which only interpolates considering the line segment connecting two instances.

In a nutshell the main steps of our approach are:

1) Project the data set input space into a lower dimension (in this work, we project the data into 3D space)
2) Build a mesh using Delaunay Tessellation in the projected space. This process will divide the input space into a set of tetrahedrons.
3) For each tetrahedron in the mesh which have at least one instance of the minority class as a vertex, generate a new minority class instance by interpolating inside this tetrahedron. The interpolation is performed considering the projection of the barycenter of the tetrahedron into the original space.

These steps are detailed in the remaining of this section.

A. Delaunay Tessellation

A Delaunay triangulation is a graph that have the property that, given a set of two dimensional points (2D), three no collinear points of this set form a Delaunay's triangle if and only if the circumscribe circle (circumcircle) associated with it, if there is no other point in the set that is within this circumcircle [4].

On the other hand, for three dimensional points, four no collinear points of this set form a Delaunays tetrahedron if and only if the circumscribe sphere (circumsphere) associated with it, if there is no point in the set that is within this circumsphere around this tetrahedron. The process can be generalized to larger dimensions, where a n-dimensional simplex if formed if and only if the circumscribed hypersphere does not contain any other point.

The Delaunay tessellation has some interesting properties. In the plane, the triangulation is a maximum planar graph, and completely divides the space by triangles (if no four points are co-linear). The center of the circumsphere of a Delaunay polyhedron is a vertex of the Voronoi cells. Figure 1 shows an example of a Voronoi diagram (in red) and a Delaunay tessellation (in this case a triangulation — the dashed line) for a two-dimensional space. The closest neighbor P_c to any point P_x is on an edge $\overline{P_c P_x}$ in the Delaunay triangulation, as the nearest neighbor graph is a subgraph of the Delaunay triangulation. This unique set of neighboring points defines the neighborhood of the point and represents a parameter free definition of the surroundings of a point. The triangulation maximizes the minimum angle of the triangles, avoiding the occurrence of silver triangles [2]. For dimensions greater than two, no point in is inside the circumscribed hypersphere of any simplex in the tessellation. Delaunay tessellation may be

[2]A silver triangle is a triangle with one or two extremely acute angles, hence with a long/thin shape

used to approximate a manifold by a simple mesh [7], [18], in computational geometry [5], [10] and planning in automated diving [1], among others.

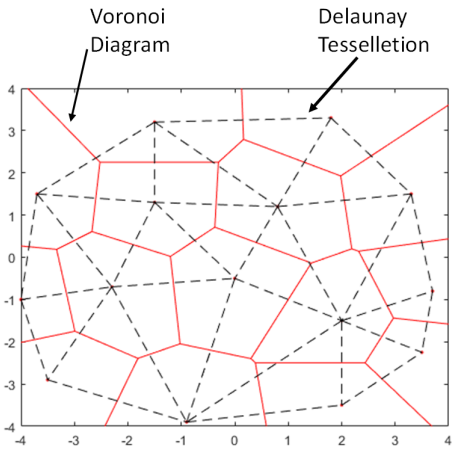

Fig. 1. Generic Voronoi and Delaunay diagrams.

B. Description

Let a data set be composed by instances (**X**,**Y**), where **X** is a matrix corresponding to the feature space and **Y** is a vector of labels associated to each instance, as shown in Equation 3. For the sake of explanation, we add an extra column representing the index **i** of each instance in **X**. Each $\{x_0, x_1, \cdots, x_n\}$ represents one instance per line. Finally, y_i in the **X** vectors corresponds to value or target of vector with index **i**. **X** (Equation 3) have m samples and, each sample, have n features.

$$(\mathbf{X}, \mathbf{Y}) = \begin{matrix} i_0 \\ i_1 \\ i_2 \\ i_3 \\ \vdots \\ i_m \end{matrix} \begin{bmatrix} x_0 & x_1 & \cdots & x_{n-1} \\ x_0 & x_1 & \cdots & x_{n-1} \\ x_0 & x_1 & \cdots & x_{n-1} \\ x_0 & x_1 & \cdots & x_{n-1} \\ \vdots & \vdots & \ddots & \vdots \\ x_0 & x_1 & \cdots & x_{n-1} \end{bmatrix}, \begin{bmatrix} y_0 \\ y_1 \\ y_2 \\ y_3 \\ \vdots \\ y_m \end{bmatrix} \tag{3}$$

Prior to building the mesh, in this work, data with n dimensions are transformed in three dimensional space ($3D$), or, samples **X** in equation 3 are transformed in $\mathbf{X}^3_{projected}$ in equation 6 by using some feature reduction technique. The reason for applying a dimensionality reduction is twofold: first, m points and n dimensions, a generic mesh contains $\mathcal{O}(m^{\lceil n/2 \rceil})$ polyhedrons, where each polyhedron is composed by $n+1$ points. Thus, we will have many polyhedrons to interpolated, and each polyhedron will be composed by many points. Second, the computational complexity of mesh building will explosively increase with large data dimensions. Hence, projecting the high-dimensional data to a lower dimension space is a way to make the problem of mesh building more manageable. As we are projecting into 3D space, the mesh is formed by tetrahedrons, and the interpolation process is carried out with four points. However, the process is generic and in future work, larger dimensions like $4D$ and $5D$ will be tested.

$$\mathbf{X}^n \xrightarrow{\text{Projection}} \mathbf{X}^3_{projected} \tag{4}$$

After that, the Delaunay tessellation (Equation 5) is built in the projected space using the Python scipy wrapper to QHull library [2]. As the $\mathbf{X}^3_{projected}$ is in 3D space, the generated mesh tessellation corresponds to a set of polyhedrons with four points (tetrahedrons).

$$\mathbf{X}^3_{projected} \xrightarrow{\text{Mesh}} \mathbf{T}^4 \tag{5}$$

As a result, a 3D tetrahedron mesh simplex, as sown in figure 2, is available to next step (Equation 6), where k tetrahedrons are created and $t^k_0, t^k_1, t^k_2, t^k_3$ are indexes corresponding to instance indexes $i_0 \ldots i_m$ in **X**.

$$\mathbf{T} = \begin{bmatrix} t^0_0 & t^0_1 & t^0_2 & t^0_3 \\ t^1_0 & t^1_1 & t^1_2 & t^1_3 \\ t^2_0 & t^2_1 & t^2_2 & t^2_3 \\ t^3_0 & t^3_1 & t^3_2 & t^3_3 \\ \vdots & \vdots & \vdots & \vdots \\ t^k_0 & t^k_1 & t^k_2 & t^k_3 \end{bmatrix} \tag{6}$$

Figure 3 shows a pictorial representation of a mesh element (a tetrahedron) with points $\{t_0, t_1, t_2, t_3\}$ correspond to indexes of **X**.

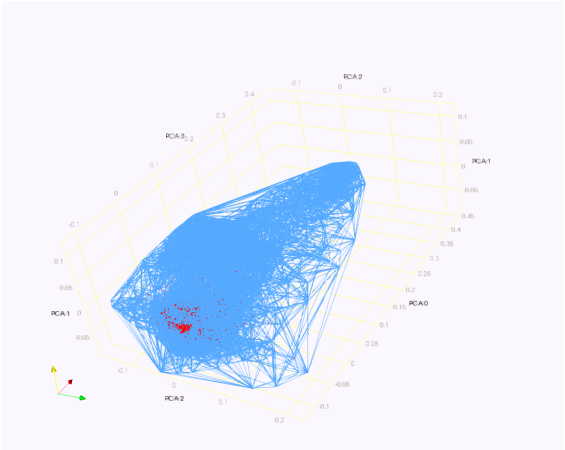

Fig. 2. x3data - 3D Tetrahedron Mesh. Minority Class in Red

C. New Synthetic Instances Generation

New Instances are created using each tetrahedron's barycenter. The barycenter corresponds to the point which has the same distance to all vertices in the tetrahedron. An example of the barycenter of a tetrahedron is show as the red point in Figure 3. To do that, we select only the tetrahedrons that have a vertex which the instance corresponds to a minority class instance. The vector **B** with barycenter point in *barycentric's coordination system* is used, as shown in Equation 7 show **B** vector.

$$\mathbf{B} = [b_0, b_1, b_2, b_2] = [1/4, 1/4, 1/4, 1/4]. \tag{7}$$

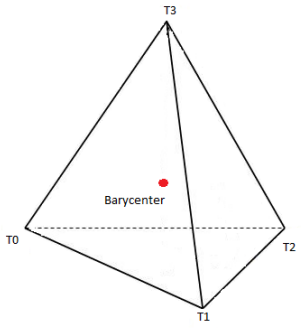

Fig. 3. 3D Element Mesh - Tetrahedron

Where **B** (Equation 7) is the *barycentric's coordination* and $b_0 + b_1 + b_2 + b_3 = 1$, necessarily.

$$\begin{pmatrix} t_0 \leftrightarrow i_\alpha \\ t_1 \leftrightarrow i_\beta \\ t_2 \leftrightarrow i_\gamma \\ t_3 \leftrightarrow i_\theta \end{pmatrix} \tag{8}$$

Equation 8 show correspondent indexes linking **T** with **X**. With one tetrahedron $\{t_0, t_1, t_2, t_3\}$ it is possible to find correspondent $xin\mathbf{X}$. Equation 9 is a generic tetrahedron with for elements x.

$$Tetrahedron = \mathbf{TH} = \begin{bmatrix} x_0^{i_\alpha} & x_1^{i_\alpha} & x_2^{i_\alpha} \cdots & x_n^{i_\alpha} \\ x_0^{i_\beta} & x_1^{i_\beta} & x_2^{i_\beta} \cdots & x_n^{i_\beta} \\ x_0^{i_\gamma} & x_1^{i_\gamma} & x_2^{i_\gamma} \cdots & x_n^{i_\gamma} \\ x_0^{i_\theta} & x_1^{i_\theta} & x_2^{i_\theta} \cdots & x_n^{i_\theta} \end{bmatrix} \tag{9}$$

Finally, new artificial samples are built by Equation 10. Or a **dot product** between **B** and **TH**

$$DSample_{1,n} = \left[\mathbf{B} \cdot \mathbf{TH}\right] \tag{10}$$

In this work, each tetrahedron generates a new sample. These new samples are used to train KNN algorithm.

D. Experiments

To validate our proposal, we select 15 unbalanced datasets from [12]. We select data sets that have only numerical features. Table I shows a summary of the main statics of the selected data sets: the number of samples, features and IR (Imbalanced Ratio). As can be seen from the table, the data sets have a great variety in terms of number of samples, features and imbalance ratios.

In this paper, we experiment with three different data reduction techniques to project the data sets into three dimensions: Multidimensional Scaling (MDS [17]), t-distributed Stochastic Neighbor Embedding (t-SNE [8]) and Principal Component Analysis (PCA [17]). PCA is a linear dimensionality reduction technique using Singular Value Decomposition of the data to project it to a lower dimensional space. MDS and t-SNE are non-linear techniques. The MDS algorithm aims to put each point in n-dimensional space such that the distances between object are preserved as well as possible. It converts similarities between data points to joint probabilities and tries to minimize the Kullback-Leibler divergence between the joint probabilities of the low-dimensional embedding and the high-dimensional data. Figure 4 show a diagram pipeline used to test this new approach.

TABLE I
DATASETS

Datasets	Code-Names	IR	Samples	Features
Ecoli	x1data	8.6	336	7
SatImage	x3data	9.3	6,435	36
Pen Digits	x4data	9.4	10,992	16
Abalone	x5data	9.7	4,177	8
Spectrometer	x7data	11	531	93
US Crime	x10data	12	1,994	122
Yeast ML8	x11data	13	2,417	103
Libras Move	x13data	14	360	90
Coil 2000	x15data	16	9,822	85
OIL	x18data	22	937	49
Wine Quality	x20data	26	4,898	11
Yeast ME2	x22data	28	1,484	8
Ozone Level	x24data	34	2,536	72
Mammography	x25data	42	11,183	6
Abalone 19	x27data	130	4,177	8

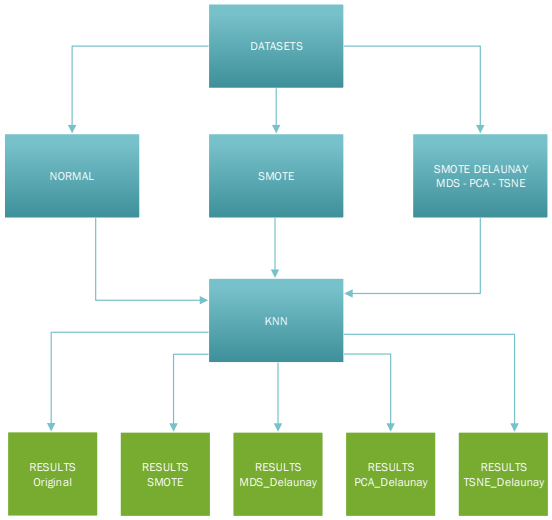

Fig. 4. Computational Experiments Diagram

All the experiments were carried out using cross validation with five folds subdivisions (5-fold cross validation). All datasets described in table I were used to validate the results. Our approach was implemented in Python with the INTEL PYTHON Distribution, and experiments were carried out using the Scikit-learn library [13]. As described early, data sets were projected to three dimensions. All algorithms were run with default parameters. The default number of neighbors in kNN is set to five in Scikit-learn.

IV. Results

Table II shows average area under the ROC curve (AUC [14]) about the experiments with our new geometric interpolation method, evaluated with three data projection techniques (as described in Section III-D). We have included as the base line SMOTE and the data set without any treatment (Original). The best result for each data set is highlighted in boldface. The last row shows the average rank position of each method.

Analyzing the table, it can be observed that applying the class imbalance treatment techniques is important for kNN with imbalance data. In only one case, there is no improvement in terms AUC compared with the treated data. This occurs for x4data, where the AUC using the original data is quite high (0.99859), and there is little room for improvement. SMOTE was the best algorithm in five cases, while our proposed approach was the best algorithm in nine cases (four using MDS, three using PCA and two using t-SNE projection techniques).

Furthermore, for the other three data sets were SMOTE was not able to improve performance (x1data, x13data and x15data), one of the variations of our proposed method have succeeded. In three cases (x5data, x7data, and x25data) the improvement given by SMOTE is very limited, and our method was able to achieve a much better performance. In the four cases that SMOTE wins, our algorithm did not beat the kNN with original data in only one case (x3data), and SMOTE has a larger difference (an difference in AUC larger than 0.01) in another dataset (x11data). We believe that this may be due to a poor data projection into three dimensions. For the other three, the differences of our proposal and SMOTE can be considered very low (an difference in AUC lower than 0.01).

In order to evaluate whether the differences among methods are statistically significant, we use a non-parametric Friedman multiple comparison testing. The Friedman test is the non-parametric equivalent of the two-way ANOVA. Under the null hypothesis, it states that all the algorithms are equivalent, so rejecting this hypothesis implies the existence of significant differences among the performance of all algorithms. When the null hypothesis is rejected by the Friedman test, we can proceed with a post-hoc test to detect what differences among the methods are significant. For this, we used Shaffer post-hoc multiple comparison for controlling the familywise error.

Results of the statistical tests are presented in the form of a diagram of significant differences. In these diagrams, results were ordered by decreasing performance, where the best algorithms are placed to the left of figure. A thick joining two methods that there is no statistical significance among the methods. Figure 5 shows all the pairwise comparison of the data methods evaluated in this study. Although there is no statistical difference between our proposed method and SMOTE, the three variations are slightly better than SMOTE. Moreover, the test shows a significance difference between our method and the untreated data when we use PCA and MDS for data projection. On the other hand, it is not possible to point

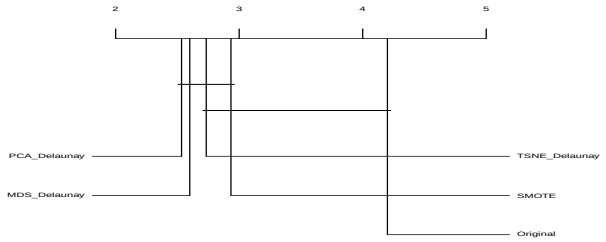

Fig. 5. All pairwise comparison

statistical difference among SMOTE, our method coupled with t-SNE and the untreated data.

V. Conclusion

In this paper a new Geometric Oversampling based on Delaunay tessellation was proposed. Experimental evaluation with 15 data sets shows an improvement in KNN classification quality, in terms of AUC. To generate the mesh, our method uses a data projection technique. We achieve better results sing PCA for this purpose, although a deeper investigation is necessary to study variating the dimensions of the projections.

This new method show us that it is necessary a deeper investigation. We plan to investigate this new method with other classifications methods like Decision Trees, Neural Networks, Support Vector Machine, among others. Furthermore, the tessellation may provide other information to improve the method. Aspects such as mesh quality, mesh density and others properties could be tested with KNN and others classifications algorithms. These properties may provide useful information for selecting better regions to interpolate (in a similar fashion as, e.g., borderline SMOTE and ADASYM), and thus with potential for improving performance. Others notable tetrahedron's points like orthocenter could be used to analyze classification performance, as well as explore mesh refinement techniques.

References

[1] S. J. Anderson, S. B. Karumanchi, and K. Iagnemma. Constraint-based planning and control for safe, semi-autonomous operation of vehicles. In *Intelligent Vehicles Symposium (IV), 2012 IEEE*, pp. 383–388. IEEE, 2012.

[2] C. B. Barber, D. P. Dobkin, and H. Huhdanpaa. The quickhull algorithm for convex hulls. *ACM TRANSACTIONS ON MATHEMATICAL SOFTWARE*, 22(4):469–483, 1996.

[3] N. Chawla, K. Bowyer, L. Hall, and W. Kegelmeyer. SMOTE: Synthetic Minority Over-sampling Technique Nitesh. *Journal of Artificial Intelligence Research*, pp. 321–357, jun 2002. doi: 10.1111/j.1464-410X.2007.07404.x

[4] M. De Berg, O. Cheong, M. Van Kreveld, and M. Overmars. *Computational Geometry: Algorithms and Applications*. 2008. doi: 10.1007/978-3-540-77974-2

[5] T. De Kok, M. Van Kreveld, and M. Löffler. Generating realistic terrains with higher-order delaunay triangulations. *Computational Geometry*, 36(1):52–65, 2007.

[6] E. Frank, M. A. Hall, and I. H. Witten. *The WEKA Workbench*, chap. Online Appendix. Morgan Kaufmann, fourth ed., 2016.

[7] Z. Gao, Z. Yu, and M. Holst. Feature-preserving surface mesh smoothing via suboptimal delaunay triangulation. *Graphical models*, 75(1):23–38, 2013.

TABLE II
AVERAGE AUC FOR ORIGINAL DATA, SMOTE AND OUR PROPOSED METHODS.

Dataset	Original	SMOTE	MDS_Delaunay	PCA_Delaunay	TSNE_Delaunay
x1data	0.89315	0.88820	**0.92270**	0.91167	0.86392
x3data	0.80393	**0.83671**	0.79491	0.79056	0.79905
x4data	**0.99859**	0.99832	0.99796	0.99809	0.99839
x5data	0.76432	0.76679	0.79190	0.79596	**0.80572**
x7data	0.93179	0.93826	0.95696	**0.95756**	0.95522
x10data	0.81226	**0.84689**	0.84309	0.84343	0.84448
x11data	0.55675	**0.60948**	0.58236	0.58177	0.56126
x13data	0.97026	0.94548	0.94311	**0.98486**	0.97209
x15data	0.60993	0.59941	0.62095	0.61307	**0.62178**
x18data	0.75020	0.82547	**0.85803**	0.85356	0.82969
x20data	0.69748	0.72347	**0.74003**	0.72928	0.73863
x22data	0.80530	**0.84921**	0.82828	0.83431	0.84848
x24data	0.62363	**0.66032**	0.65211	0.64844	0.62618
x25data	0.87092	0.87135	0.89700	**0.90587**	0.89186
x27data	0.53879	0.65676	**0.72606**	0.71149	0.62997
Average rank	4.2	2.93	2.6	**2.53**	2.73

[8] A. Gisbrecht, A. Schulz, and B. Hammer. Parametric nonlinear dimensionality reduction using kernel t-SNE. *Neurocomputing*, 147(1):71–82, 2015. doi: 10.1016/j.neucom.2013.11.045

[9] G. Haixiang, L. Yijing, J. Shang, G. Mingyun, H. Yuanyue, and G. Bing. Learning from class-imbalanced data: Review of methods and applications. *Expert Systems with Applications*, 73:220–239, may 2017. doi: 10.1016/j.eswa.2016.12.035

[10] R. Kolluri, J. R. Shewchuk, and J. F. O'Brien. Spectral surface reconstruction from noisy point clouds. In *Proceedings of the 2004 Eurographics/ACM SIGGRAPH symposium on Geometry processing*, pp. 11–21. ACM, 2004.

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

[12] G. Lemaitre, F. Nogueira, C. K. Aridas, and D. V. R. Oliveira. Imbalanced dataset for benchmarking, Sept. 2016. These data are originally from other repository. See the references to know which licensing applied to them. doi: 10.5281/zenodo.61452

[13] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12:2825–2830, 2011.

[14] R. C. Prati, G. E. A. P. A. Batista, and M. C. Monard. A survey on graphical methods for classification predictive performance evaluation. *IEEE Trans. Knowl. Data Eng.*, 23(11):1601–1618, 2011. doi: 10.1109/TKDE.2011.59

[15] R. C. Prati, G. E. A. P. A. Batista, and D. F. Silva. Class imbalance revisited: a new experimental setup to assess the performance of treatment methods. *Knowl. Inf. Syst.*, 45(1):247–270, 2015. doi: 10.1007/s10115-014-0794-3

[16] R Core Team. *R: A Language and Environment for Statistical Computing*. R Foundation for Statistical Computing, Vienna, Austria, 2017.

[17] Ruiping Wang, Shiguang Shan, Xilin Chen, Jie Chen, and Wen Gao. Maximal Linear Embedding for Dimensionality Reduction. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 33(9):1776–1792, sep 2011. doi: 10.1109/TPAMI.2011.39

[18] A. Samat, P. Gamba, S. Liu, P. Du, and J. Abuduwaili. Jointly informative and manifold structure representative sampling based active learning for remote sensing image classification. *IEEE Transactions on Geoscience and Remote Sensing*, 54(11):6803–6817, 2016.

[19] L. Torgo. *Data Mining with R, learning with case studies*. Chapman and Hall/CRC, 2nd ed., 2016.

[20] W. A. Young, S. L. Nykl, G. R. Weckman, and D. M. Chelberg. Using Voronoi diagrams to improve classification performances when modeling imbalanced datasets. *Neural Computing and Applications*, 26(5):1041–1054, jul 2015. doi: 10.1007/s00521-014-1780-0

[21] X. Zhang, Y. Li, R. Kotagiri, L. Wu, Z. Tari, and M. Cheriet. KRNN: k Rare-class Nearest Neighbour classification. *Pattern Recognition*, 62:33–44, 2017. doi: 10.1016/j.patcog.2016.08.023