Report v3

Marcelina Kurek 305741, Mateusz Stączek 305757, Jakub Wiśniewski 298850, Hanna Zdulska 298852

5 maja 2021

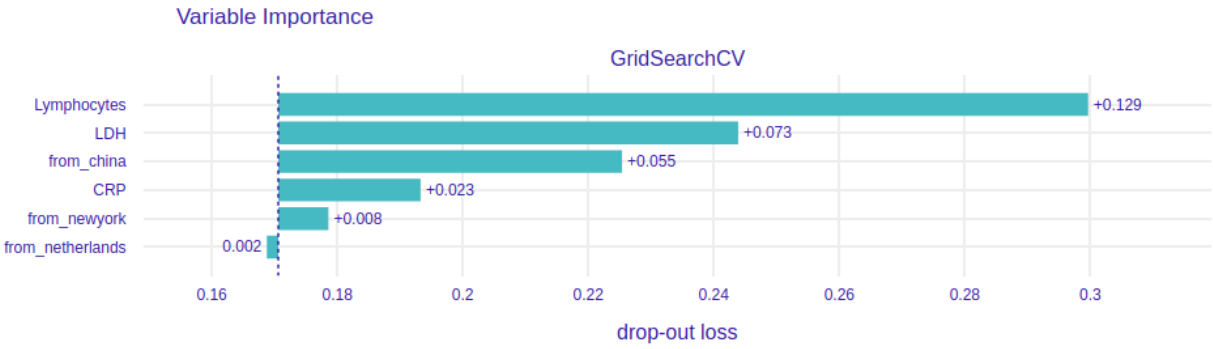
1 Introduction

Recently, we have been working with the model presented in the article Yan et al. (2020) Ąn interpretable mortality prediction model for COVID-19 patients". After performing external validation of this model on data from New York (Barish et al. (2021)) and Netherlands (Quanjel et al. (2021)), it became clear, that this model is not portable. Our next goal was to explore the possibility of building a model on multinational data. The goal was the same - to predict who would survive COVID-19, but we focused on the model performing well on data from every region of the world, not just one. In other words, we aimed to create a portable model.

2 Lazy Predict

To determinate the best model to train, we used Lazy Predict on two sets on data - one with continent of origin and second without it. The results are shown in Table 1 and Table 2.

To determinate how import was place of origin we trained RandomForestClassifier on data with said place. After grid search best model was with parameters max_depth 5 and n_estimators 50. It scored 0.72 on ROC AUC metric. We can see in image 2 two most important features are blood related. However third is information whether patient was from China or not.



Rysunek 1: Variable importance on RandomForestClassifier

Additionally using Baniecki et al. (2020) Dalex we created surrogate model to understand our model. Again samples originating from China were third most important factor.

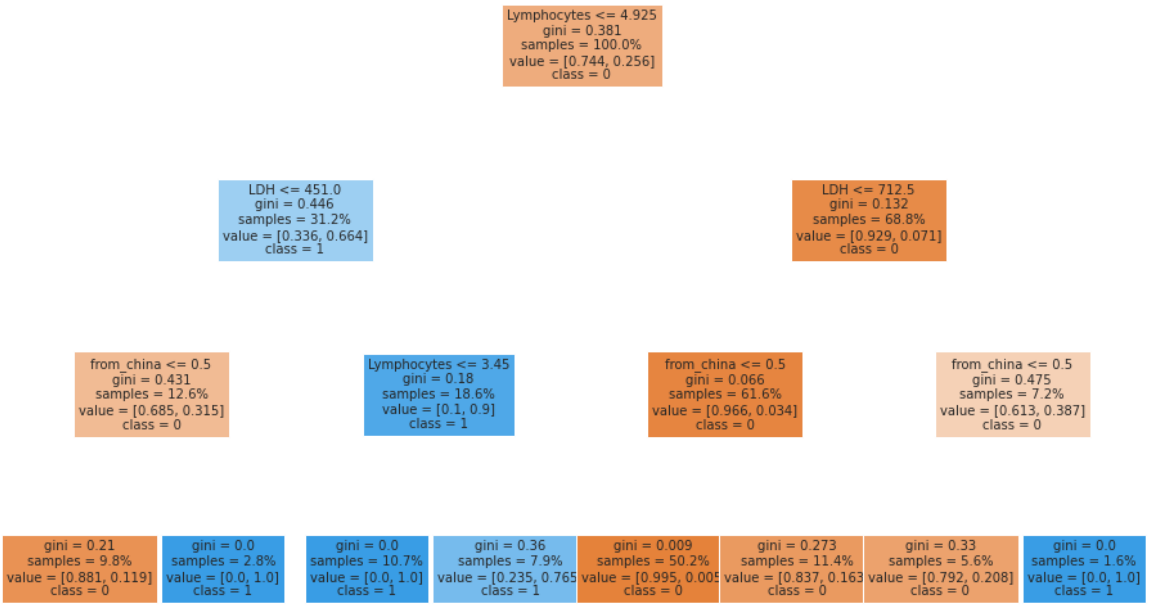
Additionally, we performed grid search for hyperparameter tuning on the 3 best models, considering data without country of origin. The highest ROC AUC score was reached by AdaBoostClassifier, NearestCentroid and KNeighborsClassifier. Hyperparameter optimisation changed the ROC AUC score from 0.71 to 0.80 for AdaBoostClassifier and from 0.69 to 0.84 for KNeighborsClassifier.

Model	Accuracy	Balanced Accuracy	ROC AUC	F1 Score	Time Taken
AdaBoostClassifier	0.753	0.714	0.714	0.751	0.191
NearestCentroid	0.700	0.702	0.702	0.708	0.023
KNeighborsClassifier	0.726	0.689	0.689	0.725	0.038
NuSVC	0.735	0.689	0.689	0.731	0.247
ExtraTreesClassifier	0.730	0.687	0.687	0.728	0.232
RandomForestClassifier	0.730	0.684	0.684	0.727	0.434
Perceptron	0.656	0.679	0.679	0.666	0.022
BaggingClassifier	0.733	0.673	0.673	0.724	0.076
LabelPropagation	0.719	0.668	0.668	0.714	0.157
LabelSpreading	0.721	0.668	0.668	0.715	0.228
LGBMClassifier	0.707	0.666	0.666	0.706	0.263
XGBClassifier	0.712	0.666	0.666	0.709	0.516
SVC	0.737	0.666	0.666	0.723	0.070
LogisticRegression	0.733	0.661	0.661	0.718	0.030
BernoulliNB	0.705	0.659	0.659	0.702	0.019
GaussianNB	0.702	0.647	0.647	0.696	0.019
LinearSVC	0.726	0.645	0.645	0.706	0.150
CalibratedClassifierCV	0.721	0.642	0.642	0.703	0.236
QuadraticDiscriminantAnalysis	0.716	0.640	0.640	0.700	0.020
LinearDiscriminantAnalysis	0.719	0.637	0.637	0.698	0.049
DecisionTreeClassifier	0.667	0.635	0.635	0.670	0.021
RidgeClassifier	0.721	0.633	0.633	0.697	0.032
RidgeClassifierCV	0.721	0.633	0.633	0.697	0.022
SGDClassifier	0.723	0.617	0.617	0.686	0.024
ExtraTreeClassifier	0.651	0.612	0.612	0.653	0.018
PassiveAggressiveClassifier	0.440	0.498	0.498	0.438	0.022
DummyClassifier	0.519	0.464	0.464	0.521	0.017

Tablica 1: LazyClassifier results on dataset without origin

Model	Accuracy	Balanced Accuracy	ROC AUC	F1 Score	Time Taken
RandomForestClassifier	0.784	0.740	0.740	0.779	0.326
XGBClassifier	0.770	0.735	0.735	0.768	0.223
LabelPropagation	0.767	0.729	0.729	0.765	0.114
BaggingClassifier	0.777	0.727	0.727	0.771	0.074
AdaBoostClassifier	0.767	0.726	0.726	0.764	0.185
LGBMClassifier	0.758	0.722	0.722	0.757	0.967
LabelSpreading	0.763	0.722	0.722	0.760	0.140
ExtraTreesClassifier	0.772	0.722	0.722	0.766	0.232
KNeighborsClassifier	0.758	0.715	0.715	0.755	0.042
NuSVC	0.779	0.708	0.708	0.765	0.076
GaussianNB	0.728	0.707	0.707	0.731	0.024
LogisticRegression	0.770	0.706	0.706	0.759	0.065
NearestCentroid	0.702	0.703	0.703	0.710	0.024
CalibratedClassifierCV	0.765	0.699	0.699	0.753	0.209
LinearDiscriminantAnalysis	0.772	0.699	0.699	0.757	0.041
SVC	0.774	0.699	0.699	0.758	0.071
ExtraTreeClassifier	0.733	0.698	0.698	0.732	0.036
LinearSVC	0.765	0.698	0.698	0.753	0.096
RidgeClassifier	0.765	0.684	0.684	0.746	0.035
RidgeClassifierCV	0.765	0.684	0.684	0.746	0.040
SGDClassifier	0.735	0.679	0.679	0.727	0.036
DecisionTreeClassifier	0.702	0.675	0.675	0.705	0.031
BernoulliNB	0.716	0.673	0.673	0.714	0.025
Perceptron	0.756	0.652	0.652	0.721	0.032
PassiveAggressiveClassifier	0.688	0.647	0.647	0.688	0.036
QuadraticDiscriminantAnalysis	0.649	0.525	0.525	0.591	0.030
DummyClassifier	0.519	0.464	0.464	0.521	0.026

Tablica 2: LazyClassifier results on dataset with origin



Rysunek 2: Surrogate model for RandomForestClassifier

3 Attempting to obtain a fair classifier

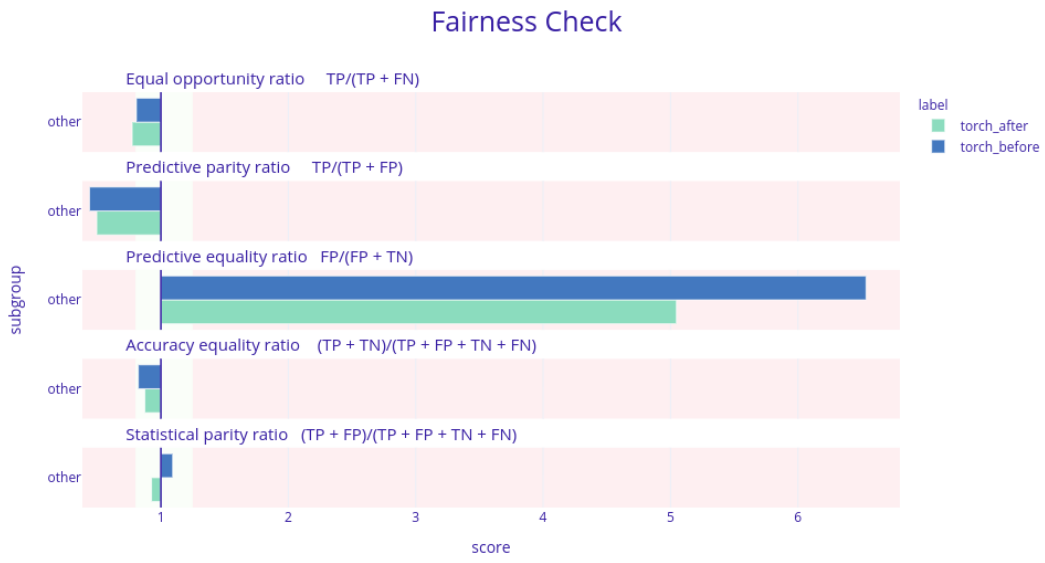
We attempted to train a fair classifier. This task was a little hard to do as the data was combined from different sources. Nevertheless, the first step on this road was to choose the classifier that we wanted to work with. The protected attribute had 2 levels - *china* and *other*. Therefore, the mitigation methods had a higher chance of reducing the amount of bias. All plots were made with the use of test set (25% of original data).

3.1 Which model to choose?

We focused on 2 packages that allow in-process mitigation - fairlearn (Bird et al., 2020) and fairtorch. Python package fairtorch uses solutions from fairlearn neural network training. Fairlearn is package from Microsoft that enables to decrease fairness in *scikit-learn* models. From *scikit-learn* we chose **Histogram-based Gradient Boosting Classification Tree** and of course the other model was the neural net.

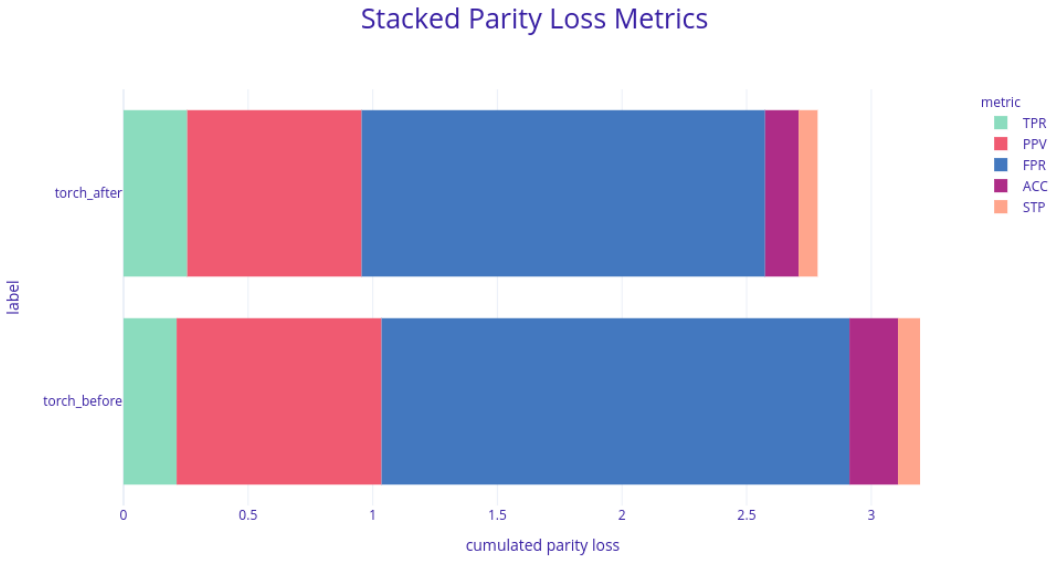
3.2 fairtorch

In this section we will show how the neural net performed in terms of fairness. The net was not so deep (2 hidden layers with 128 and 64 neurons respectively) and achieved somewhat good results in terms of performance. We made 2 models - one without any mitigation and the other with mitigation constraints.



Rysunek 3: Fairness check on 2 models.

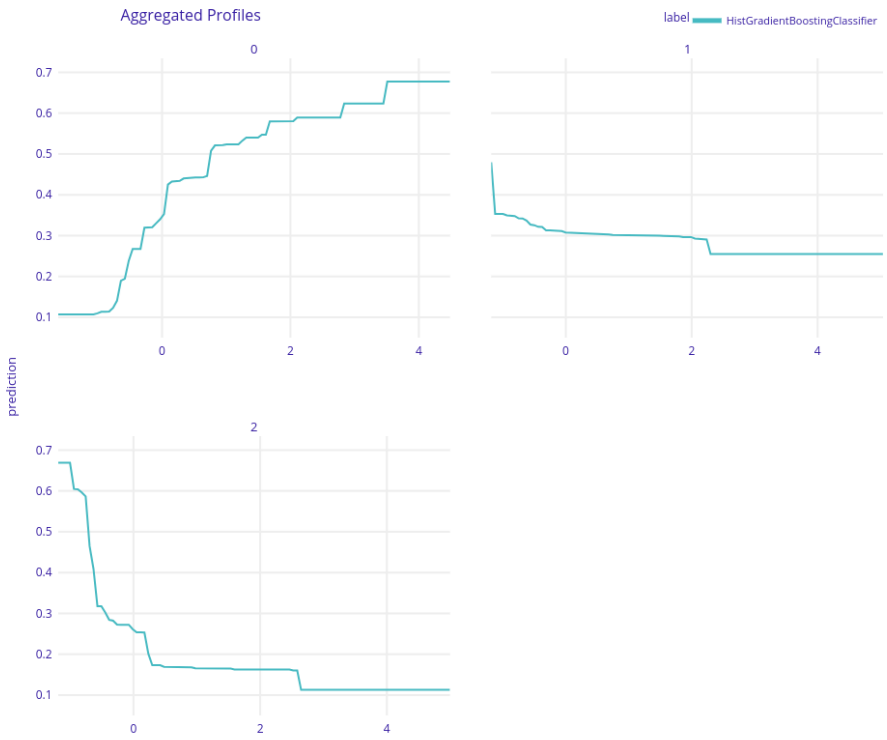
As we can see in Figure 3, the model after mitigation was not much better than the original one despite the fairness constraints. It can be also depicted using other fairness visualization tools (Figure 4).



Rysunek 4: Stacked plot on 2 models

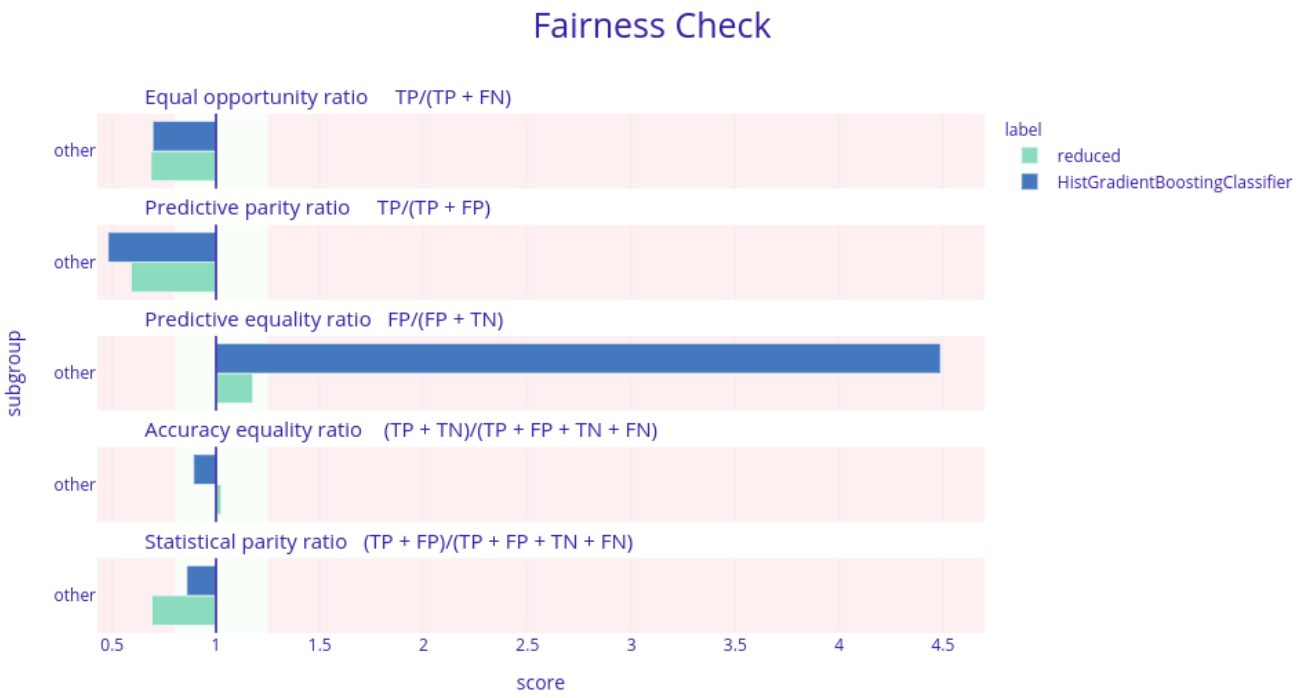
3.3 fairlearn

Next, we checked how reductions in *fairlearn* were performing. As said before, here we looked at Histogram-based Gradient Boosting Classification Tree. It was suitable for this job because of it's monotonic output - it did not over-fit to the data and gave plausible partial dependence plots as shown in Figure 5.



Rysunek 5: PDP plot of Histogram-based Gradient Boosting Classification Tree

Then we checked fairness on this unmitigated model. The next step was to compare this model and the model after mitigation (or as said in language of *fairlearn* - reduction). The results were quite surprising as the amount of bias was drastically reduced (Figure 6).



Rysunek 6: Fairness check on both classifiers

This however came with cost. The AUC and Recall were reduced by 0.1. This is not a good news because it makes the model less benefiting.

In the future, we plan to check such reductions on a wider spectrum of classifiers.

4 Summary

Our work shows a great potential for further development of models trained on medical data coming from sources from different parts of the world. So far, models we created got lower scores than a model for a single data source. We hope that in the future, more medical data will become available as it is crucial to develop a truly international model.

Literatura

Li Yan, Hai-Tao Zhang, Jorge Goncalves, Yang Xiao, Maolin Wang, Yuqi Guo, Chuan Sun, Xiuchuan Tang, Liang Jing, Mingyang Zhang, Xiang Huang, Ying Xiao, Haosen Cao, Yanyan Chen, Tongxin Ren, Fang Wang, Yaru Xiao, Sufang Huang, Xi Tan, Niannian Huang, Bo Jiao, Cheng Cheng, Yong Zhang, Ailin Luo, Laurent Mombaerts, Junyang Jin, Zhiguo Cao, Shusheng Li, Hui Xu, and Ye Yuan. An interpretable mortality prediction model for covid-19 patients. *Nature Machine Intelligence*, 2(5):283–288, May 2020. ISSN 2522-5839. doi: 10.1038/s42256-020-0180-7. URL `https://doi.org/10.1038/s42256-020-0180-7`.

Matthew Barish, Siavash Bolourani, Lawrence F. Lau, Sareen Shah, and Theodoros P. Zanos. External validation demonstrates limited clinical utility of the interpretable mortality prediction model for patients with covid-19. *Nature Machine Intelligence*, 3(1):25–27, Jan 2021. ISSN 2522-5839. doi: 10.1038/s42256-020-00254-2. URL `https://doi.org/10.1038/s42256-020-00254-2`.

Marian J. R. Quanjel, Thijs C. van Holten, Pieternel C. Gunst-van der Vliet, Jette Wielaard, Bekir Karakaya, Maaike Söhne, Hazra S. Moeniralam, and Jan C. Grutters. Replication of a mortality prediction model in dutch patients with covid-19. *Nature Machine Intelligence*, 3(1):23–24, Jan 2021. ISSN 2522-5839. doi: 10.1038/s42256-020-00253-3. URL `https://doi.org/10.1038/s42256-020-00253-3`.

Hubert Baniecki, Wojciech Kretowicz, Piotr Piatyszek, Jakub Wisniewski, and Przemyslaw Biecek. dalex: Responsible machine learning with interactive explainability and fairness in python. 2020. URL https://arxiv.org/abs/2012.14406.

Sarah Bird, Miro Dudík, Richard Edgar, Brandon Horn, Roman Lutz, Vanessa Milan, Mehrnoosh Sameki, Hanna Wallach, and Kathleen Walker. Fairlearn: A toolkit for assessing and improving fairness in AI. Technical Report MSR-TR-2020-32, Microsoft, May 2020. URL https://www.microsoft.com/en-us/research/publication/fairlearn-a-toolkit-for-assessing-and-improving-fairness-in-ai/.