Report – Project 1

Marc Egli (17-709-155), Noah Mamié (17-607-714), Nicolas Koch (17-602-103)

April 2023

All code used to produce the results discussed in this report can be found here:

- Part 1: Heart failure: https://github.com/nmamie/ML4H_Project1/tree/master/heart_failure
- Part 2: Pneumonia: https://github.com/nmamie/ML4H_Project1/tree/master/pneumonia

1 Heart Disease

1.1 Exploratory Analysis

We first describe our findings of our exploratory data analysis (EDA) and later discuss their implications.

Summary Statistics

To get a first overview over the data, we compiled summary statistics depicted in Table 1. It makes plain that we are dealing with continuous/numerical features as well as categorical ones of 2–4 different levels.

On first sight there seemed to be no missing values. This was revoked in the next analysis.

Distributions

Figures 1–3 display the distributions of the label and the predictors. Generously speaking, apart from `Oldpeak`, the numerical features seem to approximately follow a Gaussian distribution and don't show indications of fat tails. This is important to determine adequate strategies for imputation and estimation of standard deviation (e.g. for standardization). The categorical features appear rather imbalanced. For instance, the majority of the samples are male. Importantly, though, the label `HeartDisease`, seems balanced.

On further inspection, the variables `Cholesterol` and `RestingBP` seemed to have unusual "0" entries. While there was only one such occurrence for `RestingBP` in the training data, there were 141, nearly 20% of the sample size, for `Cholesterol`. Compared to the

	Age	RestingBP	Cholesterol	MaxHR	Oldpeak
count	734.00	734.00	734.00	734.00	734.00
mean	53.52	132.06	197.59	136.17	0.87
std	9.42	18.62	108.98	25.33	1.08
min	29.00	0.00	0.00	60.00	-2.00
25%	47.00	120.00	172.25	120.00	0.00
50%	54.00	130.00	222.00	138.00	0.50
75%	60.00	140.00	267.00	155.00	1.50
max	77.00	200.00	529.00	195.00	6.20

	Sex	ChestPainType	FastingBS	RestingECG	ExerciseAngina	ST_Slope
count	734	734	734	734	734	734
unique	2	4	2	3	2	3
top	M	ASY	0	Normal	N	Flat
freq	573	381	563	440	439	364

Table 1: Summary statistics of numerical (top) and categorical (bottom) features.

distribution of the other values, this seems erroneous, and we suspected that "0" is a placeholder for missing data. Also `Oldpeak` has notably many "0" entries, namely over 40% of the total observations. Yet, this lies within the range of the other values. It does not raise excessive suspicion given our (limited) medical knowledge.

Dependencies

In the following, we examine dependence structures between `HeartDisease` and the individual features as well as between the features themselves.

Figures 4 and 5 illustrate the distributions of the features grouped by `HeartDisease`. We would expect features whose distribution is more different depending on the value of `HeartDisease` to offer more predictive value. From the plots, we get indications that this may most likely be the case for `MaxHR`, `Oldpeak`, and several of the categorical features.

Next, we examined rank correlations, as determined by Spearman's ρ, to capture any type of (monotonic) dependence structure between the variables. In case of binary variables (which includes the outcome variable `HeartDisease`), it corresponds to linear correlation. Results are depicted in Figure 6 and Table 2.

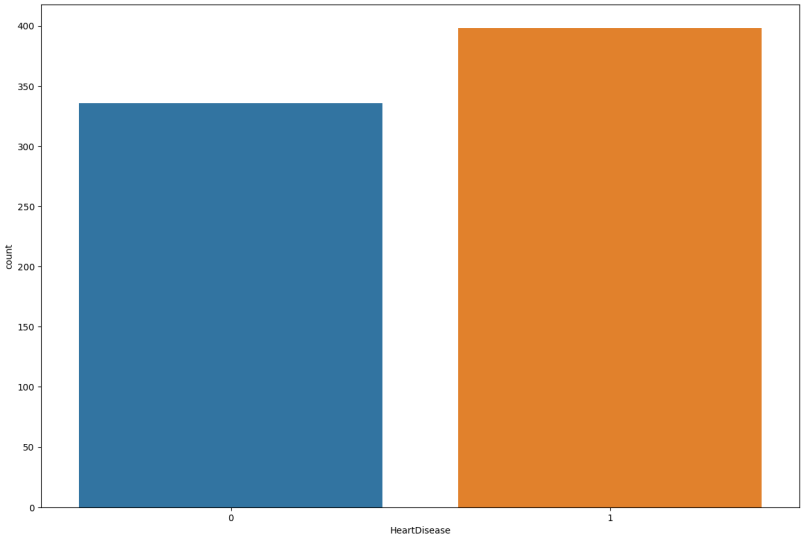


Figure 1: Distribution of the target variable, `HeartDisease`. Importantly, it seems balanced.

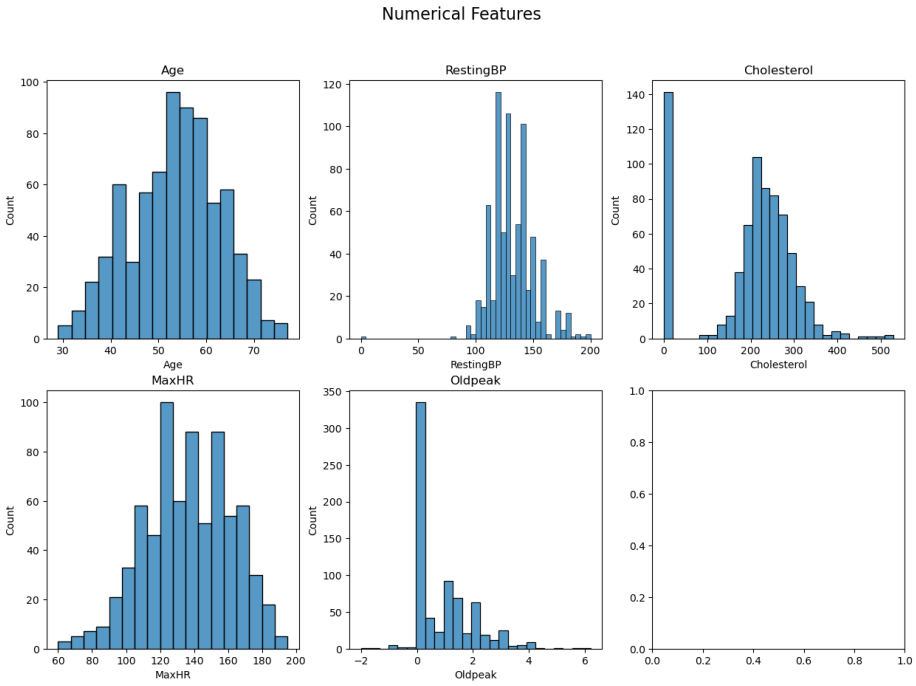


Figure 2: Distribution of continuous features. They seem approximately Gaussian. Suspicious peaks can be seen at 0 for both `Cholesterol` and `Oldpeak`.

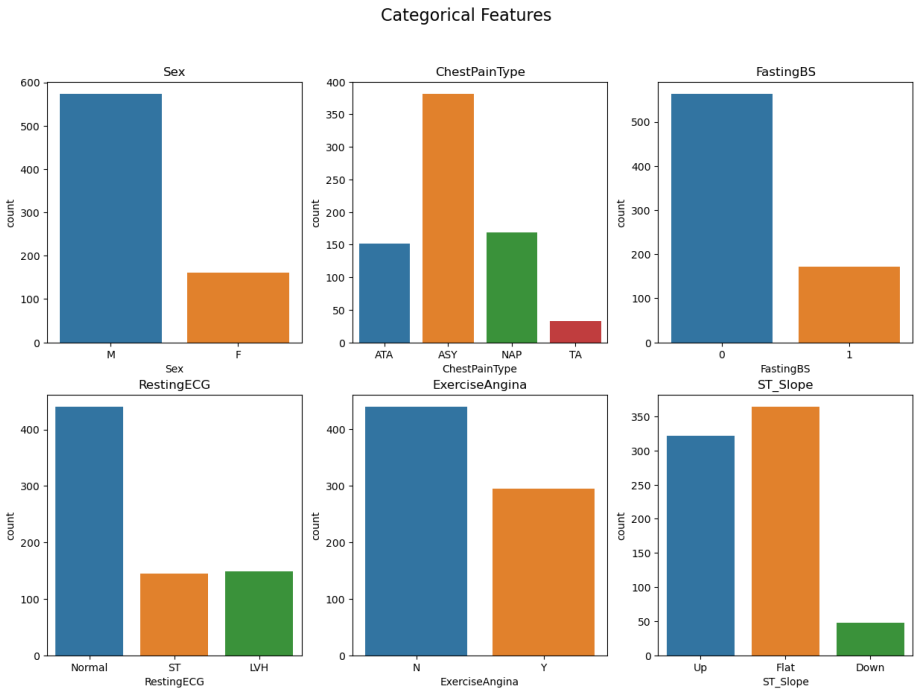


Figure 3: Distribution of categorical features. Some seem to be unevenly distributed. For prediction performance, there should be enough cases for each category, but results should be interpreted with more care for people that are underrepresented in our data set (e.g. `Sex = "F"` & `Chest_pain_type = "IA"`).

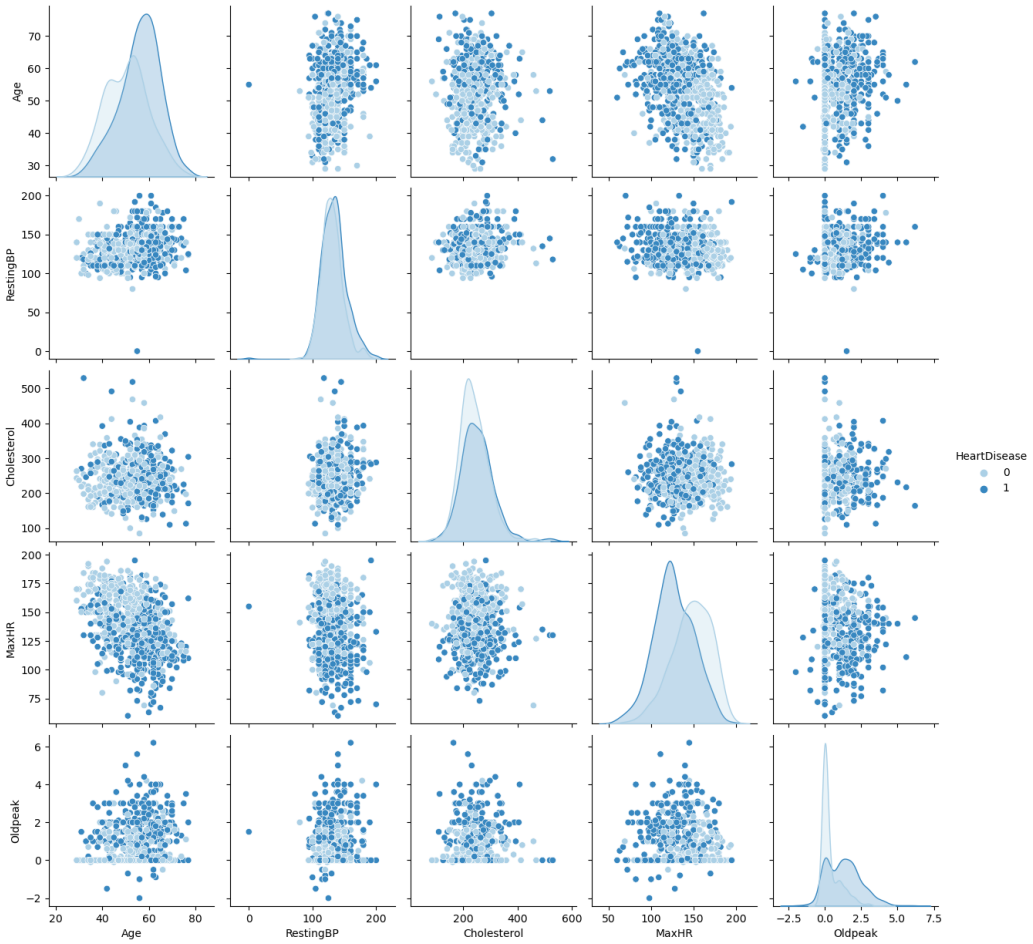


Figure 4: Conditional distributions of numerical features given `HeartDisease`. Varying conditional distributions indicate that `Age`, `MaxHR`, and `Oldpeak` are related to `HeartDisease`.

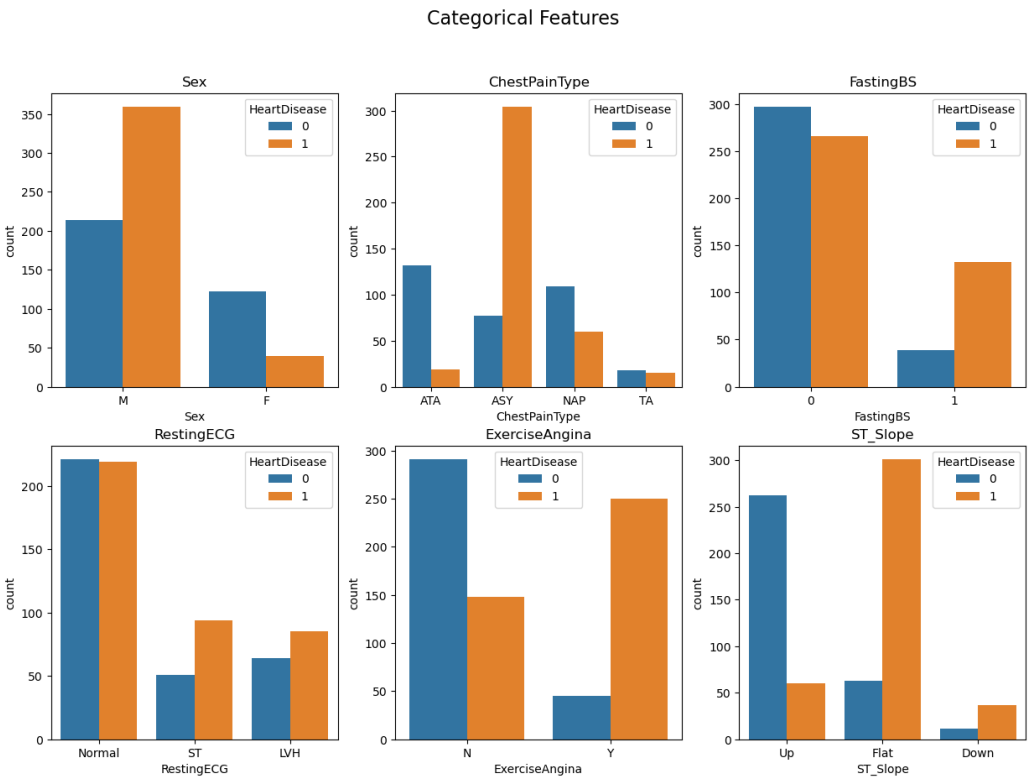


Figure 5: Distributions of categorical features grouped by value of `HeartDisease`. We have indications for several variables to be related with `HeartDisease`.

Feature	Spearman ρ	correlation w other features
`ST_Slope = "Flat"`	0.57	–
`ExerciseAngina = "Y"`	0.50	`MaxHR` (-0.41); `Oldpeak` (0.42)
`Oldpeak`	0.44	`ExerciseAngina = "Y"` (0.42)
`ChestPainType = "ATA"`	-0.43	–
`MaxHR`	-0.41	`ExerciseAngina = "Y"` (-0.41)

Table 2: Top 5 features with highest absolute Spearman rank correlation coefficient (ρ) with `HeartDisease` and their correlation with other features with which they are significantly correlated ($|\rho| > 0.4$).

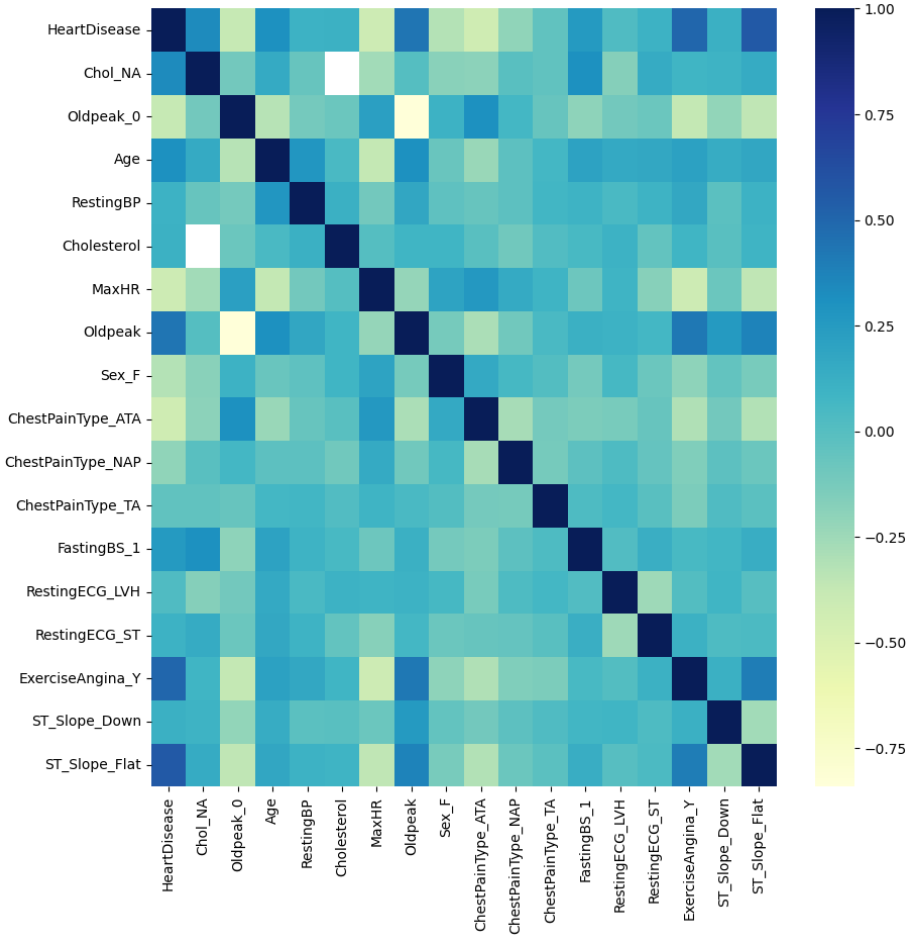


Figure 6: Heatmap of rank correlations between the variables as determined by Spearman's ρ.

The plots and rank correlations suggest `HeartDisease` has a strong positive association

with `ST_Slope = "Flat"`, `ExerciseAngina = "Y"`, and `Oldpeak`. Strong negative associations can be seen with `MaxHR` and `ChestPainType = ATA`.

Importantly, some of these features are also highly correlated with *each other*. For example, `ExerciseAngina = "Y"` and `Oldpeak` are significantly positively correlated, and hence their correlation with `HeartDisease` might partially be explained through the respective other predictor.

Implications

From the EDA, we can derive some actionable implications for further analyses.

Firstly, there are implications for adequate *preprocessing* of the dataset, namely:

1. *One-Hot Encoding:* Categorical features have to be converted to dummy variables, since most implementations cannot handle them. Mapping their categories to numerical values would invalidate results, as there is no numeric relationship between them. Whether to keep all dummies or drop a defined reference category depends on the specific model used.

2. *Imputation:* (Suspected) missing values in `Cholesterol` & `RestingBP` should be imputed to uphold sample size. We decided to go with simple mean imputation to keep things simple and traceable. As the features seem Gaussian, the mean is more statistically efficient. Though we also tried median imputation and obtained near-identical results, which was to be expected based on the distributions.

3. *Indicators:* As the occurrence of e.g. a missing value alone could be of predictive value, indicator variables are a useful addition. Here, we add them to indicate initial 0 entries in `Cholesterol` and `Oldpeak` and name them `Chol_NA` and `Oldpeak_0`, respectively.

4. *Standardization:* For estimating some models and assessing feature importances, variable standardization is essential. As we do not have indications of fat tails, we use the standard estimator as opposed to the robust one. We also standardize dummy variables. Although this seems unnatural, it is theoretically valid and improves comparability.

Secondly, we draw attention to less actionable findings that nonetheless are important to keep in mind:

1. *Correlated predictors:* This increases the variance of our estimates, reducing our confidence about their validity and significance. Even more importantly, however, when using (automatic) variable selection, it can introduce randomness in the choice of relevant and irrelvant (in terms of slope coefficients or more general feature importance scores) variables. For instance, while we would expect at least one of `ExerciseAngina = "Y"` and `Oldpeak` to appear relevant in later analyses, their correlation with each other makes it less likely that both are deemed relevant simultaneously, yet which of the two may be somewhat arbitrary.

2. *Imbalanced predictors:* These entail two consequences. First, results for underrepresented groups, such as females, may be less valid due to interactions and reduced sample size. Second, reduced variance in the predictors generally increases variance in estimates, decreasing our confidence about their validity. However, although suboptimal, we cannot do much about them.

3. *`Oldpeak`:* Erroneous "0" entries in the predictor `Oldpeak` would be highly problematic. However, we cannot confidently say that they are. Additionally, as they constitute over 40% of the sample size, there is little remaining data to reliably predict these values, while single value imputation could significantly skew results. We hope to capture any relevant information from the "0" entries with the indicator variable.

1.2 Logistic Lasso Regression

Implementation

Logistic regression is a linear model for the log-odds of the labels. Lasso regularization introduces a penalty linear in the absolute coefficients, in addition to the maximum likelihood objective. Equivalently, it upper bounds the sum of coefficients' absolute values, thereby shrinking them (relative to the maximum likelihood estimates). Therefore, two considerations are crucial to ensure viable results:

1. *Standardization:* When using regularization, variables with larger variances are at an advantage, since they will be associated with small coefficients. To ensure a fair comparison purely based on variables' predictive value, we should therefore put all on the same scale by standardizing them. We did this for numerical and dummy variables alike.

2. *Group penalization:* When working with categorical features, one usually creates dummy variables and, to avoid multicollinearity, defines one category as reference. However, we want to penalize the inclusion of entire variables, not only some of their categorical values. Hence, we purposefully use dummies for *all* categories of a variable and use the *group lasso*.

Using the standardized data, we chose the best of 10 different group-penalty terms in the range 0.001–0.5 through 5-fold cross-validation, optimizing the F1 criterion. We chose not to encourage within-group sparsity. We relied on the `group-lasso` library's `LogisticGroupLasso` function.

Performance

The performance of our logistic regression classifier is summarized in Figure 7. It achieved an overall accuracy of 84%. Precision is 83% and recall/sensitivity is 90%, yielding an F1-score of 86%, while specificity is 73%. The AUC of the classifier is 0.90, which is well above the uninformed level of 0.50. Hence, we may work with the results.

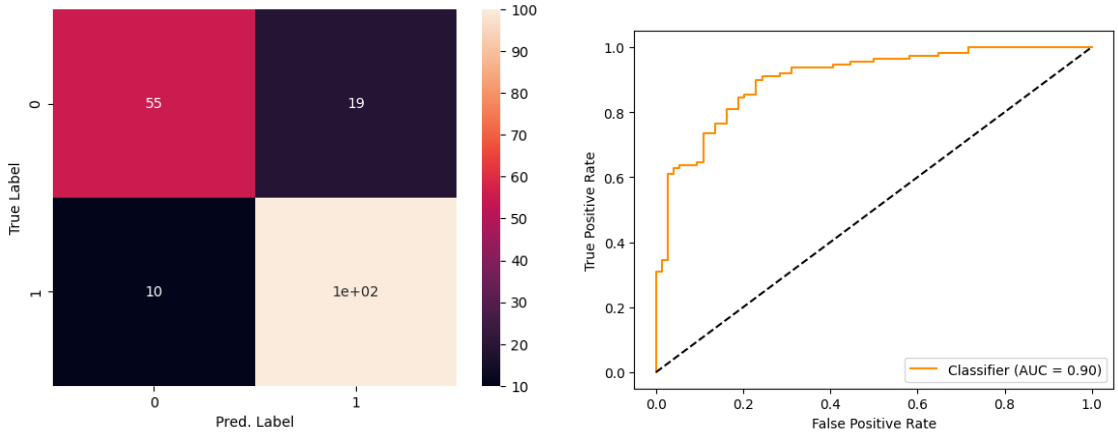


Figure 7: Confusion Matrix and ROC of the logistic classifier's predictions on the test dataset.

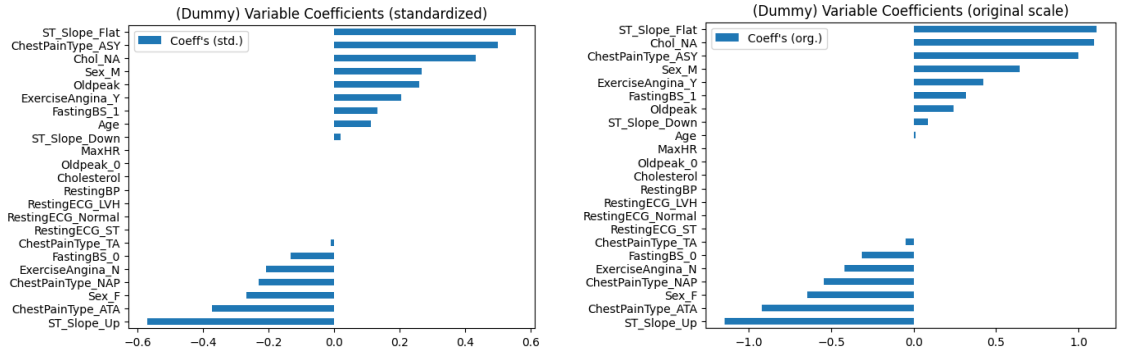


Figure 8: Estimated LASSO coefficients on standardized (left) and original (right) scale.

Interpretation

Figures 8–9 summarize the estimated model. The plots can be interpreted as follows:

Standardized coefficient estimates indicate the increase in the log-odds of `HeartDisease` associated with a 1 standard deviation increase in the variables, and hence allow us to directly compare the importances of specific categories. For example, `ChestPain = "ASY"` and `ST_Slope = "Flat"` seem to be strongly associated with heart disease. We also find evidence that whether `Cholesterol` was 0 originally, and thus interpreted as missing by us, had predictive value for heart disease. This could hint at an anomaly in the data collection.

Original coefficient estimates allow us to infer the association between risk of heart disease and the variables in the intuitive scales. For example, we would expect the log-odds of `HeartDisease` to increase by ≈ 0.01 for every additional year of age. Note that, since we included dummies for all categories, coefficients of categorical variables need to be interpreted *relative to the other dummies*. For example, `ST_Slope = "Flat"` increases

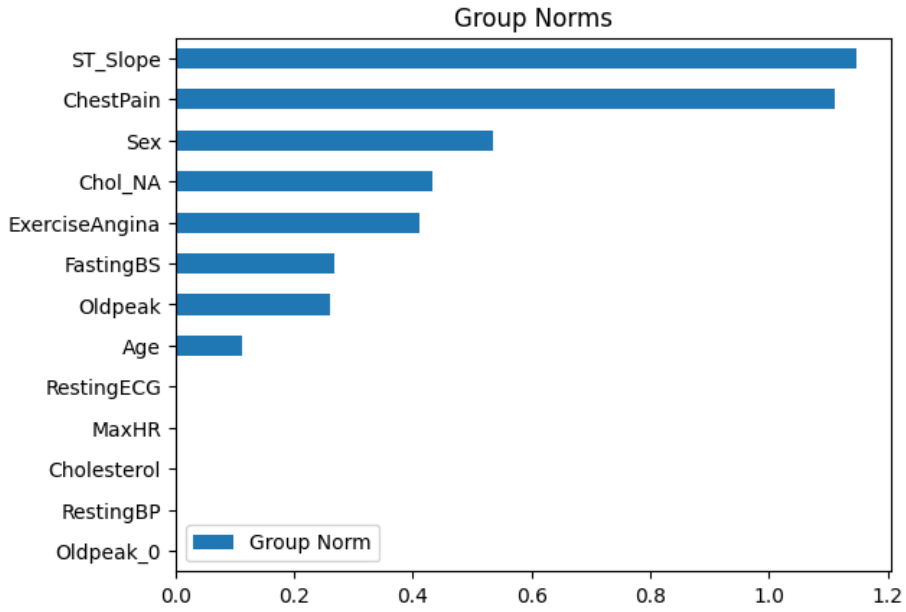


Figure 9: Group L1 norms of fitted LASSO coefficients, i.e. the sum of absolute coefficients of a group of features.

the log-odds of `HeartDisease` by ≈ 1.1, while `ST_Slope = "Up"` decreases it by ≈ 1.2, and hence the log-odds of a patient with `ST_Slope = "Flat"` are about 2.3 higher than those of a patient with `ST_Slope = "Up"`.

Group norms allow us to compare variable importances. These constitute the sum of absolute values of groups' *standardized* coefficients. In this case, `ST_Slope` and `ChestPain` offered the most predictive value for `HeartDisease`. As may have been expected, only one of `ExerciseAngina` and `MaxHR` were selected, yet both `ExerciseAngina` and `Oldpeak` are, despite their heavy correlation.

Discussion: Re-fitting for de-biasing

Variable selection and variance reduction from regularization comes at the cost of biased coefficient estimates. Refitting an unregularized model with only the selected variables can de-bias the coefficient estimates. The original issues with an unregularized model, such as irrelevant noise variables, multicollinearity, correlated predictors, and potentially high-dimensionality, would have been resolved at this point.

A problem with refitting could be *post-selection bias*, wherein we may be overly confident in our estimates or even variable importances. Theoretically, there might a random association between an irrelevant variable and heart disease in our training data, making it more likely that it is selected and given a large coefficient estimate also upon refitting. This may be resolved by refitting the model on a *separate* dataset ("sample splitting"), where

this association is unlikely to be repeated. However, this comes at the cost of reduced sample size.

We believe that the benefits of sample splitting concentrate on estimation of significance, confidence bounds, etc. (Bühlmann, 2021, Lecture 9). If we are only interested in coefficient estimates, re-fitting on the same dataset should be fine in the majority of practical cases. For other objectives, a great many extensions of the lasso have been developed. For instance, the *adaptive* (Zou, 2006) and *relaxed lasso* (Meinshausen, 2007), to (partially) de-bias estimates and improve prediction; *stability selection* to get more reliable variable selection with controls (Meinshausen and Bühlmann, 2010); multiple *sample splitting* to get reliable estimates for confidence bounds (Meinshausen et al., 2009);, etc.

1.3 Decision Tree

Implementation

To build a classification tree, we relied on `sklearn`'s `DecisionTreeClassifier` function. Like in the logistic classifier, we used all dummies for all categories, as this makes all categories equally accessible for the tree and multicollinearity is generally not an issue[1]. In contrast, we used the non-standardized data to ease interpretation.

Since trees easily overfit, we performed 5-fold cross-validation with the F1 criterion to choose the tree's optimal depth from $\{1, ..., 7\}$ and minimum number of samples per leaf node from $\{2^0, ..., 2^6\}$. Each tree was fitted using the Gini impurity measure. This resulted in an optimal tree of depth 3 with a minimum of 8 samples per leaf node.

Performance

The performance of our classification tree is summarized in Figure 10. It achieved an overall accuracy of 79%, precision of 79%, and recall/sensitivity of 88%, yielding an F1-score of 79%, while specificity is 65%. The AUC of the classifier is 0.86, which is well above the uninformed level of 0.50. Hence, we may work with the results.

Overall, the tree performs slightly worse than the logistic model. Yet, for such a simple and interpretable model it has impressive performance.

Interpretation

The fitted decision tree is depicted in Figure 11. To obtain a prediction for a sample, one can simply follow the tree nodes from top to bottom. For instance, a patient for whom `ST_Slope` $\neq$ `"Up"`, `ChestPainType = "ASY"`, and `Oldpeak` $\leq$ `1.85` would be predicted to have heart disease.

Feature importances, measured in reduction of Gini impurity, are depicted in Figure 12. Just like the logistic classifier, the tree selected `ST_Slope = "Up"`, `ChestPainType = "ASY"`,

[1] For example, with a default category for the 4-level categorical feature `ChestPainType`, the tree would have to create 3 splits to select samples of the default category (setting the other 3 to 0), which is highly inefficient. Instead, with all dummies, any category can be accessed with 1 split.

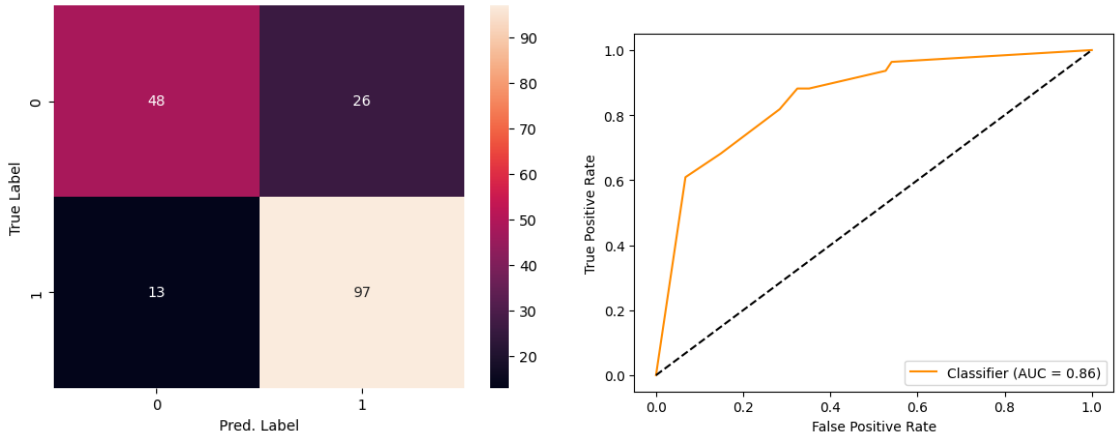


Figure 10: Confusion Matrix and ROC of tree's predictions on the test dataset.

`Sex = "M"`, and `Oldpeak` as some of the most important variables. Although the logistic classifier assignmed more importance to `ExerciseAngina` than `Oldpeak`, recall that these two are highly correlated, and hence this difference may partially stem from randomness in the optimization of the regularized objective resp. the splits of the tree.

1.4 MLP

Implementation

Using `PyTorch`, we defined a simple feedforward neural network composed of 3 fully connected layers of width 10. We used ReLU activations and dropout regularization with $p = 0.1$ after the first two. The output layer is a single node, representing the log-odds of the label, which we transformed to a probability estimate with the sigmoid transform. Using binary cross-entropy loss and the `Adam` optimizer, we trained the network for 100 epochs.

For the MLP, we worked with non-standardized data and dropped the dummies for default categories, both of which ease interpretation of the final model.

Performance

The performance of our MLP is summarized in Figure 13. It achieved an overall accuracy of 84%, precision of 85%, and recall/sensitivity of 90%, yielding an F1-score of 87%, while specificity is 76%. The AUC of the classifier is 0.90, which is well above the uninformed level of 0.50. Hence, we may work with the results.

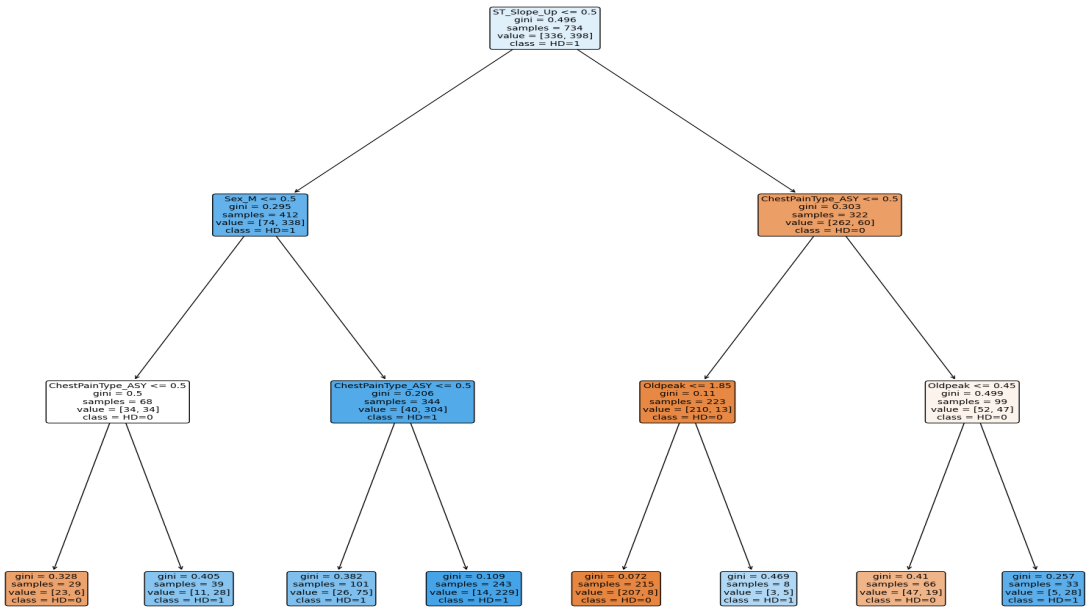


Figure 11: Fitted classification tree. If condition in node is fulfilled, then go left, otherwise go right. Color schemes represent the majority class as a gradient (blue: `HeartDisease = 0`; orange: `HeartDisease = 1`.

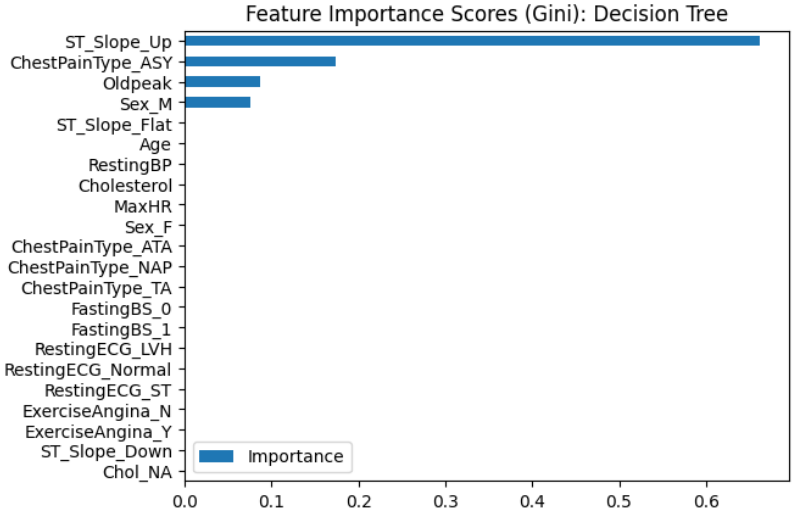


Figure 12: Feature importance of fitted tree as measured by reduction of Gini impurity.

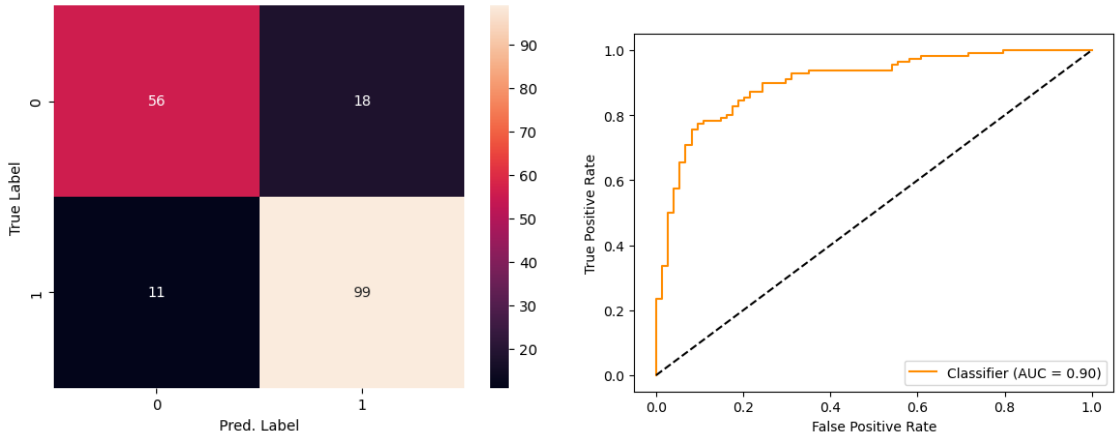


Figure 13: Confusion Matrix and ROC of MLP's predictions on the test dataset.

Interpretation

Recall that we dropped one dummy per feature (see Table 3 for default categories). This means that dummy effects are to be interpreted relative to the default categories. To calculate feature importance we calculated Shapley values, as proposed by Lundberg and Lee (2017). Figure 14 shows the Shapley values for 8 correctly classified observations from the test set (4 positive, 4 negative). While the exact contributions of a feature differ from observation to observation the top 10 predictors, measured by Shapley value, seem to be consistent. Similar to the tree and LASSO methods, `ST_Slope, ChestPainTyp, Sex, ExerciseAngina, Age & Oldpeak` seem to be important predictors for coronary heart disease. The top 10 predictive features can also be seen in Figure 15, where a point corresponds to one observation. In line with our previous analysis `Sex = "M", ST_Slope = "Flat", ExerciseAngina = "Y", ChestPainType = "ASY"`, seem to be associated with a higher risk of heart disease. Interestingly, the model made use of the `Chol_NA` variable, our dummy variable for if the Cholesterol value was imputed/missing. This could hint at an anomaly in the data collection that should be further investigated.

Feature	Default Category
Sex	M
ChestPainType	ASY
FastingBS	0
RestingECG	Normal
ExerciseAngina	N
ST_Slope	Up

Table 3: Default categories used in MLP and NAM.

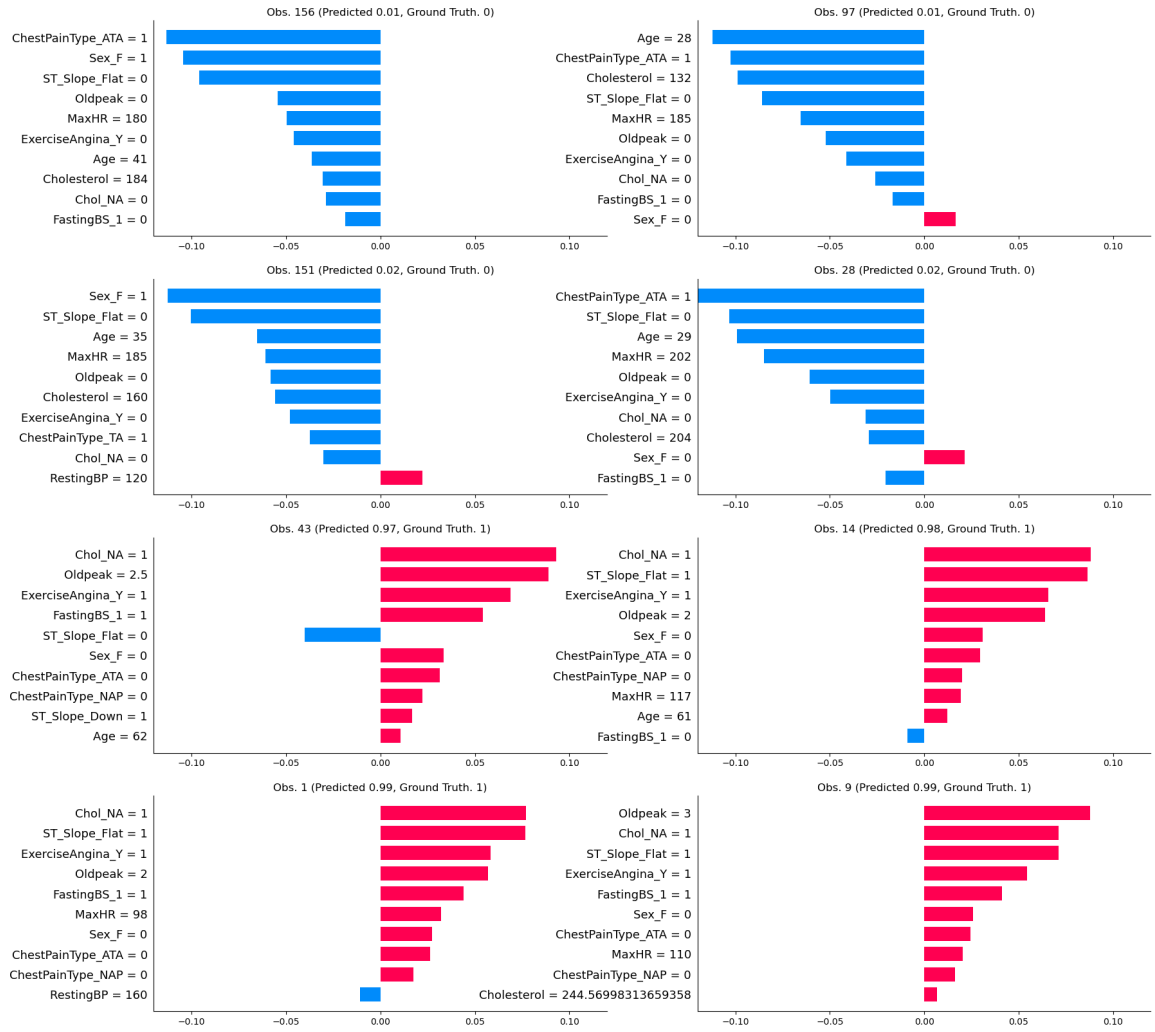


Figure 14: Depicted are the shap values for 4 positive and 4 negative examples.

Challenge 1: Neural Additive Model

Implementation

For the NAM, we followed the tutorial of Kübler (2017). Using `PyTorch`, we implemented a neural network with so-called "block layers", which constitute layers consisting of different blocks whose nodes are unconnected to nodes from other blocks. Figure 16 gives a general illustration of this architecture. Since we want to keep the features separate, one block was implemented per feature. We used a total of 3 block layers, in which the first two are of width 5 and the third is of width 1. The final layer is a single node connected to all of these outputs from the different block layers. We also used leaky ReLU activations after every

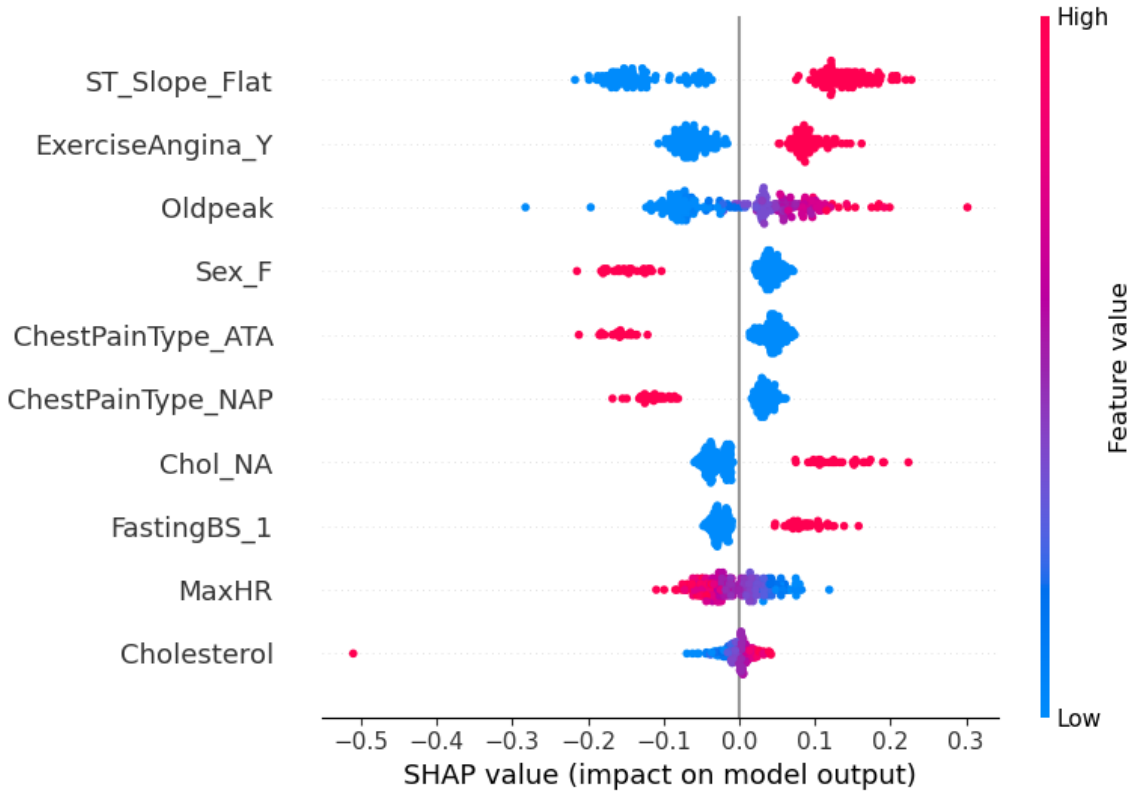


Figure 15: Depicted are the 10 most important features based on shapley value and how they influenced individual prediction, which are depicted as single points. The feature importance is consistent with the single observations in Figure 14 and previous LASSO and tree analysis.

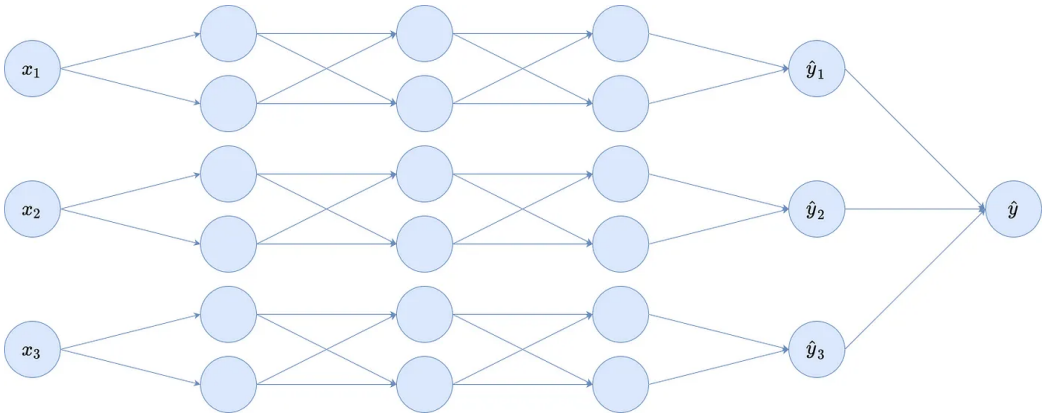


Figure 16: Illustration of the NAM architecture for width 2 and depth 4. Note that the effectively implemented NAM is of width 5 and depth 3.

layer and, during training, dropout regularization with $p = 0.2$ after the first two (blocked) layers.

We turned the predictive function into predicted probabilities for `HeartDisease` using the sigmoid transformation on the final output. We trained the network using binary cross-entropy loss for 1000 epochs.

As with the MLP, we dropped a default dummy variable and fitted the model on unstandardized data to ease interpretation of individual effects of features.

Performance

The performance of our NAM is summarized in Figure 17. It achieved an overall accuracy of 83%. Precision is 83% and recall/sensitivity is 90%, yielding an F1-score of 86%, while specificity is 73%. The AUC of the classifier is 0.90, which is well above the uninformed level of 0.50. Hence, we may work with the results.

Interpretation and relation to logistic model & MLP

Like most classifiers, NAMs model the log-odds $f(\boldsymbol{x})$ of the label, `HeartDisease`, given an input vector $\boldsymbol{x} = (x_1, ..., x_d)$. MLPs, as outlined above, make no further assumptions/restrictions and model $f(\cdot)$ through a fully-connected deep neural network. In contrast, NAMs assume that $f(\cdot)$ is additive in its input dimensions, i.e. that it is of the form

$$f(\boldsymbol{x}) \;=\; \sum_{j=1}^{d} f_j(x_j),$$

which implies that there are assumed to be no interaction effects between features. Restricting $f_j(\cdot)$ to be affine in x_j, so that $f_j(x_j) = \beta_j x_j + c_j$, one obtains the logistic regression model, as outlined in Section 1.2. In contrast, NAMs model $f_j(\cdot)$ as neural networks, which

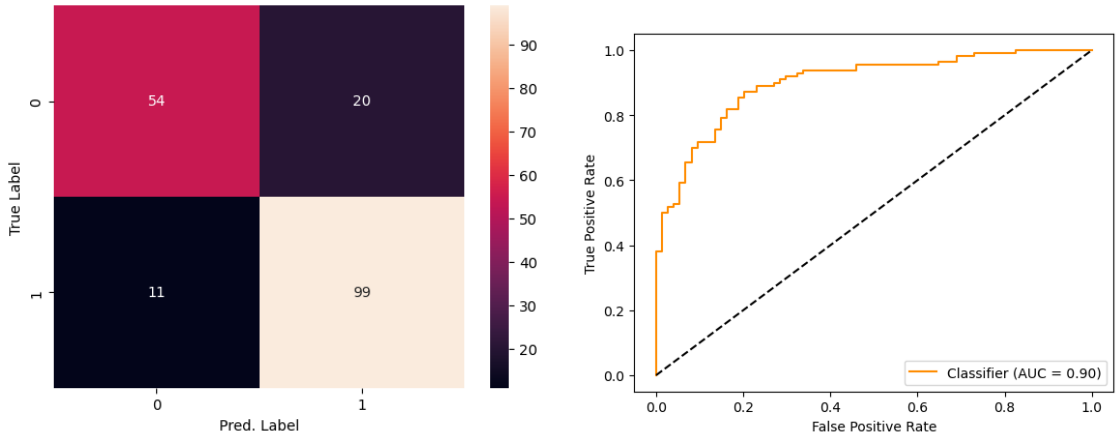


Figure 17: Confusion Matrix and ROC of NAM's predictions on the test dataset.

are significantly more flexible. Overall, NAMs thus lie inbetween logistic regression and MLPs in terms of flexibility as well as interpretability. The additivity assumption allows to examine individual features' importances without worrying about complex interactions as in MLPs.

As mentioned, the mapping $x_j \mapsto f_j(x_j)$ represents a feature's contribution to the log-odds of `HeartDisease` for given values. For the NAM we fitted to the heart disease data, these are displayed in Figure 18. For the dummy variables, the depicted domain is $[0, 1]$, while for numerical variables it is $[\bar{x}_j \pm 3 \cdot \text{sd}(x_j)]$. For dummy variables, there is no interpretation of functional form along $(0, 1)$, but instead the overall change from 0 to 1 is of interest.

Since we used the non-standardized data for training the NAM, we can easily interpret the modelled functions. For example, there seems to be a very monotonic linear relationship between the log-odds of `HeartDisease` and `Age`, `RestingBP`, and `Cholesterol`[2]. Conversely, it is negatively associated with `Oldpeak` below 0, but positively for larger values. The slope of numerical variables' curves indicates the marginal association with the log-odds of `HeartDisease`. For example, a unit increase in `Oldpeak` beyond 0 is associated with a ca. 0.5 increase in the log-odds of `HeartDisease`. Note that the axes are not necessarily to scale, so that a 45° line does not correspond to a 1-to-1 association.

As mentioned, effects of categorical variables are interpreted from changes from 0 to 1, indicating their effect on the log-odds of `HeartDisease`. These are displayed in Figure 19. Recall that we dropped one dummy per variable, and hence these are to be interpreted relative to the default category. For instance, the log-odds of `HeartDisease` for patients with `ST_Slope = Flat` are about 1.5 higher than those of patients with `ST_Slope = Up`.

To compare feauture importances, we have to somewhat abandon absolute interpretability and instead consider changes in terms of standard deviations. Again, standardizing

[2]This likely explains the similarity in performance to the logistic model.

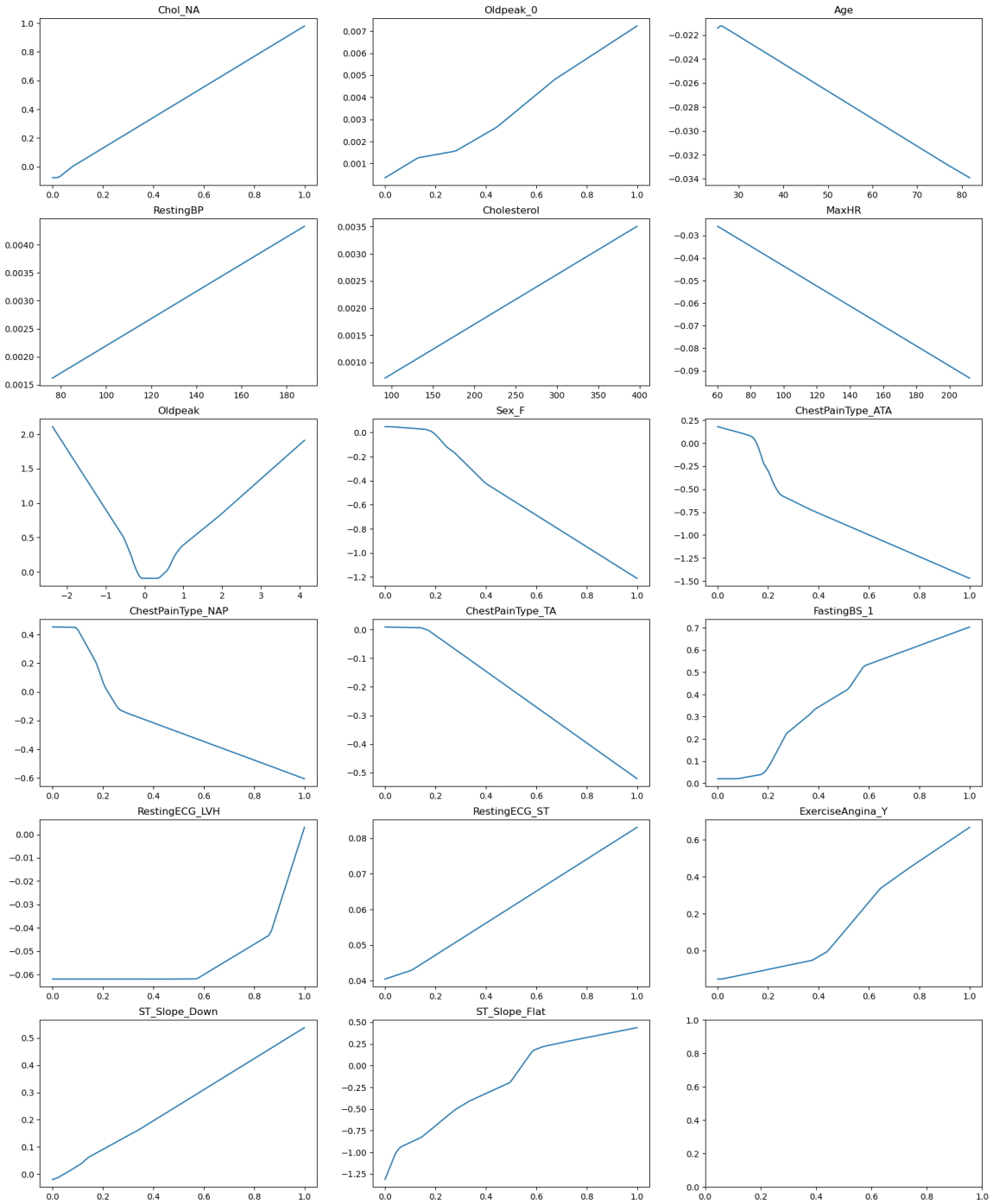


Figure 18: Feature contributions to the log-odds of `HeartDisease` according to the NAM depending on feature value, i.e., $f_j(x_j)$ for all included features j.

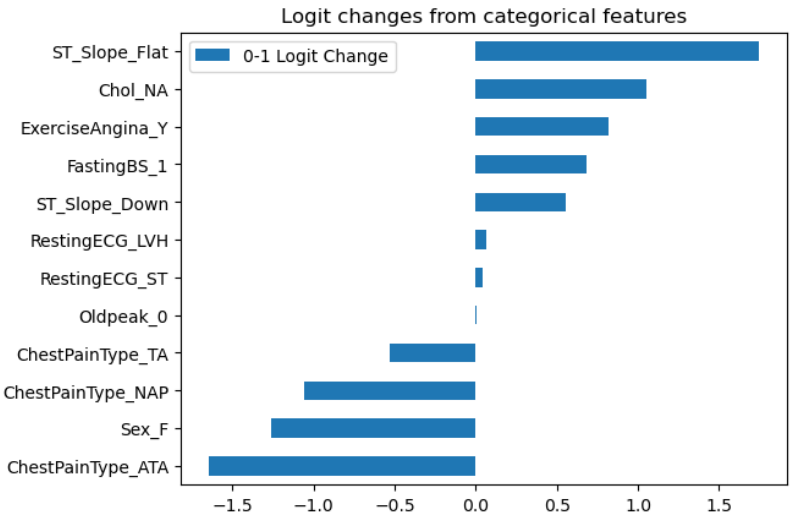


Figure 19: Change in the log-odds of `HeartDisease` from changing dummy variables from 0 to 1, i.e. $f_j(x_j = 1) - f_j(x_j = 0)$.

dummy variables is somewhat unnatural, but levels the playing field. In Figure 20, we show the changes in the log-odds of `HeartDisease` from changing a feature value from 1 SD *below* average to 1 SD *above* average, while keeping other features constant. This corresponds to the quantities $f_j(\bar{x}_j + \text{sd}(x_j)) - f(\bar{x}_j + \text{sd}(x_j))$. Similar to the previous models, `ST_Slope` proves a highly relevant variable, as does `ChestPainType`, `Chol_NA`, and `ExerciseAngina`.

Lastly, we examine individual features' contributions to heart disease log-odds for specific samples in Figure 21, i.e. the quantities $\left|f_j(x_j^+)\right|$ and $\left|f_j(x_j^-)\right|$, where x_j^+ denotes the value of the j-th feature of a positive sample, and analogously for x_j^-. These draw a relatively similar picture, also compared to previous models. Feature importances as measured thereby are quite consistent across the samples. However, features that strongly increase the log-odds of `HeartDisease` are, expectably, different than the ones decreasing it. For instance, the log-odds were mostly shaped by `Chol_NA = 1` and `ExerciseAngina = "Y"` for positive samples, but by `ChestPainType = "ATA"`, `Sex = "F"` for negative samples.

We would argue, however, that *changes* in the log-odds from changes in feature values, as shown in Figures 19 and 20, are more informative about feature importance than logit contributions from specific samples, as shown in Figure 21. This is because logit contributions are not centered. For example, $f_j(x_j = 0)$ may be very large, but since $f_j(x_j = 1)$ may be even much larger, this does not mean that $x_j = 0$ contributes positively to heart disease. Instead, it is the *change*, $f_j(x_j = 1) - f_j(x_j = 0) > 0$, which indicates that patients for which $x_j = 1$ are at an increased risk of heart disease compared to patients with $x_j = 0$. Further, importance for positive and negative samples should be equivalent, since a label being positive directly implies that it is not negative, and vice versa.

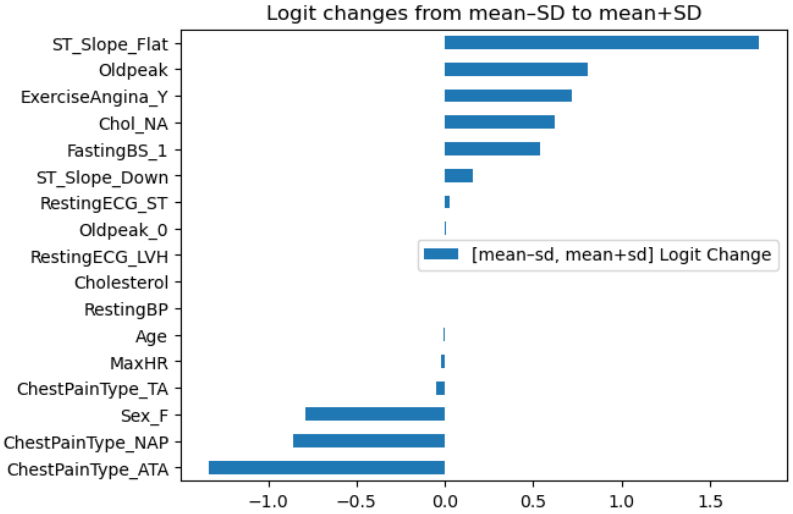


Figure 20: Change in the log-odds of `HeartDisease` from changing variables from 1 SD below average to 1 SD above average, i.e. $f_j(\bar{x}_j + \text{sd}(x_j)) - f(\bar{x}_j + \text{sd}(x_j))$.

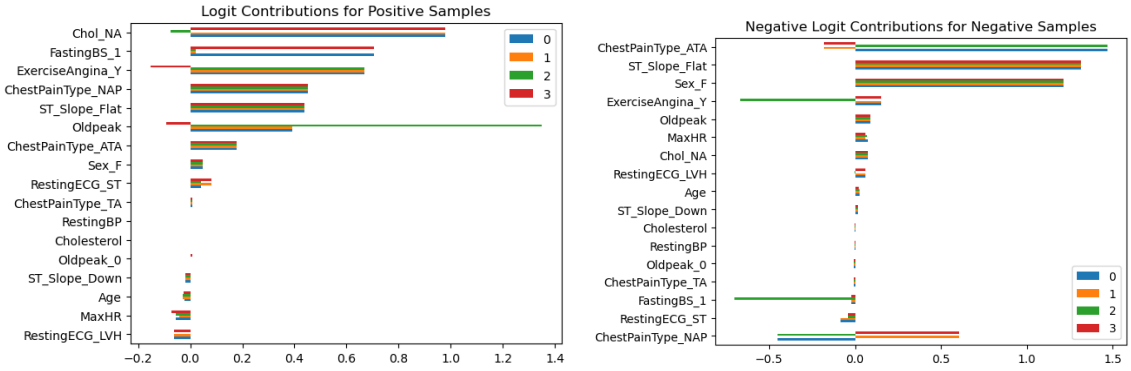


Figure 21: Contributions to the log-odds of `HeartDisease` of individual features for 4 positive (left) and 4 negative (right) samples, i.e. $f_j(x_j^+)$ and $f_j(x_j^-)$, where $x^+_{\texttt{HeartDisease}} = 1$ and $x^-_{\texttt{HeartDisease}} = 0$. Note that for the negative samples, the *negative* logit contributions are depicted.

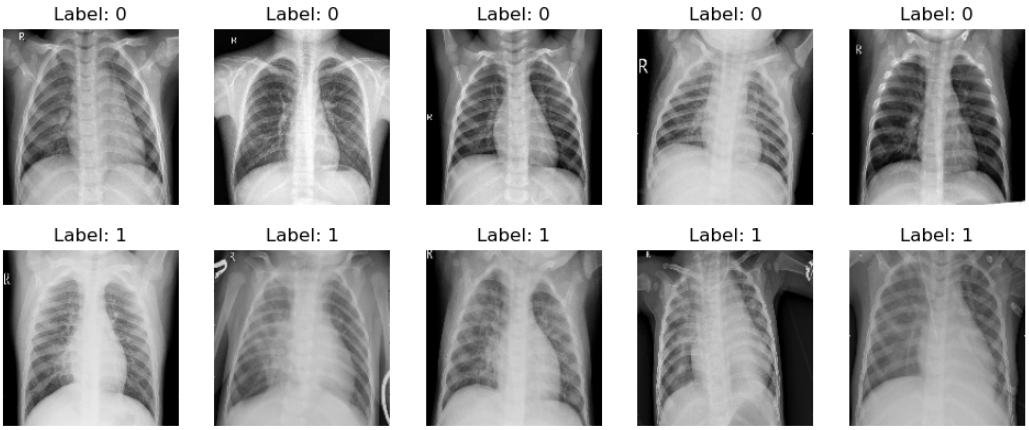


Figure 22: Raw images of healthy (above) and pneumonic (below) samples.

2 Pneumonia

2.1 Data Exploration and Preprocessing

This task explores the use of X-ray images to detect pneumonia and differentiate between viral and bacterial pneumonia.

Exploration

Figure 22 displays five raw images of healthy and pneumonic samples. There, we see that pneumonia samples tend to have more light colored parts in the chest/lung area.

The images also contain potential biases, such as brightness variations, the position of the X-ray image and external disturbances on the image, i.e. annotations and tubes used by doctors.

Lastly, we discovered that the dataset is heavily imbalanced, with significantly more pneumonia cases than normal cases, as illustrated by Figure 23. Additionally, there are more bacterial pneumonia cases than viral pneumonia cases.

Preprocessing

Since the initial train/validation split provided by the original dataset only features 16 validation samples for several thousand training samples, we chose to merge these datasets and split the combined set to an 80/20 training and evaluation split.

To address the imbalanced dataset and increase sample size, we appropriately applied data augmentation techniques, e.g. rotations, flips and shifts of the original image. We also used bias correction techniques, e.g. data normalization and padding, to reduce the impact of brightness variations and the position of the X-ray image.

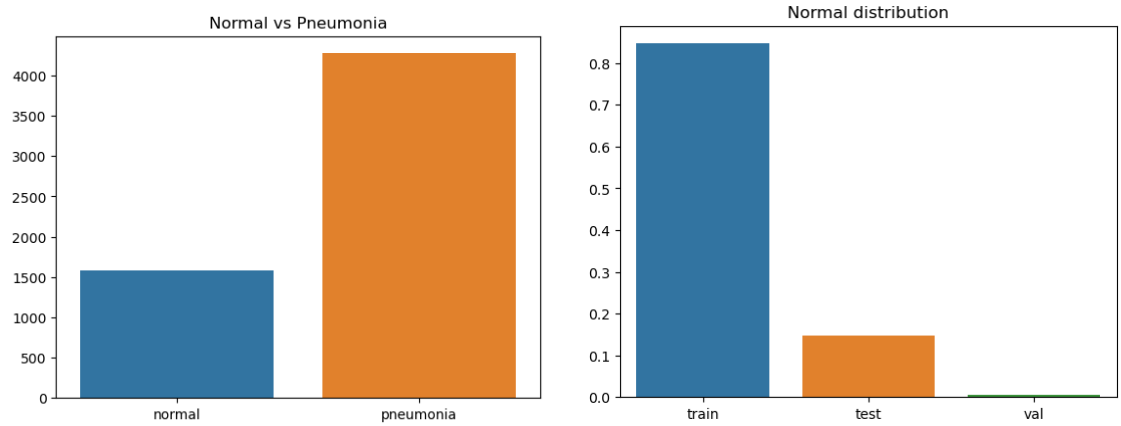


Figure 23: The distributions of the original dataset show that there is an imbalance between normal and pneumonia samples (left), while the splits between train, test and validation data demonstrates that most samples were assigned to the train split (right).

To further improve the results, one could have blended out the external disturbances mentioned previously, since they proved to be major disturbances in the later training (see Section 2.4).

2.2 Building a CNN classifier

In the following, we fit a total of two separate networks: a simple classifier and one that applies transfer learning on a pre-trained model. Initially, we built a small CNN based on `TensorFlow`. However, although it achieved satisfying accuracy, its attribution maps seemed to simply work as edge detectors.

In an attempt to improve expainability, we used a pre-trained `ResNet-50` network based on `PyTorch` and trained a few extra layers on our own dataset. While accuracy was comparable, the resulting attribution maps definitely seemed to focus on more reasonable regions of the pneumonia X-ray scans. This is probably due to the fact that the `ResNet-50` was trained on a much larger dataset (`ImageNet`) and thus has a much more complex architecture.

In the following, we will describe results for both models because this allows for an interesting comparison between two classifiers with inherently different architectures and model complexities. However, the focus for the explainability results of the following sections is on the `ResNet-50` based model, the reasons for this being elaborated on in Section 2.3.

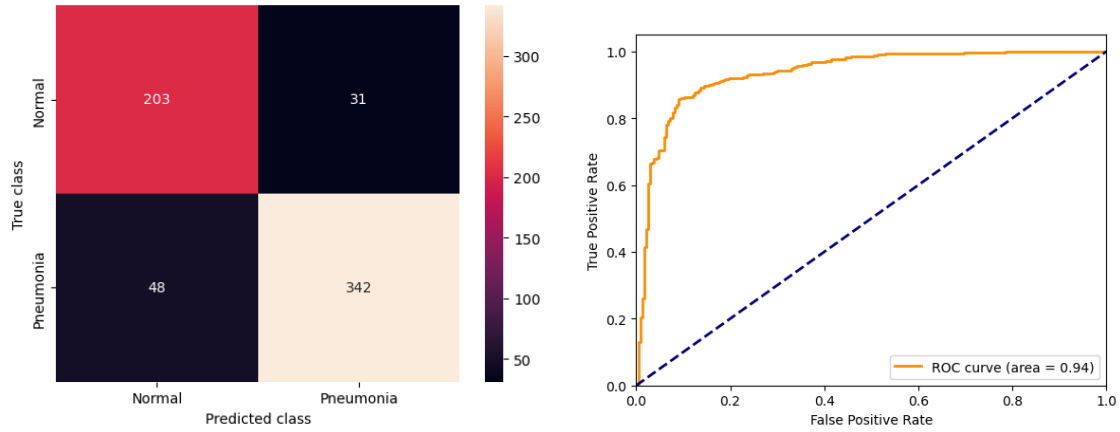


Figure 24: Confusion Matrix and ROC of the smaller CNN's predictions on the test dataset.

Architecture

The smaller CNN classifier consists of three convolutional layers, including BatchNormalization and Dropout. We used a pre-trained `ResNet-50`[3] model and replaced its fully connected layer at the top with (trainable) linear and ReLU layers, thereby invoking transfer learning on the pneumonia X-ray dataset.

Training Details

To improve the efficiency of training, we used learning rate reduction and early stopping based on validation loss as callbacks. Both models were initialized for 20 epochs, but training was stopped by the callbacks after around 10 epochs. We minimized the binary cross-entropy loss using the `Adam` optimizer with an initial learning rate of 0.001.

Performance

Accuracy on the test set ranged was 0.87 ± 0.03 for the smaller model and 0.86 ± 0.05 for the bigger model. The resulting AUC of the smaller model was 0.94 ± 0.02 and the bigger model 0.93 ± 0.04[4]. The performance metrics for the smaller CNN are displayed in Figure 24, while the metrics for the `ResNet-50` model are visualized in Figure 25. Since both models demonstrate an excellent performance on the test set, their results may be used for further analysis.

[3] https://pytorch.org/hub/nvidia_deeplearningexamples_resnet50/

[4] Both models were trained several times to ensure reproducibility, resulting in the mentioned performance metrics and confidence intervals.

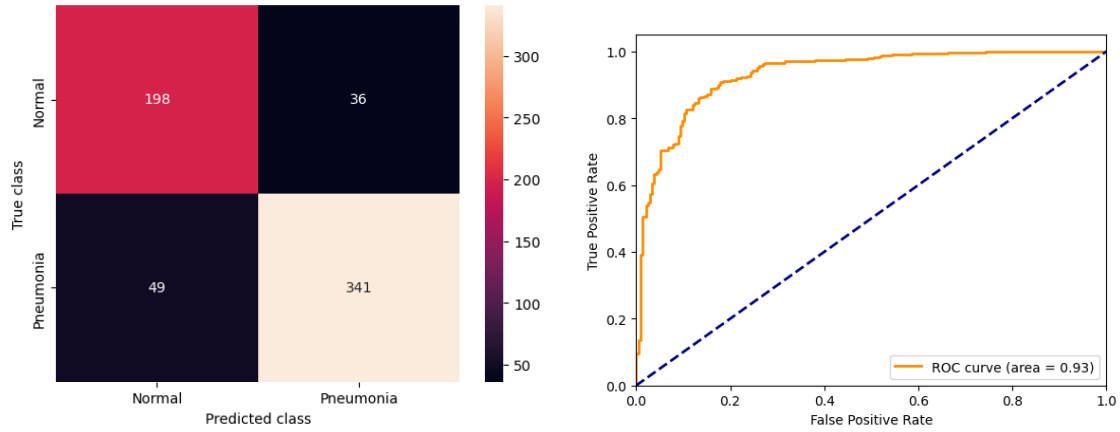


Figure 25: Confusion Matrix and ROC of the ResNet50-model's predictions on the test dataset.

2.3 Integrated Gradients for Explainable AI

Implementation

The integrated gradients method was implemented based on Sundararajan et al. (2017), thereby following the main formula described in the paper:

$$\text{IntegratedGrads}_i^{approx}(x) ::= (x_i - x_i') \times \sum_{k=1}^{m} \frac{\partial F(x' + \frac{k}{m} \times (x - x'))}{\partial x_i} \times \frac{1}{m} \tag{1}$$

For brevity we forgo a detailed explanation of this method and refer interested readers to the original paper. The main idea of this method is to explain the relationship between a model's predictions in terms of its features, by computing gradients of model output predictions with respect to input features.

The implementation for the two classifiers outlined in Section 2.2 differed slightly, due to the classifiers being built upon the `TensorFlow` and the `PyTorch` frameworks respectively. While we relied on `PyTorch`'s implementations of the integrated gradients attribution method for the small network, we implemented it manually for `TensorFlow` following one of the library's authors' tutorial for the large network[5].

Sensible Regions and Biases

We found that the integrated gradient method works well as an edge detector and for outlining the shape of areas that lead to its decision. For example, on pneumonic samples, the shape of the spine region was highlighted. As noted in Section 2.1, the spine region

[5] https://www.tensorflow.org/tutorials/interpretability/integrated_gradients

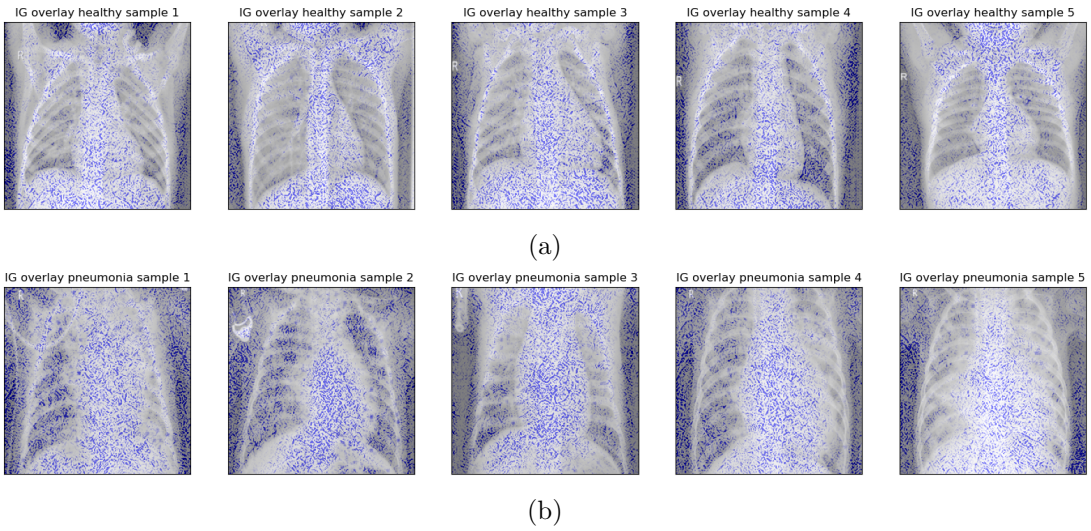


Figure 26: Integrated gradient attribution maps for healthy patients in (a) and patients featuring the disease in (b). These visualizations were generated from predictions by the `ResNet-50` model.

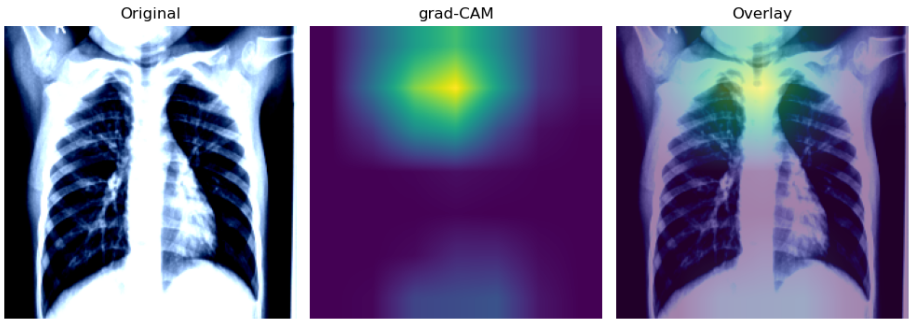
is indicative of the disease, since this region is usually "widened" for pneumonia samples[6]. Hence, such attributions are sensible.

Generally, integrated gradient attributions are consistent between samples and also classifiers. We also discovered that the human annotations and auxiliary devices (e.g. tubes) seem to disturb the classifier and its attribution maps are influenced by them. Also, other biases, i.e. the positioning of the body on the X-ray image, had an impact on the predictions, as displayed in the attribution maps in Figure 26. As mentioned previously, it would make sense to put an effort into removing them to improve accuracy as well as explainability of the classifier.

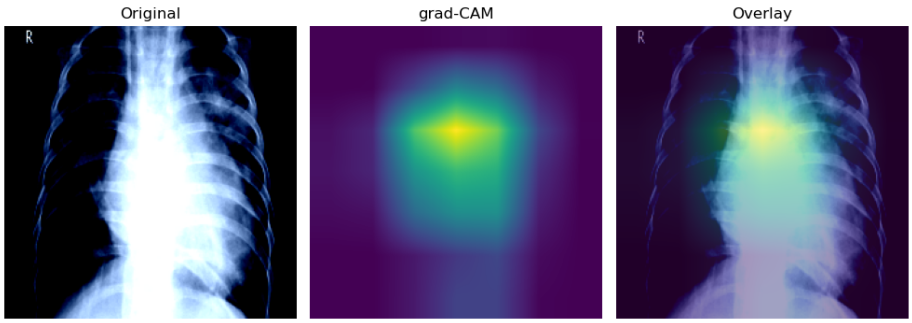
2.4 Grad-CAM for Explainable AI

Grad-CAM generates a heatmap indicating the regions of the image that contribute the most to the prediction made by the model (Selvaraju et al., 2017). Therefrom, we observed that, for healthy samples, the pre-trained model focused on areas outside the chest cavity, indicating the absence of abnormal white spots in the lung region (see Figure 27a). For pneumonic samples, however, the heatmap produced by Grad-CAM often highlighted areas within the lung region, particularly where lighter spots were visible, likely caused by viral or bacterial pneumonia (see Figure 27b). The results were, in general, consistent across samples and indicate a certain level of trustworthiness and robustness. However, as also

[6]This is due to inflammations of the air sacs in one or both lungs (see Section 3.4), but since the classifiers were not pre-trained with any medical knowledge they were probably unlikely to have learned such a detail.



(a)



(b)

Figure 27: Grad-CAM heatmaps for healthy patients in (a) and patients featuring the disease in (b). These visualizations were generated from predictions by the `ResNet-50` model.

previously observed, external disturbances in the form of tubes and annotations seemed to work as biases, which is demonstrated in Figure 28.

In contrast, we noted that the smaller CNN worked more like an edge detector when analyzing the Grad-CAM attributions, similar to its earlier layer attributions. These observations were consistent across many samples, but we noticed that some samples "learn random parts" or "learn biases" due to said external disturbances.

Comparing the Grad-CAM attribution method with the integrated gradients method discussed in Section 2.3, it appears that the latter approach finds more general patterns in a classifier's predictions and, additionally, is prone to noise. There exist, however, methods for filtering this noise and also for "smoothing" the integrated gradients, resulting in more refined attribution maps (Sundararajan et al., 2017). In contrast, Grad-CAM, especially for deeper neural networks, is able to discover more specific attributions. We find that the heatmaps generated by Grad-CAM allow a better interpretation of the model's final prediction, which is reasonable since they are produced directly from the last convolutional layer.

Overall, the sensible regions highlighted by Grad-CAM aligned with the predictions

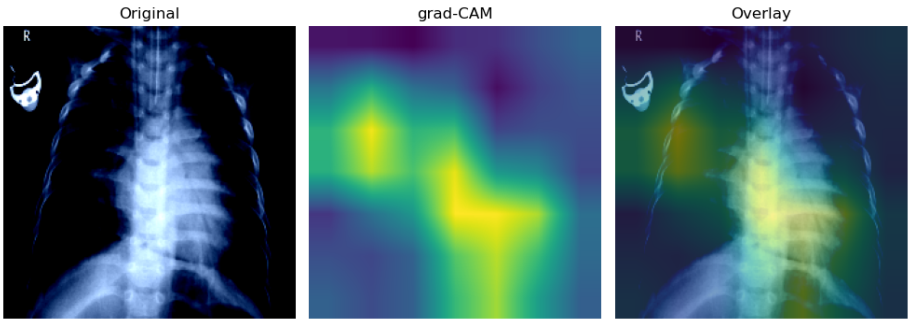


Figure 28: A Grad-CAM heatmap for a healthy patient that was disturbed by a tube that appears on the X-ray scan. This visualization was generated from predictions by the `ResNet-50` model.

made by the models, although a non-negligible part of the data featured certain biases that influenced the model's attribution maps. It would be beneficial to eliminate these biases in future studies.

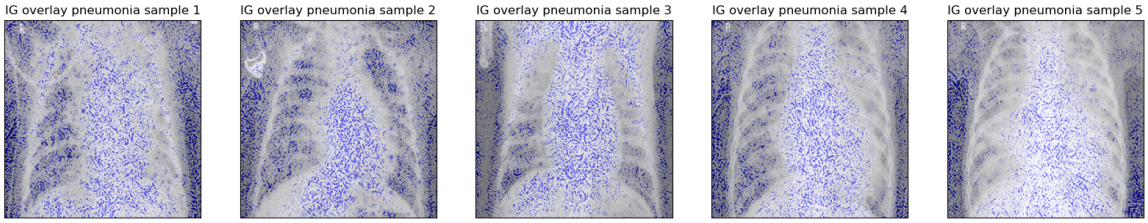
2.5 Data Randomization Test

The data randomization was performed by retraining both the small and large classifier (see Section 2.2) on the same test set, but with randomly permuted labels. This experiment was performed according to the original paper by Adebayo et al. (2018) and aims at identifying whether or not the classifiers exhibit the same heatmaps for the original and the randomly permuted labels. This allows to determine the trustworthiness of the saliency maps of specific methods.

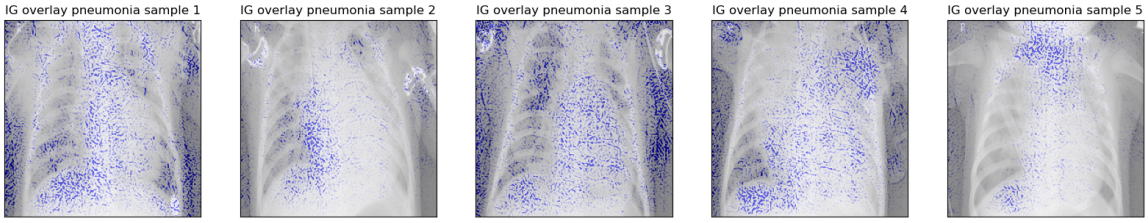
Integrated Gradients

As mentioned previously, the integrated gradients method seems to find more general trends as opposed to the Grad-CAM method, resulting in an edge detector for the smaller CNN model. This is consistent when retraining the classifier on random label and, thereby, passes the data randomization test.

Performing the same test for the integrated gradient saliency maps on the bigger model proved to be less successful. As can be inferred from Figure 29a, the classifier trained with the original labels appears to be somewhat like an edge detector, focusing mostly on the "light" areas around the "darker" lung region. The classifier trained with permuted labels, on the other hand, appears to focus on random points that do not seem to follow a reasonable pattern (see Figure 29b). Therefore, the integrated gradients method fails the data randomization test for the bigger model. These observations were all consistent across samples, with some minor exceptions where distortions, e.g. brightness or external tubes, disturbed the predictions.



(a)



(b)

Figure 29: Integrated gradient heatmaps for the classifier trained on the original labels in (a) and the classifier trained on the randomly permuted labels in (b). These visualizations were generated from predictions by the `ResNet-50` model.

Grad-CAM

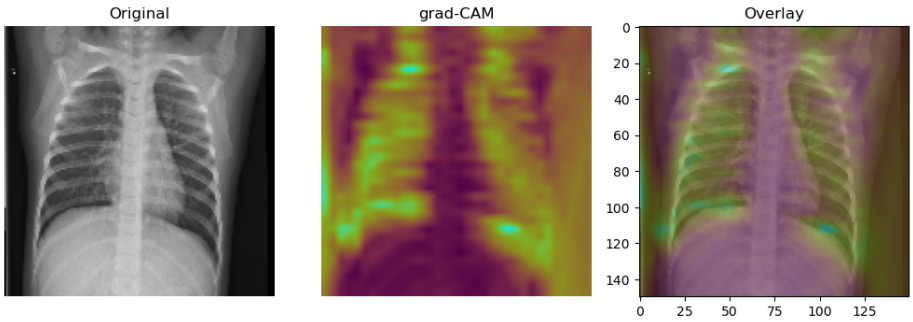
The results for the Grad-CAM method were even more interesting, seeing that the resulting heatmaps exhibited more specific focus areas in certain settings. Analzying the output for the smaller CNN model that is displayed in Figure 30, we note that the heatmaps generated from training on the original labels are focusing on the darker lung regions, while the heatmaps resulting from a random permutation of the labels appear as edge detectors. Therefore, Grad-CAM fails the data randomization test for the smaller CNN model.

Applying Grad-CAM to the pre-trained `ResNet-50` model, on the other hand, is successful on the data randomization test. While training the classifier on the original labels enables Grad-CAM to discover attributions that are located close to white spots in the lung region, the random permutation of labels causes the classifier to folow seemingly random patterns in its "learning" approach. Again, these conclusions are applicable across samples and, therefore, are deemed to be generalizable.

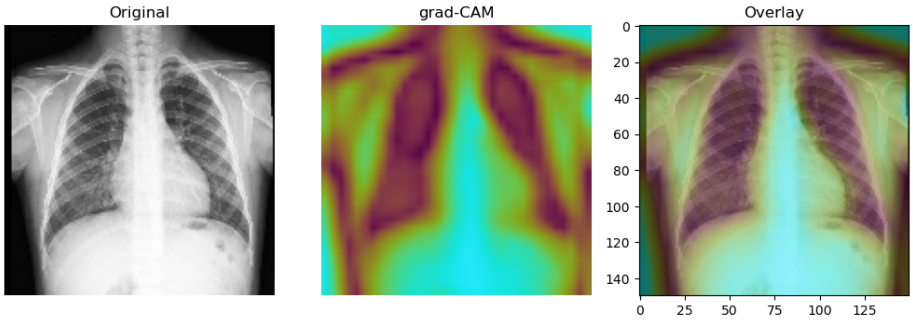
Challenge 2: Prototype Learning

In this section, we implemented the idea of prototype learning proposed in (Kim et al., 2016). Thereby, we first tried to construct the proposed classifier by following the description in the paper. Additionally, we explored the original codebase[7] of the paper's authors and modified it for our dataset. Since both approaches resulted in some interesting insights, we

[7] https://github.com/BeenKim/MMD-critic



(a)



(b)

Figure 30: Grad-CAM heatmaps for the classifier trained on the original labels in (a) and the classifier trained on the randomly permuted labels in (b). These visualizations were generated from predictions by the smaller CNN model.

deem them fit for comparison in this section.

Performance

First, we implemented the Nearest Neighbour classifier from the original paper, which proposed that for a test point the problem should reduce to finding the maximum kernel distance of our prototypes in set S to the test point $\hat{x}$: $arg\max_{i \in S} k(\hat{x}, x_i)$, then outputting the label of the found prototype. Testing this with a subset of random prototype points we achieved an accuracy of 0.375 ± 0.00. Surprisingly, this is significantly worse than even a random classifier. Looking at the codebase of the authors, they used a standard `sklearn` 1-nearest neighbour classifier. This was confusing for us, as it had nothing to do with the RBF kernel proposed in the paper. We decided to follow the authors' approach instead of our previous attempt.

Using a standard Nearest Neighbour Classifier `k = 1` with some randomly sampled prototypes, we obtained an accuracy of 0.7 ± 0.01 on the test set (which was a vast improvement). Implementing the loss function used by (Kim et al., 2016), whereby prototypes that are representative of the overall data distribution are selected, we refit the KNN Classifier with the selected prototypes. The resulting accuracy of 0.74 ± 0.03 is still subpar compared to the deep learning models presented in Section 2.2. But using this method we could visualise prototypical examples.

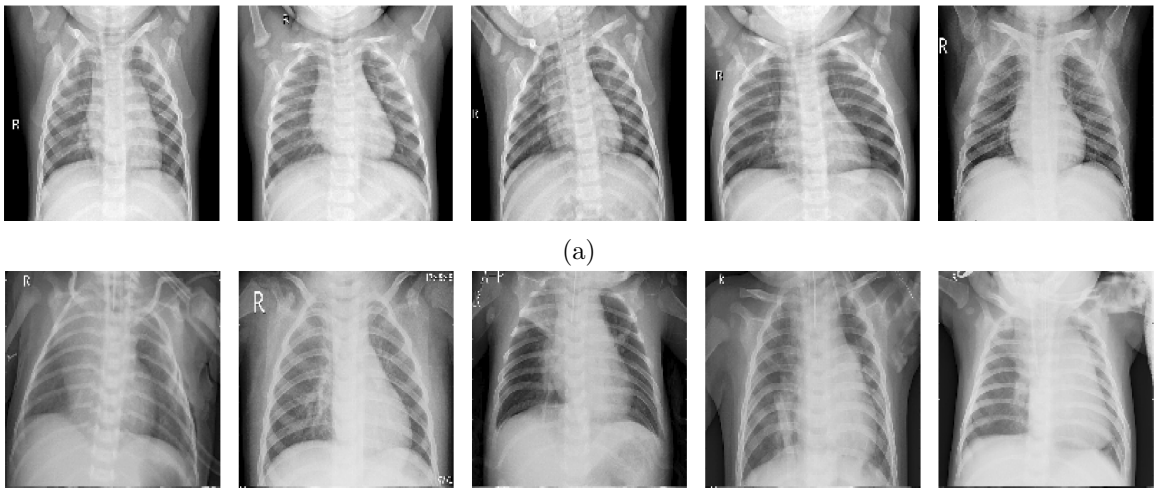
Finally, the re-implementation of the original code on the pneumonia X-ray scans resulted in an accuracy of 0.72 ± 0.02[8], which is slightly below the accuracy we achieved in our approach. The reason for this small discrepancy, besides randomness, is most likely a different preprocessing of the images or a slightly different implementation of the greedy algorithm for selecting prototypes.

Selecting prototypes and criticisms

Prototypes are selected by minimizing a sample approximation of the maximum mean discrepancy (MMD) between the prototypes' and the overall distribution. For our dataset, these are visualized in Figure 31, whereby a clear pattern among these samples becomes evident. These prototypes share the same approximate level of natural brightness, color saturation and are, additionally, almost perfectly centered and there is no apparent rotation.

We further selected criticism samples from the dataset, which are samples dissimilar to the prototypes. They are displayed in Figure 32). Comparing criticisms and prototypes, we note that the selected criticisms mostly consist of samples that are in a strange rotation or lack a certain degree of brightness that complicates a differentiation between light and dark regions. Including these samples in a dataset when fitting a classifier will most likely lead to a decreasing confidence in its predictions, which demonstrates the usefulness in discovering the most and the least representative samples in a data distribution.

[8] Reported as an "error" in the original code, which is simply 1 - accuracy.



(a)

(b)

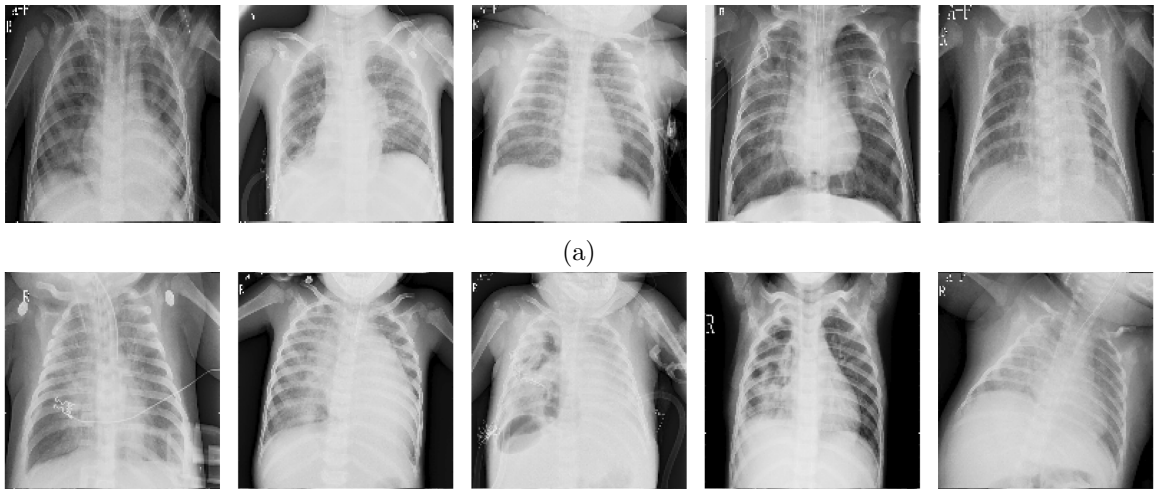
Figure 31: Selected prototypes by the greedy MMD algorithm for healthy samples in (a) and for disease samples in (b). These visualizations were generated from predictions by the fitted classifier from the original codebase.

Potential improvements

To improve performance, we see two options. First, as noted by the authors of the original paper, the choice of kernel function likely influences results. Depending on the task, dataset, etc., the most adequate similarity measure may be a different one and would have to be identified or determined from prior knowledge. Second, the base classifier could be extended. For instance, classification based on Gaussian processes is a more flexible method to model latent class probabilities and might be able to identify patterns that were missed by kNN, which can similarly account for different similarity measures. Alternatively, a latent representation generated by a CNN could be fed to the kNN classifier, which could improve accuracy. However, whether the greedy prototype selection algorithm would still be applicable under such modifications would have to be re-assessed.

Comparison to integrated gradients and Grad-CAM

Since the method outlined by Kim et al. (2016) is inherently different to the saliency maps discussed in Section 2.3 and Section 2.4, the usefulness of a specific method depends entirely on the specific use case and task setting. When aiming at a post-hoc explanation of a complex CNN model to convince a doctor of its value in analyzing medical datasets, applying either the integrated gradients or Grad-CAM attribution method is probably more reasonable. One reason is doctors can directly infer from the presented heatmaps whether or not the classifiers are actually learning sensible patterns that confirm current medical knowledge or are, unfortunately, taking a shortcut.



(a)

(b)

Figure 32: Selected criticisms by the MMD-critic algorithm for healthy samples in (a) and for disease samples in (b). These visualizations were generated from predictions by the fitted classifier from the original codebase.

We find that there also exist scenarios where prototype learning might be very useful. Specifically, this method exhibits a lot of potential in "boosting" the confidence of a classifier, being capable of identifying typical and atypical data points. Therefore, this approach could be applied on a dataset during the preprocessing phase. Prototypes and criticisms could then be further employed when training a deeper network, thereby likely improving its performance[9].

3 General Questions

3.1 Consistency of results

Heart Disease

The sets of features that the models primarily relied on for prediction are nearly congruent. For all of them, the five most important features included `ST_Slope`, `ChestPainType`, and `Sex`. Thereof, `ST_Slope` was always assigned the highest importance score. Hence, results for the heart disease dataset are consistent.

Pneumonia

Analyzing the resulting visualizations from the different attribution methods on the pneumonia X-ray scans, we note that the methods seem to recognize different patterns. While

[9]Note that this is simply a hypothesis and, to the best of our knowledge, was not confirmed yet.

the integrated gradient method is tracing the shape of the spine region around the lung[10], the Grad-CAM heatmaps discovered more sensible patterns, i.e. white patches inside the lung region that are strong indicators of pneumonia[11].

3.2 Explainability/Interpretability

Heart Disease

A very interpretable model is the tree (see Figure 11). One can follow the decision splits to an end node, then predict the majority class of that node. For instance, a patient for whom `ST_Slope` $\neq$ `"Up"`, `ChestPainType = "ASY"`, and `Oldpeak` $\leq$ `1.85` would be predicted to have heart disease.

Pneumonia

The Grad-CAM attribution method aims at a post-hoc explanation[12] of a Convolutional Neural Network (CNN), which is a deep Al model specialized on images. It is based on several (deep) layers that "learn" a computer representation of images and their features (shapes, color, etc.). Grad-CAM is constructed to analyze the last in the series of these convolutional layers, on which decision of the classifier is based. Grad-CAM then proceeds to generate a heatmap based on this analysis, which highlights regions on the image that contribute the most towards the final decision of the classifier.

3.3 Accuracy/Interpretability tradeoff

Heart Disease

Our models - ranging from a highly interpretable decision tree to a not directly interpretable neural network - all performed similarly well (accuracy: 0.82 ± 0.02; AUC: 0.89 ± 0.03). The performance of the fully connected MLP and the the less flexible NAM were virtually equivalent. Thus, we have no evidence of trade-offs between accuracy and interpretability for this dataset. Still, we suspect that if one had put pure emphasis on accuracy, e.g. by fitting a complex ensemble, one could have outperformed our set of models.

Pneumonia

Since we essentially only built a single model for the pneumonia data, we cannot make a direct comparison between more interpretable and more flexible classifiers here. However, since implemented both a smaller CNN and a pre-trained model, we could observe that the pre-trained model focused on more specific patterns, while the smaller CNN learned broader

[10] Note that a minority of the samples were able to trace white spots inside the darker lung regions as well.

[11] https://www.lung.org/lung-health-diseases/lung-disease-lookup/pneumonia/symptoms-and-diagnosis

[12] Interpretation of the classifiers is not possible due to black-box nature of CNN.

patterns[13]. Nevertheless, their accuracy on the test set was very comparable. Hence, the improvement in interpretability did not come at the cost of performance.

3.4 Alignment with medical knowledge

Heart Disease

Broadly, according to current knowledge, factors associated with heart disease which were included in our dataset are chest pain, exercise angina, and rapid and irregular heartbeat[14]. Risk factors for developing heart disease include older age, male sex, high blood pressure, and high cholesterol[15]. Thus, the features assigned high importance by our models are representative for, but not exhaustive of, the factors relevant according to domain knowledge.

Pneumonia

Analyzing the explainability results that were obtained during our experiments, it becomes evident that the models were most likely not complex enough, disturbed by certain biases, or otherwise influenced when deciding on which areas of the X-ray scans to focus on most. The closest results to current medical knowledge were achieved by the pre-trained `ResNet-50` model that, according to the Grad-CAM attribution maps, mostly focused on white spots or patches that appear in the darker background of one or both lungs[16]. These spots indicate inflammations of the air sacs ("alveoli") and are usually analyzed by doctors to determine the severity for existence of pneumonia and to differentiate between viral and bacterial cases of the disease in a patient.

3.5 Deployment in practice

Heart Disease

We would choose the decision tree when working with practitioners due to its interpretability. It allows to make a prediction by answering 3 simple Yes/No questions about the patient. Hence, clinicians can easily check the validity of the procedure with which the algorithm makes its predictions. Empirically, accuracy does not suffer excessively compared to more involved approaches.

Pneumonia

Since the different methods can be implemented in an efficient manner, we would probably compare several attribution methods in a real-life scenario in case of inconsistencies. If,

[13] This is not surprising, since the difference in complexity between the models is significant, allowing the `ResNet-50` based classifier to learn more complex and specific patterns.

[14] https://www.mayoclinic.org/diseases-conditions/heart-failure/symptoms-causes/syc-20373142

[15] https://www.mayoclinic.org/diseases-conditions/heart-disease/symptoms-causes/syc-20353118

[16] https://healthmatch.io/pneumonia/pneumonia-chest-xray#what-you-need-to-know-about-pneumonia

however, we had to choose a single method, we would go for Grad-CAM, since generating saliency maps directly from the last convolutional layer leverages the learnings of the model over several layers and is able to outline specific areas that lead to the models decision. Additionally, we note that, as mentioned before, applying the prototype learning method during the preprocessing phase in conjunction with a deeper network might improve the confidence of the classifier and, therefore, result in more refined attribution maps. We propose for interested readers to further explore this idea and will certainly keep it in the back of our minds for future experiments.

References

Adebayo, J., J. Gilmer, M. Muelly, I. Goodfellow, M. Hardt, and B. Kim (2018). Sanity checks for saliency maps. In S. Bengio, H. Wallach, H. Larochelle, K. Grauman, N. Cesa-Bianchi, and R. Garnett (Eds.), *Advances in Neural Information Processing Systems*, Volume 31. Curran Associates, Inc.

Bühlmann, P. (2021). High-Dimensional Statistics. https://stat.ethz.ch/~buhlmann/teaching/highdim-stats-HS2021.html.

Kim, B., R. Khanna, and O. O. Koyejo (2016). Examples are not enough, learn to criticize! criticism for interpretability. *Advances in neural information processing systems 29*.

Kübler, R. (2017). Interpretable Neural Networks with PyTorch. https://towardsdatascience.com/interpretable-neural-networks-with-pytorch-76f1c31260fe.

Lundberg, S. M. and S.-I. Lee (2017). A unified approach to interpreting model predictions. In I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett (Eds.), *Advances in Neural Information Processing Systems 30*, pp. 4765–4774. Curran Associates, Inc.

Meinshausen, N. (2007). Relaxed Lasso. *Computational Statistics & Data Analysis 52*(1), 474–393.

Meinshausen, N. and P. Bühlmann (2010). Stability Selection. *Journal of the Royal Statistical Society 72*, 417–473.

Meinshausen, N., L. Meier, and P. Bühlmann (2009). p-Values for High-Dimensional Regression. *Computational Statistics & Data Analysis 104*(488), 1671–1681.

Selvaraju, R. R., M. Cogswell, A. Das, R. Vedantam, D. Parikh, and D. Batra (2017, Oct). Grad-cam: Visual explanations from deep networks via gradient-based localization. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*.

Sundararajan, M., A. Taly, and Q. Yan (2017). Axiomatic attribution for deep networks. In *International conference on machine learning*, pp. 3319–3328. PMLR.

Zou, H. (2006). The Adaptive Lasso and its Oracle Properties. *Journal of the American Statistical Association 101*(476), 1418–1429.