Imbalance Learning Ensemble Optimization

Maksim Rafailovich Aliev, Sergey Borisovich Muravyov
National Research University ITMO
St.Petersburg, Russian Federation
imaxaliev@gmail.com, mursmail@gmail.com

***Abstract*—Our world is far from being perfectly balanced, and therefore most real-world data is inherently imbalanced. To effectively work with such data, there are various approaches to imbalanced learning. However, finding the most successful model configuration for a specific dataset is a problem, even for an experienced data scientist. To solve it, there are various approaches to automated machine learning. However, their applicability to imbalanced learning problems is an open question. This paper presents an approach to the optimization search of bagging and boosting ensembles (based on re-weighting and undersampling techniques), and also provides an experimental comparison with the well-known FLAML solution. The benchmark results on data with the imbalance level of moderate and extreme demonstrated worthy competition in f1-measure and an overwhelming advantage in balanced accuracy. The optimization search time was on average several minutes longer for our solution. The proposed approach is implemented as an open source framework and can be found on Github (ImbaML).**

I. Introduction

The problem of imbalanced classification is quite common. Examples include, but not limited to: detection of spam or fraud, prediction of failures in the hardware infrastructure of a data center, medical diagnostics of rare diseases, identification of hacking attempts to an Internet resource by intruders, modeling of insurance risks, prediction of environmental disasters. The proportion of the positive class (*PCR*) for the case of binary classification is expressed by the formula:

$$PCR = \frac{Total\ Number\ Of\ Positive\ Class\ Instances}{Total\ Number\ Of\ Instances} \quad (1)$$

For the case of balanced classification $PCR \approx 0.5$. For imbalanced $PCR \leq 0.4$. If $0.2 \leq PCR \leq 0.4$, then the level of imbalance is mild. If $0.01 \leq PCR \leq 0.2$, then the level of imbalance is moderate. If $PCR \leq 0.01$, then the level of imbalance is extreme. In the case of mild imbalance, it is still possible to rely on classical machine learning approaches, but for moderate and extreme imbalance they are no longer as effective.

To effectively solve this problem, methods based on *resampling* and *reweighting* are used. *Resampling* is a change in the proportion of class instances by adding or removing instances. *Undersampling* is removing instances of the dominant class. While *oversampling* is adding instances of the minor class. There are many different implementations of each of the methods, including random generation or removal, interpolation, or the nearest neighbor method. *Reweighting*, in turn, is a technique for changing the weight of a class instance. Common scenario is *upweighting*, where the weight value of a given class is multiplied by a certain value. In most cases, reweighting techniques are implemented in ensemble methods(boosting and bagging).

Metrics that are robust to imbalances in the proportion of class instances are used as measures of assessing the predictive ability of the model, such as: f1-measure, AUC-PR(area under precision and recall curve), balanced accuracy, precision and recall [1].

At the same time, manual search for the most effective of the above methods for a given data set is quite a difficult task even for an experienced data scientist. In such cases, various approaches to automated machine learning (*AutoML*) usually come to the rescue. They also allow you to automate data processing and modeling processes. However, their effectiveness in solving the problem of imbalanced binary classification is an open question [2, 3].

This work is aimed at analyzing the effectiveness of existing *AutoML* solutions and proposing our own approach based on optimization over the configuration space of ensemble algorithms implementing the *reweighting* and *resampling* techniques. The described approach made it possible to achieve decent results on the benchmark for imbalanced classification relative to the f1 and balanced accuracy measures for the test sample and the time of optimization search.

II. Related work

This section will cover the most popular *AutoML* solutions for classification tasks on tabular data. Attention will also be paid to their mechanisms for solving imbalanced learning problems, if mentioned in the relevant papers. All existing solutions can be divided into two categories: those focused on selecting a model and setting its hyperparameters and those focused on ensembling of the most effective models.

The AutoGluon(AG) solution is based on three main principles: training as many different models as possible, bagging them to obtain predictions, and then stacking these models to combine their predictions into a final model(also known as *meta-model*). *Stacking* is an ensemble technique, the principle of which is similar to the design of neural networks, except that at each layer there are machine learning models (in the case of AG, these are *bag models*), which pass predictions (in the case of classification, their probabilities) on the data set (averaged by layer) as additional features to the next layer. The *metamodel* aggregates predictions from previous layers to form final predictions. During the development of AG, its creators tested the performance of 1310 models on 200 different data sets [4]. AG has several presets that

conditionally affect the quality of predictions by determining the composition and order of algorithms in the search space [5].

The Tree-based pipeline optimization tool (TPOT) uses the genetic programming (GP) mechanism to design machine learning pipelines. GP selects genetic operators (in the case of TPOT, pipeline structural elements) based on specified fitness measures, mutation operations, and crossover. As the name suggests, pipelines are presented in tree form. Model ensemble is not performed, so its applicability for the imbalanced learning tasks is questionable [6].

The approach presented in the Auto-sklearn solution is to "warm up" the optimization search based on *meta-learning* and use the *ensemble selection* method. Simple, statistical and information-theoretical *meta-features*, such as the number of instances, the number of features, the number of classes, entropy, etc., are used as data for *meta-learning* [7]. The optimization algorithm is Bayesian optimization, which is a sequential design strategy for global optimization of "black box" functions. *Ensemble selection* is a technique for greedy design of an ensemble of models, the structural elements of which are selected based on maximization of the value of the validation metric [8].

The FLAML solution created at Microsoft Research is designed to search for a machine learning model and optimize its hyperparameters with a focus on low fit and prediction time costs, as well as an emphasis on model interpretability. The optimization strategy considers the structure of the search space to order the algorithms to trade-off validation error and time spent. FLAML iteratively decides the model, sample size and resampling strategy based on their compound impact on the mentioned variables. Ensemble techniques are not used [9].

The LightAutoML solution developed in Sber is another alternative for finding a lightweight model. The search space mainly consists of boosting and linear models. It is mainly applicable to the financial sector, where imbalanced data is quite common [10].

III. Methodology

This article describes an improved version of the previously proposed approach (Algorithm 1) [11].

Algorithm 1 Automated imbalanced ensemble learning

Input: imbalanced dataset: *D*, quality metric: *M*, number of trials: *T*.
Output: ensemble model: *Best*, quality: *Q*.

1: Data processing of *D*.
2: Train and test splits: *train* and *test*.
3: Initialize TPE(*train*, *M, T,* reweighting and undersampling ensembles).
4: **For** trial in *T* **do**
5: Choose ensemble configuration: *E*, based on TPE internal logic
6: *Score* ← Calculate mean cross-validation score(*train, M, E*)
7: Compare with *Best* and overwrite *Best* if *Score* is greater
8: **end for**
9: *Predictions* ← Predict(*Best, test)*
10: *Q* ← Evaluate(*Predictions, M*)
11: **Return** *Best, Q*

The input parameters are: a tabular dataset with class imbalance, quality metric name and optionally number of trials. Dataset undergoes a pre-processing stage that includes: filling in missing values, encoding categorical data into numerical form, etc. Next, the dataset is split into training and testing in a ratio of 80 to 20 (with stratification). Later, the training set will also be split into training and validation parts during the *cross-validation* procedure(also with stratification). This procedure involves splitting the training dataset into 8 folds, one of which is used to calculate the specified validation metric. The search space includes the following classifiers: AdaUBoost, AdaCost, AsymBoost, BalancedRandomForest, BalancedBagging [12]. The number of optimization trials is 70. For large datasets, the search space is halved.

The Ray framework is used as a computing core, and HyperOpt is used to represent the search space and optimize it [13, 14]. The optimization algorithm chosen is Tree-structured Parzen estimator (TPE). It is a computationally efficient implementation of Bayesian optimization. The search space in this case is represented as a tree, which allows limiting the choice of incompatible configurations of the model and its hyperparameters. Its main drawback is its sequential nature, which limits the possibilities of parallel search. However, in our implementation, Ray is used, which allows running several trials concurrently, distributing them across threads. As a result of optimization, the model with the highest value of the validation metric is saved to evaluate its predictive ability [15].

Presented methodology is a combination of *boosting* and *bagging* ensemble methods, which, together with the optimization algorithm used, allow achieving decent results on test data with effective time costs.

IV. Evaluation

To evaluate the efficiency of the proposed solution, we will conduct a benchmark with the FLAML framework on an unlimited time budget. The choice of FLAML is due to the fact that this solution is designed on the same principle as the approach proposed in this article, i.e. on the search for the best individual model, which is subsequently used for predictions on test data. We have also tried other solutions, like Auto-sklearn, LightAutoML and TPOT, but the first two were incompatible with the scikit-learn version used by the imbalanced-learn package(we needed it to run benchmarks from Zenodo); the latter one is very computationally expensive and therefore takes a long running time to achieve decent results.

As a validation metric, we will use the f1-measure(harmonic mean of accuracy and recall) and balanced accuracy(average of recall on each class). Data sets will be taken from Zenodo [16]. The problem being solved is classification. Below is information on the sets used indicating

the proportion of the positive class of the target variable (Table I) and the results of running FLAML and ImbaML (the proposed solution) upon the f1-measure, alongside optimization search time for each solution(Table II).

Table I. Characteristics of datasets

Dataset name	PCR
ecoli	0,12
optical_digits	0,11
satimage	0,11
pen_digits	0,11
abalone	0,1
sick_euthyroid	0,1
spectrometer	0,09
car_eval_34	0,08
isolet	0,08
us_crime	0,08
yeast_ml8	0,08
scene	0,08
libras_move	0,07
thyroid_sick	0,07
coil_2000	0,06
arrhythmia	0,06
solar_flare_m0	0,05
oil	0,04
car_eval_4	0,04
wine_quality	0,04
letter_img	0,04
yeast_me2	0,04
webpage	0,03
ozone_level	0,03
mammography	0,02
protein_homo	<0,01
abalone_19	<0,01

Table II. Benchmark results. Value of F1-measure on test sample

	Value of f1-measure		Optimization search time(min.)	
Dataset name	ImbaML	FLAML	ImbaML	FLAML
ecoli	0.706	0.6	5	1
optical_digits	0.92	0.977	12	2
satimage	0.598	0.711	13	2
pen_digits	0.998	0.998	16	1
abalone	0.401	0.205	7	3
sick_euthyroid	0.816	0.831	7	1
spectrometer	0.842	0.842	8	1
car_eval_34	0.931	0.885	4	1
isolet	0.807	0.866	112	85
us_crime	0.522	0.458	5	2
yeast_ml8	0.187	0	9	3
scene	0.286	0.053	21	26
libras_move	0.75	0.75	5	1
thyroid_sick	0.852	0.884	6	1
coil_2000	0.207	0.084	16	3
arrhythmia	1	0.75	3	2
solar_flare_m0	0.258	0.19	4	1
oil	0.556	0.364	4	1
car_eval_4	0.897	1	4	1
wine_quality	0.324	0.385	4	3
letter_img	0.929	0.957	7	2
yeast_me2	0.36	0.4	4	1

webpage	0.619	0.812	74	17
ozone_level	0.276	0	6	1
mammography	0.672	0.719	5	3
protein_homo	0.831	0	23	75
abalone_19	0.071	0	1	2

For a more visual presentation of the results on the test sample (upon f1-measure), bar charts are provided (Fig. 1, Fig. 2).

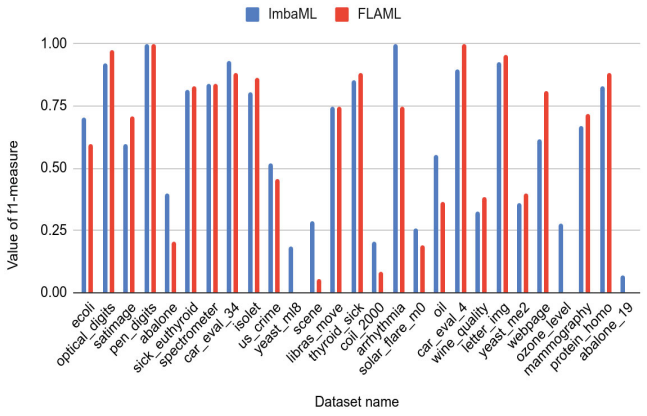

Fig. 1. Comparison of performance upon f1-measure

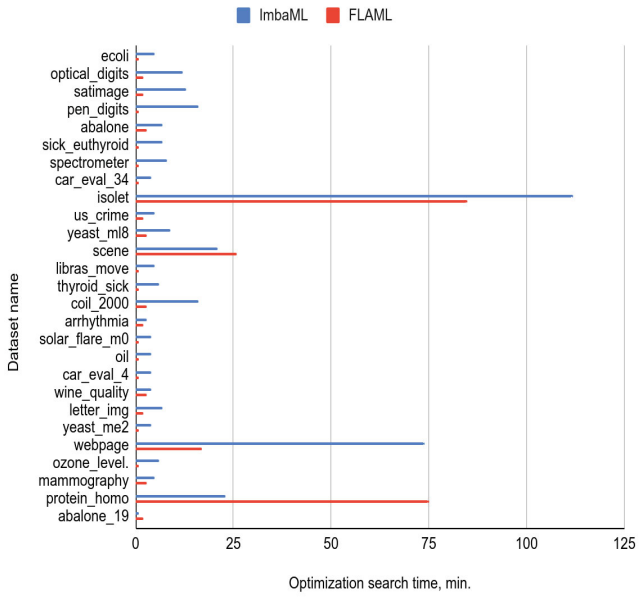

Fig. 2. Comparison of optimization search time for the first run

As a result we got a decent competitiveness upon f1-measure. In some cases, FLAML failed to find even a minimally efficient solution, i.e. the f1-measure value was zero. On average, optimization search in our solution takes a few minutes more than in FLAML. In the worst and best case scenarios, the difference for both sides was approximately an hour.

We also conducted the Mann-Whitney U-rank test to more accurately assess the results obtained. This type of statistical test is used to test if two independent samples(from different populations) have equal distributions. The null hypothesis is that the distributions of the ImbaML and FLAML quality scores are equal. The alternative is that the distributions are not equal. The significance level is 0.5. As a result, we obtained a p-value of 0.84, so we cannot reject the null hypothesis. There is no stochastically significant advantage from any side.

The results upon a balanced accuracy metric are also presented (Table III).

TABLE III. BENCHMARK RESULTS. VALUE OF BALANCED ACCURACY ON TEST SAMPLE

	Value of balanced accuracy		Optimization search time(min.)	
Dataset name	ImbaML	FLAML	ImbaML	FLAML
ecoli	0.888	0.714	6	1
optical_digits	0.984	0.977	19	2
satimage	0.878	0.811	14	2
pen_digits	0.997	0.998	14	1
abalone	0.79	0.559	8	5
sick_euthyroid	0.955	0.907	8	1
spectrometer	0.909	0.934	8	1
car_eval_34	0.955	0.923	5	1
isolet	0.947	0.913	133	66
us_crime	0.887	0.674	9	2
yeast_ml8	0.584	0.5	23	3
scene	0.696	0.512	45	27
libras_move	0.803	0.8	7	1

thyroid_sick	0.979	0.912	7	2
coil_2000	0.686	0.518	16	3
arrhythmia	0.994	0.8	7	2
solar_flare_m0	0.681	0.562	7	1
oil	0.782	0.622	8	1
car_eval_4	0.988	1	6	1
wine_quality	0.825	0.632	14	3
letter_img	0.975	0.959	12	2
yeast_me2	0.851	0.647	6	1
webpage	0.902	0.864	46	17
ozone_level	0.671	0.499	9	2
mammography	0.933	0.807	8	3
protein_homo	0.956	0.909	170	80
abalone_19	0.849	0.5	6	2

For a more visual presentation of the results on the test sample (upon balanced accuracy), bar charts are provided (Fig. 3, Fig. 4).

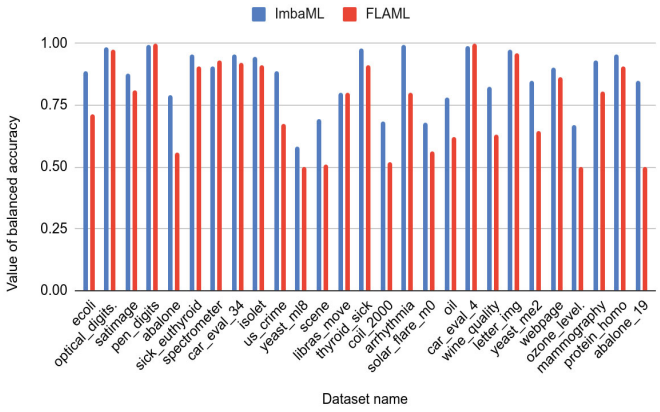

Fig. 3. Comparison of performance upon balanced accuracy measure

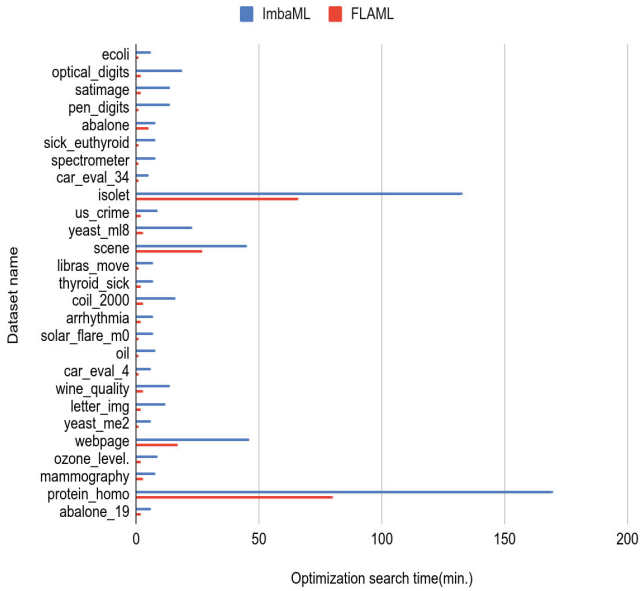

Fig. 4. Comparison of optimization search time for the second run

As a result, our solution was almost always more efficient by a balanced accuracy measure. In most cases, the time it took to find a model was approximately ten minutes longer than in FLAML. However, in this run, more time was spent by our solution in computationally expensive cases, while the time spent by FLAML remained comparable to the previous run.

Again, we conducted the Mann-Whitney U-rank test to evaluate the results obtained, but this time with another alternative hypothesis. The null hypothesis is that the distributions of the ImbaML and FLAML quality scores are equal. The alternative is that the distribution of ImbaML is stochastically greater than that of FLAML. The significance level is 0.05. As a result, we obtained a p-value of 0.02, therefore we reject the hypothesis of equality and accept the alternative one. This time, there is a stochastically significant advantage of the ImbaML solution.

V. Conclusion

The described approach allowed us to achieve positive results on the benchmark datasets for imbalanced classification relative to the balanced accuracy measure and decent results relative to the f1-measure. Statistical tests confirmed results obtained. On average, we also achieved comparable results for the optimization search time.

The developed solution has decent practical applicability, however it makes sense to integrate advanced ensemble techniques (like *stacking* or *ensemble construction,* etc.) to be competitive with state-of-the-art AutoML solutions, like Auto-Gluon, etc. It is also planned to refine the methodology for the case of multi-class classification and maybe add functionality of working with data of different modalities(like

text, audio, images, etc.). The solution is open source and can be found on Github[1].

References

[1] Haixiang G., Yijing L., Shang J., Mingyun G., Yuanyue H., Bing G. Learning from class-imbalanced data: Review of methods and applications. *Expert systems with applications*, 2017, vol. 73, pp.220-239.

[2] Hutter F., Kotthoff L., Vanschoren J.. Automated machine learning: methods, systems, challenges. *Springer Nature*, 2019.

[3] He X, Zhao K, Chu X. AutoML: A survey of the state-of-the-art. *Knowledge-based systems*, 2021 vol. 212, 106622.

[4] Salinas D., Erickson N.. Tabrepo: A large scale repository of tabular model evaluations and its automl applications. *arXiv,* 2023. *arXiv:2311.02971*.

[5] Erickson N., Mueller J., Shirkov A., Zhang H., Larroy P., Li M., Smola A. Autogluon-tabular: Robust and accurate automl for structured data, *arXiv*, 2020. arXiv:2003.06505.

[6] Le T, Fu W, Moore J. Scaling tree-based automated machine learning to biomedical big data with a feature set selector. *Bioinformatics*, 2020, vol.36, pp. 250-256.

[7] Rivolli A, Garcia L, Soares C, Vanschoren J, de Carvalho A. Meta-features for meta-learning. *Knowledge-Based Systems*, 2022, vol. 240, 108101.

[1] https://github.com/AxiomAlive/ImbaML

[8] Feurer M, Eggensperger K, Falkner S, Lindauer M, Hutter F. Auto-sklearn 2.0: The next generation. *arXiv*, 2020, arXiv:2007.04074.

[9] Wang C, Wu Q, Weimer M, Zhu E. Flaml: A fast and lightweight automl library. *Proceedings of Machine Learning and Systems*, 2021, vol.3, pp. 434-447.

[10] Vakhrushev A, Ryzhkov A, Savchenko M, Simakov D, Damdinov R, Tuzhilin A. Lightautoml: Automl solution for a large financial services ecosystem. *arXiv*, 2021, arXiv:2109.01528.

[11] Aliev M., Muravyov S. Imba: Configuration-free Imbalanced Learning. *Proc. of 36th FRUCT Conference of Open Innovations Association (FRUCT)*, pp. 88-93.

[12] LemaÃŽtre G, Nogueira F, Aridas CK. Imbalanced-learn: A python toolbox to tackle the curse of imbalanced datasets in machine learning. *Journal of machine learning research*, 2017, vol. 18, pp. 1-5.

[13] Moritz P, Nishihara R, Wang S, Tumanov A, Liaw R, Liang E, Elibol M, Yang Z, Paul W, Jordan M, Stoica I. Ray: A distributed framework for emerging AI applications. *In13th USENIX symposium on operating systems design and implementation (OSDI 18)*, 2018, pp. 561-577.

[14] Komer B, Bergstra J, Eliasmith C. Hyperopt-sklearn. *Automated machine learning: methods, systems, challenges*, 2019, pp. 97-111.

[15] Watanabe S. Tree-structured parzen estimator: Understanding its algorithm components and their roles for better empirical performance. *arXiv*, 2023, arXiv:2304.11127.

[16] Ding Z. Diversified ensemble classifiers for highly imbalanced data learning and their application in bioinformatics, 2011.