Sample Bias Effect on Meta-Learning

Mariane Salomão dos Reis^{1,2}, Ana Carolina Lorena^{1}

^{1}Computer Science Department – Instituto Tecnológico da Aeronáutica (ITA)
São José dos Campos – SP – Brazil
aclorena@ita.br

^{2}Time de Data Science – Itaú Unibanco
São Paulo – SP – Brazil
mariane.salo@gmail.com

Abstract. *Sample bias is a common issue on traditional machine learning studies but rarely considered when discussing meta-learning. It happens when the training data sample lacks or overemphasizes one or more characteristics, compared to others. Herewith, models trained on such data may become inaccurate for some instances. This work aims to analyze this issue in the meta-learning context. Indeed, in most of the meta-learning literature, a random sample of datasets is taken for building meta-models. Nonetheless, there is no discussion over a possible side-effect bias not controlled in such random sampling. This work aims to analyze these effects, in order not only to discuss their consequences, but also to start a debate over the need of their consideration in meta-learning research.*

1. Introduction

Machine learning (ML) refers to a vast set of tools and algorithms for analyzing and understanding data [Gareth et al. 2013]. Mostly, it involves building a predictive model for estimating an output for a given input data. This process can be simplified into five main steps:

1. *Data Gathering*: finding, collecting and integrating data from different sources (e.g: APIs, websites, data warehouse);

2. *Data Preparation and Preprocessing*: data may be in different shapes, so there is the need to fix errors and rearrange formats;

3. *Feature Selection and Engineering*: not every information is relevant and frequently it is necessary to refine how data is going to be inputted into our model, by either creating new features or selecting a subset of the available features;

4. *Modeling*: involves choosing which model to use, set its hyperparameters values and performing the algorithm training;

5. *Prediction and Model Interpretation*: checking if the model generalizes well to new data and interpreting the patterns obtained.

Each of these steps requires human labor, often with several trial and error episodes until a good outcome is achieved. In terms of an enterprise it means that if one wants to scale ML production, a large specialized team is needed. However, several companies do not have the resources or means necessary to build such a technical team. Therefore, comes the urge to automatize the end-to-end ML process, which is the objective of the Automated machine learning (AutoML) research area [Hutter et al. 2019].

AutoML aims to make ML process populated with data-driven and automated decisions in a user friendly way. In other words, the objective is to democratize the ML usage. Herewith, an user with less expertise can provide the data to the AutoML system, which will decide the best approach for this new problem [Hutter et al. 2019].

Meta-Learning is a subarea of AutoML and has interest in "learning to learn" [Vanschoren 2019]. When humans master something new, we normally make associations with knowledge we obtained in previous experiences. These associations provide us a possibility to learn across tasks. Meta-Learning (MtL) tries to emulate this human behavior by observing the performance of different ML techniques on diverse tasks, calling these observations meta-data. Using this meta-data, MtL tries to leverage the efficiency of the learning process when presented to new similar tasks [Vanschoren 2019].

It is possible to use MtL to either: select an algorithm for a new problem, rank a set of candidate algorithms, or estimate the predictive accuracy of one or more classifiers. Each of these cases can be modelled as a learning task by themselves. In the latter case, one may see MtL as a regression task, whilst the former is a standard classification task. Bensusan and Kalousis [Bensusan and Kalousis 2001] show that this estimation can be done with a high degree of confidence.

Normally, when performing learning from data, we assume that the training and the testing data belong to the same distribution. Nonetheless, this assumption may not hold in most of the real world problems. This difference can be attributed to dynamic characteristics of the problem or, in most of the cases, to faults in the data gathering process, making the training dataset not a good representation of our problem global properties [Zadrozny 2004]. This issue is also named *sample bias* in the literature. An example would be training a voice detection algorithm with a dataset composed of male voices only, which will fail to generalize to female voices. As a result of faults similar to this, some models have been recently noticed as spreaders of sexism, racism or even xenophobic behavior [Maloney 2017, Wallis 2018, Comi 2018].

There are countless works on the effects of sample bias in traditional ML algorithms and also studies proposing ways of overcoming this problem by making adaptions on classical learning algorithms, specially in econometrics studies [Liu and Ziebart 2014, Huang et al. 2007]. However, limited discussions have been made involving MtL and sample bias. Indeed, in most works from the literature the datasets to be employed as learning experience are chosen at random. Since the meta-model induction is a learning problem itself, the sample bias issue can also be observed and the meta-model may not generalize well to some new datasets.

There are also some works demonstrating that many of the common repositories of benchmark datasets used by the Machine Learning (ML) community lack diversity [Macià and Bernadó-Mansilla 2014, Muñoz et al. 2018] and propose strategies to generate more diverse datasets. For instance, [Macià and Bernadó-Mansilla 2014] show experimentally that the classification datasets of the UCI repository [Dua and Graff 2017] are not challenging enough and many ML techniques are able to achieve a high predictive performance on them. These studies differ from what we consider here but corroborate our investigation, once the datasets commonly gathered in MtL studies come from such public repositories.

In this work, we intent to explore more the sample bias effect in a MtL setup, encouraging the community to look towards solutions for this issue also on the AutoML field. This paper is organized as follows: Section 2 presents the materials and methods employed in this work. Section 3 presents some experiments used to evaluate sample bias in MtL. And Section 4 concludes this work.

2. Materials and Methods

In order to develop our analysis, we followed the steps shown in Fig. 1, detailed next.

Data Gathering
Data Preparation
Sample Construction
Meta-learning task
Sample comparison

Figure 1. Methodology adopted in the work.

2.1. Data Gathering

When using MtL it is necessary to gather a set of datasets for which the ML solutions are known. They must be also characterized by some measures or meta-features, which results on a meta-dataset construction. The meta-dataset is similar to a regular dataset used in ML, the difference is that each line represents one particular dataset and the columns exhibit their characteristics or meta-features, accompanied with a label according to the MtL task considered (e.g., the best of a pool of algorithms for the dataset). Fig. 2 demonstrates a meta-dataset construction. Consequently, to build a meta-dataset, several datasets are needed. The present work uses datasets sampled from the OpenML repository [Vanschoren et al. 2014], using a Python API in their collection [Feurer et al. 2019]. OpenML is an open repository of ML benchmarks, including several datasets, pre-computed meta-features and also results from runs of various ML algorithms.

We used the API specification to select and extract meta-features from datasets that were submitted to supervised classification and had ML algorithms run with a 10-fold cross validation procedure for estimating accuracy performance. These meta-features are accessed by the qualities of the dataset object provided by the API [Feurer et al. 2019]. Datasets with unknown meta-features were filtered out from this selection.

Next, we collected data on the performance obtained by classifiers in the predictive tasks considered. An OpenML task contains information on ML algorithms applied on a specific dataset, along with their performance. For each dataset, different ML algorithms may be applied. In our MtL experiments we disregard which are the ML algorithms employed and retrieve the best accuracy value reported for each dataset. Each dataset in our meta-dataset will then be labeled according to the best achievable performance as reported in the OpenML platform. The objective of the meta-learner will be to predict

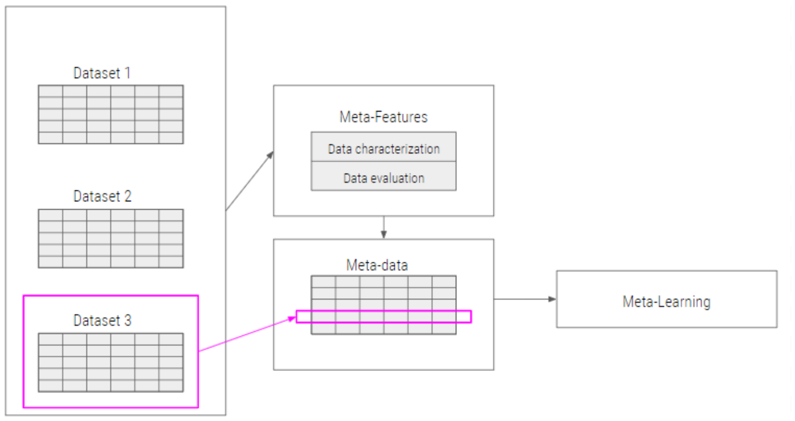

Figure 2. Meta-data creation Prodecure.

the top performance that may be achieved for a new dataset, in a regression task. We are not aware of similar approaches in the literature, although it is sound to expect similar predictive performances for datasets with very similar characteristics.

As a result of this step, we have a meta-dataset composed of 670 datasets and 107 meta-features, each one labeled according to the best accuracy performance in cross-validation registered in OpenML.

2.2. Data Preparation

As an initial step of our data preparation phase, we checked if the collected meta-features made sense when considering our problem of predicting a model performance. The existent meta-features are usually classified in the related literature as simple, statistical, information-theoretic, complexity, model-based, and landmarkers [Vanschoren 2019]. Table. 1 presents some of the collected metrics per category.

Table 1. Examples of meta-features per group

Category	**Meta-features**
simple	Number of instances, Number of features, Number of classes
statistical	Skewness, Kurtosis, Correlation, Sparsity
information	Class entropy, Mutual information
complexity	Data consistency, Concept variation
model-based	Number of nodes, Number of Leaves, Information gain
landmarkers	Landmarker(1NN), Landmarker(NB), Landmarker(Tree)

In order to clean our meta-data, we removed the columns that correspond to irrelevant information to the MtL algorithm. This happened for some meta-features which had the same value for all datasets, or other meaningless information, such as the time and date when someone uploaded the learning task. Afterwards, we analyzed the correlation between the meta-features and used a threshold above 0.95 to remove some meta-features. The idea was to remove redundant information. The comparison did not take under consideration the target meta-feature. As a result of this process we cleared away 56 meta-features and 61 meta-features remained to the further steps

As a final cleaning step, we removed all datasets with missing values for at least one meta-feature, reducing our meta-dataset to 169 rows and 61 columns. It is worth noting that there is no simple method to try to input those missing information, since each one of them regards to a different dataset.

2.3. Obtaining the Training Samples

Considering X the meta-feature space, Y the target-feature space and S_i a binary space representing a given relation. If we build a dependence relation of some kind considering properties of the elements of X, we are able to select a sample when it assumes a true value for S_i [Zadrozny 2004]. The idea is to vary the S_i values according to a given condition, which can impose different biases in the training samples used in MtL. We created three types of S_i, considering characteristics that could influence the accuracy of a classifier if underrepresented in a training meta-dataset.

- Type 1: S_i reflects the dataset sparsity, as computed by Equation 1 [Ho and Basu 2002]. We calculate the median value of the sparsity values for all datasets available. Every dataset with sparsity value higher than this threshold has $S_i =$ true and it is false otherwise. The idea is to obtain two distinct sets: one with dense datasets as measured by Equation 1 and other with sparse datasets according to Equation 1. As discussed in [Ho and Basu 2002], the sparsity of a dataset is directly related to the complexity of the classification problem. More sparse datasets may pose more challenges, since there may be many regions of the input space underrepresented. Ho and Basu [Ho and Basu 2002] proposed the use of the measure presented in Equation 1 to have a rough estimate of the dataset sparsity.
$$sparsity = \frac{\sharp Instances}{\sharp Features} \tag{1}$$
- Type 2: S_i in this case is dependent on the dataset imbalance ratio [Fernández et al. 2018], given by Equation 2. A similar procedure to the first sample is taken, but here we use Equation 2 instead of Equation 1. It is also known in the related literature that a high IR may harm the predictive performance of classification models, specially for the minority class examples.
$$IR = \frac{\sharp InstancesOnMajorityClass}{\sharp InstancesOnMinorityClass} \tag{2}$$
- Type 3: S_i depends on the number of target classes of the dataset. In this case, we separated binary classification problems ($S_i = 1$) from multiclass problems ($S_i = 0$). It is also known that a higher number of classes usually implies in a higher complexity of the classification problem [Lorena et al. 2008]. Within this sample, 43 datasets are binary and the remaining 126 are multiclass.

The three previous criteria are used to create the biased samples. Therefore, we have a bias concerning the sparsity of the datasets (dense vs sparse), the IR (high IR vs low IR) and the number of classes (binary vs multiclass). We also performed some random sampling of the datasets (three training-testing pairs, due to the stochastic nature of the process). In order to simplify references to the samples we are going to use the following notations:

- Sample j Part 1: If the column S_i is True or 1;
- Sample j Part 2: If the column S_i is False or 0.

Aiming to inspect the effect of sample bias, we trained a meta-learner on each of the parts of the samples and tested its performance on the sample counterpart. Then, it is possible to compare the performance on each of the training and test pairs. The objective is to verify if training the meta-learner on a training sample with some type of bias may influence the obtained results on a different sample containing datasets with a complementary bias. We also compare the results attained to those obtained in a leave-one-out (LOO) procedure using each one of the samples alone. That would correspond to the case in which only datasets with a biased characteristic are given for both training and testing the meta-learner. Fig. 3 illustrates the procedure.

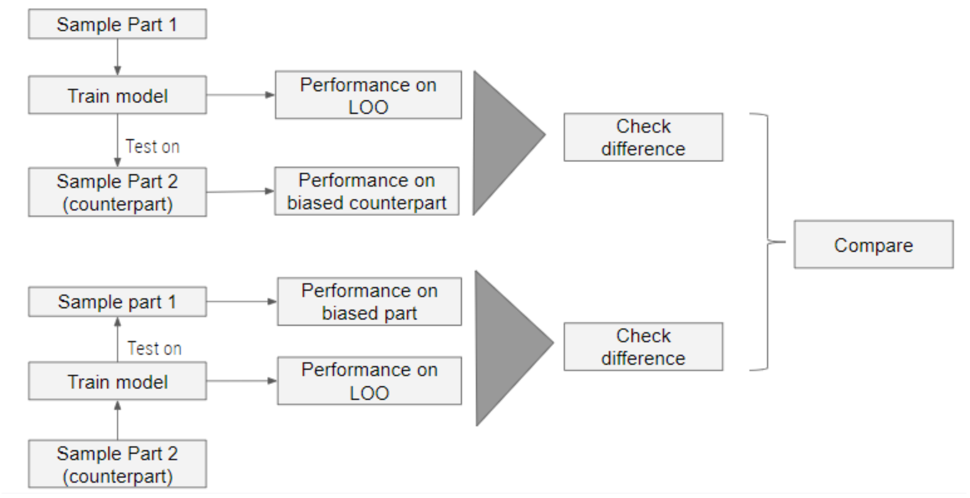

Figure 3. Modeling Procedure.

2.4. Model Training

A Random Forest (RF) regressor [Breiman 2001, Ho 1995] was chosen for building the meta-learners in this work. This choice was motivated by the good performance this technique has achieved in previous MtL experiments from the literature [Garcia et al. 2018].

For each training round performed, an hyper-parameter optimization by Grid Search was run, using 3-fold cross validation on training data and focusing on two parameters of the RF: maximum depth from the tree (max_depth) and number of trees on the forest (n_estimators). Grid Search verifies the performance for each combination of hyperparameters specified, which were here varied as follows:

- max_depth: 2, 4, 6, 8, 10
- n_estimators: 4, 8, 16, 32, 50

All meta-models induced are evaluated according to the following performance metrics: Mean Absolute Error (MAE), Mean Squared Error (MSE) and Root Mean Squared Error (RMSE) [Gareth et al. 2013]:

$$MAE = \frac{1}{n}\sum_{i=1}^{n}|y_i - y_{pred}|$$

$$MSE = \frac{1}{n}\sum_{i=1}^{n}(y_i - y_{pred})^2$$

$$RMSE = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(y_i - y_{pred})^2}$$

where n is the number of meta-instances in the test data partitions, y_i is the true label of meta-instance i (best accuracy reported at OpenML for the dataset) and y_{pred} corresponds to the values predicted by the meta-learner.

3. Results

Table 2 details the acronyms used for the samples tested. Table 3 presents the MAE, MSE and RMSE values achieved in all experiments performed.

Table 2. Acronyms used for the samples generated

Acronym	Sample name	Description
R_1P_1	*Random sample 1, part 1*	First part of first stochastic chosen sample
R_1P_2	*Random sample 1, part 2*	Second part of first stochastic chosen sample
R_2P_1	*Random sample 2, part 1*	First part of second stochastic chosen sample
R_2P_2	*Random sample 2, part 2*	Second part of second stochastic chosen sample
R_3P_1	*Random sample 3, part 1*	First part of third stochastic chosen sample
R_3P_2	*Random sample 3, part 2*	Second part of third stochastic chosen sample
SpP_1	*Sample of type 1, part 1*	Biased sample with sparsity higher than median sparsity
SpP_2	*Sample of type 1, part 2*	Biased sample with sparsity lower or equal than median sparsity
IrP_1	*Sample of type 2, part 1*	Biased sample with balance higher than median IR
IrP_2	*Sample of type 2, part 2*	Biased sample with balance lower or equal than median IR
ClP_1	*Sample of type 3, part 1*	Biased sample with multi-class target
ClP_2	*Sample of type 3, part 2*	Biased sample with binary target

Table 3. Performance Results from each model

Data	MAE LOO	MAE test	MSE LOO	MSE test	RMSE LOO	RMSE test
R_1P_1	0.045	0.057	0.004	0.007	0.004	0.083
R_1P_2	0.036	0.046	0.004	0.004	0.004	0.061
R_2P_1	0.036	0.049	0.003	0.005	0.003	0.071
R_2P_2	0.047	0.082	0.005	0.015	0.005	0.124
R_3P_1	0.040	0.043	0.004	0.004	0.004	0.064
R_3P_2	0.043	0.052	0.003	0.006	0.003	0.081
SpP_1	0.037	0.065	0.004	0.010	0.004	0.098
SpP_2	0.048	0.072	0.005	0.010	0.005	0.100
IrP_1	0.039	0.049	0.004	0.004	0.004	0.065
IrP_2	0.039	0.065	0.003	0.010	0.003	0.100
ClP_1	0.073	0.054	0.010	0.006	0.010	0.076
ClP_2	0.037	0.087	0.002	0.015	0.002	0.121

In order to compare the results considering the case where (i) the same biased data is used for both training and testing and (ii) a biased sample is used for training and a complementary testing dataset is employed, we created a graph that computes the difference between the performances achieved in such cases, that is, the performance on complementary datasets subtracted by the performance on a same biased dataset. For instance, considering the biased sample SpP_1 based on sparsity and the MAE performance metric, the following steps are followed to obtain the MAE difference:

1. LOO is performed using the biased data sample SpP_1 and the MAE obtained is recorded as MAE_1.
2. A model is trained using all SpP_1 data and has its MAE performance measured in the data sample counterpart SpP_2, leading to MAE_2.
3. The $MAE_2 - MAE_1$ difference is computed and plotted, corresponding to the last but one column in Fig. 4.

The previous procedure is repeated for each sample type and performance metric among MAE, MSE and RMSE. Fig. 4 to Fig. 6 present these results, where the bar labels represent which data sample was used as training set in the LOO experiment.

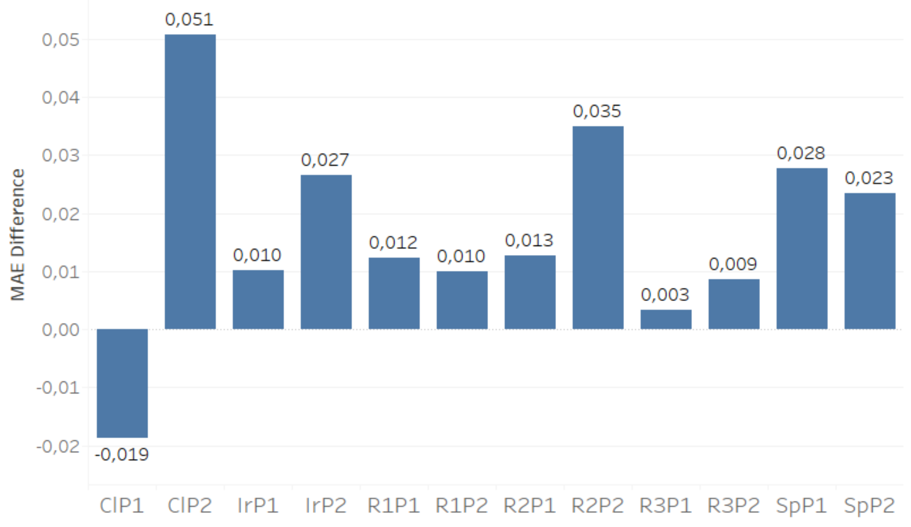

Figure 4. Mean Absolute Error Comparison.

From Fig. 4, we can see that in most of the cases there is a positive difference between using one meta-dataset for training and the complementary counterpart for testing, compared to the use of the biased meta-dataset for both training and testing in a leave-one-out setup. This was expected, since in the latter case datasets sharing the same characteristic are used for both training and testing and in the former case we have datasets with complementary characteristics. But this type of result was never assessed formally before. The only exception is a negative difference of MAE on ClP_1. This can be an artifact related to the fact that this is the smallest training sample we have, which is almost three times smaller than its test counterpart. Interestingly, for one of the random runs the difference is large and it may be the case a bias has been created by the random sampling in this run. Also, we can detect the biggest MAE difference happened with ClP_2, followed by R_2P_2 and SpP_1. Fig. 5 concerns the MSE results and present a similar behavior, where ClP_2 has a negative difference and $Clp2$ shows the highest difference. Meanwhile, R_2P_2 takes second lead and IrP_2 takes third lead. When using RMSE to compare the models (Fig. 6), we have no negative difference. Also, in this case the highest contrast appear on R_2P_2.

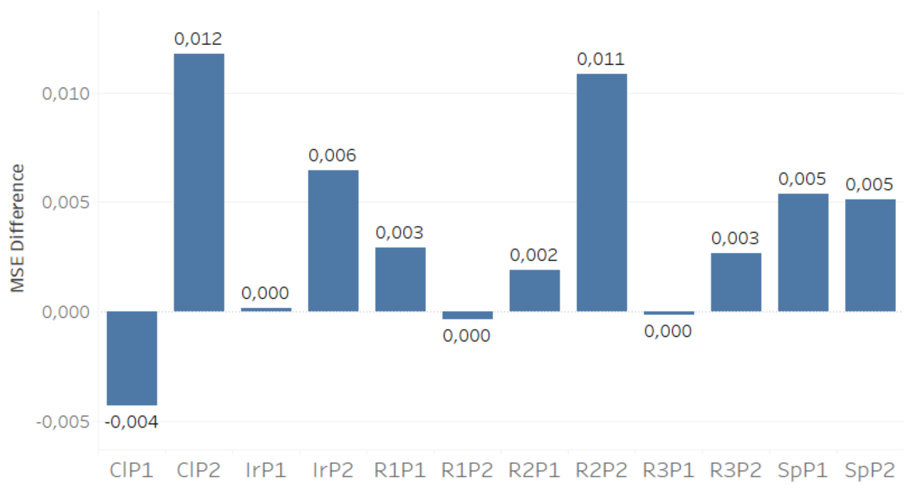

Figure 5. Mean Squared Error Comparison.

In order to explore further the reasons why the model trained on R_2P_2 behaved this way, we performed a descriptive analysis over all the random samples concerning the sparsity, the IR and the number of classes of the datasets they present. The idea was to verify if this sample had somehow perpetuated one of those bias that were already mapped. But it was not possible to notice in the graphs, from Fig. 7 to Fig. 9, any clear difference between the random samples concerning the measured characteristics. In Fig. 7 (Fig. 8), the orange bar corresponds to the proportion of observations on the data sample that have a sparsity (IR) larger than the median sparsity (IR), whereas the blue bar corresponds to percentage of observations on the data sample with sparsity (IR) lower or equal than the median sparsity (IR). For both sparsity and IR, there is a similar proportion of datasets in the two different categories considered in each case, on all random samples. In Fig. 9, the datasets are divided into binary or multiclass and in this case there is a larger proportion of binary than of multiclass datasets in all random samples, a characteristic inherited from the OpenML repository, which gathers a larger amount of binary classification problems. Again the proportions recorded are similar for all random samples, although R_2P_2 has a slightly lower proportion of multiclass datasets than the other random samples. Therefore, other characteristics may be also impacting the obtained results. But, overall, we have demonstrated that a simple random sampling of a pool of datasets to train a meta-learner can lead to a poor MtL performance afterwards and that some care must be taken in order to prevent such cases. Proposing strategies to circumvent this problem remains as an open future work.

4. Conclusion

As a conclusion from this work, we can assume that only selecting samples randomly to apply meta-learning is not sufficient to guarantee overcoming the sample selection bias. Also, it is clear that meta-learning algorithms have their performance deprecated when trained on biased meta-data and then presented to data whose trait were not even closely learned before. For this reason, it is necessary to enhance discussion over the topic in the field. Moreover, propose solutions to overthrow this issue, such as robust algorithms or better sampling strategies.

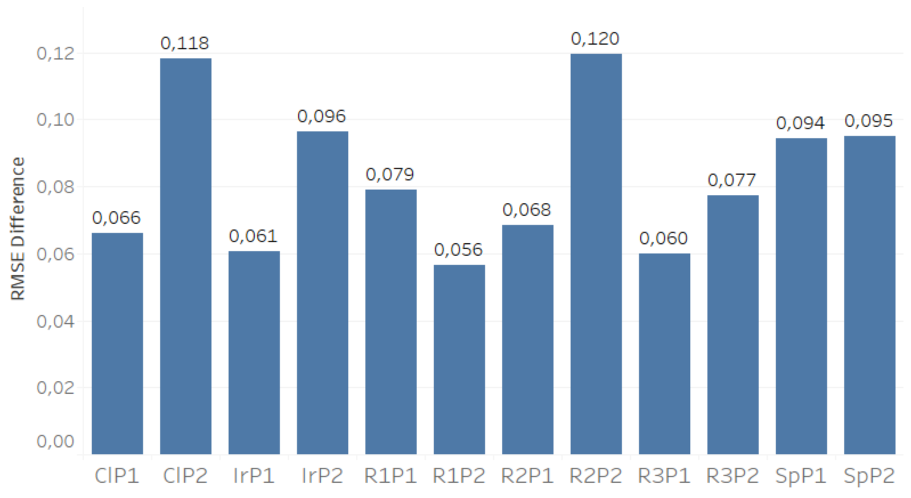

Figure 6. Root Mean Squared Error Comparison.

Acknowledgment

The authors thank the support of Itaú Unibanco and express their appreciation to the unmentioned people, who made constructive comments on the original manuscript. Also, recognize Slidesgo design features that helped construct the flowcharts presented.

References

Bensusan, H. and Kalousis, A. (2001). Estimating the predictive accuracy of a classifier. In *European Conference on Machine Learning*, pages 25–36. Springer.

Breiman, L. (2001). Random forests. *Machine learning*, 45(1):5–32.

Comi, M. (2018). Is artificial intelligence racist? (and other concerns), https://towardsdatascience.com/is-artificial-intelligence-racist-and-other-concerns-817fa60d75e9.

Dua, D. and Graff, C. (2017). UCI machine learning repository, http://archive.ics.uci.edu/ml.

Fernández, A., García, S., Galar, M., Prati, R. C., Krawczyk, B., and Herrera, F. (2018). *Learning from imbalanced data sets*. Springer.

Feurer, M., van Rijn, J. N., Kadra, A., Gijsbers, P., Mallik, N., Ravi, S., Müller, A., Vanschoren, J., and Hutter, F. (2019). Openml-python: an extensible python api for openml. *arXiv preprint arXiv:1911.02490*.

Garcia, L. P., Lorena, A. C., de Souto, M. C., and Ho, T. K. (2018). Classifier recommendation using data complexity measures. In *2018 24th International Conference on Pattern Recognition (ICPR)*, pages 874–879. IEEE.

Gareth, J., Daniela, W., Trevor, H., and Robert, T. (2013). *An introduction to statistical learning: with applications in R*. Springer.

Ho, T. K. (1995). Random decision forests. In *Proceedings of 3rd international conference on document analysis and recognition*, volume 1, pages 278–282. IEEE.

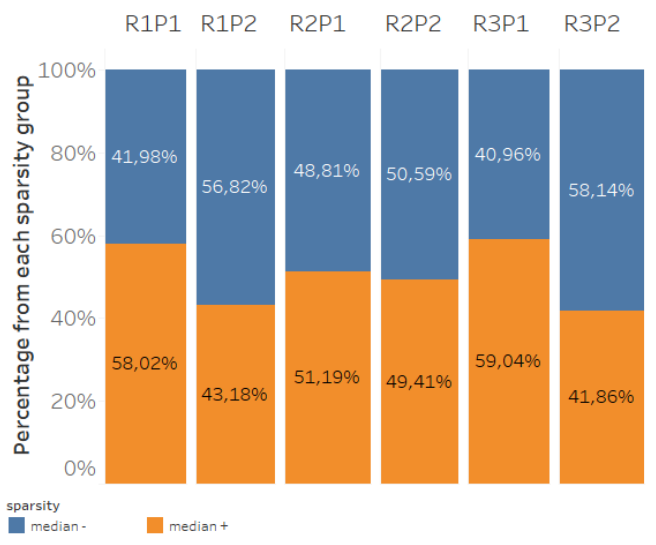

Figure 7. Sparsity proportion over random samples.

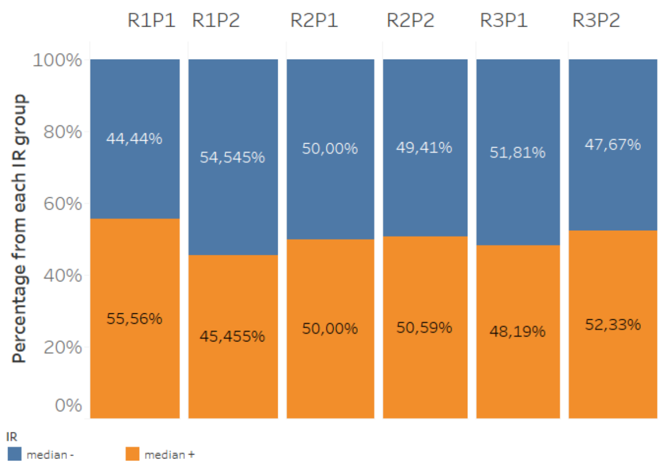

Figure 8. IR proportion over random samples.

Ho, T. K. and Basu, M. (2002). Complexity measures of supervised classification problems. *IEEE Trans. on pattern analysis and machine intelligence*, 24(3):289–300.

Huang, J., Gretton, A., Borgwardt, K., Schölkopf, B., and Smola, A. J. (2007). Correcting sample selection bias by unlabeled data. In *Advances in NIPS*, pages 601–608.

Hutter, F., Kotthoff, L., and Vanschoren, J. (2019). *Automated machine learning: methods, systems, challenges*. Springer Nature.

Liu, A. and Ziebart, B. (2014). Robust classification under sample selection bias. In *Advances in neural information processing systems*, pages 37–45.

Lorena, A. C., De Carvalho, A. C., and Gama, J. M. (2008). A review on the combination of binary classifiers in multiclass problems. *Artificial Intelligence Review*, 30(1-4):19.

Macià, N. and Bernadó-Mansilla, E. (2014). Towards uci+: A mindful repository design. *Information Sciences*, 261:237–262.

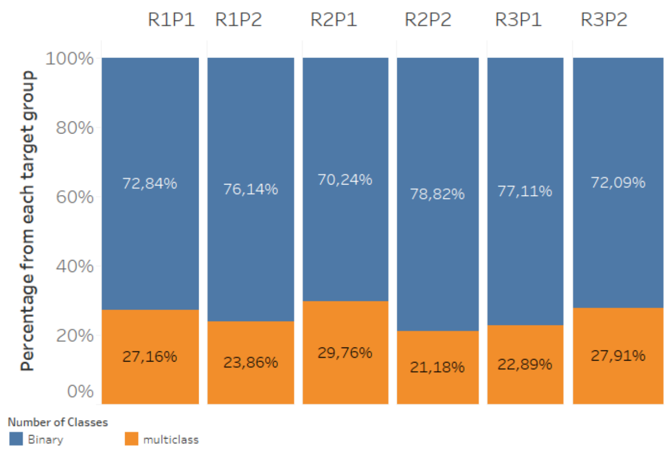

Figure 9. Number of classes proportion over random samples.

Maloney, C. (2017). Weapons of math destruction: How big data increases inequality and threatens democracy. *Journal of Markets & Morality*, 20(1):194–197.

Muñoz, M. A., Villanova, L., Baatar, D., and Smith-Miles, K. (2018). Instance spaces for machine learning classification. *Machine Learning*, 107(1):109–147.

Vanschoren, J. (2019). Meta-learning. In *Automated Machine Learning*, pages 35–61. Springer, Cham.

Vanschoren, J., Van Rijn, J. N., Bischl, B., and Torgo, L. (2014). Openml: networked science in machine learning. *ACM SIGKDD Explorations Newsletter*, 15(2):49–60.

Wallis, J. (2018). Is artificial intelligence sexist?, https://www.theglobeandmail.com/business/careers/leadership/article-is-artificial-intelligence-sexist/.

Zadrozny, B. (2004). Learning and evaluating classifiers under sample selection bias. In *Proceedings of the 21st international conference on Machine learning*, page 114.