RESEARCH

Project-2

Sujan Darai (sujad96@zedat.fu-berlin.de), Matanat Mammadli (matanam94@zedat.fu-berlin.de), Samra Hamidovic (samrah96@zedat.fu-berlin.de), Ibraimi Ibraim (ibraimii@hu-berlin.de)

Full list of author information is available at the end of the article

Abstract

Goal of the project: We aim to analyze original, dirty, imputed, and cleaned data, compare the model performance, and explore feature extraction.

Main results of the project: Model evaluation on dirty datasets was causing errors but not for imputed datasets. Imputed datasets perform well on training datasets but decrement in performance on adding imputation concentration.

Personal key learning:

1 Sujan Darai: Learned to make data dirty, data imputation using mean and median
2 Matanat Mammadli: Learned to describe methods and graphs of our results in a better way
3 Samra Hamidovic: Learned to deal with problematic code and about the concept of making data dirty

Estimated working hours:

1 Sujan Darai: 8 hours
2 Matanat Mammadli: 8 hours
3 Samra Hamidovic: 8 hours

Project evaluation: 1

No. of words: 1708 words.

Keywords: Metabolomics, Imputation, Support vector machine, Principle component regression

1 Introduction

Data quality is a fundamental aspect in the development of robust machine learning models, surpassing the significance of sophisticated algorithms. Missing data poses a common challenge in real-world datasets, exerting a substantial influence on the outcomes of predictive models.

In this research, our primary focus lies in comprehending the repercussions of unclean data on model performance, with a specific emphasis on investigating the efficacy of imputation strategies employing mean and median techniques. These approaches were chosen for their simplicity and widespread use in handling missing data. The dataset under examination in this study is, like in the previous project, the MTBLS136 dataset, encompassing a comprehensive serum LC-MS dataset linked to postmenopausal hormone use.

By examining the performance metrics of the models on both original and altered datasets, a better understanding for the role of data quality and the effectiveness of imputation methods in fixing the issue of missing data can be achieved.

Subsequently, the performance of two distinct models is we trained and evaluated: Principal Component Regression (PCR) and Linear Kernel Support Vector Machine (SVM). By intentionally manipulating the data, real-world scenarios can be simulized, where data quality might be compromised.

2 Goal of the project

The overarching goal of this study is to observe the impact of uncleaned data on model performance and compare the efficacy of mean and median imputation strategies. This analysis aims to provide valuable guidance on selecting the most suitable imputation strategy for handling missing data in real-world applications.

3 Data and preprocessings

We have chosen the same dataset as in Project One. The MTBLS136 dataset which consists of a serum LC-MS dataset. These datasets consist of 949 named metabolites associated with postmenopausal hormone use. Multiple metabolites were compared in the group of women including 332 estrogen users and 337 estrogen plus progestin users versus 667 non-users. In this research, metabolite levels were compared between estrogen-only and estrogen-plus progestin users. This means it consists of the metabolite concentration of users that is supposed to be due to estrogen-only versus estrogen-plus-progestin hormones. [1]

Like in project one, all the required packages were imported and additionally a new package called seaborn with a random seed of 18. The dataset was loaded in the jupyter notebook. The data is then divided into two tables, namely: DataTables and PeakTables using the function `cb.utils.load_dataXL`, like in project one. To get a better understanding of the given dataset, we observed the first 5 rows of the DataTable dataset and the PeakTable dataset. In the next step we checked how many null values are in each column in DataTable with `DataTable.isnull().sum()` and then printed the total sum of null values with `print(DataTable.isnull().sum().sum())`. The DataTable consists of 310432 null values in total. After that we provided a summary statistics of the data stored in a DataTable.

Now for the main step, we made the DataTable dirty by removing 5%, 10% and 15% of the values and imputed them with mean and median imputation. This part will is described in more detail in the methods part.

4 Methods

4.1 Process to make data dirty

We attempted to make data dirty by removing 5% of data. We randomly generated null values from 5% of the data. We did those same steps with 10% and 15% of the data as well, created randomly generated null values.

We also made data dirty by changing values randomly, generating random integers in the range between 1 and 900000000. First we did it with 5% of data, then with 10% of data and finally, with 15% of data.

4.2 Data imputation (mean and median technique)

We proceeded to clean our impure data by imputing the missing or changed values. First, we cleaned our data using mean imputation technique. For that we separated the dataframe into two subsets: one containing only numerical columns and the

other containing only non-numerical columns.
For each numerical column we filled missing values with the mean of that column. For each non-numerical column we filled missing values with the mode (most frequent value) of that column.
We then concatenated numerical and non-numerical columns back together along the columns axis, resulting in new dataframe. We did these steps for 5 %, 10 % and 15 % dirty data, that we created earlier.
We did same steps mentioned above, separated our dataframe into two subsets, but for our numerical column, we filled missing values with the median of that column. For non-numerical column we still used mode to fill the missing values.

4.3 Data cleaning

We cleaned our data by filtering out rows with a percentage of missing values less than 20%. We ensured that only rows with a relatively low percentage of missing data in the `Perc_missing` column are retained in the DataFrame PeakTableClean.

4.4 Hyperparameter optimization

We began by splitting our data into training and testing sets using the `train_test_split` function from scikit-learn. The training set ($DataTrain$) contains 2/3 of the data, while the test set (`DataTest`) contains 1/3 of the data. We also split the target variable Y accordingly. Following the split, we extracted the training and testing datasets separately. For each dataset, we applied a logarithmic transformation and then imputed missing values using k-nearest neighbors with k=3.
Then we did hyperparameter optimization for our linear SVM model using k-fold cross-validation and evaluated the model's performance using AUC and R^2Q^2 metric. Our hyperparameter here being optimized through cross-validation is C, which represents the regularization parameter for the Support Vector Machine (SVM) model. We plotted our model with AUC and R^2Q^2 metrics. This metric evaluates the goodness of fit and predictive ability of the model.

4.5 Bootstrap evaluation

After preprocessing the data, we trained a model on the preprocessed data, built bootstrap models, trained each bootstrap model, and evaluated their performance. First of all, natural logarithm transformation was applied to the predictor variables, then scaled version of the log-transformed predictor variables was computed, missing values in the scaled data (XBootScale) were imputed using k-nearest neighbors imputation with k=3. Then the model was trained and bootstrap procedure was initiated to create multiple bootstrap models (bootnum=100 specifies that 100 bootstrap models will be created). The bootstrap models are trained using the data generated from bootstrap samples. This step involves creating 100 bootstrap samples from the original data and training a model on each sample.
The performance of the bootstrap models was evaluated. This involves assessing the model's performance metrics, such as accuracy or error, on both the training and test datasets.
By generating multiple bootstrap samples and training models on each sample, we can estimate the uncertainty associated with the model's predictions and evaluate its generalization performance across different subsets of the data.

5 Results and Discussion

We have used PCR and SVM-lin methods only due to word limitation.

5.1 Evaluation using AUC-ROC

After removing the 5, 10, and 15 % of values from the data (i.e. replacing the original values with null values), different imputation techniques such as mean imputation and median imputation were applied to fill the null values. Initially, two machine learning algorithms such as PCR and SVM-Lin were applied first to missing datasets, (datasets having 5, 10, and 15 % removed values). The machine learning models crash because of the datasets' excessive null values. Then the machine learning models were performed on the imputed datasets. To measure the robustness and performance of the models, an area under ROC curves was measured for both training and testing datasets. Figures 1 and 2 show ROC curves of PCR models with different percentage concentrations of data imputed with mean and median values. The training datasets perform better than the testing datasets but the performances of both the training datasets seem to decrease with imputation concentration. The same phenomena can also be seen in the ROC curves of SVM-Lin as in Figures 3 and 4 with different imputation methods while the performances of testing datasets seem to be similar in all cases. The results from the ROC curves were overfitted on training datasets because of the many similar entries from imputation techniques.

5.2 Important metabolites

Important metabolites from two models with different imputation techniques such as mean and median were extracted and are shown in Tables 1, 2, 3, 4, 5, and 6. Although very different metabolites were predicted by the PCR and SVM-Lin models, 2-amino heptanoate metabolite is the only one predicted from both models. The main reason behind the prediction of different important metabolites could be the selection of machine learning algorithms chosen. There might be other common metabolites but here, in this case, only the top 3 were chosen which makes the result diverse. Cystine and 1-linoleoyl-GPA (18:2)* are the two important metabolites extracted from the PCR and SVM-Lin models respectively from almost all the imputation techniques. Metabolites such as cystine, and 2-amino heptanoate extracted from the two models make a significant impact on estrogen and estrogen plus progestin hormone function pathways. 1-linoleoyl-GPA (18:2) is a metabolite derived from linoleic acid and predicted by the SVM-Lin models and its concentration changes during the postmenopausal period in women and affects the estrogen reaction pathways. Other metabolites including S-methylcysteine, S-methylcysteine sulfoxide, lactosyl-N-behenoyl-sphingosine (d18:1/22:0)*, 2'-deoxyuridine, eugenol sulfate, and succinimide, etc were predicted as important metabolites, it also might affect the metabolic pathways but could possibly the results of imputation technique. To sum up, although a few similar important metabolites were extracted from both models, the imputation technique does not make the data cleaner but instead might introduce noise and screw the results.

Number	PCR	SVM-lin
1	cystine	2'-deoxyuridine
2	2-aminoheptanoate	2-aminoheptanoate
3	S-methylcysteine	1-linoleoyl-GPA (18:2)*

Table 1: Important features i.e. metabolites on 5% imputed data with mean imputation

Number	PCR	SVM-lin
1	S-methylcysteine	1-linoleoyl-GPA (18:2)*
2	S-methylcysteine sulfoxide	lactosyl-N-behenoyl-sphingosine (d18:1/22:0)*
3	cystine	succinimide

Table 2: Important features i.e. metabolites on 10% imputed data with mean imputation

Number	PCR	SVM-lin
1	2-aminoheptanoate	1-linoleoyl-GPA (18:2)*
2	cystine	androstenediol (3beta,17beta) disulfate (2)
3	eugenol sulfate	2-aminoheptanoate

Table 3: Important features i.e. metabolites on 15% imputed data with mean imputation

Number	PCR	SVM-lin
1	cystine	2'-deoxyuridine
2	S-methylcysteine	2-aminoheptanoate
3	carotene diol (3)	1-linoleoyl-GPA (18:2)*

Table 4: Important features i.e. metabolites on 5% imputed data with median imputation

Number	PCR	SVM-lin
1	S-methylcysteine	1-linoleoyl-GPA (18:2)*
2	cystine	lactosyl-N-behenoyl-sphingosine (d18:1/22:0)*
3	S-methylcysteine sulfoxide	succinimide

Table 5: Important features i.e. metabolites on 10% imputed data with median imputation

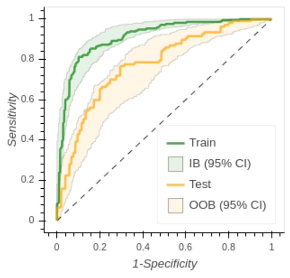
Number	PCR	SVM-lin
1	gluconate	1-linoleoyl-GPA (18:2)*
2	perfluorooctanesulfonic acid (PFOS)	2-aminoheptanoate
3	2-aminoheptanoate	androstenediol (3beta,17beta) disulfate (1)

Table 6: Important features i.e. metabolites on 15% imputed data with median imputation

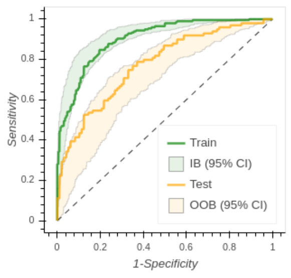
6 Contributions

Sujan Darai: running and implementing code
Matanat Mammadli: writing report (Methods, Results and Discussion)
Samra Hamidovic: running code and writing report (Goal, Data & preprocessing)
Ibraim Ibraimi: writing report (Abstract, Introduction), double check the report

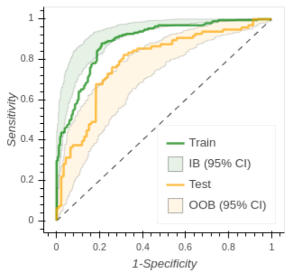
7 Appendix



(a) 5% imputed

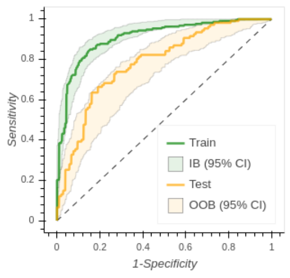


(b) 10% imputed

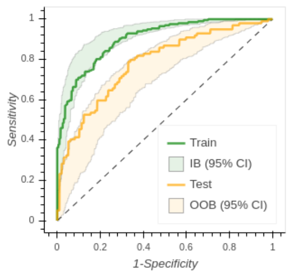


(c) 15% imputed

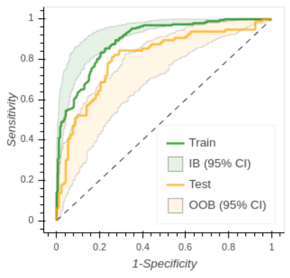
Figure 1: PCR method, imputed with mean values



(a) 5% imputed

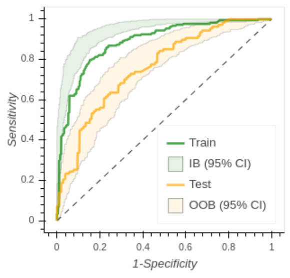


(b) 10% imputed

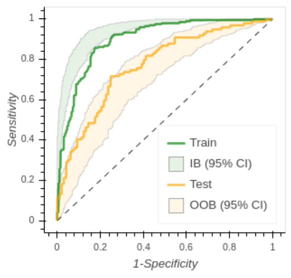


(c) 15% imputed

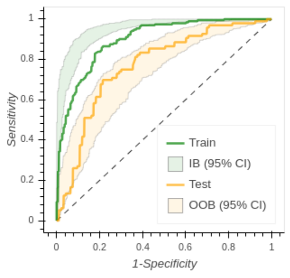
Figure 2: PCR method, imputed with median values



(a) 5% imputed

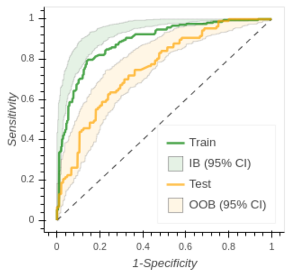


(b) 10% imputed

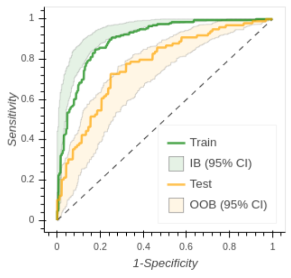


(c) 15% imputed

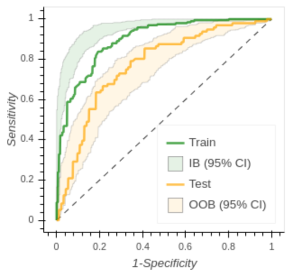
Figure 3: SVM-lin method, imputed with mean values



(a) 5% imputed



(b) 10% imputed



(c) 15% imputed

Figure 4: SVM-lin method, imputed with median values

References

1. Victoria L Stevens, Ying Wang, Brian D Carter, Mia M Gaudet, and Susan M Gapstur. Serum metabolomic profiles associated with postmenopausal hormone use. *Metabolomics*, 14:1–14, 2018.