Exploiting Prototypical Explanations for Undersampling Imbalanced Datasets

Yusuf Arslan
University of Luxembourg
Luxembourg, Luxembourg
0000-0003-4423-4725

Kevin Allix
University of Luxembourg
Luxembourg, Luxembourg
kevin.allix@uni.lu

Clément Lefebvre
BGL BNP Paribas
Luxembourg, Luxembourg
clement.c.lefebvre@bgl.lu

Andrey Boytsov
BGL BNP Paribas
Luxembourg, Luxembourg
andrey.boytsov@bgl.lu

Tegawendé F. Bissyandé
University of Luxembourg
Luxembourg, Luxembourg
tegawende.bissyande@uni.lu

Jacques Klein
University of Luxembourg
Luxembourg, Luxembourg
jacques.klein@uni.lu

***Abstract*—Among the reported solutions to the class imbalance issue, the undersampling approaches, which remove instances of insignificant samples from the majority class, are quite prevalent. However, the undersampling approaches may discard significant patterns in the datasets. A prototype, which is always an actual sample from the data, represents a group of samples in the dataset. Our hypothesis is that prototypes can fill the missing significant patterns that are discarded by undersampling methods and help to improve model performance. To confirm our intuition, we articulate prototypes to undersampling methods in the machine learning pipeline. We show that there is a statistically significant difference between the AUPR and AUROC results of undersampling methods and our approach.**

***Index Terms*—Prototypical Explanations, Undersampling, Imbalanced Datasets**

I. INTRODUCTION

Continuous digitalization has brought more and more data for processing. These digitalized data use machine learning (ML) models for various tasks, including classification. However, the classification performance of the model decreases in case of incorrect data representation [1]. One of the main reasons of incorrect data representation is the imbalance between classes [2]. When there is an imbalance between the classes, classification models often misclassify rare instances due to their bias toward majority patterns [3]. Among the reported solutions [4]–[8], the undersampling approach, which removes instances of insignificant majorities, is a quite prevalent solution to the class imbalance issue [3]. In practical settings, an undersampling procedure is applied to the data to decrease the training time and cost of the models by removing samples from the majority class, whereas it is important to select the representative samples in the undersampling procedure to have a meaningful representation of the training data. However, one of the downsides of undersampling is that significant patterns can be in removed samples [9], and removal of them causes decreases in classification performance. Prototypes [10], [11], which are always `actual samples` from the data [12], constitute prototypical explanations. Prototypical explanations summarize the datasets and can help to improve classification model performance by filling the discarded significant patterns during undersampling procedure.

This paper. This study considers the problem of undersampling of imbalanced datasets in the ML classification pipeline.

We assume that prototypical explanations select significant representative samples of the dataset as prototypes. Thus an undersampler performance can be improved with the help of prototypical explanations, which can include the discarded significant samples during the undersampling procedure to the training set. Concretely, we investigate the potential of prototypical explanations to be used together with undersamplers in the ML classification pipeline with imbalanced datasets to improve the classification performance. In our analysis, we propose to evaluate the added value of prototypical explanations to the classification performance. To that end, we explore the performance of classification models before and after the addition of prototypical explanations to the training set, which is sampled by undersampling methods.

Our study explores the following research questions:

RQ1. Are prototypes usable for undersampling?

RQ2. Does adding prototypes improve the performance of each undersampling method?

RQ3. Can prototypes improve the performance of classifiers?

Overall, we show that unique prototype samples[1] can constitute up to 6% of the training set. Moreover, it is possible to improve the AUROC performance of the models up to 45%. There is a statistically significant difference between the AUPR and AUROC results of undersampling methods and our approach.

Our key findings are:

This work is supported by the Luxembourg National Research Fund (FNR) under the projects ExLiFT (13778825) and LuxemBERT (16229163).

[1]Prototypes and prototype samples are used interchangeable in this paper.

978-1-6654-6283-9/22/$31.00 ©2022 IEEE
DOI 10.1109/ICMLA55696.2022.00228

1) Prototype samples can fill the missing significant patterns, which are discarded by undersampling methods, via actual samples from the dataset.

2) It is possible to improve the performance of the classification model (in terms of AUPR and AUROC metrics) by using prototypical explanations together with undersampling methods.

3) There is a statistically significant difference between classification results of undersampling methods and prototypical explanations articulated undersampling methods.

4) This findings can be especially important when an undersampling method does not fit the training data and classification model has bad performance.

II. BACKGROUND AND RELATED WORK

Class imbalance problem occurs when the distribution of the data is skewed between classes [3]. There are three types of available solutions to the class imbalance problem [13]:

1) Data level solutions (Undersampling / Oversampling): These approaches aim to obtain a balanced class distribution by resampling datasets. Resampling has been investigated for a long time [4]–[8]. Undersampling removes samples from the majority class. Oversampling adds more samples to the minority class by either replicating existing samples or generating new samples from existing samples.
2) Algorithmic level solutions (Cost sensitive learning / Learning function modification): These approaches aim to make either available algorithms suitable for class imbalance issues or to generate new algorithms [2], [14].
3) Ensemble learning based solutions (Integration with data sampling methods / Integration with cost sensitive learning): These approaches aim to integrate data level and algorithmic level solutions [14].

The `imbalanced-learn` package [15] is one of the most extensive Python toolbox for undersampling. The package is widely used in resampling studies [2], [14], [16]. It contains eleven undersampling methods, which do not generate new samples, namely, nearmiss (with three versions), random under sampler (RUS), edited nearest neighbours (ENN), allknn, neighbourhood cleaning rule (NCR), condensed nearest neighbour (CNN), one sided selection (OSS), instance hardness threshold (IHT), and tomek links. Each undersampling method aims balance the data with a different approach and works better in different cases. As a result, there is no superior method between them [16].

In this study, we focus on undersampling as one of the prevalent solutions to the class imbalance issue [3]. We do not use any methods that generate synthetic samples like oversampling. In fact, all of the samples in our experiments are the actual samples from the dataset.

A. ProtoDASH

ProtoDASH is an implementation of prototypical explanation technique. It is a fast prototype selection method [11]. It is implemented under IBM interpretable AI package AIX360[2]. [10] and [11] use 1-Nearest Neighbour(NN) Classifier to evaluate the performance of prototype selection methods. NN classifier is used as an indirect method to evaluate the performance of ProtoDASH, while our work differs by not evaluating ProtoDASH but employing it to improve classification model performance. Our use of ProtoDASH is in line with the growing literature of the articulation of explainable AI techniques to ML pipeline [17], [18].

B. Prototypical Explanations

Prototypical explanations represent the condensed view of a dataset via minimal subset of significant samples from the dataset [12]. Each of these selected significant samples is called as "prototype". Longstanding studies on the effect of prototypes on human-decision making show that prototypes are crucial for the development of decision-making strategies [19], [20]. Hence prototypical explanations are deployed in the context of explainable machine learning [10] and can be categorized under example-based explanation techniques [12].

A desirable prototype set for class y should have the following properties [21]:

- covers as much as possible training points of class y
- includes as few as possible training points from other classes
- is sparse (i.e., uses as few prototypes as possible)

Prototype selection requires to know how well the distribution of a dataset is represented by the sample of another distribution. Maximum Mean Discrepancy (MMD) measures the distance between two distributions so that it can be determined whether the distribution of the prototypes is close to the data distribution or not. Let P and Q are two distributions, MMD can be defined as:

$$MMD(P,Q) = ||\mu(P) - \mu(Q)||_H$$

where H is a reproducing kernel Hilbert space and $\mu(P)$ $\mu(Q)$ are kernel embeddings of P and Q respectively [22].

III. EXPERIMENTAL SETUP

A. Experiment Process

The experimental setup of this study can be seen in Fig. 1. Our goal is to study whether prototypes, which are obtained by ProtoDash, are missing in the output of undersampling methods. If prototypes are missing in the training set after the undersampling, the second goal is to assess the impact of addition of missing prototype samples into the training set. The final goal is to compare the classifier performances of undersamplers and our approach on the test set.

Training Phase and Machine Learning Algorithms: The training phase and the used algorithms are presented in Fig. 1(a) and Fig. 1(b).

In the `imbalanced-learn` package, each undersampling method aims to balance the data, but each of them takes different approaches to achieve this. Hence, each undersampling

[2]https://github.com/Trusted-AI/AIX360

method works better in different cases [16] and each dataset requires different undersampling method. In this study, we report the results of all undersampling methods for each dataset instead of reporting only the best undersampling method. The reason of reporting all undersampling methods for each dataset is to understand whether prototypes are especially useful for certain undersampling methods or not. Indeed, we inspect the results not only concerning datasets but also regarding undersampling methods. Consequently, we train thirty-three models in Fig. 1(a) by using eleven undersamplers and three classifiers.

In our approach, we add the prototype samples to the set of samples selected by undersamplers as can be seen in Fig. 1(b). We train thirty-three classifiers that use eleven prototypical explanations articulated undersamplers and three classifiers. Undersampling methods preserve the balance between classes, while prototypes do not. This is the reason of adding prototypes to the set of samples selected by undersamplers, but not using them as a standalone undersampling method. Indeed, it is conceivable to use prototypes as a standalone undersampling technique for large datasets, but such a use is out of the scope of the present study.

Each dataset has a certain amount of prototypes. The number of prototypes depends on the dataset. In our experiments, we first find the number of available prototypes for each dataset. Then we calculate the percentage of the *unique prototypes*, which are not already selected by the undersampling method, in the training set.

Testing Phase: On the testing phase, we test sixty-six classification models. Thirty-three of them are generated via eleven undersamplers and three classifiers. The remaining thirty-three of them are generated via eleven prototypical explanations articulated undersamplers and three classifiers. We compare the performances of these classification models in terms of AUPR and AUROC metrics as can be seen in Fig. 1(c). All our experiments are performed using 5-Fold cross-validation and are repeated 5 times. The averaged results are then reported. Moreover, we check the statistical significance of the results by Wilcoxon Signed Rank Test.

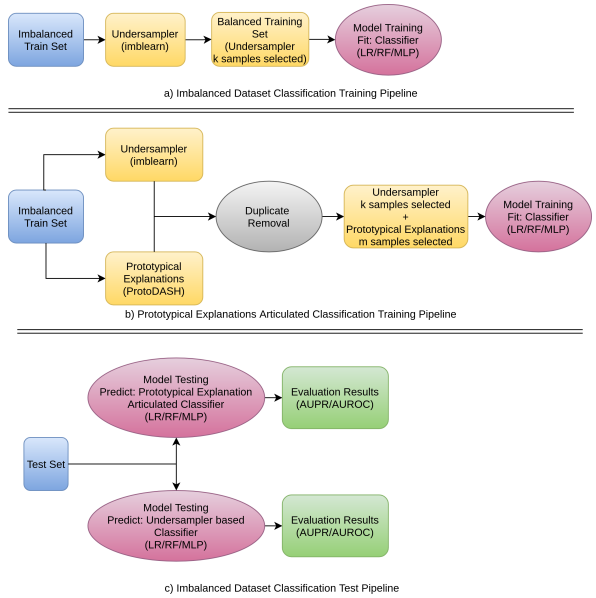

Fig. 1. Imbalanced Dataset Classification Pipeline

B. Evaluation Metrics

In this paper, we are using the following metrics and tests:

The Area Under the ROC Curve (AUROC): True-Positive Rate (TPR) is the proportion of correctly classified positive samples [23]. It shows the performance of models in the prediction of the positive class when the actual outcome is positive. False-Positive Rate (FPR) is the proportion of incorrectly classified negative samples [23]. It shows the number of positive classifications while the actual outcome is negative. The ROC Curves are plotted with TPR against FPR where TPR is plotted along the y-axis and FPR is plotted along the x-axis. AUROC is the de facto standard to evaluate classifiers under imbalance [24]. The reason is that it is independent of the selected threshold and prior probabilities. Besides, AUROC proposes a single number to compare classifiers [25].

The Area Under the PR Curve (AUPR): Precision computes the proportion of samples classified as positive that are truly positive [26]. Recall is same as TPR. PR curve shows the trade-off between precision and recall for different thresholds. AUPR evaluates output quality of a classifier. It is used especially in case of class imbalance. High precision implies a low false-positive rate and a high recall implies low false negative rate. ROC curves are suitable for balanced datasets, whereas PR curves are suitable for imbalanced datasets [27].

Wilcoxon Signed Rank Test: It is a nonparametric (distribution free) statistical test to compare the data [28]. We use this test to compare the effects of the prototype samples on classification models. Significance levels are set at the 0.05. The null hypothesis is failed to reject if p-value is greater than 0.05. The null hypothesis is rejected at a confidence level of 95% if p-value is less than 0.05.

C. Empirical Datasets

We perform our experiments by relying on seven publicly available binary classification datasets, namely, *Adult*[3], *Attrition*[4], *Churn*[5], *German Credit*[6], *Stroke Prediction*[7], *Ulb Fraud*[8], and *Wilt*[9]. The datasets' statistics can be seen in Table I.

[3]https://archive.ics.uci.edu/ml/datasets/adult
[4]https://data.world/aaizemberg/hr-employee-attrition
[5]https://www.openml.org/search?type=data&sort=runs&id=40701
[6]https://archive.ics.uci.edu/ml/datasets/statlog+(german+credit+data)
[7]https://www.kaggle.com/datasets/fedesoriano/stroke-prediction-dataset
[8]https://github.com/Fraud-Detection-Handbook
[9]https://archive.ics.uci.edu/ml/datasets/Wilt

TABLE I
DATASETS' STATISTICS. SIZE IS NUMBER OF SAMPLES. FEATURE IS C:CATEGORICAL, N:NUMERIC. #MIN/#MAJ IS THE SIZE OF THE MINORITY AND MAJORITY CLASS. IR IS THE RATIO OF IMBALANCE IN THE DATASET.

Dataset	Size	Features	#Min/#Maj	IR
Adult	32 561	4 n. / 8 c.	7841 / 24 720	3.15
Attrition	1470	27 n. / 8.c	237 / 1233	5.20
Churn	5000	5 n. / 16 c.	707 / 4293	6.07
German Credit	1000	7 n. / 13 c.	300 / 700	2.33
Stroke	5110	3 n. / 8 c.	249 / 4861	19.52
Ulb Fraud	19 084	10 n. / 2 c.	187 / 18 897	101.05
Wilt	4839	6 n.	261 / 4578	17.54

The *Adult* dataset, which is also known as "Census Income", contains 32 561 samples with 12 categorical and numerical features. The prediction task of the dataset is to find out whether a person makes more than $50K per year or not.

The *Attrition* dataset contains fictional data regarding employees. It has 1470 samples with 35 categorical and numerical features. The task is to predict the attrition of employees.

The *Churn* dataset contains accounts of telephone company customers. It has 5000 samples with 21 categorical and numerical features. The task is to predict churn of customers.

The *German Credit* dataset contains loan applications. It has 1000 samples with 20 categorical and numerical features. The task is to classify people with respect to credit risks.

The *Stroke Prediction*dataset contains electronic health records of patients. It has 5110 samples with 11 categorical and numerical features. The task is to predict stroke events.

The *Ulb Fraud*contains simulated transactions. We generate transactions for two days period. It has 19 084 samples with 12 categorical and numerical features. The prediction task of the dataset is to identify fraudulent transactions.

The *Wilt*dataset contains high-resolution remote sensing data. It has 4839 samples with 6 numerical features. The prediction task is to detect diseased trees.

D. Research Questions

RQ1. Are prototypes usable for undersampling?

To answer RQ1, we split it into two sub-questions as follows.

- *RQ1.1* How many prototypes are extracted in our datasets?
- *RQ1.2* Are those prototypes different from the original samples? (i.e., do prototypes bring something that undersampling cannot?)

The aim of the first research question is to understand whether prototypes are usable for undersampling or not. The aim of the first sub-research question is to identify the number of prototype samples in the datasets. The aim of the second sub-research question is to answer whether these prototype samples have a meaningful contribution to the training set or not, in other words, whether prototype samples bring something that undersampling cannot.

RQ2. Does adding prototypes improve the performance of each undersampling method? The aim of the second research question is to measure the improvement in the performance of each undersampling method after the addition of prototype samples. We evaluate the performance of undersampling methods before and after the addition of prototype samples.

RQ3. Can prototypes improve the performance of classifiers? The aim of the third research question is to detect the performance improvement of classification models after the addition of prototype samples. We evaluate the performance of three classification methods from three types of classifiers, namely, Logistic Regression (LR) from linear classifiers, Random Forest (RF) from ensemble classifiers, and Multi Layer Perceptron (MLP) from nonlinear classifiers before and after the addition of prototype samples.

IV. EMPIRICAL RESULTS

Answer of RQ1: To understand the usefulness of prototypes for undersampling, we split RQ1 into two sub-research questions and obtain the following answers.

Answer of RQ1.1: In this sub-research question, we start by checking the number of prototype samples in the training set as can be seen in Table II.

The number of prototype samples that are needed to get a condensed view of the training set by significant representative samples are obtained by ProtoDash and can be seen in Table II. Relatively low number of samples are selected to summarize the training set (as prototype samples). On the other hand, undersampling methods preserve the balance between classes and select more samples with respect to prototype selection methods. Therefore, prototype samples may already be selected by undersampling method. We inspect the percentage of unique prototype samples, which are not already selected by undersampler methods, in the next research question.

Answer of RQ1.2: In this sub-research question, we inspect the percentage of *unique prototype samples* that are not already selected by undersampling methods as can be seen in Table III.

This sub-research question shows that *unique prototype samples* can constitute up to 6% of the training set. This finding is in line with our assumption and thus some of the significant samples from training set are discarded by the undersampling methods. Indeed, it can be deduced that prototypes can help to form a better training set since prototype samples that are not selected by undersamplers can fill *the missing significant patterns* in training set. The next stage is to understand the effect of prototype samples in the model performance.

Answer of RQ2: In this research question, we evaluate the performance improvement of each undersampling method before and after the addition of prototype samples via AUPR and AUROC metrics.

Table IV shows the performance improvement of eleven undersamplers for each classification model after the addition of prototypes in terms of AUPR and AUROC metrics.

According to Table IV, it is possible to improve the AUROC performance of the models up to 45%, while the use of prototypes does not decrease the performance more than 1%. The biggest performance improvement is obtained for Nearmiss1 undersampling method. Our inspection reveals that Nearmiss1

TABLE II
NUMBER OF PROTOTYPE SAMPLES SUMMARIZING TRAINING SET

Datasets	Adult	Attrition	Churn	German Credit	Stroke	Ulb Fraud	Wilt
Prototype Samples	33	19	27	22	17	24	7

TABLE III
PERCENTAGE OF UNIQUE PROTOTYPE SAMPLES (NOT ALREADY SELECTED BY THE UNDERSAMPLERS) IN THE TRAINING SET

Datasets Undersampler	Adult	Attrition	Churn	German Credit	Stroke	Ulb Fraud	Wilt
CNN	0.17	2.17	1.29	2.54	1.96	2.68	0.84
ENN	0.06	0.87	0.18	1.87	0.11	0.12	0.04
AllKNN	0.06	1.10	0.19	2.28	0.12	0.12	0.04
IHT	0.14	3.00	1.30	2.17	0.44	0.18	0.11
Nearmiss1	0.15	3.05	1.53	2.29	3.47	5.87	1.17
Nearmiss2	0.15	3.26	1.52	2.24	3.40	5.87	1.15
Nearmiss3	0.18	3.29	1.56	2.33	3.50	5.87	1.16
NCR	0.05	0.89	0.22	1.96	0.11	0.12	0.05
OSS	0.03	0.35	0.12	0.73	0.10	0.12	0.04
RUS	0.15	3.07	1.52	2.29	3.44	5.87	1.16
Tomek Links	0.03	0.34	0.11	0.71	0.08	0.12	0.04

TABLE IV
AVERAGE PERFORMANCE IMPROVEMENT OF THE UNDERSAMPLERS AFTER THE ADDITION OF PROTOTYPES (%)

Model (Evaluation Metric) Undersampler	LR (AUPR)	LR (AUROC)	RF (AUPR)	RF (AUROC)	MLP (AUPR)	MLP (AUROC)
CNN	0.18	0.26	0.31	0.02	1.53	1.01
ENN	0.37	0.19	-0.15	0.15	0.52	0.72
AllKNN	0.42	0.04	0.23	-0.02	-0.52	-0.40
IHT	0.27	0.04	0.12	0.02	1.59	0.21
Nearmiss1	2.35	45.22	1.99	33.19	3.61	42.62
Nearmiss2	1.09	0.81	2.65	2.79	2.94	3.48
Nearmiss3	0.51	0.52	0.41	0.83	2.06	0.84
NCR	-0.19	-0.06	0.19	-0.12	-0.17	-0.61
OSS	0.26	0.07	-0.22	-0.19	-0.30	-0.67
RUS	0.85	0.08	0.31	0.17	1.43	-0.12
Tomek Links	0.26	-0.02	0.11	-0.24	-0.32	-0.14

performs worse than the other undersampling methods. However, Nearmiss1 performs as well as other undersamplers after the addition of prototype samples. This finding suggests that prototype samples can be helpful when undersampler does not fit the training data.

Answer of RQ3: In this research question, we evaluate the performance improvement of three classification methods, namely, LR, RF, and MLP from three types of classifiers before and after the addition of prototype samples by AUPR and AUROC metrics.

Table V shows the performance improvement of three different kinds of classification models for each dataset after the addition of prototypes in terms of AUPR and AUROC metrics.

According to Table V, it is possible to improve the classification model performance up to 2% in terms of AUPR and 7% in terms of AUROC. The only performance decrease after the addition of prototypes happened on AUPR metric of MLP model for Stroke dataset with -0.31%.

In our experiments, we perform six Wilcoxon superiority tests, which are reported here for LR (AUPR), LR (AUROC), RF (AUPR), RF (AUROC), MLP (AUPR), and MLP (AUROC), respectively: 0.003, 0.005, 0.003, 0.000, 0.014, 0.015. Hence, we could reject the null hypothesis at a confidence level of 5%, concluding that there is a statistically significant difference between the AUPR and AUROC results of undersampling methods and our approach.

A. Threats to Validity and Limitations

In our experiments, we train sixty-six classifiers from three types of classification models with default parameters and without hyperparameter tuning. We avoid hyperparameter tuning for two reasons. First, it can lead to overfitting in train set [29]. Second, hyperparameter tuning does not guarantee fair evaluation [30]. Therefore, we prefer to use the classifiers with the default parameters and evaluate the results of them on various datasets with slight to severe class imbalances.

One of the limitations of our study is that ProtoDASH does not work for large datasets, especially in industrial data, because of memory issues. Therefore, we utilize datasets that ProtoDASH can handle in the context of this study.

V. CONCLUSION AND FUTURE WORK

In this study, we inspect the performance of classification models before and after the addition of prototypes to the

TABLE V
AVERAGE PERFORMANCE IMPROVEMENT OF MODELS AFTER THE ADDITION OF PROTOTYPES (%)

Datasets Model (Evaluation Metric)	Adult	Attrition	Churn	German Credit	Stroke	Ulb Fraud	Wilt
LR (AUPR)	0.16	1.23	0.09	0.39	0.41	0.83	0.94
RF (AUPR)	0.14	0.30	0.45	0.24	0.51	0.55	1.59
MLP (AUPR)	1.53	0.74	1.49	0.58	-0.31	1.85	2.00
LR (AUROC)	2.27	4.20	3.83	2.54	6.27	4.19	6.71
RF (AUROC)	1.93	3.80	1.63	1.62	6.56	5.82	1.94
MLP (AUROC)	3.57	3.45	3.99	2.70	4.40	4.97	6.78

undersampled training sets. According to our findings, unique prototype samples, which are not already selected by undersampling methods, can constitute up to 6% of the training set. Moreover, it is possible to improve the AUROC performance of the models up to 45%. We show that there is a statistically significant difference between the AUPR and AUROC results of undersampling methods and our approach. Our approach contributes to the growing literature of the articulation of explainable AI approaches to ML pipelines.

As a future work, we would like to compare the performance of our approach with respect to oversampling and ensemble techniques. Besides, we highlight other interesting future research directions as follows: First, the training set can be enriched by the output of other explanation techniques like SHAP. Second, it is impractical to obtain prototypical explanations from big datasets because of memory issue and the selection of prototypical explanations from big datasets is worth to investigate.

REFERENCES

[1] Y. Park, J. Qing, X. Shen, and B. Mozafari, "Blinkml: Approximate machine learning with probabilistic guarantees," in *Proceedings of the 45th International Conference on Very Large Data Bases, Los Angeles, CA, USA*, 2018, pp. 1–18.

[2] A. Kulkarni, D. Chong, and F. A. Batarseh, "Foundations of data imbalance and solutions for a data democracy," in *data democracy*. Elsevier, 2020, pp. 83–106.

[3] D. Devi, S. K. Biswas, and B. Purkayastha, "A review on solution to class imbalance problem: Undersampling approaches," in *2020 ComPE*. IEEE, 2020, pp. 626–631.

[4] A. C. Davison and D. V. Hinkley, *Bootstrap methods and their application*. Cambridge university press, 1997, no. 1.

[5] P. I. Good, *Resampling methods*. Springer, 2006.

[6] M. Kuhn and K. Johnson, *Applied predictive modeling*. Springer, 2013, vol. 26.

[7] T. Sasada, Z. Liu, T. Baba, K. Hatano, and Y. Kimura, "A resampling method for imbalanced datasets considering noise and overlap," *Procedia Computer Science*, vol. 176, pp. 420–429, 2020.

[8] T. K. Dang, T. C. Tran, L. M. Tuan, and M. V. Tiep, "Machine learning based on resampling approaches and deep reinforcement learning for credit card fraud detection systems," *Applied Sciences*, vol. 11, no. 21, p. 10004, 2021.

[9] X.-Y. Liu, J. Wu, and Z.-H. Zhou, "Exploratory undersampling for class-imbalance learning," *IEEE Transactions on Systems, Man, and Cybernetics, Part B*, vol. 39, no. 2, pp. 539–550, 2008.

[10] B. Kim, R. Khanna, and O. O. Koyejo, "Examples are not enough, learn to criticize! criticism for interpretability," in *Advances in Neural Information Processing Systems*, D. Lee, M. Sugiyama, U. Luxburg, I. Guyon, and R. Garnett, Eds., vol. 29. Curran Associates, Inc., 2016.

[11] K. S. Gurumoorthy, A. Dhurandhar, and G. Cecchi, "Protodash: Fast interpretable prototype selection," *arXiv preprint arXiv:1707.01212*, 2017.

[12] C. Molnar, *Interpretable machine learning*. Lulu. com, 2020.

[13] M. Bach, A. Werner, and M. Palt, "The proposal of undersampling method for learning from imbalanced datasets," *Procedia Computer Science*, vol. 159, pp. 125–134, 2019.

[14] A. Fernández, S. García, M. Galar, R. C. Prati, B. Krawczyk, and F. Herrera, *Learning from imbalanced data sets*. Springer, 2018, vol. 10.

[15] G. Lemaître, F. Nogueira, and C. K. Aridas, "Imbalanced-learn: A python toolbox to tackle the curse of imbalanced datasets in machine learning," *The Journal of Machine Learning Research*, vol. 18, no. 1, pp. 559–563, 2017.

[16] S. Lööv, "Comparison of undersampling methods for prediction of casting defects based on process parameters," Master's thesis, University of Skövde, 2021.

[17] Y. Arslan, B. Lebichot, K. Allix, L. Veiber, C. Lefebvre, A. Boytsov, A. Goujon, T. F. D. A. Bissyande, and J. Klein, "On the suitability of shap explanations for refining classifications," in *Proceedings of the 14th International Conference on Agents and Artificial Intelligence (ICAART 2022)*, 2022.

[18] Y. Arslan, B. Lebichot, K. Allix, L. Veiber, C. Lefebvre, A. Boytsov, A. Goujon, T. F. Bissyande, and J. Klein, "Towards refined classifications driven by shap explanations," in *Cross Domain Conference for Machine Learning and Knowledge Extraction (CD-MAKE 2022)*, 2022.

[19] A. Newell, H. A. Simon *et al.*, *Human problem solving*. Prentice-hall Englewood Cliffs, NJ, 1972, vol. 104, no. 9.

[20] M. S. Cohen, J. T. Freeman, and S. Wolf, "Metarecognition in time-stressed decision making: Recognizing, critiquing, and correcting," *Human factors*, vol. 38, no. 2, pp. 206–219, 1996.

[21] J. Bien and R. Tibshirani, "Prototype selection for interpretable classification," *The Annals of Applied Statistics*, vol. 5, no. 4, pp. 2403–2424, 2011.

[22] R. Gao, F. Liu, J. Zhang, B. Han, T. Liu, G. Niu, and M. Sugiyama, "Maximum mean discrepancy test is aware of adversarial attacks," in *International Conference on Machine Learning*. PMLR, 2021, pp. 3564–3575.

[23] P. Flach and M. Kull, "Precision-recall-gain curves: Pr analysis done right," *Advances in neural information processing systems*, vol. 28, 2015.

[24] A. P. Bradley, "The use of the area under the roc curve in the evaluation of machine learning algorithms," *Pattern Recognition*, vol. 30, no. 7, pp. 1145–1159, 1997.

[25] T. R. Hoens and N. V. Chawla, "Imbalanced datasets: from sampling to classifiers," *Imbalanced learning: Foundations, algorithms, and applications*, pp. 43–59, 2013.

[26] J. Davis and M. Goadrich, "The relationship between precision-recall and roc curves," in *Proceedings of the 23rd ICML*, 2006, pp. 233–240.

[27] T. Saito and M. Rehmsmeier, "The precision-recall plot is more informative than the roc plot when evaluating binary classifiers on imbalanced datasets," *PloS one*, vol. 10, no. 3, p. e0118432, 2015.

[28] F. Wilcoxon, "Individual comparisons by ranking methods," in *Breakthroughs in statistics*. Springer, 1992, pp. 196–202.

[29] R. Turner, D. Eriksson, M. McCourt, J. Kiili, E. Laaksonen, Z. Xu, and I. Guyon, "Bayesian optimization is superior to random search for machine learning hyperparameter tuning: Analysis of the black-box optimization challenge 2020," in *NeurIPS 2020 Competition and Demonstration Track*. PMLR, 2021, pp. 3–26.

[30] A. M. F. da Cruz, "Fairness-aware hyperparameter optimization," Master's thesis, Universidade do Porto, 27th July 2020.