Novel Examination of Interpretable Surrogates and Adversarial Robustness in Machine Learning

SADIA CHOWDHURY

A THESIS SUBMITTED TO THE FACULTY OF GRADUATE STUDIES IN PARTIAL FULFILLMENT OF THE REQUIREMENTS FOR THE DEGREE OF MASTER OF SCIENCE

GRADUATE PROGRAM IN COMPUTER SCIENCE YORK UNIVERSITY
TORONTO, ONTARIO

February 2021

Abstract

Novel Examination of Interpretable Surrogates and Adversarial Robustness in Machine Learning

Sadia Chowdhury

The lack of transparent output behavior is a significant source of mistrust in many of the currently most successful machine learning tools. Concern arises particularly in situations where the data generation changes, for example under marginal shift or under adversarial manipulations. Training a (human-)interpretable surrogate model for a black-box predictor is a common approach for providing insights into the blackbox's predictive behavior. We analyze the use of decision trees for indicating marginal shift. We then investigate the role of the data generation of the student model for the validity of the interpretable surrogate. We use decision trees as part of the teacher-student framework and empirically investigate the validity of decision trees as both local and global interpretation methods.

While investigating local decision trees, we observed that the decision boundaries of the blackbox model was often sitting close to the original data manifold. This makes those regions vulnerable to imperceptible perturbations and can falsely flip the network's prediction. Hence, we aim to provide a framework for determining whether a model's label change under small perturbation is justified (and when it is not). We carefully argue that adversarial robustness should be defined as a locally adaptive measure complying with the underlying distribution. We then suggest a definition for an adaptive robust loss, an empirical version of it and a resulting data-augmentation framework.

Acknowledgments

At the very beginning, I would like to take a moment to express my gratitude towards everyone who directly and indirectly played a role to help me complete this thesis.

Firstly, I am very grateful to my supervisor Dr. Ruth Urner for giving me the opportunity to work with her for my Master's program. Her in-depth knowledge, patience, and guidance throughout the last couple of years helped me to grow both personally and academically. I would like to thank her for her invaluable help and encouragement without which this work would not have been possible. I hope to keep working with her in the future.

I wish to sincerely thank the respectable committee members Dr. Aijun An and Dr. Enamul Hoque Prince for their valuable contributions to my thesis.

I am also grateful to my fellow lab member Chester for his countless help and insightful guidance.

Last but not the least, I would like to express my profound gratitude to my family and friends. My mother encouraged me to pursue my masters while my father supported me throughout the journey. My special thanks to my sister for always looking out for me. It would not have been possible at all if it had not been for my family. Lastly, I would also like to thank my best friend for the constant support and guidance.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

During the past decade, machine learning has become an increasingly integral part of computer science. Domains including health science, banking sectors, social sciences, and many others are applying machine learning for their tasks. It's used for email spam filtering, traffic predictions, to even social media suggestions about people one may know. The increasing availability of data, improvements in algorithms and computing power has led to a lot of advancements in this field. We can define machine learning as the task of deriving a model or hypothesis from data and the environment around it. We can categorize it primarily into Supervised, Unsupervised, Semi-Supervised, and Reinforcement learning.

For this thesis, we focused on supervised machine learning, where the idea is to learn to make predictions. Given a certain set of labeled data, the model or the predictor is trained to learn the relationship between the input and the output. Once trained and evaluated, the model is then expected to predict on unseen and unlabeled data. Supervised machine learning usually produces models with high accuracy given that the available data is a true representative of the original data distribution. Classification and regression are two types of supervised learning. During classification, the outputs are grouped into different classes whereas, for regression, the output generated is a singular value from a continuous range of values. The figures below show examples of classification and regression tasks.

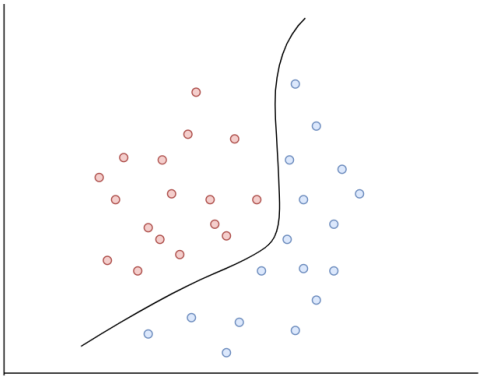

Figure 1.1: An example of classification.

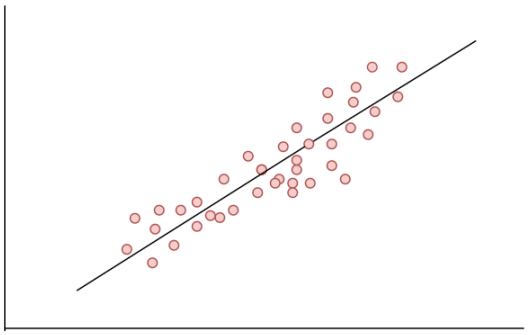

Figure 1.2: An example of regression.

Currently, neural networks and random forests are among the most successful predictors in terms of high predictive accuracy. However, these are inherently non-human understandable and are often also vulnerable to imperceptible adversarial perturbations of the input data [99]. The increasing use of machine learning in fields that have social impact, hence, require that the models are more interpretable and also robust to adversarial attacks. There have been several surveys and introductory papers stating different methods and approaches of interpretability in machine learning [13, 17, 19, 38, 45, 85]. Similarly, many researches studied and developed different types of adversarial attacks by input modifications [1, 20, 43], and several defense approaches for such attacks [81, 86, 116] most specifically for image data [3, 4].

In this thesis, we address aspects of both interpretability and robustness to adversarial perturbations in machine learning. The research started out by investigating how tools for interpretable machine learning may be exploited for addressing data shift scenarios. We investigate whether decision trees (a standard interpretable type of predictor) could be used for detecting covariate shift (we prove that it cannot), how covariate shift affects the validity of a decision tree surrogate model, and whether decision trees are a suitable local surrogate for detecting adversarially vulnerable areas of the feature space (Chapter 3). The

latter then led us to look into the notion of adversarial vulnerability more systematically. In Chapter 4, we argue that robustness needs to be redefined as a local requirement, propose a novel locally adaptive data augmentation scheme, and present a thorough empirical evaluation of this scheme.

1.1 Research Contribution

In the first part of our research, we address interpretable machine learning. We work with a type of inherently interpretable model, namely with decision trees as means of our explanation for blackbox models. We study the following research topics for the first part of the thesis:

- We investigate the use of decision trees for indicating marginal shift. We find that the decision trees are not reliable for detecting and explaining data shifts.

- We then work with the teacher-student framework where we train a neural network as the teacher model and decision trees as the surrogate model. We use the framework to investigate how well a decision tree can mimic and explain a blackbox model's performance. Additionally, we investigate the dependence of the surrogates on the unlabeled data generation. We use different distributions to generate unlabeled data points that are used for both training and then evaluating the surrogate models. While we find the trees as surrogate models to mostly be accurate and faithful to the blackbox models, we show how the validity of the surrogate model in various areas of the feature space strongly depends on the unlabeled data used for training the surrogate.

- We finally investigate the use of decision trees as a locally interpretable surrogate model. We find that decision trees generated locally around adversarial samples

provide good explanations of the model behavior. We study this further in Chapter 4.

In the second part of our thesis, we investigate adversarial robustness. We analyze how the consistency of robust learning is affected by a suitable choice of robustness parameter r, and argue that robustness should be redefined as a locally adaptive notion. In more details:

- We show that robust-Bayes and 0/1-Bayes are identical if and only if the distribution is r-separated. We also show that for every r, there exists a distribution where the two r-Bayes and 0/1-Bayes differ significantly as functions.
- We introduce the margin-rate, as a relaxed measure of r-separateness (one that allows for stochastic labels and does not require actual r-separateness of the support of P_X (that is, its support can be the full space), and relate it to suitable choices of r.
- We formally show that a slightly too largely chosen robustness parameter can lead to undesirable effects. Thus we argue that the robustness parameter r of an adversarial loss should be locally adaptive.
- We introduce our new adaptive robust loss, introduce its empirical version, and develop a novel adaptive-robust data-augmentation paradigm.
- We provide an extensive empirical evaluation of our novel adaptive robust loss and comparison of the adaptive-robust data augmentation with the fixed-range data augmentation.

1.2 Thesis Organization

The thesis is organized as follows: Chapter 2 introduces the basic notations and fundamental theorem of learning theory. Chapter 3 provides background on the topics of

interpretability and data shift in machine learning. The later sections of this chapter explain our experiments and evaluations. In Chapter 4, we look at adversarial robustness. The chapter is divided into sections explaining robustness, margins and robustness requirements. The last part of the chapter introduces the experiments and discussions on adversarial robustness. Finally, Chapter 5 concludes the thesis along with highlighting prospects of future work.

Chapter 2

Formal Framework of Learning

2.1 Overview

In this chapter, we briefly review the basic notations of statistical learning theory. This provides the basis for our investigations in the subsequent chapters, particularly for chapter 4. All definitions and results here are adapted from the introductory chapters of the textbook "Understanding Machine Learning" [93].

2.2 Statistical Learning Theory

We consider a standard setup of statistical learning theory for classification [93]. We let $\mathcal{X} \subseteq \mathbb{R}^d$ or $X = [0,1]^d$ denote the domain where d is some natural number and $\mathcal{Y}$ (mostly $\mathcal{Y} = \{0,1\}$) a (binary) label space. We assume that data is generated by some unknown distribution P over $\mathcal{X} \times \mathcal{Y}$ and let $P_{\mathcal{X}}$ denote the marginal of P over $\mathcal{X}$. Further, we use notation,

$$\mu_P(x) = \mathbb{P}_{(x,y)\sim P}[y = 1 \mid x] \tag{1}$$

to denote the *regression function* of P. A *hypothesis* or *classifier* is a function $h : \mathcal{X} \to \mathcal{Y}$. We let $\mathcal{F}$ denote the set of all Borel measurable functions from $\mathcal{X}$ to $\mathcal{Y}$ (or all functions in case of a countable domain). A *hypothesis class* is a subset of $\mathcal{F}$, often denoted by $\mathcal{H} \subseteq \mathcal{F}$. The quality of prediction of a hypothesis on an input/output pair (x, y) is measured by a *loss function* $\ell : (\mathcal{F} \times \mathcal{X} \times \mathcal{Y}) \to \mathbb{R}$. For classification problems, the quality of prediction is typically measured with the *binary* or *classification loss*:

$$\ell^{0/1}(h, x, y) = \mathbb{1}\left[h(x) \neq y\right], \tag{2}$$

where $\mathbb{1}\left[\alpha\right]$ denotes the indicator function for some predicate α.

We denote the *expected loss* (or *true loss*) of a hypothesis h with respect to the distribution P and loss function ℓ by :

$$\mathcal{L}_P(h) = \mathbb{E}_{(x,y)\sim P}[\ell(h, x, y)] \tag{3}$$

In particular, we will denote the true binary loss by $\mathcal{L}_P^{0/1}(h)$. The *Bayes classifier* is a (not necessarily unique) classifier which has the minimal true loss with regard to P. We denote the Bayes classifier with respect to the binary loss as h_P^B and its loss, the *Bayes risk* by

$$\mathcal{L}_P^B = \mathcal{L}_P^{0/1}(h_P^B) \tag{4}$$

and the true robust loss by $\mathcal{L}_P^{\mathcal{U}}(h)$. Further, we denote the *approximation error* of class $\mathcal{H}$ with respect to distribution P and loss function ℓ by

$$\mathcal{L}_P(\mathcal{H}) = \inf_{h\in\mathcal{H}} \mathcal{L}_P(h) \tag{5}$$

We denote the training sample set as $S = ((x_1, y_1), \ldots, (x_n, y_n))$, a finite set of sequence in $\mathcal{X} \times \mathcal{Y}$ and we assume that the examples in S are independently and identically

distributed according to P, this is known as the **i.i.d assumption**. The *empirical loss* of a hypothesis h with respect to loss function ℓ and sample S is defined as:

$$\mathcal{L}_S(h) = \frac{1}{n}\sum_{i=1}^{n} \ell(h, x_i, y_i) \tag{6}$$

Empirical risk minimization **ERM** is the approach of finding a predictor h from a hypothesis class $\mathcal{H}$ that minimizes the empirical error. We restrict the class of predictors to a fixed finite class $\mathcal{H}$, to prevent *ERM* from overfitting given sufficiently large training samples. For some labeled training samples, h_s is the result of applying ERM_H to S,

$$h_s \in argmin_{h\in\mathcal{H}} \mathcal{L}_S(h) \tag{7}$$

Realizability Assumption We say that P is realizable by $\mathcal{H}$ if there is a $h^* \in \mathcal{H}$ so that the true loss $\mathcal{L}_P(h^*) = 0$ (and therefore $\mathcal{L}_S(h^*) = 0$ for all S generated from P).

A *learner* $\mathcal{A}$ is a function that takes in a finite sequence of labeled instances $S = ((x_1, y_1), \ldots, (x_n, y_n))$ and outputs a hypothesis $h = \mathcal{A}(S)$. We interpret $\mathcal{L}_P(h_s) \leq \epsilon$ as an approximately correct predictor, successful output of the learner. The accuracy parameter is ϵ and it informs about the quality of prediction whereas δ is the probability of getting non-representative sample and $(1-\delta)$ is the confidence parameter of the prediction.

Definition 1 (Agnostic PAC Learning)**.** *A hypothesis class H is agnostic PAC (Probably Approximately Correct) learnable if there exist a function $m_H : (0,1) \to \mathbb{N}$, and a learning algorithm A with the following property: for all $\epsilon, \delta > 0$, and every distribution P, with $m \geq m_H(\epsilon, \delta)$ i.i.d examples generated from P, the algorithm returns a hypothesis h with probability of at least $(1-\delta)$,*

$$\mathcal{L}_P(h) \leq \min_{h'\in H} \mathcal{L}_P(h') + \epsilon \tag{8}$$

Here, ϵ corresponds to the "approximately correct" part, and δ indicates the "probably" part of "PAC". The function m_H determines the number of samples needed to achieve a probably accurate predictor for class H. We define the sample complexity of learning H as the smallest possible function m_H that satisfies the requirements of PAC learning. Every finite hypothesis class is PAC learnable with sample complexity:

$$m_H(\epsilon, \delta) \geq [\frac{log|H|/\delta]}{\epsilon}]$$

ERM works by ensuring that empirical risk of all hypothesis in H are good approximations of their true risk. If this is ensured uniformly over all hypothesis in H, we can say that ERM is a PAC learner.

A training set is called ϵ-representative (with distribution P, loss function l, domain $\mathcal{X} \times \mathcal{Y}$ and hypothesis class H) if

$$\forall h \in H, |\mathcal{L}_S(H) - \mathcal{L}_D(H)| \leq \epsilon$$

Definition 2 (Uniform Convergence)**.** *We say that a hypothesis class H has uniform convergence property if there exists a function $m_H^{UC} : (0,1)^2 \to \mathbb{N}$ such that for every $\epsilon, \delta \in (0,1)$ and for every distribution P over $\mathcal{X} \times \mathcal{Y}$ and i.i.d samples S of size $m > m_H^{UC}(\epsilon, \delta)$, then with probability of at least $1 - \delta$, S is ϵ-representative.*

If H has uniform convergence with $m_H^{UC}(\epsilon, \delta)$, then H is *PAC learnable* with the ERM algorithm and $m_H^{UC}(\frac{\epsilon}{2}, \delta)$. In most cases, the empirical risks of hypothesis h, where $h \in H$ will faithfully represent the true risk, if the uniform convergence property holds for a hypothesis class H. In general, a class is PAC learnable if and only if it has finite VC dimension.

Definition 3 (Shattering)**.** *Let H be a class of functions from X to $\{0,1\}$ and $C = \{c_1, ..., c_m\} \subset X$ and there are 2^m possible labeling. If hypothesis class H can represent all 2^m of the functions, then we can say that H shatters C.*

Definition 4. *The VC dimension of hypothesis class H is the largest finite set C that can be shattered by H.*

Some examples of VC dimension of different concept classes include, rectangle classes, which has VC dimension 4 while for interval classifiers it is 2. It is also possible for VC dimension to be infinite, in which case H is not PAC learnable. We can combine all the definitions explained above to state the fundamental theorem of learning.

Theorem 1 (The Fundamental Theorem of Statistical Learning)**.** *Let H be a hypothesis class of functions from a domain X to $\{0, 1\}$ and let the loss function be the $0 - 1$ loss. The the following is equivalent:*

1. *H has uniform convergence*
2. *ERM is a PAC learner for H*
3. *H is PAC learnable*
4. *H has a finite VC dimension*

The following notion of a *consistent learner* captures a more general notion of success of a learning algorithm: as the learner sees larger and larger samples from the data-generating distribution, the loss of the learner's output should converge to the Bayes risk.

Definition 5 (Consistency)**.** *We say that a learner $\mathcal{A}$ is* consistent *with respect to a set of distributions $\mathcal{P}$ if, for every $P \in \mathcal{P}$, every $\epsilon, \delta > 0$ we have there is a sample-size $n(P, \epsilon, \delta)$ such that, for all $n \geq n(P, \epsilon, \delta)$, we have*

$$\mathbb{P}_{S \sim P^n} \left[\mathcal{L}_P(\mathcal{A}(S)) \leq \mathcal{L}_P^B + \epsilon \right] \geq 1 - \delta \tag{9}$$

We say that $\mathcal{A}$ is universally consistent*, if $\mathcal{A}$ is consistent with respect to the class of all data-generating distributions.*

List of Notations	
Symbol	Description
R^d	A set of d dimensional vectors over a set of real numbers
X	Domain Space (a set of object that we may wish to label)
Z	Example Space (a set of examples)
Y	Label Space (set of possible labels)
S	Training Data (a sequence of labeled domain points)
P	A distribution over some set
$P_{\mathcal{X}}$	Marginal Distribution of P over X
A	Learning algorithm
$\mathcal{H}$	A hypothesis class
l	A loss function
$\ell^{0/1}$	A binary loss function
$\mathcal{L}_P(h)$	A true loss with respect to the distribution P
$\mathcal{L}_P{}^{0/1}(h)$	A true binary loss with respect to the distribution P
$\mathcal{L}_S(h)$	Empirical loss, the error the classifier h incurs over the training samples S
$S \sim P^n$	Sampling $S = z_1$,..., z_n i.i.d. according to P
C	Concept Class

Table 2.1: List of Notations

Chapter 3

Learning interpretable surrogate models

3.1 Overview

In this chapter, we present our investigations on whether and how techniques developed in the context of interpretability in machine learning may be exploited for dealing with data shift phenomena. We start out by providing brief introductions into both the topic of interpretability and transfer learning (Sections 3.2 and 3.3 respectively). We then turn to introduce decision trees as a particular type of inherently interpretable predictors and analyze them in three different scenarios: We started out by investigating whether decision trees could be employed for the purpose of detecting marginal shifts. We prove that we cannot indicate marginal shift using decision trees and provide a solid argument for why they do not seem suitable for this purpose (Section 3.5.2). We then turned to investigate decision trees as surrogate models, with an emphasis on understanding how the data used for training the surrogate affects the validity of the surrogate in various areas of the feature space. This study is summarized in Section 3.5.3. Finally, we investigated decision trees as local surrogate models (Section 3.5.4). In this section, we show that small locally trained decision trees are suitable indicators of data points lying close to the decision boundary. This led us to explore the decision boundaries of neural network blackbox predictors in

the context of adversarial perturbations more systematically in Chapter 4.

3.2 Interpretability

Machine learning provides an automated way of making decisions or predictions through learning and improving from experience. From autonomous cars [49] to judicial systems [59], machine learning is now being used in almost every field. However, the systems that are usually performing well are mostly blackbox in nature, which means that the internal mechanism of the model is unknown. Hence, it is crucial to give reasonings for the decision being made, specially in cases where human lives are involved. The goal of interpretability research for machine learning is to allow humans to comprehend why each decision is being taken. Justification of any decision is even more important now, as recently European Union legislation General Data Protection Regulation (GDPR) requires explanations from companies, that work with automatic decision systems which have a substantial impact on human lives [30]. Hence, interpretability is crucial to make systems more socially acceptable, reliable and, trustworthy.

Research on interpretability has been taking place over the past few years [38, 57, 82]. It is an interdisciplinary work where different fields of machine learning, social science, and linguistics are involved. It has also been applied in different domains such as medicine [78], finance [28], data science [71], natural language processing [52], etc. Interpretability is also needed to meet certain criteria such as fairness, ethics, safety [30], etc. It allows human to verify the correctness of the model, identify prejudiced features in data [51], justify medical diagnosis [118], judgments by judicial systems [120] and also help to understand why models perform worse in certain situations. However, interpretability potentially comes with a trade-off with model performance [50]. Hence, researchers are recently attempting to make systems and decisions both accurate and interpretable [46, 62].

There have been several survey papers [2, 18, 38, 45, 58, 101] that summarized the

recent work done on the interpretability of machine learning. These surveys identify various categories of interpretability and also discuss various explainability strategies. There are taxonomies for both evaluations of interpretability and methods of interpretability. These are listed below:

Intrinsic vs Post Hoc This category explains whether the complexity of the model is restricted to allow for interpretation or if the model interpretation is added after training. Intrinsic interpretability means that the model is inherently interpretable due to its simple structure, as one example, **decision trees** [8]. Whereas post hoc refers to the implementation of interpretation methods on the trained blackbox to explain its prediction. For example, **live and breakdown** [65, 96]. Live learns local model for regression tasks whereas breakdown is a greedy approach that decomposes model predictions into parts.

Local vs global This groups interpretability methods based on whether the model explains the behavior of a single prediction or it explains the overall blackbox behavior. Local indicates that the interpretation is done locally around one data point, such as:

- **Lime** [83] learns interpretable model locally around the prediction of one data point.
- **Anchors** [84] explain blackbox behavior with high precision rules for each sample.
- **Shap** [61] assigns importance values to all features for each prediction.
- **Lore** [44] is similar to [83] but generates more samples near decision boundaries for a better local model.

Global explanation gives interpretation to the overall model behavior. One example is the **teacher/student framework** [8]. The teacher is a complex blackbox model

and the student is a simpler interpretable model. The student model tries to mimic the blackbox behavior as an interpretation to the complex model.

Model specific vs model agnostic Interpretation tools that are model specific only works on one type of model, such as:

- **SmoothGrad saliency maps** [94] which produces saliency maps (also termed pixel attribution maps, or sensitivity maps as an explanation for a particular input and prediction) for SmoothGrad technique.
- **Grad-CAM** [44] generates visual explanations for deep neural networks.

while model agnostic methods work on any model. Examples are:

- **PDP** [32] produces partial dependence plots that shows the relationship between targets and features.
- **ICE Plots** [72] are an extension of [32]. The plots illustrate the distribution of individual conditional expectations functions.
- **Aggregated local rules (MAGIX)** [73] produces global if then rules from local explanations.

Types of output Interpretation methods also differ in the various outputs they have such as:

- **Summary of Features**: Interpretation methods may return feature importance [37] or interaction of features [22] which can be used to explain a complex model behavior.
- **Visualizations**: Many interpretability tools output visualizations as means of interpretation of a task, such as TreeView [100] which visualizes a random forest and Nomograms [105] which provide visualization for feature interactions.

- **Model weights**: The learned weights of the model may also give insight into its behavior. This is very similar to the summary of features.
- **Data Points**: Methods may return new or existing data points that can help explain a prediction, such as counterfactual explanation [42].

3.3 Data Shift

Traditional Machine learning assumes the data generation process is the same during both training and testing. Learning is usually considered in isolation and algorithms are trained to solve specific tasks. However, in most practical tasks, the data generating distribution is different in both the training and the target domain. Traditional machine learning algorithms are unable to address the shift in domains. The form of learning in these scenarios is known as transfer learning [76].

Transfer learning aims to overcome the shortcoming of traditional machine learning by utilizing knowledge from previously learned tasks [69, 102]. Here, training the model for a new task relies on previously learned tasks. This is particularly useful when there is insufficient data in the new domain, so the knowledge from the old domain can be used to learn in the new field. Labeled data might also be unavailable as unlabelled data are generally cheaper and easier to obtain. Hence, the goal of transfer learning in all these settings is to make use of the already trained predictors (or various sources of data) and avoid training from scratch. As humans, we always use knowledge from previously learned tasks to help us do new tasks. For example, learning to ride a motorcycle makes it easier to learn to drive a car. Transfer learning is being used in different domains including natural language processing [88], image classification [48], and time-series prediction [33].

Transfer learning is an umbrella term for dealing with the phenomenon of *data shift*, that is, situations where the data generation differs between training and testing. Examples of transfer learning frameworks are: **Domain adaptation** [27, 107] (adapting a predictor

trained on one source of data to perform well on one target task); **Multi-source Domain adaptation** [98, 119] (the same but multiple data sources); **Lifelong learning** [35] (the data generation change is tracked over time and predictors continuously adapted, ideally without fully retraining).

Domain adaptation is a special case of transfer learning. Domain adaptation occurs when learning takes place from source data distribution and then the learned model predictors is adopted to predicting on a different target data distribution. Data shift can occur both naturally and also due to adversarial manipulation. It is impossible to develop successful transfer learning without knowledge of the type of shift [10]. The types of shift can be categorized into [68, 77]:

- **Covariate Shift**: Also known as the shift of independent variables. The shift occurs in the distributions of the training and testing input variables. This is also the most common type of shift [12]. If $\boldsymbol{x}$ is the feature vector, $\boldsymbol{y}$ is the label and P_{train}, P_{test} describe the training and testing distribution, we can define covariate shift as:

$$P_{train}(y|x) = P_{test}(y|x), P_{train}(x) \neq P_{test}(x) \tag{10}$$

- **Prior Probability Shift**: This shift occurs due to the change in distribution of the training and testing label class [95]. Here,

$$P_{train}(y|x) = P_{test}(y|x), P_{train}(y) \neq P_{test}(y) \tag{11}$$

- **Shift under sample selection Bias**: This arises from non uniform selection of samples for training and can fall under covariate shift. So biases are formed during training and the true distribution is not captured. Hence there is a shift between the training and the testing data [47].

- **Shift under Adversarial Manipulation**: Adversaries may be powerful enough to change the test dataset; resulting in a dataset shift between the training and the testing data. This can be seen in several situations such as spam filtering classification, where the adversaries attempt to defeat the trained model with adversarial examples. [26].

There have been several surveys that looked into transfer learning such as [109], [121], [80]. Most of the work done on data shift and specifically under adversarial settings has been done for image data [70], [40], [24]. Interpretability is potentially very beneficial for dealing with data shift as it provides explanations for the shift. Firstly, it may help us to understand the linkage between the source and the target domain. This can help us evaluate how to adopt before applying the model. Secondly, results can also be used to explain if a shift has happened and it will also be possible to understand the extent and the type of shift that took place. Lastly, domain experts can get insight into the shift and improve algorithm/data accordingly. Hence, interpretability allows easy debugging and optimization of the model. w

3.4 Literature

Several surveys [2, 18, 38, 101] have been conducted on the interpretability of machine learning. The methods and taxonomy that these surveys introduced were discussed in Section 3.2. Additional papers on interpretability and specifically the ones exploring connections between interpretability and data shift are discussed here in this section.

Apart from the application of global and local interpretation tools for standard learning settings, some researches looked into interpretability for other learning settings such as, Lu et al. [60] considers interpretability from an active learning perspective. They introduce active decision set induction (ADS) to learn a set of interpretable if-else rules. On the other

hand, Thiagarajan et al. [100] suggest obtaining global interpretability via hierarchical partitioning of feature space of a complex model. The authors then use visualization to identify the changes in factors behind the predictions. An interpretable decision tree surrogate is generated based on the meta-features. Bhattacharjee et al. [9] introduce interpretability to models with non-linear kernels. They generate color based nomogram where the length/color range shows the contribution of each input variable.

[79] and [106] are introductory papers for interpretability in terms of transfer learning. The first one highlights the emerging area of interpretable explanation for transfer learning in sequential tasks. Similarly, Lee et al. [106] explains the process of learning a new task from a partial decision tree that has been generated using knowledge from a previous task. The usage of decision trees adds inherent interpretability due to the simple tree structure.

Kim et al. [54] also address the lack of enough labeled data to train the complex models. Hence, they introduce Feature Network, which trains models with defined interpretable features. It allows more efficient and interpretable transfer learning due to the usage of interpretable features. Additionally, a mapping layer is used, which helps humans to understand the relationship between the features and the outputs. Similarly, Krishnan et al. [56] utilizes the attention layer of a deep neural network as an explanation of the model predictions. They use unsupervised domain adaptation to filter out tweets in an emergency without using new examples.

Segev et al. [92] [91] also focus on model transfer learning and introduce simple model transformations based on local (and greedy) changes that rely on decision trees, which makes their algorithm interpretable. Although research on interpretability has been increasing over the past few years, there's very limited work done on interpretable domain shift. Hence, in this research, we work with explainable models and also attempt to interpret data shift.

3.5 Exploring the role of decision trees in data shift

In this section, we describe our approach and experimental findings for achieving interpretability through decision trees in data shift scenarios. We initially formally explain what decision trees are and how they are constructed (Section 3.5.1). We then investigate the trees under three different scenarios:

- We attempt to explain marginal shift using the trees as global interpretable models. We find that the decision trees are not reliable for explaining data shift.
- We then conduct experiments with the teacher-student framework where a neural network is trained as the teacher model and a decision tree is used as the surrogate model. We explore the effects of using different distributions to generate unlabeled data for both training and evaluating the trees. While we find the trees as surrogate models to mostly be accurate and faithful to the blackbox models, we show how the validity of the surrogate model in various areas of the feature space strongly depends on the unlabeled data used for training the surrogate.
- We then investigate local interpretability with decision trees. We locally train small decision trees around samples that are potentially susceptible to adversarial perturbations. We study this further in Chapter 4.

3.5.1 Decision Trees

Decision trees are non-parametric predictors that can be used in various fields for both classification and regression tasks. They are tree-based models that are considered to be inherently interpretable due to their simple hierarchical structures. The models are trained to predict the value of a target variable by learning decision rules based on the data. A decision tree is hence a predictor, $h : \mathcal{X} \rightarrow \mathcal{Y}$, which predicts the label y of a data instance

x that travels from the root node to the leaf node. For most of our work on this thesis, we will focus on binary classification by decision trees.

The trees consists of one root node and two other types of nodes: decision nodes and leaf nodes. We start with the root node, and then examine the attributes in the decision nodes and move to the other nodes based on the results from the examination. This is repeated until a leaf node is reached. For classification, the prediction is the majority value of the training data in the leaf node. For regression, the prediction is the mean value of training data in the leaf node.

Decision trees divide the feature space into axis-aligned rectangles. Each path from the root-to-leaf node defines an area of the input space. To learn a tree from a given set of data points, we initially start with an empty tree. Next, the best attribute is chosen for splitting the data to minimize the impurity of the label classes. This is a greedy approach where the chosen attribute partition is the one that minimizes the impurity of the child node relative to the parent node. There are several impurity measures to determine the best split, such as: **Entropy, Gini Index, Miss-classification**. Since we mostly focus on classification, we use Gini Index as the impurity measure. The attribute with the lowest Gini index is chosen for splitting, where P_i is the empirical probability of class i. Gini impurity measures the frequency of the mislabeling of any samples from the dataset when they are being randomly labeled.

$$Gini = 1 - \sum_{i=1}^{n} (P_i)^2 \tag{12}$$

To avoid overfitting of the tree, the growth of the tree can either be prevented by using any stopping criterion or can be pruned after its fully fitted to the training data. There are several stopping criterion such as using **validation set** or **penalization of the tree complexity**. Similarly, for pruning a top down approach called **pessimistic error pruning**

or a bottom up approach called the **critical value pruning** can be taken. There are several standard algorithms for learning decision trees, such as: **ID.3** [75], **C4.5** [74],**CART** [14], etc.

The illustration below shows a simple binary classification using decision trees in a two dimensional setting:

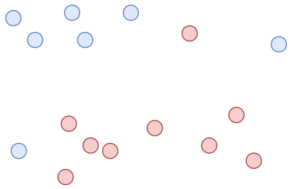

Figure 3.1: Simple 2D dataset

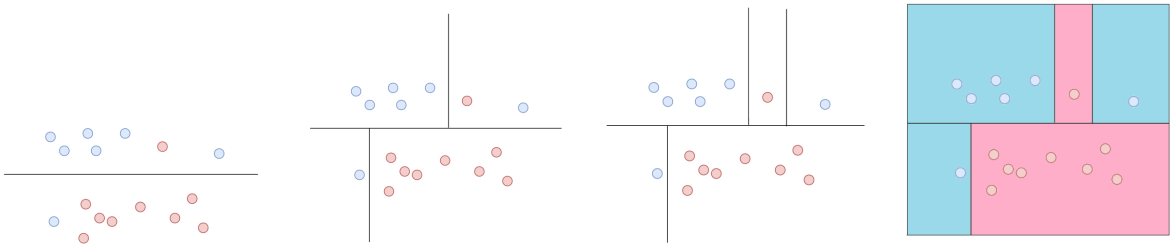

Figure 3.2: Decision tree after first, second and third split and the final outcome respectively.

Similar to the several methods described in section 3.2, decision trees are one of the procedures for achieving interpretability [8]. Decision trees are inherently interpretable as the decision path of the tree can be decomposed to explain the predictions [65]. For the next part of this chapter, we test decision trees in three different ways. We firstly investigate if the trees can indicate and explain the shift in the marginal data distribution over the feature space (covariate shift setting). We find that the trees cannot always explain data shift in all settings. Hence, decision trees are misleading as indicators for covariate shift.

Next, we conduct experiments with the teacher-student framework. The teacher-student framework consists of training two models, a complex teacher model and a simpler surrogate model known as the student. The student is trained to mimic the performance of the blackbox teacher. We again use decision tree as the interpretable surrogate model as was done in earlier work [8]. Our results confirm that the trees are mostly accurate and highly faithful to the blackbox model; they are able to produce results that mimic the blackbox performance. We then create a covariate shift settings by generating unlabeled data from various distributions for both training and testing the trees. We find that the tree performance is highly dependent on the unlabeled data generation process and how it relates to the areas the surrogate is tested on. Additionally, we show how the complexity of the trees also influence this performance. Lastly, we investigate decision trees as local surrogate models around different training samples. We find the trees to be a good indicator of local behavior around data points. We study this further in chapter 4. All the experiments in this section were done on both artificial and real UCI datasets. The UCI datasets include: Breast Cancer Diagnosis, Car, Dermatology, Iris, Postoperative and Wine data [31].

3.5.2 Trees as detectors of marginal shift

One of the questions we asked was whether we could utilize decision trees as indicators/explanations for the change in the marginal distribution of the input space. Marginal distribution defines the percentage of points that exist in certain subareas/marginals of the total area. As explained in section 3.3, several types of shift can occur which can change the marginal distribution of the overall space.

To investigate this, the marginals of the data distribution was altered to understand if the decision trees can be used to identify the underlying marginal shift that has taken place. We generated two-dimensional artificial data under the covariate shift setting, with

different marginals but the same conditional labeling in each setting. The use of artificial datasets allowed us to visualize and capture various structures. We used scikit-learn [16] to train and generate decision trees on these datasets. We generated three synthetic datasets, each containing two features, two label classes, and 5000 data points on a 10×10 input space. Only the marginal distributions were changed for each dataset. We named the three different datasets as Marginal A, Marginal B, and Marginal C. The colors of the data points on Fig 3.3 represent the two classes. The marginal distributions on all three datasets on Fig 3.7, 3.3 and 3.4 are different from each other. Decision trees were then trained for each of these datasets and the tree structures were compared.

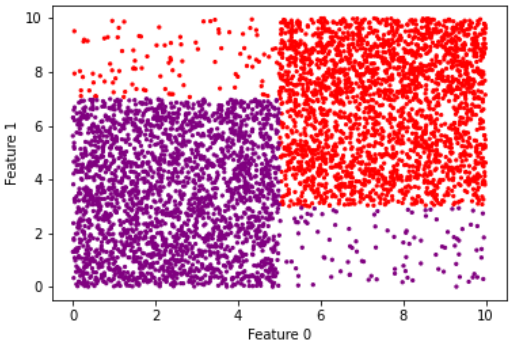

Figure 3.3: Marginal A Dataset

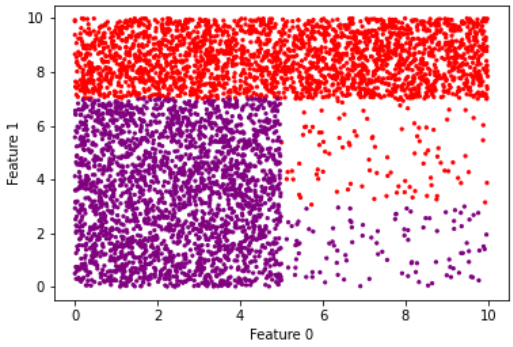

Figure 3.4: Marginal B Dataset

As seen on Fig 3.5, the tree trained from the dataset on Fig 3.3 favors a vertical split on feature 0 while the tree on Fig 3.6 trained from the dataset on Fig 3.4 splits on feature 1. The topmost split of both the tree tried to separate the largest marginals in the space. The trees were biased towards the larger marginals and the tree structures altered along with the change in the marginal. From the differing trees, we can identify that data shift has taken place between the two datasets (Fig 3.3, 3.4).

However, we also observe that the tree generated on Fig 3.8 is very similar to the tree on Fig 3.6, despite having different marginal distributions (Fig 3.7, 3.4, respectively). Since we were able to generate identical trees under two different marginal distributions,

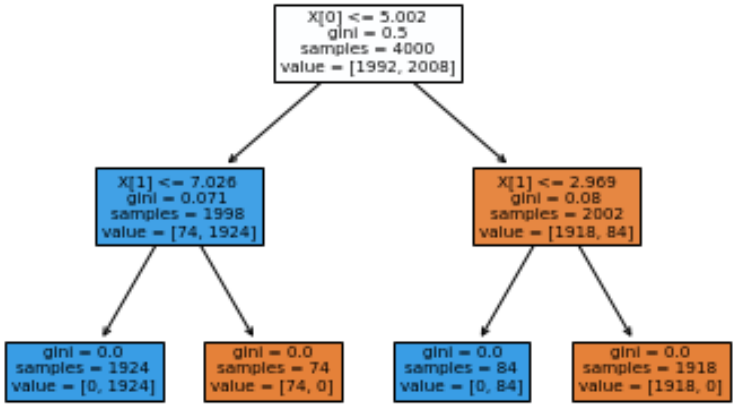

Figure 3.5: Tree generated from Marginal A

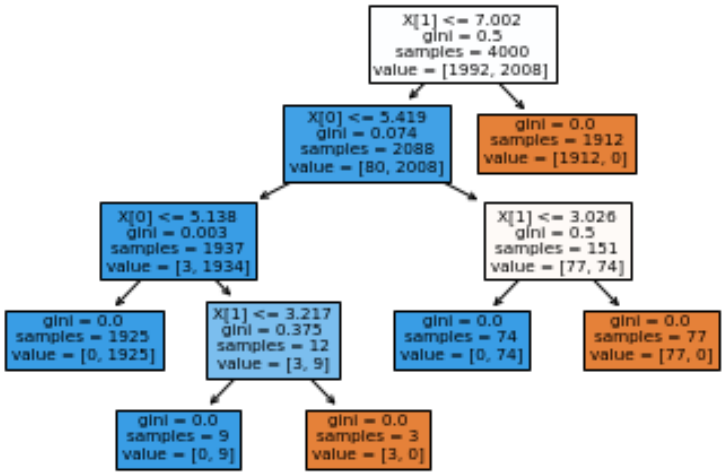

Figure 3.6: Tree generated from Marginal B

it means that the same tree structure can represent two different marginal distributions. Additionally, we cannot explain the type of shift or the reasoning behind the shift using the decision trees. Thus, trees are not suitable as detectors of marginal shift.

To summarize, our construction exhibits the following phenomenon: Out of the three different marginal distributions (Marginal A, B, C in Fig 3.3, 3.4 and 3.7) two lead to the same (or almost identical) tree (Fig 3.6 and 3.8), while one lead to a different tree structure (Fig 3.5). This shows that distributions differing in marginals can lead to differing trees, but don't always do so. Thus decision trees can not be viewed as reliable detectors of marginal (covariate) shift. The trees can give rise to misleading explanations of marginal shift.

We also explored the variable importance of the tree to see if it is more insightful than the tree splits in explaining data shift under marginal changes. Variable importance states the relative influence of each feature on the tree. However, it was very similar to the tree splits; features that had higher variable importance were also the ones appearing at the top of the tree.

3.5.3 Dependence of trees as surrogates on unlabeled data generation

Our next goal was to investigate how the data generation of the unlabeled data affects the student model in a teacher-student framework. We wanted to investigate the effect of the unlabeled data from different data distributions on both training and testing the student model. We propose that the evaluation and validity of the surrogate models should depend on the underlying data distribution.

The teacher-student framework consists of two training models; a complex model known as the teacher and a simpler model called the student. In this teacher-student framework, the student model attempts to mimic the performance of the teacher model. This allows

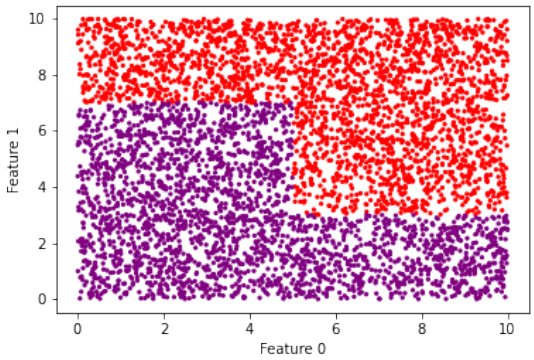

Figure 3.7: Marginal C Dataset

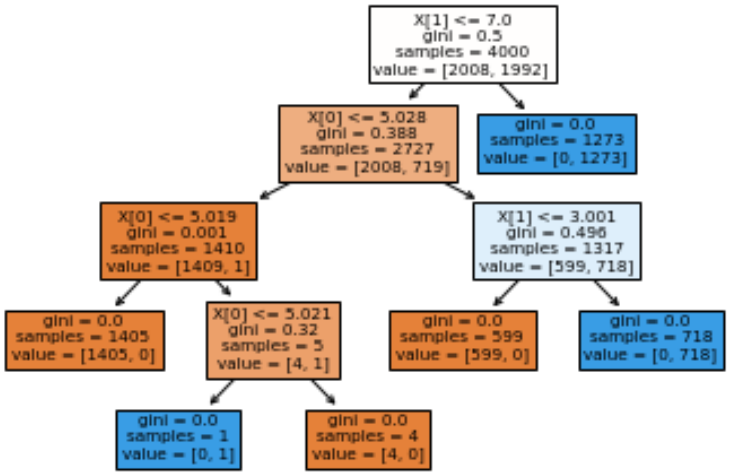

Figure 3.8: Tree generated from Marginal C

an interpretable approximation to the teacher model ideally without compromising on the performance of the original complex model too much. The student model can then be used for debugging and understanding the behavior of the blackbox model. If the student model is successful, it can also be used instead of the blackbox for situations where more transparency is needed. This framework has in general been applied in many domains such as reinforcement learning [122], speech recognition [63] and feature selection [64]. A general algorithm for the teacher-student framework is outlined on algorithm 1.

Algorithm 1: General Teacher-Student Framework

1. Train a teacher model $f_{teacher}$ on the training dataset S_{train}.
2. Construct a distribution P over the input space X by fitting the new distribution P to the training data S_{train}.
3. Generate new unlabeled data $S_{unlabeled}$ from this newly formed distribution P.
4. Label the new unlabeled $S_{unlabeled}$ by the trained teacher model $f_{teacher}$.
5. Now use the new data points $S_{unlabeled}$ to train a student model $f_{student}$.
6. Evaluate the performance of the student model $f_{student}$ relative to the teacher model $f_{teacher}$.

For our experiments, we focused on achieving a high relative performance of the student model in comparison to the teacher, even if the teacher had poor performance. Our goal was to achieve a surrogate model that correctly approximates a complex model. We primarily used neural networks as the teacher model and the decision trees as surrogate models, learned using [8] for the teacher-student experiments.

Bastani et al. [8] introduces an algorithm which only needs to obtain the output $y = f(x)$ given a input x and a blackbox model f to learn decision trees. The learned decision trees can then be used as a surrogate model to interpret the blackbox model. There are few key differences between learning a tree by using CART [14] and [8]. The algorithm in [8] forms new distribution over the original input space X by fitting a mixture of a Gaussian distribution to the training data using expectation maximization. This results in the algorithm having a generative model of the data generating distribution. Unlabelled data is then generated from this distribution at each node during the training

of the tree. The newly generated samples are used to calculate the information gain to decide on the next best split. The optimal leaf labels are also computed according to this distribution. The algorithm also showed substantial improvement in learning a tree over using a standard algorithm like CART [14]. Thus, this algorithm was used for our work, to generate the decision trees.

Along with Gaussian distribution, we introduce two other distributions; Uniform and a blend of Gaussian and Uniform distribution for generating the decision trees. Similar to [8], we form the distributions over the original input space X by fitting a mixture of the selected distribution to the training data. We wanted to investigate the following questions:

- How the student performs on the original task, how accurate is its performance?
- How well the student mimics the teacher model; if the model is faithful to the teacher model?
- How well does the student explain the teacher in domain areas that are not covered by the original training data, in areas of data shift?
- How does this change as a function of the complexity of the surrogate model?

To confirm this, we trained and tested neural networks on the original data distribution and then trained and tested decision trees with different data distributions. The experiments were done both on artificial and real UCI datasets [31]. A $80 - 20$ split was made on the UCI datasets for training and testing. For the artificial datasets, the training and the test sets were generated separately. Initially a scikit-learn [16] neural network (1 hidden layer, 500 nodes) was trained on each dataset until a good predictive accuracy was achieved. The trained teacher model was then kept the same for all the subsequent tests for that dataset.

As mentioned previously, we use three different sample generation processes to obtain new samples for training the student trees. At each node of the trees, 2000 points were generated to train these decision trees under different distributions. These samples are used to fit the tree to the data (for which Gini impurity is calculated at each node). For the tree growth, we iteratively increment the tree size. We score the tree at each step of the iteration. We set the max node as 55, however, we stopped growing the tree if there was no change in the scores over 3 iterations. The three different data generating distributions were:

- **Gaussian Extract**: A Gaussian mixture model was fitted to original training data to generate new unlabeled data. The number of components in the Gaussian mixture was set to 100. The trees trained from this distribution is denoted as Gauss Extract.
- **Uniform Extract**: Points were sampled uniformly from the range of feature space covering the training data. The trees generated from this distribution is denoted as Uni Extract.
- **Blend Gaussian Uniform Extract**: A Gaussian mixture model was fitted to original training data to generate 1000 points and the other 1000 points were sampled uniformly from the range of feature space covering the training data. The trees formed from this distribution is denoted as BlendGaussUni Extract.

The newly generated points are then labeled by the blackbox and are used to train the decision trees. We then conducted four types of tests on the various trees. To account for the randomness in the data and tree generation, tests were conducted three times and the average scores were taken. For each test, unlabeled data points were again generated under different distributions. These points were firstly labeled by the blackbox and were only used for testing the trained student model. The tests were:

- **Original Test**: Tests the trained tree performance on original test data points labeled by the blackbox model.
- **Gaussian Test**: Tests the trained tree performance on the generated Gaussian samples labeled by the blackbox model.
- **Uniform Test**: Tests the trained tree performance on uniform points labeled by the blackbox model.
- **Blend Gaussian Uniform Test**: Tests the trained tree performance on a combination of Gaussian and uniform points labeled by the blackbox model.

We present the results of our experiments in the plots of Figures 3.9 and 3.10. Both figures contain several plots of the fidelity of the surrogate student model as a function of the complexity of the surrogate (number of nodes in the decision tree). The fidelity scores of the surrogates trained on the three different distributions for the unlabeled data (Gaussian, Uniform and BlendGaussUni) are plotted in three different colors. Each row in the figures corresponds to a different dataset, whereas each column corresponds to a different test data distribution. We summarize our findings as follows:

1. We observe the plots in the first two columns (original test data and Gaussian mixture test data) are qualitatively very similar for almost all datasets. This confirms that for these datasets, the Gaussian mixture was an adequate approximation of the original data-generating distribution and the data generated from the Gaussians had similar characteristics as the original data with respect to the surrogate training.

2. We observe the fidelity of the surrogate increases with the complexity of the surrogate model. While it is natural that a more complex model can achieve higher accuracy when imitating a (potentially) complex blackbox predictor, in the context of interpretability this illustrates a natural trade-off: a more complex surrogate (ie. a larger decision tree) may achieve higher fidelity, but this comes at the cost of being less readily understandable

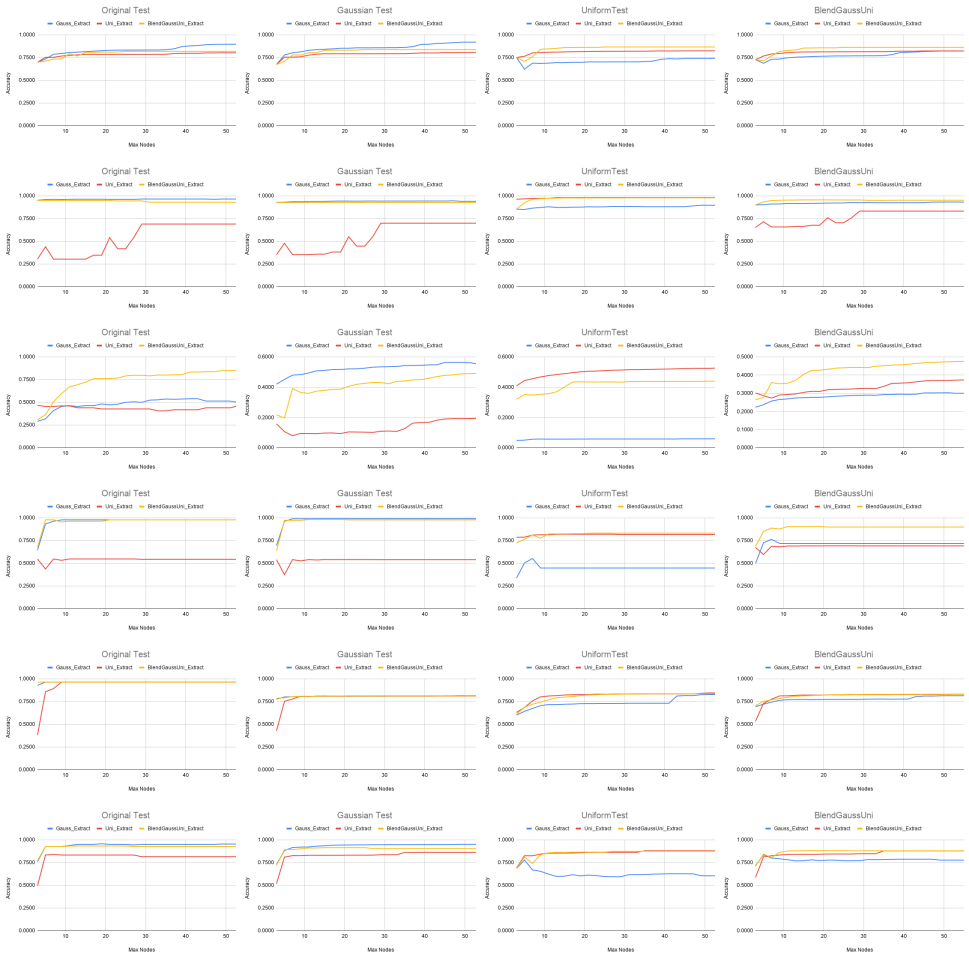

Figure 3.9: Student model (decision tree) trained on 4 data distribution and tested on data labeled by the blackbox. The graph by column: **Original Test, Gaussian Test, Uniform Test, BlendGaussianUniform Test**. Each row represent different datasets, from top to bottom: **Car, Breast Cancer Diagnosis, Dermatology, Iris, Post Operative, Wine.**

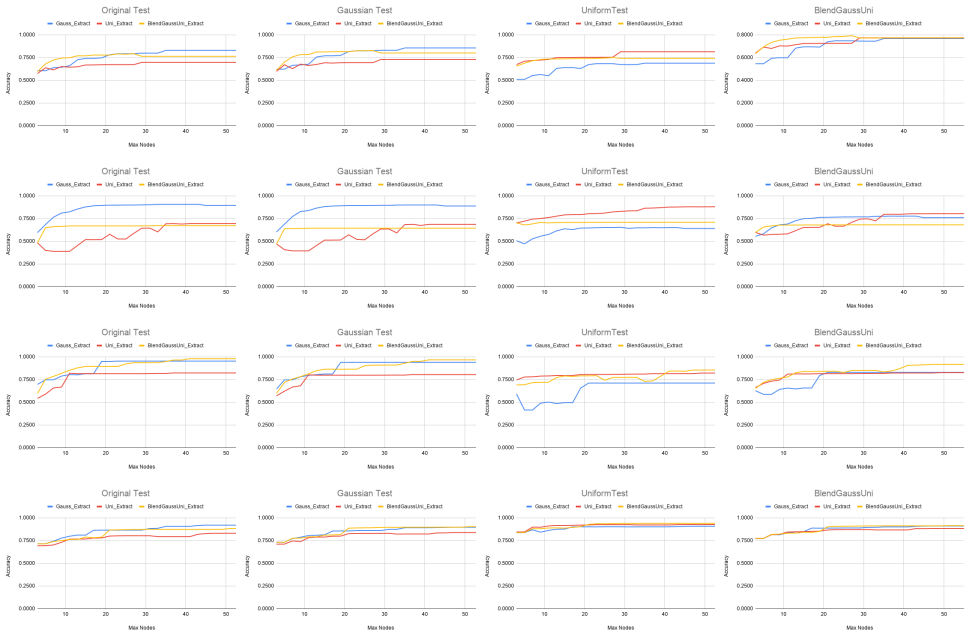

Figure 3.10: Student model (decision tree) trained on 4 data distribution and tested on data labeled by the blackbox. The data used is synthetic data from a one-dimensional manifold in two-dimensional space. The graph by column: **Original Test, Gaussian Test, Uniform Test, BlendGaussianUniform Test**. Each row represent different datasets, from top to bottom: **Box, Circle, S, Sine.**

	BlackBox Model Accuracy Scores		Student Model									
				Fidelity Scores					Accuracy Scores			
Dataset	Training	Test	Type of Tree	Relative Training	Original Test	Gauss Test	Uniform Test	BlendGaussUni Test	Training	Test	Node Count	Max Nodes
Iris	1.000	**0.956**	Gauss_Extract	0.971	0.933	0.961	0.504	0.726	0.971	0.933	5	5
			Gauss_Extract	1.000	0.978	0.991	0.449	0.719	1.000	0.933	15	15
			Gauss_Extract	1.000	0.978	0.991	0.449	0.719	1.000	0.933	15	25
			Gauss_Extract	1.000	0.978	0.991	0.449	0.719	1.000	0.933	15	35
			Gauss_Extract	1.000	0.978	0.991	0.449	0.719	1.000	0.933	15	45
			Gauss_Extract	1.000	**0.978**	0.991	0.449	0.719	1.000	**0.933**	15	55
			Uni_Extract	0.365	0.437	0.376	0.788	0.596	0.365	0.481	5	5
			Uni_Extract	0.540	0.548	0.541	0.820	0.692	0.540	0.593	15	15
			Uni_Extract	0.540	0.548	0.541	0.820	0.692	0.540	0.593	25	25
			Uni_Extract	0.538	0.544	0.540	0.817	0.692	0.538	0.589	29	35
			Uni_Extract	0.538	0.544	0.540	0.817	0.692	0.538	0.589	29	45
			Uni_Extract	0.538	0.544	0.540	0.817	0.692	0.538	0.589	29	55
			BlendGaussUni_Extract	0.962	0.978	0.973	0.764	0.851	0.962	0.933	5	5
			BlendGaussUni_Extract	0.984	0.963	0.982	0.824	0.904	0.984	0.919	15	15
			BlendGaussUni_Extract	0.971	0.978	0.978	0.832	0.899	0.971	0.933	25	25
			BlendGaussUni_Extract	0.971	0.978	0.978	0.832	0.899	0.971	0.933	31	35
			BlendGaussUni_Extract	0.971	0.978	0.978	0.832	0.899	0.971	0.933	31	45
			BlendGaussUni_Extract	0.971	**0.978**	0.978	0.832	0.899	0.971	**0.933**	31	55

Figure 3.11: The surrogate tree accuracy results on the Iris test dataset

to a human user. Tracking the fidelity closely as a function of the number of nodes in the tree as we do here can allow for choosing the smallest possible tree that achieves adequate fidelity.

3. The fidelity of the surrogate on the various test distributions clearly varies largely with the distribution that the surrogate was trained on. Naturally, the highest fidelity is achieved when training and test distribution match. We further note that when the test distribution is uniform on a larger ambient part of the feature space, the surrogate trained on the Gaussian mixture often performs poorly. This shows that, if a surrogate model is expected to offer valid explanations outside the original data distribution (here approximated with the Gaussian mixture) then it is crucial to take this into account at the training time of the surrogate and train the surrogate on the area that it is expected to provide explanations for. On the other hand, training the surrogate on the larger ambient data can reduce the fidelity on the original data distribution. Thus, simply always training a surrogate on data generated from the surrounding space can lead to poorer performance on the original data distribution and would therefore not be desirable in situations where

the explanations are required only for these areas. For many datasets (though not all) the surrogate trained on a mixture of original (or gaussian approximate) and uniform ambient data performed well both on the original and on the ambient test. This shows that using such a mixture as data for training a surrogate can provide a favorable balance of both providing in-distribution and ambient fidelity.

Similarly, table 3.11 shows the results of the evaluations conducted on the test set of the Iris dataset [31]. The surrogates trained with the gaussian data (Gauss Extract) and with the mixture of gaussian and uniform data (BlendGaussUni Extract), performs well in all three tests. In the table, we also observe that the blackbox manages to get an accuracy of 95.6% in the domain test data points. The student models (Gauss Extract and BlendGaussUni Extract) also achieve a very close accuracy of 93.3% when evaluated on the same data points. Similarly, for the original test, Gauss Extract and BlendGaussUni Extract achieve high accuracy of 97.8%, which indicates its faithfulness to the blackbox model.

The results hence show that these surrogate models were both accurate and faithful to the teacher model. They were able to mimic the blackbox performance and due to the addition of new data points, the surrogate models can also explain areas that are not covered by the original domain points. We can use the trees as an explanation or even an alternative predictive measure.

3.5.4 Trees as local surrogates

Since trees generated globally can give misleading explanations of blackbox behavior, we investigated decision trees as a local explanation. Local interpretable methods work by explaining individual predictions of blackbox machine learning models. It helps to understand the reasoning behind each model prediction. The decision trees are generated locally around each data sample. The tree structure can inform if the data sample is sitting on a

homogeneous space or on the decision boundary, error or adversarial regions.

Decision boundaries separate the data points into different classes. Throughout different experiments, we observed that the decision boundary of a trained blackbox model often sits close to the original manifold of the dataset. In such a case, we view these regions as adversarial regions. The model may classify these points (with slight perturbations) incorrectly into a different label. Hence, the model is vulnerable to adversarial attacks on these regions as a slight shift in the values of the samples, in this area, can result in different labels from the blackbox.

To investigate local tree surrogates, we firstly trained neural networks on both real and artificial datasets. We trained several variations of the network (with different hidden layers, activation functions, stopping criterion) until a good performance was achieved. For visualization purposes, we again illustrate the results on a two-dimensional artificial dataset. For this, the neural network was set as, two hidden layers of size (5,5), ReLU as the hidden layer activation function, and sigmoid as the last layer activation function. Binary cross-entropy was used as the loss function to train the network.

One of our synthetic datasets is illustrated in Fig 3.12. We generated 5000 uniform points on the input space, to see the behavior of the blackbox on areas that are not covered by the training data points. We labeled the new points by the trained blackbox model as seen in Fig 3.13. The colors on the original data manifold represent the ground truth label whereas the surrounding points represent the uniform points labeled by the blackbox model. The neural network has good predictive accuracy and the labels on the uniform samples by the blackbox correspond to the original labeling.

From Fig 3.13 we observe that some of the decision boundaries of the neural network sit on the original data manifold which makes the samples at that region vulnerable to adversarial perturbations. Hence, we generated decision trees around some of the samples that are susceptible to adversarial attacks. We also generated decision trees around other

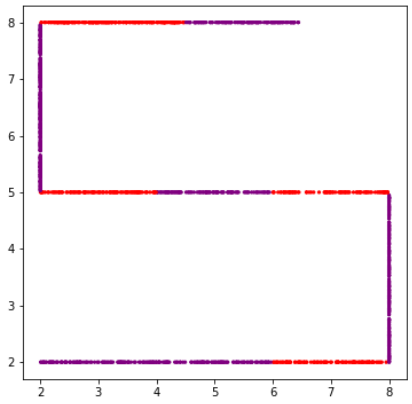

Figure 3.12: Two dimensional Artificial Dataset-S with two labels

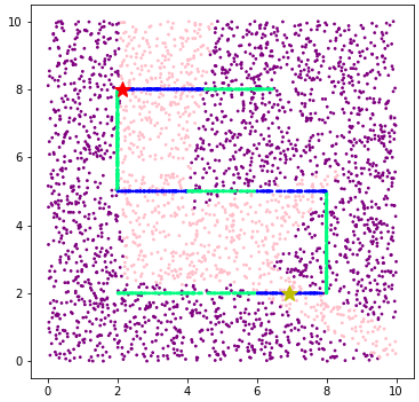

Figure 3.13: Uniform points around S dataset labeled by the blackbox

samples for comparison. In Fig 3.13, the red star indicates a sample in an adversarial zone whereas the yellow star indicates a sample in a non-adversarial (homogeneous labeling) zone. For the decision trees, we again generated 100 new samples in a ball around the selected points and labeled those points by the neural network. The radiuses for the balls of points were based on the range of the features. Finally, we trained a scikit-learn decision tree on those newly generated points.

From the trees, we can see that sample from non-adversarial zones on Fig 3.14 had a tree with one node (so, all the points had the same label). This means that the original data sample was sitting in a space with homogeneous labeling. This area is hence not vulnerable to adversarial attacks. In contrast, we observe larger trees on Fig 3.15, which indicates that the points are either sitting on adversarial regions, error regions, or decision boundaries. The surrounding of those samples has mixed labeling which suggests that the region is vulnerable to adversarial attacks. Hence, we can use the trees as an indicator of blackbox behavior on different sub-spaces. The trees can be generated locally for samples in any type of dataset. This can give insight into the blackbox behavior around different samples. The trees provide more insight into the blackbox behavior than merely testing

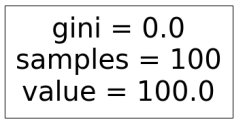

Figure 3.14: Tree trained on a sample sitting on a non adversarial zone

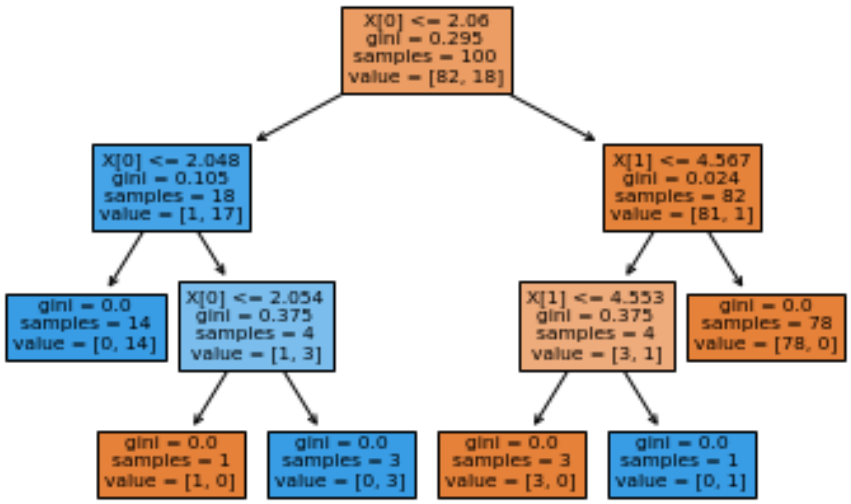

Figure 3.15: Tree trained on a sample sitting on an adversarial zone

whether the ball around the point is label homogeneous. If we find several large trees, for samples in certain regions, we can identify that region as adversarial, error, or decision boundary region. Additionally, we also investigated the vulnerability of the blackbox model using nearest neighbor and adversarial loss calculations (Algorithm 3, Chapter 4). Using these metrics, blackbox behavior can be understood further and adversarial samples and regions can be detected. We investigate adversarial examples further in the next chapter.

3.6 Discussion

From our systemic experimental evaluations, we found decision trees to be interpretable and good surrogates to a more complex model (neural networks, random forests). They can explain both local and global behavior of complex models. The simpler models can be used as a global explanation for the model performance on the training data and also on areas that are not originally covered by the domain area. However, we found surrogate models to be a misleading indicator of data shift. The trees can have the same structure under two different marginal distributions and hence cannot detect marginal data shift.

We also observed that different types of distribution for generating unlabeled data affect how faithful the surrogate model is to the teacher model. We see that only Gaussian or a combination of generated Gaussian and Uniform points for training the surrogate model, results in better overall performance during all three tests: Gaussian, Uniform, and Blend-Gaussian Uniform Test. BlendGaussUni and Gauss Extract trees can hence provide accurate and faithful interpretation to blackbox models. The tests conducted with points from different distributions are also a good measure of evaluation. These can give insight into the model performance on areas that are not covered by the original data.

From both our local and global investigation of blackbox models, we observed that models are often susceptible to adversarial attacks. The generated decision boundaries of the models were often vulnerable to slight perturbations. Hence, for our next chapter, we look into adversarial robustness. We theoretically analyze the notion of adversarial robustness and introduce an adaptive adversarial robustness measure and data augmentation. We investigate empirically on both real and synthetic datasets.

Chapter 4

A novel measure for adversarial robustness

4.1 Overview

In this chapter, we carry on our work from the previous chapter where we investigate adversarial samples. We firstly show that robust-Bayes and 0/1-Bayes are identical if and only if the distribution is r-separated. We also show that for every r, there exists a distribution where the two r-Bayes and 0/1-Bayes differ significantly as functions. Next, we introduce the margin-rate, as a relaxed measure of r-separateness and relate it to suitable choices of r. We formally show that a slightly too largely chosen robustness parameter can lead to undesirable effects. Thus we argue that the robustness parameter r of an adversarial loss should be locally adaptive. We then introduce our new adaptive robust loss, introduce its empirical version and develop a novel adaptive-robust data-augmentation paradigm. We finally provide an extensive empirical evaluation of our novel adaptive robust loss and comparison of the adaptive-robust data augmentation with the fixed-range data augmentation.

4.2 Adversarial robustness

Deep learning methods have enjoyed phenomenal successes on a wide range of applications of predictive tasks in the past decade. However, it has been demonstrated that, while these networks are often highly accurate at making predictions on natural data inputs, the performance can degrade drastically when inputs are slightly manipulated [99]. Flipping a few pixels in an image, a perturbation that is not perceivable by humans, can lead to misclassifications by the trained network. These unexpected and seemingly erratic behaviors of deep learning models have caused substantial concern over their reliability and trustworthiness. Particularly so, if these models are to be employed in applications where vulnerability to manipulations may have fatal consequences (for example if learning-based vision technologies are to be employed in self-driving cars).

Recent years have seen a surge in studies aiming to enhance the robustness of deep learning [3, 18, 39]. Practical approaches are often aimed at smoothing either on the model or on the training data level. By data-augmentation, the training data gets artificially augmented with perturbations of natural inputs as a way to promote robustness of the model during training [110, 115]. Alternatively, a trained model gets smoothed during post-processing, to not suffer sudden switches of the output class in areas where natural inputs occur [23, 89].

Theoretical studies on the problem of adversarial robustness have focused on exploring how adversarial robustness can be phrased in terms of a modified loss function and how this modified notion of loss affects learnability, both in terms of statistical and computational aspects [41, 66, 67, 114]. However, both theoretical studies and practical heuristics developed in the context of promoting robustness to adversarial attacks, are typically aimed at a fixed notion of smoothness with a fixed degree of perturbations that the model should be made robust to.

For this thesis, we take a step back, and analyze the question of when a robustness

requirement is plausible with respect to the underlying data-generating process. It has been observed before that a requirement of hard margins on a learned predictor (enforcing the learned predictor to assign constant output label in balls of fixed size around input points) can be at odds with achieving high accuracy, even if the data-generating distribution, in principle allows for accurate prediction [29, 41]. In this work, we formally argue that robustness requirements should be aligned with the underlying data-generating process and that such an alignment inherently requires a *locally adaptive notion of robustness*.

More specifically, we start by reviewing properties of Bayes optimal classifiers with respect to both standard classification loss (0/1-loss) and the most often employed notion of adversarial loss (which requires that the predictor is constant in balls of a fixed radius in addition to being accurate). We show that there are various, and natural examples of distributions, where the optimal classifiers with respect to the classification and robust loss differ drastically (they assign different labels on a proportion of mass 1/2 of the space).

We then show that the possibility of having predictors of low robust loss implies that the distribution is clusterable in a strong sense and that on such clusterable tasks, we can choose a robustness parameter so that the optimal predictors have similar loss values (in terms of classification and robust loss). However, we also show that choosing the robustness parameter slightly too large, even on such strongly clusterable tasks, can bring back the phenomenon of the optimal predictors disagreeing on a proportion of probability mass 1/2. This implies that, in these situations, any learning method that is consistent (converges to the best possible loss as training data set size increases) with respect to one loss is not consistent with respect to the other.

This motivates our proposition of redefining the robustness requirement. We argue that robustness is inherently a *local* property and that learned predictors should thus satisfy a local notion of robustness that is in line with the underlying data-generating process. While such a requirement can not readily be phrased as a loss function (that operates on a

pair of predictor and input/output data instances), we derive a natural empirical version of this requirement. This allows for evaluating the requirement on datasets. Further, we argue that our notion of locally adaptive robustness yields a natural paradigm for data augmentation, which adheres to the margin properties of the data-generating distribution.

4.3 Literature

Enhancing robustness to adversarial attacks has received an enormous amount of research attention in recent years, in particular in terms of practical advancements [3, 18, 39]. We will focus our discussion of prior work on studies relating to theoretical aspects of learning under robust loss.

Most recent theoretical studies focus on the parametric setup and analyze how introducing a robustness requirement may affect statistical convergence of the induced loss classes [5, 25, 67, 90, 114], whereas others have focused on computational implications [7, 66]. In particular, that there can be arbitrarily large gaps between the sample complexity of learning a hypothesis with respect to classification versus robust loss [25, 67]. Tsipras et al. [103] shows the discrepancy between binary and robust loss through a concrete constriction. Several studies have derived convergence bounds for classification under adversarial manipulations for fixed hypothesis classes [6, 15, 34]. Yang et al. [113] explicitly derives the connection between robustness and Lipschitzness and analyzes the robustness of the nearest neighbor classifier under r-separateness. Similarly, Gal et al. [36] uses a Bayesian framework of analysis and derives guarantees under a (fixed) r-separateness assumption.

Most related to our work are recent studies that also discuss possible options (and their implications) for phrasing a robust loss [29, 41], as well as recent studies that analyze and derive properties of optimal predictors under the robust loss and their relation to nearest neighbor predictors [11, 108]. The latter work is the first to formally study non-parametric learning for robust classification and proposes a method of data-preprocessing, and proves

implied consistency. However, robustness is considered only with respect to a fixed robustness parameter, and the pre-processing consists of pruning rather than augmenting the data.

Similar to our work, Yang et al. [112] introduces and analyzes the form of the robust-optimal predictor. It also develops a concrete attack and defense mechanism. Also, Zhang et al. [117] develops a surrogate loss for the adaptive robust loss and analyzes trade-offs between the losses in terms of this surrogate. Though different from our research, Khoury et al. [53] focuses specifically on providing theoretical insights to the phenomenon of adversarial examples due to the data-distribution sitting on a lower-dimensional manifold. This is an aspect that we only illustrated in our experiments.

Finally, we note that the relationship between local adaptivity and non-parametric methods (for example nearest neighbor methods) is well established and our work builds on this. In particular, it has been shown that nearest neighbor methods' convergence can be understood and quantified in terms of local smoothness properties of the underlying data-generating process for regression [55] as well as for classification tasks [21], and notions of clusterability of classification tasks have been broadly studied in the context of semi-supervised learning [87, 104].

4.4 Formal framework for adversarially robust classification

If X is equipped with a metric dist (for example the usual Euclidian distance metric in $\mathbb{R}^d$), then a natural choice for the set of perturbations at x is a ball $\mathcal{B}_r(x) = \{z \in X \mid \text{dist}(x, z) \leq r\}$ of radius r around x. For an $x \in X$ and $h \in \mathcal{H}$, we say that $x' \in \mathcal{B}_r(x)$ is an *adversarial* point of x with respect to h if $h(x) \neq h(x')$. We use the following definition of the adversarially robust loss:.

We consider the most commonly used notion of an (adversarially) robust loss [67, 111] and adopt some of the notation from [5]. We then define the robust loss as:

$$\ell^r(h, x, y) = \mathbb{1}\left[\exists z \in \mathcal{B}_r \ : \ h(z) \neq y\right]. \tag{13}$$

One can show that, this implies that $\ell^r(f, \cdot, \cdot)$ is a measurable function for all $f \in \mathcal{F}$ and all $r > 0$ and all P [5].

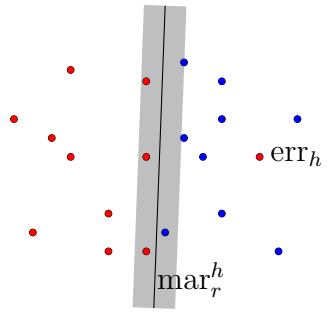

Figure 4.1: Components of robust loss

Note that, we have $\ell^r(h, x, y) = 1$ if and only if h makes a mistake on x with respect to label y, or, there is an r-close instance $z \in \mathcal{B}_r(x)$ that h labels different than x, that is, x is r-close to h's decision boundary.

The first condition holds when (x, y) falls into the *error region*,

$$\text{err}_h = \{(x, y) \in X \times \mathcal{Y}) \ \mid \ h(x) \neq y\}. \tag{14}$$

The second condition holds when x lies in the *margin area* of h. The following definition makes this notion explicit. Let $h \in \mathcal{F}$ be some hypothesis. We define the *margin area* of h, as the subset $\text{mar}_h^r \subset X$ defined by

$$\text{mar}_r^h = \{x \in \mathcal{X} \ \mid \ \exists z \in \mathcal{B}_r(x) : h(x) \neq h(z)\} \tag{15}$$

We can define notions of a Bayes classifier, and consistency of a learner $\mathcal{A}$ with respect to the robust loss analogously to these notions for the binary loss. We will denote the robust-Bayes classifier by h^{rB} and the robust Bayes risk by $\mathcal{L}_P^{rB} = \mathcal{L}_{h^{rB}}^P$. We will often simply refer to the Bayes predictors as the 0/1-optimal or the r-robust optimal predictors. We note that these optimal predictors are not unique, in particular in the case that the support of the marginal $P_\mathcal{X}$ does not cover the full space. For example, if the data-generating distribution is supported on a lower-dimensional manifold, then a 0/1-optimal predictor is only uniquely determined on that manifold (and even there only with exception of 0-mass subsets). Similarly, r-robust optimality can be fulfilled by various predictors if the data-generating distribution is strongly clusterable (see Definition 6).

4.5 ROBUSTNESS AND MARGINS

In this section, we start by investigating the implications of the existence of a low robust-loss classifier and the differences between low binary and low robust loss. We show that the optimal classifiers with respect to these losses can differ significantly, implying that optimizing for one can strongly hurt performance with respect to the other. We then analyze the relationship between the existence of robust classifiers and margin (or clusterability) properties of the underlying data-generating process and argue that, while clusterability implies the existence of robust classifiers with respect to *some* robustness parameter r, using a fixed robustness parameter can again contravene the intention of deriving predictors that are both accurate and as robust as possible.

4.5.1 Binary optimal versus robust optimal

It has been noted before that the definition of the r-robust loss implies that, even in situations where the 0/1-Bayes risk is 0, that is where the labels are deterministic, no

classifier may have 0 robust loss [29, 41]. In fact, it is not difficult to see that the existence of a classifier h with $\mathcal{L}_P^r(h) = 0$ implies that the distribution is *clusterable*, that is, $P_{\mathcal{X}}$ is supported on r-separated regions of $\mathcal{X}$ and these regions are label-homogeneous: $\mathcal{L}_P^r(h) = 0$ implies that, in particular, $\mathcal{L}_P^{0/1}(h) = 0$, that is the labeling is deterministic. In addition, we must have $P(\mathrm{mar}_h^r) = 0$, which implies that any point x in the support of $P_{\mathcal{X}}$ with $h(x) = 1$ has distance at least $2r$ from any point in that support with $h(x) = 0$. In this case, this function $h = h_P^B = h_P^{rB}$ is optimal with respect to both losses.

More generally, even if the labels are not deterministic, the optimal robust loss is larger than the optimal 0/1-loss if and only if Bayes classifier does not have a strict margin (independently of whether the labels are deterministic).

Theorem 2. *We have $\mathcal{L}_P^{rB} = \mathcal{L}_P^B$ if and only if there exists a 0/1-optimal classifier h_P^B with*

$$P_{\mathcal{X}}(\mathrm{mar}_{h_P^B}^r) = 0.$$

Proof. We first assume that $P_{\mathcal{X}}(\mathrm{mar}_h^r) > 0$ for all classifiers h that are 0/1-optimal. We fix one of them and denote it by h_P^B. Then $\mathcal{L}_P^r(h_P^B) > \mathcal{L}_P^{0/1}(h_P^B) = \mathcal{L}_P^B$, since on every point in its margin area, h_P^B suffers binary loss at most 0.5, while it suffers robust loss 1. Outside the margin area the loss contributions are identical for both loss functions. Furthermore, for any classifier h that is not 0/1-optimal, we have $\mathcal{L}_P^r(h) \geq \mathcal{L}_P^{0/1}(h) > \mathcal{L}_P^B$. Thus, independently of whether an optimal robust classifier h_P^{rB} is also 0/1-optimal or not, we have

$$\mathcal{L}_P^{rB} = \mathcal{L}_P^r(h_P^{rB}) > \mathcal{L}_P^B$$

As for the other direction, if there is a 0/1-optimal classifier h_P^B with $P_{\mathcal{X}}(\mathrm{mar}_{h_P^B}^r) = 0$, then it follows immediately, that this classifier is also optimal with respect to the robust loss and its robust loss is identical to its binary loss. Thus $\mathcal{L}_P^{rB} = \mathcal{L}_P^B$. □

Moreover, we will now show, that if the data-generating distribution does not have

a margin in the above strong sense (that is, there exists a 0/1-optimal predictor with a 0-weight margin area), then the optimal classifiers with respect to 0/1-loss and r-robust loss can differ significantly. This is independent of whether the labels are deterministic or stochastic.

Theorem 3. *Let $r > 0$ be a robustness parameter. There exist distributions P such that, for any predictors h_P^B and h_P^{rB} that are optimal with respect to 0/1-loss and r-robust loss respectively with*

$$P_{\mathcal{X}}[h_P^B \,\Delta\, h_P^{rB}] \;=\; \frac{1}{2},$$

where $h_P^B \,\Delta\, h_P^{rB} \;=\; \{x \in \mathcal{X} \;\mid h_P^B(x) \;\neq\; h_P^{rB}(x)\}$ is the set of domain points on which the two optimal classifiers differ.

Proof. We consider a distribution P, where $P_{\mathcal{X}}$ is supported (uniformly) on just two points x_0 and x_1 at distance less than r from each other. x_0 is always generated with label 0 and x_1 is always generated with label 1. Clearly, the 0/1-optimal classifier h_P^B labels accordingly: $h_P^B(x_0) = 0$ and $h_P^B(x_1) = 1$, resulting in $\mathcal{L}_P^{0/1}(h_P^B) = 0$. However, this classifier has largest possible r-robust loss: $\mathcal{L}_P^r(h_P^B) = 1$, since both points are at distance less than r from a point that h_P^B labels differently. On the other hand, any constant function h_c has robust loss $\mathcal{L}_P^r(h_c) = 1/2$, since it's margin are has weight 0 and it mislabels with probability $1/2$. This is optimal with respect to the r-robust loss. Thus, we showed that $P_{\mathcal{X}}[h_P^B \,\Delta\, h_P^{rB}] \;=\; \frac{1}{2}$. □

This example shows that binary and robust optimal predictors can differ vastly. In particular, when the robustness parameter is not chosen suitably, optimizing for one can be strongly sub-optimal (by a difference of $1/2$ in the respective loss) for the other. More formally, *any learning method, will be inconsistent with respect to one of the two losses in question.*

Of course, in the above example, the robustness parameter and distribution are constructed to not match suitably. For the particular distribution constructed, halving the robustness parameter would solve the issue.

4.5.2 Choosing a robustness parameter

In the previous section, we saw that, if the distribution is "clusterable" (in the sense that $P_{\mathcal{X}}(\mathrm{mar}^r_{h^B_P}) = 0$, for some 0/1-optimal classifier h^B_P), then the robust optimal and 0/1 optimal predictors coincide. However, this is a very strong "clusterability" or "separability" assumption on the data-generating process. In this section, we show that, in general, we can choose the robustness parameter r in dependence on "how clusterable" the distribution P is and on how close we would like the optimal predictors to be.

Note that, for a fixed predictor h, we have $P_{\mathcal{X}}(\mathrm{mar}^r_h) \geq P_{\mathcal{X}}(\mathrm{mar}^{r'}_h)$ if $r \geq r'$. Thus, the function

$$\phi^h_P(r) = P_{\mathcal{X}}(\mathrm{mar}^r_h)$$

will monotonically decrease to 0 as r goes to 0 for any predictor h. If h is a Bayes predictor, then the rate at which $\phi^h_P(r)$ converges to 0 as $r \to 0$, can be viewed as a measure of "how clusterable" the data- generating process is, that is, how fast the density of the marginal $P_{\mathcal{X}}$ vanishes towards the boundary between the two label classes.

Definition 6. *Let P be a distribution over $\mathcal{X} \times \{0, 1\}$ and let h^B_P be Bayes optimal classifier with pointwise smallest margin-rate $\phi^{h^B_P}_P(r)$. Then we define* margin-rate *of P as the function*

$$\Phi_P(r) = \phi^{h^B_P}_P(r).$$

and call h^B_P a margin-optimal *Bayes predictor. If there exists an $r > 0$ such that $\Phi_P(r) = 0$, then we call the distribution P* strongly clusterable.

In the case of deterministic labels, the margin rate coincides with the notion of *Probabilistic Lipschitzness*, which has been used as a notion of clusterability of the data-generating process in the context of active, semi-supervised learning [104]. In the case of stochastic labels, this notion is related to the geometric noise exponent [97]. However, in contrast to that notion, we do not incorporate bounds on the amount of stochasticity.

We now show that the margin rate can be used to choose a margin parameter for which the optimal robust and optimal 0/1 predictors will be close.

Theorem 4. *Let P be a data-generating distribution over $\mathcal{X} \times \{0,1\}$, let $\Phi_P : \mathbb{R}^+ \to [0,1]$ denote its margin rate, and let h_P^B denote the 0/1-optimal classifier defining the margin rate. For every $\epsilon > 0$, if we let $r \in \Phi_P^{-1}([0,\epsilon])$, then we have*

$$\mathcal{L}_P^r(h_P^B) \leq \mathcal{L}_P^{rB} + \epsilon.$$

In addition, if the labeling of P is deterministic, we have

$$P_{\mathcal{X}}[h_P^B \;\Delta\; h_P^{rB}] \leq \epsilon$$

for any robust optimal classifier h_P^{rB}.

That is, we can choose the robustness parameter so, that the robust loss of the Bayes-predictor is close to the optimal robust-loss.

Proof. Due to the way we chose the robustness parameter r here, we immediately get

$$\mathcal{L}_P^r(h_P^B) \leq \mathcal{L}_P^{0/1}(h_P^B) + \epsilon = \mathcal{L}_P^B + \epsilon$$

since $P(\mathrm{mar}_{h_P^B}^r) \leq \epsilon$. We need to argue, that no other classifier h can have a significantly smaller robust loss. As in the proof of Theorem 2, we observe that, we have $\mathcal{L}_P^r(h) \geq$

$\mathcal{L}_P^{0/1}(h) \geq \mathcal{L}_P^B$ for any classifier h. Thus, in particular $\mathcal{L}_P^r(h_P^{rB}) = \mathcal{L}_P^{rB} \geq \mathcal{L}_P^B$, which yields the first claim.

Now we assume that the labeling of P is deterministic. This implies that $\mathcal{L}_P^{0/1}(h_P^B) = 0$, thus $\mathcal{L}_P^r(h_P^B) = \mathcal{P}_\mathcal{X}(\mathrm{mar}_{h_P^B}^r)$. Let h_P^{rB} be a robust-optimal classifier. By definition of being robust-optimal, we have $\mathcal{L}_P^r(h_P^{rB}) \leq \mathcal{L}_P^r(h_P^B) = \mathcal{P}_\mathcal{X}(\mathrm{mar}_{h_P^B}^r) \leq \epsilon$. Thus, in particular $\mathcal{L}_P^{0/1}(h_P^{rB}) \leq \epsilon$, which, in the case of deterministic labels implies $P_\mathcal{X}[h_P^B \,\Delta\, h_P^{rB}] \leq \epsilon$. □

Thus, while a clusterability assumption can yield closeness in loss values of the optimal predictors, it implies closeness of the actual functions only if the labeling is, in addition deterministic. We next argue that the above theorem's statements can not be improved upon in this regard. We show that the assumption of deterministic labels is *necessary* for the second part of the statement. On top of this, we show that even if the labels are deterministic, choosing a robustness parameter slightly larger than what the theorem suggests can again yield large differences in the optimal predictors (as functions, not just in terms of their loss values). We start by showing that the assumption of deterministic labels in Theorem 4 is necessary.

Observation 5. *Let $\epsilon > 0$ be given. There exists a data-generating distribution P over $\mathbb{R}^2 \times \{0,1\}$ with linear margin rate $\Phi_P : \mathbb{R}^+ \to [0,1]$, $\Phi_P(r) = 0.5r$ such that, for any $r \in \Phi_P^{-1}([0,\epsilon])$, we get*

$$P_\mathcal{X}[h_P^B \,\Delta\, h_P^{rB}] = \frac{1}{2}$$

Proof. We consider with uniform marginal over two rectangles in $\mathbb{R}^2$: We set $R_1 = [-2,-1] \times [-1,1]$ and $R_2 = [1,2] \times [-1,1]$. Further, we set the regression function

$$\mu(x_1, x_2) = \begin{cases} \frac{1}{2} + \frac{\epsilon}{2} \text{ if } x_2 \geq 0 \\ \frac{1}{2} - \frac{\epsilon}{2} \text{ if } x_2 \leq 0 \end{cases}$$

Now it follows that a 0/1-optimal predictor is $h_P^B = \mathbb{1}[x_2 \geq 0]$ while, for any $r \geq \epsilon/2$, we

have $h_P^{rB} = \mathbb{1}[x_1 \geq 0]$, thus $P_{\mathcal{X}}[h_P^B \,\Delta\, h_P^{rB}] = \frac{1}{2}$. □

Next, we argue that, even under deterministic labels, choosing a robustness parameter slightly larger than implied by Theorem 4, can yield largely differing optimal predictors.

Observation 6. *Let $\epsilon > 0$ be given. There exists a data-generating distribution P over $\mathbb{R} \times \{0,1\}$ that is strongly clusterable, such that, for any $r > \sup Phi_P^{-1}([0,\epsilon]$, we have $P_{\mathcal{X}}[h_P^B \,\Delta\, h_P^{rB}] = \frac{1}{2}$.*

Proof. We can use the same construction as in the proof of Theorem 3. □

4.5.3 Towards local robustness

In the previous sections, we have shown that the clusterability (as well as the amount of stochasticity in the labels) of the underlying data-generating process has a strong effect on what is a suitable robustness parameter to aim for and that, choosing the robustness parameter slightly too large, can result in inconsistent learning. We now argue that, even if the distribution is strongly clusterable and the labels are deterministic, then choosing a uniform robustness parameter may not result in the desired outcomes.

To see this, we consider a distribution over domain $\mathbb{R}^2 \times \{0,1\}$, where the support is distributed uniformly on four points, $(-1, 0.9), (-1, 1.1), (1, 0.9), (1, 2)$. Then predictor $h(x_1, x_2) = \mathbb{1}[x_2 \geq 1]$ is 0/1-optimal and also r-robust optimal for any $r \leq 0.1$. However, we may prefer a predictor h^* that keeps a larger distance from the point $(1, 0.9)$, see illustration in Figure 4.5.3 and is equally optimal with respect to the 0.1-robust loss.

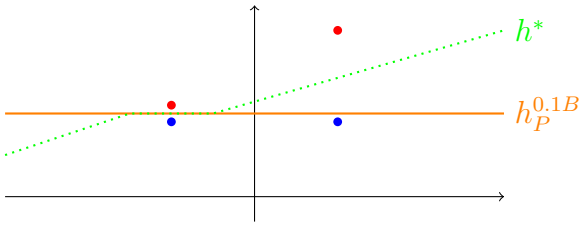

Figure 4.2: Uniform robustness requirement unsuitable.

4.6 REDEFINING THE ROBUSTNESS REQUIREMENT

In the previous section, we have argued that using a fixed robustness parameter r can lead to inconsistencies (in the sense that the optimal predictors with respect to binary and robust differ vastly) and that even under conditions where the optimal predictors can coincide (strong clusterability and suitably chosen robustness parameter), optimizing for the robust loss can lead to classifiers that do not reflect our intuition about an optimally robust predictor (Section 4.5.3). Ideally, we would like a learned predictor to be *everywhere as robust as possible*. We will next formalize this intuition using the notions developed in the previous section. We then propose an empirical paradigm based on data-augmentation to realize the novel objective.

4.6.1 A local robustness objective

Earlier work has considered how robustness can be defined as a requirement of the predictor to be *accurate* in balls around input points versus being *constant* in balls around input points [29, 41] and discussed implications of these definitions. While the former requirement better reflects what is actually desired (as well as the fact that being constant in balls can induce contradictory requirements to accuracy), it can not be phrased as a loss function $\ell : \mathcal{F}, \mathcal{X}, \mathcal{Y} \to \mathbb{R}$ [5], and thus there is no obvious empirical version of this

requirement.

We propose to phrase robustness in relation to a margin optimal Bayes predictor. A learned predictor should assign a constant label in a ball $\mathcal{B}_r(x)$ around a point x if a margin optimal Bayes predictor does so. For a predictor h and domain point x, we let $\mathcal{B}^h(x)$ denote the largest ball around x on which h assigns a constant label (potentially, the radius of this ball is 0, in which case we define $\mathcal{B}^h(x) = \{x\}$).

We will now assume that the data-generating distribution is such that the regression function is nowhere equal to 0.5. That is, for every point in the support of $P_{\mathcal{X}}$, the Bayes optimal predictor is uniquely defined and partitions the support into areas $\mathcal{X}^0$ and $\mathcal{X}^1$ where the Bayes classifier classifies 0 and 1 respectively. We then define the *margin optimal Bayes predictor* outside of the support of $P_{\mathcal{X}}$ by nearest distance to $\mathcal{X}^0$ and $\mathcal{X}^1$.

Definition 7 (Adaptive robustness)**.** *Let P be a data-generating distribution and let h_P^B denote a margin-optimal Bayes predictor, and h an arbitrary predictor. Then we define* adaptive robust loss ℓ^{ar} *as*

$$\ell^{ar}(h, x, y) = \mathbb{1}\left[h(x) \neq y \ \vee \ \mathcal{B}^{h_P^B}(x) \not\subseteq \mathcal{B}^h(x)\right]$$

This definition implies that at least for h_P^B the robust loss coincides with the binary loss. We note that similar to the requirement that a predictor should be accurate in a ball of fixed radius, the above-proposed loss is not technically a valid loss function, since it depends on h_P^B rather than just on h, x, and y. This implies that it can not straightforwardly be estimated from a data-sample. However, we next propose a substitute notion of empirical loss for the adaptive robust loss.

4.6.2 Empirical adaptive robust loss

In this subsection, we suggest an empirical version of the adaptive robust loss. Let $S = ((x_1, y_1), \ldots, (x_n, y_n))$ be a labeled dataset. For a labeled domain point (x, y) we let $\rho_S(x)$ denote the distance from x to its nearest neighbor with opposite (or different in the case of more than two classes) label in S:

$$\rho_S(x, y) = \min_{i \in [n]} \{\|x_i - x\| \mid (x_i, y_i) \in S, y_i \neq y\}.$$

In the (degenerate) case that no such point in S has a label different from y (that is, all points in S have the same label), we set $\rho_S(x, y)$ to ∞ (or the diameter of the space). Note that $\rho_S(x, y)$ is well defined for points $(x, y) = (x_i, y_i) \in S$ from the dataset S itself.

We now expand the dataset S by replacing each point with a (constant labeled) ball of radius $c \cdot \rho_S(x_i, y_i)$, for some (to be chosen) constant c.

Definition 8. *Let $S = ((x_1, y_1), \ldots, (x_n, y_n))$ be a labeled dataset. We call the collection*

$$S^c = (\mathcal{B}_{c \cdot \rho_S(x_1, y_1)}(x_1, y_1), \ldots, \mathcal{B}_{c \cdot \rho_S(x_n, y_n)}(x_n, y_n))$$

the c-adaptive robust expansion of S.

It is easy to see that, as long as $c \leq 1/2$, balls in the c-adaptive robust expansion of S overlap only if they have the same label. Thus, this expansion does not introduce any inconsistencies in the label requirements. Depending on the geometry of the data-generating process (eg. the curvature of the decision boundary of the regression function) we may also employ larger expansion parameters without introducing inconsistencies.

Using the *c-adaptive robust expansion of S*, we can define an empirical version of the adaptive robust risk for fixed-parameter c. For this, for a predictor $h : \mathcal{X} \to \mathcal{Y}$ and label y, we let $h^{-1}(y) \subseteq \mathcal{X}$ denote the part of the domain that h labels with y.

Definition 9. *Let c be an expansion parameter, $S = ((x_1, y_1), \ldots, (x_n, y_n))$ a labeled dataset and h a predictor. We define the* empirical c-adaptive robust loss of h on S *as*

$$\mathcal{L}_S^{c-ar}(h) = \frac{1}{n} \sum_{i=1}^{n} \mathbb{1}\left[\mathcal{B}_{c \cdot \rho_S(x_i, y_i)}(x_i, y_i) \not\subseteq h^{-1}(y_i)\right]$$

That is, a point $(x_i, y_i) \in S$ is counted towards the empirical c-adaptive robust empirical risk, if h does not label the whole ball $\mathcal{B}_{c \cdot \rho_S(x_i, y_i)}(x_i, y_i)$ in the expanded set with label y_i.

4.6.3 Adaptive robust data-augmentation

While the empirical c-adaptive robust risk is well defined for any predictor h and dataset S, it may, computationally, not be straightforward to verify the condition $\mathbb{1}\left[\mathcal{B}_{c \cdot \rho_S(x,y)}(x, y) \not\subseteq h^{-1}(y)\right]$, that is, to verify whether both label classes have non-empty intersections with some ball in the space. A natural estimate is to use m uniform sample points $z^1, \ldots, z^m$ from the ball $\mathcal{B}_{c \cdot \rho_S(x,y)}(x)$ and verify whether h labels all of these with y.

Similarly, for training purposes, we may want to use an sample version of the c-adaptive robust expansion of S. We call this the *m-sample-c-adaptive robust augmentation of S*. The so augmented dataset S^{mc} is a set of labeled domain points and can be used as a training data-set for a learning algorithm.

Definition 10. *Let $S = ((x_1, y_1), \ldots, (x_n, y_n))$ be a labeled dataset, and $m \in \mathbb{N}$. We call the collection*

$$S^{mc} = ((z_1^1, y_1), \ldots, (z_1^m, y_1), \ldots (z_n^1, y_n), \ldots, (z_n^m, y_n)),$$

where every z_i^j is uniformly sampled from the ball $\mathcal{B}_{c \cdot \rho_S(x_i, y_i)}(x_i)$, the m-sample-c-adaptive robust augmentation of S.

4.7 Experiments on synthetic and UCI dataset

To further validate our proposed adaptive robust data augmentation method, we present a set of illustrative experiments on various synthetic datasets. To allow for visualizations, we generate data from a "lower-dimensional manifold" in two dimensions. It has been conjectured that the data being supported on a lower-dimensional manifold is a source of the phenomenon of vulnerability to small perturbations. We term our synthetic shapes: **Sines, S-figure, NNN, circles, boxes**, see Figures 4.4 and 4.3.

The original support of data generating distributions can be seen as the green and blue lines in the first column of Figure 4.3, blue and green points representing points from the two classes. We train a ReLU Neural Network with 2-hidden layers (of 10 neurons each) data points drawn from these shapes. We also augment the training datasets with both fixed and adaptive expansion parameter.

For fixed expansion parameter, we iteratively increase the parameter in a fix sequence, $(0.1, 0.5, 1, 2, ..., 16)$. These expansion parameters were chosen based on the range of the attribute values in the datasets. For each sample in a d-dimensional dataset, a d-dimensional sphere is generated where the radius is the fixed-parameter and the current sample is the center of the sphere. Four new points are then generated in this sphere for each sample. Hence, the dataset is expanded to four times its original size after fixed-parameter expansion.

Similarly, expansion is done with adaptive expansion parameter. The key difference is in the calculation of the radius of the sphere. The nearest neighbors of each sample are investigated and the neighbor which has a different label with respect to the current sample is selected. A fraction of the distance between the current sample and the chosen neighbor is used as the radius for the sphere generation. Algorithm 2 describes the process of adaptive data augmentation.

Each of the middle columns in Figure 4.3 corresponds to augmentation with a fixed

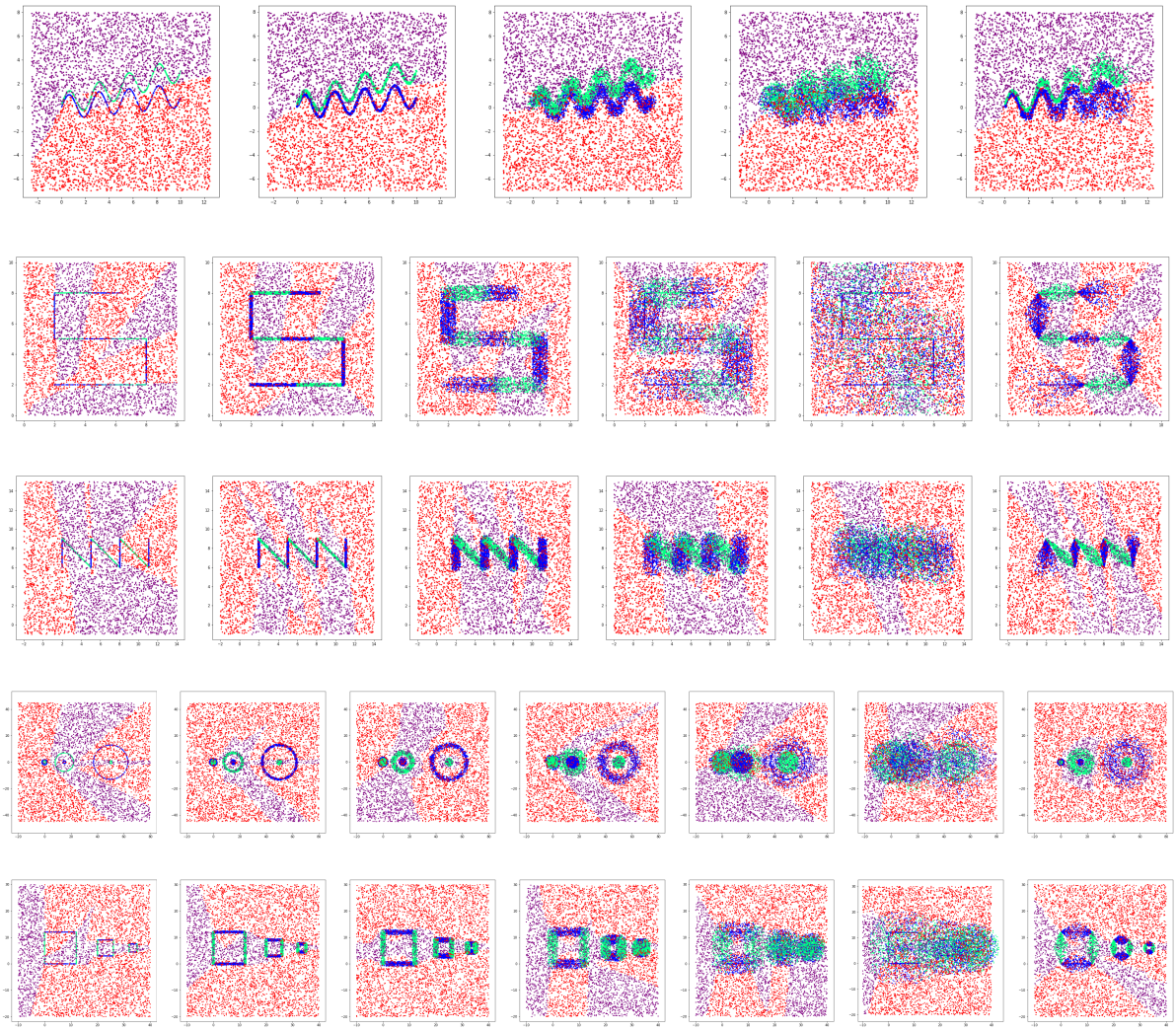

Figure 4.3: ReLU networks trained on data from a one-dimensional manifold in two-dimensional space, labeled using two classes (blue and green here). The various shapes by row: **Sines, S-figure, NNN, circles, boxes**. Left-most: original training data; various middle images: training data augmented using increasing expansion parameters; right-most: training data robust-adaptive expanded. We use data generated uniformly at random from the ambient space to illustrate the network's labeling (red and purple). Using just original training data, or only slightly augmented data, we observe that the network's decision boundary is often close to the manifold.

Algorithm 2: Adaptive Data Augmentation

Input: d dimensional training dataset with N instances x with labels y
Output: Augmented Dataset
for $i \leftarrow 0$ **to** N **do**
1. Find the nearest neighbor x_{nn} of sample x_i where $y_i \neq y_{nn}$
2. Calculate the *euclidean* distance p between these two sample x_i, x_{nn}.
3. Calculate radius, $rad = \frac{2}{3} * p$.
4. Use rad as the radius to generate the d-dimensional sphere around sample x_i
4. Generate 4 new points in the sphere for each sample x_i.
5. Label these new points with y_i; the same label as sample x_i.
6. Append these new data points to the existing dataset.

end

expansion parameter, while the last column shows the 2/3-adaptive robust augmentation of the training data. The original training dataset contains 1000 training points and the augmented datasets 5000 data points each.

We then evaluate the robust loss with various fixed robustness parameters on a test dataset drawn from the original data generating process. To estimate the robust loss, we evaluate the network on a test point (x_1, x_2) and four additional points $(x_1 - r, x_2), (x_1 + r, x_2), (x_1, x_2 - r)$ and $(x_1, x_2 + r)$ for increasing robustness parameters r (a data point is counted towards the r-robust loss here, if these four test or the four tests for an earlier tested robustness parameter $r' < r$ resulted in finding a point that the network labels differently than (x_1, x_2)). The procedure for one robust parameter r is stated in algorithm 3. Here, r_loss_{total} contain information of the previous robust loss for each sample with parameters $r' < r$.

We plot these losses in Figure 4.4. The initial point of the curves corresponds to the 0/1-loss of the trained network. Superior performance is thus a combination of a low starting point and a low continuation of a curve. We observe that the adaptive data augmentation combines achieving low classification error (the leftmost starting point of the curves) with overall good robustness for various perturbation parameters. A fixed expansion parameter on the other hand typically resulted in a higher binary loss.

Algorithm 3: r-robust loss

Input: Classifier f, N Training data samples $x \in X$ with M features, a temporary sample z, robustness parameter r, a 2D array of previous robust loss r_loss_{total}, for all training samples and parameters $r' < r$

Output: r-robust loss for a parameter r

$z \leftarrow [0] * M$
$r\_loss_{robust} \leftarrow 0$
for $i \leftarrow 0$ **to** N **do**
  if $(r\_loss_{total}[i, r'] = 1$ *for any* $r' < r)$ **then**
    $r\_loss_{robust} \leftarrow r\_loss_{robust} + 1$
    $r\_loss_{total}[i, r] \leftarrow 1$
    continue;
  end
  $r\_loss_{feature} \leftarrow 0$
  for $j \leftarrow 0$ **to** M **do**
    $z_i \leftarrow x_i$
    $z_{ij} \leftarrow x_{ij} + r$
    $t_1 \leftarrow f(z_i)$
    $z_{ij} \leftarrow x_{ij} - r$
    $t_2 \leftarrow f(z_i)$
    if $(f(x_i) \neq y_i)$ *or* $(t_1 \neq f(x_i))$ *or* $(t_2 \neq f(x_i))$ **then**
      $loss_{feature} \leftarrow loss_{feature} + 1$;
    end
  end
  if $loss_{feature} > 0$ **then**
    $r\_loss_{robust} \leftarrow r\_loss_{robust} + 1$
  end
end
return $(r\_loss_{robust})/N$

We also evaluate the adaptive robust loss on the various trained networks. To estimate the adaptive robust loss at a point (x_1, x_2), we determine its distance ρ to a point in the dataset with a different label and then generate 10 test points uniformly at random from a ball of radius 0.5ρ. If one of these gets a different label than (x_1, x_2) by the network (or if the point is mislabeled itself) it suffers adaptive robust loss 1. The table in Figure 4.5 summarizes the binary and adaptive robust losses of the various networks. We see that, the adaptive augmentation leads consistently to the lowest binary (always rank 1) and low adaptive robust loss (rank 1 and once rank 2).

Finally, we also trained ReLU neural networks on some simple UCI datasets. For each dataset, we normalized the features to take values in [0,1]. As in the experiments on the synthetic data, we trained the networks on the original data, as well as various augmented datasets, including using the 2/3-adaptive augmentation. The dataset was split into training and test data with a ratio of $80 - 20$ respectively. The r-robust loss graph on these test datasets can be seen in figure 4.6.

In Figure 4.7 and 4.5, we report the binary and adaptive robust losses of these networks. We observe, again, that the robust augmentation promotes the best performance in terms of 0/1 accuracy. Additionally, the adaptive robust loss is close to the best adaptive robust loss achieved with a fixed expansion parameter on each dataset. Using the adaptive augmentation can thus serve to save needing to search for an optimal expansion parameter on different tasks.

In summary, our initial experimental explorations here showed that the adaptive augmentation consistently yielded a robust predictor with best 0/1-loss. This confirms the intended design of an adaptive robustness and data augmentation paradigm that avoids the undesirable tradeoffs between robustness and accuracy.

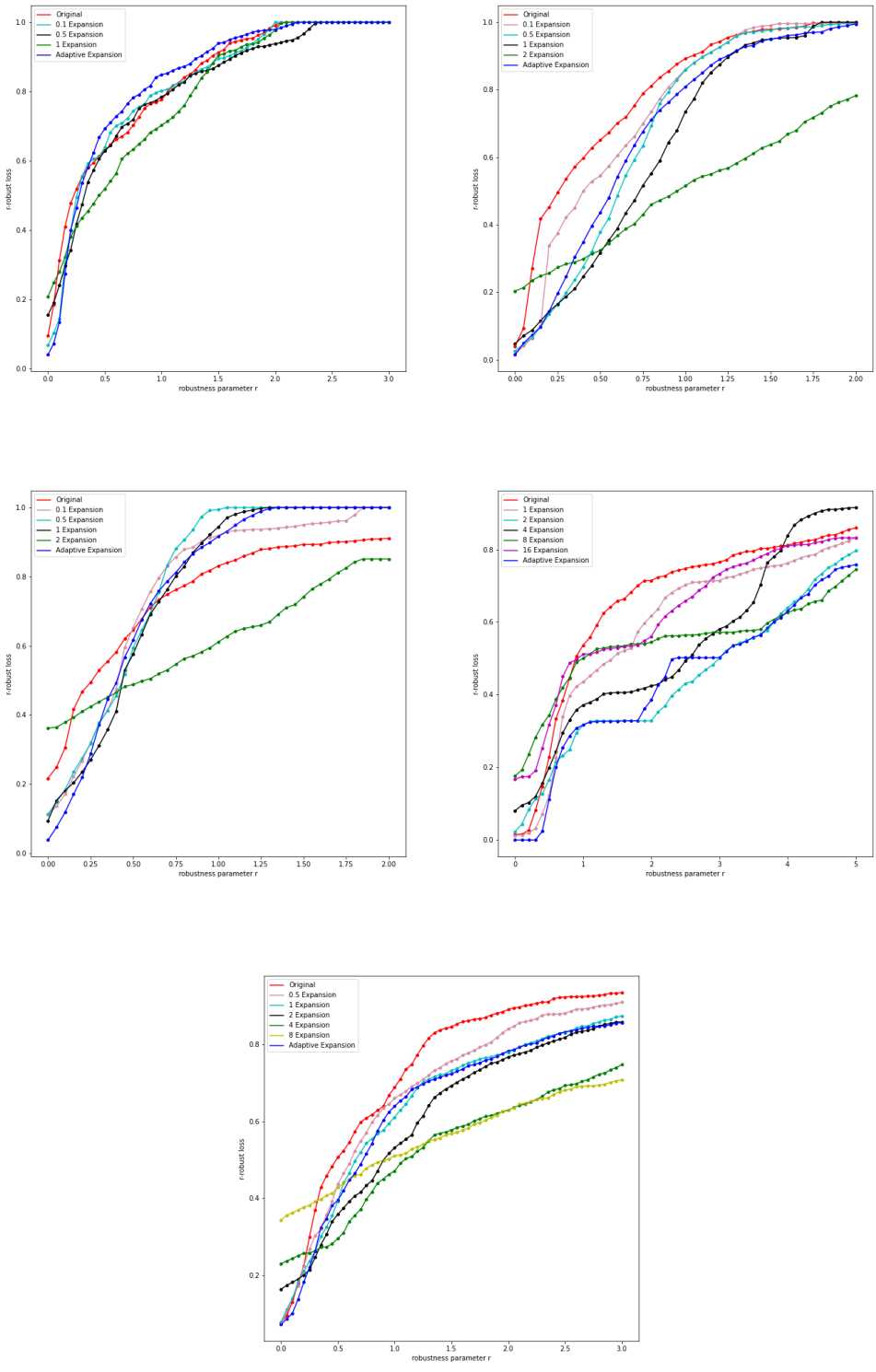

Figure 4.4: The loss curves on various synthetic datasets. From left to right: **Sines, S-figure, NNN, circles, boxes**

Dataset	Network	Adaptive Robust Loss	Binary Loss
Sines	Original	0.2882	0.104
	0.1	0.1693	0.071
	0.5	0.2443	0.147
	1	0.3116	0.177
	2	0.3521	0.208
	Adaptive	**0.1403**	**0.038**
S-figure	Original	0.3516	0.044
	0.1	0.1514	0.016
	0.5	0.0429	0.027
	1	0.0844	0.05
	2	0.2373	0.21
	Adaptive	**0.0393**	**0.017**
NNN	Original	0.3841	0.2124
	0.1	0.2609	0.1086
	0.5	0.2008	0.1048
	1	0.1969	0.0952
	2	0.386	0.3714
	Adaptive	**0.08972**	**0.04**
circles	Original	0.4483	0.0133
	0.5	0.2629	0
	1	0.3472	0.0108
	2	**0.1778**	0.0242
	4	0.3076	0.0783
	8	0.3557	0.1733
	16	0.3054	0.1633
	Adaptive	0.254	**0**
boxes	Original	0.3427	0.08
	0.5	0.2623	0.0775
	1	0.2229	0.0775
	2	0.2252	0.1667
	4	0.2839	0.2283
	8	0.4274	0.3458
	Adaptive	**0.2077**	**0.075**

Figure 4.5: Overview on the binary and adaptive robust losses of the networks trained on the various synthetic datasets with various augmentations.

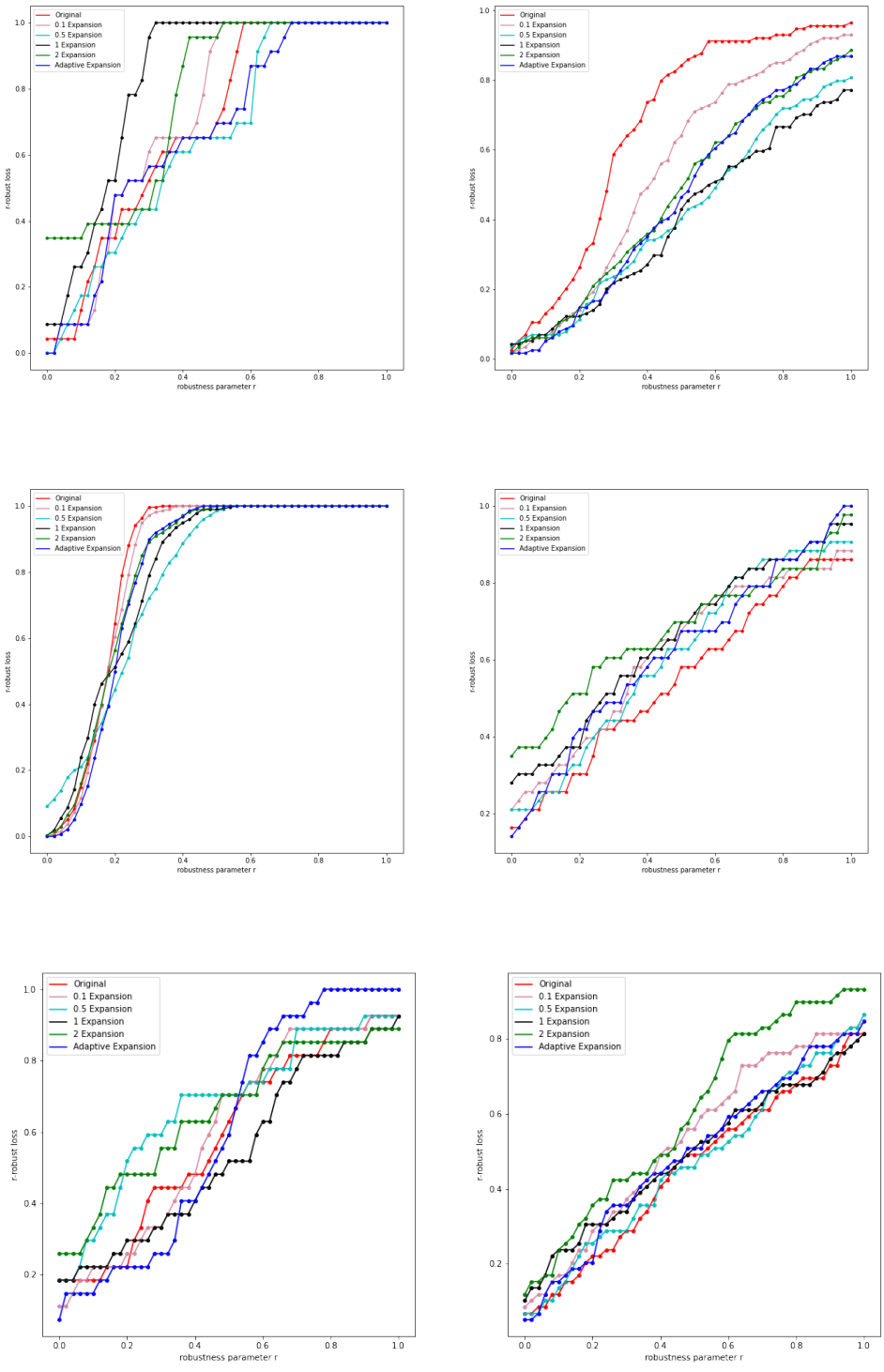

Figure 4.6: The loss curves on various UCI datasets. From left to right: **Iris, Breast Cancer, Bank Note Authentication,Heart Disease, Immunotherapy, and Parkinsons**

Dataset	Network	Adaptive Robust Loss	Binary Loss
Iris	Original	0.0957	0.0435
	0.1	**0.0783**	**0**
	0.5	0.1304	**0**
	1	0.3478	0.087
	2	0.391	0.3478
	Adaptive	0.087	**0**
Breast Cancer	Original	0.1351	0.0263
	0.1	0.0956	**0.0175**
	0.5	0.0842	0.0351
	1	0.0833	0.0439
	2	**0.0693**	**0.0175**
	Adaptive	0.0719	**0.0175**
Bank Note Authentication	Original	0.0804	**0**
	0.1	0.0479	**0**
	0.5	0.1593	0.0909
	1	0.1153	0.0036
	2	0.1058	0.0036
	Adaptive	**0.0167**	**0**
Heart Disease	Original	**0.3465**	0.1628
	0.1	0.3791	0.2093
	0.5	0.386	0.2093
	1	0.4489	0.2791
	2	0.507	0.3488
	Adaptive	0.3604	**0.1395**
Immunotherapy	Original	0.263	0.1852
	0.1	0.2926	0.1111
	0.5	0.3482	0.1852
	1	0.2333	0.1852
	2	0.437	0.2593
	Adaptive	**0.174**	**0.0741**
Parkinsons	Original	**0.1423**	0.0678
	0.1	0.1678	0.0847
	0.5	0.1542	0.0678
	1	0.2322	0.1017
	2	0.2322	0.1186
	Adaptive	0.1627	**0.0508**

Figure 4.7: Overview on the binary and adaptive robust losses of the networks trained on the various UCI datasets (test sets) with various augmentations.

Chapter 5

Conclusion

In this work, we work with the inherently interpretable method (decision trees) as means of our explanation for blackbox models and adversarial robustness as an *adaptive requirement*. For the study of interpretability, we first try to explain marginal shift using decision trees and find that it is not a suitable indicator of data shift. We then conduct experiments with the teacher-student framework where use different data distributions to generate unlabeled data to both train and evaluate the student model (decision trees). We find the surrogate to be mostly accurate and faithful to the blackbox teacher model and that their performance is strongly dependent on the unlabeled data. We also locally investigate decision trees around each data sample and find the trees to be a good explanation of local blackbox behavior.

Similarly, we motivate re-framing adversarial robustness as a requirement that should be in line with the underlying distribution's margin properties through a series of constructions where optimal classifiers for robust loss and 0/1-loss differ drastically. We propose a formal notion of such an adaptive loss, as well as an accompanying empirical version and implied data-augmentation paradigm. We believe this to be a natural and useful take on dealing with the inconsistencies (eg in terms of growing loss-class capacities, computational impossibilities, or diverging Bayes predictors) that earlier theoretical studies on learning

under adversarial loss have exhibited. We hope that our work will inspire follow-up studies in a similar vein.

5.1 Future Work

Concerning the future work, we plan to extend interpretability in regards to the logits layers of the neural network. We hope to identify out of distribution samples from the last layer of the model. Additionally, we also plan to explain marginal shift with simpler interpretable models. In terms of adversarial examples, we hope to conduct more experiments on other datasets and ensure that our adaptive robustness augmentation and loss measure can be applied to any ML applications.

Bibliography

[1] *Adversarial Perturbations of Deep Neural Networks*, pages 311–342. 2017.

[2] A. Adadi and M. Berrada. Peeking inside the black-box: A survey on explainable artificial intelligence (xai). *IEEE Access*, 6:52138–52160, 2018.

[3] Naveed Akhtar and Ajmal Mian. Threat of adversarial attacks on deep learning in computer vision: A survey, 2018.

[4] Yigit Alparslan, Ken Alparslan, Jeremy Keim-Shenk, Shweta Khade, and Rachel Greenstadt. Adversarial attacks on convolutional neural networks in facial recognition domain, 2020.

[5] Hassan Ashtiani, Vinayak Pathak, and Ruth Urner. Black-box certification and learning under adversarial perturbations. In *Proceedings of the 37th International Conference on Machine Learning, ICML 2020*, 2020.

[6] Idan Attias, Aryeh Kontorovich, and Yishay Mansour. Improved generalization bounds for robust learning. In *Algorithmic Learning Theory, ALT*, pages 162–183, 2019.

[7] Pranjal Awasthi, Abhratanu Dutta, and Aravindan Vijayaraghavan. On robustness to adversarial examples and polynomial optimization. In *Advances in Neural Information Processing Systems, NeurIPS*, pages 13760–13770, 2019.

[8] Osbert Bastani, Carolyn Kim, and Hamsa Bastani. Interpreting blackbox models via model extraction. 05 2017.

[9] V. Van Belle, B. van Calster, S. Van Huffel, J. Suykens, and P. Lisboa. Explaining support vector machines: A color based nomogram. *PLoS ONE*, 11, 2016.

[10] Shai Ben-David and Ruth Urner. On the hardness of domain adaptation and the utility of unlabeled target samples. In Nader H. Bshouty, Gilles Stoltz, Nicolas Vayatis, and Thomas Zeugmann, editors, *Algorithmic Learning Theory*, pages 139–153, Berlin, Heidelberg, 2012. Springer Berlin Heidelberg.

[11] Robi Bhattacharjee and Kamalika Chaudhuri. When are non-parametric methods robust? In *Proceedings of the 37th International Conference on Machine Learning, ICML 2020*, 2020.

[12] Steffen Bickel, Michael Brückner, and Tobias Scheffer. Discriminative learning under covariate shift. *Journal of Machine Learning Research*, 10(9), 2009.

[13] Or Biran and Courtenay Cotton. Explanation and justification in machine learning: A survey. In *IJCAI-17 workshop on explainable AI (XAI)*, volume 8, 2017.

[14] L Breiman, JH Friedman, R Olshen, and CJ Stone. Classification and regression trees. 1984.

[15] Sébastien Bubeck, Yin Tat Lee, Eric Price, and Ilya P. Razenshteyn. Adversarial examples from computational constraints. In *Proceedings of the 36th International Conference on Machine Learning, ICML*, pages 831–840, 2019.

[16] Lars Buitinck, Gilles Louppe, Mathieu Blondel, Fabian Pedregosa, Andreas Mueller, Olivier Grisel, Vlad Niculae, Peter Prettenhofer, Alexandre Gramfort, Jaques Grobler, Robert Layton, Jake VanderPlas, Arnaud Joly, Brian Holt, and Gaël Varoquaux.

API design for machine learning software: experiences from the scikit-learn project. In *ECML PKDD Workshop: Languages for Data Mining and Machine Learning*, pages 108–122, 2013.

[17] Diogo V Carvalho, Eduardo M Pereira, and Jaime S Cardoso. Machine learning interpretability: A survey on methods and metrics. *Electronics*, 8(8):832, 2019.

[18] Anirban Chakraborty, Manaar Alam, Vishal Dey, Anupam Chattopadhyay, and Debdeep Mukhopadhyay. Adversarial attacks and defences: A survey. *CoRR*, abs/1810.00069, 2018.

[19] Supriyo Chakraborty, Richard Tomsett, Ramya Raghavendra, Daniel Harborne, Moustafa Alzantot, Federico Cerutti, Mani Srivastava, Alun Preece, Simon Julier, Raghuveer M Rao, et al. Interpretability of deep learning models: a survey of results. In *2017 IEEE SmartWorld, Ubiquitous Intelligence & Computing, Advanced & Trusted Computed, Scalable Computing & Communications, Cloud & Big Data Computing, Internet of People and Smart City Innovation (SmartWorld/SCALCOM/UIC/ATC/CBDCom/IOP/SCI)*, pages 1–6. IEEE, 2017.

[20] Ashutosh Chaubey, Nikhil Agrawal, Kavya Barnwal, Keerat K. Guliani, and Pramod Mehta. Universal adversarial perturbations: A survey, 2020.

[21] Kamalika Chaudhuri and Sanjoy Dasgupta. Rates of convergence for nearest neighbor classification. In *Advances in Neural Information Processing Systems, NIPS*, pages 3437–3445, 2014.

[22] Hanjie Chen, Guangtao Zheng, and Yangfeng Ji. Building hierarchical interpretations in natural language via feature interaction detection. 2019.

[23] Jeremy M. Cohen, Elan Rosenfeld, and J. Zico Kolter. Certified adversarial robustness via randomized smoothing. In *Proceedings of the 36th International Conference on Machine Learning, ICML*, pages 1310–1320, 2019.

[24] Gabriela Csurka. Domain adaptation for visual applications: A comprehensive survey. *arXiv preprint arXiv:1702.05374*, 2017.

[25] Daniel Cullina, Arjun Nitin Bhagoji, and Prateek Mittal. Pac-learning in the presence of adversaries. In *Advances in Neural Information Processing Systems, NeurIPS*, pages 230–241, 2018.

[26] Emmanuel Gbenga Dada, Joseph Stephen Bassi, Haruna Chiroma, Shafi'i Muhammad Abdulhamid, Adebayo Olusola Adetunmbi, and Opeyemi Emmanuel Ajibuwa. Machine learning for email spam filtering: review, approaches and open research problems. *Heliyon*, 5(6):e01802, 2019.

[27] Hal Daume III and Daniel Marcu. Domain adaptation for statistical classifiers. *Journal of artificial Intelligence research*, 26:101–126, 2006.

[28] Marcos Lopez De Prado. *Advances in financial machine learning*. John Wiley & Sons, 2018.

[29] Dimitrios Diochnos, Saeed Mahloujifar, and Mohammad Mahmoody. Adversarial risk and robustness: General definitions and implications for the uniform distribution. In *Advances in Neural Information Processing Systems 31, NeurIPS*, pages 10359–10368, 2018.

[30] Finale Doshi-Velez and Been Kim. Towards a rigorous science of interpretable machine learning. *arXiv preprint arXiv:1702.08608*, 2017.

[31] Dheeru Dua and Casey Graff. UCI machine learning repository, 2017.

[32] Jane Elith, John R Leathwick, and Trevor Hastie. A working guide to boosted regression trees. *Journal of Animal Ecology*, 77(4):802–813, 2008.

[33] Hassan Ismail Fawaz, Germain Forestier, Jonathan Weber, Lhassane Idoumghar, and Pierre-Alain Muller. Transfer learning for time series classification. In *2018 IEEE International Conference on Big Data (Big Data)*, pages 1367–1376. IEEE, 2018.

[34] Uriel Feige, Yishay Mansour, and Robert Schapire. Learning and inference in the presence of corrupted inputs. In *Conference on Learning Theory, COLT*, pages 637–657, 2015.

[35] Gerhard Fischer. Lifelong learning—more than training. *Journal of Interactive Learning Research*, 11(3):265–294, 2000.

[36] Yarin Gal and Lewis Smith. Sufficient conditions for idealised models to have no adversarial examples: a theoretical and empirical study with bayesian neural networks, 2018.

[37] Joseph Gatto, Ravi Lanka, Yumi Iwashita, and Adrian Stoica. Single sample feature importance: An interpretable algorithm for low-level feature analysis. *arXiv preprint arXiv:1911.11901*, 2019.

[38] Leilani H Gilpin, David Bau, Ben Z Yuan, Ayesha Bajwa, Michael Specter, and Lalana Kagal. Explaining explanations: An overview of interpretability of machine learning. In *2018 IEEE 5th International Conference on data science and advanced analytics (DSAA)*, pages 80–89. IEEE, 2018.

[39] Ian J. Goodfellow, Patrick D. McDaniel, and Nicolas Papernot. Making machine learning robust against adversarial inputs. *Commun. ACM*, 61(7):56–66, 2018.

[40] Raghuraman Gopalan, Ruonan Li, Vishal M Patel, and Rama Chellappa. Domain adaptation for visual recognition. *Foundations and Trends® in Computer Graphics and Vision*, 8(4):285–378, 2015.

[41] Pascale Gourdeau, Varun Kanade, Marta Kwiatkowska, and James Worrell. On the hardness of robust classification. In *Advances in Neural Information Processing Systems 32, NeurIPS*, pages 7444–7453, 2019.

[42] Yash Goyal, Ziyan Wu, Jan Ernst, Dhruv Batra, Devi Parikh, and Stefan Lee. Counterfactual visual explanations. *arXiv preprint arXiv:1904.07451*, 2019.

[43] Kathrin Grosse, Nicolas Papernot, Praveen Manoharan, Michael Backes, and Patrick McDaniel. Adversarial perturbations against deep neural networks for malware classification, 2016.

[44] Riccardo Guidotti, Anna Monreale, Salvatore Ruggieri, Dino Pedreschi, Franco Turini, and Fosca Giannotti. Local rule-based explanations of black box decision systems. *CoRR*, abs/1805.10820, 2018.

[45] Riccardo Guidotti, Anna Monreale, Salvatore Ruggieri, Franco Turini, Fosca Giannotti, and Dino Pedreschi. A survey of methods for explaining black box models. *ACM computing surveys (CSUR)*, 51(5):1–42, 2018.

[46] Congjie He, Meng Ma, and Ping Wang. Extract interpretability-accuracy balanced rules from artificial neural networks: A review. *Neurocomputing*, 387:346–358, 2020.

[47] Jiayuan Huang, Arthur Gretton, Karsten Borgwardt, Bernhard Schölkopf, and Alex Smola. Correcting sample selection bias by unlabeled data. *Advances in neural information processing systems*, 19:601–608, 2006.

[48] Mahbub Hussain, Jordan J Bird, and Diego R Faria. A study on cnn transfer

learning for image classification. In *UK Workshop on Computational Intelligence*, pages 191–202. Springer, 2018.

[49] Joel Janai, Fatma Güney, Aseem Behl, Andreas Geiger, et al. Computer vision for autonomous vehicles: Problems, datasets and state of the art. *Foundations and Trends® in Computer Graphics and Vision*, 12(1–3):1–308, 2020.

[50] Ulf Johansson, Cecilia Sönströd, Ulf Norinder, and Henrik Boström. Trade-off between accuracy and interpretability for predictive in silico modeling. *Future medicinal chemistry*, 3:647–63, 04 2011.

[51] Toshihiro Kamishima, Shotaro Akaho, Hideki Asoh, and Jun Sakuma. Fairness-aware classifier with prejudice remover regularizer. In *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*, pages 35–50. Springer, 2012.

[52] Wahab Khan, Ali Daud, Jamal A Nasir, and Tehmina Amjad. A survey on the state-of-the-art machine learning models in the context of nlp. *Kuwait journal of Science*, 43(4), 2016.

[53] Marc Khoury and Dylan Hadfield-Menell. Adversarial training with voronoi constraints, 2019.

[54] D. Kim, W. Lim, M. Hong, and H. Kim. The structure of deep neural network for interpretable transfer learning. In *2019 IEEE International Conference on Big Data and Smart Computing (BigComp)*, pages 1–4, Feb 2019.

[55] Samory Kpotufe. k-nn regression adapts to local intrinsic dimension. In *Advances in Neural Information Processing Systems, NIPS*, pages 729–737, 2011.

[56] Jitin Krishnan, Hemant Purohit, and Huzefa Rangwala. Unsupervised and interpretable domain adaptation to rapidly filter tweets for emergency services, 2020.

[57] Zachary C Lipton. The mythos of model interpretability. *Queue*, 16(3):31–57, 2018.

[58] P. J. G. Lisboa. Interpretability in machine learning – principles and practice. In Francesco Masulli, Gabriella Pasi, and Ronald Yager, editors, *Fuzzy Logic and Applications*, pages 15–21, Cham, 2013. Springer International Publishing.

[59] Zhenyu Liu and Huanhuan Chen. A predictive performance comparison of machine learning models for judicial cases. In *2017 IEEE Symposium Series on Computational Intelligence (SSCI)*, pages 1–6. IEEE, 2017.

[60] Jialin Lu and Martin Ester. An active approach for model interpretation. *ArXiv*, abs/1910.12207, 2019.

[61] Scott M Lundberg and Su-In Lee. A unified approach to interpreting model predictions. In I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, editors, *Advances in Neural Information Processing Systems 30*, pages 4765–4774. Curran Associates, Inc., 2017.

[62] Yi Luo, Huan-Hsin Tseng, Sunan Cui, Lise Wei, Randall K Ten Haken, and Issam El Naqa. Balancing accuracy and interpretability of machine learning approaches for radiation treatment outcomes modeling. *BJR| Open*, 1(1):20190021, 2019.

[63] Zhong Meng, Jinyu Li, Yong Zhao, and Yifan Gong. Conditional teacher-student learning. In *ICASSP 2019-2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 6445–6449. IEEE, 2019.

[64] Ali Mirzaei, Vahid Pourahmadi, Mehran Soltani, and Hamid Sheikhzadeh. Deep feature selection using a teacher-student network. *Neurocomputing*, 383:396–408, 2020.

[65] Christoph Molnar. *Interpretable Machine Learning*. 2019. `https://christophm.github.io/interpretable-ml-book/`.

[66] Omar Montasser, Surbhi Goel, Ilias Diakonikolas, and Nathan Srebro. Efficiently learning adversarially robust halfspaces with noise. *arXiv preprint arXiv:2005.07652*, 2020.

[67] Omar Montasser, Steve Hanneke, and Nathan Srebro. VC classes are adversarially robustly learnable, but only improperly. In *Conference on Learning Theory, COLT*, pages 2512–2530, 2019.

[68] Jose G Moreno-Torres, Troy Raeder, RocíO Alaiz-RodríGuez, Nitesh V Chawla, and Francisco Herrera. A unifying view on dataset shift in classification. *Pattern recognition*, 45(1):521–530, 2012.

[69] Sinno Jialin Pan and Qiang Yang. A survey on transfer learning. *IEEE Transactions on knowledge and data engineering*, 22(10):1345–1359, 2009.

[70] Vishal M Patel, Raghuraman Gopalan, Ruonan Li, and Rama Chellappa. Visual domain adaptation: A survey of recent advances. *IEEE signal processing magazine*, 32(3):53–69, 2015.

[71] Gabriel Peyré, Marco Cuturi, et al. Computational optimal transport: With applications to data science. *Foundations and Trends® in Machine Learning*, 11(5-6):355–607, 2019.

[72] Nikaash Puri, Piyush Gupta, Pratiksha Agarwal, Sukriti Verma, and Balaji Krishnamurthy. MAGIX: model agnostic globally interpretable explanations. *CoRR*, abs/1706.07160, 2017.

[73] Nikaash Puri, Piyush Gupta, Pratiksha Agarwal, Sukriti Verma, and Balaji Krishnamurthy. Magix: Model agnostic globally interpretable explanations, 2018.

[74] J. R. Quinlan. Induction of decision trees. *Machine Learning*, 1(1):81–106, Mar 1986.

[75] J. Ross Quinlan. Induction of decision trees. *Machine learning*, 1(1):81–106, 1986.

[76] Joaquin Quionero-Candela, Masashi Sugiyama, Anton Schwaighofer, and Neil D Lawrence. *Dataset shift in machine learning*. The MIT Press, 2009.

[77] Joaquin Quionero-Candela, Masashi Sugiyama, Anton Schwaighofer, and Neil D. Lawrence. *Dataset Shift in Machine Learning*. The MIT Press, 2009.

[78] Alvin Rajkomar, Jeffrey Dean, and Isaac Kohane. Machine learning in medicine. *New England Journal of Medicine*, 380(14):1347–1358, 2019.

[79] Ramya Ramakrishnan and Julie Shah. Towards interpretable explanations for transfer learning in sequential tasks. 03 2016.

[80] Ievgen Redko, Emilie Morvant, Amaury Habrard, Marc Sebban, and Younès Bennani. A survey on domain adaptation theory. *arXiv preprint arXiv:2004.11829*, 2020.

[81] Kui Ren, Tianhang Zheng, Zhan Qin, and Xue Liu. Adversarial attacks and defenses in deep learning. *Engineering*, 6(3):346 – 360, 2020.

[82] Marco Tulio Ribeiro, Sameer Singh, and Carlos Guestrin. Model-agnostic interpretability of machine learning. *arXiv preprint arXiv:1606.05386*, 2016.

[83] Marco Túlio Ribeiro, Sameer Singh, and Carlos Guestrin. "why should I trust you?": Explaining the predictions of any classifier. *CoRR*, abs/1602.04938, 2016.

[84] Marco Tulio Ribeiro, Sameer Singh, and Carlos Guestrin. Anchors: High-precision model-agnostic explanations. In *AAAI*, 2018.

[85] Ariella Richardson and Avi Rosenfeld. A survey of interpretability and explainability in human-agent systems. In *XAI Workshop on Explainable Artificial Intelligence*, pages 137–143, 2018.

[86] Laura Rieger and Lars Kai Hansen. A simple defense against adversarial attacks on heatmap explanations, 2020.

[87] Philippe Rigollet. Generalization error bounds in semi-supervised classification under the cluster assumption. *J. Mach. Learn. Res.*, 8:1369–1392, 2007.

[88] Sebastian Ruder. *Neural transfer learning for natural language processing.* PhD thesis, NUI Galway, 2019.

[89] Hadi Salman, Jerry Li, Ilya P. Razenshteyn, Pengchuan Zhang, Huan Zhang, Sébastien Bubeck, and Greg Yang. Provably robust deep learning via adversarially trained smoothed classifiers. In *Advances in Neural Information Processing Systems 32, NeurIPS*, pages 11289–11300, 2019.

[90] Ludwig Schmidt, Shibani Santurkar, Dimitris Tsipras, Kunal Talwar, and Aleksander Madry. Adversarially robust generalization requires more data. In *Advances in Neural Information Processing Systems, NeurIPS*, pages 5014–5026, 2018.

[91] N. Segev and R. El-Yaniv. Transfer learning using decision forests. 02 2016.

[92] N. Segev, M. Harel, S. Mannor, K. Crammer, and R. El-Yaniv. Learn on source, refine on target: A model transfer learning framework with random forests. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 39(9):1811–1824, Sep. 2017.

[93] Shai Shalev-Shwartz and Shai Ben-David. *Understanding Machine Learning: From Theory to Algorithms.* Cambridge University Press, 2014.

[94] Daniel Smilkov, Nikhil Thorat, Been Kim, Fernanda B. Viégas, and Martin Wattenberg. Smoothgrad: removing noise by adding noise. *CoRR*, abs/1706.03825, 2017.

[95] Hunsoo Song, Anjin Chang, Junho Yeom, Jinha Jung, and Yongil Kim. Domain adaptation framework for deep learning based change detection in remotely sensed data under prior probability shift.

[96] Mateusz Staniak and Przemyslaw Biecek. Explanations of model predictions with live and breakdown packages. *R J.*, 10:395, 2018.

[97] Ingo Steinwart and Clint Scovel. Fast rates for support vector machines using gaussian kernels. *The Annals of Statistics*, 35(2):575–607, 2007.

[98] Shiliang Sun, Honglei Shi, and Yuanbin Wu. A survey of multi-source domain adaptation. *Information Fusion*, 24:84–92, 2015.

[99] Christian Szegedy, Wojciech Zaremba, Ilya Sutskever, Joan Bruna, Dumitru Erhan, Ian J. Goodfellow, and Rob Fergus. Intriguing properties of neural networks. In *2nd International Conference on Learning Representations, ICLR*, 2014.

[100] Jayaraman J. Thiagarajan, Bhavya Kailkhura, Prasanna Sattigeri, and Karthikeyan Natesan Ramamurthy. Treeview: Peeking into deep neural networks via feature-space partitioning, 2016.

[101] Erico Tjoa and Cuntai Guan. A survey on explainable artificial intelligence (xai): Toward medical xai. *IEEE Transactions on Neural Networks and Learning Systems*, 2020.

[102] Lisa Torrey and Jude Shavlik. Transfer learning. In *Handbook of research on machine learning applications and trends: algorithms, methods, and techniques*, pages 242–264. IGI global, 2010.

[103] Dimitris Tsipras, Shibani Santurkar, Logan Engstrom, Alexander Turner, and Aleksander Madry. Robustness may be at odds with accuracy, 2019.

[104] Ruth Urner, Shai Shalev-Shwartz, and Shai Ben-David. Access to unlabeled data can speed up prediction time. In *Proceedings of the 28th International Conference on Machine Learning, ICML*, pages 641–648, 2011.

[105] Vanya Van Belle, Ben Van Calster, Sabine Van Huffel, Johan AK Suykens, and Paulo Lisboa. Explaining support vector machines: a color based nomogram. *PloS one*, 11(10):e0164568, 2016.

[106] J. w. Lee and C. Giraud-Carrier. Transfer learning in decision trees. In *2007 International Joint Conference on Neural Networks*, pages 726–731, Aug 2007.

[107] Mei Wang and Weihong Deng. Deep visual domain adaptation: A survey. *Neurocomputing*, 312:135–153, 2018.

[108] Yizhen Wang, Somesh Jha, and Kamalika Chaudhuri. Analyzing the robustness of nearest neighbors to adversarial examples. In *Proceedings of the 35th International Conference on Machine Learning, ICML*, pages 5120–5129, 2018.

[109] Karl Weiss, Taghi M Khoshgoftaar, and DingDing Wang. A survey of transfer learning. *Journal of Big data*, 3(1):9, 2016.

[110] Huanrui Yang, Jingchi Zhang, Hsin-Pai Cheng, Wenhan Wang, Yiran Chen, and Hai Li. Bamboo: Ball-shape data augmentation against adversarial attacks from all directions. In *Workshop on Artificial Intelligence Safety 2019 co-located with the Thirty-Third AAAI Conference on Artificial Intelligence 2019 (AAAI-19), Honolulu, Hawaii, January 27, 2019*, 2019.

[111] Yao-Yuan Yang, Cyrus Rashtchian, Yizhen Wang, and Kamalika Chaudhuri. Adversarial examples for non-parametric methods: Attacks, defenses and large sample limits. *CoRR*, abs/1906.03310, 2019.

[112] Yao-Yuan Yang, Cyrus Rashtchian, Yizhen Wang, and Kamalika Chaudhuri. Robustness for non-parametric classification: A generic attack and defense. In Silvia Chiappa and Roberto Calandra, editors, *Proceedings of the Twenty Third International Conference on Artificial Intelligence and Statistics*, volume 108 of *Proceedings of Machine Learning Research*, pages 941–951, Online, 26–28 Aug 2020. PMLR.

[113] Yao-Yuan Yang, Cyrus Rashtchian, Hongyang Zhang, Ruslan Salakhutdinov, and Kamalika Chaudhuri. A closer look at accuracy vs. robustness, 2020.

[114] Dong Yin, Kannan Ramchandran, and Peter L. Bartlett. Rademacher complexity for adversarially robust generalization. In *Proceedings of the 36th International Conference on Machine Learning,ICML*, pages 7085–7094, 2019.

[115] Hang Yu, Aishan Liu, Xianglong Liu, Gengchao Li, Ping Luo, Ran Cheng, Jichen Yang, and Chongzhi Zhang. Pda: Progressive data augmentation for general robustness of deep neural networks, 2020.

[116] Haichao Zhang and Jianyu Wang. Defense against adversarial attacks using feature scattering-based adversarial training. In *Advances in Neural Information Processing Systems*, pages 1831–1841, 2019.

[117] Hongyang Zhang, Yaodong Yu, Jiantao Jiao, Eric Xing, Laurent El Ghaoui, and Michael Jordan. Theoretically principled trade-off between robustness and accuracy. In Kamalika Chaudhuri and Ruslan Salakhutdinov, editors, *Proceedings of the 36th International Conference on Machine Learning*, volume 97 of *Proceedings of Machine Learning Research*, pages 7472–7482, Long Beach, California, USA, 09–15 Jun 2019. PMLR.

[118] Zizhao Zhang, Yuanpu Xie, Fuyong Xing, Mason McGough, and Lin Yang. Mdnet:

A semantically and visually interpretable medical image diagnosis network. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 6428–6436, 2017.

[119] Han Zhao, Shanghang Zhang, Guanhang Wu, José MF Moura, Joao P Costeira, and Geoffrey J Gordon. Adversarial multiple source domain adaptation. *Advances in neural information processing systems*, 31:8559–8570, 2018.

[120] Haoxi Zhong, Yuzhong Wang, Cunchao Tu, Tianyang Zhang, Zhiyuan Liu, and Maosong Sun. Iteratively questioning and answering for interpretable legal judgment prediction. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pages 1250–1257, 2020.

[121] Fuzhen Zhuang, Zhiyuan Qi, Keyu Duan, Dongbo Xi, Yongchun Zhu, Hengshu Zhu, Hui Xiong, and Qing He. A comprehensive survey on transfer learning. *arXiv preprint arXiv:1911.02685*, 2019.

[122] Matthieu Zimmer, Paolo Viappiani, and Paul Weng. Teacher-student framework: a reinforcement learning approach. 2014.