Interpretable and Automated Bias Detection for AI in Healthcare

by

Christopher Alexiev

BASc, University of Toronto (2022)

Submitted to the Department of Electrical Engineering and Computer Science
in partial fulfillment of the requirements for the degree of

MASTER OF SCIENCE

at the

MASSACHUSETTS INSTITUTE OF TECHNOLOGY

September 2024

Authored by: Christopher Alexiev
Department of Electrical Engineering and Computer Science
August 21, 2024

Certified by: Regina Barzilay
Professor of Electrical Engineering and Computer Science
Thesis Supervisor

Accepted by: Leslie A. Kolodziejski
Professor of Electrical Engineering and Computer Science
Chair, Department Committe on Graduate Students

Interpretable and Automated Bias Detection for AI in Healthcare

by

Christopher Alexiev

Submitted to the Department of Electrical Engineering and Computer Science
on August 21, 2024 in partial fulfillment of the requirements for the degree of

MASTER OF SCIENCE

ABSTRACT

Biases in artificial intelligence systems and the data they operate over are a major hurdle to their application in clinical and biomedical settings. Such systems have frequently been shown to fail to generalize from their training data to the real world environment and often display differing levels of accuracy over different population subgroups, which has detrimental effects on patients' quality of care and on healthcare equality. Here, we introduce an automated framework for identifying and understanding nontrivial sources of bias in healthcare datasets and AI models. Our framework is data and model agnostic and does not rely on human-developed heuristics or assumptions to uncover bias. We demonstrate its effectiveness by uncovering serious and nontrivial sources of bias in three widely used clinical datasets and one biomedical dataset, over the diverse tasks of diabetes risk prediction, lung cancer risk prediction, and biomolecular toxicity prediction. Our framework is used to uncover biases caused by patient BMI and computed tomography (CT) scanner type in the data used by a cutting-edge lung cancer risk prediction AI model, causing AUC drops on the order of ten percent.

Thesis supervisor: Regina Barzilay
Title: Professor of Electrical Engineering and Computer Science

Acknowledgments

I would first like to acknowledge and thank my collaborators, MIT postdoctoral fellow Dr. Shrooq Alsenan and MIT undergrad Evan Rubel, for their contributions to this project. Shrooq helped organize the project with me and spearheaded the diabetes experiments, obtaining the diabetes data and performing much of the initial analysis with this data. Evan Rubel helped perform a significant amount of software engineering that enabled the efficient execution of the finalized versions of experiments, especially for cleaning and working with the diabetes data.

Next, I would like to thank the members of the Regina Barzilay Group whose work I am building off of. Recent graduate Dr. Yujia Bao developed the Learning to Split algorithm, which forms the bedrock upon which my project is built. I further thank him for his help and advice regarding the implementation and use of his algorithm. Current students Peter G Mikhael and Jeremy Wohlwend are responsible for creating the Sybil AI model for lung cancer risk prediction, which served as the primary testbed for my experiments. I much appreciate all the help they have provided regarding the implementation of Sybil, as well as the patience with which they treated me, despite my constant questions. I would also like to lend a shout-out to Aziz Ayed, current student in the lab, whom I bounced many ideas off of regarding the application of AI in healthcare.

I would also like to acknowledge Dr. Florian Fintelmann and Dr. Judit Simon from the Department of Radiology - Division of Thoracic Imaging and Intervention, Massachusetts General Hospital. Many thanks to them for all the knowledge they shared about CT imaging of the lungs and the available lung cancer datasets. We intend on continuing to collaborate with them for further validation of our findings.

Finally, I would like to thank my advisor, Dr. Regina Barzilay, who took a chance on me and welcomed me to the lab. She has helped provide direction to my research, enabling me to succeed, and never fails to provide clarity that the purpose of this work is to improve the lives of others. Additionally, I thank my parents, grandparents, and brother for all of their love and support, without which I would not be achieving this milestone. Most of all, I would like to thank the three samoyeds with whom I share an office, Momo, Sabre, and Yumi, for their endless cuddles and emotional support through the graduate student journey.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

AI is very promising for use in clinical settings, achieving better-than-human performance on various medical tasks [1], [2]. However, although AI models can achieve good average performance on many tasks, their use thus far remains limited in the real-world clinical setting. They can exhibit poor performance over certain subgroups of data, exacerbating biases already present in the healthcare system. Similarly, they can exhibit unexpected failure cases, causing unwanted false negative diagnoses that a human doctor would not make, with dire results for patient care [1]–[7]. Current methods for detecting the causes of these biases are labour-intensive, ineffective, and not interpretable to clinicians and AI practitioners. In order to enable the widespread adoption of AI in healthcare, we need a better way to uncover the causes of these biases in the data used to train AI models. This would enable the correction of bias in the data generation process, more informed use of AI models, and the reduction of biases in AI models. In this thesis, we investigate the implementation of such a bias detection framework.

1.1 Background

1.1.1 Defining Bias and Fairness

In this work, we broadly consider an AI model to be biased or unfair if it exhibits different levels of predictive performance over different subgroups within the data it is used on [8]–[11]. It is worth considering how this definition of bias fits into a broader rich literature on the notion of fairness in AI applications, with many different qualitatively and quantitatively defined definitions of bias and fairness. Broadly speaking, research on AI fairness in healthcare has focused on two main definitions of fairness, these being *group fairness* and *minimax fairness* [4], [5], [9], [10], [12], [13].

In group fairness, the goal is to create an AI model that performs equivalently over different patient subpopulations. One type of group fairness is *conditional prediction parity*, in which the model outputs a positive diagnosis with equal probability over different patient subpopulations; this is rarely useful in most contexts but can promote equal allocation of medical resources to different patient populations [4]. Two more common varieties of group fairness are *equal opportunity*, which aims for equal true positive rates across patient sub-

populations, and *equalized odds*, which strives for equal true positive and false positive rates across subpopulations [4], [13]. Many more variations of group fairness exist, such as to produce equivalent ROC curves [4] or calibration performance [4] across subpopulations. While common, optimizing for the group fairness approach often results in AI models that perform worse over all subpopulations [5], [12] because certain subpopulations may be intrinsically more difficult for a model to achieve a high performance over.

In minimax fairness, the goal is to maximize the performance of the AI model over its worst performing subpopulation [5], [12]. This approach aims to improve each subpopulation's performance separately instead of striving for equal performance. As with group fairness, there are different variations of minimax fairness, which can variously aim to optimize the model's loss function, accuracy, calibration, or ROC AUC over different subpopulations [5], [12]. While also commonly used in fair-ML, the minimiax fairness objective rarely leads to better AI models when using the latest de-biasing approaches [5], [12] and instead often leads to tradeoffs with other performance metrics such as classification precision [10].

In addition to these definitions, there are many other proposed variants of fairness, such as *individual fairness*, in which a model behaves similarly over similar patients, or *counterfactual fairness*, in which a model behaves the same even if a patient's non-pathological characteristics are changed in a counterfactual manner to an alternate demographic subgroup [4], [11]. There is ultimately no fully authoritative definition of fairness, with different formulations providing different benefits and meeting different equity related goals. Furthermore, different definitions of fairness are often incompatible, a mathematically proven notion known as the *impossibility theorems* [8], [11], [12], [14]. For example, it is usually impossible to simultaneously achieve both demographic parity and equalized odds. Similarly, it is usually impossible to achieve a well-calibrated AI model while simultaneously achieving equalized odds. Considering the incompatibility of different fairness definitions and the tradeoffs between them, it is apparent that there is no one definition that can be applied in all healthcare contexts, with ethical assumptions required to determine which causal pathways in the decision making can be considered fair [9].

1.1.2 Causes of Bias in Medical AI

Hidden stratification is the term describing the failure of an AI model over a clinically relevant subgroup in the data [5], [6], [15]. This can occur for several reasons [6], [9], [15]: (1) Data imbalances: When a certain subgroup, label, or data attribute is uncommon in the training data, such as a rare but dangerous cancer, an AI model may fail to learn to accommodate it. (2) *Label/annotation bias*: Inaccurate and noisy training labels for a given subpopulation can prevent an AI system from learning true patterns in the data. This can be a way that human biases creep into AI models, as human radiologists who label training data may themselves be biased or less careful on certain groups of radiology scans [2], [10]. (3) Inherent hardness or presentation disparities: Certain subpopulations may be inherently difficult to learn, for example because they look very similar to other subpopulations of data that tend to have a different label. Similarly, certain diseases may manifest differently in different subgroups of patients or through different imaging modalities. (4) Shortcut learning: AI models can be tricked by spurious correlations, in which certain data points have features that are correlated but not causal to a disease [2], [6], [7], [13], [15]–[18]. For example, if a

certain image background is correlated with disease, the AI model may fail to predict images presenting disease but without that background. Similarly, if a disease is associated with old age, a model may learn to rely on age instead of pathology when making predictions. (5) Finally, shifts in the distribution of different subpopulations from training to testing settings can make an AI model appear to do well on average over its training data, but then fail in deployment over certain groups.

Yang et al [10] identifies several such types of train-test *subpopulation shifts* that result in biased AI models and are often found in varying ratios in different datasets. One such subpopulation shift is class imbalance, in which the dataset classes (i.e. positive or negative for cancer) are found in different ratios in the test and train datasets, which can lead to low model confidence on classes less prevalent in the training data. Two more such subpopulation shifts are attribute imbalance and attribute generalization, in which subgroups with certain attributes (such as ethnicity) are present in different ratios in the test and train datasets, or entirely lacking in the training data, in the case of attribute generalization. This can lead to poor performance on subgroups with attributes that are underrepresented in the training data. Spurious correlations can also be considered a form a subpopulation shift when the spurious feature is common in the training data but lacking in deployment.

Biases can creep into AI models' training data from many sources. For example, data is typically collected from different hospitals and institutions independently and without enough engagement of developers during the data collection process. Thus, procedures, equipment, and standards vary significantly [9], [16], [19]. Even within the same hospital, inpatient departments and emergency rooms may use different equipment [19]. These factors all lead to data collection procedures rife with spurious correlations, subpopulation shift, and data labeling errors, leading to hidden stratification of model performance over different subsets of the data. Furthermore, different hospital systems and datasets may cater to different demographics of patients, corresponding to populations that are historically underrepresented and underserved by healthcare, which can then lead to the perpetuation and amplification of societal biases in clinical AI systems [12].

1.1.3 Clinical Examples of Bias

Countless studies investigating in a research setting have demonstrated the failure of common AI model architectures over subgroups stratified by various protected attributes such as race, gender, age, or intersectional combinations of these. This can occur for various reasons, such as due to dataset imbalances, inherent hardness, or shortcut learning in which an AI model learns to rely excessively on demographics instead of pathology when making decisions. Seyyed-Kalantari et al [8] demonstrate that common computer vision AI models over three large chest X-ray datasets underdiagnosed traditionally underserved populations at a higher rate, especially intersectional populations such as Hispanic female patients. Zhang et al [12] similarly demonstrate significantly stratified performance on chest X-ray prediction over sex, race, and age over two major chest X-ray dataset. Yang et al [10], [14] show stratification of performance over the intersection of race, sex, and age on the common MIMIC-CXR chest X-ray dataset and 5 other chest X-ray datasets, and they show how unfairness is generally correlated with the extent to which an AI model encodes protected attributes in its internal feature representation. They also show how stronger such encodings make the

models less able to generalize to external test data with a distribution shift. Brown et al [13] further demonstrate how models use shortcut learning to rely on a patient's age when making predictions over various dermatological and radiological datasets [13]. Finally, Pfohl et al [4] demonstrate big fairness gaps using standard AI model training techniques to predict patients' clinical outcomes in a large set of experiments involving 25 combinations of clinical outcomes, protected demographics, and health record datasets.

Regarding inherent hardness, many real world clinical datasets also have subsets of data that are inherently harder to predict, even subgroups unrelated to protected attributes. For example, Sohoni et al [15] demonstrates this also over the ISIC skin cancer dataset, showing that models they train fail over a subset of images that is more difficult for humans to visually classify correctly and for which doctors requested a biopsy be performed. In another example, Oakden-Rayner et al [6], with the help of human radiologists, demonstrate that models trained on the Adelaide Hip Fracture dataset have low performance over the subset of fractures that are subtle or in the cervical area, and show that models trained on the MURA muskoskeletal dataset to classify tissue as normal or abnormal display a stratification of AUC curves over different subsets of the abnormal data. Fractures are more successfully flagged as abnormal as opposed to subtle degenerative diseases.

Regarding label biases, Obermeyer et al [20] demonstrate how a patient risk prediction algorithm widely used in the real world by United States healthcare providers was found to be plagued by racial bias, significantly reducing the number of Black patients identified as requiring care. They find that this is because the algorithm was trained using healthcare spending as a proxy label for care needs, and society spends significantly less money than required on Black patients compared to white patients. This is an unfortunate example of how societal biases can work their way into algorithms in the form of noisy labels and label bias. Another way label biases can be introduced into AI models is through the common practice of automatic data labelling [8]. For example, the CheXpert[21] labeller, which converts radiologist's notes on X-ray scans into a positive or negative disease label, has been found to perform poorly, especially over certain age groups, by Zhang et al [12]. Unsurprisingly, they find that training AI models on these automatically generated labels leads to bias, with especially poor performance over age groups for which the automatic labeller fails.

Another example in which biases reached the real-world, this time caused by shortcut learning, would be the Moleanalyzer-Pro neural network, which had been approved for medical use in Europe. It was designed to distinguish between benign skin nevi and cancerous cutaneous melanomas. However, in the clinic, suspicious lesions are often marked by doctors using gentian violet skin markings. Winkler et al [22] show that skin markings significantly interfered with the AI's performance by increasing the melanoma probability scores and consequently the false-positive rate. Some more examples of spurious correlations and shortcut learning being demonstrated in the literature are presented as follows:

- Varoquax et al [2] demonstrate how academic goals (such as creating an AI model that performs well on a common benchmarking dataset) are not always aligned with medical goals (such as translating the AI model to a hospital setting), which results in very few AI models developed that can can be feasibly used in the real world. In their review of dataset biases, they mention how a range of factors ranging from image texture

to patient demographics results in biased experimental AI models in diverse medical fields, including chest radiology, retinal imaging, brain imaging, histopathology, and dermatology. They mention how one of the most concerning types of biases that can occur is when a model spuriously learns to classify a treatment as a disease, such as an example is demonstrated over a previously published AI model predicts the pneumothorax collapsed lung condition. They find that the model learns to predict the condition based on whether there is a chest drain in the image, a type of treatment for the condition [6]. They find that the AUC on samples without chest drains is low.

- Zech et al [19] demonstrate that CNNs used for predicting pneumonia based on chest X rays are able to identify the hospital the image is taken at and use the prevalence of pneumonia at that hospital as a confounding variable in their predictions, instead of relying on disease pathology. The most glaring example of this is when the models recognize metallic tokens that certain hospital systems use in the images as laterality labels.

- Although there are many published papers about AI models that predict Covid-19 based on chest X-rays, there is much variation in the training data used for these, with some using low quality images and some using pediatric datasets. Thus, Ahmed et al [16] recreate similar models and demonstrate how they fail catastrophically when tested on real world data. They demonstrate that the models failed to learn Covid-19 pathology, instead relying on image artifacts from non-lung areas of the image, as well as the rib cage size, which is correlated with Covid-19 because few children have Covid-19 in the training data.

- Sohoni et al [15] further demonstrate potential biases in the commonly used ISIC skin cancer dataset. They find that a brightly colored patch in the image is correlated with benign skin markings, which then acts as a spurious correlation causing the model to fail on images without the patch.

1.1.4 Uncovering Biased Subgroups

Various methods can be used to determine sources of bias in a dataset, but difficulties remain and more research is still warranted in this area [6]. The most common method is simply to segregate datasets into different groups informed by human expertise and heuristics, and separately test the model on each group. The process of stress testing an AI model like this over different manually curated groups goes by various names, such as *stratified performance evaluation stress testing* [3], *schema completion* [6], or generalization testing [18]. The vast majority of bias examples outlined above were discovered in this way, and Oakden-Rayner et al [6] demonstrate how real-world clinicians can participate in the process by manually labelling training set samples into different pathology subgroups to test model performance over. Seyyed-Kalantari et al [8] test AI models for their underdiagnosis rates over different protected attributes such as age, race, sex, and socioeconomic status, including intersectional combinations of these attributes. Their study over three large-scale chest X-ray datasets successfully uncovered that underrepresented populations, especially intersectional underrepresented populations, tended to be significantly underdiagnosed by the AI models.

It is well known that AI models' internal feature representations can learn to encode a patient's demographics or other attributes, which raises the concern that they are using such attributes as spurious correlations in disease prediction [13], [14], [23], [24]. As such, another method that can be used to check for human-expert-informed biases is to investigate whether the AI model's internal feature representation can be used to predict whether a patient belongs to a suspected biased subgroup [23]. However, Brown et al [13] demonstrate that simply encoding an attribute such as age does not mean that the model is using it unfairly as a shortcut. As such, they developed an algorithm named ShorT that can confirm whether a suspected demographic attribute is being used unfairly. Their method entails modifying the degree to which a certain data attribute is encoded by the AI model's internal representation, and then investigating how the fairness of the model changes in return. They successfully demonstrate their method, detecting spurious correlations in several clinical datasets. However, they warn that not all biases are caused by spurious correlations, demonstrating a dermatological AI model biased against certain age groups, but for which age does not get flagged as a spurious correlation using their method.

Another manual method for discovering biases is *error auditing*, which involves investigating the portions of the testing data over which a model performs poorly to see if there are any common patterns in these portions that may be causing bias. This method can pick up on unexpected spurious correlations, as demonstrated in [6], where a radiologist was manually employed to investigate the falsely predicted images and found that AI model correctness correlated with the presence of the disease treatment in the image.

An extension to the idea of using a model's mistakes to find biases is presented by Bao et al [25], who propose the PI algorithm, first splitting a dataset into different "environments", such as different hospitals where data is sourced from, then training separate AI models for each environment, and then considering the groups of correctly and incorrectly predicted points in each environment as stratified biased subgroups. One challenge with this, however, is that the environments to choose are not immediately obvious. Various algorithms further automate the process of using a model's mistakes as a guide for detecting biases. One such algorithm termed *environment inference* by Creager et al [26] involves using a fixed AI model's classification outputs as a learning signal to discover subgroups of data over which the model may become reliant on spurious features.

Another algorithmic method which can be used to discover stratification in the dataset is clustering. Each data point is represented differently in the internal feature space of an AI model. Clustering algorithms such as Gaussian Mixture Models can be used to group together similar portions of the internal representation space, and it has been shown that these similar subgroups can be representative of biases such as spurious correlations, underrepresented groups, and difficult to predict subsets in some clinical datasets [6]. Sohoni et al [15] demonstrate a clustering approach that uncovers a set of skin cancer dataset images that is more inherently difficult for both doctors and AI models to classify correctly.

While the above methods and algorithms that investigate the mistakes of an AI model over its training data can uncover some biases, they don't consider certain factors that can contribute to an AI model's mistakes, such as the model's representation power. If an AI model is too simple, it may not perform well, even on simple unbiased data subsets. To isolate true causes of bias instead of errors caused by limited model representation power, another algorithm called Learning to Split (LS) [17] identifies training and testing subgroups

such that a model trained on the training group does not generalize well to the testing group. It works by adversarially pitting a splitting function that splits data into difficult training and testing sets versus a predicting AI model that tries to do well despite the difficult splits. LS shows higher performance than the above methods at uncovering the spurious correlation subgroups in common benchmarking datsets for correcting bias.

None of the bias detection techniques discussed thus far are very interpretable. Thus, an alternative technique that has been developed to make analyzing an AI model's decisions more transparent is creating gradient heat maps that help highlight the portions of an image that are used by a model to make decisions. Winkler et al [22] use such techniques to confirm that an AI model is relying on sections of the image containing clinical skin markings instead of sections of pathological significance.

1.1.5 Overcoming Biases over Known Subgroups

If one knows which subsets of data result in stratification of model performance, then there are various steps model designers can take to mitigate the impact of biases. The most effective way to prevent a model's biases from affecting healthcare delivery is simply to abstain from using the model in totality or at least over subsets of patients that the model performs poorly over. Sometimes, this may be the most responsible approach until a better model is produced [4]. Given the variable effectiveness of algorithmic de-biasing methods, the first course of action for creating a better AI model should always be to minimize the cause of bias at the source [4], [5], [12], [14]. This could mean changing the training objective, relabelling data more accurately, collecting more data for underrepresented demographics, or abstaining from training and deploying the model on subsets of data with known spurious correlations, among many more possible courses of action.

If all reasonable efforts have been made to improve the data collection procedures, but there are still biases found in the model performances, then there are several de-biasing algorithms that can be employed to enforce fairness constraints in the model training. Here we describe several classes of algorithms that attempt to achieve this when the biased subgroups in the data are known.

The first class of algorithms we identify focuses on reducing a model's reliance on a known spurious correlation:

- Some of these algorithms are ensembling methods, such as in [27] and [28]. These methods first create a biased model that makes predictions based on the known spurious correlation, and then involves training the final model in an ensemble, summed together with the biased model, so that it learns the patterns not already captured by the biased model.

- A procedure called deep feature reweighting (DFR) [29] can be performed, which entails taking a trained but biased neural network and then retraining only its last layer on a held-out dataset which does not contain the known spurious correlation. This works on the premise that the neural network's learned internal feature representations are of high quality and the bias mainly affects the last layer, which seems to usually be the case.

- Some adversarial methods such as DANN [30] and CDANN [31] aim to retrain the internal feature representations of the AI model in order to eliminate any encoding of subgroup attributes and thus prevent the model from relying on spurious correlations with such attributes. They use an adversary network trained to recognize the subgroup from the internal representation, and then train the internal representation to make it harder for the adversary.

A second class of algorithms try to train AI models that uphold various group fairness constraints over a well defined set of data subgroups.

- One such method, which we term Fairness Violation Penalization, involves modifying the AI training objective function to penalize violation of the demographic parity, equalized odds, or equal opportunity fairness constraints between desired groups. Pfohl et al [4] investigate several such modifications over clinical datasets with mixed success, finding that increases in a specific fairness constraint were often offset by poor calibration and AUC for many of the subgroups. A related method, REx [32], penalizes variance in the expected value of the objective function over different subgroups.

- Another set of such methods devise AI training objectives based off of constrained Lagrangian optimization problems. Invariant Risk Minimization (IRM) [33] constrains the model to rely on features that are invariant to the subgroup, and FairALM [34] constrains the model to the equal opportunity constraint directly.

- Another set of methods called imbalanced learning methods strive for fairness by reweighting [35] or resampling [12] training data points to better represent uncommon groups in AI training. Sometimes [36], only the last layer of the neural network is re-trained on the group-balanced dataset. Relatedly, importance weighting [37] weighs different data points according to their relative distributions in the training and deployment scenarios.

Finally, instead of striving for equal performance across groups, one may attempt to focus more on the minimax fairness definition, with more of a focus on optimizing the model separately for each subgroup. A simple approach to this is to simply train a different AI model for each subgroup [7], [12]. Another method proposed with promising results over clinical data in [38] selects the subgroup with the worst ROCAUC at each step of training and then trains the model to improve over that group with a custom loss function. A related and more elaborate way of optimizing for minimax fairness is to use the Group DRO [39] optimization algorithm, which upweights poorly performing groups during each step of training; this algorithm is often the standard de-biasing approach, and is very popular in fair AI literature.

1.1.6 Overcoming Biases over Unknown Subgroups

Finally, there exist several methods that aim to help reduce hidden stratification of model performance in cases where the sources of bias or biased subgroups are unknown. The most general set of such methods don't focus on any specific hidden subgroup of data but rather aim to make the AI model generally more robust to biases caused by noisy ground-truth

labels and imbalanced training sets with rare classes of data such as rare diseases, so called *long-tailed distribution class imbalance* [40]. Some such methods include the following:

- Imbalanced learning methods that reweight or resample different label classes (but not arbitrary biased subgroups). One such method is BSoftmax [40], which adapts the Softmax activation function typically used in neural networks to be more unbiased in a long-tail setting and learns an optimal sampling rate for each label class. Another example is presented by Ren et al [41], where an optimal reweighting of different label classes is learned during training, instead of different sampling rates.

- Data augmentation techniques have also been proposed, such as Mixup [42], which aims to improve AI robustness against noisy data by creating extra training data points that are interpolations of real points.

A more general set of de-biasing methods follows the principle of first uncovering biased subgroups and then using these subgroups with a subgroup-aware de-biasing algorithm, a method demonstrated by the *environment inference for invariant learning* (EIIL) framework outlined in [26]. In this framework, a set of "environments" or subgroups are first identified (the EI phase) which can result in hidden stratification; this corresponds to the algorithms described here in section 1.1.4. These biased subgroups are then fed into a subgroup-aware de-biasing algorithm (the IL phase), such as IRM [33] or Group DRO [39]; this corresponds to the algorithms described in section 1.1.5. Several similar procedures have been published using this principle; for example, Creager et al [26] match their own environment inference algorithm with Group DRO and IRM. Sohoni et al [15] match clustering methods for bias detection with Group DRO, in a framework they call George. Bao et al have shown success using their PI algorithm to detect biases followed by Group DRO [25], and even more success with their LS algorithm followed by Group DRO [17].

Finally, several algorithms exist that are designed to train unbiased AI models by simultaneously finding biases in the training data and correcting for them. These tend to be methods that align with the minimax definition of fairness:

- One set of methods trains an unbiased AI model to learn from the mistakes of a biased one. One such method named Just Train Twice (JTT) [43] first trains an AI model for only a few iterations, then trains a second model in which data mispredicted by the first model is up-weighted; this is essentially a framework for using error-auditing to de-bias an AI model. Another method named Learning from Failure LfF [44] defines a training objective function which can be used to simultaneously train two AI models, amplifying the bias in one of them and learning to correct for these amplified biases in the other.

- A series of methods use ensembling, similarly to the subgroup-aware ensembling methods in 1.1.5, except that the biased model is created without prior human knowledge of the bias. Sanh et al [45] demonstrate this by setting the biased model to be a "weak" AI model that is too simple to learn complicated patterns in the data.

- Adversarial methods such as Adversarially Reweighted Learning (ARL) [46] frame de-biasing as an adversarial game between a model that is being trained to find difficult

data points and a predictive AI model trained to do well on the difficult points. ARL has similarities to Learning to Split, except that it focuses on re-weighting points in the training dataset, whereas LS focuses on curating new training and testing sets over which the predictive model fails to generalize.

- CVaR DRO is a more general version of Group DRO, which focuses on upweighting data points (instead of defined subgroups of data) that the AI model performs poorly over during each step of training.

1.2 Motivation

De-biasing algorithms are no silver bullet, understanding and eliminating the biases in data is the most important step. When choosing an algorithmic method to overcome biases, there are many tradeoffs that the different options entail, and one must be conscious that there is no silver bullet to fixing biases, whether the biased subgroups are known or unknown. First, one must consider the clinical goals, the types of biases present, and the type of fairness desired, since different de-biasing methods focus on different definitions of fairness and different biases, as outlined in Section 1.1. Second, one must be aware that de-biasing algorithms often fail to improve model performance, especially on out-of-distribution data, and different algorithms may perform differently over different datasets. Finally, one must consider the impossibility theorems, which imply that correcting biases by optimizing for one approach to fairness may result in trade-offs in other performance metrics.

The level to which de-biasing algorithms are helpful varies greatly and depends on the specific circumstances. Several major benchmarking studies have been carried out to directly compare the performance of different de-biasing algorithms over clinical datasets. Zhang et al [12] compares nine de-biasing algorithms (both group-aware and subgroup-unaware) over two chest X-ray datasets. They found some success with group-aware imbalanced learning methods, but showed tradeoffs in fairness and performance over all subgroups. Pfohl et al conduct two benchmarking studies for group-aware de-biasing algorithms in the clinical setting; one focuses on group fairness, comparing several variants of Fairness Violation Penalization [4]; another focuses on minimax fairness, comparing several variants of Group DRO with several other approaches [5]. Neither study uncovered consistent gains in fairness or worst-subgroup performance, instead frequently uncovering losses in performance of various metrics. Yang et al conduct two benchmarking studies. One of these compares 19 de-biasing algorithms (both subgroup-aware and subgroup-unaware) over 12 datasets, including several clinical datasets [10]; the other one compares five group-aware algorithms over medical datasets to see how well they correct fairness in test settings outside of the training distribution [14]. They occasionally find some some limited gains in fairness using Group DRO, DANN, and subgroup-aware imbalanced learning methods, but note that such gains come with tradeoffs. Furthermore, they find that these gains only hold when the AI model is tested on data that matches its training distribution, but the gains in fairness are not robust to shifts in data from training to testing, a common occurrence in healthcare datasets. In general, there is much variance in the de-biasing algorithms' performance over different datasets and different biased subgroups. This variance occurs both within and between studies, with no clear

algorithm consistently working the best.

In addition to the inconsistent performance of de-biasing algorithms, one must also consider how optimizing for one notion of fairness may result in trade-offs in other performance metrics or notions of fairness. Regarding minimax fairness, [10] empirically demonstrates the tradeoffs between worst-group accuracy and other performance metrics such as worst-case precision. Specifically over clinical data, [12] demonstrates over a chest X-ray dataset that optimizing for minimax fairness using JTT and ARL seems to yield AI models with worse performance and calibration, and using Group DRO does not yield meaningful gains in performance either. Regarding group fairness objectives, [12] finds that any increases in group fairness achieved using FairALM or Fairness Violation Penalization comes at the expense of worse performance over all subgroups of data. Similarly, [4] empirically demonstrates over a clinical dataset the tradeoffs between demographic parity and equalized opportunity, AUROC, and precision. They also generally find that penalizing differences in performance across the patient subgroups results in degraded performance over many of these metrics throughout the subgroups.

Considering the variance in de-biasing algorithm performance and the tradeoffs between optimizing for fairness and other metrics, a common point concluded and argued by all of these prior works is that in order to achieve equitable and reliable delivery of healthcare across different settings and patient subpopulations, it is crucial to understand the fundamental sources of bias inherent in the data, create improved data collection processes to mitigate the biases before training AI models, and consider the societal context affecting healthcare delivery.

If using de-biasing algorithms, knowledge of the biased subgroups is important. Firstly, de-biasing algorithms that require subgroup knowledge tend to perform better than those that do not. Secondly, having knowledge about the type of bias enables one to make an informed decision about which type of de-biasing algorithm to choose. Finally, having knowledge of which subgroups underperform allows one to check that biases have been corrected over these groups.

In studies that have compared subgroup-aware and subgroup-unaware de-biasing algorithms, subgroup-aware algorithms tend to do much better at increasing performance over known biased subgroups. For example, in [10], Group DRO and group-aware imbalanced learning methods perform the best, with DFR also showing some benefit, and group-unaware methods such as JTT or general robustness methods performing more poorly. In [12], the group-unaware methods JTT and ARL seem to decrease performance over a chest X-ray dataset. For increasing model fairness on out-of-distribution datasets, [14] found that methods that remove encodings of biased attributes from AI models' internal feature representations seem to work the best, with DANN slightly outperforming even Group DRO. Overall, all of the most consistently effective de-biasing algorithms rely on knowledge of biased subgroups and aim to mitigate the AI model's reliance on these biased attributes. Furthermore, even if group-unaware algorithms do not require bias annotations in training data, they usually rely on knowledge of the biased subgroups during the model validation phase in order to see which model best reduces the bias and to select the optimal training hyperparameters. As such, the results published in papers describing such algorithms, such as JTT

and the originally published EIIL algorithm [26], are overly optimistic because they assume bias knowledge during the model validation phase. Yang et al [10] demonstrate that the effectiveness of group-unaware de-biasing algorithms is further reduced when knowledge of biases is not known in the validation phase.

When using de-biasing algorithms, it is also important to understand which type of bias is present because this affects which types of de-biasing algorithms are most effective [9]. For example, most of the algorithms tested in [10] tend to work better at correcting biases caused by spurious correlations and class imbalance, but less well at correcting biases caused by attribute imbalance. Furthermore, Group DRO is not very effective at minimizing biases caused by inherently difficult portions of data [15], in which certain subgroups may have inherently unpredictable outcomes. Similarly, if biases are caused by noisy and incorrect training data labels, then certain de-biasing methods focused on such biases, such as Mixup, may be more appropriate. In fact, de-biasing algorithms are generally better suited for correcting biases caused by data imbalances, as opposed to label biases or biases caused by inherent hardness [9]. In any case, though, the best course of action would be to understand and prevent the cause of bias during the data generation phase.

Considering the variable effectiveness of different algorithms over different datasets and different types of biases, the optimal de-biasing strategy is likely to involve trying different algorithms and seeing which one most improves fairness for the biased subgroups while minimally affecting overall performance. In order to perform such validation and testing, a detailed understanding of which data comprises the biased subgroups is required.

Existing methods for uncovering biased subsets of data are not sufficient for use in clinical AI. First, bias detection methods that rely on human expertise are very labour intensive and can confirm the existence of only a limited scope of biases. Second, most automated bias detection methods have variable performance and are not guaranteed to uncover biases. Finally, even if biased subgroups are uncovered by existing automated methods, they are not usually interpretable to humans, making it hard for clinicians and algorithms designers to mitigate biases.

Bias-detection methods that rely on human experts have major limitations. For example, while stratified performance evaluations can be effective, the biases they can uncover are limited to the subset of biases that humans could have expected and have the time to test, which is usually limited to protected attributes. Furthermore, the method requires a lot of time and energy, either to manually curate subgroups or to test all of them. The cost of performing such an evaluation over intersections of protected attributes, which can have many permutations, can be prohibitive; thus, studies that have explored intersectional subgroups [8], [14] have only explored a limited subset of such groups. This means that such a bias detection method can further marginalize intersectional groups that are not well represented by common attributes for assessing fairness [4]. Furthermore, while the other major manual method of error auditing can be less labor intensive than manually curating and testing subgroups for stratified performance evaluations, it is still an intensive process and is dependent on the auditors' ability to recognize patterns in misclassified samples. Furthermore, it can have limited sensitivity for detecting certain biases, especially low-prevalance and highly discordant biased groups [6].

Meanwhile, algorithmic methods for de-biasing, such as clustering, have been rarely tested over clinical datasets, and it is not clear how consistently effective most of them are. Oakden-Rayner et al [6] show that such methods can uncover known biases in datasets in some cases but not in others, with their efficacy potentially limited by how separable the biased features are in the AI model's representation space. This shows that they could have some use but it may be limited and still require human guidance. Additionally, methods that find or correct biases by learning from a model's mistakes on the training set, such as environment inference, ARL, JTT, or group-unaware de-biasing algorithms that use ensembling methods, suffer from the same efficacy concerns as manual error auditing. Specifically, they may have limited sensitivity to low-prevalence biases and don't consider certain factors besides biases that can contribute to an AI model's mistakes in its training data, such as the model's representation power. Furthermore, a major limitation of many of the publications proposing bias-detection methods and group-unaware de-biasing methods is that they are largely tested over standard de-biasing benchmark datasets like Waterbirds, CelebA, and MultiNLI. These datasets have very well known and annotated spurious correlations that are unlikely to be so clean in practice, leading to benchmark chasing and potentially poor performance in practical clinical applications [2] where biases are less well-defined [9]. Most publications that detect, analyze, and correct biases in clinical data have used manual methods such as stratified performance evaluation to confirm the existence of already known or suspected biases stratified by protected attributes such as gender or race. The few publications that analyze automated bias detection methods over clinical data tend to focus on uncovering previously known biased subgroups, such as lung X-rays with spurious chest drains [6] or skin cancer images that contained spurious brightly coloured clinical markings or were difficult for doctors to diagnose [15]. There has yet to be a proposed algorithmic method demonstrated to help reliably and interpretably uncover unknown biases in real clinical data.

Finally, bias detection methods proposed to date are not easily interpretable and don't elucidate the exact cause of bias in the data in terms of clinical, pathological, or demographic features, making it hard for clinicians to trust an AI model. Bias detection methods that rely on clustering or investigating a model's mistakes can uncover hidden stratification and subgroups of data over which a model might fail. While these subgroups can be fed directly into de-biasing algorithms to help mitigate stratification of model performance, these subgroups are not usually readily interpretable by a human, making it hard to trust the de-biasing process and hard to correct biases in the data source. In order to understand the sources of bias, humans must manually investigate these biased subgroups, often with the labour intensive help of doctors [6]. Similarly, group-unaware de-biasing methods can be used to simultaneously find biases and de-bias algorithms, but this process does not elucidate the bias to the algorithm developer in an interpretable way. Finally, more interpretable bias-detection methods that use heat maps to analyze the locations of biases in images are only useful in limited contexts. While they can be useful for confirming suspected spurious correlations and can be helpful when used to analyze the subgroups isolated by other bias detection methods, they are of limited use to detect biases on their own because they do not help narrow down which data points in a dataset are biased. Also, AI models' decisions are often made using many portions of the input image, so not all types of biases will be obviously visible via these gradient heat map techniques. As it stands, there is no bias detection method or framework that is designed to detect and elucidate biases in terms of clinical,

demographic, or or other attributes in a way that is automated and less labour intensive.

1.3 Proposed Interpretable Subgroup Detection

As discussed in Section 1.2, there is a need for a bias detection framework for AI in healthcare that operates in an automated way and elucidates biases in an interpretable way. In order to fill this gap, we propose a two part bias detection framework which first uncovers biased subgroups using the Learning to Split bias detection algorithm, and then uses interpretable AI models to understand the defining features of these subgroups. This is demonstrated at a high level in Figure 1.1.

We select Learning to Split as the foundation of our bias-detection framework because it holds much promise as an effective bias detection tool. Firstly, it overcomes some of the issues that help limit the effectiveness of other algorithmic bias-detection methods. As described in Section 1.2, bias-detection methods that investigate an AI model's mistakes don't consider a model's representation power and can have limited sensitivity to low-prevalence biases. Similarly, methods that rely on clustering are limited by how separable the biased features are in the AI model's internal feature representation space. LS minimizes the effects of these factors because it (a) relies on increasing the generalization gap in performance between the training and testing datasets, instead of within just the training set, and (b) it does not rely on the internal feature representations of the AI model. Secondly, LS is chosen because it also considers a data point's annotated label, so it can identify a variety of different types biases including label bias, unlike some methods such as clustering; Bao et al [17] demonstrates how LS can effectively isolate noisy labels in its automatically selected subgroups. Finally, LS is chosen because it has shown promise as a tool to identify subgroups that can be fed to de-biasing algorithms such as Group DRO, even in settings without bias annotations in the validation data points. Bao et al [17] show that when feeding the biased groups identified by LS into Group DRO, the resulting de-biasing process is more effective than using using subgroups identified by other methods such as LfF, JTT, or the originally published EIIL algorithm [26], especially when biases are completely unknown including in the validation data. While Sohoni et al [15] do show that the performance of an AI model over subgroups created via clustering techniques could potentially be an effective proxy for model validation, the published results for LS combined with Group DRO still seem to be better than for the clustering methods proposed in [15] for de-biasing common benchmarking de-biasing datasets.

Once biased subgroups are detected by LS, the second step of our proposed framework involves interpreting the causes of bias in terms of clinical, demographic, or other attributes. We propose to use interpretable AI algorithms—which are designed to make their decisions easily understandable by humans—to achieve this in a way that is automated and less labour intensive, and to help catch causes of bias that could be missed when analyzing the biased subgroups manually. The neural network that is used to split the data into biased subgroups in LS is itself not directly interpretable, but training AI models to mimic its splitting decisions can elucidate the causes of bias to clinicians and AI practitioners. The interpretable model can be trained to identify biases caused by many factors, such as patient demographics, clinical attributes, image metadata such as imaging equipment type, or patterns within the

data itself.

Some obvious choices for the interpretable models include decision trees and logistic regression. Decision trees are very easy for humans to understand, visualizing the decision process as a flowchart over attributes of the data point. Logistic regression assigns a coefficient to each attribute, indicating how strongly that attribute belongs to one of the biased subgroups. Despite their simplicity, their straightforward interpretation makes these algorithms useful. Another potentially useful type of interpretable model would be rationale models [47]. These models highlight which attributes of the data (referred to as *rationales*) are most useful in making predictions. As such, they can be used as interpretable models in which the highlighted rationales reveal the biased data attributes. Invariant rationale models [48] further aim to ensure that the selected rationales are actual causal to the decision making process instead of just correlated with it. For example, a true cause of bias may be a specific type of imaging device data is taken with, but the hospital system the data is sourced from is likely to be correlated with the imaging device type. Overall, by using and comparing the outputs of multiple different interpretable models one can further check for consistency and also gain a more comprehensive understanding of the biases in the data.

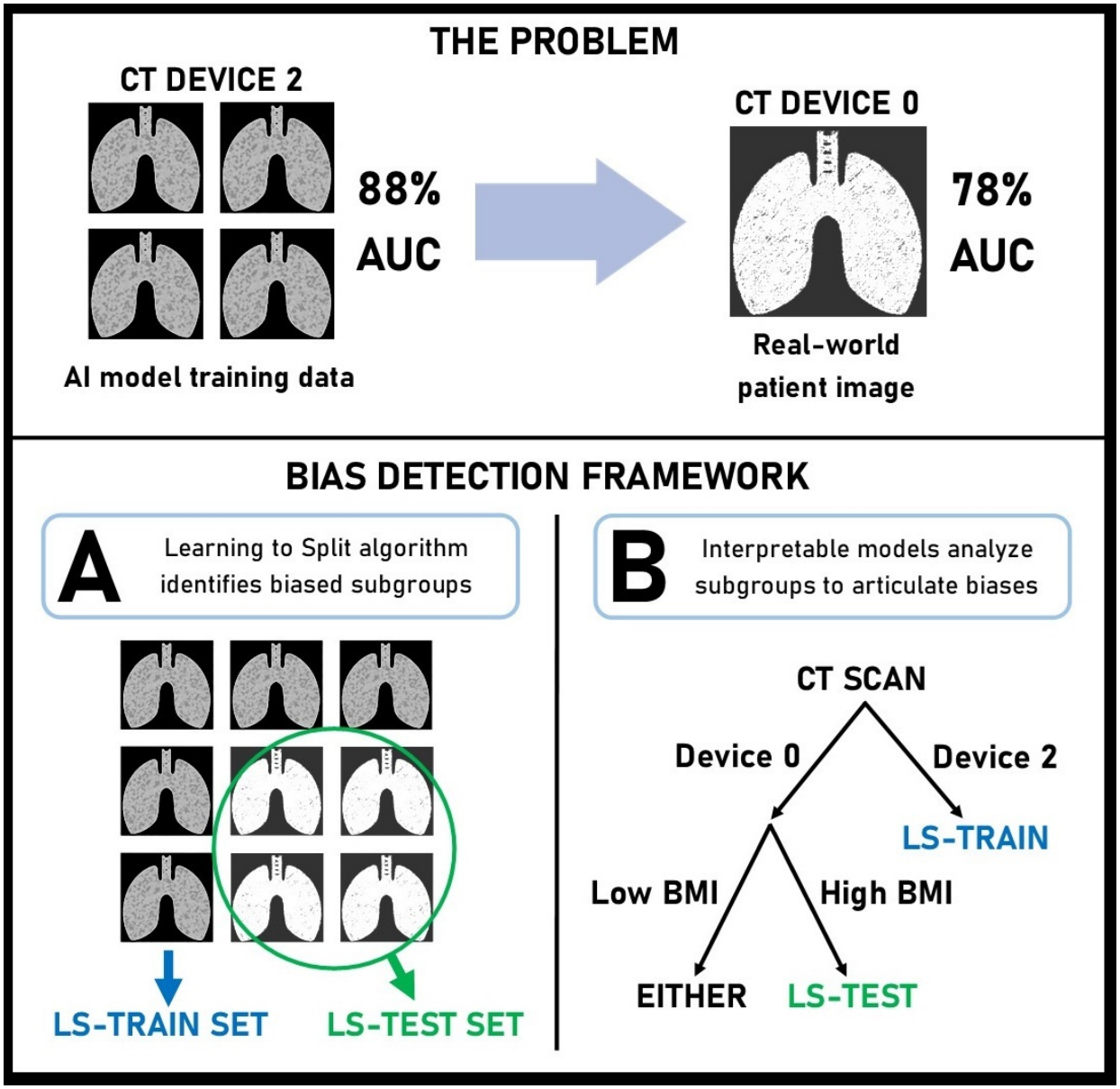


Figure 1.1: Overview of the bias detection framework.

Chapter 2

Methods and Experiments

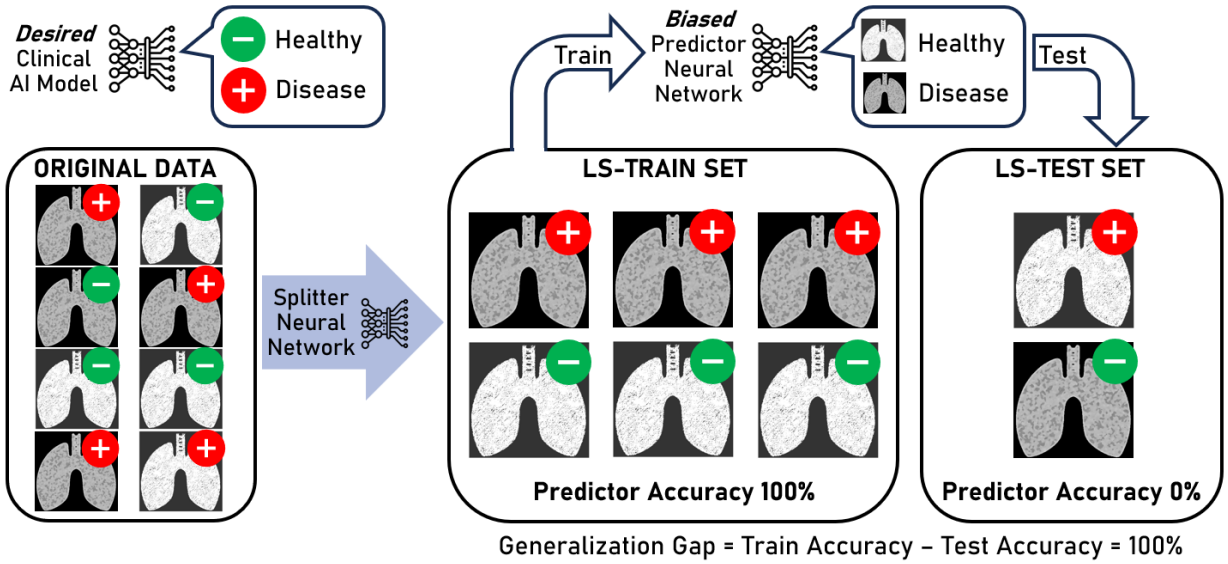
2.1 Overview

2.1.1 Overview of Framework

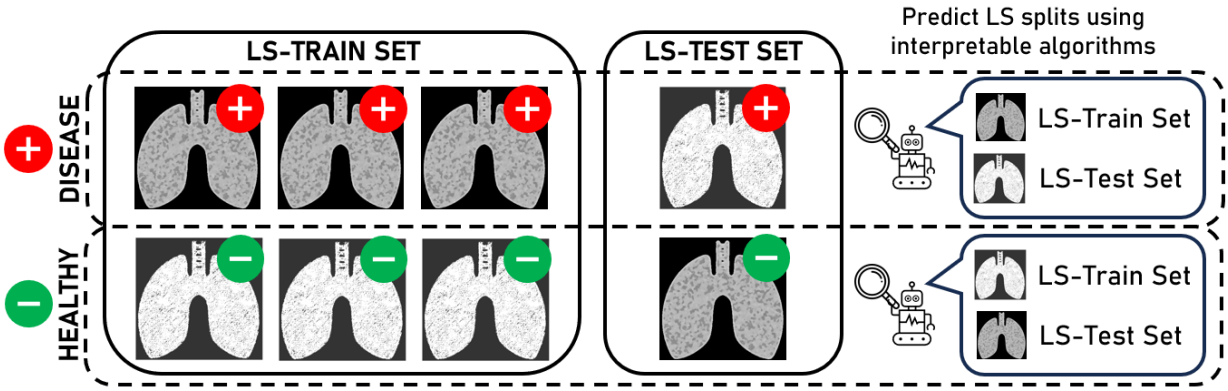
Bias Identification Framework Our bias identification framework, illustrated in Figure 1.1, comprises two primary components: First, the Learning to Split (LS) algorithm devised Bao et al [17] autonomously detects biased subsets within a dataset. Second, we employ interpretable AI algorithms to understand the reasons behind the inclusion of certain data points in these biased subsets. Once sources of biases have been identified, clinicians and algorithm developers can be alerted to take steps to improve data collection, improve the AI model, or mitigate the negative effects of the bias.

Learning to Split LS is designed to detect potentially biased subsets of the data by segregating the data into an *LS-train* set and an *LS-test* set such that an AI model trained on the LS-train set achieves low performance, or fails to generalize, on the LS-test set. It employs two neural networks: a *predictor*, with the same architecture as the clinically relevant AI model, and a *splitter*. The predictor's role is to predict the label (i.e. diseased or healthy) associated with a given data point (i.e. an X-ray scan of the lungs), while the splitter takes a data point and its label as inputs, predicting whether the data point belongs in the LS-train or LS-test set.

The LS algorithm operates in a loop, with the splitter and the predictor engaged in an adversarial game. At the start of each round, the splitter partitions the dataset into LS-Train and LS-Test splits. A new predictor is then trained over 67% of the LS-Train split, with the remaining portion used to calculate the predictor's AUC over the LS-Train split. Then, the AUC over the LS-Test set is calculated, and the difference between the predictor's AUC on the LS-Train and LS-Test sets is called the *generalization gap*. The splitter is then trained to increase the generalization gap, enabling it to create a more challenging split for the next round. LS terminates the loop once the generalization gap no longer increases. This process is illustrated in Figure 2.1a and is described in detail by Bao et al in [17].



(a) Overview of the LS Algorithm.



(b) How the LS Splits are analyzed using interpretable algorithms.

Figure 2.1: (a) How LS works and (b) how we use interpretable algorithms.

Interpreting LS Splits Once LS has established the LS-train and LS-test splits, the second step in our framework is to use interpretable algorithms to elucidate the biases found by LS. While the splitter is effective as a neural network, it may not be directly interpretable to human practitioners. To address this, we propose identifying patterns and prominent attributes in the LS-test and LS-train splits using a simpler and more interpretable algorithm trained to mimic the behavior of the splitter, which we refer to as the *interpreter*. We train the interpreter to predict whether a given data point was put into LS-train or LS-test by the splitter. The interpreter can be fed either the original data itself or metadata about the data, such as a patient's demographics or information about how an image was collected. We further acknowledge that different biases may arise for different class labels (i.e. diseased vs healthy), so we retrain the interpreter separately over data with each label. This process is illustrated in Figure 2.1b.

2.1.2 Overview of Experiments and Validation

Experiments We ran our bias detection framework over four datasets spanning three healthcare and biomedical applications: diabetes prediction from clinical data, lung cancer risk prediction from CT (computed tomography) scans, and biomolecular toxicity prediction. We worked with four types of interpretable models in our studies: RIPPER rulesets, decision trees, logistic regression, rationale models, and invariant rationale models. These models helped us to identify the most bias inducing features in each dataset. For further understanding of the make-up of the LS splits, we also graphed the distributions of various features in the LS-Train and LS-Test splits, visualizing whether a given feature is over or under-represented in the LS splits. More details about each dataset and its experiments are provided later in this chapter.

Validation Methods We performed validation of the biases found by our framework in two main ways. The first way is useful for AI practitioners who are trying to understand potential biases while training an AI model architecture over a given dataset. Specifically, while understanding which features cause bias is useful, we further wished to create manually curated splits that can act as human-understandable heuristics for the LS splits in terms of clinical and demographic attributes considered important by the interpretable models. Then, we trained and tested the AI models on the manually curated train-test splits and calculated the corresponding generalization gaps. Analyzing these manual train-test splits is useful for two reasons. Firstly, it is not always feasible to improve an AI model based on the high-dimensional biases uncovered using LS, which is why it is useful to have simplified heuristics for the LS splits. These splits are easier for AI practitioners to work with, but one must validate that they still represent true biases that could cause a specific AI model architecture to fail over a given dataset. Secondly, comparing the generalization gaps of the manual and LS splits allows one to assess how much more expressive the high-dimensional LS splits are compared to splits represented by only one or a few features.

Our second method of validation can be applied to datasets for which there exists a pre-trained AI model already in deployment and where one is trying to uncover previously undetected sources of bias affecting the model. We used the biased subgroups uncovered by our framework as hypotheses for where an existing AI model might fail, and then performed stratified performance evaluations with the AI models over these subgroups. Specifically, the models were tested on both the manual train split and manual test splits defined above, and their performance gaps between the splits were compared.

2.2 Detailed Methods

2.2.1 Using LS

Implementation Details We generally use the LS implementation of Bao et al, with some minor modifications to enable use with the intricate NLST lung cancer dataset. The splitter is trained as a typical neural network, using gradient descent to minimize an objective function. The objective function is designed to push points from the previously allocated

LS-Test split into the LS-Train split if they were classified correctly by the predictor. In order to prevent the splitter from allocating meaningless splits, there are two regularization terms added to the objective function. The split ratio regularizer encourages a desired percentage of data points to be assigned to the LS-Train split, which is left at 75%, as encouraged by Bao et al. [17]. This prevents the splitter from increasing the generalization gap by allocating all samples to the LS-Test set. This term is weighted by a hyperparameter W_{RATIO}, which controls the importance of the sparsity regularizer relative to the main loss function that trains the splitter to increase the generalization gap. Meanwhile, the label balance regularizer encourages that same fraction of data points to have positive labels in the LS-Test and LS-Train sets. This prevents all of the positive labels from ending up in only one of the splits. The label balance regularizer is weighted by a hyperparameter $W_{BALANCE}$.

One can refer to Bao et al. for more implementation details, access to software, and for thorough validation of the algorithm and how it has been shown to be effective for de-biasing when using the uncovered LS-Train and LS-Test splits as subgroups in Group DRO. We present all the hyperparameter values we used in our experiments in Table 2.1. W_{GAP} is the hyperparameter that controls the weight of the loss function that increases the generalization gap; W_{GAP}, W_{RATIO}, and $W_{BALANCE}$ are all normalized to sum to one. The *Convergence Threshold* defines how long to train the splitter; if the loss has not changed by more than the convergence threshold compared to the average loss over the previous five iterations, then splitter training terminates. The *Splitter Training Buffer Length* refers to a hyperparameter controlling a modification we make to the LS algorithm to allow it to handle large data such as the NLST lung cancer CT scans, described in the following paragraph. Other parameters are defined in the implementation of Bao et al. When selecting hyperparameters, we aimed to be as consistent as possible with the default choices of Bao et al. However, small adjustments were sometimes made to the regularizer weights in order to encourage a better compromise between high generalization gap and effective split size and label balance regularization.

Modifications to Learn to Split Algorithm for Large Data Because of the large size of the NLST CT scans in computer memory, we could only effectively operate our GPUs with batch sizes of up to one or two when training the LS splitter. However, the split ratio and label balance regularizing terms used by the objective function when training the LS splitter are not compatible with a small batch size. The original LS publication by Bao et al defines these regularizing terms using KL divergence computed over the training batches of data; however, when the batch size is on the order of one or a few samples, this KL divergence computation becomes meaningless. In order to get around this limitation, we redefined the regularizing terms to operate over a FIFO buffer of data points from previous training batches; the buffer need not hold the entire CT scans but instead merely their labels and their probabilities of being assigned to the LS-Train set (as assigned by the splitter). The length of this buffer is listed as one of the parameters in Table 2.1; we set the buffer to hold 100 data points during our experiments.

Before describing how our modified regularizing terms work, we describe some notation: LENGTH defines the number of data points in the buffer, RATIO defines the fraction of data we wish to be in the LS-Train set, and S defines the output logits from the splitter neural network for a given data point. For a given data point i in the buffer, its label is

Table 2.1: Hyperparameters used when running Learning to Split. *(Default)* signifies the default parameter used in Bao et al [17]. Multi Layer Perceptron architectures are described by the number of inputs to each hidden layer.

	Diabetes	Tox21	NLST ResNet	NLST Sybil
W_{GAP}	1	1	1	
W_{RATIO}	1	1	0.04	
$W_{BALANCE}$	3	1	0.4	
Splitter Training Batch Size	200 (Default)		1	
Splitter Training Batches/Epoch	100 (Default)		1400	
Splitter Convergence Threshold	0.001 (Default)			
Splitter Training Learning Rate	0.001 (Default)			
Patience	5 (Default)			
Predictor Training Batch Size	200 (Default)		1	N/A (Follow Sybil Training Protocol)
Predictor Training Batches/Epoch	100 (Default)		5000	
Predictor Training Learning Rate	0.001 (Default)			
Predictor Model Structure	MLP: 1024, 1024	MLP: 1024, 1024, 1024	ResNet 18	Sybil
Splitter Model Structure				ResNet 18
Splitter Training Buffer Length	N/A		100	

y_i and its probabilities of being assigned to the LS-Train or LS-Test splits, respectively, are $\mathrm{P}_i(\mathrm{TRAIN})$ and $\mathrm{P}_i(\mathrm{TEST})$. These probabilities are calculated using the softmax activation function and the logits S produced by the splitter for that data point. Finally, the indicator function $\mathbb{1}[\mathrm{CONDITION}]$ is 1 if CONDITION is true and 0 otherwise. The function *cross_entropy* represents the cross entropy function *torch.nn.functional.cross_entropy* implemented by PyTorch and available at https://pytorch.org/docs/stable/generated/torch.nn.functional.cross_entropy.html.

The split ratio regularizer Ω_{RATIO} is an objective function that, when minimized, encourages a RATIO fraction of samples to be in the LS-Train split. For it, we use the cross entropy measure of distribution similarity between the splitter logits in the current training batch and an indicator function informed by the buffer. The indicator function is 1 for every element of the batch if the current fraction of samples in the buffer assigned by the splitter to the LS-Train set is less than the desired ratio, and 0 otherwise:

$$\Omega_{\mathrm{RATIO}} = \mathrm{cross\_entropy}(S, \mathbb{1}[\mathrm{P}_{\mathrm{BUFF}}(\mathrm{TRAIN}) < \mathrm{RATIO}]) \tag{2.1}$$

We implement the probability $\mathrm{P}_{\mathrm{BUFF}}(\mathrm{TRAIN})$ of picking a random sample from the buffer that is assigned to the LS-Train set by the splitter as follows:

$$\mathrm{P}_{\mathrm{BUFF}}(\mathrm{TRAIN}) = \frac{1}{\mathrm{LENGTH}} \sum_{i \in \mathrm{BUFF}} \mathrm{P}_i(\mathrm{TRAIN}) \tag{2.2}$$

The split ratio regularizer $\Omega_{\mathrm{BALANCE}}$ is an objective function that, when minimized, encourages the same fraction of samples to have positive labels in both the LS-Train and

LS-Test sets. For it, we also use the cross entropy between the splitter logits in the current training batch and an indicator function. The indicator function is one if the current data point's label, Y, is present in a higher ratio in the the portion of the buffer assigned to the LS-Test split than in the LS-Train split:

$$\Omega_{\text{BALANCE}} = \text{cross\_entropy}(S, \mathbb{1}[\text{P}_{\text{BUFF}}(\text{Y} \mid \text{TRAIN}) < \text{P}_{\text{BUFF}}(\text{Y} \mid \text{TEST})]) \tag{2.3}$$

We implement the probability $\text{P}_{\text{BUFF}}(\text{Y} \mid \text{TRAIN})$ of picking a random sample from the buffer with a label Y, given that the data point is assigned to the LS-Train set by the splitter, as follows (and similarly for $\text{P}_{\text{BUFF}}(\text{Y} \mid \text{TEST})$):

$$\text{P}_{\text{BUFF}}(\text{Y} \mid \text{TRAIN}) = \frac{\sum_{i \in \text{BUFF}} \mathbb{1}[y_i = \text{Y}] \text{P}_i(\text{TRAIN})}{\sum_{i \in \text{BUFF}} \text{P}_i(\text{TRAIN})} \tag{2.4}$$

$$\text{P}_{\text{BUFF}}(\text{Y} \mid \text{TEST}) = \frac{\sum_{i \in \text{BUFF}} \mathbb{1}[y_i = \text{Y}] \text{P}_i(\text{TEST})}{\sum_{i \in \text{BUFF}} \text{P}_i(\text{TEST})} \tag{2.5}$$

2.2.2 Using Interpretable Algorithms

Summary We worked with four types of interpretable models in our studies: RIPPER rulesets, decision trees, logistic regression, rationale models, and invariant rationale models. For each one, we trained separate interpretable models over data with positive and negative labels for the relevant pathology (i.e. positive or negative for lung cancer). All the data points were labelled as positive or negative according to whether they had been placed in the LS-Test split by the LS splitter, and then the interpretable models were trained to predict these labels. We performed Monte Carlo cross validation to assess the performance of the interpretable models at predicting the LS-split of a given data point: we trained eight models, each over a different random subset of 67% of the data, and tested the models' performance over the remaining 33% of the data. When presenting results in Chapter 3, we present the average accuracy over these 8 runs, the average precision values for the LS-Train and LS-Test data points, and the average recall values for the LS-Train and LS-Test data points. All presented averages are accompanied by standard error values.

Decision Trees Decision trees act as a flowchart based on attributes of a data point that allow one to decide whether the data point belongs in the LS-Train or LS-Test split. For decision trees, we used the *sklearn.tree.DecisionTreeClassifier* implementation from scikit-learn, which we trained with the *class_weights* parameter set to *balanced*. This weighs each training data point inversely proportionally to the frequency of its class, to enable training despite the LS-Train and LS-Test sets being different sizes. The scikit-learn decision tree algorithm is trained on all categorical and numerical variables as is and then outputs both a visualization of the tree as well as feature importances for each feature, which encapsulate how much the feature is used by the decision tree to make decisions and are normalized to sum to one over all features. We ranked features as important based on their feature importances as outputted by scikit-learn, which we averaged over the eight runs in the cross validation. When presenting outputs of the interpretable models in Chapter 3, we present

the three most important features selected by the decisions trees, along with their average feature importances (in brackets). We ran decision trees to various depths, with the depth specified in tables where findings are presented.

Logistic Regression Logistic regression performs binary classification by fitting a linear equation decision boundary to the dataset between the LS-Train and LS-Test data points. The magnitude of the coefficients for each feature in the linear equation can be used to decide how much that feature impacts the decision, and the sign (positive or negative) of the coefficient determines whether the feature. Specifically, the odds is defined as odds = P(LS-Test)/P(LS-Train), or the ratio of the probabilities that data point belongs in the LS-Test and LS-Train splits. The odds ratio $R = \text{odds}(x+1)/\text{odds}(x)$ is the ratio by which the odds changes if feature x in a data point changes from 0 to 1. Each feature's coefficient, β, can be used to calculate the odds ratio from the equation $R = e^{\beta}$. For experiments, we used the *sklearn.linear_model.LogisticRegression* implementation from scikit-learn, which we trained with the *class_weights* parameter set to *balanced*, as with the decision trees. The scikit-learn logistic regression program outputs the coefficient for each input feature, from which we then calculate the odds ratio for each feature. We ranked features as important based on the magnitude of the odds ratio, which we calculated as $R = e^{|\beta|}$ and averaged over the eight runs in the cross validation. When presenting outputs of the interpretable models in Chapter 3, we present the three most important features selected by logistic regression in this way, along with their average odds ratio magnitudes (in brackets, and with a negative sign appended if β is negative). When analyzing the odds ratio, it is meaningless unless compared to a reference category that has an odds ratio of one. For example, if the male demographic is the reference category, then the odds ratio for the female demographic describes the change in odds ratio if a data point changes from male to female but all else is held constant. When pre-processing data to feed into interpretable models, we set all binary and categorical features to be one hot encoded and drop the reference category for that feature, since its β is one. To enable comparison between numerical and categorical features, we convert numerical features to categorical features by splitting the feature into three quantiles, where the central quantile (which for simplicity is set to include data points with unknown or undefined values) is set as the reference category. For example, for the feature of age, the logistic regression would assign odds ratios to the low and high age categories, whereas the middle age category would be the reference category. For categorical and binary features, the reference category is dataset-specific.

RIPPER Rulesets For the RIPPER rulesets [49], we used the publicly available Python implementation from https://github.com/imoscovitz/wittgenstein. The algorithm is trained on all categorical and numerical variables as is and then it outputs a list of possible logical rules (for example: "age>50 and BMI>20"), of which at least one must be met for the data point to be categorized as part of the LS-Test set by the ruleset. When using the algorithm, we set the maximum number of rules per ruleset to four and the maximum number of conditions per rule to two. We ranked features as important based on the number of rulesets the feature appears in over the 8 runs. When presenting outputs of the interpretable models in Chapter 3, we present the three most important features selected in this manner, along

with the number of rulesets they appear in (in parentheses).

Rationale Models Rationale models [47] work by identifying a small subset of the input features that can best be used on their own to predict whether a data point belongs in the LS-Test set or LS-Train set. They work by training two neural networks simultaneously, an encoder that selects the top few features (the *rationales*) and a decoder that uses these features to predict the output. The output is a mask array that features several ones, representing the locations of rationales in the input feature array, and zeroes elsewhere. A regularizing term added to the objective function aims to ensure that the number of rationales is close to the desired number (the *sparsity*); this term is weighted by a hyperparameter $W_{SPARSITY}$, which controls the importance of the sparsity regularizer relative to the main loss function that increases the decoder's accuracy. Our Python implementation is based on that in https://github.com/yala/text_nn, so it differs from the original implementation [47] in that we use a Gumbel Softmax function [50] to output a mask array, with a temperature parameter controlling how closely the mask resembles only zeroes and ones. The temperature parameter starts high around one at the start of training, but is gradually reduced by a fraction every few steps until it is near 0 by the end of training, resulting in all mask values being close to zero or one. During testing, the mask array values are rounded to either 0 or 1. We re-implemented the code in PyTorch Lightning to suit our needs; furthermore, the original code was designed to select rationales as words in a paragraph of text, but our implementation selects rationales as elements from an input array of numerical and categorical features. Instead of Recurrent Neural Networks as in the original implementations, we used Multi Layer Perceptrons (MLPs) for both the encoder and decoder neural networks. We trained the models using the typical gradient descent algorithm for neural networks; further implementation details, such as the sparsity, the training batch size, or the exact MLP structure, are dataset specific. We pre-processed data by keeping numerical features as they were and converting all categorical features into one hot encodings (but we did not drop the reference variable as we did for logistic regression). During training, data points were sampled with equal frequency for each class. Once trained, the average mask value was computed for each feature over the test data; a feature's average mask value equals the fraction of data points over which that feature is one in the mask array. We ranked features based on their average mask values, averaged further over the 8 runs. When presenting outputs of the interpretable models in Chapter 3, we present the three most important features selected in this manner, along with their average mask values (in brackets). When training the rationale models, hyperparameters were selected by experimentation with the goal of stable convergence upon rationales with the desired sparsity and to maximize the rationale model's accuracy at predicting the LS split. We present all the hyperparameters used to run the rationale models in Table 2.2. In the table, *Temp. Steps* is the number of training steps between each lowering of the Gumbel temperature, which is lowered by multiplying it by *Decay Rate*; and *MLP* refers to the structure of both the encoder and decoder MLPs, given by the number of inputs to each hidden layer. Remaining parameters, such as those regarding learning rate, batch size, and number of epochs, refer to the corresponding training parameters in PyTorch.

Table 2.2: Hyperparameters used for running rationale and invariant (Inv.) rationale models. Multi Layer Perceptron architectures are described by the number of inputs to each hidden layer.

(a) For data points with negative labels.

NEGATIVE LABEL	NLST Sybil (Inv.)	NLST Sybil	NLST ResNet (Inv.)	NLST ResNet	Tox21	MIMIC (Inv.)	MIMIC	RHG
Dropout	0.2							
Learn. Rate	0.001							
Temp. Steps	100					30		
MLP	100, 50, 50, 50, 10					50, 50, 50, 10		
Batch Size	256				128	64		
Sparsity	3				4	2		
$W_{SPARSITY}$	0.02				0.005	0.1		0.007
$W_{DIFFFERENCE}$	5	N/A	5	N/A	N/A	30	N/A	N/A
Decay Rate	0.88				0.85	0.97		0.93
Epochs	30				50	15		20

(b) For data points with positive labels.

POSITIVE LABEL	NLST Sybil (Inv.)	NLST Sybil	NLST ResNet (Inv.)	NLST ResNet	Tox21	MIMIC (Inv.)	MIMIC	RHG
Dropout	0.2							
Learn. Rate	0.005				0.001			
Temp. Steps	20				50	10		30
MLP	100, 50, 50, 50, 10					50, 50, 50, 10		
Batch Size	32				16	64		32
Sparsity	3				4	2		
$W_{SPARSITY}$	0.02				0.05	0.45	0.1	0.003
$W_{DIFFFERENCE}$	2	N/A	2	N/A	N/A	30	N/A	N/A
Decay Rate	0.98				0.8	0.93	0.89	0.93
Epochs	80				50	15		20

Invariant Rationale Models We also worked with invariant rationale models [48], which we present findings for in Chapter 3. These work in a similar way to rationale models, but aim to ensure that the selected rationales are causal to instead of just correlated with whether a data point ends up in the LS-Train or LS-Test sets. They consider that different data points may have been obtained from different environments, but that the decoder neural network that predicts the LS split from the rationales should work no matter which environment the data is sourced from. Chang et al [48] propose a similar framework to the rationale model, but where a second environment unaware decoder is added and the training enforces both decoders to achieve similar performance. In order to achieve this, a choice of environment variable is needed, which can consist of a feature that is known to be largely non-causal to

choice of the LS-split; this is subjective and dataset-specific. Because it is subjective, we do not emphasize the findings from invariant rationale models, and merely investigate their use to measure their potential utility. We implemented the invariant rationale model in PyTorch Lightning identically to how we implemented the rationale model, except with the addition of the environment unaware decoder neural network, which we also set to be an MLP. There is an additional regularizing term in the objective function which enforces minimization of the performance difference between the two decoders. This regularizing term is weighted by an additional hyperparameter $W_{DIFFERENCE}$, which is dataset-specific. We pre-processed data similarly to for the rationale model, but during training we prepared an extra set of input features with the chosen environment variable removed. Important features are determined from the mask values and then presented in Chapter 3 as for the rationale models. We present all the hyperparameters used to run the rationale models in Table 2.2.

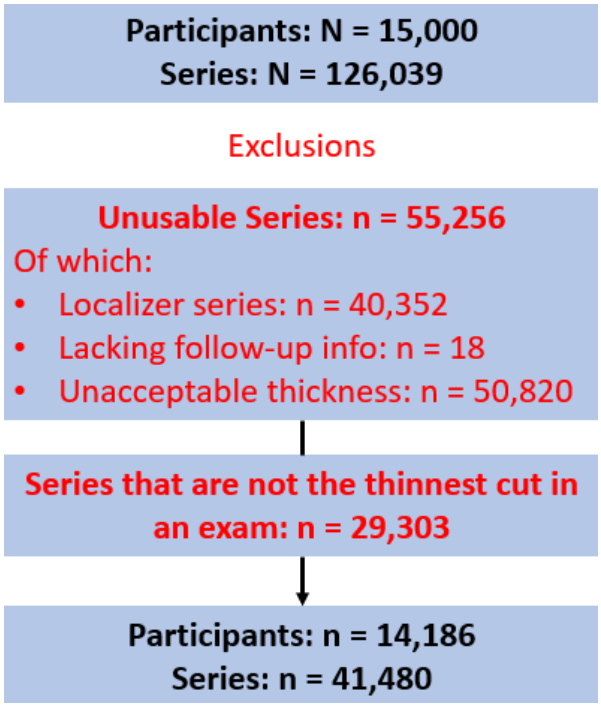
Visualizing Split Statistics For further understanding of the make-up of the LS splits, we graphed the distributions of various features in the LS-Train and LS-Test splits. An example of this is Figure 3.1, which shows the fraction of the splits and overall data consisting of each CT scanner type. This helps to show whether a given device type is over or under-represented in a given LS split compared to its fraction in the overall data.

2.2.3 Datasets

Lung Cancer Data The lung cancer data consists of low dose computed tomography (LDCT) series from the National Lung Screening Trial [51] (NLST). NLST is a major United States lung cancer screening database consisting of X-ray scans from over 50,000 current or former smokers without lung cancer symptoms, with exams taken at up to three time points for each patient. We used data from the subset of 15,000 patients in the NLST database which were used with institutional review board approval for the development of the Sybil AI model. These patients each feature one initial exam and up to two followup exams, producing a total of 126,039 X-ray series. Using a similar filtering process used in the Sybil model's development, we filtered out unacceptable X-ray series and kept only the densest series (or thinnest slice thickness cut) from each exam. After filtering, we were left with 41,480 X-ray series from 14,186 patients to run through our bias detection algorithm. Following the training process of Sybil, each LDCT was considered a unique data point, and we did not associate different scans from the same patient with each other. A summary of the data selection process is presented in Figure 2.2a.

Each X-ray scan we used is a 3D image processed to consist of 200 2D images. When validating biases on the pre-trained Sybil model, we used 256×256 pixel 2D images, as in the original model. When running our bias detection framework, we used 128×128 pixel 2D images to reduce computation time. Otherwise, image formatting and image pre-processing for both test and train images were done using the procedure described in the original Sybil paper [52].

Each series is labelled as positive or negative for cancer in each of the 6 years following the imaging date, based on whether biopsy-confirmed lung cancer is diagnosed in or before a given year. A summary of the number of data points with positive cancer labels for each follow-up year is presented in Figure 2.2b. Note that the number of cancer cases detected is



(a) Exclusions we make to the NLST dataset.

Follow-up Year	Number of Valid Samples	No Cancer	Cancer
1	41,480	40,878	602
2	41,366	40,354	1,012
3	41,157	39,788	1,369
4	40,750	39,086	1,664
5	39,181	37,254	1,927
6	28,980	26,879	2,101

(b) Number of valid samples in the NLST dataset by followup year, including breakdown with and without cancer.

Male / Female	Mean Smoking Years	Mean BMI
24,477 / 17,003	40.0	27.9

(c) The male/female ratio, mean years smoked, and mean BMI in the followup year 0 set of NLST data points. For the mean BMI, the calculation excludes 136 samples with a recorded bmi of 0.

Figure 2.2: Some basic properties of the NLST dataset.

cumulative, and the number of valid data points for each year decreases due to lack of follow-up data and unknown cancer status. In addition, each image is labelled with metadata about the imaging process and patient demographics. The metadata includes the CT scanner device vendor (or device type), which is a categorical variable taking on one of four anonymized corporations; the distribution of these is presented in Table 2.3. The remaining metadata we analyzed includes: hospital/institution (33 categories for anonymized institutions 0 through 32), race (unknown or one of 6 categories for white, Black, Asian, American Indian or Alaska Native, Native Hawaiian or Other Pacific Islander, more than one race), education level (unknown or one of 8 categories for 8th grade or less, 11th grade or less, high school, post high school training, some college, Bachelors degree, graduate school, other), gender (binary), average number of cigarettes smoked per day, total years smoked before NLST trial randomization, years since the patient has quit smoking, age, weight, height, BMI, whether patient has known lung cancer history, whether patient has known non-lung cancer history, whether patient family has known lung cancer history, whether patient has a known disease history (separate binary variables for asthma, asbestosis, bronchiectasis, childhood asthma, chronic bronchitis, COPD, diabetes, emphysema, lung fibrosis, heart disease, hypertension,

pneumonia, sarcoidosis, silicosis, tuberculosis), whether patient is known to have worked at least one year in an at-risk occupation (separate binary variables for asbestos processing, baking, meat processing, coal mining, cotton/jute processing, farming, fire fighting, grain milling, steel milling, rock mining, painting, welding), and whether the cancer is in a known location (separate binary variables for carina, left hilum, lingula, left lower lobe, left main stem bronchus, left upper lobe, mediastinum, right hilum, right lower lobe, right middle lobe, right main stem bronchus, right upper lobe).

Table 2.3: Statistics of the NLST dataset sorted by device vendor.

	Vendor 0			Vendor 1			Vendor 2			Vendor 3		
Year	Total	No Cancer	Cancer (% of Total)	Total	No Cancer	Cancer (% of Total)	Total	No Cancer	Cancer (% of Total)	Total	No Cancer	Cancer (% of Total)
1	**23,007**	22,663	344 (1.5)	**674**	658	16 (2.4)	**14,776**	14,576	200 (1.4)	**3,023**	2,981	42 (1.4)
2	**22,946**	22,356	590 (2.6)	**672**	650	22 (3.4)	**14,735**	14,399	336 (2.3)	**3,013**	2,949	64 (2.2)
3	**22,834**	22,030	804 (3.6)	**670**	634	36 (5.7)	**14,657**	14,213	444 (3.1)	**2,996**	2,911	85 (2.9)
4	**22,602**	21,619	983 (4.5)	**663**	625	38 (6.1)	**14,511**	13,973	538 (3.9)	**2,974**	2,869	105 (3.6)
5	**21,809**	20,677	1,132 (5.5)	**631**	590	41 (6.9)	**13,886**	13,257	629 (4.7)	**2,855**	2,730	125 (4.6)
6	**16,549**	15,319	1,230 (8.0)	**526**	482	44 (9.1)	**9,897**	9,207	690 (7.5)	**2,008**	1,871	137 (7.3)

We ran our bias detection framework over the NLST dataset using the Sybil model as the LS predictor architecture, predicting the probability a patient would develop cancer in each of the 6 follow-up years after the CT scan. The LS splitter model architecture we selected is a subcomponent of the Sybil model, specifically the Resnet-18 encoder implemented by PyTorch as *torchvision.models.video.r3d_18*, available at https://pytorch.org/vision/main/models/generated/torchvision.models.video.r3d_18.html. During each iteration of LS, the predictor was trained identically to how the original Sybil model was trained by Mikhael et al [52]. We used Sybil's cancer predictions for the first follow-up year in our experiments; the LS algorithm maximizes the generalization gap in the ROC-AUC of the predictor for the first follow-up year between the LS-Train and LS-Test sets. When analyzing the LS-splits, we fed the metadata of the CT scans into the interpretable models, and not the CT scans themselves.

In order to compare biases inherent to the NLST data over different AI models, we also ran our bias detection framework in a case where Resnet was used to predict cancer instead of Sybil. For these experiments, the Resnet-18 encoder component of Sybil was used as both the LS predictor and splitter architectures. As the predictor, it was trained to predict whether cancer occurs during the first follow-up year.

When using the the logistic regression interpretable model, we arbitrarily used the following reference categories: white for race, 8th grade or less for education, male for gender, device vendor 0 for device type, and institution 0 for the institution. When using the invariant rationale model, we set the environment variable to be the patient's education level, as

we posited that education should not be directly causal to any pathologies directly visible in the CT scan that can determine which LS split a data point belongs to. We acknowledge that this is a subjective choice and as such do not focus on the findings from the Invariant Rationale models, only investigating them for completeness.

Clinical Diabetes Data We sampled two datasets of diabetes patients: the first is the Medical Information Mart for Intensive Care (MIMIC-IV) dataset [53], [54] version 2.2 [55] and the second is a dataset published by Rich Healthcare Group (RHG) in China [56] for predicting type 2 diabetes (T2D). A summary of both data subsets we use, some important features, and statistical summaries are presented in Table 2.4.

We labelled the MIMIC dataset to signify whether a patient has secondary diabetes or type 2 diabetes, and the data additionally contains the following attributes: age, systolic pressure, diastolic pressure, BMI, gender (binary), hypertension status, kidney failure status, pregnancy status, smoking status, and race (6 categories for white, Black, Hispanic, Asian, Indigenous, or Pacific Islander). We derived the diabetes labels based on ICD diagnosis codes regarding prior diagnoses documented in the raw MIMIC data files. Specifically, we labelled patients as positive if they had ICD-9 [57] or ICD-10 [58] diagnosis codes pertaining to type 2 or secondary diabetes. When selecting hospital admissions to use as MIMIC data points, we first excluded data points for which the patient has multiple or unknown races. We then kept only hospital admission records for which the difference between the admission date and the vitals recording date (the time lag) is within a 14-day window. For each patient in this filtered dataset, we then kept only the hospital admission with the least time lag, and we then further excluded records which are missing relevant demographic or clinical data entries. We labelled patients' hypertension, kidney failure, and pregnancy statuses based on ICD diagnosis codes.

The RHG dataset has a label signifying whether a patient will develop diabetes in the future (the median follow-up time is 3.1 years), as well as the following features: age, systolic pressure, diastolic pressure, BMI, gender (binary), fasting plasma glucose (FPG), cholesterol level, triglyceride level, HDL-C level, LDL level, alanine aminotransferase level (ALT), aspartate aminotransferase level (AST), blood urea nitrogen level (BUN), creatinine clearance test results (CCR), alcohol drinking status (3 categories for never, current, or previous), smoking status, and family history of lung cancer. The authors of RHG consider FPG $\geq$ 7.00 mmol/L to be the diagnostic criteria for T2D when labelling their dataset. When selecting RHG data points, we filtered out any samples missing relevant data entries. In order to create a more balanced data subset, we used all diabetic data points and randomly selected ten percent of nondiabetic data points to use.

Table 2.4: Diabetes datasets summary. Table courtesy of Shrooq Alsenan.

Dataset	Num. of patients	Diabetes Definition	Diabetes		Mean age	Mean BMI	Gender		Smoker		Drinker		Family history	
			Y	N			M	F	Y	N	Y	N	Y	N
MIMIC-IV	37246	ICD-9 or ICD-10 diagnosis codes	7472	29774	58.89	28.35	15451	21795	5905	31341	n/a	n/a	n/a	n/a
RHG	16450	FPG $\geq$ 7.00 mmol/L	4174	12276	44.98	24.20	11200	5250	3443	13007	3069	13381	915	15535

The LS predictor and splitter architectures for this task were set to be multi layer percep-

trons (MLPs), a simple and versatile neural network architecture commonly used in research studies that attempt to predict diabetes from clinical data [59]–[61]. The predictor takes in an the array of patient features stored in MIMIC or RHG and returns the probability that the patient has diabetes (in the case of MIMIC) or will develop diabetes (RHG). When analyzing the LS-splits, we fed into the interpretable models the same clinical features that were fed into the predictor and splitter neural networks.

When using the the logistic regression interpretable model with MIMIC data, we arbitrarily used the following reference categories: white for race, female for gender, negative for hypertension, negative for kidney failure, negative for pregnancy, and negative for smoking. When using the invariant rationale model with MIMIC, we set the environment variable to be the patient's race, assuming that this variable is likely to have limited causal relation to diabetes pathology and the rationale decoder should be able to predict LS split without knowledge of the race (although race may be highly correlated with diabetes pathology and LS split). We acknowledge that this assumption may not hold (race may be causal to the LS splitter's choice) and as such do not focus on findings from the Invariant Rationale models, only discussing them for completeness. We do not use the invariant rationale model with RHG data as there is no clear choice of environment variable. When using the the logistic regression interpretable model with RHG data, we arbitrarily used the following reference categories: male for gender, negative for smoking, currently drinking for alcohol consumption, and negative for family history of diabetes.

Biomolecular Data To investigate our framework's utility on biomolecular design applications, we leveraged the Tox21 database, a major database of the toxicological properties of tens of thousands of molecules. We used a subset [62], [63] of 12,707 molecules, where each one's structure is represented as a numerical array, and each one is labelled as positive, negative, or unknown for 12 different toxicological assays (NR-AR, NR-AhR, NR-AR.LBD, NR-Aromatase, NR-ER, NR-ER.LBD, NR-PPAR.gamma, SR-ARE, SR-ATAD5, SR-HSE, SR-MMP, SR-p53). We considered the task of creating a neural network to predict the androgen receptor (NR-AR) assay for a given molecule, and used our framework to find biases in the data that may negatively impact performance over this task.

The LS predictor and splitter architectures for this task were set to be multi layer perceptrons, with the predictor taking in the molecular structure array and returning the probability that the molecule is AR positive. When analyzing the LS-splits, we fed the 11 remaining assay labels into the interpretable models, and not the molecules themselves. For logistic regression, the reference category for each feature is the unknown category.

2.2.4 Validation

Manually Curated LS-Split Proxies While understanding which features are important to the LS splitter is useful, we further wished to create manually curated splits that can act as human-understandable heuristics for the LS splits in terms of clinical and demographic attributes. To create these manually curated splits, we first reran the RIPPER ruleset algorithm and decision tree algorithm but only with the top few important features as found previously by the interpretable models; we forwent logistic regression and rationale models in this step because they are not very meaningful with so few input features. Findings

from these runs of RIPPER and decision trees are presented in Chapter 3, similarly to how we present runs of interpretable models when provided all input features. We manually investigated the resulting rulesets and trees, coming up with simplified rules that roughly encapsulated the patterns found across the trees in the Monte Carlo cross validation runs. These rules can be as simple as segregating a dataset into two parts based on patient age, to more complicated rules such as the decision trees we manually curated and present in Table 3.2b. When curating these manual splits, we also considered the distribution of the data in order to ensure that they did not break the LS regularization rules significantly more than the original LS splits did. The manually curated decision trees presented in Table 3.2b show how many samples from the dataset belong in the LS-Train and LS-Test splits at each node; the high levels of discrepancy between these categories in different leaf nodes of the trees show that the manual splits do a fairly good job of discriminating between the LS splits.

In order to validate that these manual splits represent biased subgroups that can cause the model training and testing process to fail, we trained models over some of the manually curated training splits and tested them over the manually curated testing splits. Once tested, a generalization gap was calculated, just as in LS. In these trials, the AI models and training parameters were the same as for the original LS predictor, but a splitter was not trained. Where feasible, multiple runs were carried out and the average train and test accuracies and generalization gaps were calculated, along with their standard errors (specifically, diabetes data was put through 5 runs and Tox21 data was put through 6 runs). The generalization gaps are presented in Table 3.1 and the split size and label balance statistics are further presented in Chapter 3.

Stratified Performance Analysis We further validated the biases found by our framework in a way that can be applied to datasets for which there exists a pre-trained AI model already in deployment and where one is trying to uncover previously undetected sources of bias affecting the model. To do this, we treated the biased subgroups uncovered by our framework as hypotheses for where an existing AI model may fail, and then performed stratified performance evaluations with the AI model over these subgroups. Specifically, we tested AI models on the manually curated splits defined above and looked for performance gaps between the splits.

We perform several stratified performance evaluations with Sybil, with results presented in Figure 3.3. For a complete understanding of the model performance, we separately tested the original ensembled Sybil model, as defined by the ensemble of five different pre-trained model weights from Mikhael et al [52], over its original training (but not validation) data and its original testing data. We further performed the evaluation over each follow-up year for which data was available, calculating the ROC-AUC values of Sybil for each follow-up year. In total, we performed 4 stratified performance evaluations: one of these evaluations was over subgroups sratified by BMI with a cutoff point of 27.2; one of these was stratified by device type, with separate groups for device types 0 and 2; one of these was only over data points with device type 0 but stratified by BMI with a cutoff point of 27.2; and one was only over data points with device type 2 but stratified by BMI with a cutoff point of 27.2. Each of these four evaluations led to 24 AUC-ROCS values, obtained as the product of two stratified subgroups, two datasets (the Sybil training and testing datasets), and 6 follow-up

years.

We further performed stratified performance evaluations over MLPs trained to predict diabetes over the RHG and MIMIC datasets. The MLPs were the same as those used in LS experiments and were trained using the same training hyperparameters and procedures, except that they were trained over 50% of the data, with validation and model selection performed using another 20% of the data. The remaining 30% of the data was the test set. ROC-AUC values are calculated in the validation and test sets independently for high and low age groups as the stratifying feature. The experiments were repeated with different random subsets of the data 5 times to calculate the standard errors. For MIMIC, low age is less than 62; for RHG, low age is less than 45. These ages were chosen to roughly correspond to the LS-identified age splits.

2.2.5 Error Auditing Baseline

We further assessed the benefit of using our LS-based bias detection framework compared to a commonly used error auditing baseline bias detection method. We performed error auditing using the original ensembled Sybil model with its output calibrator, as defined by the ensemble of five different pre-trained model weights from Mikhael et al [52]. After running the model on its original test dataset, we computed the error for each data point for the first follow-up year, which is computed as $|p - y|$, where p is the probability of cancer in the first year as predicted by the model and the label y is 1 if there is actually cancer and 0 otherwise. Then, the data points were ranked by their error values, and the 30% of data points with the highest error values were placed in a manually curated *inaccurate split*, and the 70% of data points with the lowest error values were placed in a manually curated *accurate split*. These inaccurate and accurate splits could then be analyzed for biases just as the LS-Test and LS-Train splits, respectively. We graph distributions of various features in the accurate and inaccurate splits in Figure 3.4.

Chapter 3

Results

3.1 Overview

The generalization gaps produced by the LS algorithm for each of the datasets are presented in Table 3.1. The accuracies of the interpretable models at predicting the LS-split, as well as the features they identified as biased, are presented later in this section; a summary of the most important biased features is presented in Table 3.2. We are successfully able to make manually curated splits—such as the simplified decision trees presented in 3.2b—that act as simplified proxies for the LS splits and cause generalization gaps during the training and testing process (These generalization gaps are further presented in Table 3.1). Nevertheless, we still generally find that the LS splits are more expressive, having higher generalization gaps, confirming the effectiveness of the LS algorithm at finding highly nontrivial subgroups. We further corroborate the findings of the interpretable models by graphing the occupancies of the LS-Train and LS-Test splits for various biased features in Figures 3.1, 3.5, 3.6, and 3.7 for NLST, MIMIC, RHG, and Tox21 data, respectively.

Table 3.1: Comparing LS splits with manually curated splits.

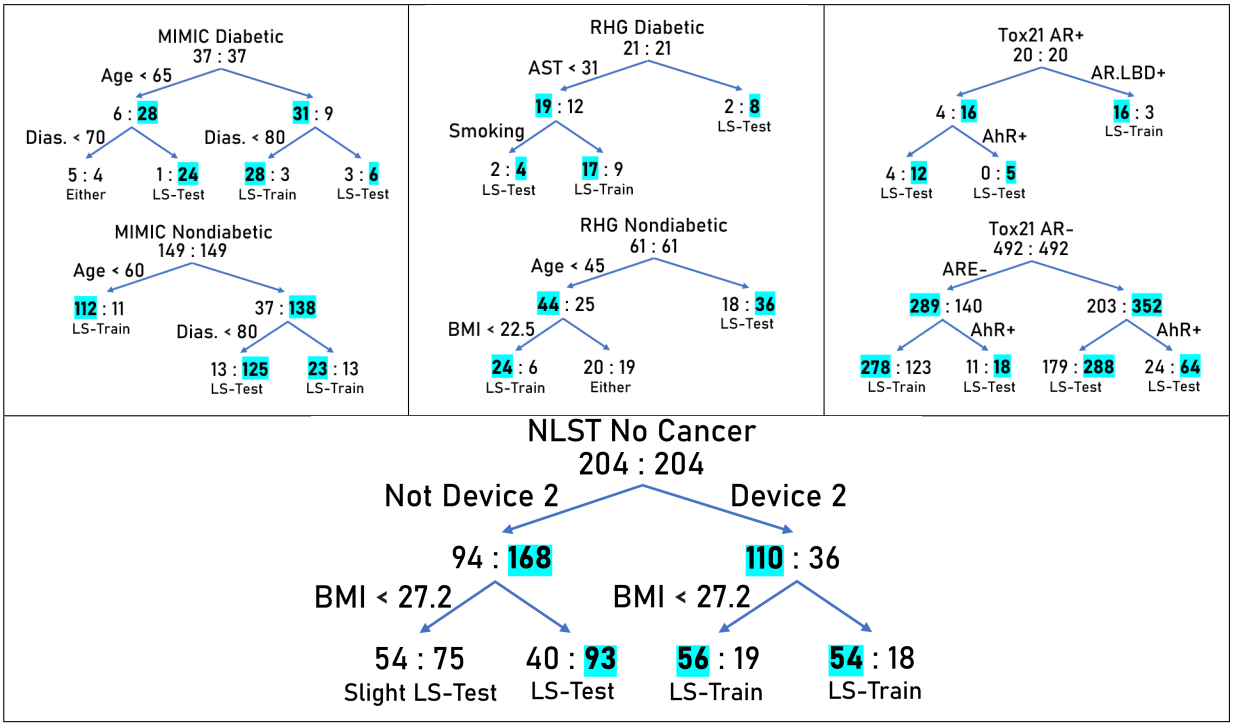
	Manual Splits			LS Splits	
Dataset	Split Curation	Train / Test ROC-AUC (%)	Generalization Gap (%)	Train / Test ROC-AUC (%)	Generalization Gap (%)
MIMIC	Age $\geq$ 62 Diabetic, Age $<$ 62 Nondiab. $\rightarrow$ Train; Age $<$ 62 Diabetic, Age $\geq$ 62 Nondiab. $\rightarrow$ Test	99.3±0.4 / 3.0±0.8	96±1	∼100.0 / 0.3	99.7
	Table 3.2b Decision Tree	98.4±0.2 / 9.4±0.3	88.9±0.2		
RHG	Age $\geq$ 45 Diabetic, Age $<$ 45 Nondiab. $\rightarrow$ Train; Age $<$ 45 Diabetic, Age $\geq$ 45 Nondiab. $\rightarrow$ Test	99.73±0.02 / 7.3±0.3	92.4±0.3	97.2 / 41.7	55.5
	Table 3.2b Decision Tree	98.3±0.1 / 50.4±0.7	47.9±0.8		
Tox21-AR	AhR- $\rightarrow$ Train, AhR+ $\rightarrow$ Test	85±1 / 53±3	31±3	∼100 / 41	59
NLST (ResNet) (Year 1)	Device 1 & 2 $\rightarrow$ Train, Device 0 & 3 $\rightarrow$ Test	65 / 49	16	77 / 50	26
NLST (Sybil) (Year 1 unless noted otherwise)	Device 1 & 2 $\rightarrow$ Train, Device 0 & 3 $\rightarrow$ Test	88 / 78	10	88 / 77	11
	As above, year 2	85 / 73	12		
	As above, year 3	81 / 71	11		
	As above, year 4	81 / 69	12		
	As above, year 5	80 / 68	12		
	As above, year 6	79 / 69	11		
	Device 0 Only, BMI $\geq$ 27.2 $\rightarrow$ Train, BMI $<$ 27.2 $\rightarrow$ Test	73 / 62	12		
	Device 2 Only, BMI $\geq$ 27.2 $\rightarrow$ Train, BMI $<$ 27.2 $\rightarrow$ Test	77 / 73	4		

Table 3.2: Interpretable model analysis of LS splits.

(a) Summary of important features selected by interpretable models.

Dataset	Important Features			
	Over Positive Labels		Over Negative Labels	
	Rationale Model	Logistic Regression	Rationale Model	Logistic Regression
MIMIC	• Age • Kidney Failure • Diastolic	• Young/Old Age • High/Low Diastolic • High Systolic	• Age • Hypertension	• Young/old Age • Pregnant • High/Low Diastolic
RHG	• AST • Smoking • Age	• High AST • Smoking • Old Age	• Age • BMI • Gender	• Smoking • Old Age • Low ALT
Tox21-AR	• ER.LBD+ • AR.LBD+ • AhR-	• AR.LBD+ • ATAD5+ • AhR+	• HSE- • AhR+ • ARE-	• HSE+ • ARE- • HSE-
	Rationale Model	Decision Tree	Rationale Model	Decision Tree
NLST with Sybil	Inconclusive	Inconclusive	• Device: 0 • Institution: 0, 2	• Device • Institution • BMI
NLST with ResNet	Inconclusive	Inconclusive	• Device: 2, 0, 3	• Device • Institution • Weight

(b) Interpretable decision trees summarizing the biased LS splits using only the most important features. The numbers represent the number of samples in the LS-Train:LS-Test splits (in 100s of samples, except for Tox21, which is in 10s of samples). Samples are reweighted to simulate an overall equal number of LS-Train and LS-Test samples. Significant bias is highlighted in blue.



3.2 Lung Cancer Results

3.2.1 Uncovered Biases

As highlighted in Table 3.2 (and in more detail in Table 3.3), the primary features revealed to cause bias in the NLST data by our framework include the CT scanner device vendor, the institution the image is taken at, and the patient's BMI or weight. These biases are roughly consistent irrespective of whether Sybil or ResNet are used as cancer risk predictors. We note that the majority of the interpretability analysis we do for NLST is on samples without cancer, as these vastly outnumber the number of samples that are positive for cancer, which have statistically small sample sizes for some metafeature categories.

These uncovered biases are corroborated in Figure 3.1, where it is shown that the samples from Device 0 are heavily overrepresented in the LS-Test split whereas samples from device 2 are heavily overrepresented in the LS-Train split. Similarly, samples with higher BMI values tend toward the LS-Test split whereas samples with lower BMI values tend towards the LS-Train split. Although institution is also frequently highlighted by the interpretable models as causing bias, it is not immediately clear whether the institution itself is causal to the bias, or whether institution is simply correlated with causal sources of bias such as device vendor. Based on analysis presented in Figure 3.2, we show that hospitals that use devices 1 and 2 end up largely in the LS-Train split whereas remaining device types tend to end up in the LS-Test split, indicating that biases correlated with institution are likely to be caused by the device vendors used by the institution. As such, we focus our analysis on biases related to BMI and device type. We also investigated whether invariant rationale models can decouple the effects of device type and institution to find the true causal source of bias. As shown in Table 3.3a, we find that the normal rationale model highlights the most important institutions and devices as roughly equally important; however, the invariant rationale model considers device 2 with an average mask value of 7, whereas the top institution has an average mask value of only 1.2, meaning that the invariant rationale models ranks the device type as significantly more causal to the bias than the institution. Nevertheless, we once again caution against relying on the findings of the invariant rationale model because of the subjective nature of their environment selection.

By running decision tree interpretable models using just BMI and device type (with the results presented in Table 3.4), we uncover a pattern in which BMI affects the LS-split much more for certain devices than for others. We find this by visualizing an approximation of the resulting decision trees in Table 3.2b. Specifically, the decision tree indicates that there is very little difference in the distribution of BMI between the LS splits for data points of device 2, but that there is a difference in distribution for other device types. The nontrivial nature of the bias uncovered in this manner highlights a key strength of the framework we present, and manually graphing the distributions in Figures 3.1e and 3.1f confirms this pattern highlighted by the interpretable models. Furthermore, despite only using BMI and device type, the interpretable models achieve almost the same performance as when provided all features, showing that most of the decision making of the LS splitter is based on these features.

Table 3.3: Important features selected by various interpretable models for lung cancer, along with the accuracies of these models' predictions.

(a) Analysis performed with Sybil on samples without cancer.

Without Cancer	RIPPER Ruleset (Max 4 Rules)	Logistic Regression	Decision Tree (Depth=8)	Rationale Model (Sparsity=3)	Inv. Rationale Model (Sparsity=3)
Important Features	-Device 0/3 (8) -Institution 14 (5)	-Institution 23 (-9.6) -Institution 2 (-7.2) -Device 2 (-5.4)	-Device (0.65) -Institution (0.15) -BMI (0.09)	-Institution 0 (5.0) -Device 0 (4.0) -Institution 2 (4.0)	-Device 2 (7.0) -Institution 0 (1.2) -Device 0 (1.1)
Accuracy (%)	69.12 ± 0.05	70.7 ± 0.2	70.2 ± 0.2	68 ± 1	68.2 ± 0.2
Train/Test Precision (%)	73.6 ± 0.2 / 66.3 ± 0.2	72.3 ± 0.1 / 69.5 ± 0.2	70.2 ± 0.3 / 70.2 ± 0.2	70. ± 2 / 68 ± 1	73.9 ± 0.7 / 64.9 ± 0.4
Train/Test Recall (%)	58.3 ± 0.4 / 79.6 ± 0.4	65.7 ± 0.2 / 75.6 ± 0.2	69.1 ± 0.3 / 71.2 ± 0.4	65 ± 4 / 70. ± 7	55 ± 1 / 81 ± 1

(b) Analysis performed with Sybil on samples with cancer. (Loclmsb refers to cancer in left main stem bronchus.)

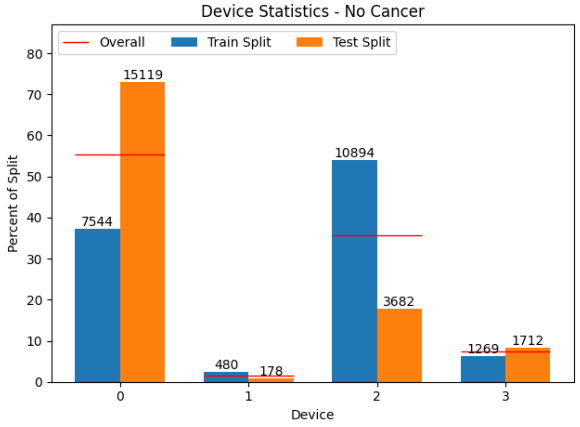
With Cancer	RIPPER Ruleset (Max 4 Rules)	Logistic Regression	Decision Tree (Depth=8)	Rationale Model (Sparsity=3)	Inv. Rationale Model (Sparsity=3)
Important Features	-Device 0 (8) -Cigarettes per day (2)	-Institution 6 (3.9) -Loclmsb (3.8) -Device 1 (-3.6)	-Institution (0.15) -BMI (0.12) -Years smoked (0.10)	-Institution 17 (2.0) -Years since smoking (2.0)	-Device 0 (2.0) -Gender male (2.0)
Accuracy (%)	41 ± 5	65.2 ± 0.9	66 ± 1	40 ± 10	57 ± 10.
Train/Test Precision (%)	17.6 ± 0.9 / 87 ± 2	21 ± 2 / 85.5 ± 0.4	20 ± 1 / 84.1 ± 0.9	20 ± 10 / 50 ± 20	14 ± 4 / 80 ± 10
Train/Test Recall (%)	71 ± 8 / 35 ± 8	38 ± 3 / 70 ± 1	34 ± 3 / 72 ± 1	60 ± 20 / 40 ± 20	50 ± 20 / 60 ± 10

(c) Analysis performed with ResNet on samples without cancer.

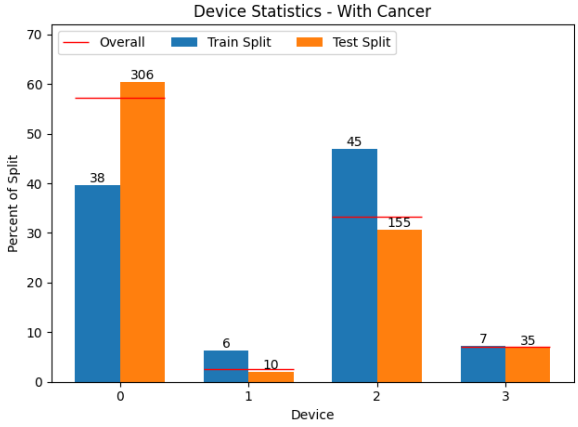
Without Cancer	RIPPER Ruleset (Max 4 Rules)	Logistic Regression	Decision Tree (Depth=8)	Rationale Model (Sparsity=3)	Inv. Rationale Model (Sparsity=3)
Important Features	-Device 0/3 (8)	-Institution 2 (-8.1) -Institution 23 (-7.8) -Device 2 (-6.8)	-Device (0.70) -Institution (0.14) -Weight (0.06)	-Device 2 (5.0) -Device 0 (3.0) -Device 3 (2.0)	-Device 2 (6.0) -Device 0 (2.0) -Institution 0 (1.0)
Accuracy (%)	73.9 ± 0.1	75.28 ± 0.06	74.4 ± 0.1	73.2 ± 0.5	72.9 ± 0.5
Train/Test Precision (%)	68.7 ± 0.1 / 77.0 ± 0.2	69.0 ± 0.1 / 79.6 ± 0.1	66.5 ± 0.2 / 80.8 ± 0.2	68 ± 1 / 76.5 ± 0.2	68 ± 1 / 76.2 ± 0.3
Train/Test Recall (%)	64.1 ± 0.2 / 80.53 ± 0.08	70.0 ± 0.2 / 78.84 ± 0.05	73.6 ± 0.4 / 74.9 ± 1	63.5 ± 0.8 / 80 ± 1	63.0 ± 1.0 / 80. ± 1

(d) Analysis performed with ResnNet on samples with cancer.

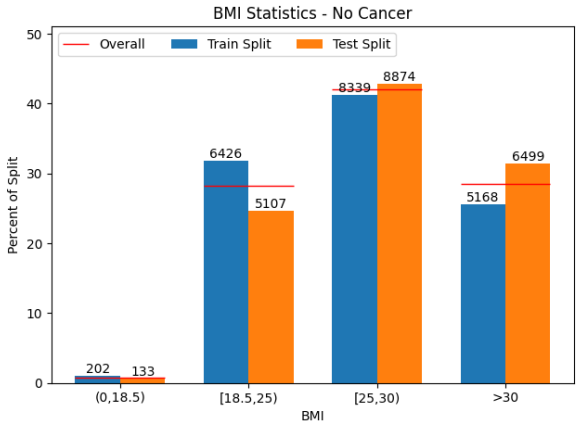
With Cancer	RIPPER Ruleset (Max 4 Rules)	Logistic Regression	Decision Tree (Depth=8)	Rationale Model (Sparsity=3)	Inv. Rationale Model (Sparsity=3)
Important Features	-Device 0 (8)	-Device 2 (-6.6) -Institution 6 (5.1) -Institution 4 (-4.4)	-Device (0.15) -BMI (0.12) -Institution (0.11)	-Device 2 (3.0) -Device 0 (2.0)	-Device 0 (4.0) -Device 2 (4.0) -Years smoked (3.0) -Institution 0 (3.0)
Accuracy (%)	60 ± 2	77.1 ± 0.7	79 ± 1	60 ± 10	59 ± 6
Train/Test Precision (%)	12 ± 1 / 95.7 ± 0.8	13 ± 2 / 93.2 ± 0.5	14 ± 2 / 93.5 ± 0.6	12 ± 2 / 80 ± 10	12 ± 2 / 96.0 ± 0.8
Train/Test Recall (%)	66 ± 7 / 60. ± 3	33 ± 4 / 81.0 ± 0.7	34 ± 4 / 82 ± 1	60 ± 10 / 60 ± 10	66 ± 8 / 58 ± 7



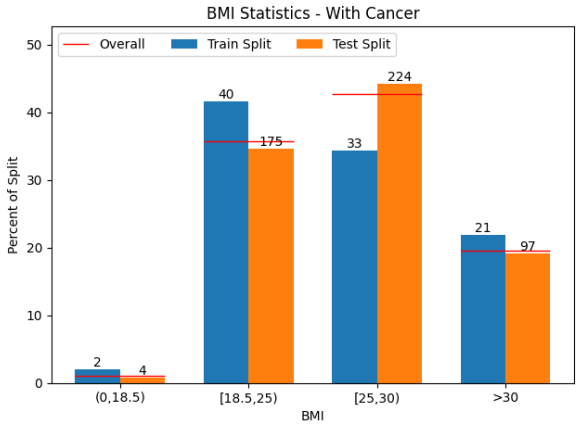
(a) By Device Type - No Cancer



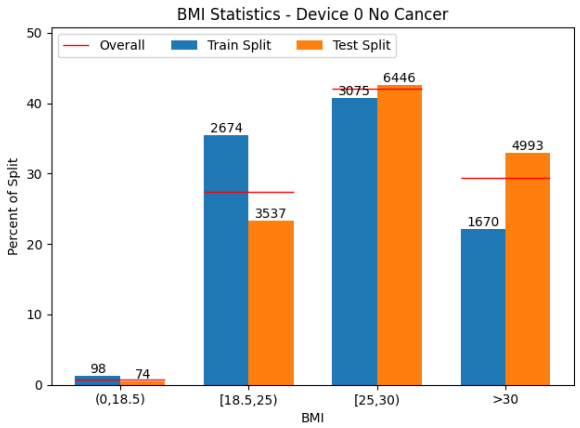
(b) By Device Type - With Cancer



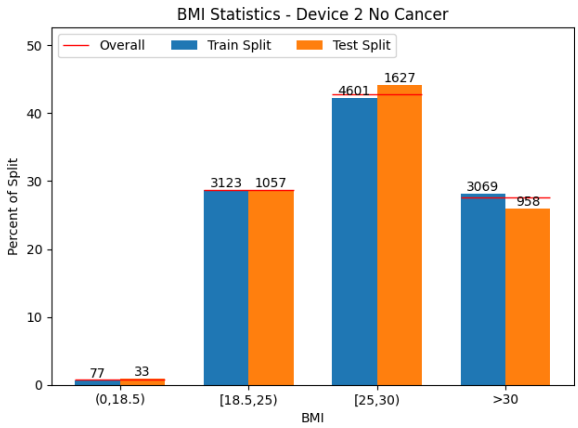
(c) BMI Barchart - No Cancer



(d) BMI Barchart - With Cancer



(e) BMI Barchart - Only Datapoints from Device 0 without cancer



(f) BMI Barchart - Only Datapoints from Device 2 without cancer

Figure 3.1: Statistics for device type and BMI in the LS-generated train/test splits with Sybil.

Table 3.4: Important features selected by various interpretable models for NLST data, when using only the most relevant features (device vendor and BMI).

(a) Analysis performed on samples without cancer.

Without Cancer	RIPPER Ruleset (Max 4 Rules)	Decision Tree (Depth=3)
Important Features	-Device 0 (6) -Device 2 (6) -Device 3 (3) -BMI (3, with all having BMI > 31)	-Device 2 (0.87) -BMI (0.07) -Device 1 (0.06)
Accuracy (%)	68.6 ± 0.1	68.98 ± 0.09
Train/Test Precision (%)	74.6 ± 0.1 / 65.1 ± 0.2	74.0 ± 0.4 / 66.0 ± 0.2
Train/Test Recall (%)	55.1 ± 0.5 / 81.7 ± 0.2	57.1 ± 0.4 / 80.5 ± 0.5

(b) Analysis performed on samples with cancer.

With Cancer	RIPPER Ruleset (Max 4 Rules)	Decision Tree (Depth=3)
Important Features	-Device 0 (8) -BMI (7, with 6 having BMI > 25)	-BMI (0.72) -Device 0 (0.20) -Device 3 (0.05)
Accuracy (%)	43 ± 8	64 ± 4
Train/Test Precision (%)	19 ± 2 / 88 ± 2	23 ± 2 / 86 ± 1
Train/Test Recall (%)	74 ± 7 / 40 ± 10	45 ± 6 / 68 ± 5

3.2.2 Validating Biases

To validate that the discovered causes of bias cause the model to fail to generalize from training to testing, we manually curated simplified test/train splits to represent the uncovered biases, as highlighted in Table 3.1. Some of these splits are segregated purely based on device type and some are segregated purely based on BMI. We then trained and tested the Sybil model over these splits to confirm that the biases our framework uncovered actually do result in generalization gaps over the NLST dataset with the Sybil architecture. Specifically, training the Sybil model on data containing device 2 and testing it on data containing device 0 results in a generalization gap of 10-12 percent, depending on the follow-up year of cancer status being used (results for all six follow-up years are presented in Table 3.1); this is almost as much as the generalization gap uncovered by the LS algorithm. Furthermore, we find that for data from device type 0, training Sybil on high BMI data and testing it on low BMI data makes a similarly high generalization gap. However, in line with the predictions made by our framework, BMI does not seem to act as a significant bias for data from device 2, resulting in a much lower generalization gap (shown in table 3.5c). All of the manually curated splits, along with their sizes and label ratios, are presented in totality in Table 3.5.

Since Sybil is a pre-trained and publicly available cancer prediction model, we further performed stratified performance evaluation (presented in Figure 3.3) to investigate whether the sources of bias we uncovered in the NLST data affect not just Sybil's architecture on adversarial train/test splits, but also whether these biases manifest as poor performance by the pre-trained Sybil model that currently exists publicly. As shown in Figure 3.3b, we find that device type does negatively affect the pre-trained Sybil model. In line with what our framework suggests, Sybil's ROC-AUC performance on its original NLST test data drops by up to 15 percentage points on device 0 compared to device 2, depending on the follow-up year being investigated. Further in line with our framework's predictions, Sybil's performance on patients with high BMI drops by up to 10 percentage points in ROC-AUC for data from device 0, an effect that is not observed for data from device 2 (see Figure 3.3c).

Table 3.5: Test-train splits used with NLST, along with their generalization gaps and regularization properties (including both LS splits and manual splits).

(a) Running Sybil model.

	Generalization Gap in ROC-AUC Percentage (Train AUC – Test AUC)	Split Size Regularization: Train/Test Number of Samples (%)	Label Balance Regularization: Train/Test Samples with Cancer (%)
Initial Random Split	2 (89 – 87)	31,124 (75.0) / 10,356 (25.0)	451 (1.45) / 151 (1.46)
LS-Created Split (Best)	11 (88 – 77)	20,283 (48.9) / 21,691 (51.1)	96 (0.47) / 506 (2.33)
Manual Split: Devices 0 & 3 → Test Devices 1 & 2 → Train	10 (88 – 78)	15,450 (37.2) / 26,030 (62.8)	216 (1.40) / 386 (1.48)

(b) Running ResNet-18 model.

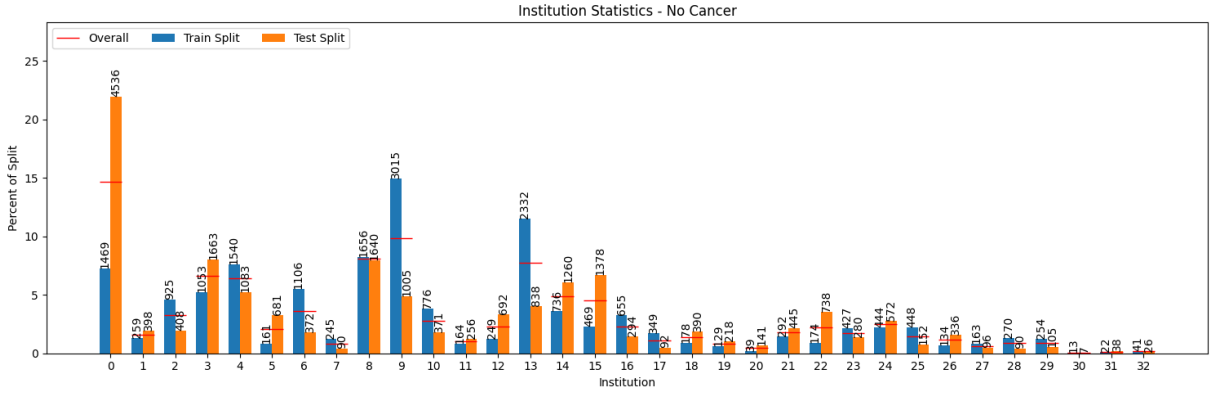
	Generalization Gap in ROC-AUC Percentage (Train AUC – Test AUC)	Split Size Regularization: Train/Test Number of Samples (%)	Label Balance Regularization: Train/Test Samples with Cancer (%)
Initial Random Split	3 (58 – 55)	31,124 (75.0) / 10,356 (25.0)	451 (1.45) / 151 (1.46)
LS-Created Split (2nd Best)	20 (75 – 54)	31,087 (74.9) / 10,393 (25.1)	121 (0.39) / 481 (4.63)
LS-Created Split (Best)	26 (77 – 50)	16,473 (39.7) / 25,007 (60.3)	47 (0.29) / 555 (2.22)
Manual Split: Devices 0 & 3 → Test Devices 1 & 2 → Train	16 (65 – 49)	15,450 (37.2) / 26,030 (62.8)	216 (1.40) / 386 (1.48)

(c) Running Sybil model over custom splits based on BMI. By analyzing the LS-generated splits, we find that overweight BMI values tend to lean towards the train split and vice versa for samples of device vendor 0. However, this pattern was much less pronounced in datapoints of device vendor 2. This made us hypothesize that the generalization gap caused by an overweight train set would be much larger for device type 0 than device vendor 2, a hypothesis which is empirically supported by this table.

	Generalization Gap in ROC-AUC Percentage (Train AUC – Test AUC)	Split Size Regularization: Train/Test Number of Samples (%)	Label Size Regularization: Train/Test Samples with Cancer (%)
Manual Split over Device 0: BMI < 27.2 → Test BMI >= 27.2 → Train	12 (73 – 62)	7,227 (49.0) / 7,516 (51.0)	96 (1.33) / 133 (1.77)
Manual Split over Device 2: BMI < 27.2 → Test BMI >= 27.2 → Train	4 (77 – 73)	7,227 (49.0) / 7,516 (51.0)	104 (1.44) / 326 (1.68)

3.2.3 Error Auditing Baseline

We further assessed the benefit of using our LS-based bias detection framework compared to an error auditing baseline bias detection method, which entails analyzing features of the data points that are most inaccurately predicted by the pre-trained AI model. Running error auditing over the pre-trained Sybil model reveals a difference in performance over different BMI values, but does not reveal significant difference in performance over different device types or at the intersection of device types and BMI (see Figure 3.4). This demonstrates the potential of analyzing the expressive splits generated by LS to detect nontrivial biases compared to other baseline bias detection schemes.

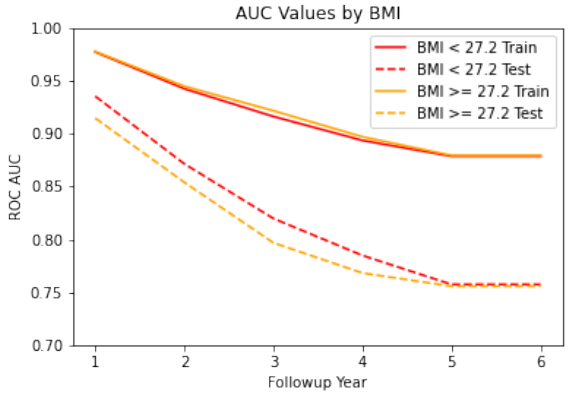


(a) Institution statistics in the LS splits for Sybil.

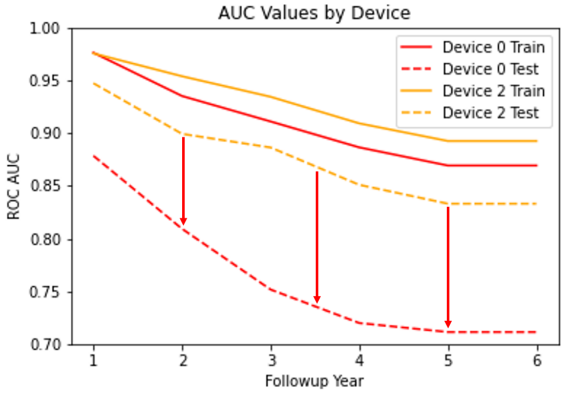
Institution	% of Data	% Device 0	% Device 1	% Device 2	% Device 3	LS Split
0	14.7	100.0				Test
9	9.8			100.0		Train
8	8.0	59.2	0.1	40.6		Unclear
13	7.7			100.0		Train
3	6.6	100.0				Test
4	6.4	51.1		48.9		Train
14	4.9				100.0	Test
15	4.5	99.8			0.2	Test
6	3.6			100.0		Train
2	3.3	87.5		0.9	11.6	Train*
10	2.8	15.4		84.6		Train
24	2.5	99.8	0.2			Test
12	2.3	100.0				Test
16	2.3		33.1	54.9	12.0	Train
22	2.2	100.0				Test
5	2.1	100.0				Test
21	1.8	99.9			0.1	Test
23	1.7				100.0	Train*
1	1.6	100.0				Test
25	1.5	0.3		99.3	0.3	Train
18	1.4	100.0				Test
26	1.2	100.0				Test
17	1.1			100.0		Train
11	1.0	100.0				Test
19	0.9	96.0	4.0			Test
28	0.9	0.3		99.7		Train
29	0.9			100.0		Train
7	0.8	2.0	98.0			Train
27	0.6	99.2		0.8		Train*
20	0.5	100.0				Test
32	0.2	100.0				Unclear
30	0.1			100.0		Unclear
31	0.1	100.0				Unclear

(b) The device composition of the most prevalent institutions (ordered by prevalence), showing strong correlation between institutions that use devices from vendor 0 and institutions placed into the LS-Test set by LS. An asterisk indicates that the institution does not follow the pattern of devices 0 and 3 belonging to the LS-Test split.

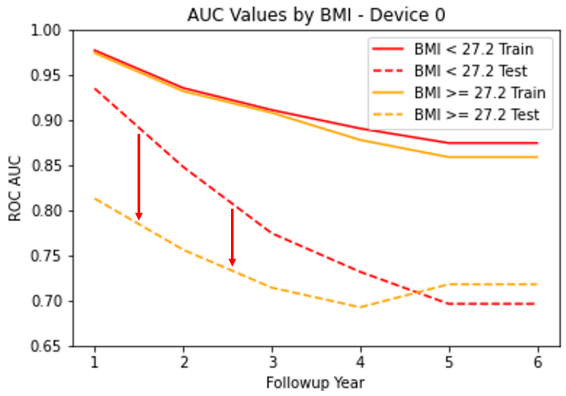
Figure 3.2: Analysis of relationship between institution and device type in LS splits.



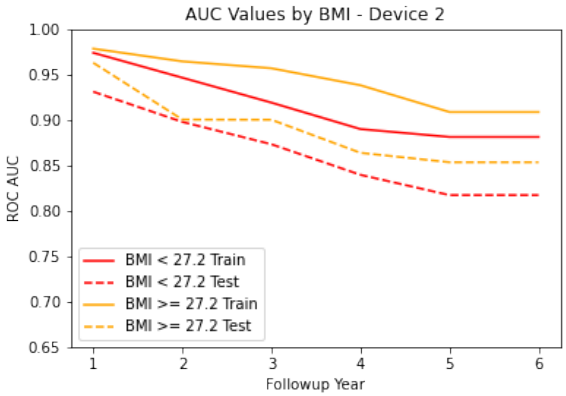
(a) AUCs by BMI



(b) AUCs by Device

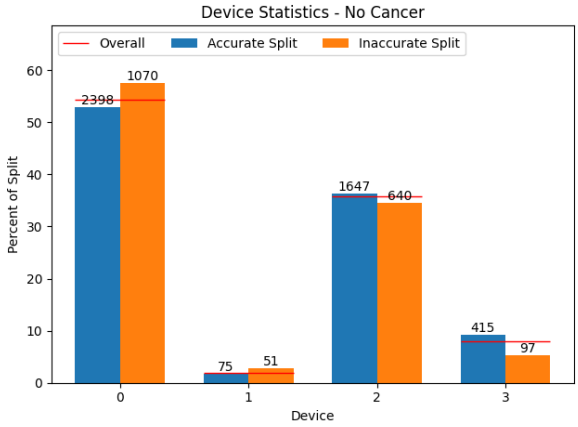


(c) AUCs by BMI - Device 0

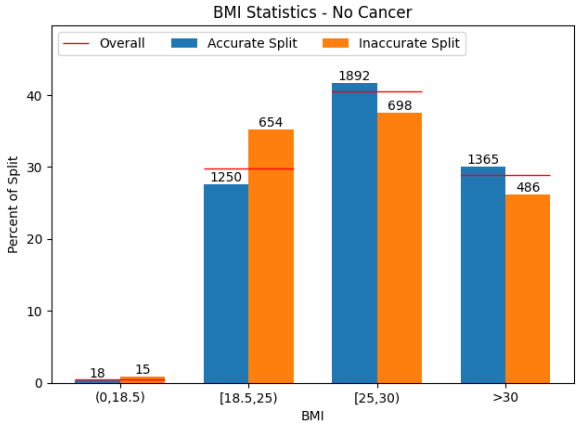


(d) AUCs by BMI - Device 2

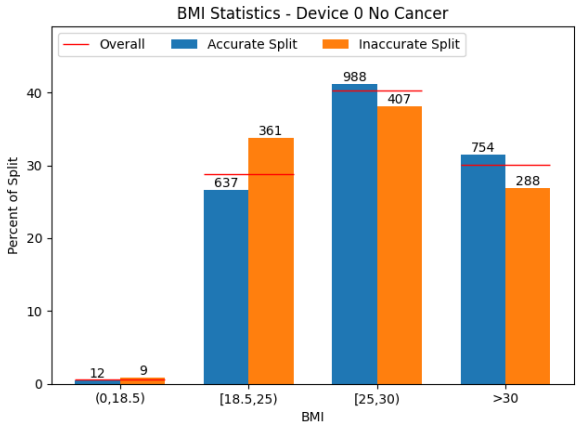
Figure 3.3: Visualizations of AUCs by year, stratified by device type and BMI to perform stratified performance evaluations on the original Sybil model.



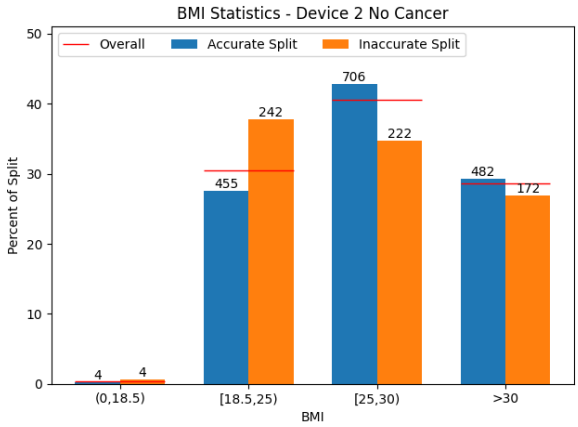
(a) By Device Type - No Cancer



(b) BMI Barchart - No Cancer



(c) BMI Barchart - Only Datapoints from Device 0 without cancer



(d) BMI Barchart - Only Datapoints from Device 2 without cancer

Figure 3.4: Statistics of device type and BMI in the error auditing baseline's accurate/inaccurate splits.

3.3 Diabetes Results

3.3.1 Uncovered Biases

As highlighted in Table 3.2 (and in more detail in Table 3.6), some of the primary features consistently revealed by the interpretable models to cause bias in diabetes data are age and diastolic pressure in the MIMIC dataset, age in the RHG dataset, aspartame aminotransferase (AST) levels and smoking status in diabetic samples from the RHG dataset, and BMI in nondiabetic samples from the RHG dataset.

As there are significant numbers of both diabetic and nondiabetic samples in both datasets, we were able to analyze the findings of the interpretable models on each group independently. As with NLST data, we created simplified heuristics for the LS-splits by running a shallow decision tree algorithm using just the most important features highlighted by the interpretable models (analysis shown in Table 3.7), with an approximation of the resulting trees visualized in Table 3.2b. Investigating the resulting trees and the feature statistics graphed in Figures 3.5 and 3.6, we see that the correlation between a feature's value and its corresponding LS split tends to be reversed for diabetic samples compared to nondiabetic samples. For example, diabetic MIMIC samples with high age are concentrated in the LS-Train split, but nondiabetic samples with high age end up in the LS-Test split. Effectively, this means that training an AI model on diabetic samples with older ages causes it to fail when tested on the less common population of diabetic samples with younger age. A similar pattern emerges with diastolic pressure, in which training an AI model on diabetic samples with lower diastolic pressure causes it to fail when tested on diabetic samples with higher diastolic pressure. Our framework here uncovered that the MLPs used to predict diabetes are prone to shortcut learning with age and diastolic pressure when using the MIMIC dataset, and are brittle when applied over segments of the population such as younger diabetic people. This makes sense considering that there is a positive correlation between age and diabetes in the MIMIC dataset, and considering that diastolic pressure has a negative correlation with old age and tends to be lower in older diabetics than older nondiabetics [64].

Using our methods further enables discovery of unexpected biases acting over small subsets of the population. For example, the interpretable models picked up on pregnancy as a potentially serious spurious correlation in the MIMIC data. Although only a small fraction of MIMIC samples are pregnant, all pregnant diabetic samples end up in the LS-Test set and virtually all pregnant nondiabetic samples end up in the LS-Train set (Figures 3.5c and 3.5f). Upon closer inspection, it becomes clear that essentially all of the pregnant MIMIC samples are nondiabetic in the data subset we use, so pregnancy can act as a spurious correlation in the MIMIC dataset that tricks an AI model into thinking that any such samples are nondiabetic, thus risking failure on pregnant diabetic patients. Although affecting only a small minority of the dataset and not included in our further decision tree analysis, our framework flagged a bias that could pose a grave risk to the safety of AI models as applied to pregnant patients. Similarly, in the RHG dataset, although affecting only the small portion of patients who smoke, the interpretable models flagged smoking as a major cause of being put in the LS-Test set for diabetic patients. This is demonstrated further by the decision trees visualized in Table 3.2b, which show that even though diabetics with low AST tend

Table 3.6: Important features selected by various interpretable models for diabetes, along with the accuracies of these models' predictions.

(a) Analysis performed on MIMIC samples with diabetes.

With Diabetes	RIPPER Ruleset (Max 4 Rules)	Logistic Regression	Decision Tree (Depth=8)	Rationale Model (Sparsity=2)	Inv. Rationale Model (Sparsity=2)
Important Features	-Age ≤ 57 (8) -Diastolic > 86 (8, of which 7 are > 80)	-Young/Old Age (38.5/-22.4) -High/Low Diastolic (9.5/-5.0) -High Systolic (3.8)	-Age (0.51) -Diastolic (0.30) -Systolic (0.12)	-Age (4.0) -Kidney Failure (3.0) -Diastolic (2.0)	-Diastolic (2.2) -Male (1.9) -No Kidney Failure (1.8)
Accuracy (%)	84.2 ± 0.5	86.4 ± 0.4	93.5 ± 0.2	73 ± 4	60. ± 5
Train/Test Precision (%)	82 ± 1 / 89 ± 1	90.5 ± 0.3 / 80.8 ± 0.4	94.6 ± 0.1 / 92.0 ± 0.4	81 ± 2 / 66 ± 4	75 ± 2 / 56 ± 6
Train/Test Recall (%)	94 ± 1 / 70. ± 2	86.6 ± 0.6 / 86.1 ± 0.4	94.7 ± 0.3 / 91.8 ± 0.2	72 ± 5 / 75 ± 1	48 ± 11 / 79 ± 4

(b) Analysis performed on MIMIC samples without diabetes.

Without Diabetes	RIPPER Ruleset (Max 4 Rules)	Logistic Regression	Decision Tree (Depth=8)	Rationale Model (Sparsity=2)	Inv. Rationale Model (Sparsity=2)
Important Features	-Age > 68 (8) -Hypertension (1) -Diastolic ≤ 60 (1)	-Young/Old Age (-54.5/38.1) -Pregnancy (-18.0) -High/Low Diastolic (-14.5/11.4)	-Age (0.69) -Diastolic (0.21) -BMI (0.04)	-Age (5.0) -Hypertension (3.0)	-Age (3.5) -No Hypertension (3.4) -Diastolic (3.1)
Accuracy (%)	84.6 ± 0.2	90.7 ± 0.1	94.41 ± 0.08	77 ± 5	73 ± 4
Train/Test Precision (%)	86.2 ± 0.1 / 80.7 ± 0.4	95.6 ± 0.1 / 82.4 ± 0.3	97.7 ± 0.1 / 88.7 ± 0.2	91 ± 2 / 65 ± 6	86 ± 2 / 59 ± 4
Train/Test Recall (%)	91.3 ± 0.2 / 71.4 ± 0.2	90.1 ± 0.1 / 91.8 ± 0.3	93.8 ± 0.1 / 95.6 ± 0.2	73 ± 8 / 85 ± 4	71 ± 5 / 77 ± 3

(c) Analysis performed on RHG samples with diabetes.

With Diabetes	RIPPER Ruleset (Max 4 Rules)	Logistic Regression	Decision Tree (Depth=8)	Rationale Model (Sparsity=2)
Important Features	-AST ≥ 34 (8) -Smoking (5) -FPG ≤ 4.9 (3)	-High AST (17.5) -Smoking (7.9) -Old Age (-2.6)	-AST (0.27) -ALT (0.14) -FPG (0.14)	-AST (5.0) -Smoking (5.0) -Age (4.0)
Accuracy (%)	84.6 ± 0.3	75.9 ± 0.3	75.8 ± 0.8	74 ± 2
Train/Test Precision (%)	88.3 ± 0.3 / 54 ± 1	93.9 ± 0.2 / 37.2 ± 0.4	92.2 ± 0.4 / 38 ± 1	88.9 ± 0.9 / 34 ± 2
Train/Test Recall (%)	94.0 ± 0.5 / 36.1 ± 0.2	76.2 ± 0.3 / 74.0 ± 0.9	77 ± 1 / 68 ± 2	78 ± 3 / 52 ± 5

(d) Analysis performed on RHG samples without diabetes.

Without Diabetes	RIPPER Ruleset (Max 4 Rules)	Logistic Regression	Decision Tree (Depth=8)	Rationale Model (Sparsity=2)
Important Features	-Age ≥ 52 (8) -Triglyceride ≥ 2.6 (8) -Nonsmoking (8)	-Smoking (-4.1) -Old Age (-2.6) -Low ALT (-2.6)	-ALT (0.26) -Age (0.25) -AST (0.10)	-Age (4.0) -BMI (3.0) -Gender (2.6)
Accuracy (%)	78.9 ± 0.3	73.6 ± 0.3	73.1 ± 0.4	65 ± 1
Train/Test Precision (%)	85.8 ± 0.3 / 46 ± 1	92.8 ± 0.2 / 39.1 ± 0.5	90.8 ± 0.2 / 38.6 ± 0.6	88.7 ± 0.7 / 30.4 ± 0.7
Train/Test Recall (%)	88.5 ± 0.8 / 40. ± 2	73.2 ± 0.3 / 75.1 ± 0.2	74.3 ± 0.5 / 68.3 ± 0.9	65 ± 2 / 65 ± 4

Table 3.7: Important features selected by various interpretable models for diabetes data, when using only the most relevant features.

(a) Analysis performed on MIMIC samples without diabetes, using only diastolic pressure and age.

Without Diabetes	RIPPER Ruleset (Max 4 Rules)	Decision Tree (Depth=2)
Important Features	-Age > 68 (8)	-Age (0.79) -Diastolic (0.21)
Accuracy (%)	84.57 ± 0.10	88.69 ± 0.06
Train/Test Precision (%)	86.0 ± 0.1 / 81.1 ± 0.2	92.0 ± 0.3 / 82.4 ± 0.4
Train/Test Recall (%)	91.5 ± 0.2 / 70.9 ± 0.4	90.8 ± 0.3 / 84.6 ± 0.6

(b) Analysis performed on MIMIC samples with diabetes, using only diastolic pressure and age.

With Diabetes	RIPPER Ruleset (Max 4 Rules)	Decision Tree (Depth=2)
Important Features	-Diastolic > 80 (8, of which 4 are > 81) -Age < 51 (8, of which 6 are < 57)	-Age (0.62) -Diastolic (0.38)
Accuracy (%)	82.7 ± 0.5	86.5 ± 0.5
Train/Test Precision (%)	79.9 ± 0.7 / 90.1 ± 0.5	87.4 ± 0.5 / 85 ± 1
Train/Test Recall (%)	95.4 ± 0.4 / 63 ± 2	91 ± 1 / 80 ± 1

(c) Analysis performed on RHG samples without diabetes, using only BMI and age.

Without Diabetes	RIPPER Ruleset (Max 4 Rules)	Decision Tree (Depth=2)
Important Features	-Age > 58 (8, of which 3 are > 51) -BMI > 26 (8)	-Age (0.67) -BMI (0.33)
Accuracy (%)	80.4 ± 0.3	54 ± 1
Train/Test Precision (%)	83.8 ± 0.3 / 48 ± 1	93.0 ± 0.4 / 27.3 ± 0.4
Train/Test Recall (%)	94.1 ± 0.8 / 22 ± 2	46 ± 2 / 85.1 ± 2

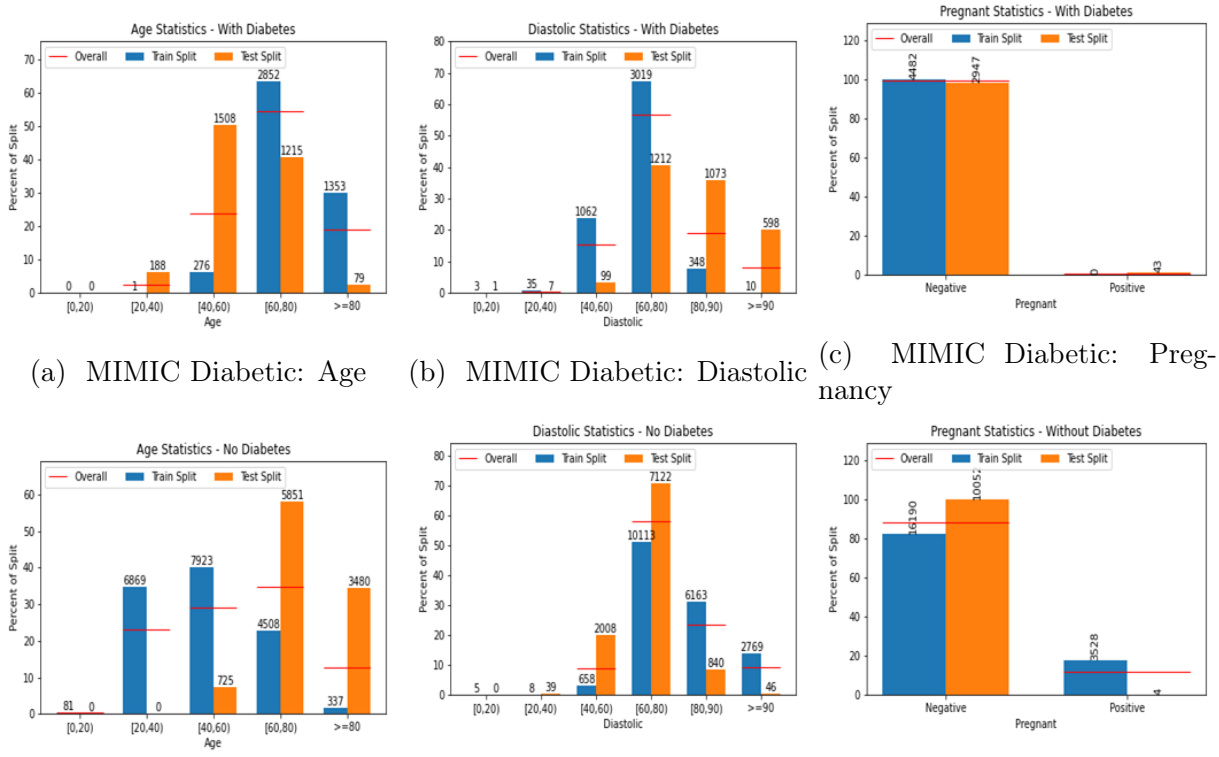
(d) Analysis performed on RHG samples with diabetes, using only AST level and smoking status.

With Diabetes	RIPPER Ruleset (Max 4 Rules)	Decision Tree (Depth=2)
Important Features	-AST > 22 (8) -Smoking (8)	-AST (0.76) -Smoking (0.24)
Accuracy (%)	84.6 ± 0.3	78.5 ± 0.4
Train/Test Precision (%)	88.3 ± 0.3 / 56 ± 2	90.7 ± 0.2 / 40.1 ± 0.9
Train/Test Recall (%)	94.0 ± 0.2 / 38.2 ± 0.8	82.7 ± 0.5 / 57.7 ± 0.5

towards the LS-Train set, those who both smoke and have low AST end up in the LS-Test set. This demonstrates how our framework can uncover informative nontrivial subgroups that can be used to improve performance for minority subgroups in the dataset, such as smokers.

3.3.2 Validating Biases

As with the NLST data, we validated the diabetes heuristic train-test splits visualized in Table 3.2b to see whether they achieve similar generalization gaps as the LS splits. As shown in Table 3.1, the ROC-AUC gaps are similar, at 88.9 ± 0.2 with the manual split versus 99.7 with LS for MIMIC, and 47.9 ± 0.8 with the manual split versus 55.5 with LS for RHG. This shows that the heuristic splits are not only informative from a medically interpretable perspective, but are also useful for creating AI-failure-inducing groups over which de-biasing algorithms can be applied. Finally, since age is a major biasing feature for both datasets, we tested the generalization gaps achieved by simply segregating the data by age (as informed by the LS splits). We showed that training on younger diabetic people and testing on older nondiabetic people induces catastrophic failure of the AI models, with AUC reductions on the order of 90%. Although such extremely spuriously correlated train-test splits are contrived and unlikely to be encountered in reality, this highlights how our framework uncovered that age is a feature over which the MLP model is vulnerable to overreliance on for these datasets, and which AI practitioners should take extra care with. Both the complex manual splits and

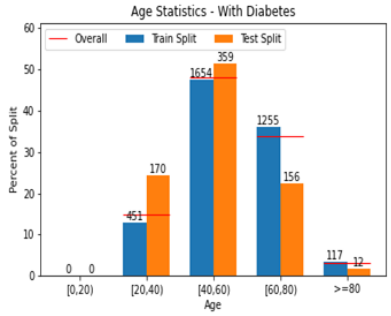


(a) MIMIC Diabetic: Age (b) MIMIC Diabetic: Diastolic (c) MIMIC Diabetic: Pregnancy

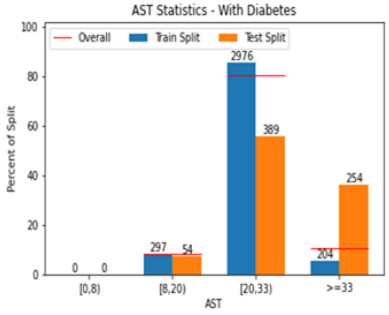
(d) MIMIC Nondiabetic: Age (e) MIMIC Nondiabetic: Diastolic (f) MIMIC Nondiabetic: Pregnancy

Figure 3.5: Statistics of important features in the LS-generated train/test splits for MIMIC.

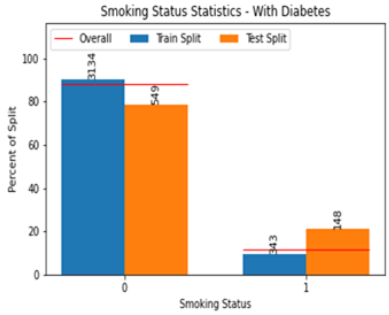
the simple manual splits based on age described in detail in Table 3.8. In order to see how age impacts pre-trained MLPs that predict diabetes, we perform stratified performance evaluations and find that MLPs have a 10% gap in AUC performance between younger and older age groups in the MIMIC dataset, even if trained on a random subset of the data (see Table 3.9).



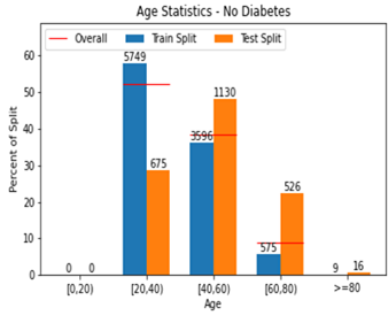
(a) RHG Diabetic: Age



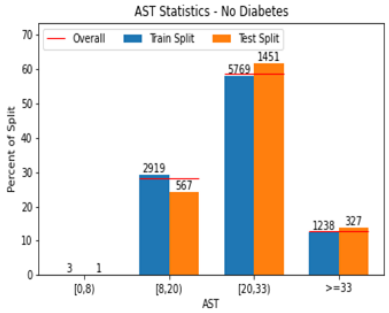
(b) RHG Diabetic: AST



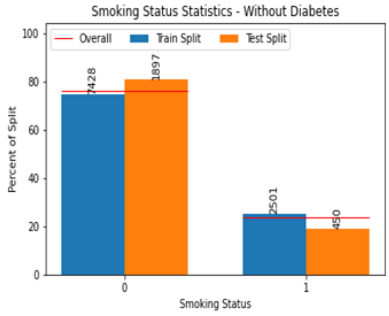
(c) RHG Diabetic: Smoking Status



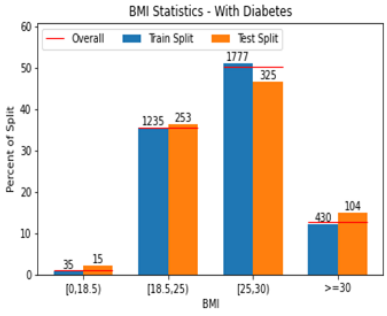
(d) RHG Nondiabetic: Age



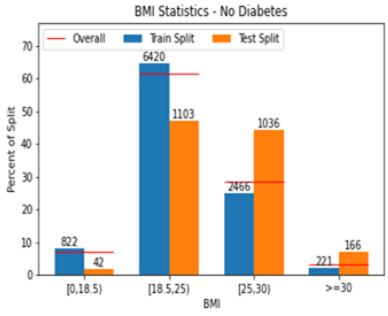
(e) RHG Nondiabetic: AST



(f) RHG Nondiabetic: Smoking Status



(g) RHG Diabetic: BMI



(h) RHG Nondiabetic: BMI

Figure 3.6: Statistics of important features in the LS-generated train/test splits for RHG.

Table 3.8: Test train splits used with diabetes data, along with their generalization gaps and regularization properties (including LS splits and manually curated splits).

(a) For MIMIC data. The complex manual split refers to the splits described by the decision tree in Table 3.2b.

	Generalization Gap in ROC-AUC Percentage (Train AUC – Test AUC)	Split Size Regularization: Train/Test Number of Samples (%)	Label Balance Regularization: Train/Test Samples with Diabetes (%)
Initial Random Split	1.2 (76.4 – 75.2)	27,980 (75.1) / 9,266 (24.9)	5,597 (20.0) / 1,875 (20.2)
LS-Created Split (Best)	99.7 (100.0 – 0.3)	24,200 (65.0) / 13,046 (35.0)	4,482 (18.5) / 2,990 (22.9)
Simple Manual Split: Age ≥ 62, Diabetic OR Age < 62, Nondiabetic → Train Age < 62, Diabetic OR Age ≥ 62, Nondiabetic → Test	96 ± 1 (99.3 ± 0.4 – 3.0 ± 0.8)	21,898 (58.8) / 15,348 (41.2)	5,097 (18.5) / 2,375 (22.9)
Complex Manual Split (As described by decision tree)	88.9 ± 0.2 (98.4 ± 0.2 – 9.4 ± 0.3)	23,626 (63.4) / 13,620 (36.6)	4,077 (17.3) / 3,395 (24.9)

(b) For RHG data. The complex manual split refers to the splits described by the decision tree in Table 3.2b.

	Generalization Gap in ROC-AUC Percentage (Train AUC – Test AUC)	Split Size Regularization: Train/Test Number of Samples (%)	Label Balance Regularization: Train/Test Samples with Diabetes (%)
Initial Random Split	1.2 (76.4 – 75.2)	27,980 (75.1) / 9,266 (24.9)	5,597 (20.0) / 1,875 (20.2)
LS-Created Split (Best)	99.7 (100.0 – 0.3)	24,200 (65.0) / 13,046 (35.0)	4,482 (18.5) / 2,990 (22.9)
Simple Manual Split: Age ≥ 62, Diabetic OR Age < 62, Nondiabetic → Train Age < 62, Diabetic OR Age ≥ 62, Nondiabetic → Test	96.2 ± 1.2 (99.3 ± 0.4 – 3.0 ± 0.8)	21,898 (58.8) / 15,348 (41.2)	5,097 (18.5) / 2,375 (22.9)
Complex Manual Split (As described by decision tree)	88.9 ± 0.2 (98.4 ± 0.2 – 9.4 ± 0.3)	23,626 (63.4) / 13,620 (36.6)	4,077 (17.3) / 3,395 (24.9)

Table 3.9: Stratified performance evaluation (by age) over MLPs trained to predict diabetes. For MIMIC, low age is less than 62; for RHG, low age is less than 45.

	Validation ROC-AUC			Test ROC-AUC		
	Overall	Low Age	High Age	Overall	Low Age	High Age
MIMIC	76.9 ± 0.4	79.5 ± 0.6	69.1 ± 0.3	76.4 ± 0.3	79.4 ± 0.6	69.2 ± 0.6
RHG	91.8 ± 0.3	89.8 ± 0.4	87.9 ± 0.4	91.5 ± 0.3	89.7 ± 0.5	87.7 ± 0.6

3.4 Biomolecular Results

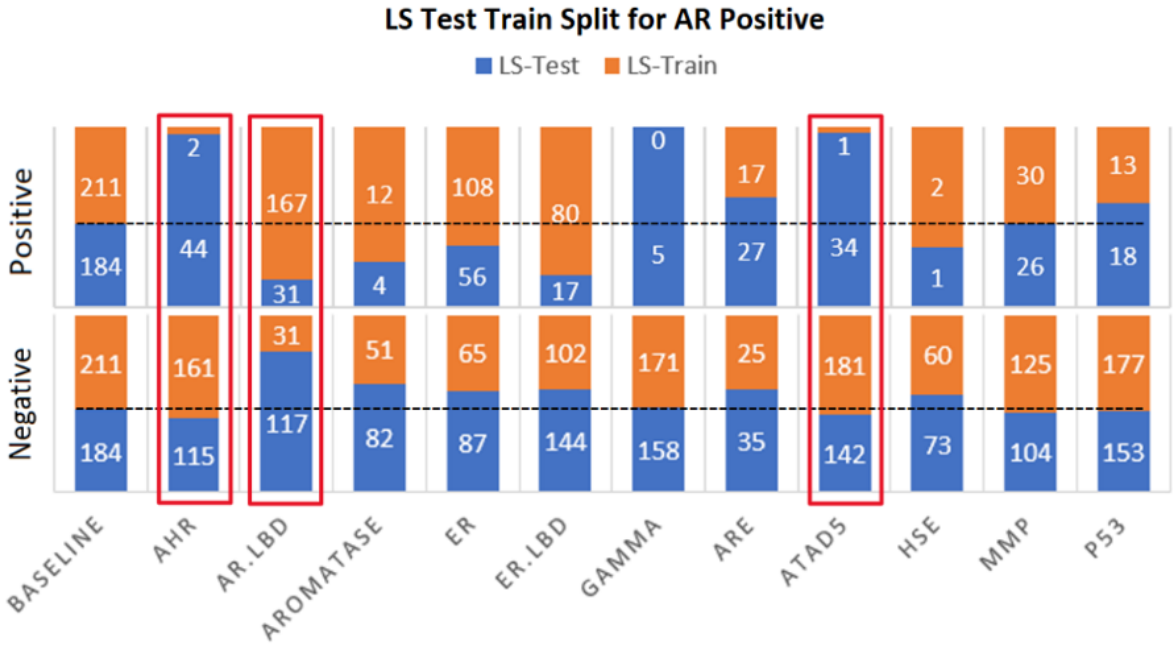
3.4.1 Uncovered Biases

As highlighted in Table 3.2 (and detailed in Table 3.10), some of the primary features revealed to cause bias when predicting NR-AR were the fellow assays SR-ATAD5, SR-AR.LBD, and NR-AhR for molecules that are NR-AR positive, or the fellow assays NR-AhR, SR-HSE, and SR-ARE for molecules that are NR-AR negative.

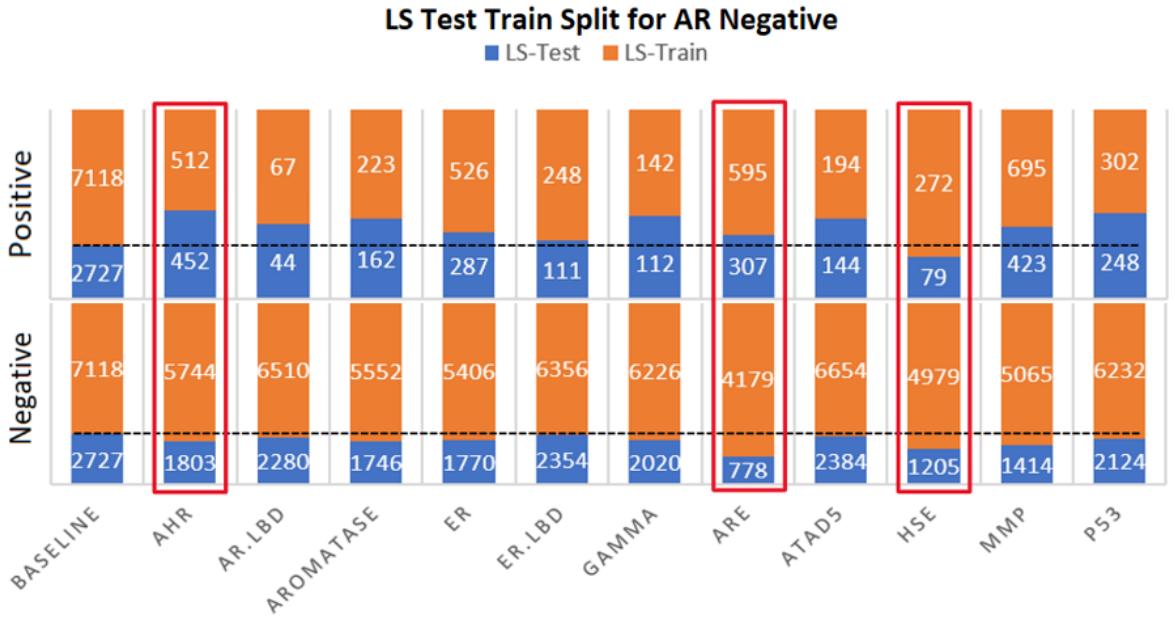
These uncovered biases are corroborated for the case of NR-AR positive molecules in Figure 3.7a, where it is shown that the SR-ATAD5 and NR-AhR positive molecules are overrepresented in the LS-Test set while SR-AR.LBD positive molecules are overrepresented in the LS-Train set. To investigate biases at the intersection of these features, we ran the RIPPER ruleset algorithm for the NR-AR positive case over just these features (presented in Table 3.11), obtaining a ruleset predicting that datapoints that are not SR-AR.LBD positive are likely to be in the test set even if they are negative for NR-AhR. This is not immediately obvious by investigating individual features separately without the interpretable models. However, it is confirmed by constructing the decision tree in Table 3.2b, then using it to segregate points, and then noting that the leaf node for points that are neither AR.LBD positive nor AhR positive is biased to the LS-Test set. This further demonstrates how interpretable models can be used to efficiently understand how features interact better than by just looking at individual feature statistics in the LS splits separately.

3.4.2 Validating Biases

As summarized in Table 3.1, we manually curated simplified test/train splits segregated by whether a molecule is positive or negative for NR-AhR (described in more detail in Table 3.12). This achieves a generalization gap on the order of 30 percent when training an MLP to predict the NR-AR assay, which is high but also less than the generalization gap achieved by LS. This demonstrates both that our framework successfully identifies biasing features, and that the splits identified by LS are more expressive and contain more informative and nontrivial biases than can be uncovered by investigating single features at a time.



(a) AR Positive



(b) AR Negative

Figure 3.7: Statistics of various Tox21 assays in the LS-generated train/test splits. For a given assay, the top row considers samples that are positive for it and the bottom row considers negative samples. The baseline column refers to all samples with the relevant AR label. Numbers indicate the number of data points. Important biasing features are encircled in red.

Table 3.10: Important features selected by various interpretable models for Tox21, along with the accuracies of these models' predictions.

(a) Analysis performed on AR- samples.

AR-	Ripper Ruleset (Max 4 rules)	Logistic Regression	Decision Tree (Depth=4)	Rationale Model (Sparsity=4)
Important Features	-AhR+ (7) -MMP+ (2) -HSE- (2)	-HSE+ (-3.5) -ARE- (-2.1) -HSE- (-1.9)	-ARE (0.34) -HSE (0.25) -AhR (0.16)	-HSE- (5.9) -AhR+ (3.5) -ARE- (3.0)
Accuracy (%)	68.9 ± 0.3	66.2 ± 0.2	65.8 ± 0.6	64.2 ± 1.0
Train/Test Precision (%)	82.8 ± 0.3 / 45.4 ± 0.3	84.1 ± 0.3 / 43.2 ± 0.5	84.5 ± 0.2 / 43.0 ± 0.5	83 ± 1 / 41.0 ± 0.9
Train/Test Recall (%)	71.9 ± 0.5 / 60.8 ± 0.7	65.6 ± 0.3 / 67.9 ± 0.7	64 ± 1 / 69.4 ± 0.9	64 ± 3 / 64 ± 6

(b) Analysis performed on AR+ samples.

AR+	RIPPER Ruleset (Max 4 Rules)	Logistic Regression	Decision Tree (Depth=4)	Rationale Model (Sparsity=4)
Important Features	-AR.LBD- (8) -ER.LBD- (7) -AhR+ (6)	-AR.LBD+ (-8.0) -ATAD5+ (6.1) -AhR+ (6.0)	-AR.LBD (0.68) -Aromatase (0.07) -ATAD5 (0.06)	-ER.LBD+ (4.0) -AR.LBD+ (3.0) -AhR- (3.0)
Accuracy (%)	82 ± 1	81.1 ± 0.9	79 ± 1	73 ± 3
Train/Test Precision (%)	84 ± 1 / 80. ± 2	83 ± 1 / 79 ± 1	81 ± 2 / 78 ± 2	77 ± 4 / 73 ± 4
Train/Test Recall (%)	83 ± 2 / 81 ± 2	81 ± 2 / 81 ± 2	79 ± 2 / 79 ± 2	70 ± 7 / 77 ± 5

Table 3.11: Important features selected by various interpretable models for diabetes data, when using only the most relevant features.

(a) Analysis performed on AR-, using only AhR, ARE, and HSE as inputs.

AR-	RIPPER Ruleset (Max 4 Rules)	Decision Tree (Depth=2)
Important Features	-AhR+ (8) -HSE- (1)	-ARE- (0.84) -AhR- (0.08) -AhR+ (0.04)
Accuracy (%)	64.7 ± 0.6	59.8 ± 0.6
Train/Test Precision (%)	83.8 ± 0.4 / 41.8 ± 0.5	86.2 ± 0.5 / 39.0 ± 0.4
Train/Test Recall (%)	63 ± 1 / 68 ± 2	53 ± 1 / 78 ± 1

(b) Analysis performed on AR+ samples, using only AhR, ATAD5, and AR.LBD as inputs.

AR+	RIPPER Ruleset (Max 4 Rules)	Decision Tree (Depth=2)
Important Features	-AhR+ (8) -AhR- (5) -ATAD5+ (3)	-AR.LBD+ (0.91) -AhR+ (0.07) -ATAD5+ (0.01)
Accuracy (%)	79 ± 2	81.5 ± 0.9
Train/Test Precision (%)	81 ± 3 / 79 ± 2	85 ± 1 / 78 ± 2
Train/Test Recall (%)	81 ± 2 / 78 ± 3	80. ± 1 / 84 ± 1

Table 3.12: Train-test splits used with Tox21 data, along with their generalization gaps and regularization properties (including both LS and manually curated splits).

	Generalization Gap in ROC-AUC Percentage (Train AUC – Test AUC)	Split Size Regularization Train/Test Number of Samples (%)	Label Balance Regularization: Train/Test Samples with AR+ (%)
Initial Random Split	1 (79 – 78)	7,677 (75.0) / 2,563 (25.0)	291 (3.8) / 104 (4.2)
LS-Created Split (Best)	59 (99.95 – 41.25)	7,329 (71.6) / 2,911 (28.4)	211 (2.9) / 184 (6.3)
Manual Split: AhR- →Train AhR+ → Test	31 ± 3 (85 ± 1 – 53 ± 3)	7823 (88.6) / 1010 (11.4)	276 (3.5) / 46 (4.6)

Chapter 4

Discussion

We have proposed a bias detection framework to uncover nontrivial biases in clinical AI training data, automatically elucidating biased subgroups in the data in terms of clinical, demographic, or other attributes. The proposed approach does not require human-specified heuristics, is task-agnostic, and is adaptable to diverse AI models and data modalities. We have demonstrated the effectiveness of our framework on a variety of real-world datasets. Without any human intervention, it identified imaging device as a major source of bias in the NLST lung cancer dataset, as well as biases caused by patient BMI but which are dependent on the device type. We validated that retraining models with the identified risky data subgroups does indeed produce poorly performing models. More pressingly, upon performing stratified performance evaluations, we confirmed that these biases manifest as performance stratification over different data subgroups in the real world Sybil lung cancer risk prediction AI model. We similarly identified high-risk precision subgroups on the basis of age and systolic pressure in two commonly used diabetes datasets, and we validated that a common AI model architecture is vulnerable to shortcut learning over these features. We also identified pregnant patients as an unexpected minority subgroup at risk for shortcut learning in these data, flagging this group to algorithm developers who may otherwise be oblivious to biases until after algorithm implementation.

Many regulations [65] outline the need to prevent bias but do not mention which tools can be used to fight bias. Through our experiments, we demonstrated that our framework's use as such a tool in the real world is effective, flexible, and complementary to existing tools such as stratified performance evaluations and de-biasing algorithms. It can be applied to existing models to create hypotheses about potentially biased subgroups over which the model might fail, which can then be tested using stratified performance evaluations. Such evaluations can then be used to decide when or whether to trust the model [4]. Alternatively, the framework can be applied before training new models to be aware of high risk subgroups in the data over which the model could fail. By increasing the proportion of training done over these high-risk subgroups, imbalanced learning methods [12], [35]–[37] can then be used to create better models. Similarly, the subgroups can be used to de-bias models using algorithms such as Group DRO or DANN. Most importantly, given the competing definitions of fairness and the limitations of de-biasing algorithms, our framework can be used to better understand and then minimize the cause of bias at the source [4], [5], [12], [14]. For example, this could mean changing the training objective, relabelling data more accurately, or collecting more

data for underrepresented demographics.

We have empirically demonstrated that our framework can uncover a wider range of biases than error auditing, which was not able to capture device type as a source of bias in the NLST dataset. This demonstrates how our framework can be a more sensitive bias detection method than existing subgroup detection methods—such as algorithms that investigate a model's mistakes on its training data distribution [6], [26], [43]–[46] or investigate an AI model's internal representation space [6], [15]—because it is not limited by the fact that a model's mistakes and internal representations are not always predictive of biases [6], [17], [29]. Finally, our innovation of using interpretable models for analyzing subgroups overcomes the limitation that existing methods are impractical and labour intensive, forcing clinicians to manually investigate uncovered subgroups [6]. The use of interpretable models could even be an extension to existing bias subgroup detection methods, making them complementary to the Learning to Split algorithm.

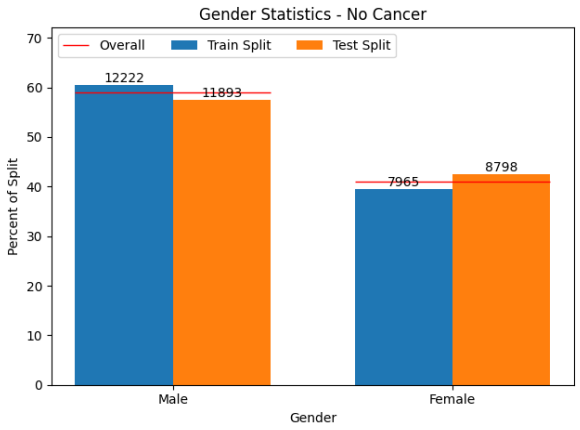
In this thesis, we have discussed findings obtained using the decision tree, logistics regression, and rationale interpretable models in Chapter 3. We also point out that different algorithms have different strengths and complement each other. For example, whereas rationalization models can highlight a few specific features, logistic regression [66] can be used to understand how significant all of the different features are relative to each other. There are also many more possible options for interpretable algorithms, and many of these are freely available online, such as through the scikit-learn Python programming library [66]. One possible interpretable model to investigate more is the invariant rationale model [48], which aims to ensure that selected features are actually causal to the decision making process instead of just correlated with it. While we did investigate this model, we did not emphasize its findings due to limitations in the subjective nature of its implementation. However, we think it is worth researching more in the future, along with other methods more broadly that aim to decouple causation and correlation when uncovering biased features. One possible extension to our work would be to use interpretable models that can function over input images directly, such as the CT Scans themselves, instead of just metadata. There exist interpretable models that highlight which pixels in an image contribute to the model's decisions [67]. Such methods have been used to confirm the existence of spurious correlations in image data, such as radiologists' markings [19] or surgical markings [22]; integrating them as interpretable models in our framework would help guide their use towards biased subsets in a precision manner. Using interpretable models directly over images would further overcome a limitation of our experiments, which is that the interpretability models we have tested are limited to uncovering biases that can be described by the metadata provided with the dataset.

Along these lines, one of the remaining limitations of our work thus far is that it only provides knowledge of which subgroups might be biased in terms of clinical or demographic attributes. However, it does not always provide a mechanistic understanding of these biases or why subgroups may be harder to generalize to, such as why the Sybil model performs differently on different X-ray scanners. Similarly, it does not differentiate between biases inherent to the data and those caused by poor choice of AI model architecture [7], [11]. While interpretable models that can analyze images directly will help towards answering these questions, additional future work should focus on finding ways to translate knowledge of subgroups into mechanistic understanding of biases. Such understanding would help enable

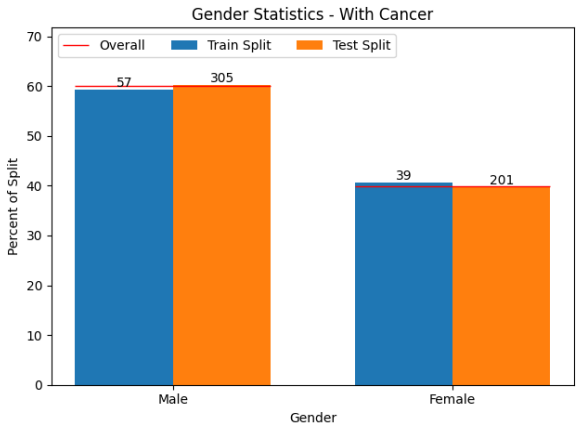
AI tools to achieve equitable healthcare delivery with high performance in all subgroups and minimize the chance of unexpected failures.

Appendix A

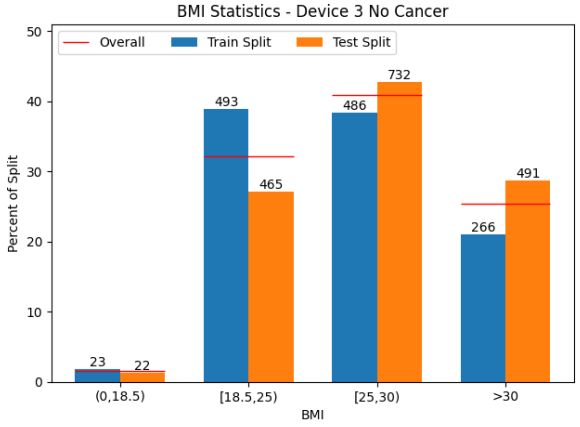
Additional LS Split Statistics



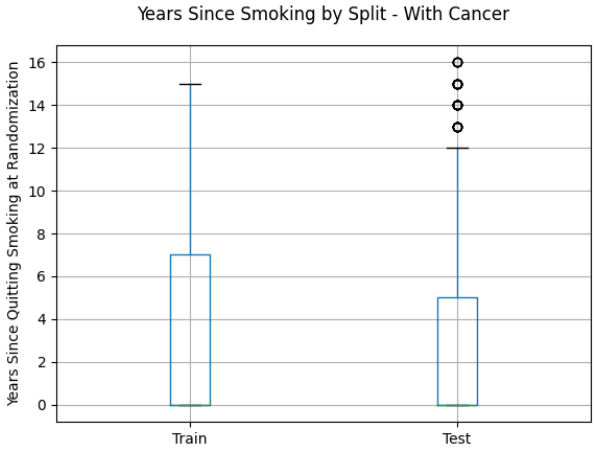
(a) By Gender - No Cancer



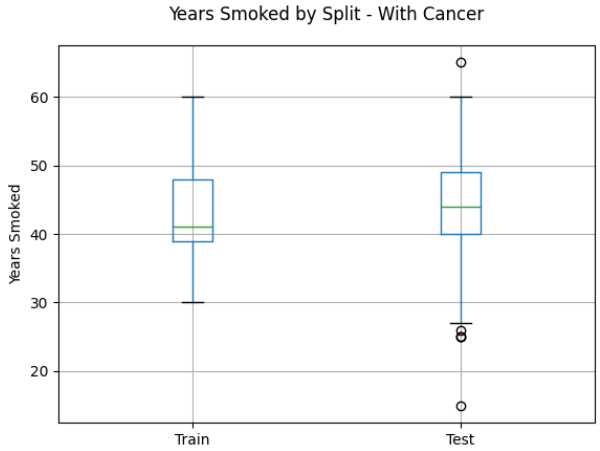
(b) By Gender - With Cancer



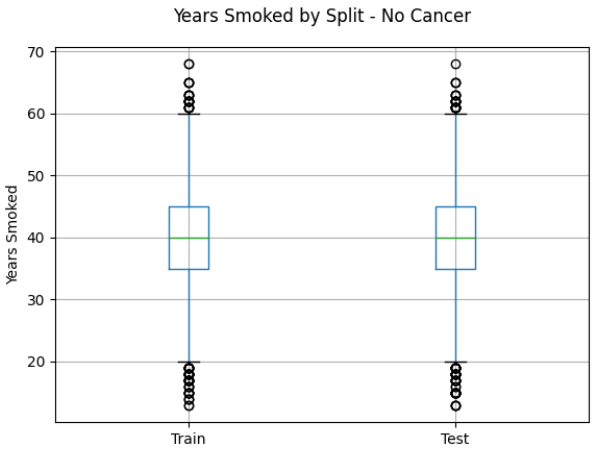
(c) BMI Barchart - Only Datapoints from Device 3 without cancer



(d) Years Since Quit Smoking - With Cancer

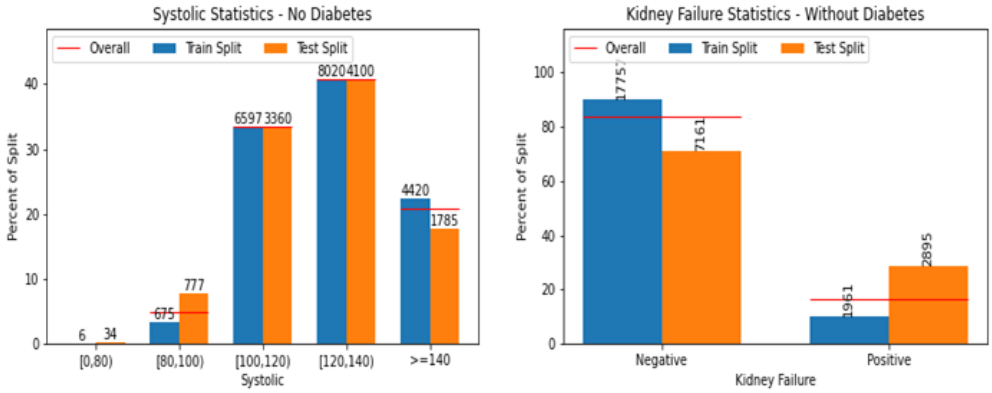


(e) Years Smoked - With Cancer



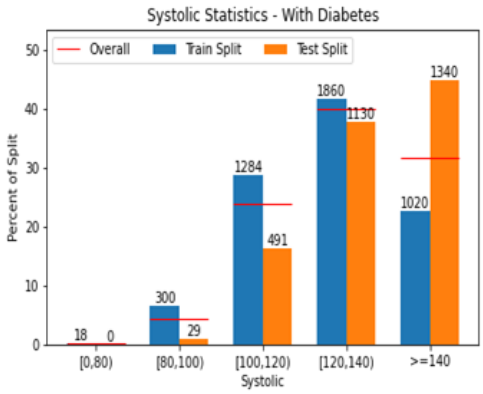
(f) Years Smoked - No Cancer

Figure A.1: Some additional feature statistics in the LS-generated train/test splits for NLST with Sybil.

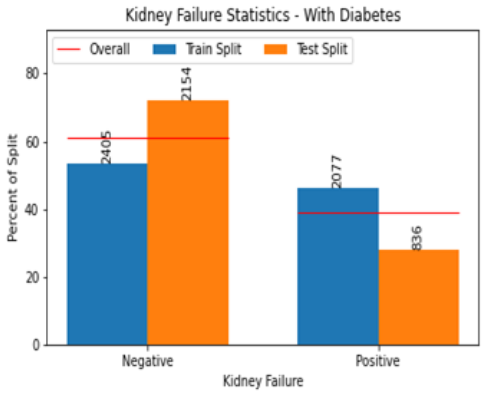


(a) Systolic Pressure - No Diabetes

(b) Kidney Failure - No Diabetes

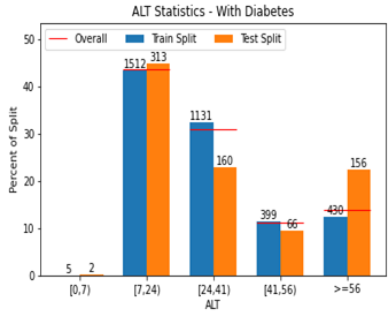


(c) Systolic Pressure - Diabetes

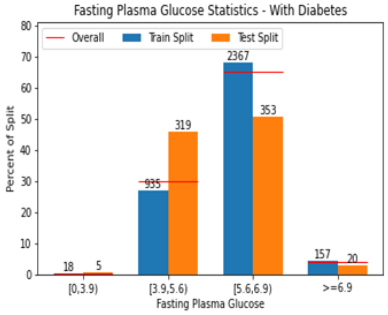


(d) Kidney Failure - Diabetes

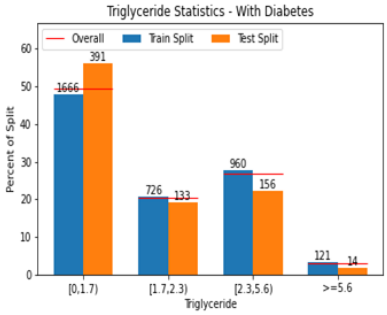
Figure A.2: Some additional feature statistics in the LS-generated train/test splits for MIMIC.



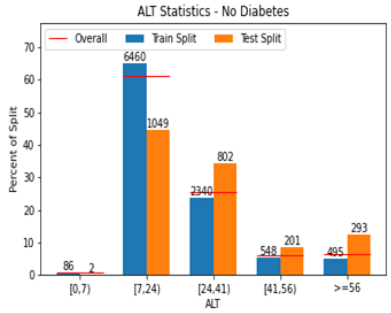
(a) RHG Diabetic: ALT



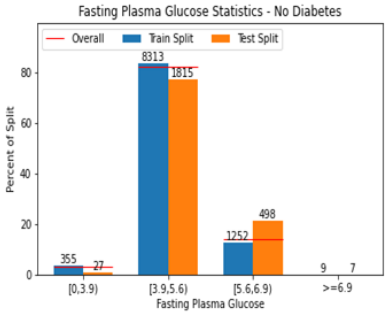
(b) RHG Diabetic: FPG



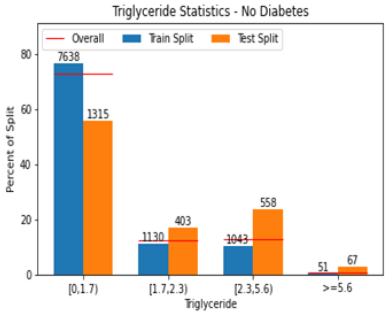
(c) RHG Diabetic: Triglyceride Levels



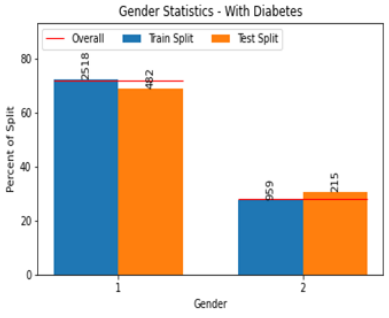
(d) RHG Nondiabetic: ALT



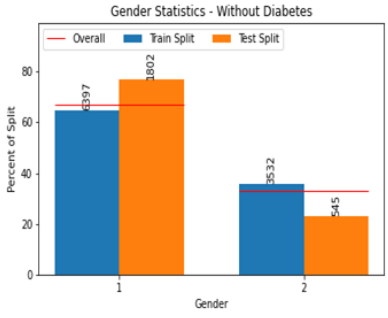
(e) RHG Nondiabetic: FPG



(f) RHG Nondiabetic: Triglyceride Levels



(g) RHG Diabetic: Gender



(h) RHG Nondiabetic: Gender

Figure A.3: Some additional feature statistics in the LS-generated train/test splits for RHG.

References

[1] A. Zhang, L. Xing, J. Zou, and J. C. Wu, "Shifting machine learning for healthcare from development to deployment and from models to data," en, *Nat. Biomed. Eng.*, vol. 6, no. 12, pp. 1330–1345, Dec. 2022.

[2] G. Varoquaux and V. Cheplygina, "Machine learning for medical imaging: Methodological failures and recommendations for the future," *npj Digital Medicine*, vol. 5, no. 1, p. 48, Apr. 2022, ISSN: 2398-6352. DOI: 10.1038/s41746-022-00592-y. URL: https://doi.org/10.1038/s41746-022-00592-y.

[3] T. Eche, L. H. Schwartz, F.-Z. Mokrane, and L. Dercle, "Toward generalizability in the deployment of artificial intelligence in radiology: Role of computation stress testing to overcome underspecification," *Radiology: Artificial Intelligence*, vol. 3, no. 6, Nov. 2021. DOI: 10.1148/ryai.2021210097. URL: https://doi.org/10.1148/ryai.2021210097.

[4] S. R. Pfohl, A. Foryciarz, and N. H. Shah, "An empirical characterization of fair machine learning for clinical risk prediction," *J Biomed Inform*, vol. 113, p. 103 621, Jan. 2021.

[5] S. R. Pfohl, H. Zhang, Y. Xu, A. Foryciarz, M. Ghassemi, and N. H. Shah, "A comparison of approaches to improve worst-case predictive model performance over patient subpopulations," *Sci Rep*, vol. 12, no. 1, p. 3254, Feb. 2022.

[6] L. Oakden-Rayner, J. Dunnmon, G. Carneiro, and C. Re, "Hidden stratification causes clinically meaningful failures in machine learning for medical imaging," in *Proceedings of the ACM Conference on Health, Inference, and Learning*, ser. CHIL '20, Toronto, Ontario, Canada: Association for Computing Machinery, 2020, pp. 151–159, ISBN: 9781450370462. DOI: 10.1145/3368555.3384468. URL: https://doi.org/10.1145/3368555.3384468.

[7] I. Banerjee, K. Bhattacharjee, J. L. Burns, H. Trivedi, S. Purkayastha, L. Seyyed-Kalantari, B. N. Patel, R. Shiradkar, and J. Gichoya, ""shortcuts" causing bias in radiology artificial intelligence: Causes, evaluation, and mitigation," en, *J. Am. Coll. Radiol.*, vol. 20, no. 9, pp. 842–851, Sep. 2023.

[8] L. Seyyed-Kalantari, H. Zhang, M. B. A. McDermott, I. Y. Chen, and M. Ghassemi, "Underdiagnosis bias of artificial intelligence algorithms applied to chest radiographs in under-served patient populations," *Nature Medicine*, vol. 27, no. 12, pp. 2176–2182, Dec. 2021, ISSN: 1546-170X. DOI: 10.1038/s41591-021-01595-0. URL: https://doi.org/10.1038/s41591-021-01595-0.

[9] C. Jones, D. C. Castro, F. De Sousa Ribeiro, O. Oktay, M. McCradden, and B. Glocker, "A causal perspective on dataset bias in machine learning for medical imaging," en, *Nat. Mach. Intell.*, Feb. 2024.

[10] Y. Yang, H. Zhang, D. Katabi, and M. Ghassemi, "Change is hard: A closer look at subpopulation shift," *arXiv preprint arXiv:2302.12254*, 2023.

[11] N. Mehrabi, F. Morstatter, N. Saxena, K. Lerman, and A. Galstyan, "A survey on bias and fairness in machine learning," *ACM computing surveys (CSUR)*, vol. 54, no. 6, pp. 1–35, 2021.

[12] H. Zhang, N. Dullerud, K. Roth, L. Oakden-Rayner, S. Pfohl, and M. Ghassemi, "Improving the fairness of chest x-ray classifiers," in *Proceedings of the Conference on Health, Inference, and Learning*, G. Flores, G. H. Chen, T. Pollard, J. C. Ho, and T. Naumann, Eds., ser. Proceedings of Machine Learning Research, vol. 174, PMLR, Jul. 2022, pp. 204–233. URL: https://proceedings.mlr.press/v174/zhang22a.html.

[13] A. Brown, N. Tomasev, J. Freyberg, Y. Liu, A. Karthikesalingam, and J. Schrouff, "Detecting shortcut learning for fair medical AI using shortcut testing," en, *Nat. Commun.*, vol. 14, no. 1, p. 4314, Jul. 2023.

[14] Y. Yang, H. Zhang, J. W. Gichoya, D. Katabi, and M. Ghassemi, "The limits of fair medical imaging AI in real-world generalization," en, *Nat. Med.*, Jun. 2024.

[15] N. Sohoni, J. A. Dunnmon, G. Angus, A. Gu, and C. Ré, "No subclass left behind: Fine-grained robustness in coarse-grained classification problems," in *Proceedings of the 34th International Conference on Neural Information Processing Systems*, ser. NIPS '20, Vancouver, BC, Canada: Curran Associates Inc., 2020, ISBN: 9781713829546.

[16] K. B. Ahmed, G. M. Goldgof, R. Paul, D. B. Goldgof, and L. O. Hall, "Discovery of a generalization gap of convolutional neural networks on covid-19 x-rays classification," *IEEE Access*, vol. 9, pp. 72 970–72 979, 2021. DOI: 10.1109/ACCESS.2021.3079716.

[17] Y. Bao and R. Barzilay, *Learning to split for automatic bias detection*, 2022. arXiv: 2204.13749 [cs.LG].

[18] R. Geirhos, J.-H. Jacobsen, C. Michaelis, R. Zemel, W. Brendel, M. Bethge, and F. A. Wichmann, "Shortcut learning in deep neural networks," *Nature Machine Intelligence*, vol. 2, no. 11, pp. 665–673, 2020.

[19] J. R. Zech, M. A. Badgeley, M. Liu, A. B. Costa, J. J. Titano, and E. K. Oermann, "Variable generalization performance of a deep learning model to detect pneumonia in chest radiographs: A cross-sectional study," *PLoS medicine*, vol. 15, no. 11, e1002683, 2018.

[20] Z. Obermeyer, B. Powers, C. Vogeli, and S. Mullainathan, "Dissecting racial bias in an algorithm used to manage the health of populations," *Science*, vol. 366, no. 6464, pp. 447–453, 2019.

[21] J. Irvin, P. Rajpurkar, M. Ko, *et al.*, "Chexpert: A large chest radiograph dataset with uncertainty labels and expert comparison," English (US), in *33rd AAAI Conference on Artificial Intelligence*, AAAI Press, 2019, pp. 590–597.

[22] J. K. Winkler, C. Fink, F. Toberer, *et al.*, "Association between surgical skin markings in dermoscopic images and diagnostic performance of a deep learning convolutional neural network for melanoma recognition," *JAMA Dermatology*, vol. 155, no. 10, p. 1135, Oct. 2019. DOI: 10.1001/jamadermatol.2019.1735. URL: https://doi.org/10.1001/jamadermatol.2019.1735.

[23] J. W. Gichoya, I. Banerjee, A. R. Bhimireddy, *et al.*, "Ai recognition of patient race in medical imaging: A modelling study," en, *Lancet Digit. Health*, vol. 4, no. 6, e406–e414, Jun. 2022.

[24] B. Glocker, C. Jones, M. Bernhardt, and S. Winzeck, "Algorithmic encoding of protected characteristics in chest x-ray disease detection models," en, *EBioMedicine*, vol. 89, p. 104 467, Mar. 2023.

[25] Y. Bao, S. Chang, and R. Barzilay, "Predict then interpolate: A simple algorithm to learn stable classifiers," *CoRR*, vol. abs/2105.12628, 2021. arXiv: 2105.12628. URL: https://arxiv.org/abs/2105.12628.

[26] E. Creager, J.-H. Jacobsen, and R. Zemel, "Environment inference for invariant learning," in *Proceedings of the 38th International Conference on Machine Learning*, M. Meila and T. Zhang, Eds., ser. Proceedings of Machine Learning Research, vol. 139, PMLR, 18–24 Jul 2021, pp. 2189–2200. URL: https://proceedings.mlr.press/v139/creager21a.html.

[27] C. Clark, M. Yatskar, and L. Zettlemoyer, "Don't take the easy way out: Ensemble based methods for avoiding known dataset biases," in *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, Hong Kong, China: Association for Computational Linguistics, Nov. 2019, pp. 4069–4082. DOI: 10.18653/v1/D19-1418. URL: https://aclanthology.org/D19-1418.

[28] H. He, S. Zha, and H. Wang, "Unlearn dataset bias in natural language inference by fitting the residual," in *Proceedings of the 2nd Workshop on Deep Learning Approaches for Low-Resource NLP (DeepLo 2019)*, Hong Kong, China: Association for Computational Linguistics, Nov. 2019, pp. 132–142. DOI: 10.18653/v1/D19-6115. URL: https://www.aclweb.org/anthology/D19-6115.

[29] P. Izmailov, P. Kirichenko, N. Gruver, and A. G. Wilson, "On feature learning in the presence of spurious correlations," in *Advances in Neural Information Processing Systems*, S. Koyejo, S. Mohamed, A. Agarwal, D. Belgrave, K. Cho, and A. Oh, Eds., vol. 35, Curran Associates, Inc., 2022, pp. 38 516–38 532. URL: https://proceedings.neurips.cc/paper_files/paper/2022/file/fb64a552feda3d981dbe43527a80a07e-Paper-Conference.pdf.

[30] Y. Ganin, E. Ustinova, H. Ajakan, P. Germain, H. Larochelle, F. Laviolette, M. March, and V. Lempitsky, "Domain-adversarial training of neural networks," *Journal of Machine Learning Research*, vol. 17, no. 59, pp. 1–35, 2016. URL: http://jmlr.org/papers/v17/15-239.html.

[31] Y. Li, X. Tian, M. Gong, Y. Liu, T. Liu, K. Zhang, and D. Tao, "Deep domain generalization via conditional invariant adversarial networks," in *Proceedings of the European Conference on Computer Vision (ECCV)*, Sep. 2018.

[32] D. Krueger, E. Caballero, J.-H. Jacobsen, A. Zhang, J. Binas, D. Zhang, R. L. Priol, and A. Courville, "Out-of-distribution generalization via risk extrapolation (rex)," in *Proceedings of the 38th International Conference on Machine Learning*, M. Meila and T. Zhang, Eds., ser. Proceedings of Machine Learning Research, vol. 139, PMLR, 18–24 Jul 2021, pp. 5815–5826. URL: https://proceedings.mlr.press/v139/krueger21a.html.

[33] M. Arjovsky, L. Bottou, I. Gulrajani, and D. Lopez-Paz, *Invariant risk minimization*, 2020. arXiv: 1907.02893 [stat.ML]. URL: https://arxiv.org/abs/1907.02893.

[34] V. S. Lokhande, A. K. Akash, S. N. Ravi, and V. Singh, "Fairalm: Augmented lagrangian method for training fair models with little regret," in *Computer Vision – ECCV 2020*, A. Vedaldi, H. Bischof, T. Brox, and J.-M. Frahm, Eds., Cham: Springer International Publishing, 2020, pp. 365–381, ISBN: 978-3-030-58610-2.

[35] Y. Cui, M. Jia, T.-Y. Lin, Y. Song, and S. Belongie, *Class-balanced loss based on effective number of samples*, 2019. arXiv: 1901.05555 [cs.CV]. URL: https://arxiv.org/abs/1901.05555.

[36] B. Kang, S. Xie, M. Rohrbach, Z. Yan, A. Gordo, J. Feng, and Y. Kalantidis, *Decoupling representation and classifier for long-tailed recognition*, 2020. arXiv: 1910.09217 [cs.CV]. URL: https://arxiv.org/abs/1910.09217.

[37] J. Byrd and Z. C. Lipton, *What is the effect of importance weighting in deep learning?* 2019. arXiv: 1812.03372 [cs.LG]. URL: https://arxiv.org/abs/1812.03372.

[38] M. Lin, T. Li, Y. Yang, *et al.*, "Improving model fairness in image-based computer-aided diagnosis," en, *Nat. Commun.*, vol. 14, no. 1, p. 6261, Oct. 2023.

[39] S. Sagawa, P. W. Koh, T. B. Hashimoto, and P. Liang, "Distributionally robust neural networks for group shifts: On the importance of regularization for worst-case generalization," *arXiv preprint arXiv:1911.08731*, 2019.

[40] J. Ren, C. Yu, s. sheng shunan, X. Ma, H. Zhao, S. Yi, and h. Li, "Balanced meta-softmax for long-tailed visual recognition," in *Advances in Neural Information Processing Systems*, H. Larochelle, M. Ranzato, R. Hadsell, M. Balcan, and H. Lin, Eds., vol. 33, Curran Associates, Inc., 2020, pp. 4175–4186. URL: https://proceedings.neurips.cc/paper_files/paper/2020/file/2ba61cc3a8f44143e1f2f13b2b729ab3-Paper.pdf.

[41] M. Ren, W. Zeng, B. Yang, and R. Urtasun, "Learning to reweight examples for robust deep learning," in *International conference on machine learning*, PMLR, 2018, pp. 4334–4343.

[42] H. Zhang, M. Cissé, Y. N. Dauphin, and D. Lopez-Paz, "Mixup: Beyond empirical risk minimization," *CoRR*, vol. abs/1710.09412, 2017. arXiv: 1710.09412. URL: http://arxiv.org/abs/1710.09412.

[43] E. Z. Liu, B. Haghgoo, A. S. Chen, A. Raghunathan, P. W. Koh, S. Sagawa, P. Liang, and C. Finn, "Just train twice: Improving group robustness without training group information," in *Proceedings of the 38th International Conference on Machine Learning*, M. Meila and T. Zhang, Eds., ser. Proceedings of Machine Learning Research, vol. 139, PMLR, 18–24 Jul 2021, pp. 6781–6792.

[44] J. Nam, H. Cha, S. Ahn, J. Lee, and J. Shin, "Learning from failure: De-biasing classifier from biased classifier," in *Advances in Neural Information Processing Systems*, H. Larochelle, M. Ranzato, R. Hadsell, M. Balcan, and H. Lin, Eds., vol. 33, Curran Associates, Inc., 2020, pp. 20 673–20 684. URL: https://proceedings.neurips.cc/paper_files/paper/2020/file/eddc3427c5d77843c2253f1e799fe933-Paper.pdf.

[45] V. Sanh, T. Wolf, Y. Belinkov, and A. M. Rush, "Learning from others' mistakes: Avoiding dataset biases without modeling them," *CoRR*, vol. abs/2012.01300, 2020. arXiv: 2012.01300. URL: https://arxiv.org/abs/2012.01300.

[46] P. Lahoti, A. Beutel, J. Chen, K. Lee, F. Prost, N. Thain, X. Wang, and E. H. Chi, "Fairness without demographics through adversarially reweighted learning," in *Proceedings of the 34th International Conference on Neural Information Processing Systems*, ser. NIPS '20, Vancouver, BC, Canada: Curran Associates Inc., 2020, ISBN: 9781713829546.

[47] T. Lei, R. Barzilay, and T. Jaakkola, "Rationalizing neural predictions," *arXiv preprint arXiv:1606.04155*, 2016.

[48] S. Chang, Y. Zhang, M. Yu, and T. Jaakkola, "Invariant rationalization," in *Proceedings of the 37th International Conference on Machine Learning*, H. D. III and A. Singh, Eds., ser. Proceedings of Machine Learning Research, vol. 119, PMLR, 13–18 Jul 2020, pp. 1448–1458. URL: https://proceedings.mlr.press/v119/chang20c.html.

[49] W. W. Cohen, "Fast effective rule induction," in *Machine Learning Proceedings 1995*, A. Prieditis and S. Russell, Eds., San Francisco (CA): Morgan Kaufmann, 1995, pp. 115–123, ISBN: 978-1-55860-377-6. DOI: https://doi.org/10.1016/B978-1-55860-377-6.50023-2. URL: https://www.sciencedirect.com/science/article/pii/B9781558603776500232.

[50] E. Jang, S. Gu, and B. Poole, *Categorical reparameterization with gumbel-softmax*, 2017. arXiv: 1611.01144 [stat.ML]. URL: https://arxiv.org/abs/1611.01144.

[51] National Lung Screening Trial Research Team, "Reduced lung-cancer mortality with low-dose computed tomographic screening," *New England Journal of Medicine*, vol. 365, no. 5, pp. 395–409, 2011.

[52] P. G. Mikhael, J. Wohlwend, A. Yala, L. Karstens, J. Xiang, A. K. Takigami, P. P. Bourgouin, P. Chan, S. Mrah, W. Amayri, *et al.*, "Sybil: A validated deep learning model to predict future lung cancer risk from a single low-dose chest computed tomography," *Journal of Clinical Oncology*, vol. 41, no. 12, pp. 2191–2200, 2023.

[53] A. E. W. Johnson, L. Bulgarelli, L. Shen, *et al.*, "MIMIC-IV, a freely accessible electronic health record dataset," en, *Sci. Data*, vol. 10, no. 1, p. 1, Jan. 2023.

[54] A. L. Goldberger, L. A. 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, "PhysioBank, PhysioToolkit, and PhysioNet: Components of a new research resource for complex physiologic signals," en, *Circulation*, vol. 101, no. 23, E215–20, Jun. 2000.

[55] A. Johnson, L. Bulgarelli, T. Pollard, S. Horng, L. A. Celi, and R. Mark, *MIMIC-IV (version 2.2)*, 2023. DOI: https://doi.org/10.13026/6mm1-ek67.

[56] Y. Chen, X.-P. Zhang, J. Yuan, B. Cai, X.-L. Wang, X.-L. Wu, Y.-H. Zhang, X.-Y. Zhang, T. Yin, X.-H. Zhu, *et al.*, "Association of body mass index and age with incident diabetes in chinese adults: A population-based cohort study," *BMJ open*, vol. 8, no. 9, e021768, 2018.

[57] United States Centers for Medicare Medicaid Services, *International Classification of Diseases,Ninth Revision, Clinical Modification (ICD-9-CM)*, 2021. URL: https://archive.cdc.gov/#/details?url=https://www.cdc.gov/nchs/icd/icd9cm.htm.

[58] United States Centers for Disease Control and Prevention, *ICD-10*, 2023. URL: https://www.cms.gov/medicare/coding-billing/icd-10-codes.

[59] M. M. Hatmal, S. M. Abderrahman, W. Nimer, Z. Al-Eisawi, H. J. Al-Ameer, M. A. I. Al-Hatamleh, R. Mohamud, and W. Alshaer, "Artificial neural networks model for predicting type 2 diabetes mellitus based on VDR gene FokI polymorphism, lipid profile and demographic data," en, *Biology (Basel)*, vol. 9, no. 8, p. 222, Aug. 2020.

[60] A. E.-S. El-Bashbishy and H. M. El-Bakry, "Pediatric diabetes prediction using deep learning," en, *Sci. Rep.*, vol. 14, no. 1, p. 4206, Feb. 2024.

[61] H. Naz and S. Ahuja, "Deep learning approach for diabetes prediction using PIMA indian dataset," en, *J. Diabetes Metab. Disord.*, vol. 19, no. 1, pp. 391–403, Jun. 2020.

[62] R. Huang, M. Xia, D.-T. Nguyen, T. Zhao, S. Sakamuru, J. Zhao, S. A. Shahane, A. Rossoshek, and A. Simeonov, "Tox21challenge to build predictive models of nuclear receptor and stress response pathways as mediated by exposure to environmental chemicals and drugs," *Frontiers in Environmental Science*, vol. 3, 2016, ISSN: 2296-665X. DOI: 10.3389/fenvs.2015.00085. URL: https://www.frontiersin.org/articles/10.3389/fenvs.2015.00085.

[63] A. Mayr, G. Klambauer, T. Unterthiner, and S. Hochreiter, "Deeptox: Toxicity prediction using deep learning," *Frontiers in Environmental Science*, vol. 3, 2016, ISSN: 2296-665X. DOI: 10.3389/fenvs.2015.00080. URL: https://www.frontiersin.org/articles/10.3389/fenvs.2015.00080.

[64] E. Osher and N. Stern, "Diastolic Pressure in Type 2 Diabetes: Can target systolic pressure be reached without "diastolic hypotension"?" *Diabetes Care*, vol. 31, no. Supplement$_2$, S249–S254, Feb. 2008, ISSN: 0149-5992. DOI: 10.2337/dc08-s262. eprint: https://diabetesjournals.org/care/article-pdf/31/Supplement_2/S249/469512/zdc1020800s249.pdf. URL: https://doi.org/10.2337/dc08-s262.

[65] U.S. Food and Drug Administration, *Artificial intelligence and medical products how cber, cder, cdrh, and ocp are working together*, Mar. 2024. URL: https://www.fda.gov/media/177030/download.

[66] F. Pedregosa, G. Varoquaux, A. Gramfort, *et al.*, "Scikit-learn: Machine learning in Python," *Journal of Machine Learning Research*, vol. 12, pp. 2825–2830, 2011.

[67] M. Ancona, E. Ceolini, C. Öztireli, and M. Gross, "Towards better understanding of gradient-based attribution methods for deep neural networks," in *International Conference on Learning Representations*, 2018. URL: https://openreview.net/forum?id=Sy21R9JAW.