`ImbalancedLearningRegression` - A Python Package to Tackle the Imbalanced Regression Problem

Wenglei Wu[1][0000−0001−5584−5543], Nicholas Kunz[2][0000−0002−3218−2131], and Paula Branco[1][0000−0002−9917−3694] ✉

[1] Faculty of Engineering, University of Ottawa, Ottawa, Ontario, Canada
wwu077@uottawa.ca, pbranco@uottawa.ca
[2] College of Engineering, Cornell University, Ithaca, New York, United States
nhk37@cornell.edu

Abstract. This package helps Python users address imbalanced regression problems. Popular Python packages exist for imbalanced classification. However, there is still little Python support for imbalanced regression. Imbalanced regression is a well-known problem that occurs across domains, where a continuous target variable is poorly represented on ranges that are important to the end-user. Here, a re-sampling strategy is applied to modify the distribution of the target variable, biasing it towards the end-user interests so that downstream learning algorithms can be trained on the most relevant cases. The package provides an easy-to-use and extensible implementation of eight state-of-the-art re-sampling methods for regression, including four under-sampling and four over-sampling techniques. Code related to this paper is available at: `https://github.com/paobranco/ImbalancedLearningRegression`.

1 Introduction

Imbalanced domains are characterized by having an imbalanced target variable. A model trained on an imbalanced data set cannot focus on the important regions and thus is not able to predict well the most important rare cases [2]. Research has been more intensive on the imbalanced classification problem, with a vast number of re-sampling techniques being proposed. However, this issue also occurs in regression tasks where the target variable is continuous. To define the important and unimportant ranges of the target variable, we use the notion of relevance function that can be either estimated from the data distribution or explicitly provided by the end-user [12]. In the automatic method, low-density ranges are mapped to high relevance values while high-density ranges are mapped to low relevance values. The formed ranges can be thought of as different minority (important) and majority (unimportant) classes, in a classification setting.

Implementations of a high diversity of re-sampling techniques for class imbalance are available in Python (imbalanced-learn [10]) and R (imbalance [5], UBL [1]). However, this is not the case for imbalanced regression for which some methods exist in R (UBL [1]), but only one package exists in Python that implements a single over-sampling method: SMOGN [3,9]. The proposed Python package `ImbalancedLearningRegression` fills this gap.

2 The `ImbalancedLearningRegression` Package

Our package provides different re-sampling techniques for the imbalanced regression problem in Python based on the data analysis libraries `pandas`, `numpy`, and `scikit-learn`. At the current stage of development, eight re-sampling methods have been implemented, including four over-sampling methods: Random Over-sampling (RO) [4,11], SMOTE [14], Introduction of Gaussian Noise (GN) [4], ADASYN [8]; and four under-sampling methods: Random Under-sampling (RU), Condensed Nearest Neighbor (CNN) [7], TomekLinks [13], Edited Nearest Neighbor (ENN) [15]. These methods perform differently in terms of data manipulation, execution time, and the number of samples created or deleted. It is up to the user to select an appropriate method for re-sampling a specific domain. The representation of the data sets through `pandas` data frame in `ImbalancedLearningRegression` gives the end-user the flexibility to apply any pre-processing steps before and/or after the use of `ImbalancedLearningRegression`.

For the sake of usability, only two parameters are required to be specified to execute a re-sampling method in the package: (i) the data set in the form of a pandas data frame, and (ii) the name of the target variable. The remaining parameters have default values that globally correspond to the following assumptions: the less dense target variable regions are the most important ones, and the user's goal is to balance the important and unimportant cases. End-users can change any parameter to control the behavior of the re-sampling strategy. `ImbalancedLearningRegression` is organized into several modules and is therefore consistent, maintainable, and extensible. Future collaborators can take advantage of its structure to implement improvements or add more re-sampling techniques for the imbalanced regression problem. The package can be used on any OS supported by Python, including Windows, macOS, and Linux. It is fully open-source and is available under a GNU General Public License v3 (GPLv3). The source code can be found at `https://github.com/paobranco/ImbalancedLearningRegression`, and an introduction video is available at `https://youtu.be/BanN904NyX0`. The documentation can be found at `https://imbalancedlearningregression.readthedocs.io/en/latest`. The package can be easily installed via PyPI[3] using `pip install ImbalancedLearningRegression`.

3 Some Application Examples

We present a basic use case of re-sampling with the Ames Housing data set [6] to show how simple it is to use `ImbalancedLearningRegression`. This data set illustrates a regression task where `SalePrice` is the continuous target variable. We applied four different re-sampling methods with default parameter settings. The complete code of execution is shown below.

[3] `https://pypi.org/project/ImbalancedLearningRegression/`

```
import ImbalancedLearningRegression as iblr
import pandas as pd

housing = pd.read_csv("housing.csv")
housing_smote = iblr.smote(data = housing, y = "SalePrice")
housing_gn = iblr.gn(data = housing, y = "SalePrice")
housing_cnn = iblr.cnn(data = housing, y = "SalePrice")
housing_enn = iblr.enn(data = housing, y = "SalePrice")
```

The first two lines import our package `ImbalancedLearningRegression`, as well as the data analysis library `pandas`. The following line loads the data from a file to a standard `pandas` data frame. Each one of the next four lines applies a re-sampling method available in the package. In this example, we selected SMOTE, GN, CNN, and ENN methods. Two parameters are necessary to be specified to run the techniques: the instance of the `pandas` data frame is assigned to the parameter `data`, and a string of the name of the target variable is assigned to the parameter `y` that represents the target variable. Users can also control the degree of re-sampling by setting the parameter `samp_method`, or control the threshold of classifying majority and minority by setting the parameter `rel_thres`. For more details regarding the optional parameters, please refer to the package documentation.

The original Ames Housing data set contains 1460 samples. After applying SMOTE, GN, CNN, and ENN, the number of samples in the modified data sets changed to 1974, 1459, 401, and 1428 respectively. Figure 1 shows the density distribution of our data set before and after applying the four different re-sampling techniques. We observe that the distribution of the Ames Housing data set changes considerably when SMOTE, GN, and CNN are applied, whereas it is only slightly affected when ENN is used.

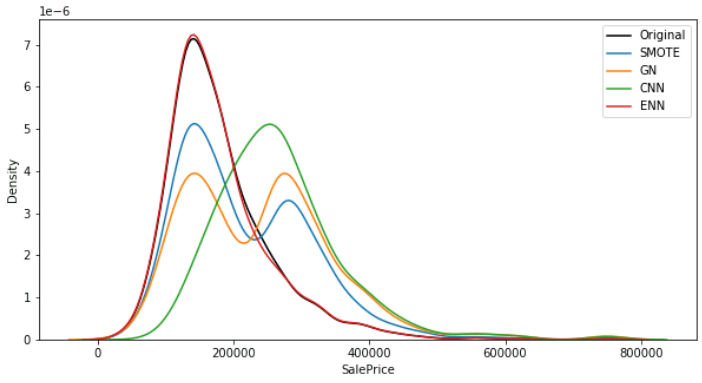

Fig. 1. Density distribution of Ames Housing data set before and after applying four re-sampling methods using `ImbalancedLearningRegression` package.

4 Conclusion

Here we introduced the `ImbalancedLearningRegression` package that allows the application of multiple re-sampling techniques to address the imbalanced problem in regression tasks in a Python environment. This package provides an easy-to-use, extensible, and freely available implementation of solutions for this problem.

Acknowledgements We would like to thank Xinzi Hu, Lingyi Kong, and Chengen Lyu for their contributions to the re-sampling implementations.

References

1. Branco, P., Ribeiro, R.P., Torgo, L.: UBL: an R package for utility-based learning (2016), `https://arxiv.org/abs/1604.08079`
2. Branco, P., Torgo, L., Ribeiro, R.P.: A survey of predictive modeling on imbalanced domains. ACM Computing Surveys (CSUR) **49**(2), 1–50 (2016)
3. Branco, P., Torgo, L., Ribeiro, R.P.: SMOGN: a pre-processing approach for imbalanced regression. In: First international workshop on learning with imbalanced domains: Theory and applications. pp. 36–50. PMLR (2017)
4. Branco, P., Torgo, L., Ribeiro, R.P.: Pre-processing approaches for imbalanced distributions in regression. Neurocomputing **343**, 76–99 (2019)
5. Cordón, I., García, S., Fernández, A., Herrera, F.: Imbalance: Oversampling algorithms for imbalanced classification in r. Knowledge-Based Systems **161**, 329–341 (2018), `https://doi.org/10.1016/j.knosys.2018.07.035`
6. De Cock, D.: Ames, iowa: Alternative to the boston housing data as an end of semester regression project. Journal of Statistics Education **19**(3) (2011)
7. Hart, P.: The condensed nearest neighbor rule (corresp.). IEEE transactions on information theory **14**(3), 515–516 (1968)
8. He, H., Bai, Y., Garcia, E.A., Li, S.: ADASYN: Adaptive synthetic sampling approach for imbalanced learning. In: 2008 IEEE international joint conference on neural networks. pp. 1322–1328. IEEE (2008)
9. Kunz, N.: SMOGN: Synthetic minority over-sampling technique for regression with gaussian noise (2020), `https://pypi.org/project/smogn`
10. Lemaître, G., Nogueira, F., Aridas, C.K.: Imbalanced-learn: A python toolbox to tackle the curse of imbalanced datasets in machine learning. JMLR **18**(17), 1–5 (2017), `http://jmlr.org/papers/v18/16-365.html`
11. Menardi, G., Torelli, N.: Training and assessing classification rules with imbalanced data. Data mining and knowledge discovery **28**(1), 92–122 (2014)
12. Ribeiro, R.P.: Utility-based regression. Ph.D. thesis, Dep. Computer Science, Faculty of Sciences - University of Porto (2011)
13. Tomek, I.: Two modifications of cnn. IEEE Trans. Systems, Man and Cybernetics **6**, 769–772 (1976)
14. Torgo, L., Ribeiro, R.P., Pfahringer, B., Branco, P.: Smote for regression. In: Portuguese conference on artificial intelligence. pp. 378–389. Springer (2013)
15. Wilson, D.L.: Asymptotic properties of nearest neighbor rules using edited data. IEEE Transactions on Systems, Man, and Cybernetics (3), 408–421 (1972)