Heuristic Data Augmentation Improves the Efficiency of Active Learning Methods

Joao Fonseca[1*], Fernando Bacao[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

Abstract goes here.

1 Introduction

Introduction goes here.

2 Active Learning Methods

Review on Data Augmentation Methods go here.

3 Data Augmentation Methods

Review on Data Augmentation Methods go here.

4 Proposed Method

5 Methodology

This section describes the different elements included in the experimental procedure. The datasets used were acquired in open data repositories and its sources and preprocessing steps are defined in Subsection 5.1. The choice of classifiers used in the experiment are defined in Subsection 5.2. The metrics chosen to measure AL performance and overall classification performance are defined in Subsection 5.3. The experimental procedure is described in Subsection 5.4. The implementation of the experiment and resources used to do so are described in Subsection 5.5.

The methodology developed serves 2 purposes: (1) Compare classification performance once all the AL procedures are completed (*i.e.,* optimal performance of a classifier trained via iterative data selection) and (2) Compare the amount of data required to reach specific performance thresholds (*i.e.,* number of AL iterations required to reach similar the optimal classification performance).

5.1 Datasets

The datasets used to test the proposed method are publicly available in open data repositories. Specifically, they were retrieved from OpenML and the UCI Machine Learning Repository. The datasets were chosen considering different domains of application, imbalance ratios, dimensionality and number of target classes, all of them focused on classification tasks. The goal is to demonstrate the performance of the different AL frameworks in various scenarios and domains. The data preprocessing approach was similar across all datasets. Table 1 describes the key properties of the 10 preprocessed datasets where the experimental procedure was applied.

Dataset	Features	Instances	Minority instances	Majority instances	IR	Classes
IMAGE SEGMENTATION	14	1155	165	165	1.0	7
MFEAT ZERNIKE	47	1994	198	200	1.01	10
TEXTURE	40	1824	165	166	1.01	11
WAVEFORM	40	1666	551	564	1.02	3
PENDIGITS	16	1832	176	191	1.09	10
VEHICLE	18	846	199	218	1.1	4
MICE PROTEIN	69	1073	105	150	1.43	8
GAS DRIFT	128	1987	234	430	1.84	6
JAPANESE VOWELS	12	1992	156	323	2.07	9
BASEBALL	15	1320	57	1196	20.98	3

Table 1: Description of the datasets collected after data preprocessing. The sampling strategy is similar across datasets. Legend: (IR) Imbalance Ratio

The data preprocessing pipeline is depicted as a flowchart in Figure 1. The missing values are removed from each dataset by removing the corresponding observations. This ensures that the input data in the experiment is kept as close to its original form as possible. The non-metric features (*i.e.,* binary, categorical and ordinal variables) were removed since the application of G-SMOTE is limited to continuous and discrete features. The datasets containing over 2000 observations were downsampled in order to maintain the datasets to a manageable size. The data sampling procedure preserves the relative class frequency of the dataset, in order to maintain the Imbalance Ratio (IR) originally found in each dataset (where $IR = \frac{count(C_{maj})}{count(C_{\min})}$). The remaining features of each dataset are scaled to the range of $[-1, 1]$ to

ensure a common range across features.

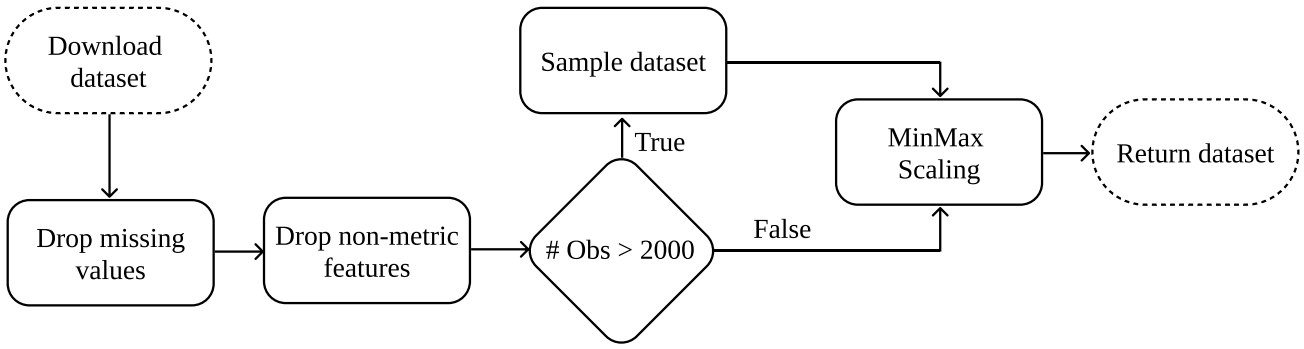

Figure 1: Data preprocessing pipeline.

The preprocessed datasets were stored into a SQLite database file and is available along with the experiment's source code in the GitHub repository of the project (see Subsection 5.5).

5.2 Machine Learning Algorithms

We used a total of 4 classification algorithms and a heuristic data augmentation mechanism. The choice of classifiers was based on the popularity and family of the classifiers (tree-based, nearest neighbors-based, ensemble-based and linear models). Our proposed method was tested using a Decision Tree (DT) [1], a K-nearest neighbors classifier (KNN) [2], a Random Forest Classifier (RF) [3] and a Logistic Regression (LR) [4]. Since the target variables are multi-class, the LR classifier was implemented using the one-versus-all approach. The predicted class is assigned to the label with the highest likelihood.

The oversampler G-SMOTE was used as a data augmentation method. The typical sampling strategy of oversampling methods is to generate artificial observations on non-majority classes such that the number of majority class observations matches those of each non-majority class. We modified this sampling strategy to generate observations for all classes, as a percentage of the number of observations in the majority class. In addition, the original G-SMOTE algorithm was modified to accept data selection probabilities based on classification uncertainty. These modifications are discussed in Section 4.

Every AL procedure was tested with different selection criteria: Random Selection, Entropy and Breaking Ties. The baseline used is the standard AL procedure. As a benchmark, we add the AL procedure using G-SMOTE as a normal oversampling method, as proposed in [5]. Our proposed method was implemented using G-SMOTE as a data augmentation method to generate artificial observations for all classes, while still balancing the class distribution, as described in Section 4.

5.3 Evaluation Metrics

Considering the imbalanced nature of the datasets used in the experiment, commonly used performance metrics such as Overall Accuracy (OA), although being intuitive to interpret, are insufficient quantify a model's classification performance [6]. The Cohen's Kappa performance metric, similar to OA, is also biased towards high frequency classes since its definition is closely related to the OA metric, making its behavior consistent with OA [7]. However, these metrics remain popular choices for the evaluation of classification performance. Other performance metrics like $Precision = \frac{TP}{TP+TN}$, $Recall = \frac{TP}{TP+FN}$ or $Specificity = \frac{TN}{TN+FP}$ are calculated as a function of True/False Positives (TP and FP) and True/False Negatives (TN and FN) and can be used at a per-class basis instead. In a multiple dataset with varying amount of target classes and meanings, comparing the performance of different models using these metrics becomes impractical.

Based on the recommendations found in [6, 8], we used 2 metrics found to be less sensitive to the class imbalance bias, along with OA as a reference for easier interpretability:

- The Geometric-mean scorer (G-mean) consists of the geometric mean of Specificity and Recall [8]. Both metrics are calculated in a multiclass context considering a one-versus-all approach. For multiclass problems, the G-mean scorer is calculated as its average per class values:

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

- The F-score metric consists of the harmonic mean of Precision and Recall. The two metrics are also calculated considering a one-versus-all approach. The F-score for the multi-class case can be calculated using its average per class values [6]:

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

- The OA consists of the number of TP divided by the total amount of observations. Considering c as the label for the different classes present in a target class, OA is given by the following formula:

$$OA = \frac{\sum_c \mathrm{TP}_c}{\sum_c (\mathrm{TP}_c + \mathrm{FP}_c)}$$

The comparison of the performance of AL frameworks is based on its data selection and augmentation efficacy. Specifically, an efficient data selection/generation strategy allows the production of classifiers with high performance on unseen data while using as least non-artificial training data as possible. To measure the performance of the different AL setups, we follow the recommendations found in [9]. The performance of an AL setup will be compared using two AL-specific performance metrics:

- Area Under the Learning Curve (AULC). It is the sum of the classification performance over a validation/test set of the classifiers trained of all AL iterations. To facilitate the interpretability of this metric, the resulting AULC scores are fixed within the range $[0, 1]$ by dividing the AULC scores by the total amount of iterations (*i.e.*, the maximum performance area).

- Data Utilization Rate (DUR) [10]. Measures the percentage of training data required to reach a given performance threshold, as a ratio of the percentage of training data required by the baseline

framework. This metric is also presented as a percentage of the total amount of training data, without making it relative to the baseline framework. The DUR metric is measured at 45 different performance thresholds, ranging between $[0.10, 1.00]$ at a 0.02 step.

5.4 Experimental Procedure

The evaluation of different active learners in a live setting is generally expensive, time-consuming and prone to human error. Instead, a common practice is to perform their evaluation in an offline environment using labeled datasets [11]. In this scenario, since the dataset is already labeled, the annotation process is done at zero cost. Figure 2 depicts the experiment designed for one dataset over a single run.

...

This processed is averaged over 3 runs using different random seeds

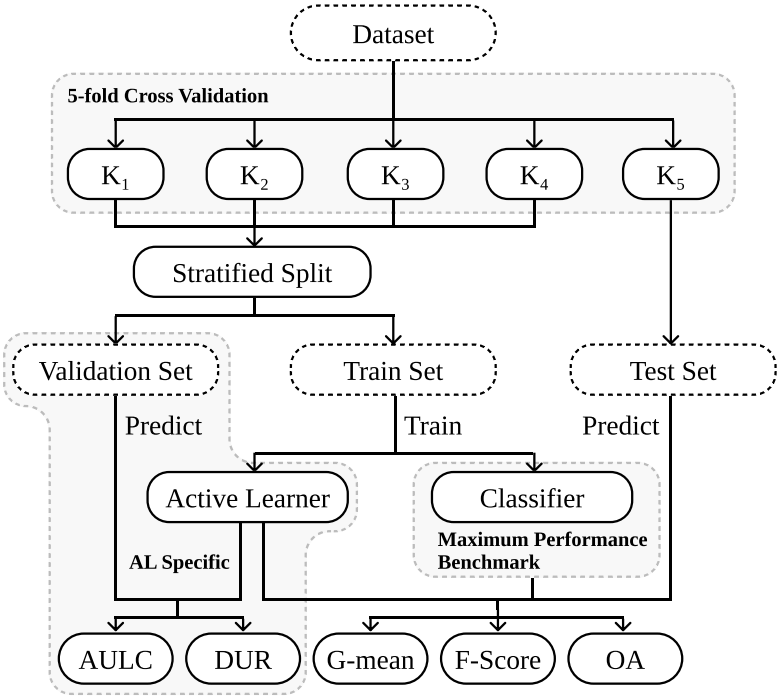

Figure 2: Experimental procedure flowchart.

5.5 Software Implementation

The experiment was implemented using the Python programming language, along with the Python libraries Scikit-Learn [12], Imbalanced-Learn [13], Geometric-SMOTE [14], Research-Learn and ML-

Research libraries. All functions, algorithms, experiments and results are provided in the GitHub repository of the project.

6 Results & Discussion

6.1 Results

Classifier	Evaluation Metric	Standard	G-SMOTE	Proposed
DT	Accuracy	2.50 ± 0.81	2.20 ± 0.4	**1.30 ± 0.64**
DT	F-score	2.50 ± 0.81	2.10 ± 0.3	**1.40 ± 0.8**
DT	G-mean	2.70 ± 0.64	2.00 ± 0.45	**1.30 ± 0.64**
KNN	Accuracy	2.40 ± 0.8	1.90 ± 0.54	**1.70 ± 0.9**
KNN	F-score	2.60 ± 0.66	1.80 ± 0.4	**1.60 ± 0.92**
KNN	G-mean	2.80 ± 0.4	1.70 ± 0.46	**1.50 ± 0.81**
LR	Accuracy	2.60 ± 0.66	2.10 ± 0.54	**1.30 ± 0.64**
LR	F-score	2.80 ± 0.4	2.00 ± 0.45	**1.20 ± 0.6**
LR	G-mean	2.80 ± 0.4	2.00 ± 0.45	**1.20 ± 0.6**
RF	Accuracy	2.60 ± 0.66	1.90 ± 0.54	**1.50 ± 0.81**
RF	F-score	2.60 ± 0.66	2.00 ± 0.45	**1.40 ± 0.8**
RF	G-mean	2.80 ± 0.4	**1.60 ± 0.49**	**1.60 ± 0.8**

Table 2: Mean rankings of the AULC metric over the different datasets (7), folds (5) and runs (3) used in the experiment. This means that the use of G-SMOTE almost always improves the results of the original framework.

Classifier	Evaluation Metric	Standard	G-SMOTE	Proposed
DT	Accuracy	0.733 ± 0.092	0.732 ± 0.087	**0.740 ± 0.087**
DT	F-score	0.695 ± 0.088	0.698 ± 0.09	**0.705 ± 0.092**
DT	G-mean	0.804 ± 0.065	0.811 ± 0.06	**0.816 ± 0.062**
KNN	Accuracy	0.816 ± 0.091	0.818 ± 0.088	**0.822 ± 0.091**
KNN	F-score	0.775 ± 0.102	0.784 ± 0.108	**0.788 ± 0.111**
KNN	G-mean	0.852 ± 0.084	0.866 ± 0.072	**0.869 ± 0.074**
LR	Accuracy	0.802 ± 0.091	0.812 ± 0.088	**0.821 ± 0.086**
LR	F-score	0.749 ± 0.112	0.773 ± 0.116	**0.784 ± 0.115**
LR	G-mean	0.839 ± 0.093	0.870 ± 0.065	**0.875 ± 0.064**
RF	Accuracy	0.861 ± 0.076	0.861 ± 0.075	**0.862 ± 0.077**
RF	F-score	0.823 ± 0.105	0.827 ± 0.105	**0.829 ± 0.105**
RF	G-mean	0.886 ± 0.077	**0.895 ± 0.063**	**0.895 ± 0.065**

Table 3: Average AULC of each AL configuration tested. Each AULC score is calculated using the G-mean scores of each iteration in the validation set. By the end of the iterative process, each AL configuration used a total of 750 instances of the 960 instances that compose the training set.

G-mean Score	Classifier	Standard	G-SMOTE	Proposed
0.60	DT	3.2%	**3.1%**	3.2%
0.60	KNN	3.6%	2.6%	**2.5%**
0.60	LR	3.9%	**2.2%**	**2.2%**
0.60	RF	2.4%	**2.1%**	**2.1%**
0.66	DT	4.6%	4.6%	**4.2%**
0.66	KNN	4.9%	3.7%	**3.5%**
0.66	LR	5.7%	3.2%	**3.1%**
0.66	RF	3.0%	2.8%	**2.7%**
0.70	DT	6.6%	6.1%	**5.8%**
0.70	KNN	8.5%	5.0%	**4.7%**
0.70	LR	9.5%	4.6%	**4.3%**
0.70	RF	4.5%	**3.2%**	3.3%
0.76	DT	16.5%	13.0%	**12.7%**
0.76	KNN	17.8%	9.7%	**9.0%**
0.76	LR	16.6%	10.0%	**7.8%**
0.76	RF	10.1%	**5.5%**	**5.5%**
0.80	DT	36.1%	30.4%	**27.1%**
0.80	KNN	22.7%	18.0%	**17.8%**
0.80	LR	25.2%	16.0%	**14.2%**
0.80	RF	15.5%	**9.0%**	9.5%
0.86	DT	60.5%	56.7%	**54.5%**
0.86	KNN	39.9%	**37.0%**	37.8%
0.86	LR	32.6%	27.5%	**27.0%**
0.86	RF	28.0%	**25.7%**	**25.7%**
0.90	DT	72.5%	70.7%	**67.8%**
0.90	KNN	49.9%	50.3%	**49.3%**
0.90	LR	52.5%	53.8%	**49.3%**
0.90	RF	44.6%	**42.6%**	43.5%
0.96	DT	100.0%	**99.5%**	100.0%
0.96	KNN	79.4%	75.6%	**71.6%**
0.96	LR	87.5%	83.1%	**79.8%**
0.96	RF	63.6%	64.2%	**63.1%**

Table 4: Mean data utilization of AL algorithms, as a percentage of the training set.

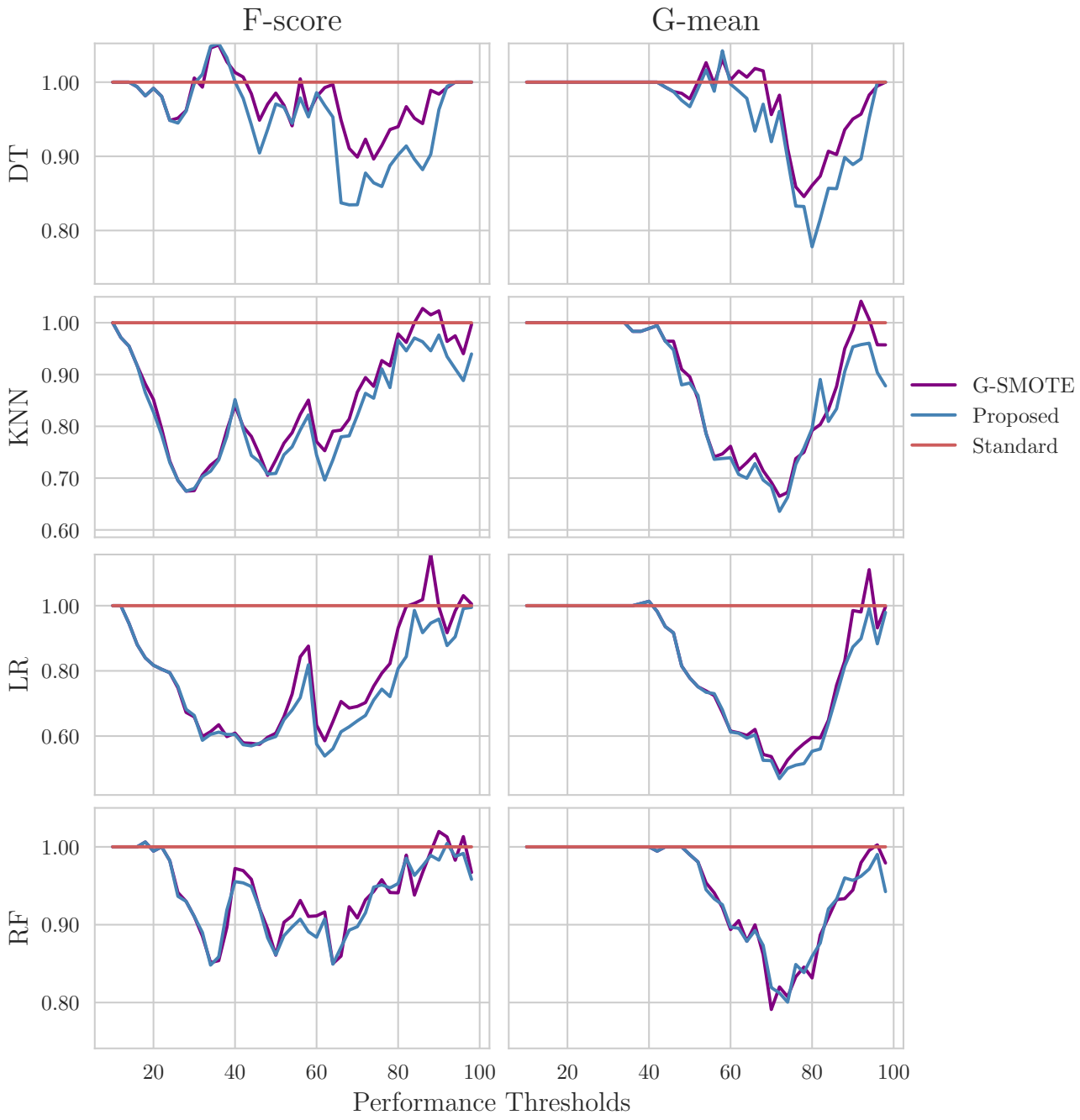

Figure 3: Mean data utilization rates. The y-axis shows the percentage of data (relative to the baseline AL framework) required to reach the different performance thresholds.

Classifier	Evaluation Metric	MP	Standard	G-SMOTE	Proposed
DT	Accuracy	0.809 ± 0.086	0.802 ± 0.089	0.806 ± 0.089	**0.812 ± 0.087**
DT	F-score	0.774 ± 0.107	0.772 ± 0.096	0.775 ± 0.101	**0.781 ± 0.103**
DT	G-mean	0.853 ± 0.081	0.854 ± 0.069	0.860 ± 0.067	**0.864 ± 0.068**
KNN	Accuracy	0.882 ± 0.085	**0.883 ± 0.087**	0.877 ± 0.087	0.881 ± 0.093
KNN	F-score	0.848 ± 0.116	0.849 ± 0.115	0.847 ± 0.118	**0.852 ± 0.121**
KNN	G-mean	0.896 ± 0.094	0.899 ± 0.09	0.904 ± 0.078	**0.907 ± 0.08**
LR	Accuracy	0.855 ± 0.074	**0.870 ± 0.073**	0.858 ± 0.077	**0.870 ± 0.076**
LR	F-score	0.812 ± 0.113	0.835 ± 0.105	0.825 ± 0.106	**0.838 ± 0.106**
LR	G-mean	0.875 ± 0.099	0.895 ± 0.075	0.899 ± 0.059	**0.907 ± 0.059**
RF	Accuracy	0.897 ± 0.08	0.905 ± 0.078	0.904 ± 0.078	**0.906 ± 0.077**
RF	F-score	0.867 ± 0.107	**0.877 ± 0.103**	0.875 ± 0.108	**0.877 ± 0.108**
RF	G-mean	0.911 ± 0.081	0.917 ± 0.078	0.923 ± 0.067	**0.925 ± 0.065**

Table 5: Optimal classification scores. The Maximum Performance (MP) classification scores are calculated using classifiers trained using the entire training set.

6.2 Statistical Analysis

Dataset	p-value	Significance
Baseball	2.6e-02	True
Gas Drift	2.0e-06	True
Image Segmentation	3.8e-05	True
Japanese Vowels	2.1e-02	True
Mfeat Zernike	9.5e-04	True
Mice Protein	1.5e-08	True
Pendigits	5.3e-05	True
Texture	5.4e-06	True
Vehicle	9.2e-03	True
Waveform	4.3e-03	True

Table 6: Adjusted p-values using the Wilcoxon signed-rank method. Bold values are statistically significant at a level of $\alpha = 0.05$. The null hypothesis is that the performance of the proposed framework is similar to that of the original framework.

6.3 Discussion

7 Conclusion

References

[1] C. Wu, *The decision tree approach to classification.* Purdue University, 1975.

[2] T. Cover and P. Hart, "Nearest neighbor pattern classification," *IEEE Transactions on Information Theory*, vol. 13, no. 1, pp. 21–27, 1967.

[3] T. K. Ho, "Random decision forests," in *Proceedings of the Third International Conference on Document Analysis and Recognition (Volume 1) - Volume 1*, ICDAR '95, (USA), p. 278, IEEE Computer Society, 1995.

[4] J. A. Nelder and R. W. Wedderburn, "Generalized linear models," *Journal of the Royal Statistical Society: Series A (General)*, vol. 135, no. 3, pp. 370–384, 1972.

[5] J. Fonseca, G. Douzas, and F. Bacao, "Increasing the Effectiveness of Active Learning: Introducing Artificial Data Generation in Active Learning for Land Use/Land Cover Classification," *Remote Sensing 2021, Vol. 13, Page 2619*, vol. 13, p. 2619, jul 2021.

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

[7] M. Fatourechi, R. K. Ward, S. G. Mason, J. Huggins, A. Schloegl, and G. E. Birch, "Comparison of evaluation metrics in classification applications with imbalanced datasets," in *2008 seventh international conference on machine learning and applications*, pp. 777–782, IEEE, 2008.

[8] M. Kubat, S. Matwin, *et al.*, "Addressing the curse of imbalanced training sets: one-sided selection," in *Icml*, vol. 97, pp. 179–186, Citeseer, 1997.

[9] D. Kottke, A. Calma, D. Huseljic, G. Krempl, and B. Sick, "Challenges of reliable, realistic and comparable active learning evaluation," in *CEUR Workshop Proceedings*, vol. 1924, pp. 2–14, sep 2017.

[10] T. Reitmaier and B. Sick, "Let us know your decision: Pool-based active training of a generative classifier with the selection strategy 4ds," *Information Sciences*, vol. 230, pp. 106–131, 5 2013.

[11] J.-F. Kagy, T. Kayadelen, J. Ma, A. Rostamizadeh, and J. Strnadova, "The practical challenges of active learning: Lessons learned from live experimentation," *arXiv preprint arXiv:1907.00038*, 6 2019.

[12] 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 É. Duchesnay, "Scikit-learn: Machine Learning in Python," *Journal of Machine Learning Research*, vol. 12, no. Oct, pp. 2825–2830, 2011.

[13] 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*, vol. 18, no. 17, pp. 1–5, 2017.

[14] G. Douzas and F. Bacao, "Geometric SMOTE a geometrically enhanced drop-in replacement for SMOTE," *Information Sciences*, vol. 501, pp. 118–135, Oct. 2019.