Computer Methods and Programs in Biomedicine 242 (2023) 107803

Contents lists available at ScienceDirect

Computer Methods and Programs in Biomedicine

journal homepage: www.elsevier.com/locate/cmpb

Extremely missing numerical data in Electronic Health Records for machine learning can be managed through simple imputation methods considering informative missingness: A comparative of solutions in a COVID-19 mortality case study

Pablo Ferri [a,*], Nekane Romero-Garcia [b], Rafael Badenes [b,c,d], David Lora-Pablos [e,f], Teresa García Morales [e], Agustín Gómez de la Cámara [e], Juan M. García-Gómez [a], Carlos Sáez [a]

[a] Biomedical Data Science Lab, Instituto Universitario de Tecnologías de la Información y Comunicaciones, Universitat Politècnica de València, Camino de Vera s/n, Valencia 46022, Spain
[b] Departament de Cirugia, Universitat de València, Spain
[c] Instituto INCLIVA, Hospital Clínico Universitario de Valencia, Spain
[d] Department Anesthesiology, Surgical-Trauma Intensive Care and Pain Clinic, Hospital Clínic Universitari, Valencia, Spain
[e] Instituto de Investigación imas12, Hospital 12 de Octubre, Madrid, Spain
[f] Facultad de Estudios Estadísticos, Universidad Complutense de Madrid, Spain

ARTICLE INFO

Keywords:
Machine learning
Missing data
Data imputation
Informative missingness
Electronic health records
COVID-19

ABSTRACT

Background and objective: Reusing Electronic Health Records (EHRs) for Machine Learning (ML) leads on many occasions to extremely incomplete and sparse tabular datasets, which can hinder the model development processes and limit their performance and generalization. In this study, we aimed to characterize the most effective data imputation techniques and ML models for dealing with highly missing numerical data in EHRs, in the case where only a very limited number of data are complete, as opposed to the usual case of having a reduced number of missing values.

Methods: We used a case study including full blood count laboratory data, demographic and survival data in the context of COVID-19 hospital admissions and evaluated 30 processing pipelines combining imputation methods with ML classifiers. The imputation methods included missing mask, translation and encoding, mean imputation, k-nearest neighbors' imputation, Bayesian ridge regression imputation and generative adversarial imputation networks. The classifiers included k-nearest neighbors, logistic regression, random forest, gradient boosting and deep multilayer perceptron.

Results: Our results suggest that in the presence of highly missing data, combining translation and encoding imputation—which considers informative missingness—with tree ensemble classifiers—random forest and gradient boosting—is a sensible choice when aiming to maximize performance, in terms of area under curve.

Conclusions: Based on our findings, we recommend the consideration of this imputer-classifier configuration when constructing models in the presence of extremely incomplete numerical data in EHR.

1. Introduction

Electronic Health Records (EHRs) are a rich source of data for Machine Learning (ML) algorithms, as they contain a comprehensive record of each patient's medical history [1]. However, real-world data present huge challenges potentially limiting the successful development and prospective use of ML models, where incompleteness is among the most frequent issues [2]. Missing values in EHRs can be originated from human errors during the data entry process, patient reluctancy to provide specific information, the absence of results for specific tests [3], potential errors in complex data extraction pipelines—including mapping from the original EHRs vendors data models to the required format for data reuse—or to different patient's clinical workflows on a wide casuistry from which to generate the target data.

* Corresponding author.
E-mail address: pabferb2@upv.es (P. Ferri).

https://doi.org/10.1016/j.cmpb.2023.107803
Received 21 March 2023; Received in revised form 28 August 2023; Accepted 5 September 2023
Available online 7 September 2023

The missing value challenge, pervasive in EHRs, may severely impact ML outcomes. If no action is taken, serious bias may be introduced, potentially deriving in poor model performance, which may hinder medical data decision processes assisted by ML models [4]. On the other hand, improper data imputation processes may lead to inaccurate predictions, introducing spurious patterns and compromising the model's generalizability, potentially jeopardizing the safety and efficacy of clinical decision-making based on the ML model's output [5]. Additionally, special warning must be put in no imputing jointly training and test data, since this will lead to falsely overestimated performance metrics. Hence, this incompleteness issue must be tackled in a proper way when developing ML models from EHR data.

The nature of missing values conditions the approaches to deal with them. We can group these approaches into two categories: strategies based on discarding incomplete information and strategies focused on imputing the missing values [6]. Dropping either the case or the feature itself may be a straightforward action to handle the missing values, effective if there is no pattern associated with them and if the number of missing values is low. On the other hand, imputation techniques are based on filling the missing values with a value calculated from the ones that are complete in our data, considering univariate to multivariate approaches to calculate these imputed values [6,7].

The missing value problem becomes tougher as the incompleteness ratio increases. When this ratio is high, dropping rows or columns in our dataset may imply losing almost all the data, leading to data scarcity and being completely useless since novel incoming cases will present missing values with high probability. Thus, one must rely on data imputation approaches. However, the choice of a specific imputation strategy when data incompleteness is high is not easy, especially if one considers that some techniques may be more suited to specific ML predictive algorithms than others.

In the present work, we aimed to identify the most effective missing imputation techniques and ML models for dealing with numerical missing values in EHRs when data incompleteness is high. In contrast to using synthetic data or artificially introducing random missingness on a dataset, we considered a problem-representative case study of real-world EHR data from COVID-19 patients, with a high number of missing values, more realistic and potentially not-random, and the potential usage of the developed ML models in medical decision-making pipelines.

Although there is a vast number of studies dealing with data incompleteness in EHR data and COVID-19 mortality prediction [3,4,8-12], we aim to address an incompleteness ratio of around 95%, which is uncommon to find in the literature, although significantly present in real-world data analyses. Likewise, we have included in our study novel deep learning [13] approaches, including adversarial generation-discrimination for data imputation, and deep classifier ML models. Our study is unique in its challenging aim, with potentially useful conclusions for health artificial intelligence and ML researchers dealing with high incompleteness in EHR.

2. Materials

2.1. Dataset

2.1.1. Overview

We used 35 411 cases of patients at emergency admission with SARS-CoV2 infection from *Hospital 12 de Octubre*, Madrid, Spain, and *Hospital Clínico Universitario de Valencia*, Spain. The dataset comprised 22 variables including demographics, physical exploration, full blood count laboratory tests and patient survival. The use of data was approved by the Ethical Committees of the two hospitals and *Universitat Politècnica de València*.

2.1.2. Data quality assessment

To avoid the inclusion of potentially faulty data—besides missing data, the aim of our study—consistency bounds were established for each selected feature. Specifically, no negative values were allowed, nor percentage values above 100%. Those values labeled as erroneous were set to missing, but without dropping the entire case.

2.1.3. Data summary

Table 1 summarizes the numerical variables in our dataset and their completeness level, and Table 2 summarizes the categorical variables, including the target-dependent variable 30-day mortality. Data summary per hospital can be found in *Supplementary material*.

Tables 1 and 2 stand out for the high overall incompleteness, with some variables with a completeness ratio of around 5%, as well as the huge class imbalance present in our target variable.

2.2. Framework

The implementation language was Python [14], using the libraries Numpy [15] and Pandas [16] for data management, Scikit-learn [17] and PyTorch [18] for modeling and Optuna [19] for hyperparameter tuning.

3. Methods

3.1. Data preparation

3.1.1. Data splitting

First, data was split using a holdout [20] methodology, with proportions of 80% for training and 20% for testing. Next, k-fold cross-validation [20] splits were carried out over the training set, taking $k = 4$, without repetition—i.e., exhaustive evaluation—deriving in four data pairs each one with proportions 75% for training and 25% for validation.

To avoid potential overfitting issues and truly validate imputers and models performances, posterior preprocessing, imputing and modeling operations were carried out taking into account the nature of the data group. Hence, no retraining of scaling bounds, imputers or models was conducted in the validation or test sets; the configurations used were the ones previously learned in their respective training sets.

Table 1
Numerical variables in the studied COVID-19 dataset.

Variable name	Completeness (%)	5-th percentile	Median	95-th percentile
Age (years)	100	12.75	49.74	89.26
Heart rate (BPM)	20.75	60	89	124
Systolic BP (mmHg)	21.14	94	126	168
Diastolic BP (mmHg)	21.13	48	71	97
Temperature (°C)	22.24	35.9	36.9	39
CO_2 pressure (mmHg)	5.4	25	34	52
O_2 pressure (mmHg)	5.31	25	64	141
Red blood cells (million/mcL)	35.03	3.25	4.56	5.53
Hematocrit (%)	34.03	29.7	40.8	48.8
Hemoglobin (g/dL)	34.48	9.8	13.8	16,6
Platelets (million/mcL)	30.83	0.12	0.23	0.43
White blood cells (cells/mcL)	35.98	3.9	7.8	17.1
Eosinophils (%)	20.53	0.1	0.7	4.6
Lymphocytes (%)	28.18	4.5	17.4	38.6
Monocytes (%)	27.26	3.3	7.7	14.2
Neutrophils (%)	29.21	48.5	72.5	89.9
Bicarbonate (mEq/L)	5.28	18	23	32
Chloride (mEq/L)	31.82	90	99	106
Potassium (mmol/L)	6.21	3.42	4.14	5.22
Sodium (mEq/L)	6.35	130	138	144

Table 2
Categorical variables in the studied COVID-19 dataset.

Variable name	Completeness (%)	Category name	Frequency (%)
Sex	100	Male	42.3
		Female	57.7
30-day mortality (target variable for prediction)	100	Survival	94.9
		Exitus	5.1

3.1.2. Data preprocessing

Regarding categorical variables, sex was dummy encoded, dropping the redundant variable of each set, while the label 30-day mortality was one-hot encoded, creating a new variable for each label class.

Focusing on numerical features, we scaled our data to improve imputers and ML models performances, reduce the effect of outliers and accelerate the learning processes. Specifically, we considered two approaches: robust scaling—suitable for imputers not requiring strictly bounded inputs—and min-max scaling—appropriate for imputers requiring data to strictly belong to the [0, 1] interval.

Specifically, the equation for the robust scaling of each feature p was the following one:

$$x_s^{(p)} = \frac{x_o^{(p)} - q_{50}^{(p)}}{q_{75}^{(p)} - q_{25}^{(p)}} \tag{1}$$

where $x_s^{(p)}$ refers to the scaled feature value, $x_o^{(p)}$ to the original feature value, $q_{50}^{(p)}$ is the median of feature p, $q_{75}^{(p)}$ is the third quartile of feature p and $q_{25}^{(p)}$ is the first quartile of feature p.

Likewise, the formula for min-max scaling applied over each feature p was:

$$x_s^{(p)} = \begin{cases} 0, \ x_o^{(p)} < l_{2.5}^{(p)} \\ \dfrac{x_o^{(p)} - l_{2.5}^{(p)}}{u_{97.5}^{(p)} - l_{2.5}^{(p)}}, \ l_{2.5}^{(p)} \leq x_o^{(p)} \leq u_{97.5}^{(p)} \\ 1, \ x_o^{(p)} > u_{97.5}^{(p)}, \end{cases} \tag{2}$$

where $x_s^{(p)}$ refers to the scaled feature value, $x_o^{(p)}$ to the original feature value, $l_{2.5}^{(p)}$ to the 2.5-th percentile and $u_{97.5}^{(p)}$ to the 97.5-th percentile.

3.2. Imputation methods

We have examined six different imputation techniques, which we will now present in order from lower to higher complexity. Among these techniques, some are univariate techniques, utilizing only values from the same feature to perform imputation. On the other hand, others are multivariate, taking into account information from multiple features simultaneously. Despite the increased complexity of the multivariate methods, they can offer additional benefits when there are relationships among features that can be leveraged for data imputation.

The source code that has been developed to implement these strategies, along with the preceding data preparation procedures and subsequent classification models, can be accessed via the following link: https://github.com/bdslab-upv/extremiss.git.

3.2.1. Missing mask

Baseline imputation technique that we consider given its simplicity. It consists of the replacement of the missing values by a value of zero and the filled values by a value of one, in those features presenting missing values. Hence, for each feature p in our data:

$$x_i^{(p)} = \begin{cases} 0, \ if \ x_s^{(p)} is \ missing \\ 1, \ if \ x_s^{(p)} is \ not \ missing \end{cases} \tag{3}$$

being $x_i^{(p)}$ the imputed value for feature p and $x_s^{(p)}$ the scaled value for feature p.

3.2.2. Translation and encoding

Based on the hypothesis that extremely missing data in numerical features may introduce noise, thus impeding the imputation process, we propose here an imputing approach termed *translation and encoding*. In this method, non-missing numerical data features undergo translation and then missing values in these numerical features are encoded.

Specifically, this approach is a univariate imputation method, with the initial step involving the scaling of numerical values using a min-max scaling scheme—as defined in the previous section—to normalize them within the interval [0, 1]. Subsequently, an additive translation is applied, shifting all the values towards the right on the real number line. We defined that operation this way:

$$x_t^{(p)} := x_s^{(p)} + \delta, \ \delta \in \mathbb{R} \ \ni 0 < \delta < 1, \tag{4}$$

where $x_t^{(p)}$ is the translated feature value and δ is a positive real scalar, constrained to be below 1, that is added to execute the translation operation. It is worth noting that tuning this parameter δ in a validation set is recommended to optimize model performance. While each dataset may require its own tuning, we suggest initially evaluating values in the range of (0.1, 0.5).

Following the translation step, missing values are encoded using the value 0. This choice of 0 as the encoding value is efficient for many models, such as neural network models, where 0 values are not considered in the forward pass, and they do not affect any partial derivatives of the loss function with respect to the weights in the backward pass. As a result, this value can be reserved exclusively for encoding missing values.

By using this encoding scheme, informative missingness, if present, is directly captured, as a specific value (0) is employed to represent missing values. Unlike the missing mask approach, our method preserves information about the original feature domain, as we have solely applied linear operations in the process. This ensures that the encoded missing values still align with the overall characteristics of the feature and are distinguishable from non-missing values in subsequent model training and evaluation.

3.2.3. Mean

Univariate imputing strategy. We have considered it since it is widely used while its implementation is simple. For each feature p, it consists of replacing the missing values with the mean value along each feature. Hence, it can be described this way:

$$x_i^{(p)} = \overline{x}_s^{(p)} \tag{5}$$

being $x_i^{(p)}$ the imputed value for feature p and $\overline{x}_s^{(p)}$ the sample mean of the non-missing values for feature p.

3.2.4. K-nearest neighbors

Multivariate imputation strategy consisting of finding the K-nearest neighbors [21] for each missing feature of a specific observation, based on the other observation features which are not missing, and then averaging the values of those neighboring points presenting a non-missing value in the feature we aim to fill. Hence, for each missing value associated to a feature p:

$$x_i^{(p)} = \overline{x}_K^{(p)} \tag{6}$$

being $x_i^{(p)}$ the imputed value, and $\overline{x}_K^{(p)}$ the average value along feature p of the K nearest neighbors in terms of Euclidean distance, calculated using the filled values. We selected this approach due to its non-parametric nature, meaning it does not assume any underlying data distribution. Moreover, it leverages local information from the data

rather than employing a global pattern approach, which is the case for the majority of imputation techniques.

3.2.5. Bayesian ridge regression

Bayesian ridge regression [22] is a multivariate imputation strategy that addresses missing values in each feature by modeling them using the available remaining features in a round-robin fashion [23]. Unlike point estimates, it employs a distributional approach—a Bayesian paradigm—to handle imputations. Furthermore, this method demonstrates robustness in the face of feature multicollinearity, thanks to the inclusion of the ridge penalty [24]. We chose this approach because it provides a multivariate perspective while maintaining a relatively straightforward structure as a linear model, all the while effectively handling multicollinearity concerns.

3.2.6. Generative adversarial imputation networks

Generative adversarial imputation networks constitute a multivariate imputation method based on generative adversarial neural networks [25], which are grounded in deep learning principles [13]. In this approach, the generator examines certain features of the input data, imputes the missing components based on the observed information, and then produces a complete observation. Conversely, the discriminator analyzes this observation and tries to distinguish between the features that were genuinely observed and those that were imputed [26]. We assessed this approach because generative adversarial imputation networks possess the capability to handle complex data structures and effectively capture non-linear dependencies. These attributes are particularly valuable when aiming to achieve high-quality imputations.

3.3. Classification

Five machine learning classifier families were considered, each with increasing implementation complexity with respect the previous one: K-nearest neighbors [21], logistic regression [27], random forest [28], gradient boosting [29] models and deep multi-layer perceptron [13,30].

K-nearest neighbors and logistic regression models were considered for their simplicity of implementation, interpretability and explainability capabilities, and their ability to serve as baseline references for performance evaluation. The random forest and gradient boosting models were adopted as they attain excellent results in many previous works within the COVID-19 triage literature [10,11,31,32]. In addition, they are inherently interpretable and explainable, attributes that confer value within the medical field, as evidenced by [11]. Lastly, despite not receiving as much widespread consideration as the aforementioned tree ensemble methods, deep multi-layer perceptrons were employed due to the fact that deep learning is at the forefront of numerous highly intricate artificial intelligence tasks [33,34,35]. Therefore, we hypothesize that incorporating them could contribute additional predictive value compared to other methods.

3.3.1. K-nearest neighbors

K-nearest neighbors is a lazy and non-parametric learning approach that does not assume any underlying data distribution. It is capable of capturing complex non-linear dependencies within the data. The method operates by examining the K nearest instances in the training set to a given observation, utilizing a distance metric such as the Euclidean distance in our case. Subsequently, it assigns the class label to the observation based on the class that is most prevalent among its K nearest neighbors.

3.3.2. Logistic regression

Logistic regression is a linear and parametric model approach that is both efficient to train and valuable for establishing baseline performance. The model constructs predictions by applying a sigmoid function to a linear combination of the features, including a bias term in the process. This sigmoid transformation allows logistic regression to predict probabilities and classify data into different classes. The simplicity and interpretability of logistic regression make it a widely used method for binary classification tasks.

3.3.3. Random forest

Random forest is a tree ensemble model that effectively captures non-linear dependencies within the data. It accomplishes this by combining multiple decision trees. One of the key advantages of random forests is their ability to reduce prediction error variance rather than bias [28], making them robust and less prone to overfitting. During the training process, particular emphasis was given to two crucial hyperparameters: the number of weak learners (i.e., decision trees) and the depth of each tree. We determined the impact of these parameters on model performance and selected an appropriate combination that yielded the best results, following the approach outlined in Section 3.4.

3.3.4. Gradient boosting

Gradient boosting is a tree ensemble model that effectively captures non-linear dependencies in the data. It places more emphasis on reducing prediction error bias rather than variance [29]. To construct the trees in the ensemble, we used the mean squared error with an improvement score as proposed by Friedman [29] as the splitting criteria. Similarly, we thoroughly analyzed the influence of the number of decision trees and tree depth, employing the hyperparameter optimization strategy described in Section 3.4.

3.3.5. Deep multi-layer perceptron

The deep multi-layer perceptron is a differentiable model rooted in artificial neural networks, composed of multiple processing layers intended to learn data representations at various levels through an end-to-end approach [13]. Training of this model relies on gradient descent algorithms [36], where gradients are computed using backpropagation [37].

The implemented architecture was based on a multi-layer perceptron [30], comprising dense and output blocks. A dense block integrates a fully connected layer [38], a layer normalization [39] layer to manage internal covariate shift, a ReLU activation function to prevent exploding activations [40], and a dropout layer [41] to minimize neuron co-adaptation. An output block is composed of a fully connected layer and a softmax activation function, providing a normalized score ranging from 0 to 1 for each class within the 30-day COVID-19 mortality label.

Regarding model training, the optimizer considered was AdamW, due to its learning adaptability, noisy gradients management and learning process stability [36,42]. We took as loss function the focal cross-entropy loss [43], given its smoother behavior with respect cross-entropy. This loss function was weighted considering class frequency, to reduce the effect of class imbalance, and regularized with weight decay [44], to reduce the risk of overfitting. Likewise, a mini-batch training approach was adopted, to dispose of a trade-off between proper weight learning and efficiency in terms of computation time and memory. Finally, the weights of layers with a ReLU activation function were initialized with Kaiming's initialization [45], while layers with a softmax activation function were initialized with Xavier's initialization [46].

3.4. Hyperparameter tuning

In this section, we present the hyperparameter optimization scheme employed to determine the optimal combinations of hyperparameters for imputers and classifiers that heavily rely on their values. Therefore, the methodology outlined here was separately applied to the translation and encoding imputer, the k-nearest neighbors imputer, the generative adversarial imputer, the k-nearest neighbors classifier, the random forest classifier, the gradient boosting classifier, and the deep multi-layer perceptron classifier.

Imputer's and ML models' hyperparameters—e.g., translation scalar,

number of neighbors, number trees, tree depth, learning rate, batch size, networks architecture, etc.—were selected following an active learning [47] Bayesian multi-step hyperparameter optimization strategy, similarly to previous works [34]. This consists in a procedure where an auxiliary probabilistic model—a generative model—was trained iteratively to 1) estimate the probability of the objective performance metric given a set of hyperparameters and 2) sample new hyperparameter values on each iteration expecting to improve the performance metric.

Thus, multiple surrogate models, which in our work consisted of tree-structured Parzen estimators [48], were iteratively updated. The allowed sampling space was discrete, since a continuous sampling space may derive in overfitting issues due to the curse of dimensionality [49]. Likewise, the performance metric considered in our work was the area under curve (AUC) since it takes into account class imbalance and it is not dependent on a specific threshold [50].

Finally, it must be pointed out that the decision threshold to translate predicted class probabilities to a specific class was set maximizing using the Youden index [51]. We chose this rule to balance recall (sensitivity) and specificity. Likewise, to allow an unbiased evaluation, this threshold was calculated in the training set, without using any information from

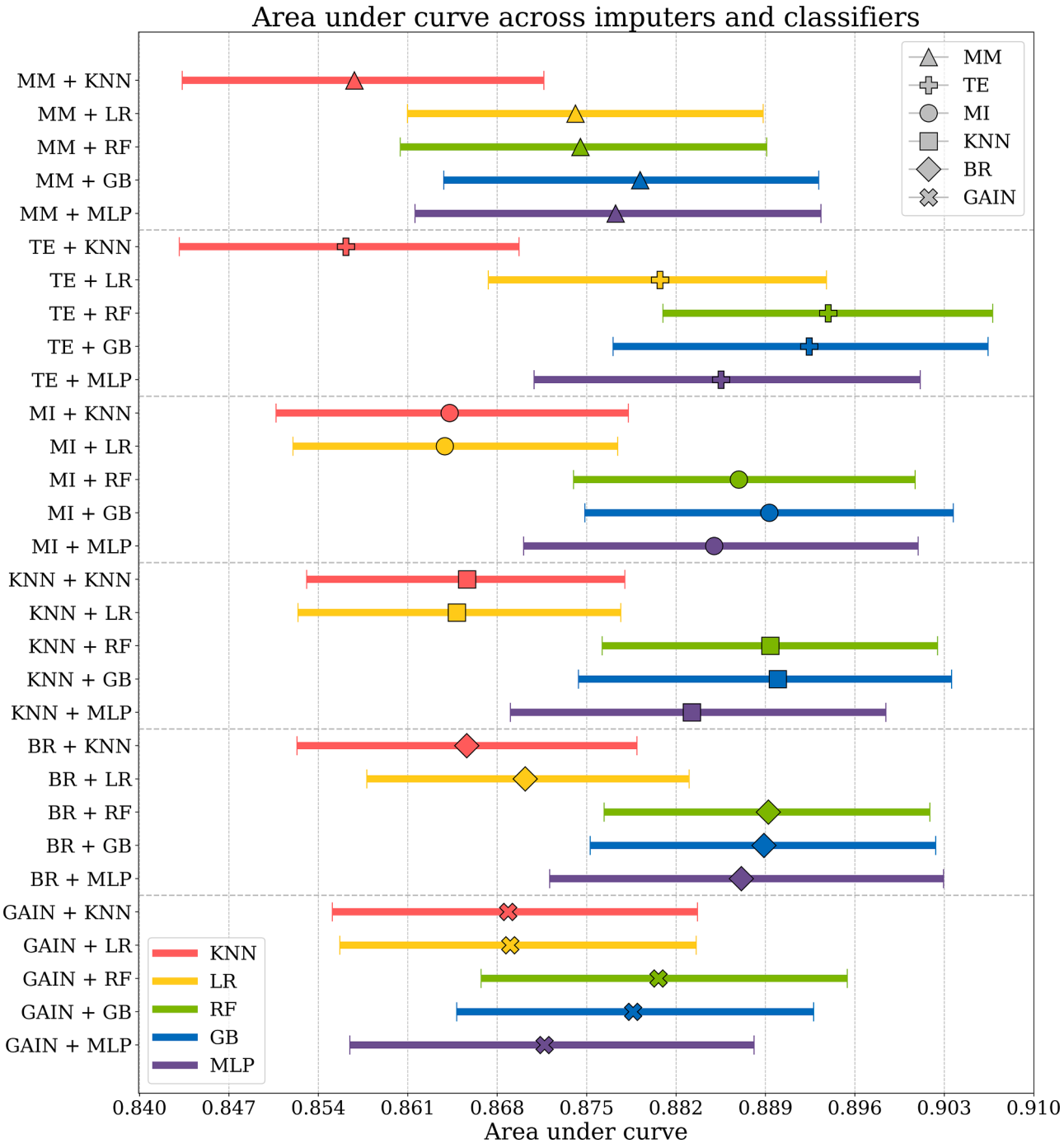

Fig. 1. Areas under curve for 30-day COVID-19 mortality prediction in the test set for the different imputer and classifier configurations. Confidence intervals (95%) obtained with bootstrap are also provided. Each imputation method is represented with a different marker while each classifier by a different color.
Abbreviations: MM, missing mask; TE, translation and encoding; MI, mean imputation; KNN, k-nearest neighbors; BR, Bayesian regression; GAIN, generative adversarial imputation networks; LR, logistic regression; RF, random forest; GB, gradient boosting; MLP, multi-layer perceptron.

the test set to estimate it.

3.5. Evaluation

We calculated the AUC, recall (sensitivity) and specificity in the test set to assess the performance of the models developed. A total of 1000 resampling operations based on bootstrap [52] were carried out to obtain non-parametric distributions for each metric value. Likewise, the 95% confidence intervals for each metric were calculated from these empirical distributions.

4. Results

Next, we show the 30-day COVID-19 mortality prediction performance for the different imputers and classifiers in the test set, in terms of AUC, recall and specificity, considering the metric values calculated as well as the confidence intervals obtained with bootstrap. Additional graphics representing these metrics can be found in *Supplementary material*.

Fig. 1 shows that missing mask—the baseline imputation method—achieves reasonably good results, although is not present among the

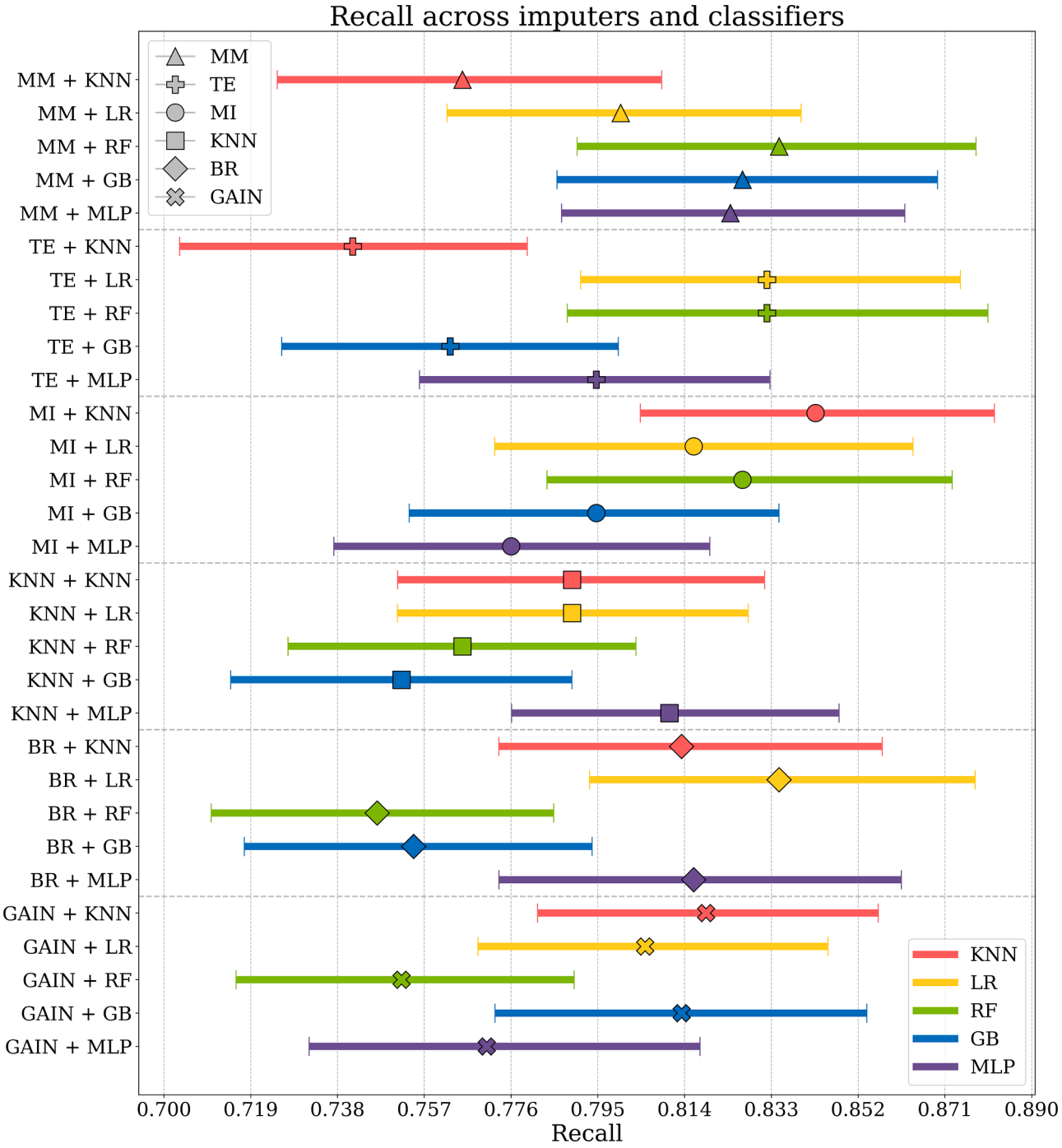

Fig. 2. Recalls for 30-day COVID-19 mortality prediction in the test set for the different imputer and classifier configurations. Confidence intervals (95%) obtained with bootstrap are also provided. Each imputation method is represented with a different marker while each classifier by a different color.
Abbreviations: MM, missing mask; TE, translation and encoding; MI, mean imputation; KNN, k-nearest neighbors; BR, Bayesian regression; GAIN, generative adversarial imputation networks; LR, logistic regression; RF, random forest; GB, gradient boosting; MLP, multi-layer perceptron.

top imputer-classifier configurations. Translation and encoding—the imputing approach proposed in this work to consider informative missingness while keeping information about the original feature domain—attains the best results, specifically when combined with the ensemble tree models, getting an AUC of 0.894 with the random forest classifier, the highest of our work. Mean imputation, K-nearest neighbors' imputation and Bayesian regression produce better results than the missing mask baseline but fail to surpass the translation and encoding approach. Finally, the generative adversarial imputation method, the most complex imputation strategy, although above the missing mask approach, does not present imputer-classifier configurations at the top of the performance scale.

Fig. 2 shows that the missing mask baseline attains better recall values than most other imputation methods, except mean imputation which offers the best recall in conjunction with the KNN classifier. Noteworthy, the translation encoding strategy, the Bayesian regression imputer and the generative adversarial imputation scheme present similar recall values, but the variance in them is lower for the generative adversarial method. Finally, the K-nearest neighbors imputer is the one with the worst overall recall metrics.

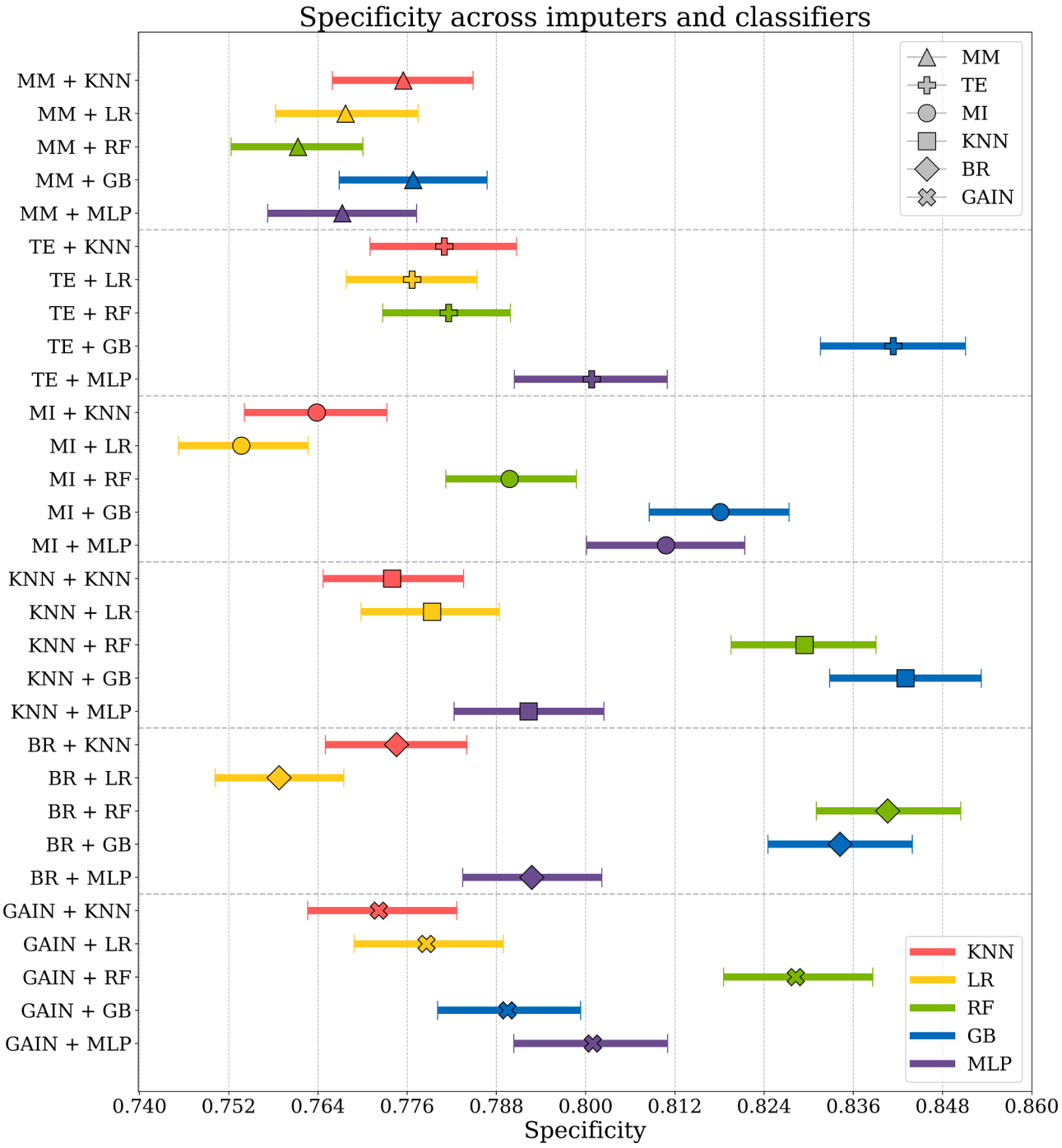

Fig. 3. Specificities for 30-day COVID-19 mortality prediction in the test set for the different imputer and classifier configurations. Confidence intervals (95%) obtained with bootstrap are also provided. Each imputation method is represented with a different marker while each classifier by a different color.
Abbreviations: MM, missing mask; TE, translation and encoding; MI, mean imputation; KNN, k-nearest neighbors; BR, Bayesian regression; GAIN, generative adversarial imputation networks; LR, logistic regression; RF, random forest; GB, gradient boosting; MLP, multi-layer perceptron.

Fig. 3 shows a reversed behavior respect Fig. 3, with missing mask offering low specificity values, followed by mean imputation. Likewise, translation and encoding, K-nearest neighbors, Bayesian regression and generative adversarial imputation provide significantly better outcomes, especially when combined with decision tree ensemble models. Finally, it stands out the variability in the specificity value metrics, notably superior to the ones appreciated in the AUC and recall metrics.

5. Discussion

5.1. Relevance

Results from our work show that is possible to deal with extremely missing numerical data on EHR without having to drop any case or column, and hence, without severely reducing the clinical applicability scenario to those cases presenting all feature values. This is of utmost importance, especially in the clinical domain, since medicine needs to deal with incomplete information by nature. Likewise, focusing on the COVID-19 context, the pipelines developed can offer a data-based second opinion aiming to enhance the clinical decision process in the presence of highly sparse incoming data.

After analyzing what the evaluated imputers and classifiers offer and considering AUC as reference metric—recall and specificity have more room for adjusting and reaching a trade-off by changing the saturation threshold if the AUC is higher—the simple translation and encoding approach stands out as the best configuration, in conjunction with tree ensemble classifiers, especially with the random forest classifier.

The fact that missing mask achieves good results without considering the value itself may imply the presence of informative missingness, since patients presenting more severe clinical frameworks could require more tests and analysis. However, that comes at the cost of low specificities. On the other hand, translation and encoding performs better in respect this baseline because it considers this informative missingness but at the same time keeps the information about the feature domain, giving more detail to define the decision boundaries for the posterior classifiers.

Despite the sophistication of the generative adversarial imputing approach, it gets the second worst results in terms of average AUC, just after the missing mask approach. We hypothesize that this is due to the extreme numerical data sparseness, which introduces noise and highly hinders the learning process for this imputer. Although the remaining methods perform better than the generative approach, they do not surpass the translation and encoding approach. Thus, we can state that is an effective and efficient strategy that should be tested when dealing with data of the same nature as the one presented in this work.

Therefore, we recommend considering the evaluation of the translation and encoding approach, combined with tree ensemble models such as random forest or gradient boosting, when dealing with datasets that have highly missing data exhibiting similar characteristics. By similar characteristics, we mean the presence of extreme missing data in continuous features, without necessarily requiring explicit alignment with mortality prediction tasks. Conversely, datasets with different missing patterns, such as temporal missing data [53,54], may demand further investigation into the suitability of this method.

Moreover, we encourage researchers to adhere to the data splitting and preprocessing steps presented in this work, since improper data handling during imputation would lead to misleading results, as discussed in [55]. Hence, it is vital to exercise caution when carrying out these operations.

Thus, given its implementation simplicity and computational efficiency, it is worth assessing the translation and encoding approach combined with tree ensemble models, especially before considering more sophisticated alternatives that might not offer significant predictive value due to the extreme missing data present in the dataset.

5.2. Limitations

The main limitations of this work are associated with the inherent difficulty of predicting the 30-day COVID-19 mortality, since patients presenting similar feature values can derive a completely different outcome. Thus, this inter-patient variability, summed with the complexity of dealing with highly missing numerical data set bounds to the maximum performance attainable. However, as we aimed to compare different imputing and classifier approaches relatively, these limitations are not critical to carrying out our analyses. Lastly, we acknowledge that our findings are specific to our COVID-19 mortality prediction task, and although these could be representative enough on specific similar cases with extremely missing data situations, further experimentation would help validate the applicability of our results to other medical prediction scenarios.

5.3. Future work

As future work, we aim to extend this study to other real-world biomedical and EHR datasets especially those presenting similar missingness structures, to provide additional evidence of our findings. Likewise, we aim to study end-to-end imputation and classification pipelines, where the imputer and the classifier are trained synchronously, and not first the imputer and then the classifier.

6. Conclusions

In this work, we have compared the adequateness of multiple imputation and classification pipelines to handle extremely missing numerical data in real-world Electronic Health Records (EHRs). We have developed 30 imputation-classifier pipelines to predict COVID-19 mortality, considering features with missingness ratios up to 95%. Our results indicate that translation and encoding—univariate method that considers informative missingness—outperforms complex imputation methods especially when combined with random forest and gradient boosting models. Given our findings, we recommend considering translation with encoding when constructing models in the presence of extremely incomplete numerical data in EHRs.

Declaration of Competing Interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Acknowledgements

This work was supported by FONDO SUPERA COVID-19 by CRUE-Santander Bank grant “Severity Subgroup Discovery and Classification on COVID-19 Real World Data through Machine Learning and Data Quality assessment (SUBCOVERWD-19)” and by the Ministry of Science, Innovation and Universities of Spain program FPU18/06441.

Supplementary materials

Supplementary material associated with this article can be found, in the online version, at doi:10.1016/j.cmpb.2023.107803.

References

[1] S. Yang, P. Varghese, E. Stephenson, K. Tu, J. Gronsbell, Machine learning approaches for electronic health records phenotyping: a methodical review, J. Am. Med. Inform. Assoc. 00 (0) (2022).

[2] N. McCombe, et al., Practical strategies for extreme missing data imputation in dementia diagnosis, IEEE J. Biomed. Health Inform. 26 (2) (Feb. 2022) 818–827, https://doi.org/10.1109/JBHI.2021.3098511.

[3] J.M. Baron, K. Paranjape, T. Love, V. Sharma, D. Heaney, M. Prime, Development of a 'meta-model' to address missing data, predict patient-specific cancer survival and provide a foundation for clinical decision support, J. Am. Med. Inform. Assoc. 28 (3) (Mar. 2021) 605–615, https://doi.org/10.1093/jamia/ocaa254.

[4] J.G. Ibrahim, H. Chu, M.-H. Chen, Missing data in clinical studies: issues and methods, JCO 30 (26) (Sep. 2012) 3297–3303, https://doi.org/10.1200/JCO.2011.38.7589.

[5] B.J. Wells, K.M. Chagin, A.S. Nowacki, M.W. Kattan, Strategies for handling missing data in electronic health record derived data, EGEMS (Wash DC) 1 (3) (Dec. 2013) 1035, https://doi.org/10.13063/2327-9214.1035.

[6] A.K. Tripathi, G. Rathee, H. Saini, Taxonomy of missing data along with their handling methods, in: 2019 Fifth International Conference on Image Information Processing (ICIIP), Nov. 2019, pp. 463–468, https://doi.org/10.1109/ICIIP47207.2019.8985715.

[7] T. Emmanuel, T. Maupong, D. Mpoeleng, T. Semong, B. Mphago, O. Tabona, A survey on missing data in machine learning, J. Big Data 8 (1) (Oct. 2021) 140, https://doi.org/10.1186/s40537-021-00516-9.

[8] J. Li, M. Wang, M.S. Steinbach, V. Kumar, G.J. Simon, Don't do imputation: dealing with informative missing values in EHR data analysis, in: 2018 IEEE International Conference on Big Knowledge (ICBK), Nov. 2018, pp. 415–422, https://doi.org/10.1109/ICBK.2018.00062.

[9] M.M. Banoei, R. Dinparastisaleh, A.V. Zadeh, M. Mirsaeidi, Machine-learning-based COVID-19 mortality prediction model and identification of patients at low and high risk of dying, Crit. Care 25 (1) (Dec. 2021) 328, https://doi.org/10.1186/s13054-021-03749-5.

[10] D. Bertsimas, et al., COVID-19 mortality risk assessment: an international multi-center study, PLoS One 15 (12) (Dec. 2020), e0243262, https://doi.org/10.1371/journal.pone.0243262.

[11] Explainable machine learning for early assessment of COVID-19 risk prediction in emergency departments, IEEE Access. 8 (Oct. 2020) 196299–196325, https://doi.org/10.1109/ACCESS.2020.3034032.

[12] E. Casiraghi, et al., A method for comparing multiple imputation techniques: a case study on the U.S. national COVID cohort collaborative, J. Biomed. Inform. 139 (Mar. 2023), 104295, https://doi.org/10.1016/j.jbi.2023.104295.

[13] Y. LeCun, Y. Bengio, G. Hinton, Deep learning, Nature 521 (7553) (May 2015), https://doi.org/10.1038/nature14539. Art. no. 7553.

[14] G. van Rossum (Guido), Python Reference Manual, Department of Computer Science [CS], 1995 no. R 9525. CWI, Jan. 01,Accessed: Mar. 08, 2022. [Online]. Available, https://ir.cwi.nl/pub/5008.

[15] S. van der Walt, S.C. Colbert, G. Varoquaux, The NumPy array: a structure for efficient numerical computation, Comput. Sci. Eng. 13 (2) (Mar. 2011) 22–30, https://doi.org/10.1109/MCSE.2011.37.

[16] W. McKinney, Data structures for statistical computing in python, in: presented at the Python in Science Conference, Austin, Texas, 2010, pp. 56–61, https://doi.org/10.25080/Majora-92bf1922-00a.

[17] F. Pedregosa et al., "Scikit-learn: machine learning in python," *Machine Learning in Python*, p. 6.

[18] A. Paszke et al., "Automatic differentiation in PyTorch," p. 4.

[19] T. Akiba, S. Sano, T. Yanase, T. Ohta, M. Koyama, Optuna: a next-generation hyperparameter optimization framework, in: Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining, ACM, Anchorage AK USA, Jul. 2019, pp. 2623–2631, https://doi.org/10.1145/3292500.3330701.

[20] R. Kohavi, A study of cross-validation and bootstrap for accuracy estimation and model selection, Ijcai 14 (2) (1995) 1137–1145.

[21] T. Cover, P. Hart, Nearest neighbor pattern classification, IEEE Trans. Inf. Theory 13 (1) (Jan. 1967) 21–27, https://doi.org/10.1109/TIT.1967.1053964.

[22] M.E. Tipping, Sparse bayesian learning and the relevance vector machine, J. Mach. Learn. Res. 1 (Sep. 2001) 211–244, https://doi.org/10.1162/15324430152748236.

[23] J. Furnkranz, "Round ROBIN CLASSIfiCATion".

[24] T.A. Johansen, On Tikhonov regularization, bias and variance in nonlinear system identification, Automatica 33 (3) (Mar. 1997) 441–446, https://doi.org/10.1016/S0005-1098(96)00168-9.

[25] I. Goodfellow, et al., Generative adversarial networks, Commun. ACM 63 (11) (Oct. 2020) 139–144, https://doi.org/10.1145/3422622.

[26] J. Yoon, J. Jordon, and M. van der Schaar, "GAIN: missing data imputation using generative adversarial nets." arXiv, Jun. 07, 2018. Accessed: Dec. 14, 2022. [Online]. Available: http://arxiv.org/abs/1806.02920.

[27] J.A. Nelder, R.W.M. Wedderburn, Generalized linear models, J. R. Stat. Soc. Ser. A 135 (3) (1972) 370–384, https://doi.org/10.2307/2344614.

[28] T.K. Ho, Random decision forests, in: Proceedings of 3rd International Conference on Document Analysis and Recognition 1, Aug. 1995, pp. 278–282, https://doi.org/10.1109/ICDAR.1995.598994.

[29] J.H. Friedman, Greedy function approximation: a gradient boosting machine, Ann. Statis. 29 (5) (2001) 1189–1232.

[30] C. Van Der Malsburg, Frank rosenblatt: principles of neurodynamics: perceptrons and the theory of brain mechanisms, in: G. Palm, A. Aertsen (Eds.), Brain Theory, Springer, Berlin, Heidelberg, 1986, pp. 245–248, https://doi.org/10.1007/978-3-642-70911-1_20. Eds.

[31] M.C. Ottenhoff, et al., Predicting mortality of individual patients with COVID-19: a multicentre Dutch cohort, BMJ Open 11 (7) (Jul. 2021), e047347, https://doi.org/10.1136/bmjopen-2020-047347.

[32] C. Feng, G. Kephart, E. Juarez-Colunga, Predicting COVID-19 mortality risk in Toronto, Canada: a comparison of tree-based and regression-based machine learning methods, BMC Med. Res. Methodol. 21 (1) (Dec. 2021) 267, https://doi.org/10.1186/s12874-021-01441-4.

[33] D. Silver, et al., Mastering the game of Go with deep neural networks and tree search, Nature 529 (7587) (Jan. 2016), https://doi.org/10.1038/nature16961. Art. no. 7587.

[34] P. Ferri, et al., Deep ensemble multitask classification of emergency medical call incidents combining multimodal data improves emergency medical dispatch, Artif. Intell. Med 117 (Jul. 2021), 102088, https://doi.org/10.1016/j.artmed.2021.102088.

[35] J. Jumper, et al., Highly accurate protein structure prediction with AlphaFold, Nature 596 (7873) (Aug. 2021), https://doi.org/10.1038/s41586-021-03819-2. Art. no. 7873.

[36] S. Sun, Z. Cao, H. Zhu, and J. Zhao, "A Survey of Optimization Methods from a Machine Learning Perspective," *arXiv:1906.06821 [cs, math, stat]*, Oct. 2019, Accessed: Jan. 03, 2022. [Online]. Available: http://arxiv.org/abs/1906.06821.

[37] Hecht-Nielsen, Theory of the backpropagation neural network, in: International 1989 Joint Conference on Neural Networks 1, 1989, pp. 593–605, https://doi.org/10.1109/IJCNN.1989.118638.

[38] I. Goodfellow, Y. Bengio, A. Courville, Deep Learning, MIT Press, 2016.

[39] J.L. Ba, J.R. Kiros, and G.E. Hinton, "Layer Normalization," *arXiv:1607.06450 [cs, stat]*, Jul. 2016, Accessed: Jan. 03, 2022. [Online]. Available: http://arxiv.org/abs/1607.06450.

[40] A. Hannun et al., "Deep Speech: scaling up end-to-end speech recognition," *arXiv:1412.5567 [cs]*, Dec. 2014, Accessed: Jan. 03, 2022. [Online]. Available: http://arxiv.org/abs/1412.5567.

[41] G.E. Hinton, N. Srivastava, A. Krizhevsky, I. Sutskever, and R.R. Salakhutdinov, "Improving neural networks by preventing co-adaptation of feature detectors," *arXiv:1207.0580 [cs]*, Jul. 2012, Accessed: Mar. 16, 2022. [Online]. Available: http://arxiv.org/abs/1207.0580.

[42] S. Ruder, "An overview of gradient descent optimization algorithms," *arXiv:1609.04747 [cs]*, Jun. 2017, Accessed: Jan. 03, 2022. [Online]. Available: http://arxiv.org/abs/1609.04747.

[43] T.-Y. Lin, P. Goyal, R. Girshick, K. He, and P. Dollar, "Focal loss for dense object detection," p. 9.

[44] A. Krogh and J.A. Hertz, "A simple weight decay can improve generalization," p. 9.

[45] K. He, X. Zhang, S. Ren, and J. Sun, "Delving deep into rectifiers: surpassing human-level performance on imagenet classification," *arXiv:1502.01852 [cs]*, Feb. 2015, Accessed: Jan. 03, 2022. [Online]. Available: http://arxiv.org/abs/1502.01852.

[46] X. Glorot and Y. Bengio, "Understanding the difficulty of training deep feedforward neural networks," p. 8.

[47] B. Settles, "Active Learning Literature Survey," p. 47.

[48] J.S. Bergstra, R. Bardenet, Y. Bengio, and B. Kégl, "Algorithms for Hyper-Parameter Optimization," p. 9.

[49] R. Bellman, Dynamic programming and lagrange multipliers, Proc. Natl. Acad. Sci. USA. 42 (10) (Oct. 1956) 767–769.

[50] A.P. Bradley, The use of the area under the ROC curve in the evaluation of machine learning algorithms, Pattern Recognit. 30 (7) (Jul. 1997) 1145–1159, https://doi.org/10.1016/S0031-3203(96)00142-2.

[51] W.J. Youden, Index for rating diagnostic tests, Cancer 3 (1) (1950) 32–35, https://doi.org/10.1002/1097-0142(1950)3:1<32::AID-CNCR2820030106>3.0.CO;2-3.

[52] B. Efron, R.J. Tibshirani, An Introduction to the Bootstrap, CRC Press, 1994.

[53] Y. Luo, Evaluating the state of the art in missing data imputation for clinical data, Brief. Bioinformat. 23 (1) (Jan. 2022) bbab489, https://doi.org/10.1093/bib/bbab489.

[54] C. Sáez, O. Zurriaga, J. Pérez-Panadés, I. Melchor, M. Robles, J.M. García-Gómez, Applying probabilistic temporal and multisite data quality control methods to a public health mortality registry in Spain: a systematic approach to quality control of repositories, J. Am. Med. Inform. Assoc. 23 (6) (Nov. 2016) 1085–1095, https://doi.org/10.1093/jamia/ocw010.

[55] S. Ramachandra, G. Vandewiele, D.V. Mijnsbrugge, F. Ongenae, and S. Van Hoecke, "Perfectly predicting ICU length of stay: too good to be true." arXiv, Nov. 10, 2022. doi: 10.48550/arXiv.2211.05597.