MI2 SUMMER CONF

August 2, 2021
Warsaw, MiNI PW 107

Agenda

11:30 - 11:45 Welcome session
11:45 - 12:10 Elevator pitch I
12:10 - 13:00 Poster session I
13:00 - 13:30 Lunch
13:30 - 13:45 Regular talk
13:45 - 14:10 Elevator pitch II
14:10 - 15:00 Poster session II
15:00 - 15:15 Best Poster Contest
15:15 - 15:30 Close-up session

Regular talk

Monitoring of AI regulations
Giziński Stanisław

The growing number of AI applications, also for high-stake decisions, increases the interest in Explainable and Interpretable Machine Learning (XI-ML). This trend can be seen both in the increasing number of regulations and strategies for developing trustworthy AI and the growing number of scientific papers dedicated to this topic. To ensure the sustainable development of AI, it is essential to understand the dynamics of the impact of regulation on research papers as well as the impact of scientific discourse on AI-related policies.
This paper introduces a novel framework for joint analysis of AI-related policy documents and eXplainable Artificial Intelligence (XAI) research papers. The collected documents are enriched with metadata and interconnections, using various NLP methods combined with a methodology inspired by Institutional Grammar. Based on the information extracted from collected documents, we showcase a series of analyses that help understand interactions, similarities, and differences between documents at different stages of institutionalization.
To the best of our knowledge, this is the first work to use automatic language analysis tools to understand the dynamics between XI-ML methods and regulations. We believe that such a system contributes to better cooperation between XAI researchers and AI policymakers.

Elevator pitch I + Poster session I

SUCCESSful data visualization using International Business Communication Standard
Sawicki Bartosz, Ułasik Kinga

International Business Communication Standard (IBCS) contains practical proposals for the design of business communication. It complies with the rules of the seven areas that form the acronym SUCCESS (Say, Unify, Condense, Check, Express, Simplify, Structure). Until now, no open source tools implementing the IBCS in R were available. We created an R package, which facilitates creating IBCS compliant charts. Since the charts are generated using Scalable Vector Graphics (SVG), they can be easily embedded in HTML documents. Moreover, the package could be useful in reports unification.

triplot4python: the remedy for dealing with correlated features in explanations
Krzyziński Mateusz, Żółkowski Artur

Many commonly used explainable artificial intelligence (XAI) methods for estimating feature importance are problematic as they ignore dependencies between variables, often assuming their independence, which leads to unrealistic settings and misleading explanations. To fill this gap, we propose an extended Python version of the triplot R package (K. Pękala, P. Biecek, K. Woźnica). It is a collection of methods that use information about associations between features. triplot4python enables interactive analysis of triplot (i.e., a new type of explanatory visualization) and also ensures that all variables are taken into account, including categorical ones.

The medLIME algorithm
Grudzień Adrianna

The medLIME algorithm is an explanatory method based on image perturbations. Its prototype was the LIME algorithm - but not without significant drawbacks, especially in the context of medical imaging. The superiority of the medLIME algorithm is primarily the ability to freely select the areas that we want to analyze, as well as the aforementioned perturbations, thanks to which we can compare with the original image, and thus better understand which features have a significant impact on the prediction of the model.

Multitasking and transfer learning capabilities in Deep Learning
Kałuska Maria

The aim of the project is to check the impact of additional tasks and transfer learning on effectiveness of the network. Initially, the network in question performed three tasks: reconstruction, classification and segmentation; it was then extended to the task of detection and more detailed segmentation by using a modified version of Masked RCNN with Unet as backbone. The project aims to answer the question whether, thanks to multitasking and transfer learning, it is possible to train the network to detect various covid changes on a very small set of 80 photos.

Elevator pitch II + Poster session II

Guide through jungle of models!
forester: An R package to automatically select between tree-based models
Hoang Thien Ly, Szmajdziński Szymon

Designing a machine learning model for a specific task is an arduous, time-consuming process. To simplify this process, we introduce the R package forester that offers tools to automatically test various tree-based models without pre-processing the data. An extension of our package is well connected with DALEX package, which provides metrics and explanations about the best models. In robust versions of the forester package, we will add feature engineering and hyperparameter tuning functions.

FairPANs - bringing fairness to neural networks
Ruczyński Hubert

The main topic of this study is the implementation and further research in the area of obtaining fair tabular data classifiers with the use of neural networks. To obtain such results, we modify the idea of GANs (Generative Adversarial Networks) by swapping the generator with the classifier and adapting the adversarial to recognize the label of a sensitive value. This way, PAN (Predictive Adversarial Network) should bring us much more fair predictions. In result, we present a mitigation technique suitable for neural networks and explore this field even more.

metaMIMIC: an analysis of hyperparameter transfer possibilities for tabular data using MIMIC-IV database
Grzyb Mateusz, Trafas Zuzanna

Transfer learning (TL) is a machine learning (ML) research problem concerning applying knowledge gained while solving past ML tasks to new ones. TL is leveraged mostly when considering neural networks, computer vision, and natural language processing. But what about simpler ML models, the ones based on tabular data? In our work, we examine hyperparameter transfer possibilities for tree boosting models. To achieve this, we create a benchmark of similar problems from the medical domain based on the MIMIC-IV database, where the same set of features is used to predict different targets. On this dataset, we evaluate multiple XGBoost hyperparameter sets and compare the results between different tasks. Then we repeat the procedure for the results obtained on diverse problems available through the OpenML website. The performed analysis shows that TL is possible even for tabular data,

SUCCESSful data visualization using International Business Communication Standards

Bartosz Sawicki, Kinga Ułasik

Faculty of Mathematics and Information Science, Warsaw University of Technology

IBCS

The IBCS Association is an open not-for-profit organization that supports the promotion, maintenance and further development of the International Business Communication Standards (IBCS®). 1.0 version of IBCS was published in 2013 by Rolf Hichert and Jürgen Faisst. Since 2017 1.1 version is available.

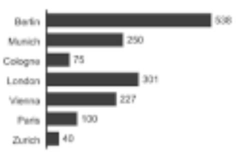
Standards contain practical proposals for the design of business communication. The main goal is to design charts in a proper conceptual, perceptual and semantic way. In order to achieve this objective, SUCCESS rules were proposed. The charts can be later applied in reports, presentations and dashboards.

- **S** say - convey a message
- **U** unify - apply semantic notation
- **C** condense - increase information density
- **C** check - ensure visual integrity
- **E** express - choose proper visualization
- **S** simplify - avoid clutter
- **S** structure - organize content

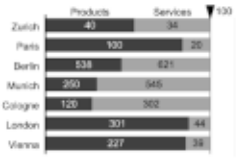
Horizontal charts

Used to show structural data of one period of time in the form of bars. Main rules applying to creating this kind of charts are avoiding truncated axes, no clipped columns and appropriate proportions.

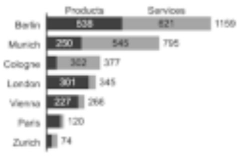
Basic



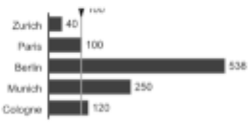
Part of the whole



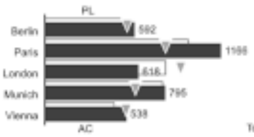
Accumulation



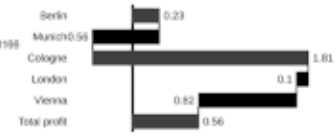
Referenced



Data series comparison



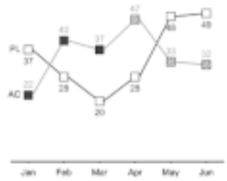
Contribution



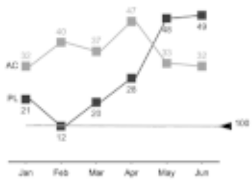
Line charts

Show time series data. Only imaginary value axis, which is starting at 0. Integrated labels to show exact values. Width of the x-axis categories depends on the unit: from the thinnest days to the widest years. Use colors to indicate scenarios.

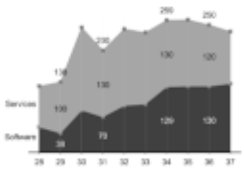
Basic



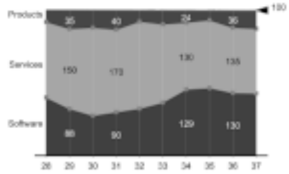
Referenced



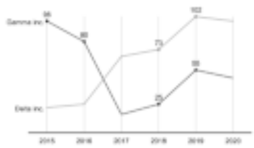
Accumulation



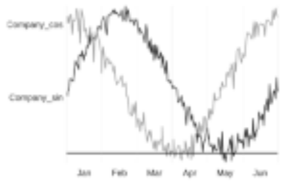
Part of the whole



Highlighted points



Many datapoints



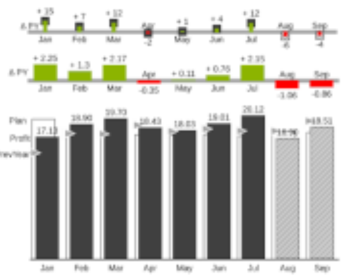
Column charts

Visualize time series data. Avoid truncated axis and clipped columns. No Y-axis but integrated labels. Use colors to indicate scenarios. Pins for relative (in %) variance and bars for absolute variance

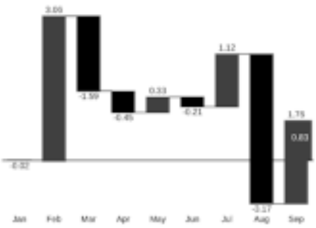
Accumulation

Basic

Grouped column chart with absolute and relative variances



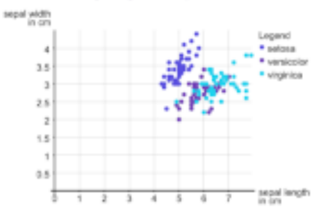
Contribution



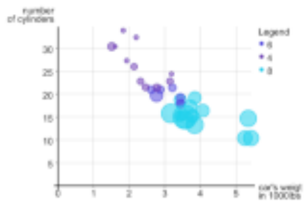
Scatter plots

Use charts with two value axes to show two dimensional positioning of elements. Add colors for different categories to create third dimension. Show even fourth numeric dimension by changing point size and creating bubble plot.

Highlighted points



Bubble plot



Contact

sawicki.bartosz@interia.pl
kinia.u@poczta.fm

GitHub repository

All visualizations were generated using an R package created during the summer instership in MI2Data Lab.

triplot4python:
the remedy for dealing with correlated features in explanations

Mateusz Krzyziński[1], Artur Żółkowski[1]

[1]Faculty of Mathematics and Information Science, Warsaw University of Technology

Introduction

Estimating feature importance is one of the key tasks in the explanatory model analysis. Many commonly used explainable artificial intelligence methods for this task are problematic as they ignore dependencies between features, often assuming their independence, which leads to unrealistic settings and misleading explanations [1].

To fill this gap, we propose an extended Python version of the **triplot** R package [2]. It is a collection of methods that use information about associations between features to create explanations. **triplot4python** enables interactive analysis of triplot (i.e., a new type of explanatory visualization) and also ensures that all features are taken into account, including categorical ones. The module is still work-in-progres and will be included in the **dalex** Python package [3].

Grouping features

Group of features (aspects) can be both defined by a user and automatically created. In auto-grouping, the hierarchical clustering is used. Distances between features are based on the association matrix – the more dependent the features are, the „closer" they are to each other.

Computing association between features by default

- two numerical features: the association is the absolute value from the Spearman's rank correlation coefficient;
- two categorical features: the association is the value of Cramér's V with bias correction (based on Pearson's chi-squared statistic);
- one numerical and one categorical feature: the association is the value of eta-squared η^2 (based on H-statistic from Kruskal-Wallis test).

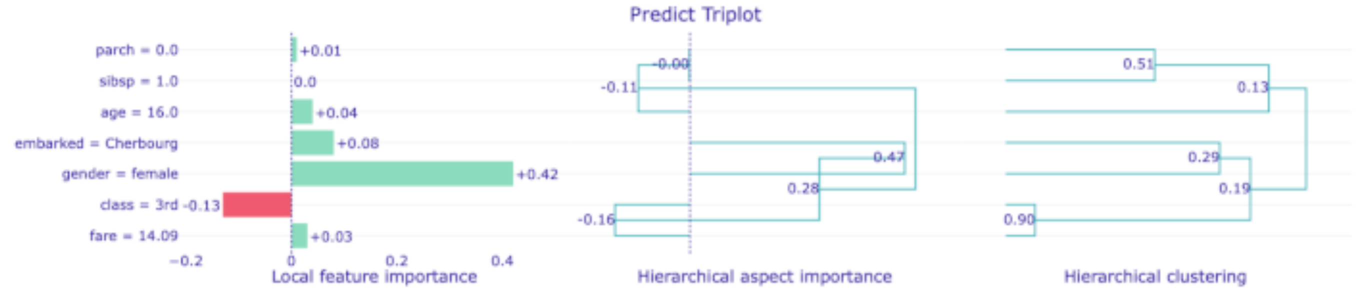


Figure 1: Predict Triplot shows the hierarchical features importance for the selected, single observation from the titanic dataset. The importance values come from the Local Aspect Importance method.

Triplot

Triplot is a tool that creates explanations based on a hierarchical feature association structure. It can be used both on the local level (**Figure 1**) and on the global level (**Figure 2**).

The triplot analysis enables a deeper understanding of the influence of dependencies between the features on the model prediction, allows to find an appropriate approach to grouping features, and also provides a background for further model exploration.

Triplot gives a **more holistic explanation** of the importance of features by combining three panels:

- **Hierarchical clustering** – global association structure between features visualized by hierarchical clustering dendrogram,
- **Hierarchical aspect importance** – the importance of groups of features determined by hierarchical clustering,
- **(Local) feature importance** – the importance of every single feature.

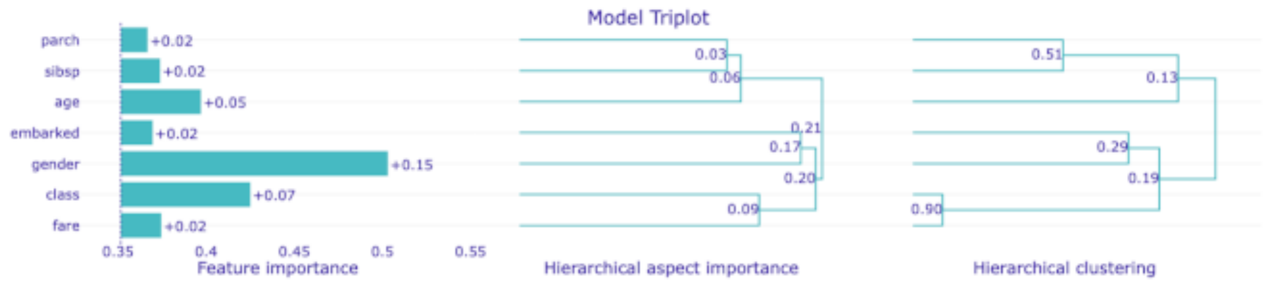


Figure 2: Model Triplot shows the hierarchical features importance for the titanic dataset. The importance values come from the Variable Importance method implemented in dalex [3].

Local Aspect Importance

Local Aspect Importance is a method that provides the calculation of aspects importance for a selected observation. It allows the use of knowledge about the dependencies between features. Moreover, in the case of many features, this increases the clarity of the explanation.

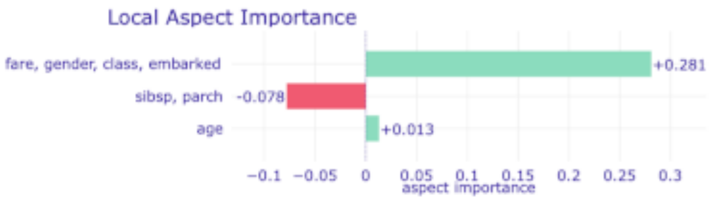


Figure 3: Local Aspect Importance plot shows the group of features importance for selected, single observation from the titanic dataset. Groups are created based on the chosen correlation cutoff level h (here h=0.14).

References

[1] P. Biecek and T. Burzykowski. Explanatory Model Analysis. Chapman and Hall/CRC, New York, 2021.

[2] K. Pekala, K. Woznica, and P. Biecek. Triplot: model agnostic measures and visualisations for variable importance in predictive models that take into account the hierarchical correlation structure. arXiv:2104.03403, 2021.

[3] H. Baniecki, W. Kretowicz, P. Piatyszek, J. Wisniewski, and P. Biecek. dalex: Responsible Machine Learning with Interactive Explainability and Fairness in Python. arXiv:2012.14406, 2020.

Acknowledgements

We are greatful to Hubert Baniecki for substantive support. We also thank Katarzyna Pękala, Katarzyna Woźnica and Przemysław Biecek for discussions about the design and development of the presented solution.

- ✉ mateusz.krzyzinski.stud@pw.edu.pl
- ✉ artur.zolkowski.stud@pw.edu.pl

Guide through jungle of models!
forester: An R package to automatically select between tree-based models

Szymon Szmajdziński, Hoang Thien Ly, Anna Kozak

Faculty of Mathematics and Information Science, Warsaw University of Technology

Introduction

Designing a machine learning model for a specific task is an arduous, time-consuming process. To simplify this process, we introduce the R package **forester** that offers tools to automatically test various tree-based models without pre-processing the data. An extension of our package is well connected with **DALEX** package, which provides metrics and explanations about the best models. In robust versions of the **forester** package, we will add feature engineering and hyperparameter tuning functions.

Benefits of forester package

1. **No requirements for data** - There is no need to create particular object for each model. The package deals with common data structures, such as: data frames, matrices, data tables. It partly performs feature engineering so the users do not have to.
2. **Simple user interface** - One function with three parameters, that is all it takes to create the model.
3. **Automatic hyperparameter optimization** - Besides having the trained model, tuple of hyperparameters in future will be automatically optimized and selected.
4. **Comparing and selecting best model** - **Forester** package is able to make comparisons between metrics of built models and choose the best one.
5. **Providing explanations** - Explanation plays a crucial role in eliminating reluctance and increasing trust for decision makers while using model's results. With the assistance of **DALEX** package, **forester** enables users to create explanations in both local and global levels.

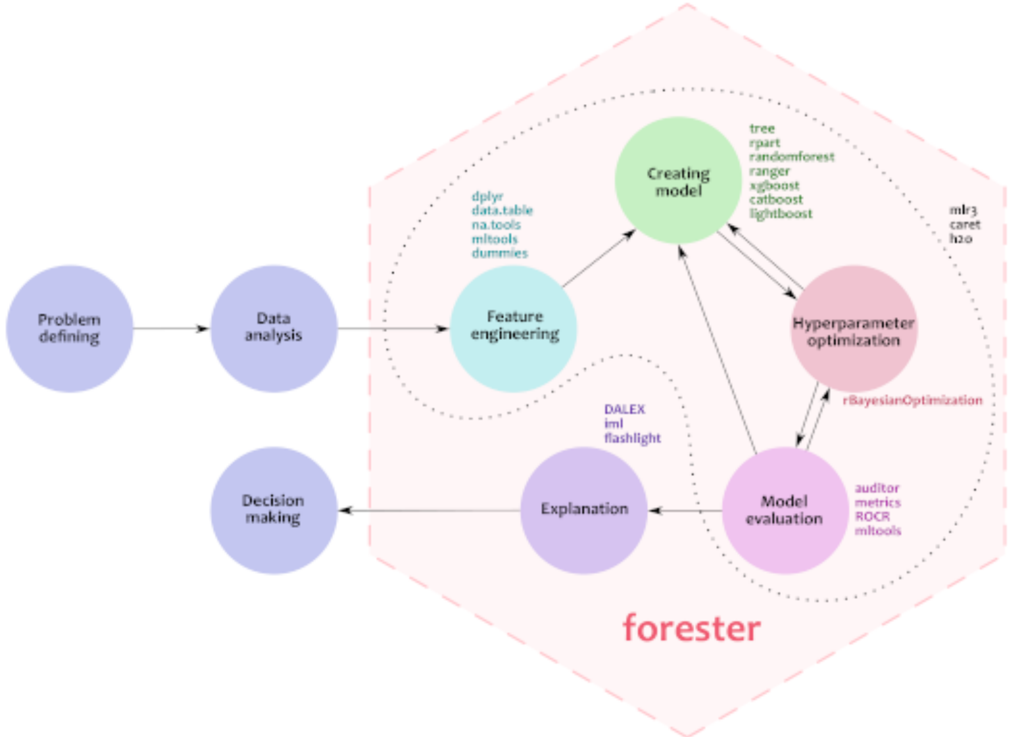


Figure 1: Role of forester package in general Machine Learning pipeline.

Package Structure

With functions in **forester** package, users can create tree-based model in an unified, simple formula. With only three parameters, user can create various models. First parameter is data with no requirements of pre-processing. After that, user has to specify the name of target column and type of the task, classification or regression. **Forester** automatically does the rest. In upcoming versions of package, we will add features for feature engineering and hyperparameter tuning.

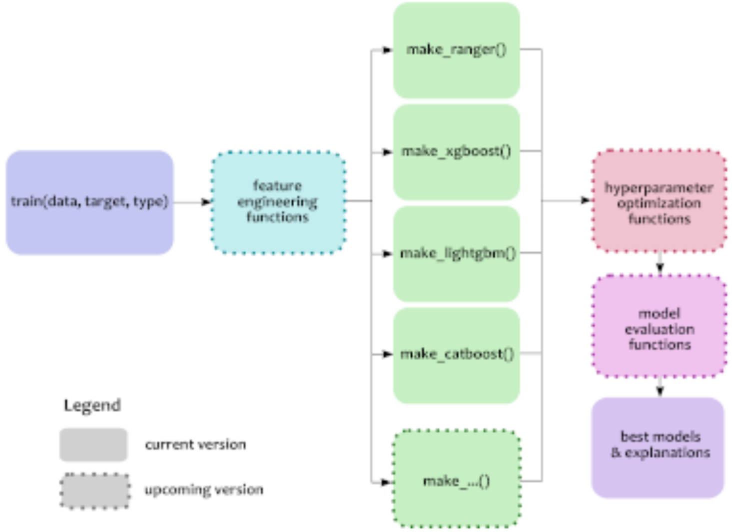


Figure 2: Structure of forester package.

Example

```
library(forester)

# simple use of forester package
best_model <- train(data = mtcars, target = "mpg", type = "regression")

# use with DALEX
mp <- DALEX::model_parts(best_model)
plot(mp)
```

Contact info

- szymon.szmajdzinski.stud@pw.edu.pl
- hoang_thien.ly.stud@pw.edu.pl
- https://github.com/ModelOriented/forester

FairPANs - bringing fairness to neural networks

AUTHORS

Hubert Ruczyński

Jakub Wiśniewski

ABSTRACT

The main topic of this study is the implementation and further research in the area of obtaining fair tabular data classifiers with the use of neural networks. To achieve such results, we modify the idea of GANs (Generative Adversarial Networks) by swapping the generator with the classifier and adapting the adversarial to recognize the label of a sensitive value. This way, PAN (Predictive Adversarial Network) should bring us much fairer predictions. As a result, we present a mitigation technique suitable for neural networks and explore this field even more.

INTRODUCTION TO FAIRNESS

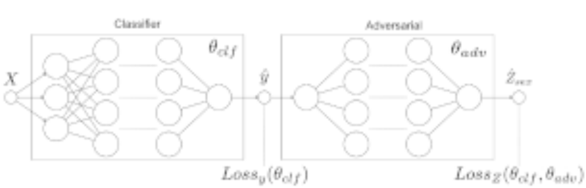
Consider the idea of the algorithm that has to predict whether giving credit to a person is risky or not. It is learning on real data of giving credits which were biased against females (historical fact). In that case, the model learns this bias, which is not only included in the simple sex variable but also is hidden inside other variables. Fairness enables us to detect such bias and handles a few methods to fight it. To learn more, I recommend the article 'Fairmodels: A Flexible Tool For Bias Detection, Visualization, And Mitigation' by Jakub Wisniewski and Przemysław Biecek.

INTRODUCTION TO GANS

Generative Adversarial Networks are two neural networks that learn together. The Generator has to generate new samples that are indistinguishable from original data and the adversarial has to distinguish if the observation is original or generated. The generator is punished whenever the adversarial makes the correct prediction. After such process generator eventually learns how to make indistinguishable predictions and adversaries' accuracy drops up to 50% when a model cannot distinguish the two classes.

MEET FAIRPANS

FairPANs are the solution to bring fairness into neural networks. We mimic the GANs by subsetting generator with classifier (predictor) and adversarial has to predict the sensitive value (such as sex, race, etc) from the output of the predictor. This process eventually leads the classifier to make predictions with indistinguishable sensitive values. The idea comes from blogs: Towards fairness in ML with adversarial networks (Stijn Tonk) and Fairness in Machine Learning with PyTorch (Henk Griffoen) however, our implementation in R offers slightly different solutions.



CUSTOM LOSS FUNCTION

The crucial part of this model is the metric we use to engage the two models into a zero-sum game. This is captured by the following objective function:

$$min_{\theta_{clf}}[Loss_y(\theta_{clf}) - \lambda Loss_Z(\theta_{clf}, \theta_{adv})]$$

So, it learns to minimize its prediction losses while maximizing that of the adversarial (due to lambda being positive and minimizing a negated loss is the same as maximizing it). The objective during the game is simpler for the adversarial: predict sex based on the income level predictions of the classifier. This is captured in the following objective function:

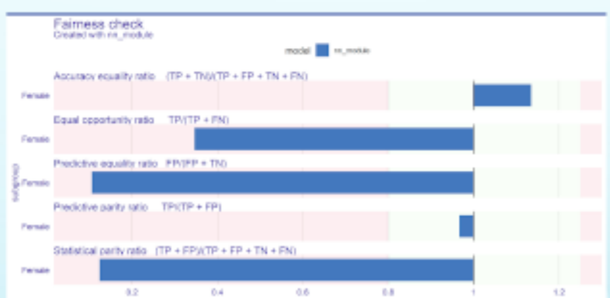
$$min_{\theta_{clf}}[Loss_Z(\theta_{clf}, \theta_{adv})]$$

The adversarial does not care about the prediction accuracy of the classifier. It is only concerned with minimizing its prediction losses.

Firstly we pretrain classifier and adversarial. Later we begin the proper PAN training with both networks: we train the adversarial, provide its loss to the classifier, and after that, we train the classifier. This method shall lead us to fair predictions of the FairPAN model.

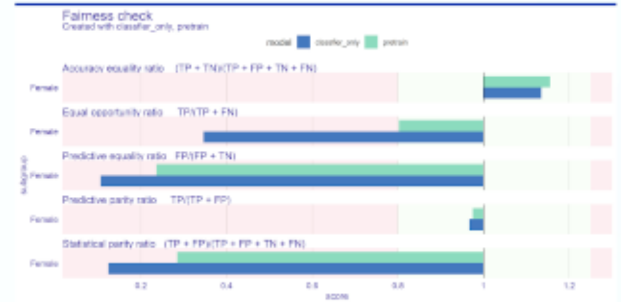
RESULTS

To show, that this method is reasonable and works properly, let us introduce you to a simple example of how the FairPAN works. The graph below represents the 5 most important fairness metrics for the adult data set, just after pretrain. For each of the metrics, which are explained in the plot, we calculate a ratio between the two classes. We assume that the model is fair according to the metric when it satisfies the 4/5 rule (the bar isn't on the red background). The 4/5 rule is satisfied when both labels' in a sensitive class get similar results (ratio between 4/5 and 5/4). Let's note that the most important for us to minimize it is the STP ratio (Statistical Parity Ratio).

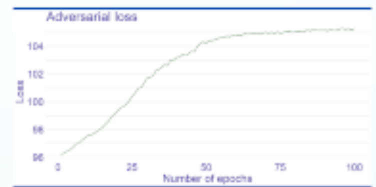


As we can see, the STP ratio after the pretrain is really bad, which means that there is a lot to improve in this case.

To accurately show the progress made by FairPAN, the next plot represents the metrics for the classifier model after 50 epochs of training. As we can see, during the training it worsens its fairnessmetrics in order to optimize its acc

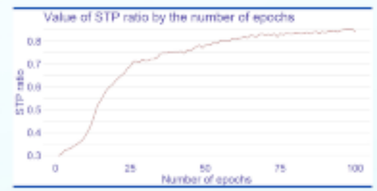


To check if our model works properly, we've decided to track the three most important metrics which are: loss and accuracy of the adversarial model and of course STPR itself. These statistics indicate if the model is learning well and makes desirable progress. [1][2] Loss function should grow and accuracy should diminish because it means that adversarial is worse with every iteration. It means that the classifier makes more fair predictions (the adversary can't say which label is which). [3] STPR in this case should grow towards the 0.8-1.25 range which means that predictions are fairer. Moreover, we decided to monitor the accuracy of the classifier [4].



[1] Loss of the adversarial model

[2] Accuracy of the adversarial model



[3] STP ratio for PAN model

[4] Accuracy of PAN model

As we can see from the last plot, FairPAN training led to significant STP ratio improvement, so that the model became fair according to this metric. It isn't absolutely costless, because other fairness metrics worsened however, it is impossible to improve one metric without worsening the others. As you can see from the table below, performance metrics also diminished, but the loss is pretty low in comparison to ordinary mitigation techniques and we lose only 1.5% of the accuracy to make our model fair.

	Accuracy	AUC	F1	Precision	Recall
Classifier	0.850	0.871	0.541	0.773	0.417
PAN	0.837	0.857	0.536	0.680	0.443

metaMIMIC: an analysis of hyperparameter transfer possibilities for tabular data using MIMIC-IV database

Mateusz Grzyb [1], Zuzanna Trafas [2], Katarzyna Woźnica [1]
[1]Faculty of Mathematics and Information Science, Warsaw University of Technology
[2]Faculty of Computing and Telecommunications, Poznan University of Technology

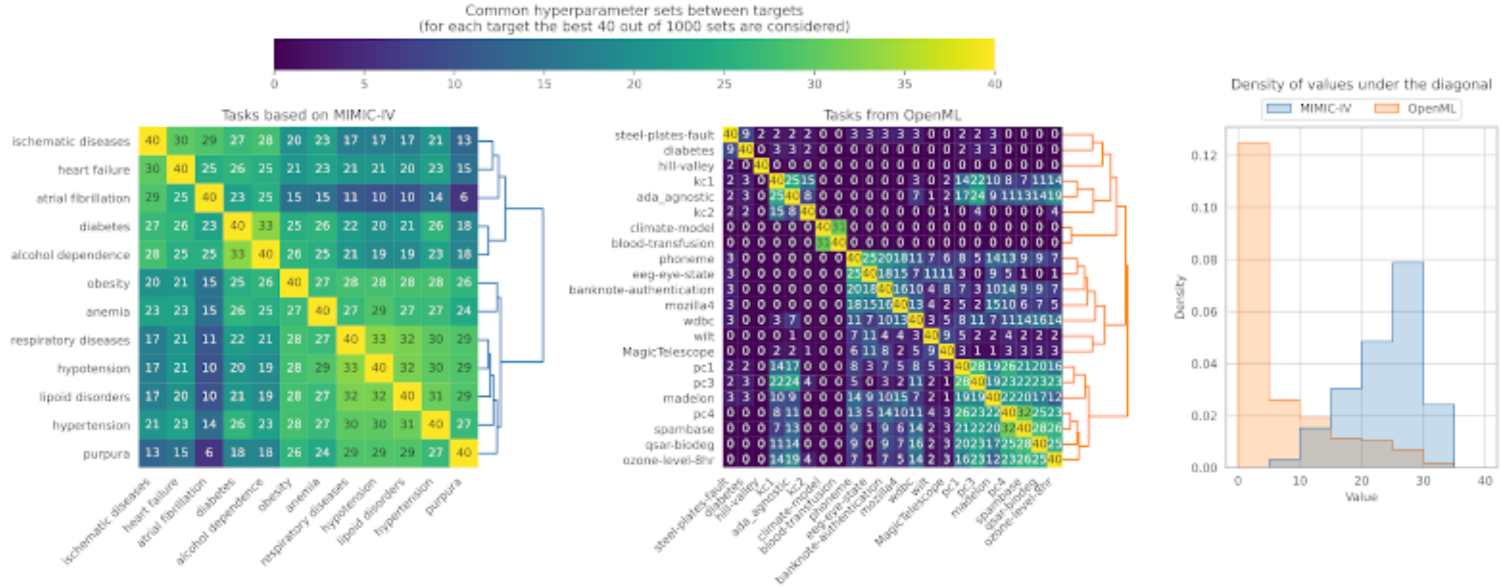


Figure 1: Similarity of the best (in regard to the 4-CV ROC AUC measure) hyperparameter sets for MIMIC-IV and OpenML tasks. Dendrograms are based on a distance defined as 40 − *value* and the Ward's hierarchical clustering method.

Motivation

- Transfer learning enables us to choose better starting points for training of neural networks based on previously solved problems. However, it has not yet been successful for simpler models trained on tabular data.
- Tuning hyperparameters improves the performance of boosting models yet can be time-consuming - it is hard to propose good hyperparameter settings a priori.
- Lack of good benchmarks for transfer learning on tabular data.

We present the results of a hyperparameter transfer between different XGBoost models built on MIMIC-IV database based tasks, which can significantly reduce tuning times.

The MIMIC-IV benchmark

MIMIC (Medical Information Mart for Intensive Care) is a freely available database comprising de-identified health-related data of over 60,000 patients. [1]

This data enabled us to create a benchmark of similar medical domain problems - using the same set of features we intend to predict 12 different patient conditions (e.g. diabetes or anemia).

Experiment setup

Using the proposed benchmark, we evaluated and ranked (based on the 4-CV ROC AUC measure) 1000 different XGBoost hyperparameter settings for the 12 tasks each. Then, we compared these rankings and repeated the procedure for the ones from MementoML [2], where the same hyperparameter grid was evaluated using 22 models based on selected classification tasks from the OpenML repository (Figure 1).

Furthermore, using the same rankings we simulated different hyperparameter search methods and contrasted them with random search expected value trajectories (Figure 2).

Results

The results presented in Figure 1 show how many hyperparameter sets are shared among the best 40 for each task from the respective source. Both the colors in the matrices and the shapes of the density histograms prove that the tasks based on the MIMIC-IV database have more common sets than the tasks from the OpenML repository.

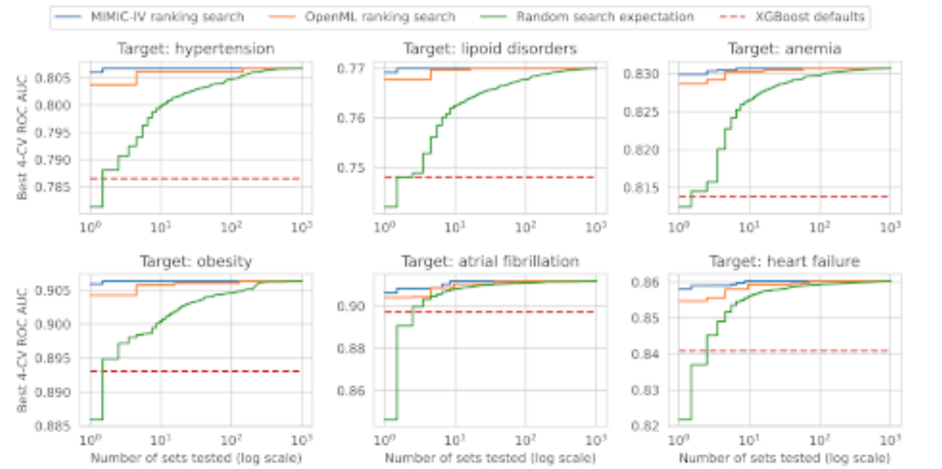


Figure 2: Velocity of convergence to the optimal hyperparameter set for multiple benchmark tasks.

Figure 2 shows how fast the optimal hyperparameter set can be obtained using different search methods for 6 tasks based on the MIMIC-IV database. MIMIC-IV and OpenML based search orders were created by normalising 4-CV ROC AUC values for each task, summing new values for each hyperparameter set and sorting in a descending fashion. Expectation of random search performance was determined by using Beta distribution properties and inverse empirical quantile function of measure values. It is visible that MIMIC-IV based search order performs the best.

Conclusions

Presented experiment results prove that hyperparameter transfer learning for tabular data is possible and can be beneficial for model tuning times. Hyperparameter sets transfer better when the considered tasks are related. The exact effect of the tasks relation on the transfer learning strength is going to be a matter of our further research.

References

[1] A Johnson, L Bulgarelli, T Pollard, S Horng, LA Celi, and R Mark. Mimic-iv (version 1.0), 2020.

[2] Wojciech Kretowicz and Przemysław Biecek. Mementoml: Performance of selected machine learning algorithm configurations on openml100 datasets, 2020.