ETH
Swiss Federal Institute of Technology Zurich

Seminar for Statistics

Department of Mathematics

Master Thesis Spring 2024

Luca Brilhaus

External Validity of Predictors in ICU Data

Submission Date: April 4th, 2024

Co-Advisor: Malte Londschien
Advisor: Prof. Dr. Peter Bühlmann

To my mother, Monika, and my uncle, Horst.

Preface

Supplementary Material

Instructions and the relevant code needed to reproduce this thesis can be found in the GitHub repository (https://github.com/lbrilh/masterthesis). We recommend installing the Python enviroment and following the manual provided in the GitHub repository before running the code.

Acknowledgements

First, I wish to express my sincere gratitude to my supervisor Prof. Dr. Peter Bühlmann for taking the responsibility for my work and for investing time to discuss conceptual and guiding questions.

This endeavor would not have been possible without Malte Londschien. His high personal commitment, reliability as well as the weekly instructive supervision meetings were essential for this work. Thank you very much Malte.

Last but not least, I would like to express my gratitude to the *Seminar for Statistics*, which created the framework conditions for this work and did everything to help me with conceptional and administrative questions. I should also mention the computing resources provided by them, without which my computations would not have been feasible.

Abstract

The increasing availability of electronic health records holds the potential to significantly enhance patient outcomes using predictive models. This thesis examines the external validity of predictors in Intensive Care Unit (ICU) data. Leveraging harmonized data from four major ICU databases in the United States and Switzerland, we explored the distributional robustness of predictors and models developed in one hospital or dataset and applied to another. In a first step, we investigated a fine-tuning approach using a small dataset from the target to adapt models trained on one dataset to the target. This approach was then applied to both traditional methods and robust techniques like Anchor Regression. A boosted version of Anchor Regression and a refitted Random Forest demonstrated increased robustness. We then investigated the robustness of Magging and Data Shared Lasso, both designed for handling multiple datasets. While Magging indicated robustness in a linear setting, Data Shared Lasso did not. Furthermore, we analyzed Data Shared Lasso's ability to identify similarities and differences among datasets, yielding promising results. We concluded by evaluating the robustness of stepwise regression applied to Data Shared Lasso and observed that strong regularization lead to robust stepwise regression.

Contents

List of Figures

List of Tables

Notation

Since this thesis, despite its applied nature, is located at the Mathematics Department of ETH Zürich, we adhere to the convention of speaking in the first-person plural 'we'. Furthermore, only equations that are referenced elsewhere are equipped with a number.

Variables

- $\emptyset$: Empty set.
- $\lambda \in \mathbb{R}, \gamma > 0$: Scalars.
- $G \in \mathbb{N}$: Number of groups.
- $g \in \{1, \dots, G\}$: Group indice.
- $n \in \mathbb{N}$: Sample size.
- $n_{tune} \in \mathbb{N}$: Fine-tuning sample size.
- $n_g \in \mathbb{N}$: Sample size of group $g \in \{1, \dots, G\}$; $n_1 + \ldots + n_G = n$.
- i, j: Indices in $\{1, \dots, n\}$.
- $B \subset \{1, \dots, n\}$: Subset of indices.
- $y = (y_1, \dots, y_n) \in \mathbb{R}^n$: Response in n-dim interpolation setting.
- $\hat{y} \in \mathbb{R}^n$: Estimate of y.
- $Y^{tune} = (Y_1^{tune}, \dots, Y_n^{tune}) \in \mathbb{R}^{n_{tune}}$: Response in the fine-tuning dataset.
- $Y(g)$ or $y_g \in \mathbb{R}^{n_g}$: Response in group g.
- $X \in \mathbb{R}^{n \times p}$: Design matrix. Each row corresponds to one observation and each column to one predictor.
- $x_i \in \mathbb{R}^p$: Observation.
- $X_i^{tune} \in \mathbb{R}$: 1-dim predictor in the fine-tuning dataset.
- $X(g)$ or $X_g \in \mathbb{R}^{n_g \times p}$: Design matrix of group g. Each row corresponds to one observation in group g and each column to one covariate.
- $Z \in \mathbb{R}^{n \times p(G+1)}$: Augmented design matrix.
- $g_l \in \{1, \dots, G\}, l \in \{1, \dots, n\}$: group indicator for each observation.
- $\tilde{g} = (g_1, \dots, g_n) \in \mathbb{R}^n$: Vector indicating group for each observation.

- $U = (U_1, \ldots, U_n) \in \mathbb{R}^n$: Residuals given by $y - \hat{y}$.
- $\Theta^* \in \mathbb{R}^p$: True underlying regression coefficient vector.
- $\widehat{\Theta} \in \mathbb{R}^p$: Estimator of Θ^*.
- $\widehat{\Theta}_{(-B)} \in \mathbb{R}^p$: Estimator of Θ^* without using data with indices in B.
- $\widehat{\Theta}_g$ or $\widehat{\Theta}(g) \in \mathbb{R}^p$: Regression coefficient estimate in group g.
- $\widehat{\Theta}_{agg}$: Magging regression coefficient estimate.
- $\tilde{\Theta} \in \mathbb{R}^{p(G+1)}$: Augmented coefficient vector.
- $m(\cdot)$: True underlying relationship with y, i.e. $y = m(X)$.
- $m_g(\cdot)$: True underlying relationship in group g.
- $\widehat{m}(\cdot)$: Estimate of $m(\cdot)$.
- $\widehat{m}_g(\cdot)$: Estimate of $m_g(\cdot)$.
- $\widehat{m}_{-B}(\cdot)$: Estimator of $m(\cdot)$ without using data with indices in B for estimation.
- $\rho(\cdot, \cdot)$: Loss function.
- $A \in \mathbb{R}^q$: Exogenous variables used as Anchors.
- P_A: L_2-projection on the span of A.
- Π: Estimated projection matrix on A's column space.
- Id: Identity matrix.
- $w_g \geq 0$: Magging weight of group g.
- $w = (w_1, \ldots, w_G) \in \mathbb{R}^G$: Magging weight vector.
- C_G: Space of possible Magging weight vector.
- $\Sigma \in \mathbb{R}^{p \times p}$: Covariance matrix of X.
- $\varepsilon \in \mathbb{R}$: Independent error.
- $r_g \in \mathbb{R}$: Degree of sharing.

Abbreviations and Objects

- ERM: Empirical Risk Minimization.
- MSE: Mean Squared Error (see Section 3.2).
- CV: Cross Validation (see Section 3.2).
- k-fold CV: k-fold Cross Validation (see Section 3.2).
- i.i.d.: Independent and identical distributed.
- Bagging: Bootstrap aggregating (see Section 3.3.3).
- ICU: Intensive Care Unit.
- train: Train dataset.

- eicu: eICU Dataset (see Section 2.1).
- hirid: HiRID Dataset (see Section 2.1).
- mimic: MIMIC-III Dataset (see Section 2.1).
- miiv: MIMIC-IV Dataset (see Section 2.1).

Statistical Models

- OLS: Ordinary Least Squares (see Appendix B.2.1).
- LASSO: Least Absolute Shrinkage and Selection Operator (see Section 3.3.2).
- RIDGE: Ridge Regression (see Section 3.3.1).
- RF: Random Forest (see Section 3.3.3).
- DSL: Data Shared Lasso (see Section 5.2).
- GBDT: Gradient Boosted Decision Tree (see Section 3.3.4).
- ANCHOR: Anchor Regression (see Section 3.3.5).
- REFITLGBM: Reffited Gradient Boosted Decision Tree (see Section 3.3.7).
- ANCHOR BOOST: Boosted Version of Anchor Regression (see Section 3.3.6).
- MAGGING: Maximin Aggregating (see Section 4.2).

Chapter 1

Introduction

The increasing availability of electronic health records offers a chance to implement statistical support systems in intensive care units. Ensuring robust performance of these prediction models is crucial, especially when we aim to apply them across various healthcare institutions. Distributional robustness is a framework for evaluating the performance of prediction models under distribution shifts[1]. In this context, we consider a prediction model to be robust when its performance remains relatively stable across datasets despite distributional shifts. However, ICU prediction models are commonly trained using a variation of empirical risk minimization (ERM), like minimizing the mean squared error (MSE) on the train data (Rockenschaub, Hilbert, Kossen, von Dincklage, Madai, and Frey (2023)). This approach assumes similarity between patient populations in both training and testing phases to perform effectively (Gulrajani and Lopez-Paz (2020)). However, hospitals vary in the types of patients they treat and in their clinical pathways for diagnostics and treatment, leading to inherent heterogeneity (Sauer et al. (2022)). Ignoring these structural differences can result in poor predictive performance and potentially misleading interpretations of the model (Reyna et al. (2020)). Training on data from multiple hospitals may help address this issue. Shared associations across multiple hospitals may be more likely to be applicable to new hospitals. Unfortunately, the real-world effectiveness of this strategy is not well-examined (Wynants, Kent, Timmerman, Lundquist, and Van Calster (2019)).

In this work, we use harmonised data from four ICU data sources from the US and Switzerland to investigate how reliable predictors and models can be transferred from one hospital/dataset to another. After presenting the available data, and introducing the clinical concepts used in this work (Chapter 2), we proceed with the main parts of this thesis consisting of Chapters 3 - 6. In Chapter 3, we investigate the benefits of using a small fine-tuning dataset from the test population to improve or "tune" the efficacy of different statistical models. For this, we introduce Anchor Regression (Rothenhäusler, Meinshausen, Bühlmann, and Peters (2020)), a method designed to improve distributional robustness, extend it via boosting and compare their performance with more traditional methods like Random Forest. In Chapter 4 we introduce Magging (Bühlmann and Meinshausen (2016)), an estimator designed to capture effects which are common to all data, generalize it using the plug-in principle to accommodate for the use of non-linear methods and apply it to our data. We present another method aiming to capture shared effects

[1] Sometimes also referred to as distributional differences.

across datasets, Data Shared Lasso (DSL), in Chapter 5 (Gross and Tibshirani (2016)). After introducing Data Shared Lasso, we apply DSL in the same setting as Magging and compare their performance. Afterwards, we compare DSL and Magging in a more controlled setting on artificial data. We finish Chapter 5 by assessing DSL's capability to enhance our understanding of dataset differences, potentially improving model distinguishability. We contrast this using a Random Forest, a slightly modified version of pooled Lasso, and running separate Lasso's on each dataset. We conclude the main part by evaluating the robustness of stepwise regression applied on DSL and attempt to control it (Chapter 6). Our conclusion, recommendations, as well as an outlook on future work is given in Chapter 7.

As a final note, a few words on the conventions used in this thesis: We use "predictor," "feature," and "covariate" interchangeably throughout this work. Similarly, we use "target variable" and "response variable" interchangeably. When employing linear methods, we always include the intercept, although not explicitly mentioned. When referencing to model performance, we do so in terms of mean squared error (MSE) on unseen datasets. In Chapter 6, we define robustness slightly differently, as a monotone decrease in MSE on datasets exhibiting distributional shifts. Unless explicitly stated, we use "hospital" and "dataset" synonymously.

Chapter 2

Available Data

This section describes the available data and the challenges associated with it. We outline our study population and explain the harmonization process implemented to ensure interoperability among the datasets.

2.1 Data sources

We used retrospective open-access ICU data comprising of anonymised patient information from four sources spanning three countries: eICU (electronic ICU) version 2.0 (Pollard, Johnson, Raffa, Celi, Badawi, and Mark (2019)) from the US, MIMIC-III (Medical Information Mart for Intensive Care III) version 1.4 (Johnson, Pollard, and Mark (2016)) from the US, MIMIC-IV (Medical Information Mart for Intensive Care IV) version 2.2 (Johnson, Bulgarelli, Pollard, Horng, Celi, and Mark (2023)) from the US and HiRID (High Time Resolution ICU Dataset) version 1.1.1 (Faltys, Zimmermann, Lyu, Hüser, Hyland, Rätsch, and Merz (2021)) from Switzerland. eICU, HiRID, MIMIC-III and MIMIC-IV were accessed through PhysioNet (Goldberger, Amaral, Glass, Hausdorff, Ivanov, Mark, Mietus, Moody, Peng, and Stanley (2000)). All data used in this study were collected as part of standard care procedures. All datasets were recorded at a single hospital with the exception of eICU, which is a multi-center dataset of 208 hospitals. Given that the datasets were independently created, they lack uniform data structures or identifiers. To enable interoperability and allow their utilization within a unified prediction model, we employed preprocessing using the ricu R package version 0.5.5 (Bennett, Plečko, Ukor, Meinshausen, and Bühlmann (2023)). The ricu package pre-defines numerous clinical concepts and their loading process from a specified data source, offering a standardized interface to access the data.

2.2 Study population

Each ICU stay was discretized into hourly bins. To ensure adequate data quality and sufficient information for prediction, we excluded stays with:

(1) Missing or invalid admission and discharge times

(2) A length of less than six hours in the ICU

(3) Less than four hours of observations

(4) More than 12 consecutive hours in the ICU without a single clinical measurement

The unit of observation were single, continuous ICU stays. As some datasets did not allow to identify which ICU stays belonged to which patient, all available ICU stays were included in the analysis. Consequently, patients could contribute more than one ICU stay. For example, if a patient was discharged to a general ward and subsequently readmitted to the ICU, that patient contributed two ICU stays.

2.3 Features and preprocessing

For each ICU stay, we extracted 46 features: four static features and 42 time-varying features. Dynamic features include seven vital signs, like heart rate, respiratory rate or blood pressure, 33 laboratory results, e.g. blood gases, electrolytes, and two more variables measuring input (fraction of inspired oxygen) and and output (urine) of a patient. The dynamic variables were measured in 1h intervals. Static features include age, weight, sex, and height. The static variables were measured once at entry to the ICU. No information on comorbidities, medication, or procedures was used in the models. Tables 2.1 and 2.2 provide a comprehensive list of features, including their ricu concept name and corresponding units of measurement. Feature selection was based on availability across the datasets.

Biologically implausible feature values were automatically set to missing by the ricu package. The remaining raw feature values were screened for systematic differences between data sources. Due to the routine nature of the data, measurements could be missing, e.g. infrequent measurement of expensive laboratory tests.

We applied a log transformation to the following predictors:

alp, alt, ast, bili, bnd, bun, cai, ck, ckmb, crea, crp, fgn, fio2, glu, hgb, k, lact, lymph, methb, mg, o2sat, pco2, ph, phos, plt, po2, ptt, tnt, urine, wbc

The outcome of this transformation is depicted in Figure 2.1. Additionally, histograms of the features before the log transformation can be found in Appendix B.1.

Feature	**ricu**	**Unit**
Static		
Age at hospital admission	age	Years
Sex	sex	Male or Female
Patient height	height	cm
Patient weight	weight	kg

Table 2.1: Static concepts used as input to the prediction models.

Feature	ricu	Unit
Time-varying		
Blood pressure (systolic)	sbp	mmHg
Blood pressure (diastolic)	dbp	mmHg
Heart rate	hr	beats/minute
Mean arterial pressure	map	mmHg
Oxygen saturation	o2sat	%
Respiratory rate	resp	breaths/minute
Temperature	temp	°C
Albumin	alb	g/dL
Alkaline phosphatase	alp	IU/L
Alanine aminotransferase	alt	IU/L
Aspartate aminotransferase	ast	IU/L
Base excess	be	mmol/L
Bicarbonate	bicarb	mmol/L
Bilirubin (total)	bili	mg/dL
Bilirubin (direct)	bili_dir	mg/dL
Blood urea nitrogen	bun	mg/dL
Calcium	ca	mg/dL
Creatinine	crea	mg/dL
Creatinine kinase MB	ckmb	IU/L
Chloride	cl	mmol/L
CO2 partial pressure	pco2	mmHg
C-reactive protein	crp	mg/L
Glucose	glu	mg/dL
Hemoglobin	hgb	g/dL
Lactate	lac	mmol/L
Lymphocytes	lymph	%
Mean corpuscular hemoglobin	mch	pg
Mean corpuscular hemoglobin concentration	mchc	g/dL
Mean corpuscular volume	mcv	fL
Methemoglobin	methb	mg/dL
Neutrophils	neut	%
O2 partial pressure	po2	mmHg
Partial thromboplastin time	ptt	sec
pH of blood	ph	
Phosphate	phos	mg/dL
Platelets	plt	1,000 / µL
Potassium	k	mmol/L
Sodium	na	mmol/L
Troponin T	tnt	ng/L
White blood cells	wbc	1,000 / µL
Fraction of inspired oxygen	fio2	%
Urine output	urine	mL

Table 2.2: Dynamic concepts used as input to the prediction models.

In addition to the features described in Table 2.1 and Table 2.2, we extracted the following features:

- Ethnicity: White, Black, Asian, Other
- Region: Midwest, Northeast, South, West
- Teachingstatus: True or False
- Numbedscategory, indicating the number of beds in the hospital: <100, 100-249, 250-499, >=500
- Age group: Child (0-19), Young adult (20-39), Middle-age (40-65), Senior (65+)

Patients' ethnicity was exclusively available in US datasets, while Region, Teaching status, and Number of beds category were specific to the eICU dataset. The age group "Child" did not exist in HiRID. The features, ethnicity, region, teachingstatus, numbedscategory and age group were only used for predictions in Appendix B.3. We included all static and dynamic features to train models in Chapters 3 - 6.

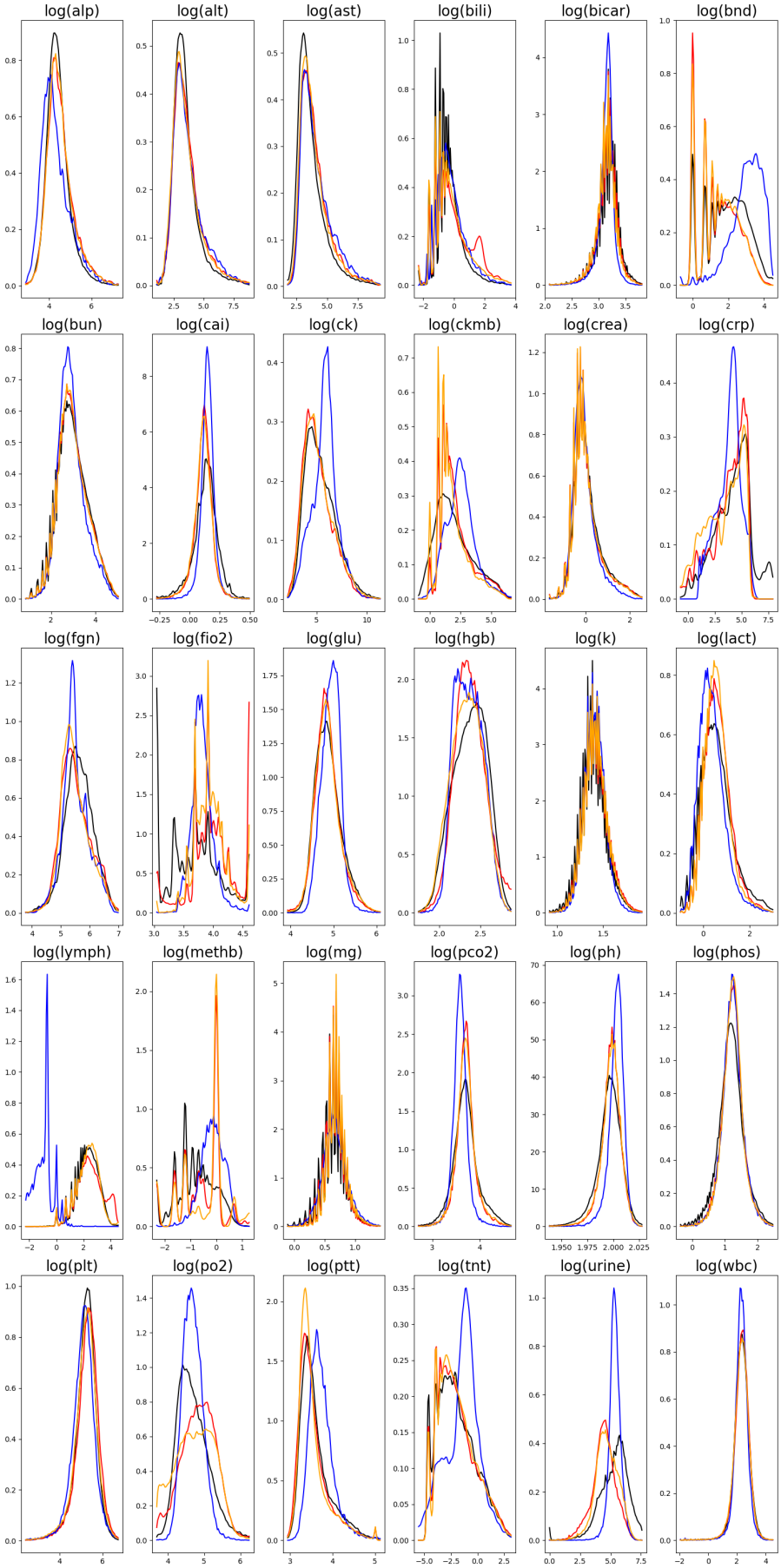


Figure 2.1: Log transformed features. eICU: Black, HiRID: Blue, MIMIC-III: Red, MIMIC-IV: Orange.

Chapter 3

Fine-tuning approach

In this Chapter, we introduce and motivate the fine-tuning approach (c.f. Section 3.1 within a clinical context. We then apply this approach to the statistical methods introduced in Section 3.3 and evaluate their performance on the datasets discussed in Chapter 2.1.

3.1 Motivation

As we have seen in Chapter 1, even tough enough training data may be available, employing models trained in one hospital to another presents challenges. However, in scenarios where data from the target hospital is available, there might not be enough data to train a model, or it simply may not be desirable to do. In scenarios where there is a relatively small sample size of the target available, later called fine-tuning dataset, or could be quickly acquired, we propose an approach leveraging both our training data, which contains a large number of observations, and the fine-tuning dataset: We train the model parameters using the data from the train hospital while selecting hyperparameters minimizing the mean squared error on the tuning data from the target hospital. The idea/objective of this tuning dataset is to accurately capture the characteristics of our target hospital, thus improving the model's robustness. For this approach to work, we restrict ourselves to methods including hyperparameters.

3.2 Model Evaluation

We evaluate the prediction of the target variable using the mean squared error (MSE).

Definition 3.2.0.1 (MSE)**.** *Given a vector $y \in \mathbb{R}^n$ and its estimate $\hat{y}$, we define the mean squared error as*

$$MSE := \frac{1}{n} \sum_{i=1}^{n} (y_i - \hat{y}_i)^2$$

Thus, the MSE measures how close the estimated $\hat{y}$ are to the original y by considering the squares of errors. The lower the MSE, the closer are the fitted values to the original values. Note that one strong outlier may corrupt this score function.

For models that require hyperparameter tuning, we select the hyperparameters using a variant of K-fold Cross-Validation (Hastie, Tibshirani, and Friedman (2009)).

Definition 3.2.0.2 (K-fold CV)**.** *Given data* $(X_1, Y_1), \ldots, (X_n, Y_n) \in \mathbb{R}^2$*, with* Y_i *the response of the feature* X_i*. We randomly partition the data set with indices* $\{1, \ldots, n\}$ *into* K *equally sized (as equal as possible) subsets* B_k *of* $\{1, \ldots, n\}$ *such that* $\cup_{k=1}^{K} B_k = \{1, \ldots, n\}$ *and* $B_j \cap B_k = \emptyset$ $(j \neq k)$*. We can now set aside a* k*-th test data set including all sample points whose indices are elements of* B_k*. K-fold cross-validation then uses the sample points with indices not in* B_k *as training set to construct a regression estimator*

$$\widehat{m}_{n-|B_k|}^{(-B_k)}.$$

Thus, $\widehat{m}_{n-|B_k|}^{(-B_k)}$ *is the regression estimator obtained by training using all observations whose indicies are in* $\{1, \ldots, n\} \setminus B_k$*. The cross-validated performance of estimator* $\widehat{m}(\cdot)$ *is then,*

$$K^{-1} \sum_{k=1}^{K} |B_k|^{-1} \sum_{i \in B_k} \rho\left(Y_i, \widehat{m}_{n-|B_k|}^{(-B_k)}(X_i)\right),$$

where $\rho(\cdot, \cdot)$ *is a loss function.*

This definition can easily be generalized for p-dimensional feature vectors X_i as long the regression estimator estimate $\hat{Y}$ is in $\mathbb{R}$. In this work, we choose the l_2-loss function: $\rho(Y_i, \hat{Y}_i) = (Y_i - \hat{Y}_i)^2$

3.3 Models

For both pre-implemented and custom models, we used the following Python libraries: LightGBM version 4.1.0 and Scikit-learn version 1.3.2. Using these libraries, we evaluated the predictive performance of Ridge Regression (Scikit-learn), Random Forest (LightGBM), GBDT (LightGBM), Anchor Regression, a gradient boosted[1] version of Anchor regression called Anchor Boost, and a Scikit-learn compatible version of a GBDT allowing to refit[2] called RefitLGBM.

We employed the implementation of Anchor Regression developed by Malte Londschien. Information on how to access this implementation are provided in the GitHub repository.

Before applying the fine-tuning approach to ICU data, we introduce different approaches to model the relationship between the response $y \in \mathbb{R}^n$ and the design matrix $X \in \mathbb{R}^{n \times p}$. We organize the p features in the design matrix X, where each row corresponds to one ICU stay and each column to one feature.

3.3.1 Ridge

Ridge Regression can be similarly expressed as OLS (c.f. Appendix B.2.1) but also penalizes the L_2-norm of the coefficient vector:

$$\widehat{\Theta} = \arg\min_{\Theta \in \mathbb{R}^p} \|y - X\Theta\|_2^2 + \lambda \|\Theta\|_2 = \arg\min_{\Theta \in \mathbb{R}^p} \|y - X\Theta\|_2^2 = (X^T X + \lambda \mathrm{Id})^{-1} X^T y \tag{3.3.1.1}$$

As the regularization parameter λ increases, the Ridge estimator's coefficients are shrunk towards zero, but they are not set exactly to zero (Hastie et al. (2009)).

[1] That is, we sequentially fit the resulting residuals until a stopping criteria is met, e.g. a number of boosting rounds.

[2] We keep the structure of the tree and add new data to the leaves following the trees decision rules, thus updating or refitting the values of the leaves.

3.3.2 Lasso

The Least Absolute Shrinkage and Selection Operator (LASSO) can be similarly expressed as the Ridge estimator but penalizes the absolute magnitude of the coefficient vector:

$$\widehat{\Theta} = \arg\min_{\Theta\in\mathbb{R}^p} \|y - X\Theta\|_2^2 + \lambda \|\Theta\|_1 = \arg\min_{\substack{\Theta\in\mathbb{R}^p \\ \|\Theta\|_1<\lambda}} \|y - X\Theta\|_2^2 \text{ [3]} \tag{3.3.2.1}$$

Even though we do not have a closed form solution for equation 3.3.2.1 we can solve it easily via optimization, since the function $\Theta \in \{\Theta \in \mathbb{R}^p \mid \|\Theta\|_1 < \lambda\} \to \|y - X\Theta\|_2^2$ is continuous and convex. Tibshirani (2011) shows that the LASSO solution tends to be sparse. That is $\Theta = 0$ for most $i = 1, \ldots, p$. The larger λ, the more $\Theta_i = 0$, and hence the simpler the resulting model.

3.3.3 Random Forest

To define a Random Forests (RF) introduced by Breiman (2001), we will first define what a Tree is. A (decision) Tree is a graph without circles and a distinct root node. Every node has at most two children and every leaf has a value assigned to it. At each non-leaf node there is a boolean condition testing if one variable is greater than some value and a pointer to one child depending on the boolean value. To evaluate a tree we start at the root node, test the boolean expression and go to the node indicated by the resulting pointer. We repeat this until we reach at a leaf, where we return the value assigned to it. To build such a Tree, we will recursively partition the feature space using greedy splits[4] decreasing the MSE[5] each time. If the set we want to split contains less than a certain amount of training points, we stop. To build a Random Forest we will bootstrap-aggregate[6] many such Trees[7]. The prediction of the Random Forest for a new point x is then the mean of the predictions from all the Trees.

3.3.4 GBDT

Gradient boosting decision tree's or GBDT for short is an ensemble method using decision trees. In contrast to a RF, we do not train decision trees in parallel but in sequence. In each iteration, GBDT learns the decision trees by fitting the negative gradients (also known as residual errors). The prediction of the GBDT is then the combined prediction of these sequentially fitted trees. For more details on GBDT's, we refer to Friedman (2001).

3.3.5 Anchor Regression

In addition to the assumption of a linear relationship between response variable and features, we assume the presence of q exogenous variables, called "anchors", generating heterogeneity, i.e. variables that are not part of the features but can potentially influence

[3] The last two terms are equivalent by lagrangian optimization.

[4] For computational reasons, we will only use splits along one feature. So we 'cut' our feature space into rectangles.

[5] To calculate the MSE, we need a prediction. Let P be the current partition, then the predicted value for some $x \in A, A \in P$ is the mean of the responses of all the points in A (included in the training data).

[6] That is we will sample (with replacement) several times n observations from our original data and fit a Tree to each such sample.

[7] Building the Tree, this time we will not test every feature at each node (for the MSE minimization) but a node-specific subsample of the features. Thus, also the 'second best split' can be selected.

both the response and features. Thus, in addition to our design matrix X and response vector y, we also have a matrix $A \in \mathbb{R}^{n \times q}$ containing observations of the anchors.

The Anchor estimator is then obtained by:

$$\widehat{\Theta} = \arg\min_{\Theta} \|(\mathrm{Id} - \Pi_A)(Y - X\Theta)\|_2^2 + \gamma \|\Pi_A(Y - X\Theta)\|_2^2$$

where $\Pi_A \in \mathbb{R}^{n \times n}$ is the projection matrix on the column space of A. If $A^T A$ is invertible, we can write $\Pi_A = A(A^T A)^{-1} A^T$. The estimator can be computed by running OLS (c.f. Appendix B.2.1) on $\tilde{X} = (\mathrm{Id} + (\sqrt{\gamma} - 1)\Pi_A)X$ and $\tilde{Y} = (\mathrm{Id} + (\sqrt{\gamma} - 1)\Pi_A)X$. Rothenhäusler et al. (2020) showed that Anchor Regression is robust against linear shifts in the mean of the distribution with γ controlling the strength of the shift. For more details on Anchor Regression, we refer to Rothenhäusler et al. (2020).

3.3.6 Anchor Boost

We present Anchor Boost for $X, Y \in \mathbb{R}$, however they can easily be generalized to higher dimensions of X, where the feature X is then a p-dimensional feature vector. Anchor Boost works as follows:

(1) Fit Anchor Regression from the data $\{(X_i, Y_i); i = 1, \ldots, n\}$, yielding a first function estimate $\widehat{m}_{Anchor}(\cdot)$

(2) Compute residuals

$$U_i = Y_i - \widehat{m}_{Anchor}(X_i), \ (i = 1, \ldots, n)$$

(3) Fit the residuals $\{(X_i, U_i); i = 1, \ldots, n\}$ using a GBDT, yielding a function estimate of the residuals

$$(X_i, U_i) \to \widehat{m}_{GBDT}(\cdot)$$

(4) Return the boosted estimate of Y_i:

$$\hat{Y}_i = \widehat{m}_{Anchor}(X_i) + \widehat{m}_{GBDT}(X_i) \text{ for } i = 1, \ldots, n$$

This can be implemented in Python as follows:

```python
from sklearn.base import BaseEstimator, RegressorMixin
from sklearn.model_selection import KFold

from ivmodels.anchor_regression import AnchorRegression

class AnchorBoost(BaseEstimator, RegressorMixin):
    """
    Boosted version of Anchor Regression.
    The residuals of Anchor Regression are fitted using a LGBMRegressor.

    Parameters
    ----------
    anchor_params: dict
        Parameters for the Anchor Model.

    lgbm_params: dict
        Parameters for the LGBMRegressor
    """
    def __init__(self, anchor_params=None, lgbm_params=None):
        self.anchor_params = anchor_params if anchor_params is not None else {}
        self.lgbm_params = lgbm_params if lgbm_params is not None else {}

    def fit(self, X, y):
```

```
        self.anchor_model = AnchorRegression(**self.anchor_params)
        self.anchor_model.fit(X, y)
        residuals = y - self.anchor_model.predict(X)
        self.lgbm_model = LGBMRegressor(**self.lgbm_params)
        self.lgbm_model.fit(X, residuals)
        return self

    def predict(self, X):
        if not hasattr(self, 'anchor_model') or not hasattr(self, 'lgbm_model'):
            raise AttributeError("Models have not been fitted. Call fit() first."
                )
        anchor_predictions = self.anchor_model.predict(X)
        lgbm_predictions = self.lgbm_model.predict(X)
        return anchor_predictions + lgbm_predictions
```

3.3.7 RefitLGBM

We present RefitLGBM for $X, Y \in \mathbb{R}$, however they can easily be generalized to higher dimensions of X, where the feature X is then a p-dimensional feature vector. RefitLGBM works as follows:

(1) Fit a GBDT or Random Forest on the data $\{(X_i, Y_i); i = 1, \ldots, n\}$ (for a given hyper-parameter combination)

(2) Add the observations of the fine-tuning dataset $\{(X_i^{tune}, Y_i^{tune}); i = 1, \ldots, n_{tune}\}$ to the tree-structure in (1); i.e. add the observations Y_i of the fine-tuning dataset to the leaves of the tree-structure learned in (1) by following the decision rules established in (1) using X_i.

Therefore, we essentially update the leaves learned through the training data by adding the observations from the fine-tuning dataset.

This can be implemented in Python as follows:

```
from sklearn.base import BaseEstimator

class RefitLGBMRegressor(BaseEstimator):
    """
    LGBM Regressor that gets refit on new data.

    Parameters
    ----------
    prior : LGBMRegressor
        Prior model.
    decay_rate : float
        Decay rate for refitting. If `decay_rate=1`, the new data is ignored.
    """

    def __init__(self, prior=None, decay_rate=0.5):
        self.prior = prior
        self.decay_rate = decay_rate

    def fit(self, X, y):
        if not isinstance(self.prior, LGBMRegressor):
            raise ValueError("Prior must be a LGBMRegressor")
        self.model = copy.deepcopy(self.prior)
        new_booster = self.model.booster_.refit(
            data=X, label=y, decay_rate=self.decay_rate, n_jobs=1
        )
        self.model._Booster = new_booster
        return self

    def predict(self, X):
        return self.model.predict(X)
```

3.4 Application to ICU data

3.4.1 Experiment setting

We predicted the average value of the heart rate 48-72h after admission to the ICU, i.e. the average value on day 3, using static variables and the average value of the dynamic variables 0-24h after entry to the ICU, i.e. the average values of dynamic variables on day 1. We removed patient visits if there was no measurement of the heart rate in the time windows 0-24h or 48-72h. Figure 3.1 displays the density of the average heart rate on the third day post-admission to the ICU across datasets. Figure 3.1 has been generated using a kernel density estimator from Pythons Seaborn package, version 0.13.0. The density of MIMIC-III displays substantial differences from the other datasets, thus making it a point of interest for the application and examination of the fine-tuning approach (c.f. Section 3.1).

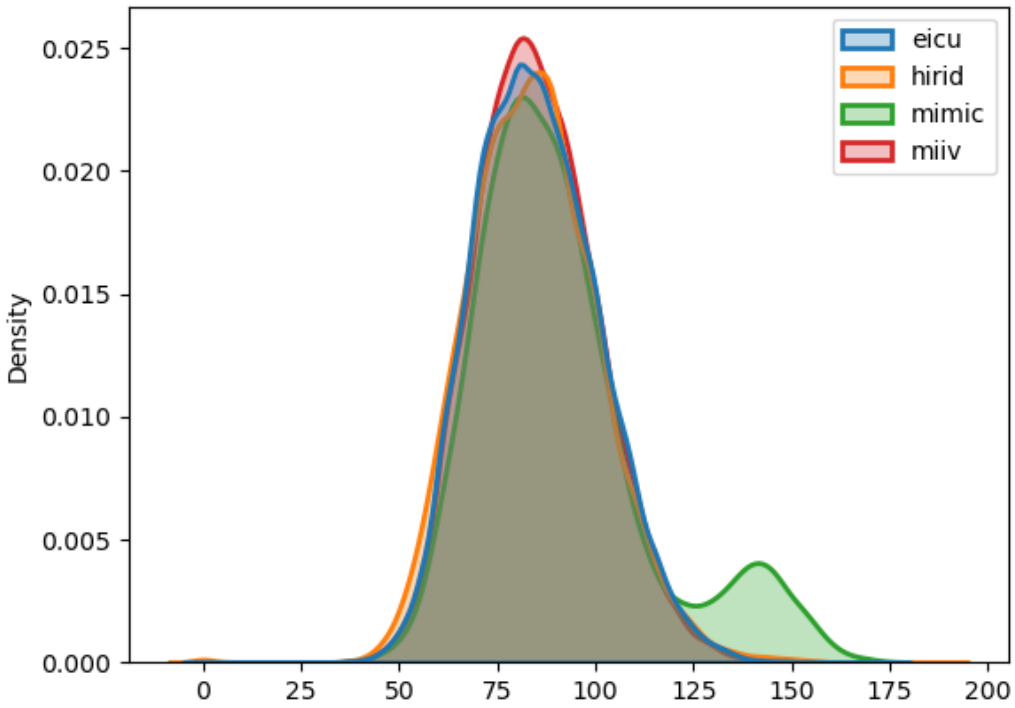


Figure 3.1: Densities of heart-rate at day 3, post-admission to the ICU, across datasets.

We used the eICU dataset for training our models and used a small fine-tuning dataset from the respective target dataset (MIMIC-III, MIMIC-IV or HiRID) consisting of n_{tune} data points. We used these n_{tune} data points for hyperparameter selection ($n_{tune} \in \{25, 50, 100, 200, 400, 800, 1600\}$). We compared the fine-tuning approach with Ridge Regression, GBDT, and Random Forest when selecting hyperparameters via the classic ERM approach, i.e. minimizing the MSE on the training data using 5-fold CV. We will refer to models obtained via the classic ERM approach as "baseline". As discussed in Section 3.1, we use a tuning dataset from the target hospital, distinct from the final evaluation dataset. To maintain a consistent sample size for the evaluation dataset, we

initially excluded 1600 data points from our target data source. Afterwards, we created the fine-tuning dataset from these excluded observations. The remaining data from the target is the evaluation dataset and will be later called by the name of the data source, e.g. MIMIC-III dataset.

We now present the fine-tuning approach as an algorithm:

(1) Select 1600 observations from the target dataset.

(2) For all hyperparameter combinations:

  (1) Fit the model using this combination of hyperparameters with the training data.

  (2) For $k = 1, \ldots, 10$:

    (1) Draw n_{tune} data points (randomly) from the 1600 observations.

    (2) Calculate the 5-fold CV error.

  (3) Calculate the average of the 5-fold CV error on these n_{tune} data points.

(3) Select the hyperparameter combination with the lowest average 5-fold CV error.

(4) Evaluate the model on the evaluation data (target dataset but (1) excluded)

We selected 10 times n_{tune} data points to mitigate the effect of poor evaluation points when dealing with a small number of observations.

Model	**Hyperparameters**
Ridge Regression	λ: 0.00001, 0.0001, 0.001, 0.01, 0.1, 1
GBDT, Random Forest	Learning rate: 0.01, 0.1, 0.3 n_estimators: 100, 800 num_leaves: 50, 200, 1024 feature_fraction: 0.5, 0.9
RefitLGBM	decay_rate: 0, 0.1, 0.3, 0.5, 0.7, 0.9, 1
Anchor Regression	γ: 1, 3.16, 10, 31.6, 100, 316, 1000, 3162, 10000 λ: 0.00001, 0.0001, 0.001, 0.01, 0.1 l1_ratio: 0, 0.2, 0.5, 0.8, 1
Anchor Boost	Anchor Regression: γ: 1, 10, 100 λ: 0.0001, 0.01 l1_ratio: 0, 0.5 GBDT: Learning rate: 0.01, 0.1, 0.3 n_estimators: 100, 800 num_leaves: 50, 200, 1024 feature_fraction: 0.5, 0.9

Table 3.1: Hyperparameters for Different Models

For AnchorBoost, we selected the hyperparameters simultaneously, meaning we chose the optimal combination for both Anchor Regression[8] and the booster concurrently (c.f. Table

[8] Elastic Net Regularization is used instead of OLS to compute Anchor Regression, hence the appearance of l1_ratio.

3.1). For RefitLGBM, we followed a similar approach, identifying the best combination of hyperparameters available for GBDT, and the decay_rate, responsible for regulating the learning rate during refitting, simultaneously. That is, we chose the best combination of learning rate, n_estimators, num_leaves, feature_fraction and decay_rate for RefitLGBM simultaneously (c.f. Table 3.1).

In this experiment, we restricted ourselves to the eICU dataset as training's data due to the necessity of anchors in Anchor Regression. We chose the eICU dataset because it encompasses data from multiple hospitals, allowing us to designate hospital ID as anchors.

3.4.2 Model specific preprocessing

For all methods, we mean-imputed missing dynamic variables. Afterwards, we centered and scaled the dynamic variables to unit variance. In Ridge Regression, Anchor Regression, and AnchorBoost we included a missingness indicator (0/1) to enable the model to distinguish between imputed and original values. Also in these models, missing values of the categorical variable "sex" were marked as "missing" and subsequently one-hot encoded, yielding to the categories "Male", "Female" and "Missing" for the variable "sex". Conversely, no one-hot encoding of categorical variables was performed for the Random Forest, GBDT, or RefitLGBM.

Afterwards, the eICU dataset contained 74587 patient visits from 188 different hospitals, the MIMIC-III dataset contained 30335 patient visits, the MIMIC-IV dataset contained 35673 patient visits and the HiRID dataset contained 8577 patient visits[9].

We did not perform preprocessing over the entire 1600 separated observations. Preprocessing of the tuning dataset only included the n_{tune} fine-tuning observations.

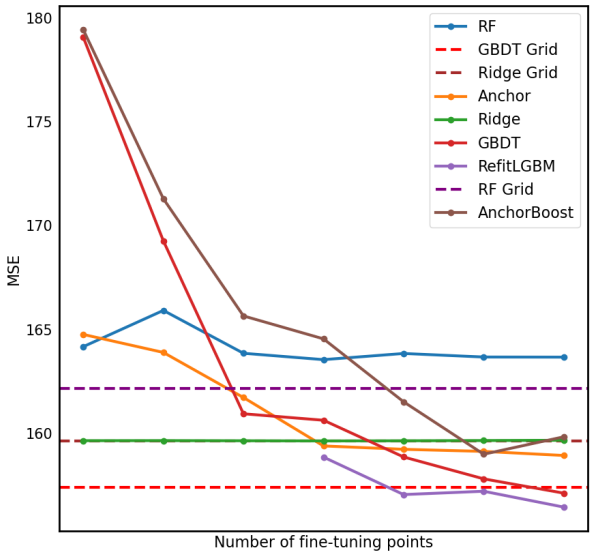
3.4.3 Results and Discussion

Figure 3.2 displays the optimal MSE achieved by our methods using n_{tune} ($n_{tune} \in \{25, 50, 100, 200, 400, 800, 1600\}$) fine-tuning data points. The dotted lines indicate the baseline MSE.

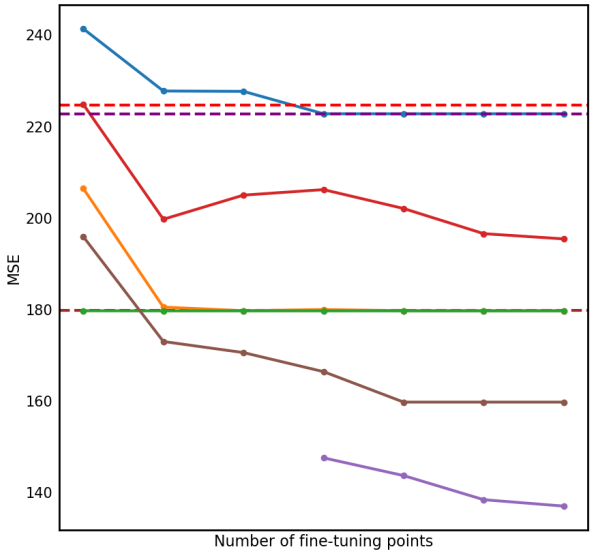
We applied a logarithmic transformation to the x-axis to address the skewed distribution and varying magnitudes of the fine-tuning data points, aiming to enhance interpretability. Given the 1600 observations in the fine-tuning dataset were extracted before applying model specific preprocessing, there is a higher likelihood of encountering missing observations for at least one feature when selecting a small number of tuning data points, e.g. $n_{tune} = 25$, leading to preprocessing failures since at least one observation is required to impute values. To mitigate this issue, we required a minimum of 200 observations for RefitLGBM.

In Figure 3.2b, AnchorBoost and RefitLGBM demonstrated superior performance and increased robustness compared to the baseline models on the dataset with notable distributional differences (MIMIC-III). Anchor Regression failed to outperform Ridge Regression, possibly due to the absence of a mean shift in the MIMIC-III dataset. RefitLGBM showed the best performance, suggesting that even a relatively small tuning dataset can help capturing key distributional characteristics[10]. Conversely, the Random Forest performed poor on MIMIC-III, failing to exceed its baseline performance. The fine-tuning approach had

[9] The fine-tuning dataset is not included.
[10] Thus improving robustness.



(a) Evaluation on HiRID.



(b) Evaluation on MIMIC-III.

Figure 3.2: MSE on the evaluation part of the dataset vs. number of tuning data points.

no noticeable impact on the robustness of Ridge Regression, whereas the performance of GBDT increased, although not to a level indicating robustness. Additionally, Figure 3.2a illustrates that when distributional differences are less pronounced, the improvement in AnchorBoost diminished but still performed well. Furthermore, the Random Forest failed to surpass its baseline on HiRID, while RefitLGBM remained top performer, albeit not as

impressive as on MIMIC-III.

Figure 3.2 demonstrates, that increasing the number of fine-tuning data points doesn't guarantee monotone performance improvements. Only Anchor Regression exhibited monotonicity.

In conclusion, the Random Forest was not well-suited for the fine-tuning approach, while AnchorBoost and RefitLGBM suggest improved robustness. In cases where such disparities were minimal, both the classical ERM approach, minimizing the MSE on the train data, and the fine-tuning approach worked reasonably well[11].

The outcomes obtained when evaluated on MIMIC-IV resembled those observed in HiRID and are depicted in Appendix B.2.2.

In this section, we have assessed various statistical learning methods using a fine-tuning approach. We introduced Anchor Regression, a method aiming to enhance efficacy in cases of mean shifts in the distribution. In the following section, we will introduce another statistical learning method designed to identify common effects among groups, potentially enhancing robustness.

[11] For $n_{tune} > 50$.

Chapter 4

Magging

In the previous Chapter, we explored an approach which aimed to improve model performance using a tuning dataset from the target. In this Chapter and in Chapter 5, our focus shifts to methods capable of handling multiple datasets, possibly originating from different distributions. In this Chapter, we introduce Magging, a method proposed by Bühlmann and Meinshausen (2016), and motivate its use in intensive care settings. We present algorithms for generating the Magging estimator using linear and non-linear methods in the ensemble. We conclude this Chapter by applying Magging to the datasets outlined in Chapter 2.

4.1 Motivation

When training models using multiple datasets or hospitals, it is unclear how to handle those datasets. For example, we could pool[1] the datasets, thus consider them as one large dataset, potentially overlooking underlying structural differences (c.f. Chapter 1) or use aggregation methods such as Bagging[2]. While Bagging is effective in homogeneous settings, its use in heterogenous settings is non-trivial[3] (Bühlmann and Meinshausen (2016)). Magging offers one possible solution to this problem. Magging aggregates models trained on individual datasets without assuming homogeneity, identifying common effects across all datasets. This offers a promising approach to improve distributional robustness by capturing shared effects across hospitals, which may also be present in future hospitals.

4.2 Introducing the Magging estimator

The fundamental idea of the Magging (**M**aximin **agg**regat**ing**) estimator is to partition the training data into G disjoint groups, leveraging the predictions of each group member to form an ensemble prediction. In this Section, we introduce the Magging estimator for linear ensemble members. A generalization to non-linear estimators can be found in Section 4.3. We adopt the definition given in Bühlmann and Meinshausen (2016):

Assume a design matrix X of dimensions $n \times p$, containing n samples of a p-dimensional predictor variable and a response vector $Y \in \mathbb{R}^n$. Suppose there exist G groups, such that

[1] Similar to the classic ERM approach outlined in Chapter 1.
[2] c.f. Section 3.3.3.
[3] e.g. why should estimators share equal aggregation weight?

the design matrix X and response vector Y can be partitioned into G disjoint groups, $X = (X(1), \ldots, X(G))^T$ where $X(g)$ has dimensions $n_g \times p$ and $Y = (Y(1), \ldots, Y(G))^T$ where $Y(g)$ is of length n_g such that $n_1 + \cdots + n_G = n$. We assume linear relationships in the groups, i.e. $Y(g) = X(g)\Theta^*(g)$, where Θ_g^* denotes the true underlying relationship. We assume to observe the same features in all groups. Let $\widehat{\Theta}_g$ be the regression coefficient estimate derived from data in subgroup $g \in \{1, \ldots, G\}$. That is $\widehat{Y}(g) = X(g)\widehat{\Theta}_g$. Consider now the collection $\{\widehat{\Theta}_g \in \mathbb{R}^p\}_{g=1}^G$. We aggregate these to a unified estimator $\widehat{\Theta}_{\text{agg}}$ as follows:

$$\begin{aligned} \textbf{Magging:} \quad \widehat{\Theta}_{\text{agg}} &:= \sum_{g=1}^{G} w_g \widehat{\Theta}_g, \\ \text{where } \mathbf{w} &:= \underset{\mathbf{w} \in C_G}{\operatorname{argmin}} \left\| \sum_g w_g X \widehat{\Theta}_g \right\|_2, \\ \text{and } C_G &:= \left\{ \mathbf{w} : \min_g w_g \geq 0 \text{ and } \sum_g w_g = 1 \right\} \end{aligned} \tag{4.2.0.1}$$

The Magging prediction is then given by: $\widehat{Y}_{\text{magging}} = \mathbf{w}^T \widehat{Y} = \sum_{g=1}^G w_g X \widehat{\Theta}_g$.

Note that even though the aggregation is a weighted average of the ensemble members, the weights are non-uniform. Moreover, the weights are independent of the response Y. If the solution is not unique Bühlmann and Meinshausen (2016), propose to choose the solution with the lowest l_2 norm of the weight vector.

The generic structure of Magging enables each group to choose its preferred regression estimator, offering flexibility in selecting the regression estimator in both linear and non-linear settings. In Section 4.3, we present the algorithms to implement Magging using both linear and non-linear methods in Python.

Bühlmann and Meinshausen (2016) demonstrated that the Magging scheme for heterogeneous data corresponds to "maximizing the minimally explained variance"[4] across all groups.

Moreover, Bühlmann and Meinshausen (2016) offer the following geometric interpretation for the Magging estimator: Let Σ represent the covariance matrix of X and consider the convex hull $\tilde{H}$ of the collection $\{\hat{\Theta}_g \in \mathbb{R}^p\}_{g=1}^G$. The Magging estimate $\widehat{\Theta}_{agg}$ is the vector closest to zero within $\tilde{H}$ with respect to the distance $d(u, v) = (u - v)^\top \Sigma (u - v)$.

4.3 The Algorithm

The Magging Estimator introduced in Section 4.2 is computed in two steps. First, for each group, we estimate the group-specific regression coefficient. Secondly, we determine the Magging weights by solving the constrained convex optimization problem (see 4.2.0.1). Remember that in the linear case $\hat{Y} = X\widehat{\Theta}$. Thus, in 4.2.0.1 we minimize the objective function $\frac{1}{n}\| \sum_g X \widehat{\Theta}_g w_g \|_2$, where $\widehat{\Theta}_g$ denotes the g-th regression coefficient estimates. We adopt the algorithm proposed in Bühlmann and Meinshausen (2016) for the linear case. Later in this Section, we will derive the Magging estimator in the non-linear case using

[4] Intuitively, maximizing the effects that are common across all groups. The idea is that if an effect is common across all groups, it cannot be averaged out; such effects persist even after the minimization of the aggregation scheme minimization.

the plug-in principle. In the linear case, the constrained optimization problem 4.2.0.1 for computing the weights can be reformulated as:

$$
\begin{aligned}
\underset{w}{\text{minimize}} \quad & \frac{1}{2} w^T H w \\
\text{subject to} \quad & Gw \leq h, \\
& Aw = b, \\
\text{where} \quad & H = \frac{1}{n} (X\hat{\Theta}_1 \dots X\hat{\Theta}_G)^T (X\hat{\Theta}_1 \dots X\hat{\Theta}_G) \text{ of size G} \times \text{G} \\
& G = -I_G \\
& A = \begin{bmatrix} 1^T \\ I_G \end{bmatrix} \text{ of size } (G+1) \times G \\
& h = 0 \text{ of size G} \times 1 \\
& b = 1^T \text{ of size G} \times 1
\end{aligned}
$$

We use the following Python packages for implementing Magging: CVXOPT version 1.3.2 and NumPy version 1.26.0. We can implement Magging in the linear case as follows:

```
import numpy as np
from cvxopt import matrix, solvers

# Assuming theta1,...,thetaG are NumPy arrays of regression estimates
theta = np.column_stack((theta1, ..., thetaG))

# Empirical covariance matrix of X
hatS = (X.T @ X) / n

# Assuming H is the matrix we want to be positive definite
# If not positive definite, add a small value to the diagonal
H = theta.T @ hatS @ theta
xi = 1e-10  # A small value, if necessary
if not is_positive_definite(H):
    H += xi * np.eye(H.shape[0])

# Constraints
A = np.vstack((np.ones((1, G)), np.eye(G)))
b = np.hstack((1, np.zeros(G)))

# Converting to cvxopt matrices
P = matrix(H)
q = matrix(np.zeros(G))  # Linear term is zero
G = matrix(-np.eye(G))   # Negative identity for inequality constraints
h = matrix(np.zeros(G))  # Zero vector for inequality constraints
A = matrix(A)
b = matrix(b)

# Solve QP problem
sol = solvers.qp(P, q, G, h, A, b)

# Weights vector w
w = np.array(sol['x']).flatten()
```

Next, we extend Magging to allow non-linear methods for group-specific estimation using the plug-in principle. In contrast to Section 4.2, where we assumed linear relationships in the groups, we now allow non-linear relations, i.e. $Y(g) = m_g(X(g))$, where m_g is some function representing the true relationship. We estimate $\widehat{Y}(g) = \widehat{m}_g(X(g))$. Hence, we can employ a similar approach as above, minimizing $\frac{1}{n}\|\sum_g \widehat{m}_g(X)w_g\|_2$.

After applying the plug-in principle, we arrive at

$$H = \frac{1}{n}(\widehat{m}_1(X) \dots \widehat{m}_G(X))^T(\widehat{m}_1(X) \dots \widehat{m}_G(X)).$$

Therefore, in the non-linear case, we can simplified above code, by calculating:

```
mhat = np.column_stack((m_1, ..., m_G))
H = mhat.T @ mhat
```

4.3.1 Example

In this Section, we demonstrate how the obtained weights from the optimization problem in 4.2.0.1 are used to make predictions on external datasets. Consider a scenario where the data from the eICU dataset can be divided into three distinct groups, denoted as A, B, and C. We represent the design matrix and response variable as follows:

$$X_{eICU} = \begin{pmatrix} A \\ B \\ C \end{pmatrix}, \quad Y_{eICU} = \begin{pmatrix} m_A(A) \\ m_B(B) \\ m_C(C) \end{pmatrix}$$

Assume $\widehat{m}_A(\cdot)$, $\widehat{m}_B(\cdot)$ and $\widehat{m}_C(\cdot)$ are the obtained group-specific estimators. We now apply these estimators on the entire X_{eICU} and obtain a collection of estimators, which we denote as $\widehat{m}_A(X_{eICU})$, $\widehat{m}_B(X_{eICU})$, and $\widehat{m}_C(X_{eICU})$.

Let w_A, w_B, and w_C be the weights obtained from the quadratic optimization problem. These weights encapsulate the contribution of each group's estimator in the ensemble prediction. Now, we can predict the response vector on the (unseen) HiRID dataset:

$$\hat{Y}_{HiRID} = w_A\widehat{m}_A(X_{HiRID}) + w_B\widehat{m}_B(X_{HiRID}) + w_C\widehat{m}_C(X_{HiRID})$$

4.4 Application to ICU data

In this Section, we calculate the Magging estimator using multiple datasets and evaluate its performance on unseen datasets. Example: First, estimate the response variable separately on both the eICU dataset and the HiRID dataset. Then, use these two estimators as an ensemble to calculate the Magging estimator. Finally, use the obtained Magging estimator to predict the response variable on the MIMIC-III and MIMIC-IV dataset. A more detailed explanation can be found in Section 4.4.3. In a first experiment, we used Lasso Regression (c.f. Section 3.3.2) as a linear method to obtain group-specific estimations and repeated this using a Random Forest (c.f. Section 3.3.3) as non-linear method.

4.4.1 Experiment setting

As in Section 3.4.1, we predicted the average heart-rate on day three (48h-72h after admission to the ICU) using the average values of dynamic features at day 1 (0h-24h after admission) and static features. We selected the hyperparameters for both methods through 5-fold CV[5] by minimizing the MSE in the respective group. The regularization parameter in Lasso Regression can take values between 0.01 and 10, while the hyperparameters for Random Forest were selected from Table 4.1.

[5] See Section 3.2.

Hyperparameter	Values
learning_rate	0.01, 0.1, 0.3
n_estimators	100, 800
num_leaves	50, 200, 1024
feature_fraction	0.5, 0.9

Table 4.1: Hyperparameters for the Random Forest

For the implementation of Random Forest and Lasso, we refer to Section 3.3.

4.4.2 Model specific preprocessing

For Lasso Regression, we employed the same preprocessing as for Ridge Regression in Section 3.4.2. Similarly, for Random Forest, we applied the preprocessing procedures described in Section 3.4.2. Therefore, the resulting datasets were identical to those in Section 3.4.2.

4.4.3 Results and Discussion

Table 4.2 and Table 4.3 displays the result when applying the Magging estimator on the respective "Target" dataset when trained on the "Train Groups".

Target	Train Groups	Weights	MSE	MSE on Test	p.d.
eicu	mimic, hirid	0.0, 1.0	150.82	150.55, 150.82	T
eicu	mimic, miiv	0.0, 1.0	148.89	150.55, 148.89	T
eicu	hirid, miiv	1.0, 0.0	150.82	150.82, 148.89	T
eicu	mimic, hirid, miiv	0.0, 1.0, 0.0	150.82	150.55, 150.82, 148.89	T
hirid	mimic, eicu	0.0, 1.0	158.76	164.91, 158.76	T
hirid	mimic, miiv	0.0, 1.0	162.22	164.91, 162.22	T
hirid	eicu, miiv	0.0, 1.0	162.22	158.76, 162.22	T
hirid	mimic, eicu, miiv	0.0, 0.0, 1.0	162.22	164.91, 158.76, 162.22	T
miiv	mimic, hirid	0.0, 1.0	139.90	137.99, 139.90	T
miiv	mimic, eicu	0.0, 1.0	138.41	137.99, 138.41	T
miiv	hirid, eicu	1.0, 0.0	139.90	139.90, 138.41	T
miiv	mimic, hirid, eicu	0.0, 1.0, 0.0	139.90	137.99, 139.90, 138.41	T
mimic	hirid, eicu	1.0, 0.0	169.73	169.73, 181.54	T
mimic	hirid, miiv	1.0, 0.0	169.73	169.73, 173.80	T
mimic	eicu, miiv	0.0, 1.0	173.80	181.54, 173.80	T
mimic	hirid, eicu, miiv	1.0, 0.0, 0.0	169.73	169.73, 181.54, 173.80	T

Table 4.2: Magging Results using Lasso for prediction of ensemble members

Target	Train Groups	Weights	MSE	MSE on Test	p.d.
eicu	mimic, hirid	1.0, 0.0	159.94	159.94, 152.96	T
eicu	mimic, miiv	0.0, 1.0	151.49	152.96, 151.49	T
eicu	hirid, miiv	1.0, 0.0	159.94	159.94, 151.49	T
eicu	mimic, hirid, miiv	1.0, 0.0, 0.0	159.94	159.94, 152.96, 151.49	T
hirid	mimic, eicu	1.0, 0.0	161.96	161.96, 165.03	T
hirid	mimic, miiv	0.0, 1.0	162.33	165.03, 162.33	T
hirid	eicu, miiv	0.892, 0.108	*161.39*	161.96, 162.33	T
hirid	mimic, eicu, miiv	0.0, 0.0, 1.0	162.33	161.96, 165.03, 162.33	T
miiv	mimic, hirid	1.0, 0.0	149.49	149.49, 140.45	T
miiv	mimic, eicu	1.0, 0.0	140.27	140.27, 140.45	T
miiv	hirid, eicu	0.0, 1.0	149.49	140.27, 149.49	T
miiv	mimic, hirid, eicu	0.0, 1.0, 0.0	149.49	140.27, 149.49, 140.45	T
mimic	hirid, eicu	0.0, 1.0	298.61	221.36, 298.61	T
mimic	hirid, miiv	1.0, 0.0	298.61	298.61, 263.55	T
mimic	eicu, miiv	0.892, 0.108	224.95	221.36, 263.55	T
mimic	hirid, eicu, miiv	0.0, 1.0, 0.0	298.61	221.36, 298.61, 263.55	T

Table 4.3: Magging results using Random Forest for prediction of ensemble members

The tables are structured as follows:

- ”Target” refers to the dataset on which we make predictions using the estimators trained on the training datasets.
- ”Train Groups” refers to the datasets we used as an ensemble to train the Magging estimator.
- ”Weights” correspond to the Magging weights assigned to each dataset.
- ”MSE” indicates the mean squared error achieved on the target data when applying the Magging Estimator.
- ”MSE on Test” represents the mean squared error achieved on the Target dataset when using only the respective ensemble member.
- ”p.d.” indicates whether the matrix H in Section 4.3 is positive definite.

For example, the first row of Table 4.2 should be understood as follows: We predict the response variable in the eICU dataset using the Magging Estimator trained using the ensemble MIMIC-III and HiRID. Magging assigns a weight of 0 to the estimator trained on MIMIC-III and a weight of 1 to the estimator trained on HiRID. The resulting Magging estimate of the response variable in the eICU dataset achieved a MSE of 150.82. When we predict the same response variable using only the Lasso trained on MIMIC-III, we obtain a MSE of 150.55. Similarly, using only the Lasso trained on HiRID, we achieve a MSE of 150.82. The matrix H is positive definite, avoiding the application of further regularization (cf. section 4.3).

We observed in Table 4.2 that Magging tends to favor one ensemble member over others. To understand this preference, we turn to Figure 5.6b and the geometric interpretation of the Magging estimator outlined in Section 4.2. The weights are influenced by the prevalence of a dominant feature[6]; this feature is the average heart rate at day one. From Section 4.2 we know that the point in the convex hull closest to zero[7] is selected. However, when a dominant feature exists, determining the closest p-dimensional point can be simplified to a 1-dimensional setting. Consequently, Magging will choose the group with the smallest average heart rate (c.f. Figure 3.1).

In the linear setting, Magging performed best on MIMIC-IV and "worst" on MIMIC-III, with a range of roughly 30 in MSE between best and worst prediction, suggesting some robustness. In the non-linear setting, Magging performed best on MIMIC-IV and worst on MIMIC-III with a range of over 100 in MSE between best and worst prediction. On HiRID, Magging performed similarly in both the linear and non-linear settings, but exhibited more pronounced differences on all other datasets, with the largest difference observed on MIMIC-III. In the linear setting, regardless of the combination of training datasets, Magging consistently achieved similar results on each train dataset suggesting robustness. However, in the non-linear setting, the estimations were less consistent.

The linear approach using Lasso outperformed the non-linear approach in 15 out of 16 cases and showd only negligible underperformance in the other case[8]. The non-linear approach using Random Forests exhibited no robustness when applied to MIMIC-III[9], significantly deteriorating performance compared to Lasso. This indicates that the feature selection characteristic of Lasso, as discussed in Section 3.3.2, plays a significant role in achieving distributional robustness of the Magging estimator.

In Appendix B.3, we repeated this experiment using different group selection methods, e.g. age groups or ethnicity, resulting in similar observations. We conclude by noting, that Magging achieved optimal weights in scenarios where access to test data is available and when training and test data could be segmented similarly. For further details, please refer to the Appendix.

In the following Chapter, we introduce another method, Data Shared Lasso, aiming to identify shared effects.

[6] That is, almost all other features are set zero. See Section 5.5.
[7] With respect to the Magging distance.
[8] Highlighted in Table 4.3.
[9] The dataset exhibiting the largest distributional difference.

Chapter 5

Data Shared Lasso

In this Chapter, we introduce Data Shared Lasso (DSL) and apply it in an ICU setting. We compare DSL with Magging using both ICU data and artificial data. We conclude this Chapter by examining DSL's efficacy in identifying similarities and differences among datasets.

5.1 Motivation

To train models on multiple datasets/hospitals, Gross and Tibshirani (2016) propose Data Shared Lasso, a model spanning the continuum between individual models for each dataset and one model for all datasets. Data Shared Lasso allows regression coefficients to vary sparsely between datasets. The resulting model captures both, shared and dataset specific coefficients, potentially improving distributional robustness by leveraging shared effects across datasets/hospitals.

5.2 Introducing Data Shared Lasso

Similar to Magging (refer to Chapter 4.2), we assume the existence of G non-overlapping groups that partition our observations that have some shared structure. That is, each group has the same set of predictors and the true underlying coefficients vary in some manner between the groups. We adopt the definition given in Gross and Tibshirani (2016):

Assume n observations of the form (x_i, y_i, g_i), where $x_i \in \mathbb{R}^p, y_i \in \mathbb{R}$ and $g_i \in \{1, \dots, G\}$. Here, p corresponds to the number of features and G is the number of groups we consider. Hence, each observation i is described by a feature vector x_i, a response variable y_i, and a group indicator g_i. The design matrix X[1] has the x_i's as rows and $y = (y_1, \dots, y_n)$. We denote by X_g the design matrix that has all rows x_i that are in group $g \in \{1, \dots, G\}$. We assume that the following relationship between the response variable y_i and the feature vector x_i:

$$y_i = x_i^T(\Theta^* + \Theta^*_{g_i}) + \varepsilon_i,$$

where $\Theta^* \in \mathbb{R}^p$ represents the shared coefficients[2], $\Theta^*_{g_i} \in \mathbb{R}^p$ represents the group-specific

[1] In the context of ICU data, each row corresponds to one ICU stay and each column to one feature.

[2] In contrast to Magging, shared coefficients/common effects in the DSL sense are not rigorously defined (c.f Section 4.2).

coefficients and $\varepsilon_i \in \mathbb{R}$ is an independent error term.

The Data Shared Lasso (DSL) estimate of the coefficient vector is then obtained by solving:

$$(\widehat{\Theta}, \widehat{\Theta}_1, \ldots, \widehat{\Theta}_G) = \underset{\Theta,\Theta_1,\ldots,\Theta_G}{\arg\min} \frac{1}{2}\sum_{i=1}^{n}(y_i - x_i^T(\Theta + \Theta_{g_i}))^2 + \lambda\|\Theta\|_1 + \sum_{g=1}^{G} r_g\|\Theta_g\|_1, \quad (5.2.0.1)$$

where λ is the global regularization parameter, and r_g are group-specific regularization parameters. The r_g's control the amount of pooling for each group. Large values of r_g indicate more sharing, while small values correspond to less sharing.

Gross and Tibshirani (2016) suggest to implement DSL using Lasso Regression (c.f. Section 3.3.2) and an augmented data approach[3]. This approach involves the construction of an augmented feature matrix and response vector as follows:

$$Z = \begin{pmatrix} X_1 & r_1X_1 & 0 & \ldots & 0 \\ X_2 & 0 & r_2X_2 & \ldots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ X_G & 0 & 0 & \ldots & r_GX_G \end{pmatrix} \in \mathbb{R}^{n\times(G+1)p},$$

Thus, we can rewrite 5.2.0.1 in standard Lasso form:

$$\frac{1}{2}\sum_i \|y_i - x_i^T(\Theta + \Theta_{g_i})\|_2^2 + \lambda\left(\|\Theta\|_1 + \sum_{g=1}^{G} r_g\|\Theta_g\|_1\right) = \frac{1}{2}\|y - Z\tilde{\Theta}\|_2^2 + \lambda\|\tilde{\Theta}\|_1,$$

$$y = (y_1^T, y_2^T, \ldots, y_G^T)^T \in \mathbb{R}^n \text{ and } \tilde{\Theta} = (\Theta^T, \frac{1}{r_1}\Theta_1^T, \ldots, \frac{1}{r_G}\Theta_G^T)^T \in \mathbb{R}^{(G+1)p}.$$

In this context y_i represents all response variables in group i.

5.3 Appplication to ICU data

5.3.1 Experiment setting

As in Section 3.4.1, we predicted the average heart-rate on day three (48h-72h after admission to the ICU) using the average values of dynamic features at day 1 (0h-24h after admission) and static features.

As indicated by Gross and Tibshirani (2016) and in Section 5.2, the selection of r_g controls the degree of sharing between the groups. To mitigate the influence of larger datasets, we set

$$r_g = 1/\sqrt{\text{group's relative size in train data}}.$$

This implies that smaller groups exhibit more sharing, whereas larger groups exhibit less. As outlined in Section 5.2, DSL can be implemented using augmented Lasso.

To implement DSL, we used the implementation of Lasso in the Scikit-learn version 1.3.2. The regularization term in the DSL was determined through minimizing the 5-fold CV[4]

[3] Data augmentation expands the original dataset by adding additional information or features.
[4] Section 3.2.

MSE on the train data. The regularization parameter λ can attain values in 0.001 and 10. For evaluation on the datasets, we used the shared coefficients $\widehat{\Theta}$[5] (c.f Formula 5.2.0.1). For example, when we trained on MIMIC-III and HiRID, we obtained shared coefficients and group-specific coefficients for MIMIC-III and HiRID. When predicting the response variable on eICU, we only used the shared coefficients[5].

5.3.2 Model specific preprocessing

We preprocessed each group individually before integrated into the final DSL dataset[6]. For each dataset, we first imputed missing dynamic values using the mean, then centered and scaled them to unit variance. For each dataset, the static variable "sex" is one-hot encoded. We dropped all observations where no sex was observed. This resulted in the removal of two observations from the eICU dataset and none from the other datasets.

Afterwards, the eICU dataset contained 74585 patient visits from 188 different hospitals, the MIMIC-III dataset contained 30335 patient visits, the MIMIC-IV dataset contained 35673 patient visits and the HiRID dataset contained 8577 patient visits.

Example: Suppose we want to use eICU, HiRID and MIMIC-III for training. The augmented design matrix is then:

$$Z = \begin{pmatrix} X_{eicu} & r_{eicu}X_{eicu} & 0 & 0 \\ X_{hirid} & 0 & r_{hirid}X_{hirid} & 0 \\ X_{mimic} & 0 & 0 & r_{mimic}X_{mimic} \end{pmatrix},$$

where $r_{eicu} = 1/\sqrt{74858/(74858 + 30335 + 8577)} \approx 1.23$. We calculate r_{hirid} and r_{mimic} analogously. Here, X_{eicu} represents the preprocessed dataset of eICU, etc.

5.3.3 Results and Discussion

Table 5.1 presents the MSE of predicting the response variable on the "Target" using the shared coefficients estimated using DSL on the "Train Data".

There is no dataset on which DSL consistently performed best. We observed its worst performance on MIMIC-III. The MSE range between best and worst performance is over 100. In each target dataset the MSE typically fluctuated within a range of 30. Whenever MIMIC-III was paired with HiRID, it resulted in the worst performance on the target dataset, possibly due to imbalanced sizes of the datasets. The larger datasets MIMIC-IV and eICU were able to mitigated this. DSL was incapable to capture common effects in MIMIC-III when trained on other datasets. We achieved best performance using a combination of eICU, HiRID and MIMIC-IV for both training and evaluation. This suggest that DSL was able to identify common effects on datasets showing similar distributional characteristics (c.f Figure 3.1). On the other hand, the shared effects identified by DSL did not generalize well to unseen distributions. Overall, DSL's dependency on dataset-specific knowledge indicates no robustness and impracticality for hospital applications.

[5]Strictly speaking, we also used the intercept.
[6]That is before creating the augmented dataset.

Target	Train Data	MSE
eicu	mimic, hirid	171.69
eicu	mimic, miiv	157.14
eicu	hirid, miiv	*149.36*
eicu	mimic, hirid, miiv	155.27
hirid	mimic, eicu	164.78
hirid	mimic, miiv	174.45
hirid	eicu, miiv	*159.74*
hirid	mimic, eicu, miiv	163.56
miiv	mimic, hirid	161.24
miiv	mimic, eicu	140.85
miiv	hirid, eicu	*138.15*
miiv	mimic, hirid, eicu	140.35
mimic	hirid, eicu	258.83
mimic	hirid, miiv	268.72
mimic	eicu, miiv	261.44
mimic	hirid, eicu, miiv	261.87

Table 5.1: MSE between Predicted and Target Data Sets

5.3.4 Comparison to Magging

In this Section, we compare the results obtained from DSL with those from Magging using linear methods (c.f. Table 4.2). We observe that Magging outperformed DSL in 9 out of 12 cases, with DSL only marginally surpassing Magging in the remaining cases[7]. On MIMIC-III, DSL performed significantly worse than Magging and was not able to achive similar consistent results on the targets. Therefore, Magging seemed more suited for hospital applications.

5.4 DSL and Magging applied on artificial data

In this Section, we compare the regression coefficient estimate of Data Shared Lasso and Magging on artificial data. For simplicity, we set the degree of sharing $r_g = r$ uniformly across groups.

We consider four groups, each consisting of 50 observations with two independent features. We organize the features in the design matrix $X \in \mathbb{R}^{200\times 8}$. Each observation is a realization from a standard normal random vector. We assume a linear relationship between our response variable $Y \in \mathbb{R}^{200}$ and X. The true underlying relationship in group g is modeled by $\Theta_g^* \in \mathbb{R}^2$, a realization of a two-dimensional normal distribution with mean $(50, 4)^T$ and standard deviation 1.

$$Y = X\Theta^* + \varepsilon,$$

with $\varepsilon \in \mathbb{R}^{200}$ a standard normal random vector and $\Theta^* = (\Theta_1^*, \ldots, \Theta_4^*)^T \in \mathbb{R}^8$.

We estimated the regression coefficients in the Magging ensemble using Lasso Regression.

[7] Highlighted in Table 5.1.

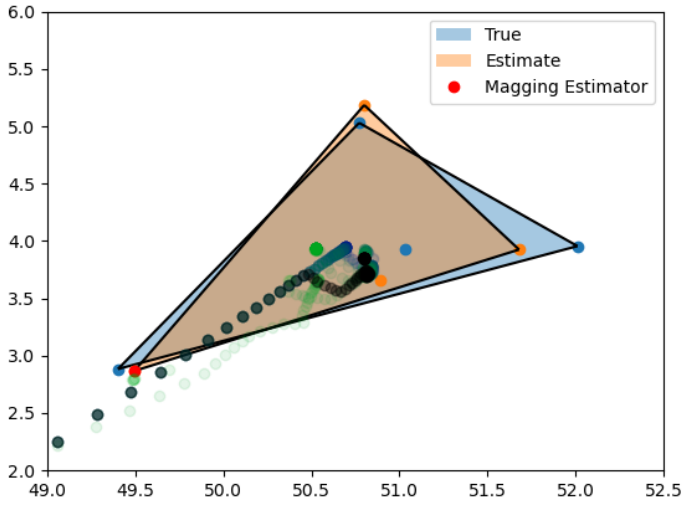


Figure 5.1: DSL and Magging applied on artificial data.

We selected the regularization strength for each group by minimizing the 5-fold CV error within the respective group. The Magging weights and the Magging estimator is calculated as in Section 4.3. We allow Lasso to choose from 75 values as regularization strength between 0.001 and 5. We calculated the shared coefficient estimate for DSL using varying degree of sharing and regularization strength. We allowed ten different values as degree of sharing between 0.001 and 5 and the same regularization strengths as for Lasso.

Figure 5.1 displays the results. Blue points[8] and the blue area represent the true coefficients Θ_g^* and their convex hull. Orange points denote Lasso coefficient estimates $\widehat{\Theta}_g$, while the red point represents the Magging regression coefficient estimate. The individual points that tend to 0 are the DSL shard effect estimates. Points having the same colour have the same r but varying regularization strength. Light green represents the shared effects estimate using the smallest r value, while dark blue represents the largest r value. The black points are obtained when selecting $r = 1/\sqrt{G}$.

We observe that for weak regularization and large sharing, i.e. small values of r, the shared effects estimated by DSL were contained in the convex hull estimated from Magging and tended to zero as both increased. This behavior of DSL converging to the origin is consistent with Section 5.2. That is, as we pool less, we expect common effects to approach zero as groups are estimated separately.

5.5 Data Identification

Gross and Tibshirani (2016) observed that DSL provided clearer insights into the similarities and differences between groups compared to pooled Lasso[9] or running individual Lasso's on each group. In this Section, we investigate whether DSL can help us to understand similarities and differences across datasets in an ICU setting.

[8] One blue point lies within the convex hull, somewhat separated from all other points in the Figure.
[9] All groups are treated as one big dataset originating from the same distribution.

To understand similarities and differences, we visualize the ten coefficients with the largest absolute values obtained from DSL, a version of pooled Lasso[10] and running separate Lasso's, using bar plots. In these bar plots, red indicates negative coefficients and blue indicates positive ones. We compare these coefficients with feature importances obtained from Random Forest (c.f. Section 3.3.3). Additionally, for the Lasso methods, we present the Lasso profiles[11] of their coefficients.

5.5.1 Experiment setting

In this Section, similar to Section 5.3, we used DSL, (pooled and separate) Lasso and (pooled[10] and separate) Random Forest to predict the heart rate on day three[12]. We set the degree of sharing in DSL following the approach outlined in Section 5.3. For the pooled Random Forest we used all groups whereas when running separate Random Forest, we only used one group at a time. To optimize the Lasso models, we selected the best hyperparameters as described in Section 5.3. We used 80% feature fraction for the Random Forest. Observations lacking data on "sex" were excluded across all methods, resulting in datasets identical to those described in Section 5.3.

We implemented Lasso Regression using Scikit-learn version 1.3.2 and used LightGBM version 4.1.0 for the Random Forest.

5.5.2 Model specific preprocessing

For DSL and pooled Lasso, we proceed as outlined in Section 5.3. For separate Lasso's, we follow similar steps, excluding the use of the augmented design matrix. Additionally, we preprocess data for the Random Forest according to Section 3.4.2

5.5.3 Results and Discussion

Figure 5.2 presents bar plots for the coefficients obtained from Data Shared Lasso. Figure 5.3 illustrates the coefficients from pooled and separate Lasso. Figure 5.4 displays the feature importance from pooled and separate Random Forest.

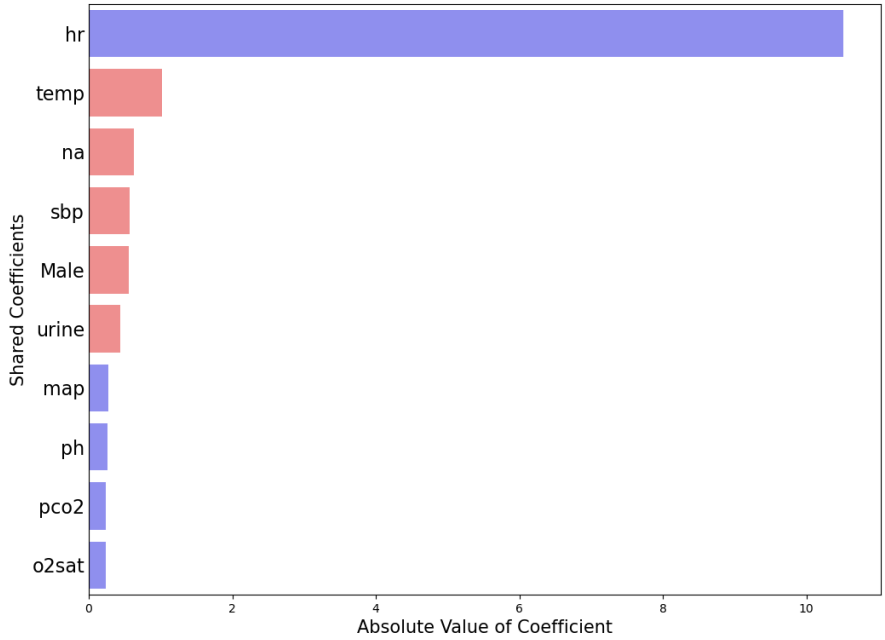
As depicted in Figure 5.3b and Figure 5.4b, the dominance of the "heart rate" coefficient in the bar plots makes it challenging to discern dataset differences[13]. However, Figure 5.2b allows for clear identification of dataset discrepancies. For instance, the importance of heart rate in MIMIC-III indicates a distributional shift compared to other datasets (c.f. Figure 3.1). The variation in the magnitude of the absolute values further indicates that MIMIC-III differs from the other datasets. Understanding feature importance and dataset differences was more challenging when examining Pooled Lasso and individual Lasso fits because a substantial part of Figure 5.3 focused on highlighting the importance of heart rate making it difficult to identify other important features and differences. Thus, by segregating shared and separate coefficients, model interpretation becomes more straightforward. Shared coefficients hold consistent meanings across datasets, while separate coefficients hold dataset-specific importance.

[10] Pooled Lasso but we prepocess each dataset separately before combing to one dataset.

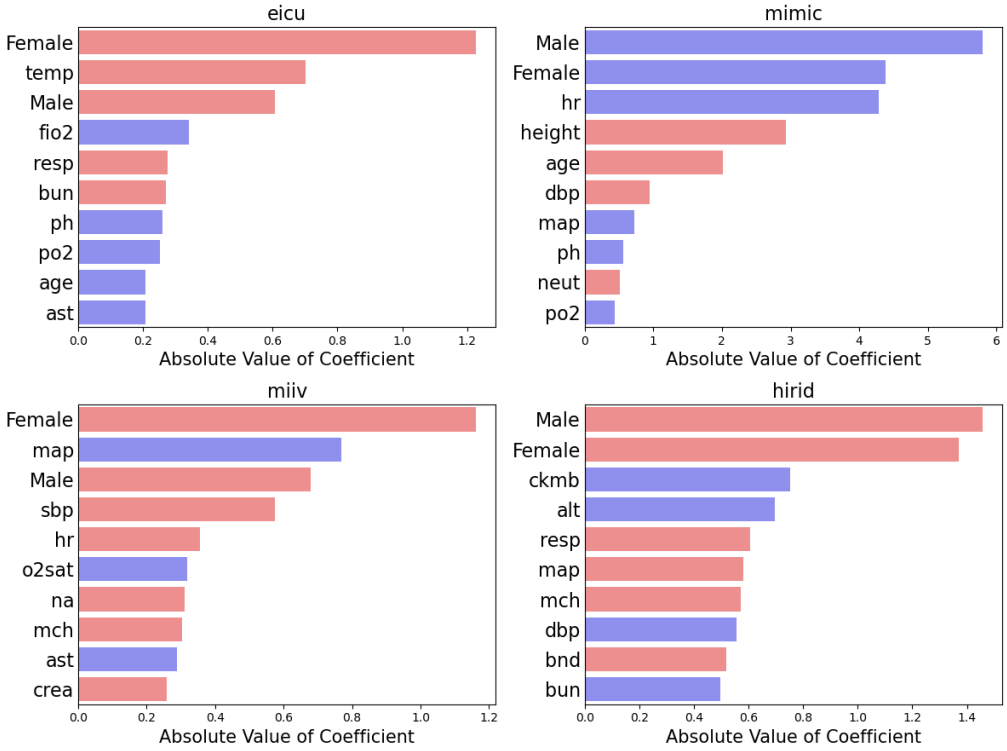
[11] Lasso paths illustrate the trajectory of coefficient values as a function of the regularization parameter. They are used to visualize the effect of regularization on different features, thus providing insight into variable selection.

[12] For DSL, pooled Lasso and pooled RF: We trained on all datasets.

[13] The clinical concept names are explained in Chapter 2.1.

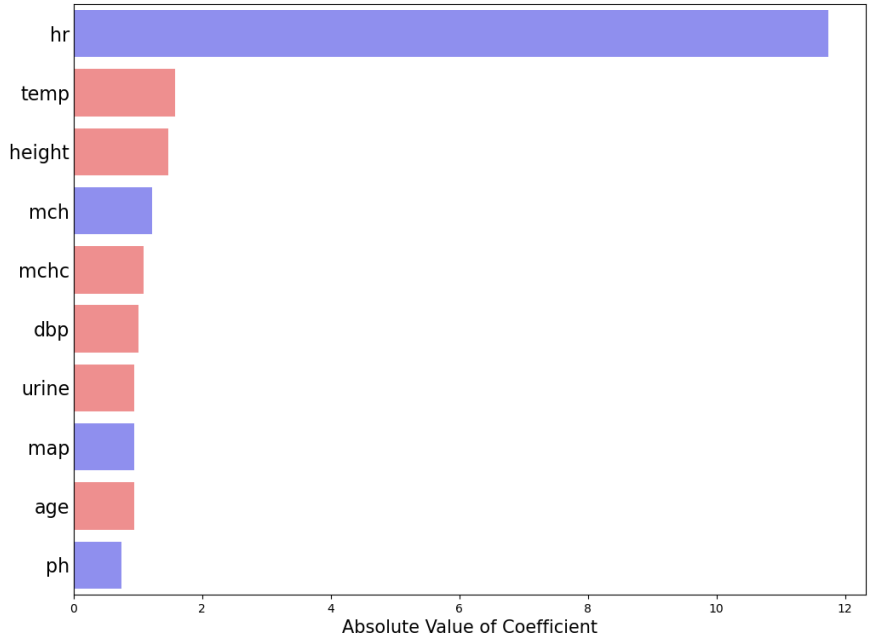


(a) Shared Coefficients.

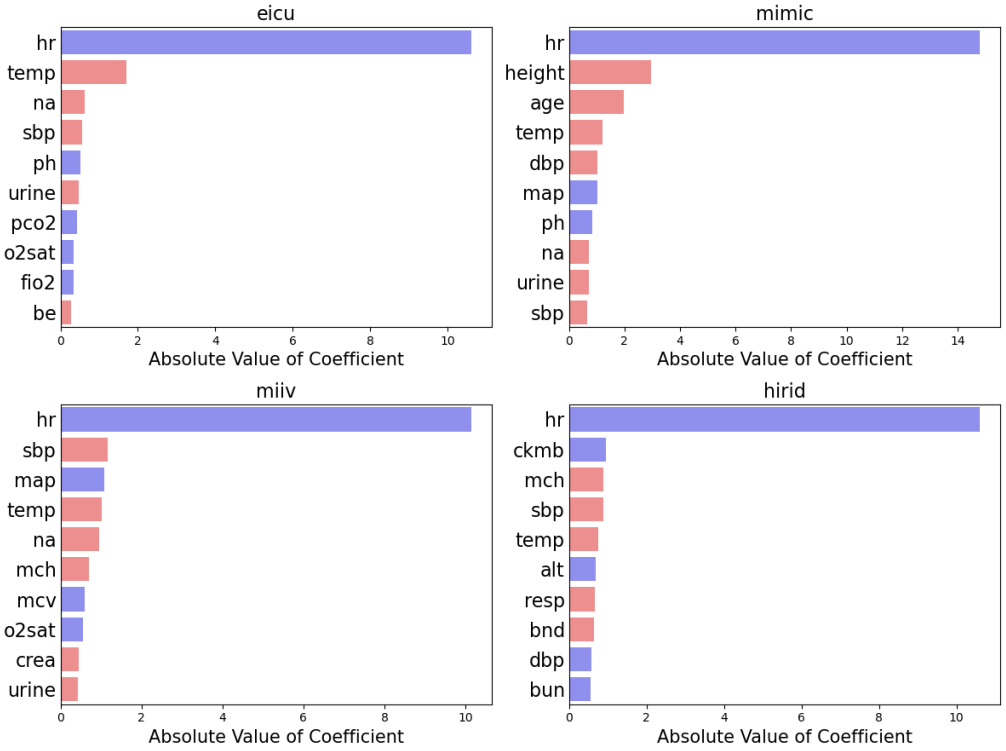


(b) Group Specific Coefficients.

Figure 5.2: Data Shared Lasso Coefficients Plots.

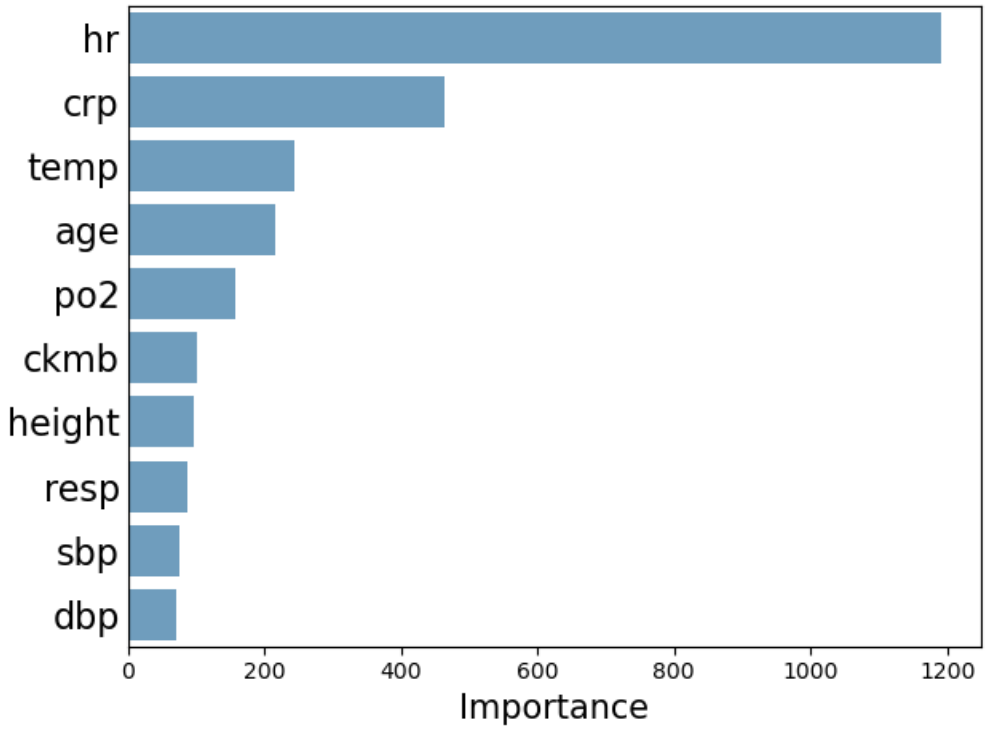


(a) Pooled Lasso Coefficients.

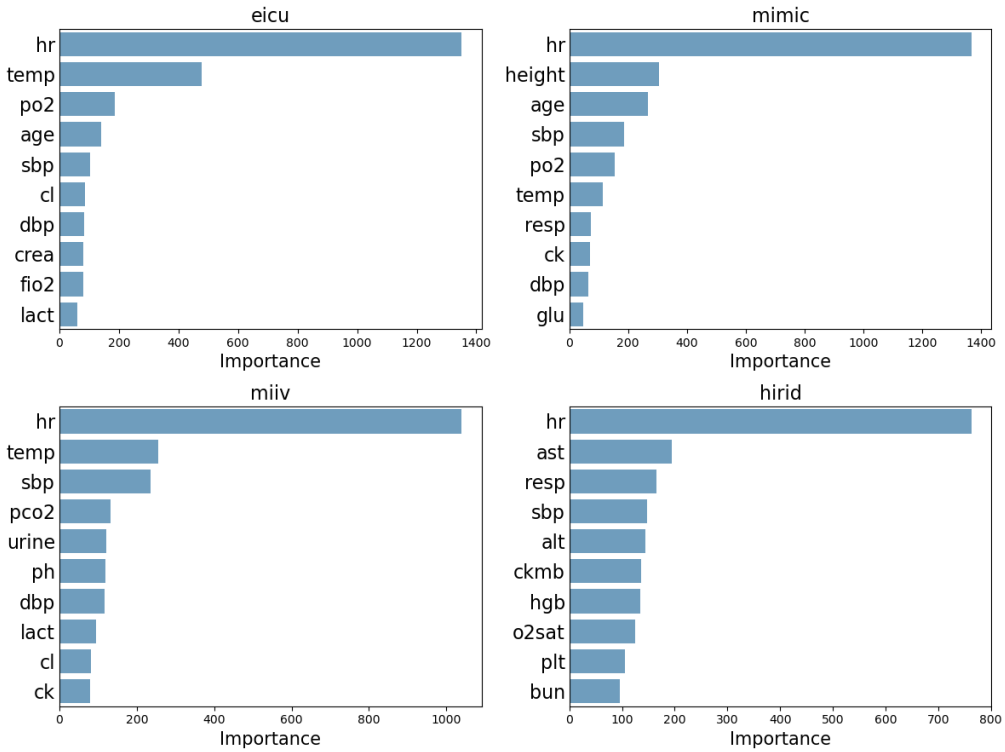


(b) Separate Lasso Models Coefficients.

Figure 5.3: Coefficients from running a separate lasso on each dataset and running pooled lasso.

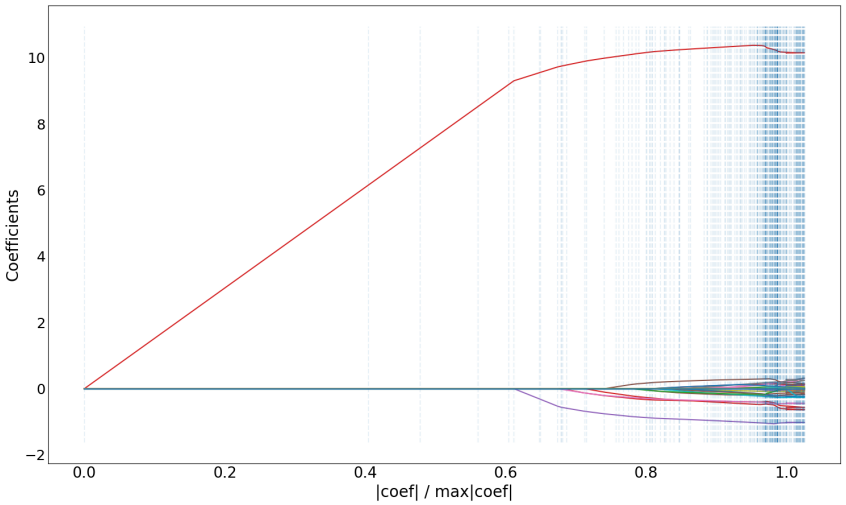


(a) Trained on entire Dataset.

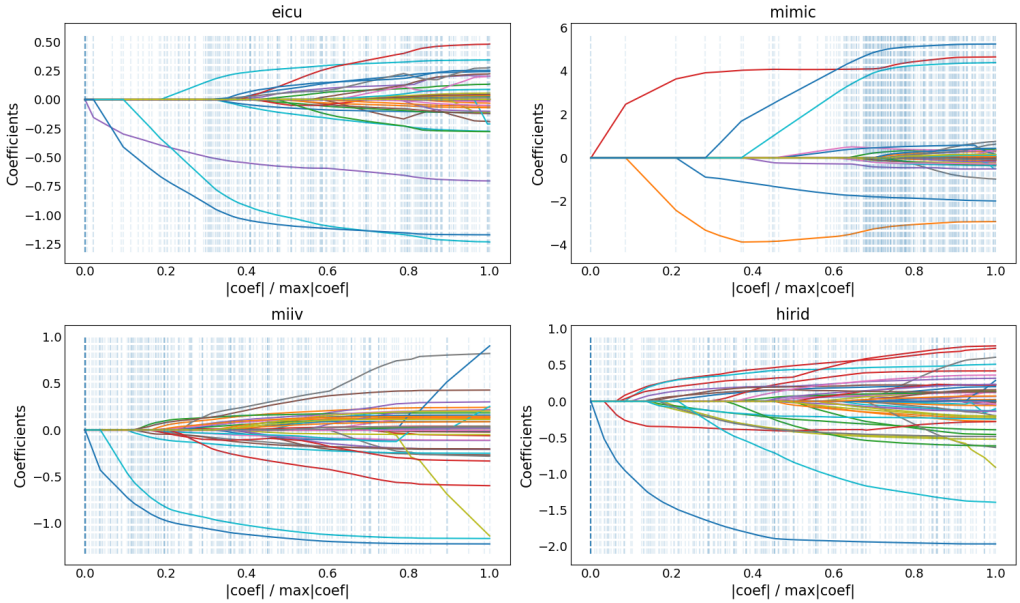


(b) Feature Importances Trained on each dataset.

Figure 5.4: Random Forest Feature Importances.

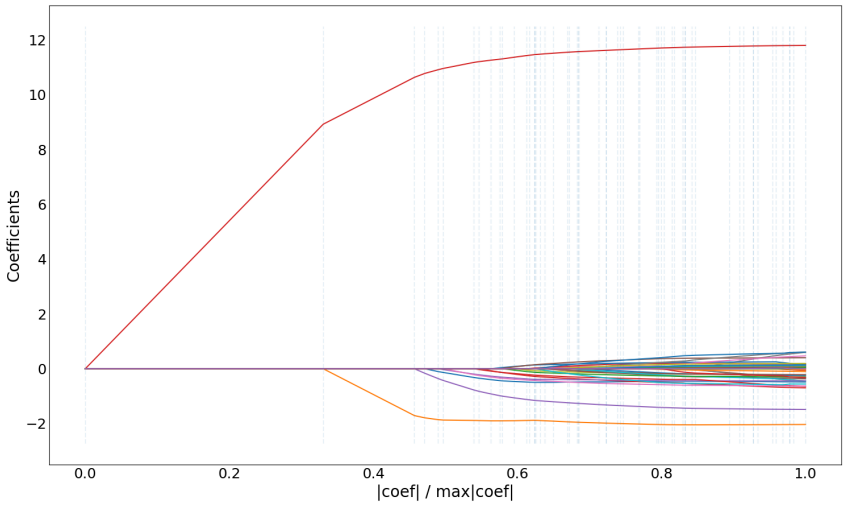


(a) Shared Effects.

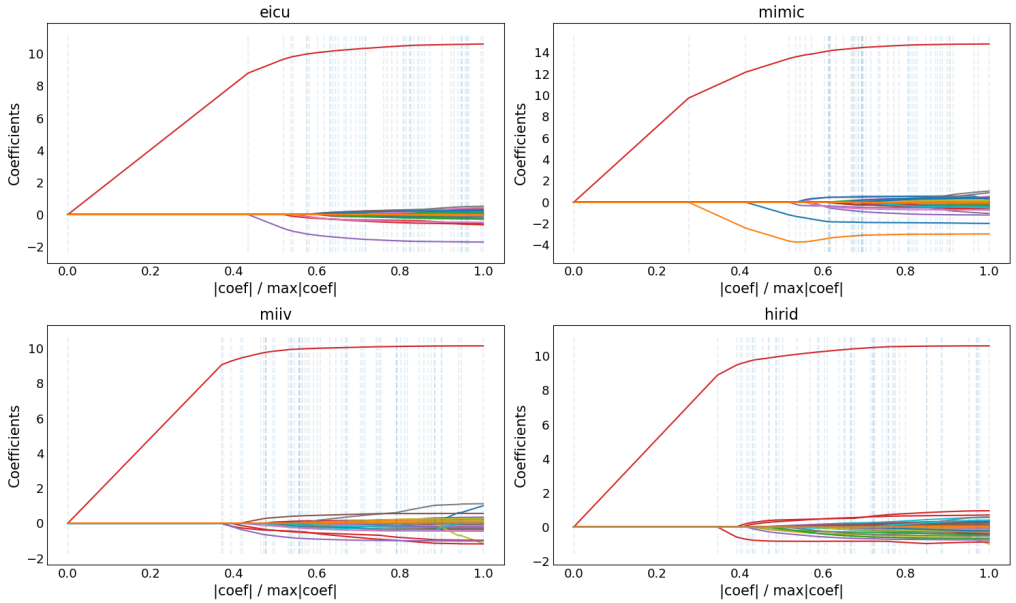


(b) Group Individual Effects.

Figure 5.5: Data Shared Lasso profiles.



(a) Pooled Lasso.



(b) Group Individual Profiles.

Figure 5.6: Pooled and Individual Lasso profiles.

Type	Concepts
Shared effects	spb, na, temp, hr
eICU	hr, temp, Male, Female
MIMIC-III	height, Female, hr, Male
MIMIC-IV	mcv, mch, Female, Male
HiRID	ckbm, mch, Female, Male

Table 5.2: Displaying the four largest features in Lasso paths (Figure 5.5), sorted by absolute magnitude.

Type	Concepts
Pooled	na, temp, height, hr
eICU	spb, na, temp, hr
MIMIC-III	temp, age, height, hr
MIMIC-IV	map, mch, sbp, hr
HiRID	sbp, ckmb, mch, hr

Table 5.3: Displaying the four largest features in Lasso paths (Figure 5.6), sorted by absolute magnitude.

Similar patterns were observable in their profiles. Figure 5.5 illustrates the Lasso coefficients as the standardized tuning parameter $|coef|/max|coef|$ is varied. We obtain the least squares estimates[14] when $|coef|/max|coef| = 1$. The dashed lines indicate positions along the x-axis where regularization values change and correspond to different levels of regularization. The largest features[15] are displayed in Tables 5.2 and 5.3. Even when shared coefficients are zero except for heart rate, internal group effects persist. The slight increase in the heart rate coefficient at the beginning appears to be more of a numerical issue.

Similar to the findings of Gross and Tibshirani (2016), traditional methods emphasized shared coefficients across all groups, masking group-specific differences. DSL was able to improve dataset identifiability by pinpointing coefficients with unique meaning for each group.

We repeated the experiment predicting the mean arterial pressure in Appendix B.4 and observed similar behavior.

[14] Appendix B.2.1.
[15] In absolute magnitude.

Chapter 6

DSL: Stepwise regression

In the previous Chapters, we explored techniques designed to handle multiple datasets at once without assuming homogeneity across datasets. We successfully applied Data Shared Lasso (DSL) in the last Chapter, helping us to understand the structural differences among datasets. In this Chapter, our focus shifts towards stepwise regression[1] and its robustness[2] using Data Shared Lasso as regression model.

6.1 Motivation

In traditional stepwise regression, the model's performance does not decrease when new variables are added. However, it is unclear if this behavior remains consistent when evaluating the model's performance on data that may differ from the training data.

As discussed in Chapter 5, Gross and Tibshirani (2016) propose DSL as a method for capturing shared effects among datasets. In this context, our goal is to assess how well DSL performs on new data when applying stepwise regression. In particular, we aim to understand feature influence across datasets and whether DSL has mechanisms to regulate this, leading to robust stepwise regression.

6.2 Application to ICU data

6.2.1 Experiment setting

As in Section 3.4.1, we predicted the average heart-rate on day three using average values of the dynamic features at day 1 and static features. We removed patient visits if there was no measurement of the heart rate in the time windows 0-24h or 48-72h. The degree of sharing r_g was controlled as in Section 5.3.

We sequentially added unused features to our model and selected the feature which reduced the MSE[3] the most. We exclusively added shared effects to our model and ignored any group-specific features[4]. This process was repeated for various levels of regularization. The regularization parameter was constrained to values within the range of 0.01 to 2.

[1] Sequentially adding the feature reducing the MSE on the train data the most.
[2] See Chapter 1.
[3] On the train data.
[4] Strictly speaking, we also included the intercept.

We restricted ourselves to these values due to the presence of a dominant feature (c.f Section 5.5) leading to strong regularization effects[5] even for relatively small values of the regularization.

We compared the performance of this approach with stepwise regression applied on Lasso (c.f. Section 3.3.2), called baseline. We selected the regularization parameter in Lasso before applying stepwise regression using 5-fold CV[6], minimizing the MSE in the train data using all features. We constrained the regularization parameter for Lasso to the same values as for DSL.

For both, Data Shared Lasso and Lasso, we used the implementation of Lasso from SciKit-learn version 1.3.2 (c.f. Section 5.2).

6.2.2 Model specific preprocessing

For Data Shared Lasso, we employed the preprocessing described in Section 5. Therefore, the resulting datasets were identical to those in Section 5. For Lasso, we employed the preprocessing described in Section 3.4.2[7] for each dataset separately. Afterwards we combined those datasets and employed Lasso on this merged dataset.

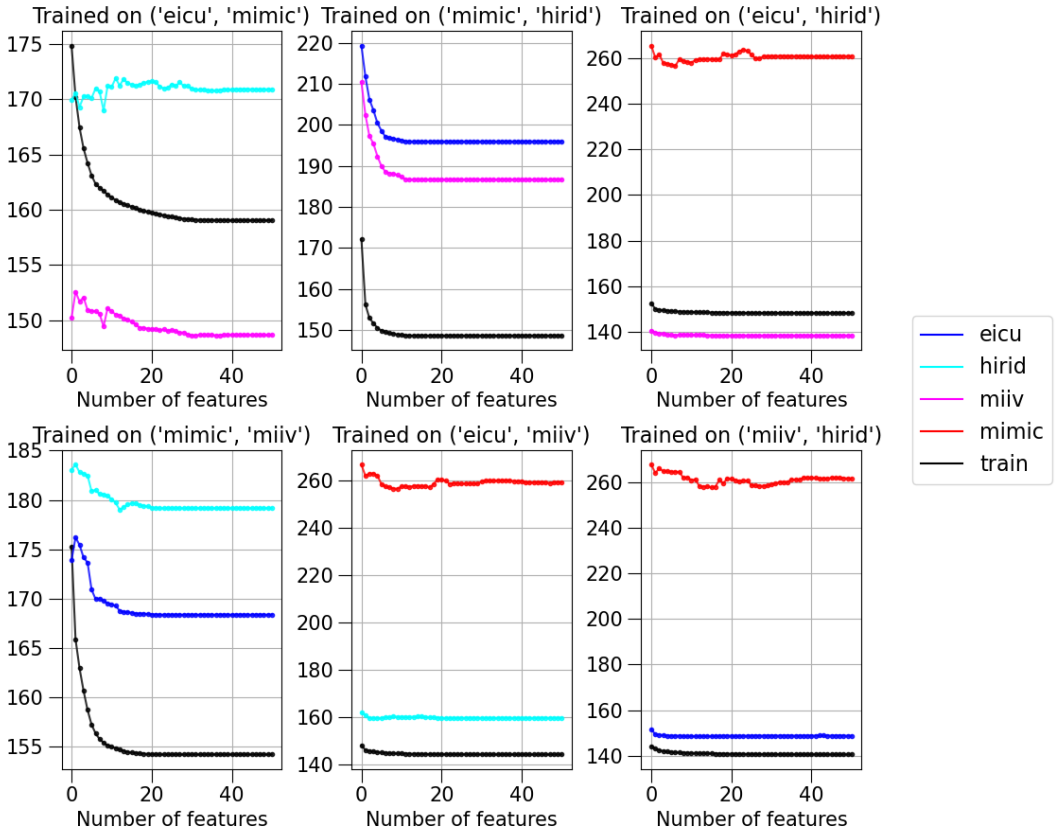
6.2.3 Results and Discussion

Figure 6.1 illustrates the results of the baseline. As features ware added to the baseline model, its performance on the training data improved. However, the baseline lacked robustness, e.g. when evaluated on HiRID and trained on eICU and MIMIC-II. Moreover, training on data that appeared to capture a significant portion of the variability among datasets, like the combination of eICU, MIMIC-III, and MIMIC-IV (see Figure 3.1), did not guarantee improved performance. This can be seen when trained on eICU, MIMIC-III and MIMIC-IV and evaluated on HiRID.

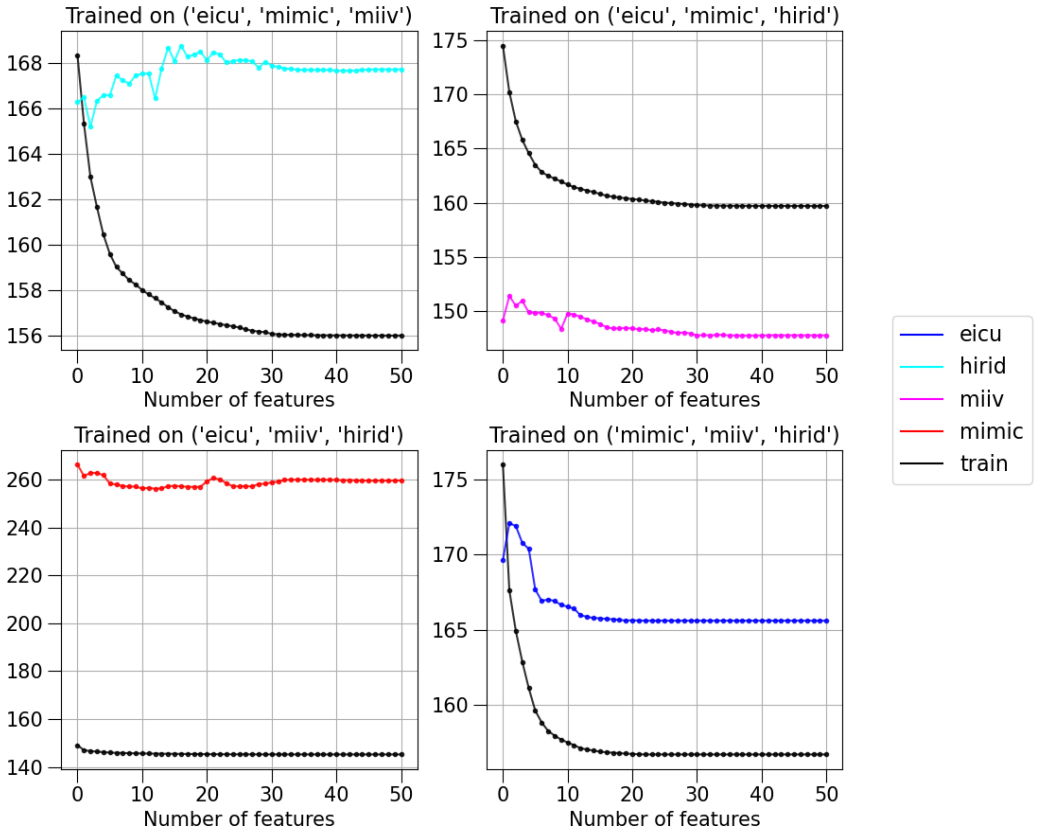
Figure 6.2 displays the performance of Data Shared Lasso. Data Shared Lasso demonstrated superior performance and robustness compared to Lasso when regularization was carefully chosen. For example, evaluated on HiRID and trained on eICU and MIMIC-III outperformed the baseline with respect to robustness and MSE when stronger regularization was chosen. In scenarios where the dataset with the strongest distributional differences, MIMIC-III, was included in the training, strong regularization tended to enhance robustness and yielded smaller MSE. However, there was one exception to this trend: When trained on eICU, MIMIC-III, and MIMIC-IV, weaker regularization slightly outperformed stronger regularization. When evaluated on MIMIC-III, weaker regularization, although less stable, outperformed stronger regularization in terms of MSE but failed to surpass the baseline. In fact, when evaluated on MIMIC-III, DSL's optimal estimate exhibited identical behavior to the baseline.

In conclusion, stepwise regression applied to Lasso lacked robustness, whereas stepwise regression applied to DSL was robust with sufficiently strong regularization. However, this robustness was associated with higher MSE when evaluated on MIMIC-III[8]. Conversely, when MIMIC-III was included in the training data, stronger regularization outperformed weaker regularization or underperformed it slightly.

[5] Most features are set to zero.
[6] See Section 3.2.
[7] For Ridge regression.
[8] The dataset with distributional differences.

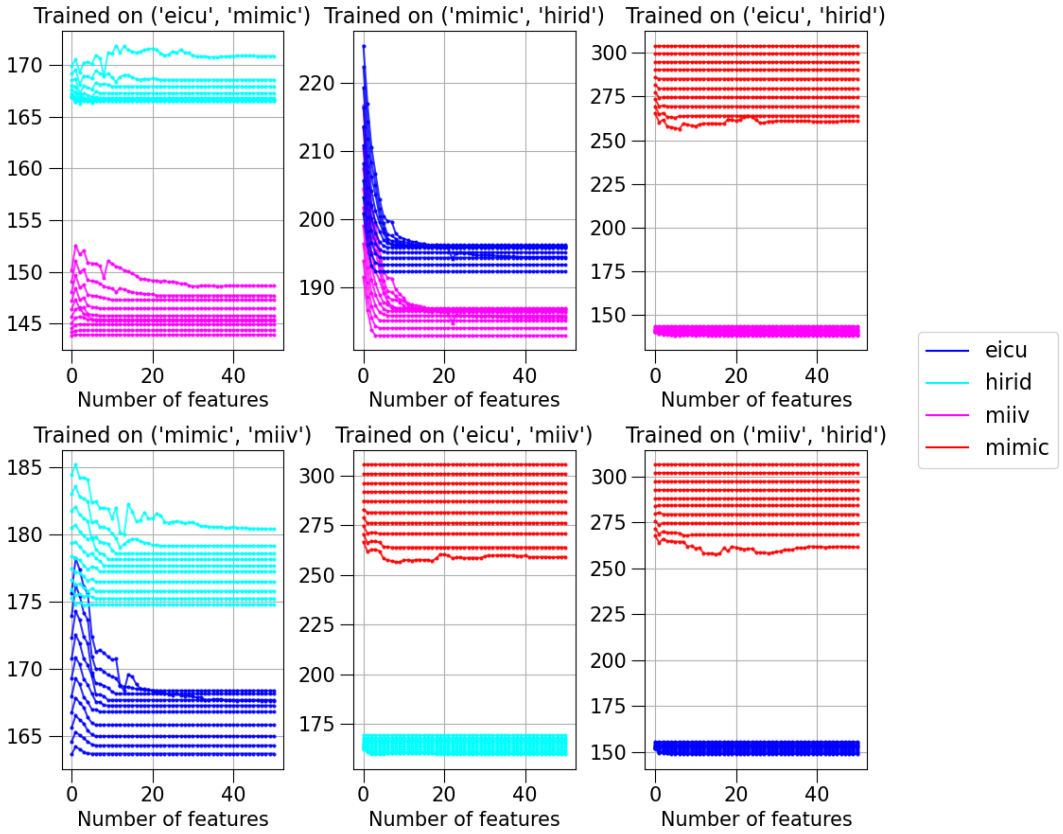


(a) MSE of the baseline vs. number of features. The baseline was trained on two datasets.

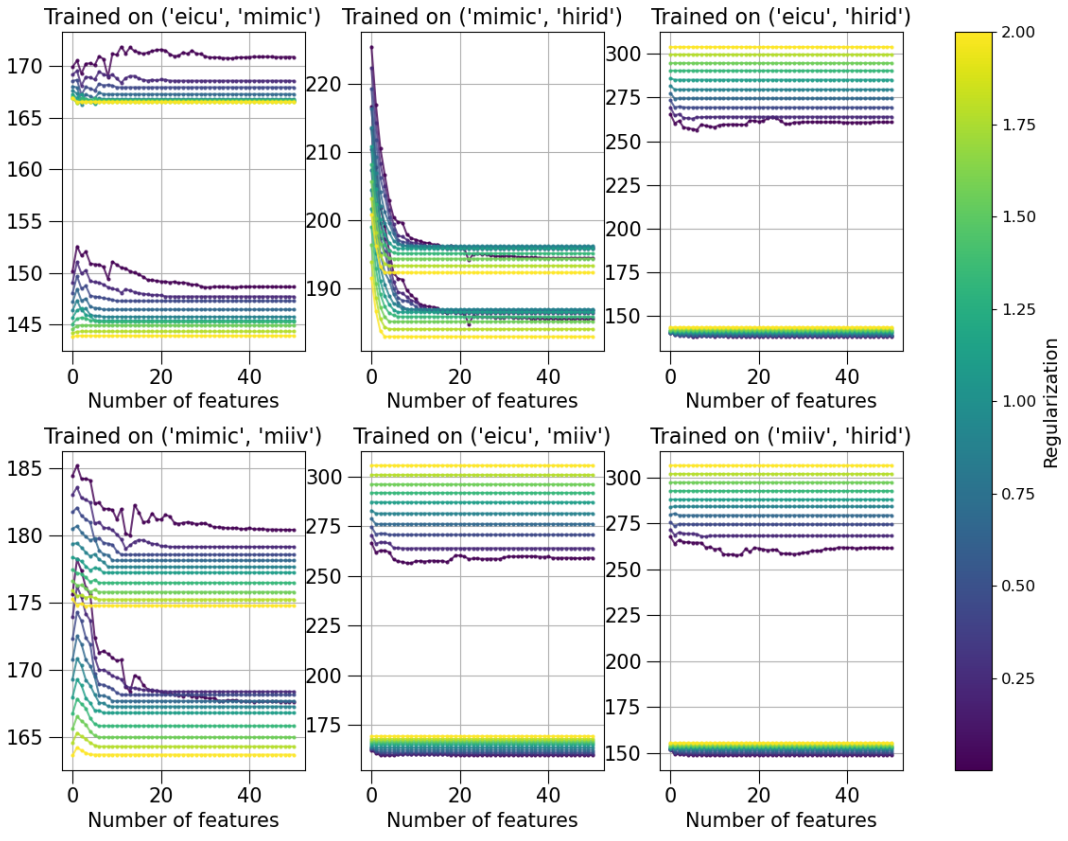


(b) MSE of the baseline vs. number of features. The baseline was trained on three datasets.

Figure 6.1: Forward Selection using Lasso as baseline.

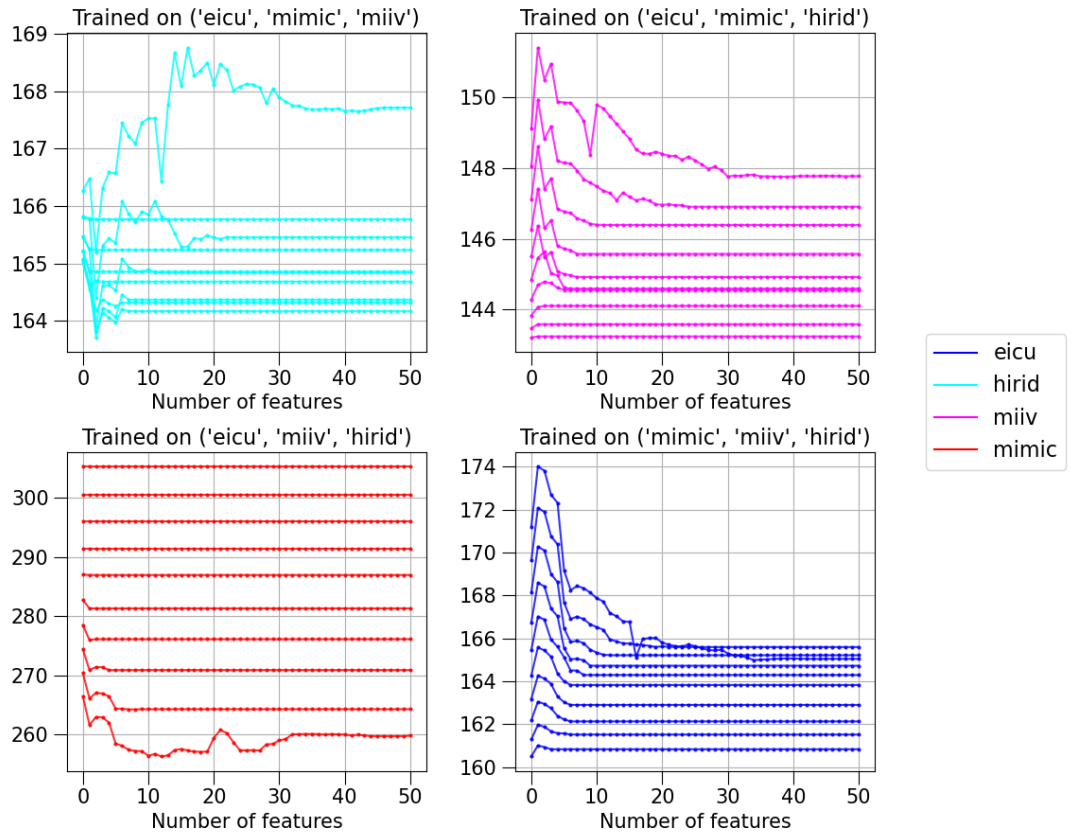


(a) MSE of DSL vs. number of features using different regularization strengths. DSL was trained on two datasets.

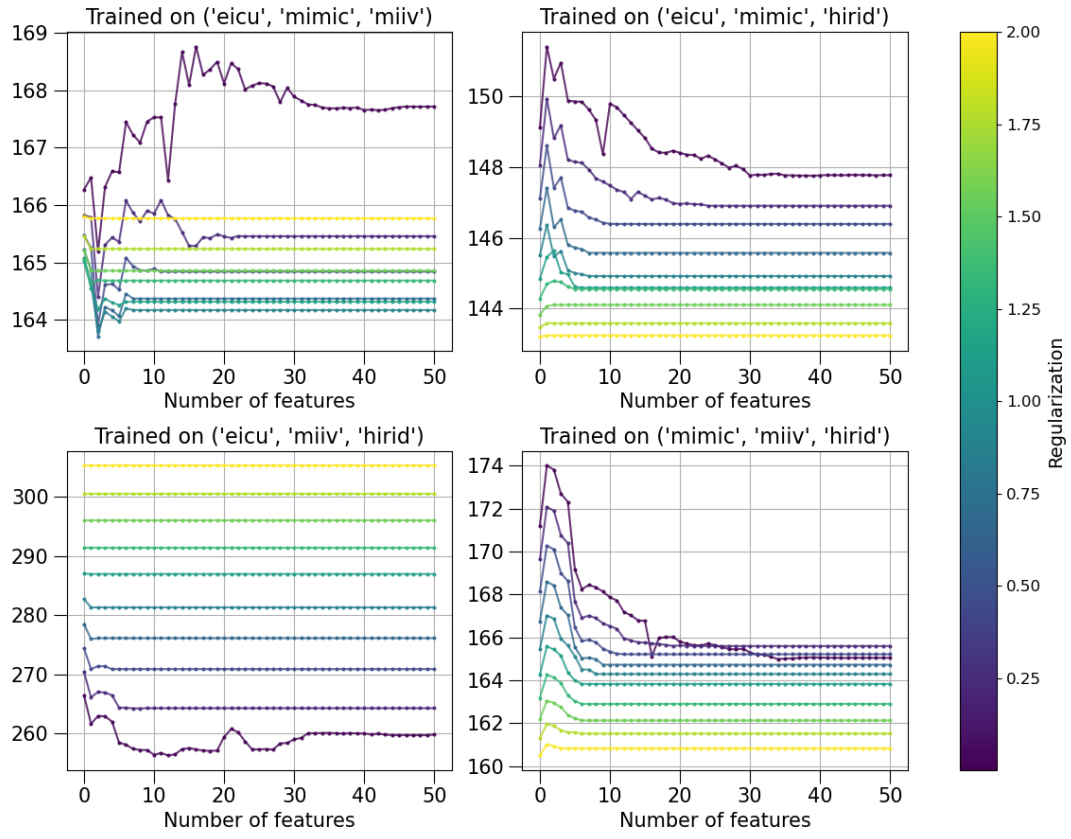


(b) Color map for different regularization strengths.

Figure 6.2: Forward Selection using DSL for various regularization strengths.



(a) MSE of DSL vs. number of features using different regularization strengths. DSL was trained on three datasets.



(b) Color map for different regularization strengths.

Figure 6.3: Forward Selection using DSL for various regularization strengths.

Chapter 7

Conclusion

In this thesis, we investigated how reliable predictors and models can be transferred from one hospital/dataset to another.

In Chapter 3, using the fine-tuning approach, we found that Anchor Regression, despite its monotonicity in MSE, could not outperform Ridge Regression. However, Anchor Boost and RefitLGBM displayed improvements, especially when applied on datasets showing distributional differences (e.g. MIMIC-III, c.f. Figure 3.2b). This indicates that even a small fine-tuning dataset can help capturing structural differences of between hospitals.

Magging demonstrated its effectiveness in capturing common effects across datasets when using Lasso to estimate ensemble members (see Table 4.2), indicating robustness. However, when applying non-linear methods like Random Forest to estimate ensemble members, Magging displayed no robustness (see Table 4.3). On the contrary, Data Shared Lasso (DSL) failed to capture common effects and proved ineffective in this regard (see Table 5.3). This indicated a discrepancy with its fundamental assumption that true regression coefficients vary only to some extent between groups. However, DSL proved effective in identifying structural differences between groups (see Section 5.5). In Chapter 6, we observed that stronger regularization lead to robust stepwise regression.

Neither Magging nor DSL were able to outperform the refitted Random Forest or Anchor Boost[1] on MIMIC-III. This underscores the conclusion in the second paragraph, that even a small fine-tuning dataset can help identify structural disparities of a hospital. Moreover, these results suggest that using the entire eICU dataset for training collectively is akin to treating eICU as a single large hospital. Hence, these results suggests that the generalizability and robustness of training improves as the number of available hospitals increases. Therefore, researchers developing ICU prediction models should dedicate significant effort to curating a training set that accurately reflects the anticipated future deployment hospitals.

7.1 Future Work

The limited number of data sources might have hindered some methods ability to detect consistent patterns. Access to more data sources could potentially lead to improved performance for some methods. However, findings from the eICU data suggest that the

[1] With a sufficiently large number of fine-tuning data points.

generalizability of standard training also gets better with an increase in the number of hospitals. This raises the question of whether there exists an optimal number of hospitals where specialized methods offer the greatest advantage.

Our results indicate that as more ICU data sources become publicly available for training, models derived from them may become more universally applicable. Increasing the representation of hospitals in the training data enhances the likelihood that there will be similar hospitals in the future, facilitating reliable knowledge transfer.

It would also be interesting to observe the performance of these methods in datasets without a dominant feature. Specifically, it would be interesting to determine if Magging is able to outperform the refitted Random Forest, using a fine-tuning dataset, when trained on a larger number of hospitals (which could not be explored due to resource limitations), or when applying the fine-tuning approach on methods aimed to capture common effects like Magging or DSL.

Bibliography

Bennett, N., D. Plečko, I.-F. Ukor, N. Meinshausen, and P. Bühlmann (2023, 06). ricu: R's interface to intensive care data. *GigaScience 12*, giad041.

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

Bühlmann, P. and N. Meinshausen (2016). Magging: Maximin aggregation for inhomogeneous large-scale data. *Proceedings of the IEEE 104*(1), 126–135.

Faltys, M., M. Zimmermann, X. Lyu, M. Hüser, S. Hyland, G. Rätsch, and T. Merz (2021). HIRID, a high time-resolution icu dataset (version 1.1.1). PhysioNet.

Friedman, J. H. (2001). Greedy function approximation: a gradient boosting machine. *Annals of statistics*, 1189–1232.

Goldberger, A. L., L. A. N. Amaral, L. Glass, J. M. Hausdorff, P. C. Ivanov, R. G. Mark, J. E. Mietus, G. B. Moody, C.-K. Peng, and H. E. Stanley (2000, June 13). Physiobank, physiotoolkit, and physionet: Components of a new research resource for complex physiologic signals. *Circulation 101*(23), e215–e220. Circulation Electronic Pages.

Gross, S. M. and R. Tibshirani (2016). Data shared lasso: A novel tool to discover uplift. *Computational Statistics & Data Analysis 101*, 226–235.

Gulrajani, I. and D. Lopez-Paz (2020). In search of lost domain generalization.

Hastie, T., R. Tibshirani, and J. Friedman (2009). *The Elements of Statistical Learning: Data Mining, Inference, and Prediction* (2 ed.). New York: Springer Series in Statistics.

Johnson, A., L. Bulgarelli, T. Pollard, S. Horng, L. A. Celi, and R. G. Mark (2023). MIMIC-IV (version 2.2). PhysioNet.

Johnson, A., T. J. Pollard, and R. G. Mark (2016). MIMIC-III clinical database (version 1.4). PhysioNet.

Pollard, T. J., A. E. Johnson, J. D. Raffa, L. A. Celi, O. Badawi, and R. G. Mark (2019). eICU collaborative research database (version 2.0). *PhysioNet*.

Reyna, M. A. et al. (2020). Early prediction of sepsis from clinical data: The physionet/-computing in cardiology challenge 2019. *Critical care medicine 48*(2), 210–217.

Rockenschaub, P., A. Hilbert, T. Kossen, F. von Dincklage, V. I. Madai, and D. Frey (2023). From single-hospital to multi-centre applications: Enhancing the generalisability of deep learning models for adverse event prediction in the icu.

Rothenhäusler, D., N. Meinshausen, P. Bühlmann, and J. Peters (2020). Anchor regression: heterogeneous data meets causality.

Sauer, C. M. et al. (2022). Systematic review and comparison of publicly available ICU data sets—a decision guide for clinicians and data scientists. *Critical Care Medicine 50*(6), e581–e588.

Tibshirani, R. (2011). Regression shrinkage and selection via the lasso: A retrospective. *Journal of the Royal Statistical Society: Series B (Statistical Methodology) 73*(3), 273–282.

Wynants, L., D. M. Kent, D. Timmerman, C. M. Lundquist, and B. Van Calster (2019). Untapped potential of multicenter studies: A review of cardiovascular risk prediction models revealed inappropriate analyses and wide variation in reporting. *Diagnostic and Prognostic Research 3*(1), 6.

Appendix A

Reproducibility

A.1 Reproduce Results

For reproducibility of the whole computations, we refer to our codebase at:

https://github.com/lbrilh/masterthesis.

In order to reproduce our computations and results, set up the directory as described in the README and execute the Python and R files by hand.

Appendix B

Further Material

B.1 Available Data

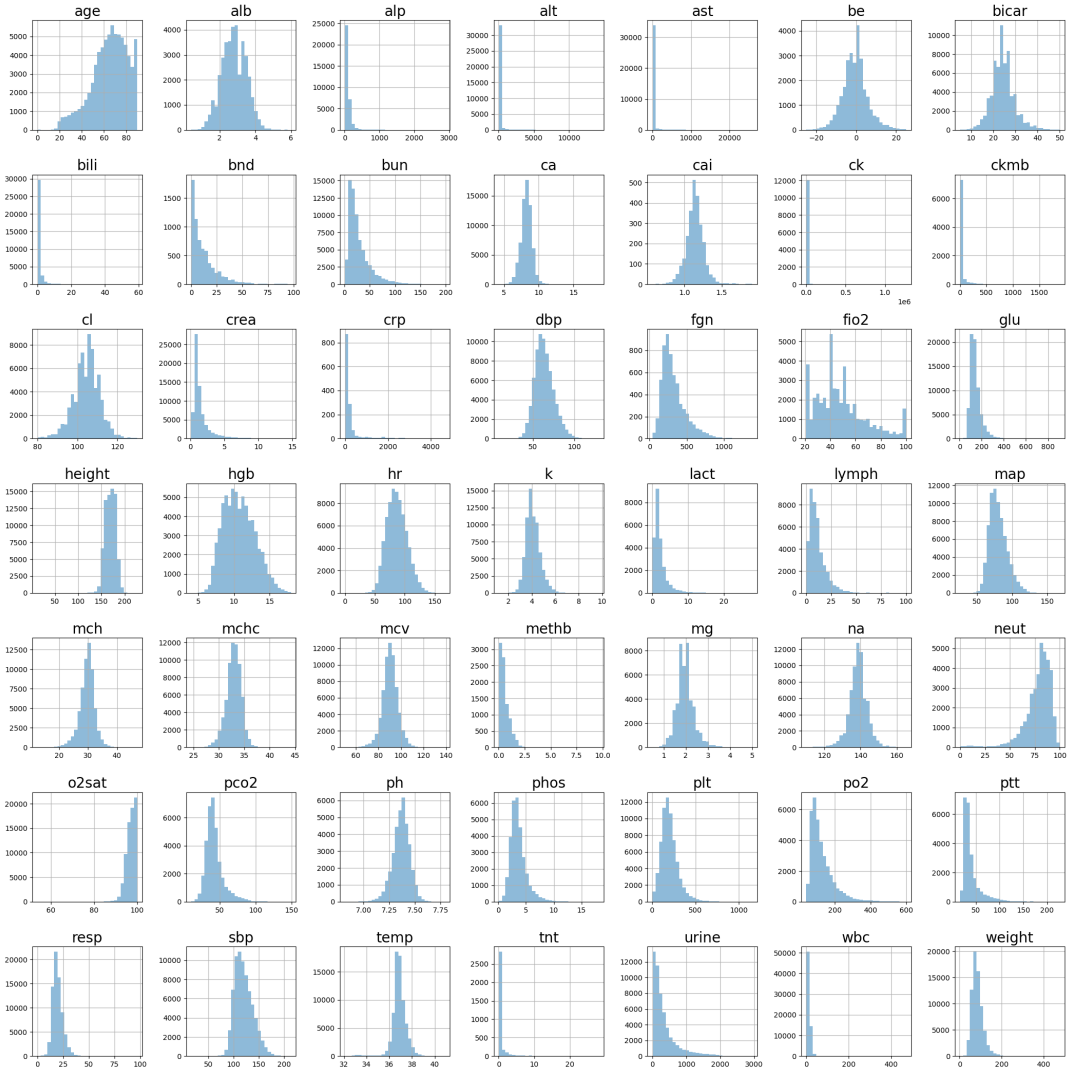


Figure B.1: Histogram of clinical concepts in eICU using 30 bins.

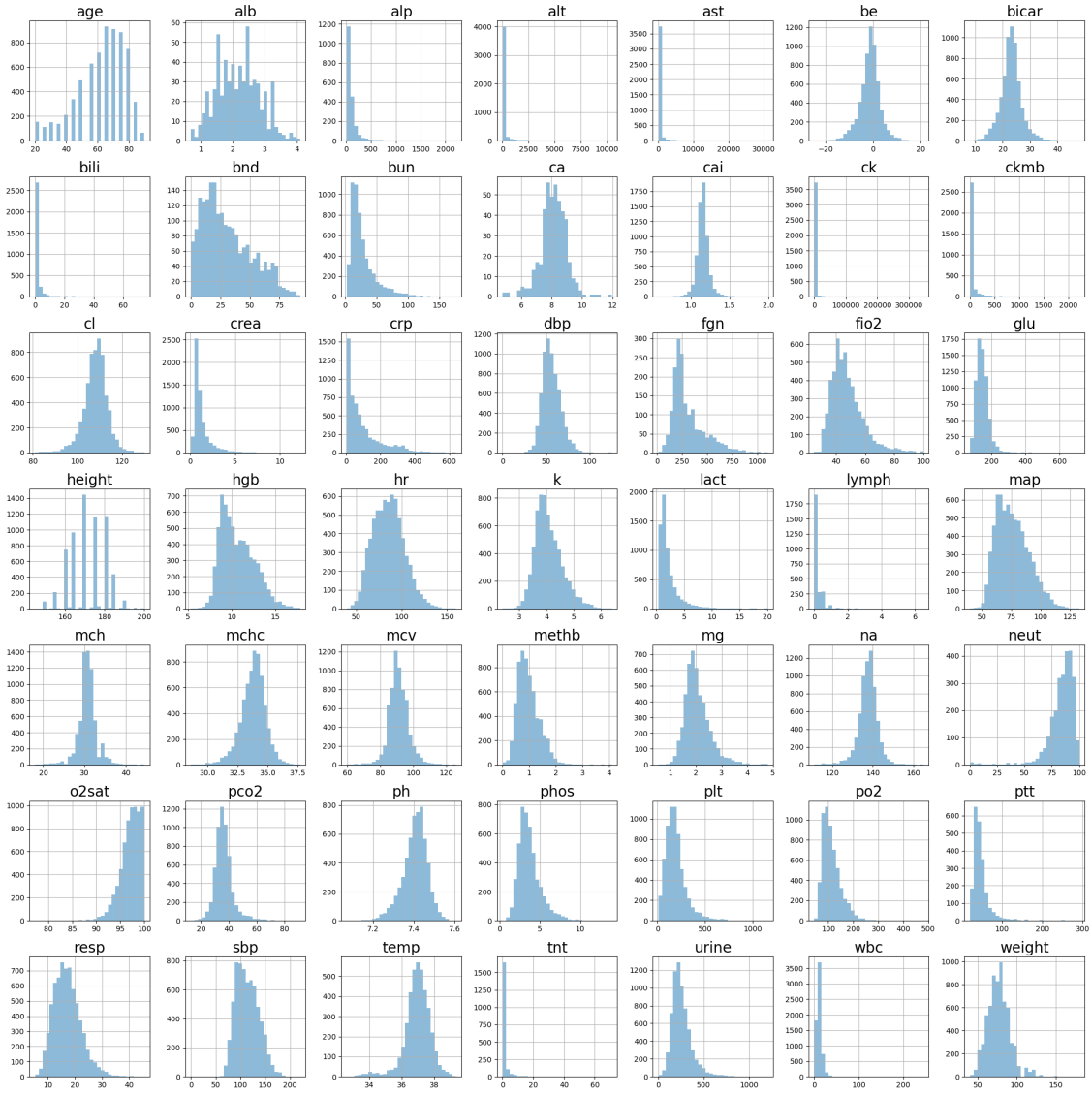


Figure B.2: Histogram of clinical concepts in HiRID using 30 bins.

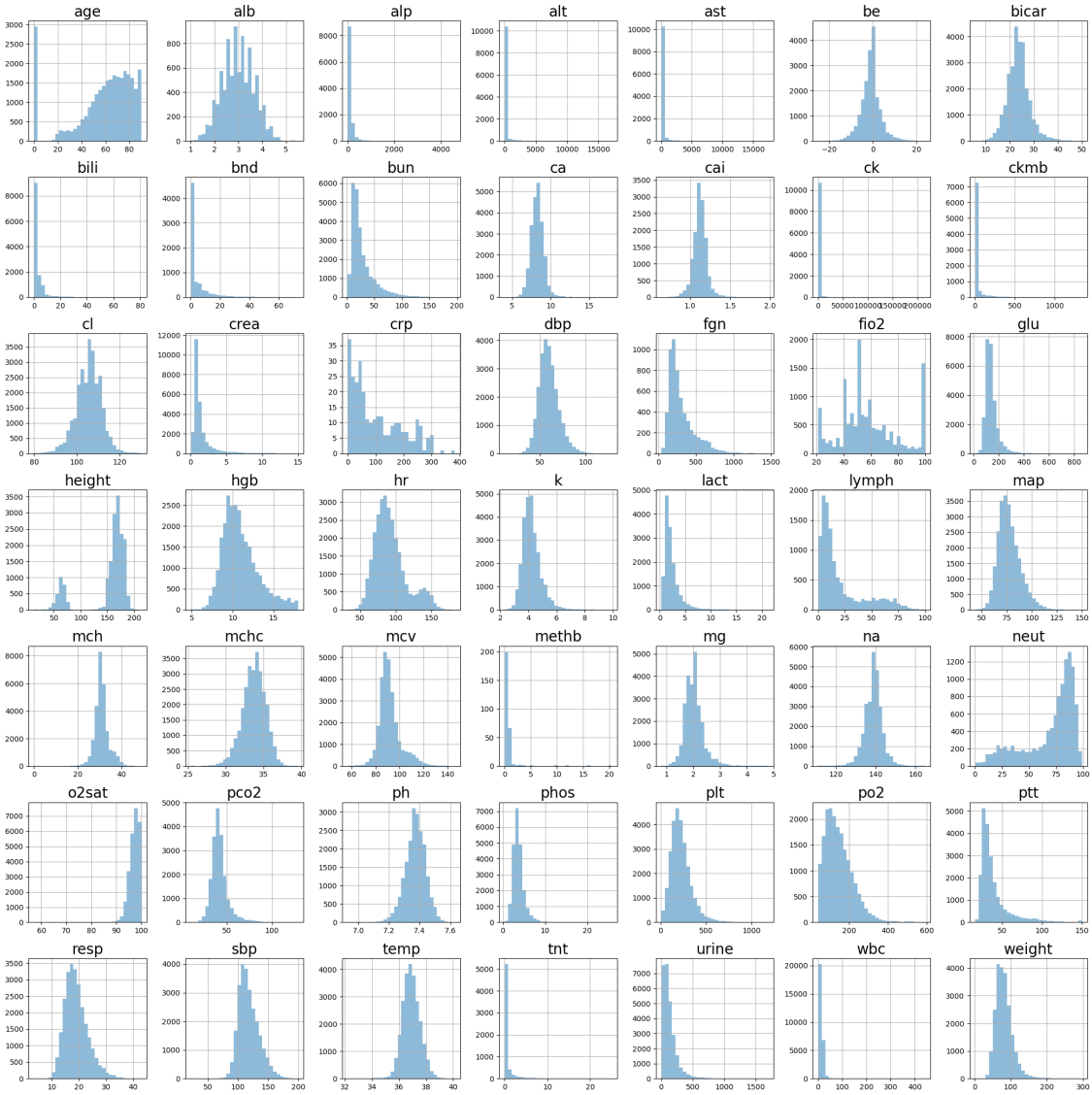


Figure B.3: Histogram of clinical concepts in MIMIC-III using 30 bins.

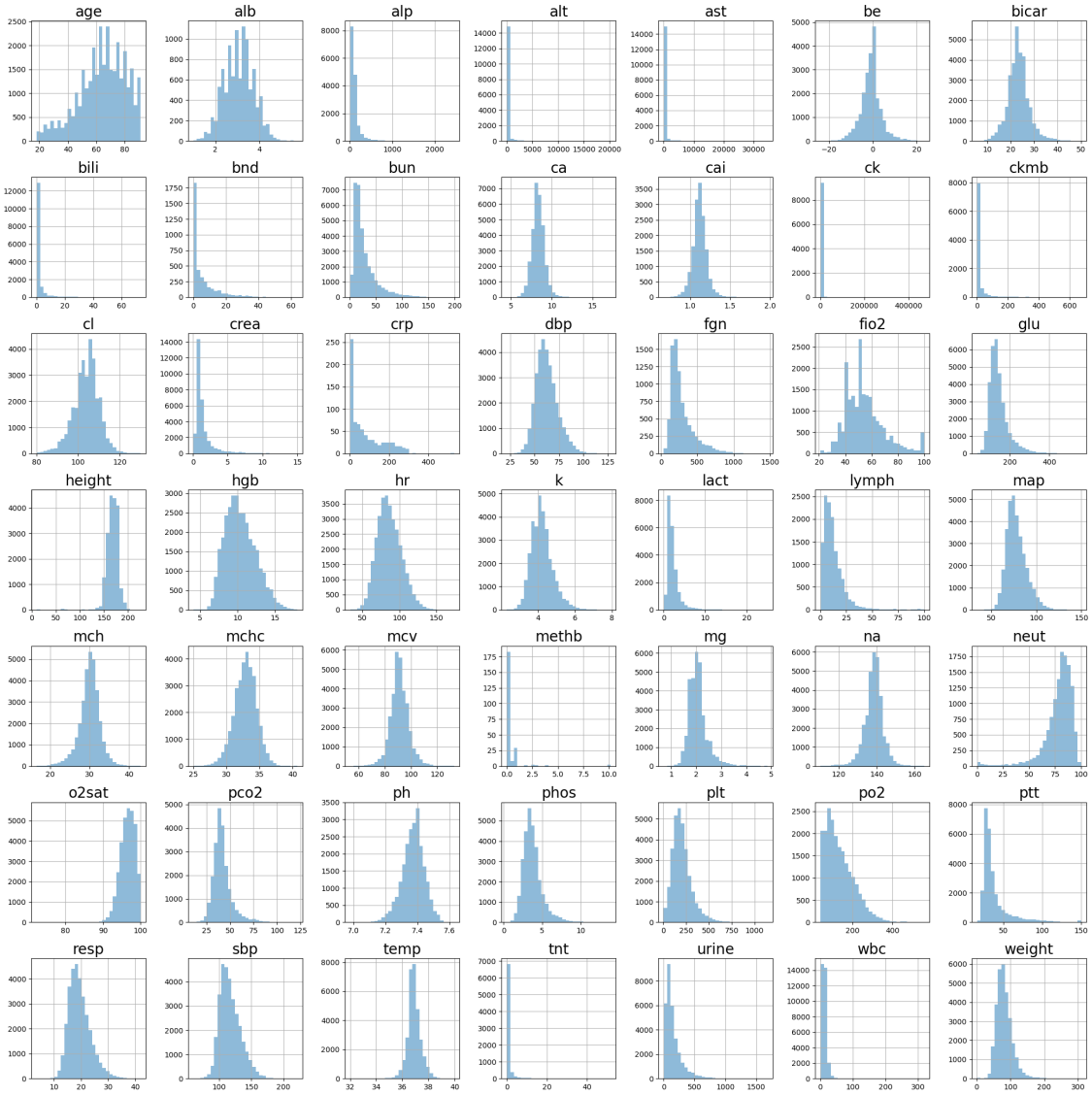


Figure B.4: Histogram of clinical concepts in MIMIC-IV using 30 bins.

B.2 Fine-tuning approach

B.2.1 OLS

The Ordinary Least Squares estimator (OLS) is a linear model that aims to minimize the sum of the squared residuals. We assume a linear relationship between y and X and allow for Gaussian noise. That is:

$$y = X\Theta^* + \varepsilon \quad \text{where} \quad \varepsilon \sim \text{i.i.d } \mathcal{N}(0, \sigma^2)$$

Assuming that $(X^T X)$ is regular, we can estimate the regression coefficients β by

$$\widehat{\Theta} = (X^T X)^{-1} X^T y = \arg\min_{\Theta \in \mathbb{R}^p} \|y - X\Theta\|_2.$$

B.2.2 Application to ICU data

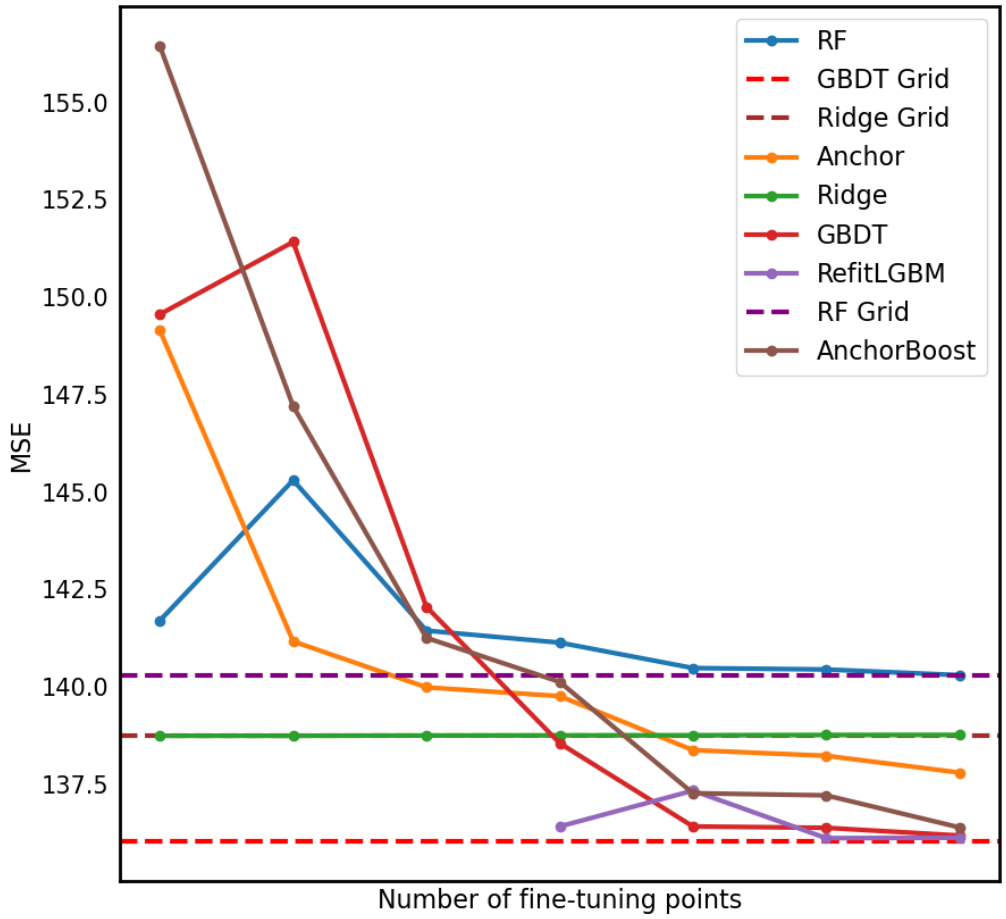


Figure B.5: MSE vs. number of fine-tuning data points on MIMIC-IV.

B.3 Magging

All following results were generated using a GBDT from LightGBM for ensemble prediction. We performed the computations as in Chapter 4.4, and used "hr" as response variable.

Parameter	Values
num_leaves	20, 30, 40
learning_rate	0.01, 0.1, 0.2
n_estimators	100, 200, 300

Target	MSE
eicu	150.57
hirid	165.02
mimic	282.50
miiv	143.28

Table B.1: Used Numbedscategory from the eICU dataset as groups. Calculated weights: [0, 0, 1, 0]. (Categories: '100 - 249', '250 - 499', '<100', '>= 500').

Target	MSE
eicu	136.40
hirid	159.32
mimic	215.17
miiv	136.96

Table B.2: Used Teachingstatus from the eICU dataset as groups. Calculated weights: [1.0, 0.0]. (Categories: False, True).

Target	MSE
eicu	147.27
hirid	165.07
mimic	224.40
miiv	140.12

Table B.3: Used Region from the eICU dataset as groups. Calculated weights: [0, 1, 0, 0]. (Regions: Midwest, Northeast, South, West).

Target	MSE
eicu	147.04
hirid	162.06
mimic	251.57
miiv	140.07

Table B.4: Used Ethnicity from the eICU dataset as groups. Calculated weights: [0, 0, 1, 0]. (Ethnicity's: Asian, Black, Other and White).

Target	MSE
eicu	149.94
hirid	172.14
mimic	109.88
miiv	133.42

Table B.5: Used Ethnicity from the MIMIC-III dataset as groups. Calculated weights: [0, 0, 0, 1].

Target	MSE
eicu	148.43
hirid	166.56
mimic	216.88
miiv	120.27

Table B.6: Used Ethnicity from the MIMIC-IV dataset as groups. Calculated weights: [0, 0, 0, 1].

Target	MSE
eicu	138.05
hirid	159.58
mimic	210.15
miiv	137.73

Table B.7: Age groups from eICU used as groups. Calculated weights: [0, 0, 1, 0]. (Age groups: child: $0-19$, young adult: $20-39$, middle-age $40-65$, senior: 65+).

Target	MSE
eicu	151.76
hirid	134.32
mimic	266.99
miiv	140.87

Table B.8: Age groups from HiRID used as groups. Calculated weights: [0.0, 0.45, 0.55]. In HiRID, the age group child does not exist. The Matrix H is not positive definite.

We conclude this Section with one comment: Magging can determine optimal weights in situations where an oracle has complete access to the test data, i.e. the weights identified by Magging using the training data match the optimal weights determined when Magging has access to the test data, utilizing the ensemble estimators from the training data. However, it's crucial to note that comparing with oracle weights is valid only when our training data can be segmented in a manner consistent with the segmentation of our test data.

B.4 Data Shared Lasso

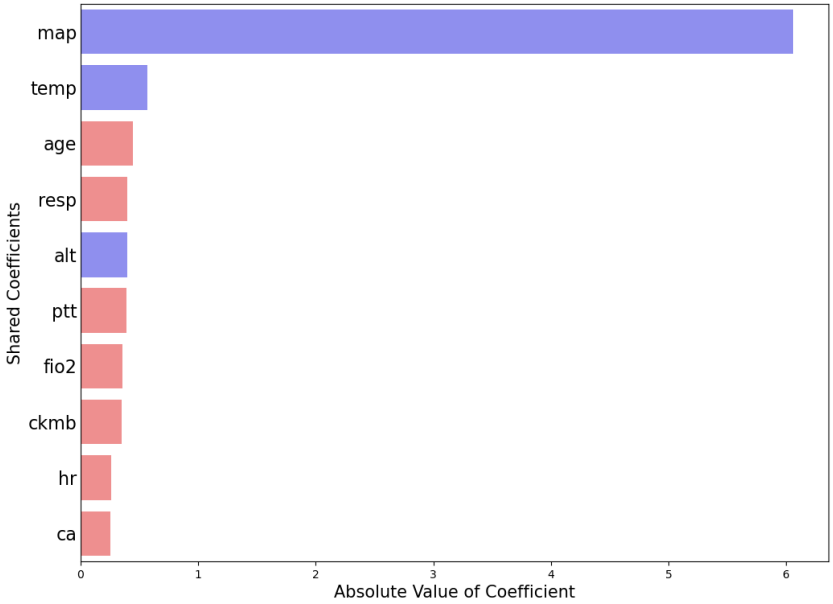
We performed computations as in Chapter 5.5. We used ”map” as response variable.

Type	**Concepts**
Shared effects	fio2, age, temp, map
eICU	Male, sbp, dbp, map
MIMIC-III	Male, mcv, mch, Female
MIMIC-IV	Male, mcv, Female, mch
HiRID	dbp, map, Female, Male

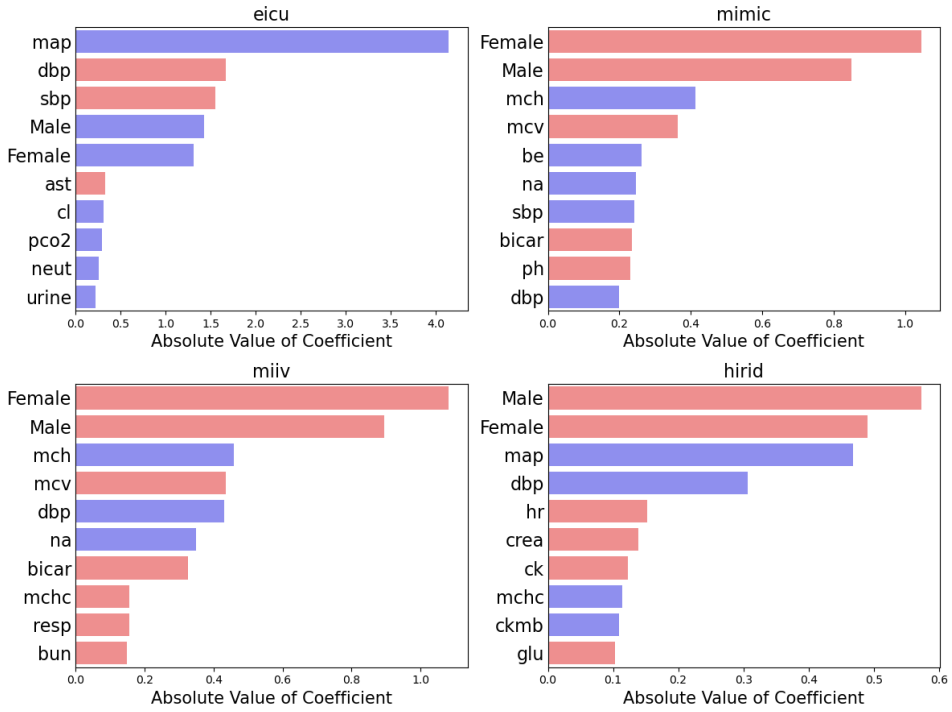
Table B.9: Displaying the four largest features in Lasso paths (Figure B.9), sorted by absolute magnitude.

Dataset	**Concepts**
eICU	temp, sbp, dbp, map
MIMIC-III	mchc, mcv, mch, map
MIMIC-IV	mchc, mcv, mch, map
HiRID	hr, mch, dbp, map
Pooled	Female, dbp, sbp, map

Table B.10: Displaying the four largest features in Lasso paths (Figure B.10), sorted by absolute magnitude.

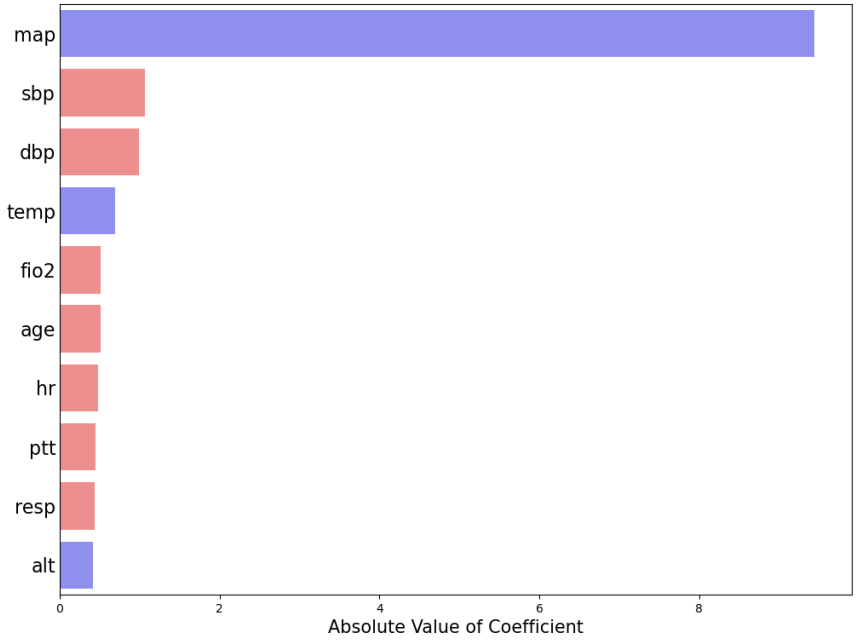


(a) Shared coefficients.

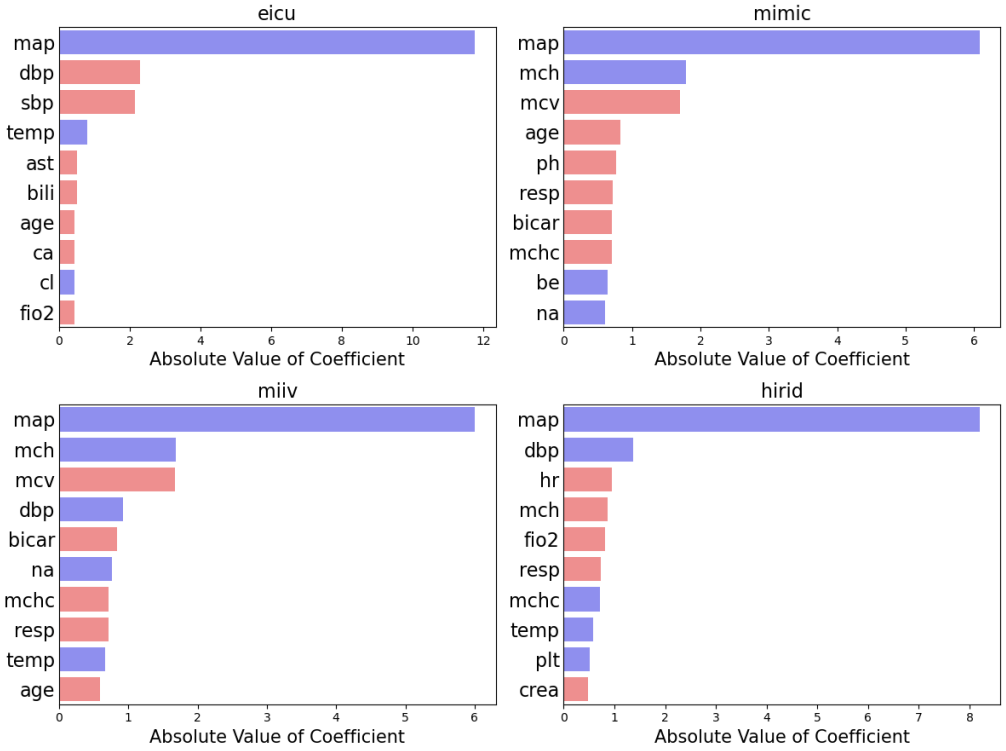


(b) Group Specific Coefficients.

Figure B.6: Data Shared Lasso coefficients bar plots.

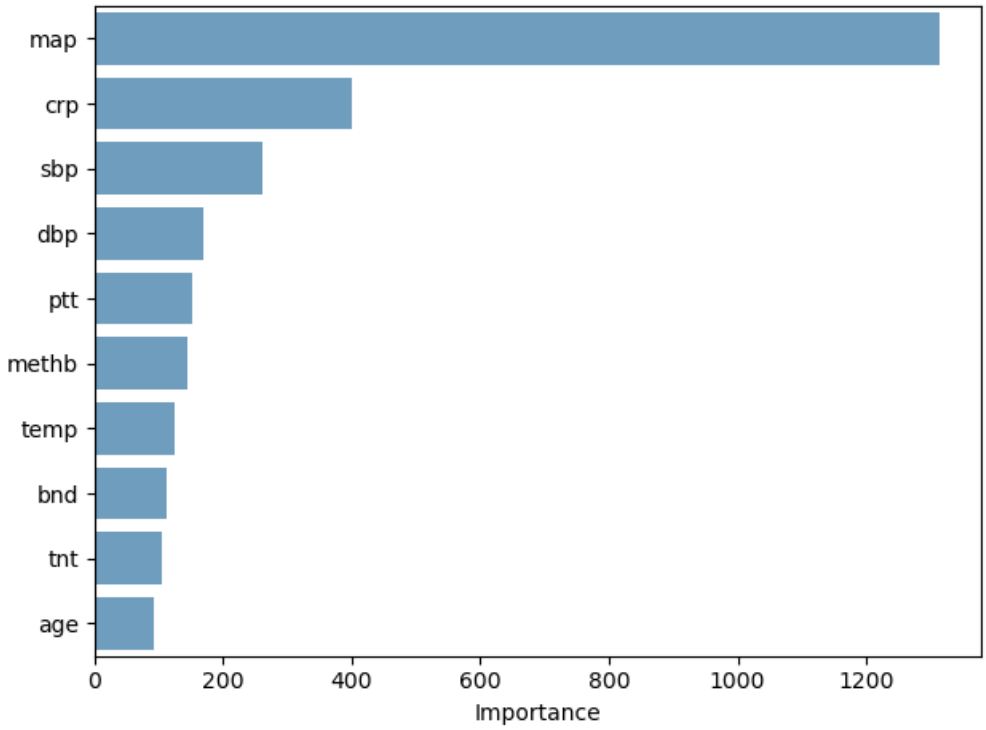


(a) Pooled Lasso coefficients.

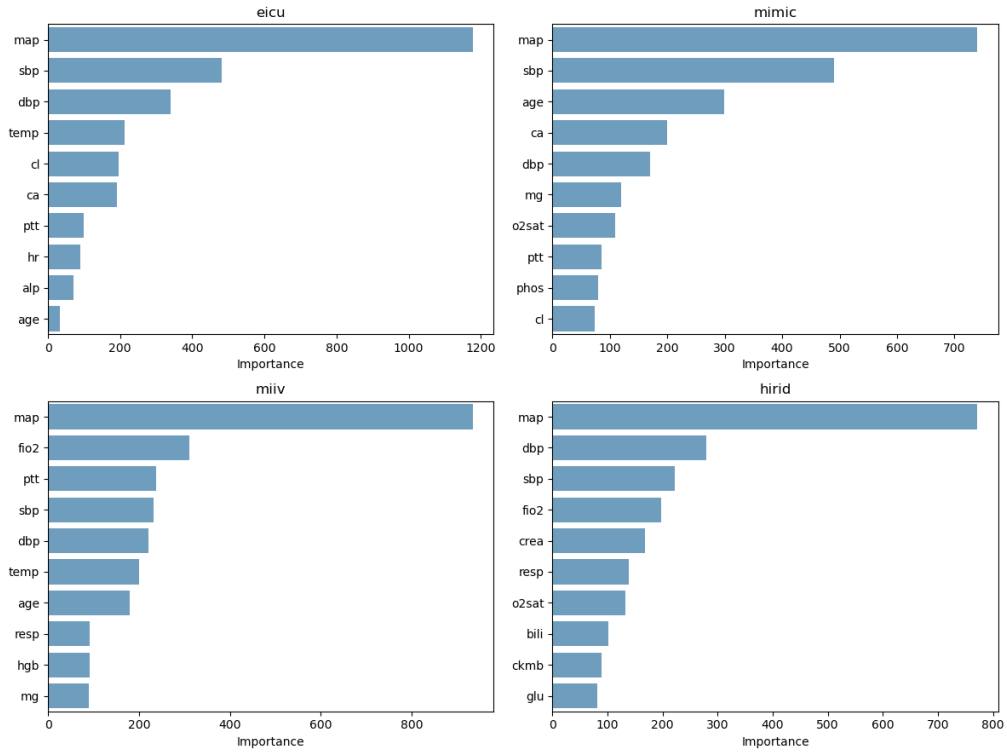


(b) Separate Lasso coefficients.

Figure B.7: Coefficients bar plots from running a separate Lasso on each dataset and running pooled Lasso.

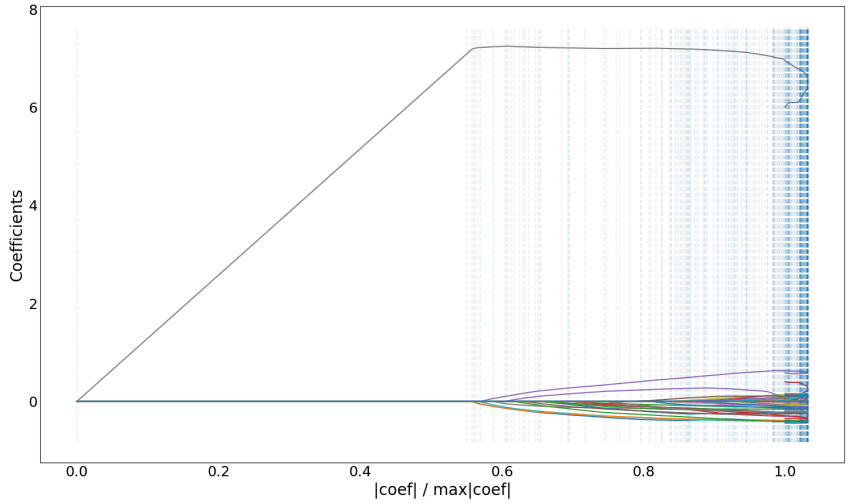


(a) Feature importances of a Random Forest trained on all datasets.

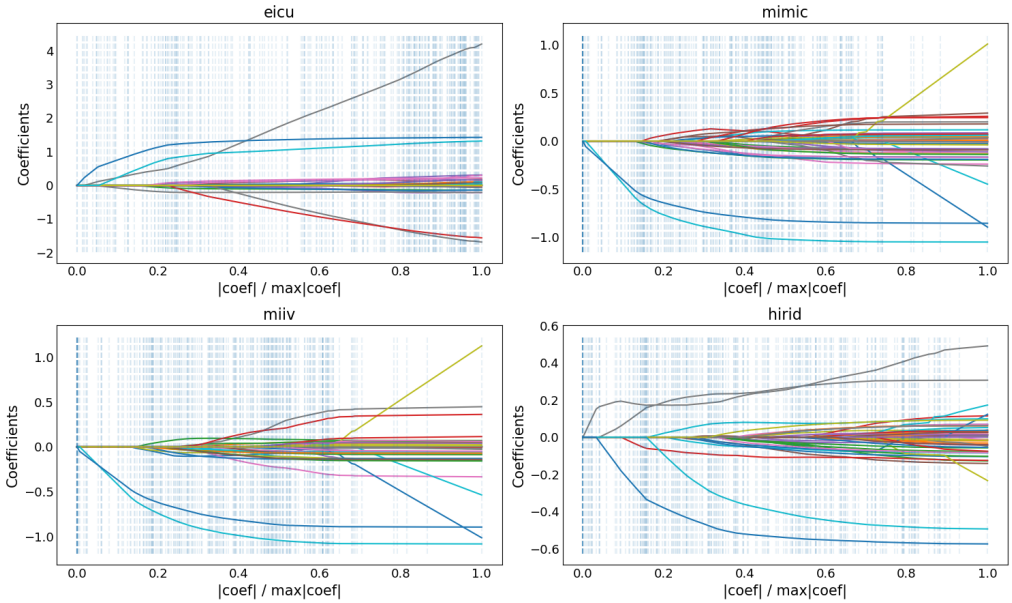


(b) Feature importances of a Random Forest trained on each dataset separately.

Figure B.8: Random Forest feature importances.

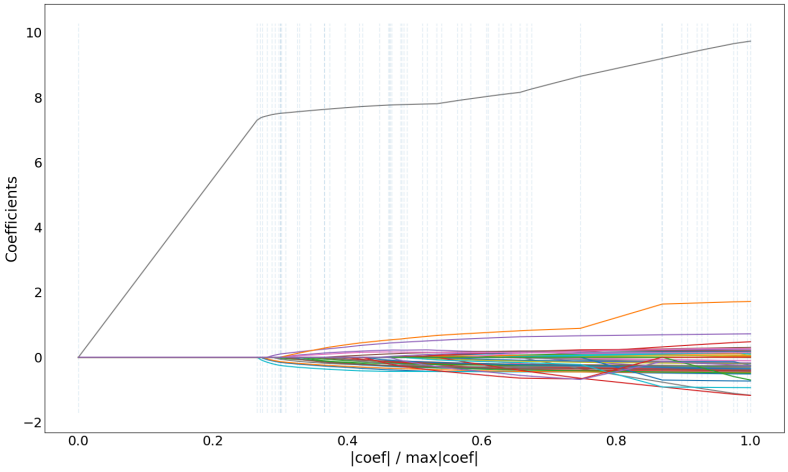


(a) Shared Effects.

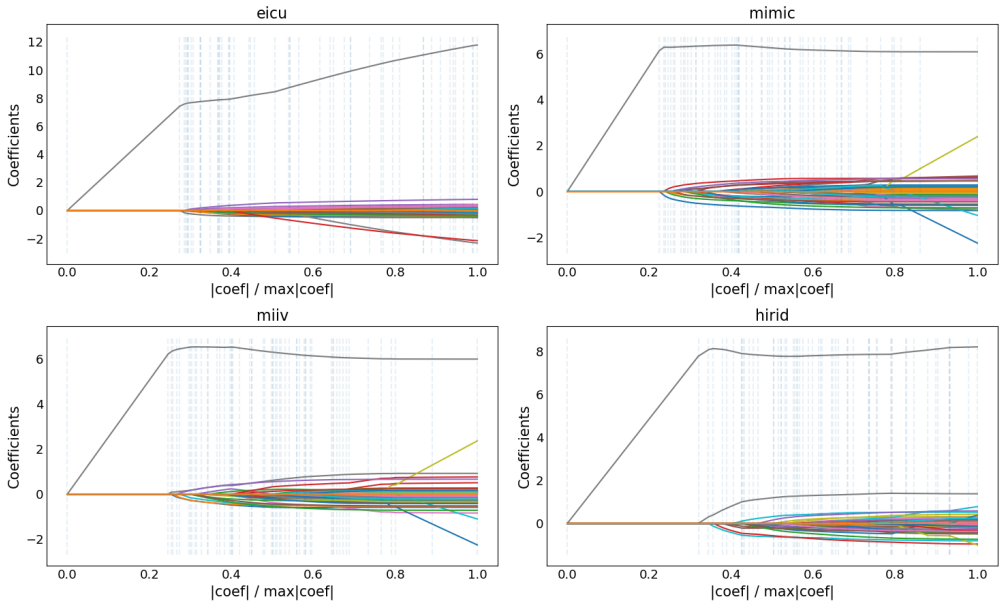


(b) Group Individual Effects.

Figure B.9: Data Shared Lasso profiles.



(a) Pooled Lasso.



(b) Group Individual Profiles.

Figure B.10: Pooled and individual Lasso profiles.

Eidgenössische Technische Hochschule Zürich
Swiss Federal Institute of Technology Zurich

Eigenständigkeitserklärung

Die unterzeichnete Eigenständigkeitserklärung ist Bestandteil jeder während des Studiums verfassten schriftlichen Arbeit. Eine der folgenden drei Optionen ist in Absprache mit der verantwortlichen Betreuungsperson verbindlich auszuwählen:

○ Ich bestätige, die vorliegende Arbeit selbständig und in eigenen Worten verfasst zu haben, namentlich, dass mir niemand beim Verfassen der Arbeit geholfen hat. Davon ausgenommen sind sprachliche und inhaltliche Korrekturvorschläge durch die Betreuungsperson. Es wurden keine Technologien der generativen künstlichen Intelligenz[1] verwendet.

○ Ich bestätige, die vorliegende Arbeit selbständig und in eigenen Worten verfasst zu haben, namentlich, dass mir niemand beim Verfassen der Arbeit geholfen hat. Davon ausgenommen sind sprachliche und inhaltliche Korrekturvorschläge durch die Betreuungsperson. Als Hilfsmittel wurden Technologien der generativen künstlichen Intelligenz[2] verwendet und gekennzeichnet.

◉ Ich bestätige, die vorliegende Arbeit selbständig und in eigenen Worten verfasst zu haben, namentlich, dass mir niemand beim Verfassen der Arbeit geholfen hat. Davon ausgenommen sind sprachliche und inhaltliche Korrekturvorschläge durch die Betreuungsperson. Als Hilfsmittel wurden Technologien der generativen künstlichen Intelligenz[3] verwendet. Der Einsatz wurde, in Absprache mit der Betreuungsperson, nicht gekennzeichnet.

Titel der Arbeit:

External Validity of Predictors in ICU Data

Verfasst von:
Bei Gruppenarbeiten sind die Namen aller Verfasserinnen und Verfasser erforderlich.

Name(n):	**Vorname(n):**
Brilhaus	Luca

Ich bestätige mit meiner Unterschrift:
- Ich habe mich an die Regeln des «Zitierleitfadens» gehalten.
- Ich habe alle Methoden, Daten und Arbeitsabläufe wahrheitsgetreu und vollständig dokumentiert.
- Ich habe alle Personen erwähnt, welche die Arbeit wesentlich unterstützt haben.

Ich nehme zur Kenntnis, dass die Arbeit mit elektronischen Hilfsmitteln auf Eigenständigkeit überprüft werden kann.

Ort, Datum	**Unterschrift(en)**
Zürich, 03.04.2024	L Brilhaus

Bei Gruppenarbeiten sind die Namen aller Verfasserinnen und Verfasser erforderlich. Durch die Unterschriften bürgen sie grundsätzlich gemeinsam für den gesamten Inhalt dieser schriftlichen Arbeit.

[1] z. B. ChatGPT, DALL E 2, Google Bard
[2] z. B. ChatGPT, DALL E 2, Google Bard
[3] z. B. ChatGPT, DALL E 2, Google Bard