Machine Learning Approaches to Challenging Problems: Interpretable Imbalanced Classification, Interpretable Density Estimation, and Causal Inference

by

Siong Thye Goh

Submitted to the Sloan School of Management
in partial fulfillment of the requirements for the degree of

DOCTOR OF PHILOSOPHY IN OPERATIONS RESEARCH

at the

MASSACHUSETTS INSTITUTE OF TECHNOLOGY

June 2018

Author .....
Sloan School of Management
April 26, 2018

Certified by .....
Cynthia Rudin
Associate Professor
Thesis Supervisor

Certified by .....
Roy Welsch
Professor
Thesis Supervisor

Accepted by .....
Patrick Jaillet
CoDirector, Operations Research Center

Machine Learning Approaches to Challenging Problems: Interpretable Imbalanced Classification, Interpretable Density Estimation, and Causal Inference

by

Siong Thye Goh

Submitted to the Sloan School of Management
on April 26, 2018, in partial fulfillment of the
requirements for the degree of
DOCTOR OF PHILOSOPHY IN OPERATIONS RESEARCH

Abstract

In this thesis, I address three challenging machine-learning problems.

The first problem that we address is the imbalanced data problem. We propose two algorithms to handle highly imbalanced classification problems. The first algorithm uses mixed integer programming to optimize a weighted balance between positive and negative class accuracies. The second method uses an approximation in order to assist with scalability. Specifically, it follows a characterize-then-discriminate approach. The positive class is first characterized by boxes, and then each box boundary becomes a separate discriminative classifier. This method is computationally advantageous because it can be easily parallelized, and considers only the relevant regions of the feature space.

The second problem is a density estimation problem for categorical data sets. We present tree- and list- structured density estimation methods for binary/categorical data. We present three generative models, where the first one allows the user to specify the number of desired leaves in the tree within a Bayesian prior. The second model allows the user to specify the desired number of branches within the prior. The third model returns lists (rather than trees) and allows the user to specify the desired number of rules and the length of rules within the prior.

Finally, we present a new machine learning approach to estimate personalized treatment effects in the classical potential outcomes framework with binary outcomes. Strictly, both treatment and control outcomes must be measured for each unit in order to perform supervised learning. However, in practice, only one outcome can be observed per unit. To overcome the problem that both treatment and control outcomes for the same unit are required for supervised learning, we propose surrogate loss functions that incorporate both treatment and control data. The new surrogates yield tighter bounds than the sum of the losses for the treatment and control groups. A specific choice of loss function, namely a type of hinge loss, yields a minimax support vector machine formulation. The resulting optimization problem requires the

solution to only a single convex optimization problem, incorporating both treatment and control units, and it enables the kernel trick to be used to handle nonlinear (also non-parametric) estimation.

Thesis Supervisor: Cynthia Rudin
Title: Associate Professor

Thesis Supervisor: Roy Welsch
Title: Professor

Acknowledgments

First, I would like to express my deepest gratitude to my advisor Professor Cynthia Rudin for everything that she has done for me. During my PhD studies, she has been a great mentor, sharing research ideas, providing guidance, and providing a lot of encouragement. I have benefited from her immense knowledge and patience. This PhD would not have been possible without her.

I would also like to thank my thesis committee, Professor Roy Welsch and Professor Peter Szolovits, for their valuable feedback and insightful suggestions. Furthermore, they have read through the whole thesis in great detail and helped me improve the thesis.

My internship experience during my PhD endeavor was fun thanks to friends and colleagues whom I met there. Thank you to Amit Chakraborty and Yuan Chao for the awesome guidance and for even using the methods I developed in my internship work.

I am thankful to my friends whom I met at MIT. They have brought color to my life at MIT. I have been extremely lucky to be surrounded by peers who are individually brilliant and also supportive of each other. I would like to thank MITMASA and MITSSS for being a family to me. There are also various student groups dedicated to the improvement of student life such as SaveTFP, Happy Club, Knots and Cheesecake, and HACK club, that made my life more enjoyable. I am grateful for friends who have been with me to support me when I really needed the help: special thanks to Chong Yang, Shujing, Jerry, Hai Wang, Dax, Gladia, Zhengjie, Guang Hao, Mila, Rosary, Jingzhi, Jian Feng, Jun Yong, Neelkanth, Yongwhan, and Avishek. To Stefanie Sun Yanzi, Dessert Zhang Anpu, Penny Dai, Lala, and Hebe Tian Fuzhen, your music has been great. To Jane Dunphy, Thalia, and Amanda, thank you for helping me improve my command of English.

My family and friends back in South East Asia have also been very supportive of me. Charmaine, Beatrice, Wei Pin, Huey Chyi, Sze Chong, and Onanlas, thanks for being connected with me all the time despite the distance. I would also like to

take the opportunity to thank Professors whom I met back in Singapore, especially Delin Chu, Karthik Natrajan, Kim Chuan Toh, Melvyn Sim, Defeng Sun, and Victor Tan, who built a strong mathematical foundation for me. I am grateful to the Kuok Foundation for giving me the opportunity to access higher education back then.

To my beloved hometown, thank you for providing help to my family when we really needed it.

To my beloved aunt, Mrs Sim Swee Yeong, your love and kindness have made me a better person. You have touched the lives of so many people including mine.

To my beloved parents, thank you for your unconditional love.

Contents

List of Figures

List of Tables

Chapter 1

Introduction and Contribution

Depending on the information that we have access to, machine learning problems can be divided into several classes. A common class of problems is known as supervised learning problems, where the outcome is known for the training data set. Various algorithms such as support vector machines (SVM), random forests, and deep learning models are suitable methods for tackling supervised learning models. However, even under this setting, there are some challenging circumstances such as large dataset sizes and missing data. Furthermore, the model may be incomprehensible, which is undesirable for decision makers. A more challenging setting is the circumstance in which the labels are not even given to us. Popular examples of such problems include clustering and density estimation. Another possible challenging scenario is an experimental setting in which we may have a reading for the outcome for each sample data point but do not have the most relevant reading for every single data set, which is the case for all unsupervised problems. Inference is needed to recover the desired information. This thesis addresses three such problems, each of a different nature.

The first problem that we address is the interpretable imbalanced data problem. For example, in an email repository, we might be interested in creating a system to filter spam emails. However, there may be many more non-spam emails than spam emails, so the data set is imbalanced. Typically, when a data set is imbalanced, classical algorithms, which optimize for accuracy, tend to focus on the class which

most data points come from and neglect the other classes. This results in many false positives or false negatives. However, one problem with this is that the smaller class might be the class of interest in which we wish to identify patterns. For instance, in the example of spam emails, the class of spam emails may be too small for classical algorithms to learn the features which are most indicative of a spam email. Typical approaches to handling such problems include oversampling or undersampling coupled with a traditional machine learning algorithm. However, it is preferable if a machine learning algorithm can tell exactly why a data point belongs to a class — that is, we want the rule of classification to be interpretable. A decision tree might serve the purpose; however, most decision tree implementations split the feature space along the feature axis greedily, and the objective function of the procedure is not clear. In Chapter 2 of this thesis, we address this problem by using an optimization approach. We define a mixed integer programming approach to place axis-parallel rectangles to characterize the minority class. This is known as the exact boxes method. To make the algorithm more efficient, we propose another algorithm, fast boxes, which is a method that can be used to warm-start the algorithm. The minority classes are first characterized using a clustering algorithm. Subsequently, for each cluster, we adjust its boundary separately. Each cluster's boundary only depends on the points around the boxes. Our Simulation results show that the exact-boxes algorithm outperforms many standard algorithms.

The second problem that we address is the density estimation problem for categorical data sets. High-dimensional histograms reveal which configurations of a dataset are more prominent or more interesting. However, as the dimensionality of the dataset grows, the number of bins increases rapidly, making the histogram less interpretable. It is harder to describe a histogram when the number of bins is huge. To overcome this problem, we use tree and list structures to introduce hierarchical structure to the histogram and also regularize the complexity of the trees to make the result that we obtain more interpretable. Reducing the complexity of the trees also makes for better generalization. We propose three different procedures. The first procedure regularizes the number of leaves of a tree while the second procedure regularizes the

number of branches of a tree. The last procedure restricts the tree structure to be a list, hence improving the efficiency of the algorithm. For the list structure, the depth and complexity of each rule is regularized. Users are allowed to choose the parameters, and hence they are able to use their domain knowledge to control the complexity of the tree structures. The regularization is implemented by using a Bayesian prior. We apply our density estimation algorithm to the Cambridge Police crime data set from 1997 to 2012 to understand the common types of modus operandi (M.O.) of housebreakers.

The third problem that we try to mitigate is the fundamental problem of causal inference. For example, in a clinical trial setting, patients are assigned to either the treatment group or the control group. However, we are not able to observe the alternative outcome which would occur if someone assigned to the treatment group were to be assigned to the control group instead, and vice versa. In some special circumstances, we might be able to observe the alternative outcome under a controlled environment when the effect of the treatment has worn off. In a clinical trial, we try to match similar patients during the design of the experiment. However, this might not be tractable as experiments might be expensive to conduct; furthermore, it is sometimes the case that we are given a dataset to analyze without being able to participate in the design of the experiment. In such settings, the most popular approach would be to perform matching. Another strategy is to apply regression methods to the treatment and control groups to compute the treatment effect. In theory, observing the treatment effect for any sample is impossible. We focus on the causal inference problem where the effect takes binary values. We first define a loss function which penalizes having the wrong predicted sign for the treatment effect. However, this objective function is not computable due to the fundamental problem of causal inference. We instead propose to use a surrogate function which is computable. This approach of using a surrogate function is similar to SVM. We present functions which can be converted to such a surrogate function. In particular, the hinge loss function is one such function, and it enables us to introduce the kernel trick to our framework. We then formulate this problem as a quadratic programming problem. By

considering its dual problem, we are able to cast the optimization problem in a form similar to that of support vector machines. We derive the generalization bounds and apply our algorithm to study the effectiveness of a social program known as "Breaking the Cycle", which studies how introducing counseling helps in reducing drug abuse and crime rates in Birmingham.

Chapter 2

Box Drawings for Learning with Imbalanced Data

2.1 Introduction

Our interest is in deriving interpretable predictive classification models for use with imbalanced data. Data classification problems having imbalanced (also called "unbalanced") class distributions appear in many domains, ranging from mechanical failure detection or fault detection, to fraud detection, to text and image classification, to medical disease prediction or diagnosis. Imbalanced data cause typical machine learning methods to produce trivial results, that is, classifiers that only predict the majority class. One cannot optimize vanilla classification accuracy and use standard classification methods when working with imbalanced data. This is explained nicely by Chawla, Japkowicz, and Kolcz [18] who write: "*The class imbalance problem is pervasive and ubiquitous, causing trouble to a large segment of the data mining community.*"

In order for the models we derive to be interpretable to human experts, our classifiers are formed as a union of axis parallel rectangles around the positive (minority class) examples, and we call such classifiers *box drawing classifiers*. These are "disjunctions of conjunctions" where each conjunction is a box. An example of a box drawing classifier we created is in Figure 2-1, exemplifying our goal to classify the

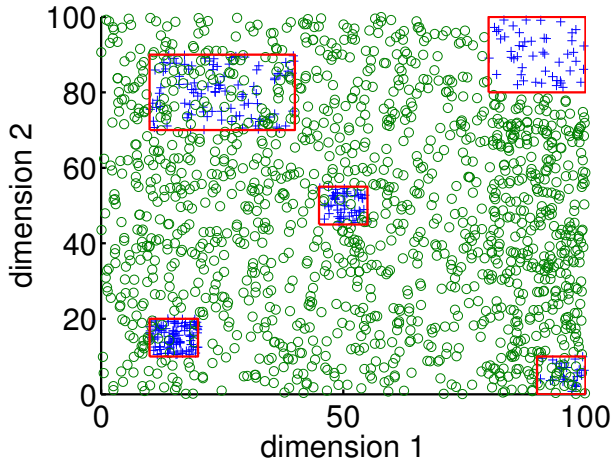


Figure 2-1: Example of box drawing classifier.

positive examples correctly even if they are scattered within a sea of negative examples. Our classifiers are regularized in several ways, to prefer fewer boxes and larger boxes. We take two polar approaches to creating box drawing classifiers, where the first is an exact method, based on mixed integer programming (MIP). This method, called *Exact Boxes* can be used for small to medium sized datasets, and provides a gold standard to compare with. If we are able to make substantial approximations and still obtain performance close to that of the gold standard, our approximations would be justified. Our second method, *Fast Boxes* makes such an approximation.

Fast boxes takes the approach of *characterize then discriminate*, where we first characterize the positive (minority) class alone, and then bring in the negative examples to form decision boundaries around each clusters of positives. This approach has significant computational advantages, in that using just the minority class in the first step requires a small fraction of the data, assuming a high imbalance ratio. Also by creating decision boundaries locally in the second step, the number of examples involved in each classifier is smaller; further, creating each classifier separately allows computations to be made parallel, though since the computation for each decision boundary is analytical, that may not be necessary for many datasets. The computation is analytical because there is a closed form solution for the placement of the decision boundary. Thus, the discriminate step becomes many parallel local analyti-

cal calculations. This is much simpler and more scalable than, for instance, a decision tree that chooses splits greedily and fails to scale with dimension and large numbers of observations.

We make several experimental observations, namely that: box drawing classifiers become more useful as data imbalance increases; the approximate method performs at the top level of its competitors, despite the fact that it is restricted to producing interpretable results; and performance can be improved on the same datasets by using the mixed integer programming method.

After related work just below, we describe the advantages of our approach in Section 2. In Section 3, we introduce our two algorithms. Experimental results will be presented in Section 4. Section 4 provides a vignette to show how box drawing models can be interpretable. In Section 5, theoretical generalization bounds will be presented for box drawing classifiers. Section 6 discusses possible approaches to make the MIP formulation more scalable.

2.2 Related Works

Overviews of work on handling class imbalance problems include those of He and Garcia [35], Chawla, Japkowiz and Kolcz [18] and Qi [68]. Many works discuss problems caused by class imbalance [93, 65]. There are many avenues of research that are not directly related to the goal of interpretable imbalanced classification, specifically kernel and active learning methods [70, 96], and work on sampling [1, 17] that includes undersampling, oversampling, and data generation, which can be used in conjunction with methods like the ones introduced here. We use a cost-sensitive learning approach in our methods, similar to Liu and Zhou [54] and McCarthy et al. [57]. We note that many papers on imbalanced data do not experimentally compare their work with the cost-sensitive versions of decision tree methods. We choose to compare with other cost-sensitive versions of decision trees as our method is a cost-sensitive method.

There is some evidence that more complex approaches that layer different learning methods seem to be helpful for learning [70, 96], though the results would not be

interpretable in that case. This, however, is in contrast with other views (e.g., [39]) that for most common datasets, simple rules exist and we should explore them.

The works most similar to ours are that of the Patient Rule Induction Method (PRIM) [29] and decision tree methods for imbalanced classification (e.g., [43]), as they partition the input space like our work. Approaches that partition space tend to recover simple decision rules that are easier for people to understand. Decision tree methods are composed using greedy splitting criteria, unlike our methods. PRIM is also a greedy method that iteratively peels off parts of the input space, though unfortunately we found it to be extremely slow — as described by Sniadecki [88], "PRIM is eloquently coined as a patient method due to the slow, stepwise mechanism by which it processes the data." Neither our Exact Boxes nor Fast Boxes methods are greedy methods, though Fast Boxes makes a different type of approximation, which is to characterize before discriminating. As discussed by Raskutti [70], one-class learning can be useful for highly imbalanced datasets — our characterization step is a one-class learning approach.

2.3 New Algorithms

We start with the mixed-integer programming formulation, which acts as our gold standard for creating box drawing classifiers when solved to optimality.

2.3.1 Exact Boxes

For box drawing classifiers, a minority class (positive) example is correctly classified only if it resides within at least one box. A majority class (negative) example is correctly classified if it does not reside in any box. We are given training examples $\{(\mathbf{x}_i, y_i)\}_{i=1}^m, \mathbf{x}_i \in \mathcal{R}^n, y_i \in \{-1, +1\}$. We introduce some notation in Table 2.1 that we will use throughout this subsection. We use this notation from here on.

The *Exact Boxes* method solves the following, where the hypothesis space $\mathcal{F}$ is the set of box drawings (unions of axis parallel rectangles), where $f \in \mathcal{F}$ has

Notation	Definitions
K	Number of parallel axes boxes
m	Number of examples
n	Number of features
i	Index for examples
j	Index for features
x_{ij}	j-th feature of example i
k	Index for box
l_{jk}	Lower boundary of feature j for box k
u_{jk}	Upper boundary of feature j for box k
v	Margin for decision boundary
$\widetilde{l}_{ijk}$	$\widetilde{l}_{ijk} = 1$ if $x_{ij} > l_{jk} + v$ and 0 otherwise
$\widetilde{u}_{ijk}$	$\widetilde{u}_{ijk} = 1$ if $x_{ij} < u_{jk} - v$ and 0 otherwise
w_{ik}	$w_{ik} = 1$ if example i is in box k and 0 otherwise
z_i	$z_i = 1$ if it is classified correctly.
S_+	Index set of example of minority class
S_-	Index set of example of majority class
c_e	A regularizer to encourage expansion of box
c_I	Weight for majority class, $c_I < 1$

Table 2.1: Notation for Box Drawings with Mixed Integer Programming

$f : \mathcal{R}^n \to \{-1, 1\}$.

$$\max_{f \in \mathcal{F}} \sum_{i:y_i=1} \mathbf{1}_{[f(\mathbf{x}_i)=1]} + C_I \sum_{i:y_i=-1} \mathbf{1}_{[f(\mathbf{x}_i)=-1]} - C_E(\#\text{of boxes of } f).$$

The objective is a weighted accuracy of positives and negatives, regularized by the number of boxes. This way, the number of boxes is not fixed, and a smaller number of clusters is preferred (analogous to nonparametric Bayesian models where the number of clusters is not fixed). Our gold standard will be the minimizer of this objective. We now derive the MIP that computes this minimizer.

If $i \in S_+$, the definitions of $\widetilde{l}_{ijk}$, $\widetilde{u}_{ijk}$, w_{ik}, and z_i give rise to the following constraints:

$$l_{jk} + v < x_{ij} \text{ iff } \widetilde{l}_{ijk} = 1 \tag{2.1}$$

$$u_{jk} - v > x_{ij} \text{ iff } \widetilde{u}_{ijk} = 1, \tag{2.2}$$

which say that x_{ij} need to be at least margin v away from the lower (resp. upper)

boundary of the box in order for $\widetilde{l}_{ijk} = 1$ (resp. $\widetilde{u}_{ijk} = 1$). Further, our definitions give rise also to

$$\sum_{j=1}^{n} \widetilde{u}_{ijk} + \widetilde{l}_{ijk} > 2n - 1 \text{ iff } w_{ik} = 1, \tag{2.3}$$

which says that for example i to be in box k, all of the $\widetilde{u}_{ijk}$ and $\widetilde{l}_{ijk}$ are 1 for box k. We also have, still for $i \in S_+$, that the example must be in one of the boxes in order to be classified correctly, that is:

$$\sum_{k=1}^{K} w_{ik} > 0 \text{ iff } z_i = 1. \tag{2.4}$$

Continuing this same type of reasoning for $i \in S_-$, the definitions of $\widetilde{l}_{ijk}$,$\widetilde{u}_{ijk}$, w_{ik}, and z_i give rise to the following constraints:

$$\begin{aligned} l_{jk} - v > x_{ij} \quad &\text{iff} \quad \widetilde{l}_{ijk} = 1 \\ u_{jk} + v < x_{ij} \quad &\text{iff} \quad \widetilde{u}_{ijk} = 1 \\ \sum_{j=1}^{n} \widetilde{u}_{ijk} + \widetilde{l}_{ijk} > 0 \quad &\text{iff} \quad w_{ik} = 0 \\ \sum_{k=1}^{K} w_{ik} > 0 \quad &\text{iff} \quad z_i = 0. \end{aligned}$$

By setting M to be a large positive constant and setting ϵ to be a small positive number (to act as a strict inequality), we now have the following formulation:

$$\max_{l,\widetilde{l},u,\widetilde{u},w,z} \left[-c_e K + \sum_{i \in S_+} z_i + c_I \sum_{i \in S_-} z_i \right] \text{ subject to}$$

$$x_{ij} - l_{jk} - v \leq M\widetilde{l}_{ijk}, \forall i \in S_+, \forall j, k \tag{2.5}$$

$$M(\widetilde{l}_{ijk} - 1) + \epsilon \leq x_{ij} - l_{jk} - v, \forall i \in S_+, \forall j, k \tag{2.6}$$

$$u_{jk} - v - x_{ij} \leq M\widetilde{u}_{ijk}, \forall i \in S_+, \forall j, k \tag{2.7}$$

$$M(\widetilde{u}_{ijk} - 1) + \epsilon \leq u_{jk} - x_{ij} - v, \forall i \in S_+, \forall j, k \tag{2.8}$$

$$\sum_{j=1}^{n} \widetilde{l}_{ijk} + \sum_{j=1}^{n} \widetilde{u}_{ijk} - 2n + 1 \leq w_{ik}, \forall i \in S_+, \forall j, k \tag{2.9}$$

$$2nw_{ik} \leq \sum_{j=1}^{n} \widetilde{l}_{ijk} + \sum_{j=1}^{n} \widetilde{u}_{ijk}, \forall i \in S_+, \forall j, k \tag{2.10}$$

$$\sum_{k=1}^{K} w_{ik} \leq Kz_i, \forall i \in S_+, \forall k \tag{2.11}$$

$$z_i \leq \sum_{k=1}^{K} w_{ik}, \forall i \in S_+, \forall k \tag{2.12}$$

$$l_{jk} - v - x_{ij} \leq M\widetilde{l}_{ijk}, \forall i \in S_-, \forall j, k \tag{2.13}$$

$$M(\widetilde{l}_{ijk} - 1) + \epsilon \leq l_{jk} - v - x_{ij}, \forall i \in S_-, \forall j, k \tag{2.14}$$

$$x_{ij} - u_{jk} - v \leq M\widetilde{u}_{ijk}, \forall i \in S_-, \forall j, k \tag{2.15}$$

$$M(\widetilde{u}_{ijk} - 1) + \epsilon \leq x_{ij} - u_{jk} - v, \forall i \in S_-, \forall j, k \tag{2.16}$$

$$\sum_{j=1}^{n} \widetilde{l}_{ijk} + \sum_{j=1}^{n} \widetilde{u}_{ijk} - 2n + 1 \leq 2n(1 - w_{ik}), \quad \forall i \in S_-, \forall j, k \tag{2.17}$$

$$1 - w_{ik} \leq \sum_{j=1}^{n} \widetilde{l}_{ijk} + \sum_{j=1}^{n} \widetilde{u}_{ijk}, \forall i \in S_-, \forall j, k \tag{2.18}$$

$$\sum_{k=1}^{K} w_{ik} \leq K(1 - z_i), \forall i \in S_-, \forall k \tag{2.19}$$

$$1 - z_i \leq \sum_{k=1}^{K} w_{ik}, \forall i \in S_-, \forall k \tag{2.20}$$

$$l_{jk} \leq u_{jk}, \forall j, k. \tag{2.21}$$

Here, (2.5) and (2.6) are derived from (2.1), (2.7) and (2.8) are derived from (2.2), (2.9) and (2.10) are derived from (2.3), (2.11) and (2.12) are derived from (2.4), equations (2.13)-(2.20) are derived analogously for S_-. The last constraint (2.21) is to make sure that the solution that we obtain is not degenerate, where the

lower boundary is above the upper boundary. In practice, M should be chosen as a fixed large number and ϵ should be chosen as a fixed small number based on the representation of numbers in the computing environment.

In total, there are $O(mnK)$ equations and $O(mnK)$ variables, though the full matrix of variables corresponding to the mixed integer programming formulation is sparse since most boxes operate only on a small subset of the data. This formulation can be solved efficiently for small to medium sized datasets using MIP software, producing a gold standard box drawing classifier for any specific number of boxes (determined by the regularization constant). The fact that Exact Boxes produces the best possible function in the class of box drawing classifiers permits us to evaluate the quality of Fast Boxes, which operates in an approximate way on a much larger scale.

2.3.2 Fast Boxes

Fast Boxes uses the approach of *characterize then discriminate*. In particular, we hypothesize that the data distribution is such that the positive examples cluster together relative to the negative examples. This implies that a reasonable classifier might first cluster the positive examples and then afterwards discriminate positives from negatives. The discrimination is done by drawing a high dimensional axis-parallel box around each cluster and then adjusting each boundary locally for maximum discriminatory power. If the cluster assumption about the class distributions is not correct, then Fast Boxes could have problems, though it does not seem to for most real imbalanced datasets we have found, as we show in the experiments. Fast Boxes has three main stages as follows.

---

1. Clustering stage: Cluster the minority class data into K clusters, where K is an adjustable parameter. The decision boundaries are initially set as tight boxes around each of the clusters of positive examples.

2. Dividing space stage: The input space of the data is partitioned to determine

which positive and negative examples will influence the placement of each decision boundary.

3. Boundary expansion stage: Each boundary is expanded by minimizing an exponential loss function. The solution for the decision boundary is analytical.

Details of each stage are provided below.

Clustering Stage

In the clustering stage, the minority class data are clustered into K clusters. Since this step involves only the minority class data, it can be performed efficiently, particularly if the data are highly imbalanced. Cross-validation or other techniques can be used to determine K. In our experiments, we used the basic k-means algorithm with Euclidean distance. Other clustering techniques or other distance metrics can be used.

After the minority class data are separated into small clusters, we construct the smallest enclosing parallel axes rectangle for each cluster. The smallest enclosing parallel axes rectangle can be computed by taking the minimum and maximum of the minority class data in each cluster and for each feature. Let $l_{s,j,k}$ and $u_{s,j,k}$ denote the lower boundary and upper boundary for the j-th dimension, for the k-th cluster. Here the subscript s is for "starting" boundary, and in the next part we will created a "revised" boundary which will be given subscript r. The "final" boundary will be given subscript f.

Dividing Space Stage

Define the set $X_{l,j,k}$ as follows:

$$X_{l,j,k} := \{x : x_j \leq l_{s,j,k}\} \cup \left\{ x : l_{s,j,k} \leq x_j \leq \frac{l_{s,j,k} + u_{s,j,k}}{2}, \right. \\ \left. l_{s,p,k} \leq x_p \leq u_{s,p,k}, p \neq j \right\}.$$

These are the data points that will be used to adjust the lower boundary of the j-th dimension of the k-th rectangle.

Similarly, we let

$$X_{u,j,k} := \{x : x_j \geq u_{s,j,k}\} \cup \left\{ x : \frac{l_{s,j,k} + u_{s,j,k}}{2} \leq x_j \leq u_{s,j,k}, \right.$$
$$\left. l_{s,p,k} \leq x_p \leq u_{s,p,k}, p \neq j \right\}.$$

These are the training examples that will be used to determine the upper boundary of the j-th dimension of the k-th rectangle.

Figure 2-2 illustrates the domain for $X_{u,j,k}$ to the right of the blue dashed line.

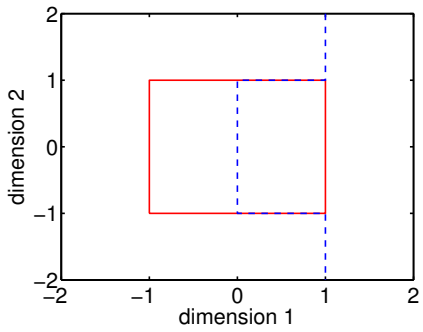


Figure 2-2: The examples used to determine the right vertical decision boundary are on the right side of the blue dotted line.

Note that this method is very parallelizable after the clustering stage. The dividing space stage computations can be done in parallel for each cluster, and for the boundary expansion stage discussed below, each boundary of each box can be determined in parallel.

Boundary Expansion Stage

In this stage we discriminate between positives and negatives by creating a 1-dimensional classifier for each boundary of each box. We use a regularized exponential loss. Specifically, for lower boundary j of box k, We minimize the following with respect to $l_{r,j,k}$ where $l_{r,j,k}$ refers to the lower boundary of the j-th dimension of k-th revised box

being determined by the loss function:

$$\begin{aligned}
&\sum_{x \in S_+^k \cap X_{l,j,k}} \exp[-(x_j - l_{r,j,k})] \\
&+ \quad c \sum_{x \in S_-^k \cap X_{l,j,k}} \exp\left[\left(x_j - l_{r,j,k}\right.\right. \\
&+ \quad \left.\left.\sum_{p \neq j} (\lfloor x_p - u_{s,p,k} \rfloor_+ + \lfloor l_{s,p,k} - x_p \rfloor_+)\right)\right] + \beta l_{r,j,k}.
\end{aligned}$$

where c is the weight for the majority class, $c < 1$, S_+^k is the set of positive examples in the k-th cluster, S_-^k is the set of examples not in the k-th cluster, β is a regularization parameter that tends to expand the box, and $\lfloor . \rfloor$ denotes $\max(., 0)$. For simplicity, we use the same parameter to control the expansion for all the clusters and all the features. Note that the term $\sum_{p \neq j} (\lfloor x_p - u_{s,p,k} \rfloor_+ + \lfloor l_{s,p,k} - x_p \rfloor_+)$ is designed to give less weight to the points that are not directly opposite the box edges (the points that are diagonally away from the corners of the box). To explain these terms, recall that the exponential loss in classification usually operates on the term $y_i f(x_i)$, where the value of $f(x_i)$ can be thought of as a distance to the decision boundary. In our case, for the lower decision boundary we use the perpendicular distance to the decision boundary $|x_j - l_{r,j,k}|$, and include the additional distance in every other dimension p for the diagonal points. For the upper diagonal points we include the distance to the upper boundary $u_{s,p,k}$, namely $x_p - u_{s,p,k}$, and analogously for the points on the lower diagonal we include distance $l_{s,p,k} - x_p$. We perform an analogous calculation for the upper boundary.

Note that we perform a standard normalization of all features to be between -1 and 1 before any computation begins, which also mitigates numerical issues when dealing with the (steep) exponential loss. Another mechanism we use for avoiding numerical problems is by multiplying each term in the objective by exp(1) and dividing each term by the same factor. We will construct the derivation of the lower boundary as

follows. We rewrite the objective to minimize:

$$R_+^{l,j,k} \exp\left(-l_{s,j,k} + 1 + l_{r,j,k}\right) + cR_-^{l,j,k} \exp\left(l_{s,j,k} - 1 - l_{r,j,k}\right) + \beta l_{r,j,k}, \tag{2.22}$$

where

$$R_+^{l,j,k} := \sum_{x \in S_+^k \cap X_{l,j,k}} \exp\left[-(x_j - l_{s,j,k} + 1)\right], \tag{2.23}$$

$$R_-^{l,j,k} := \sum_{x \in S_-^k \cap X_{l,j,k}} \exp\left[x_j - l_{s,j,k} + 1 + \sum_{p \neq j} (\lfloor x_p - u_{s,p,k} \rfloor_+ + \lfloor l_{s,p,k} - x_p \rfloor_+)\right]. \tag{2.24}$$

Because of the factors of 1 added and subtracted in the exponent, we ensure $R_+^{l,j,k}$ is at least $\exp(-1) > 0.3$, avoiding numerical problems. From there, we can solve for $l_{r,j,k}$ by taking the derivative of the objective and equating it to zero. Then we multiply both sides of the resulting equation by $\exp\left(l_{s,j,k} - 1 - l_{r,j,k}\right)$ and solve a quadratic equation. The result is below.

Proposition 1 *If $R_-^{l,j,k} > 0$, the solution to (2.22) is*

$$l_{r,j,k} = l_{s,j,k} - 1 + \log\left(\frac{-\beta + \sqrt{\beta^2 + 4cR_+^{l,j,k} R_-^{l,j,k}}}{2R_+^{l,j,k}}\right). \tag{2.25}$$

.

If $R_-^{l,j,k} = 0$ or close to zero, which can happen when there are no points outside the smallest enclosing box in direction j, we set $l_{r,j,k} = \overline{l_j}$ where $\overline{l_j}$ is the smallest value of feature j. In that case, the boundary effectively disappears from the description of the classifier, making it more interpretable.

The interpretation of the proposition is that the boundary has moved from its starting position $l_{s,j,k}$ by amount $1 - \log\left(\frac{-\beta + \sqrt{\beta^2 + 4cR_+^{l,j,k} R_-^{l,j,k}}}{2R_+^{l,j,k}}\right)$.

Similarly, we let $u_{r,j,k}$ be the revised upper boundary of the jth dimension for the k-th revised box and it can be computed as follows.

Proposition 2 *If* $R_{-}^{l,j,k} > 0$,

$$u_{r,j,k} = u_{s,j,k} + 1 + \log\left(\frac{\beta + \sqrt{\beta^2 + 4cR_{+}^{u,j,k}R_{-}^{u,j,k}}}{2cR_{-}^{u,j,k}}\right) \tag{2.26}$$

where

$$R_{+}^{u,j,k} := \sum_{x \in S_{+}^{k} \cap X_{u,j,k}} \exp\left[-(u_{s,j,k} - x_j + 1)\right], \tag{2.27}$$

$$R_{-}^{u,j,k} := \sum_{x \in S_{-}^{k} \cap X_{u,j,k}} \exp\left[u_{s,j,k} - x_j + 1 + \sum_{p \neq j}(\lfloor x_p - u_{s,p,k} \rfloor_{+} + \lfloor l_{s,p,k} - x_p \rfloor_{+})\right]. \tag{2.28}$$

The proof and interpretation are similar to Proposition 1.

If $R_{-}^{u,j,k} = 0$ or close to zero, we set $v = \overline{u_j}$ where $\overline{u_j}$ is the largest possible value for feature j.

After we learn each of the decision boundaries, we perform a final adjustment that accomplishes two tasks: (i) it ensures that the box always expands rather than contracts, (ii) further expands the box to ϵ away from the nearest negative example. This gives us final values $l_{f,j,k}$ and $u_{f,j,k}$, where subscript "f" is for final. Written out, this is:

$$\begin{aligned} l_{f,j,k} &:= \sup\{x_j | x \in S_{-}, x_j \\ &< \min(l_{r,j,k}, l_{s,j,k})\} + \epsilon, \forall j, k \end{aligned} \tag{2.29}$$

$$\begin{aligned} u_{f,j,k} &:= \inf\{x_j | x \in S_{-}, x_j \\ &> \max(u_{r,j,k}, u_{s,j,k})\} - \epsilon, \forall j, k \end{aligned} \tag{2.30}$$

where ϵ is a small number. The boxes always expand for this algorithm, which

implies that this algorithm is meant for applications where correct classification of the minority class data is crucial in practice. This expansion step can be omitted if desired, for instance if misclassifying the negative examples is too costly.

The algorithm is summarized as follows:

Overall Algorithm

Input: number of boxes K, tradeoffs c and β, Data $\{\mathbf{x_i}, y_i\}_i$.
Output: Boundaries of boxes.

1. Normalize the features to be between -1 and 1.

2. Cluster the minority class data into K clusters.

3. Construct the minimal enclosing box for each cluster, that is compute starting boundaries $l_{s,j,k}$ and $u_{s,j,k}$, the j-th dimension lower boundary and upper boundary respectively for the k'th cluster.

4. Construct data for local classifiers $X_{l,j,k}$ and $X_{u,j,k}$ based on equations (2.22) and (2.22) respectively.

5. Compute $R_+^{l,j,k}$, $R_-^{l,j,k}$, $R_+^{u,j,k}$, $R_-^{u,j,k}$, according to equations (2.23), (2.24), (2.27), and (2.28).

6. Compute $l_{r,j,k}$ based on equation (2.25) and $u_{r,j,k}$ based on equation (2.26) respectively.

7. Perform expansion based on equations (2.29) and (2.30).

8. Un-normalize by rescaling the features back to get meaningful values.

Note that after the clustering step on the minority class data, all the other steps are easily parallellizable.

2.4 Prediction Quality

Now that we have two very different algorithms for creating box drawing classifiers, we will compare their performances experimentally.

Evaluation Metric

We chose to use the area under the convex hull of the ROC curve (AUH) [67] as our evaluation metric; it is frequently used for imbalanced classification problems and considers the full ROC (Receiver Operator Characteristic) curve to evaluate performance. To compute the AUH, we compute classifiers for various settings of the tradeoff parameter c, which controls the relative importance of positive and negative classes. Each setting of c corresponds to a single point on the ROC curve, with a count of true and false positives. We compute the AUH formed by the points on the ROC curve, and normalize as usual by dividing it by the number of positive examples times the number of negative examples. The best possible result is an AUH of 1.

Baseline Algorithms

For comparison, we consider logistic regression, an SVM with a radial basis kernel, CART, C4.5, Random Forests, AdaBoost (with decision trees), C5.0, and the Hellinger Distance Decision Tree (HDDT) [20]. Most of these algorithms were listed among the top 10 algorithms in data mining [97]in 2014. We included only the supervised learning models. Note that back in 2014, neural networks were not among the top 10 algorithms in data mining. Among these algorithms, only CART, C4.5, C5.0, and HDDT yield potentially interpretable models. HDDT uses the Hellinger distance as the splitting criterion, which is robust and skew-insensitive.

In addition to the baselines above, we implemented the Patient Rule Induction Method (PRIM) for “bump hunting" [29]. This method also partitions the input variable space into box-shaped regions, but in a different way than our method. PRIM searches iteratively for sub-regions where the target variable has a maximum and peels them off one at a time, whereas our clustering step finds maxima simultaneously.

The data sets we considered are listed in Table 2.2. Some data sets (castle, corner, diamond, square, flooded, castle3D, corner3D, diamond3D, flooded3D, flooded3D) are simulated data that are able to be visualized (made publicly available at [30]). The breast and pima data sets were obtained from the UCI Machine Learning Repository [9]. The data set fourclass was obtained from LIBSVM [15]. The remaining imbalanced data sets were obtained from the KEEL (Knowledge Extraction based on Evolutionary Learning) imbalanced data repository [4]. The Iris0 data set is an imbalanced version of the standard iris data set, where two of the classes (iris-versicolor and iris-virginica) have been combined to form the majority class.

Performance analysis

Here we compare the performance of Fast Boxes with the baseline algorithms. For each algorithm (except C4.5) we set the imbalance weighting parameter to each value $[0.1, 0.2, 0.3, \ldots, 1]$. The other parameters were set in a data-dependent way; for instance, for SVM with RBF kernel, the kernel width was chosen using the sigest function in the R programming language. The data were separated into 10 folds, where each fold was used in turn as the test set. We do not prune the decision trees beyond their built-in pruning as previous research shows that unpruned decision trees are more effective in their predictions on the minority class [66, 16], and because it would introduce more complexity that would be difficult to control for. Within the training set, for the Fast Boxes algorithm we used 3-fold cross-validation to select the cluster number and expansion parameter.

Table 2.7 shows the performances in terms of AUH means and standard deviations. The values that are bolded represent the algorithms whose results are not statistically significantly different from the best algorithm using a matched pairs sign test with significance level $\alpha = 0.05$. When there was more than one best-performing classifier, the one with the smaller standard deviation was chosen as the best performer for that data set. Fast Boxes was often (but not always) one of the best performers for each dataset. This brings up several questions, such as: *Under what conditions does Fast Boxes perform well? How do its parameters effect the result? Does it tend to produce*

Data	number of examples	feature size	imbalance ratio
pima	768	8	1.8657
castle	8716	2	22.2427
corner	10000	2	99
diamond	10000	2	24.9067
square	10000	2	11.2100
flooded	10000	2	31.1543
fourclass	862	2	1.8078
castle3D	545	3	7.2576
corner3D	1000	3	28.4118
diamond3D	1000	3	33.4828
square3D	1000	3	7
flooded3D	1000	3	26.7778
breast	569	30	1.6840
abalone19	4174	9	129.4375
yeast6	1484	8	41.4
yeast5	1484	8	32.7273
yeast1289	947	8	30.5667
yeast4	1484	8	28.0980
yeast28	482	8	23.1000
yeast1458	693	8	22.1000
abalone918	731	9	16.4048
pageblocks134	472	10	15.8571
ecoli4	336	7	15.8000
yeast17	459	7	14.3
shuttle04	1829	9	13.8699
glass2	214	9	11.5882
vehicle3	846	18	2.9906
vehicle1	846	18	2.8986
vehicle2	846	18	2.8807
haberman	306	3	2.7778
yeast1	1484	8	2.4592
glass0	214	9	2.0571
iris0	150	4	2
wisconsin	683	9	1.8577
ecoli01	220	7	1.8571
glass1	214	9	1.8158
breast tissue	106	9	3.8182

Table 2.2: Summary of datasets used for experiments

trivial results? Can Exact Boxes improve upon Fast Boxes' results in cases where it does not perform well? Are the results interpretable? These are questions we will

address in the remainder of this section.

We start with a partial answer to the question of when Fast Boxes performs well — it is when the classes are more imbalanced. Figure 2-3 shows a scatter plot of the quality of Fast Boxes' performance versus the imbalance ratio of the dataset. The vertical axis represents our rank in performance among all of the algorithms we tested. The horizontal axis is the number of negatives divided by the number of positives. The performance of Fast Boxes changes from being among the worst performers when the data are not imbalanced (and the cluster assumption is false), to being among the best performers when the data are imbalanced.

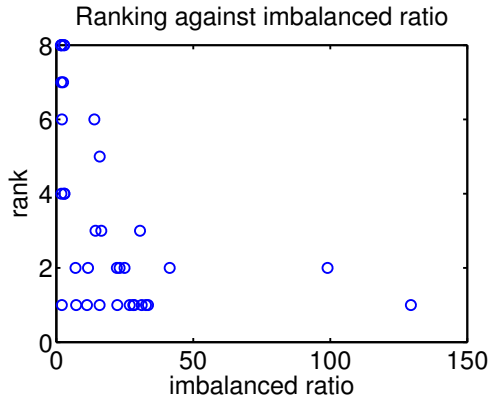


Figure 2-3: Ranking of Fast Boxes versus imbalance ratio of data

Below we provide some intuition about Fast Boxes' clusters and the expansion parameter before answering the questions posed just above.

Effect of Fast Boxes' parameter settings

We expect that if our main modeling assumption holds, which is that the positive examples naturally cluster, there should be a single best number of clusters. If we choose the number of clusters too small, we might underfit, and if we allow too many clusters, we could overfit. Figure 2-4 illustrates the cluster assumption on the diamond3D dataset, where this effect of overfitting and underfitting can be seen.

The expansion parameter is also designed to assist with generalization. We would like our boxes to be able to capture more of the positive cluster than is provided by

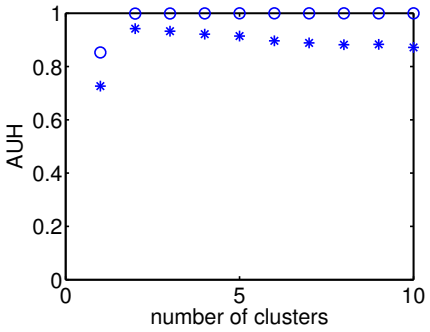


Figure 2-4: The effect of the number of clusters on AUH for the data set diamond3D. Fast Boxes was run once for each number of clusters. Training AUH is reported as circles, and testing AUH as stars.

the tightest box around the training examples, particularly since true positives are worth more than true negatives in our objective function. The exponential loss creates a discriminative classifier, but with a push outwards. Naturally, as we increase the expansion parameter, the training AUH will drop as more negative training examples are included within the box. On the other hand, the test AUH tends to increase before decreasing, as more positive examples are within the expanded box. This effect is illustrated in Figure 2-5.

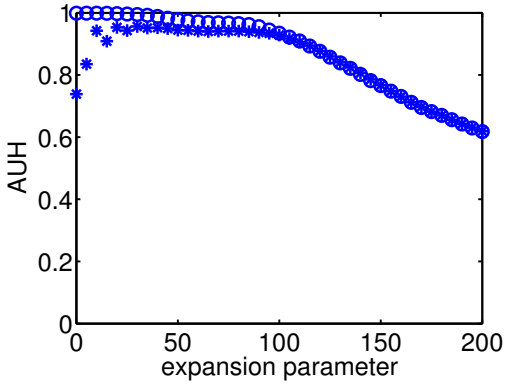


Figure 2-5: The effect of the expansion parameter on AUH for the diamond3D data set.

Considering the final expansion stage, Figure 2-6 illustrates why this stage is necessary. We visualize the iris0 dataset with dimension 1 and dimension 4, where if we had not expanded out to the nearest negative example, we would have missed a

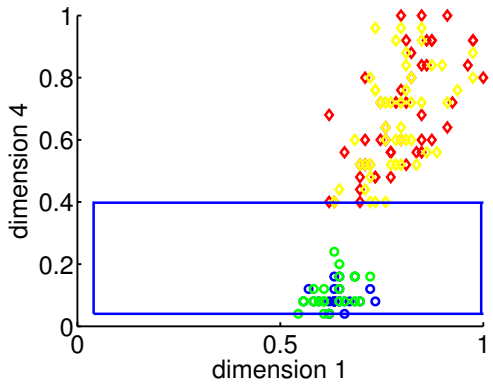


Figure 2-6: The red and yellow points are negative training points and testing point respectively, the blue and green points are positive training points and testing points respectively. If we had used the tightest decision boundary around the positive training examples, we would have missed part of the positive distribution.

part of the positive distribution within the test set.

Production of trivial rules

When the data are highly imbalanced, we have found that some of the baseline algorithms for producing interpretable models often produce trivial models, that is, models that always predict a single class. This is true even when the weighting factor on the positive class is varied throughout its full range at a reasonably fine granularity. This means that either it is not possible to obtain a meaningful model for the dataset with that method, or it means one would need to work fairly hard in order to find a weighting factor that did not produce a trivial model; that is, the range for which nontrivial models are possible is very small. Table 2.3 considers three interpretable methods we compare with, namely CART, C4.5, and C5.0. It shows the fraction of time these algorithms produce trivial models. For CART, C5.0, and Fast Boxes, the percentage was computed over 100 models computed over 10 splits and 10 options for the imbalance parameter. C4.5 does not have a built in imbalance parameter, so the percentage was computed over 10 splits.

Data	CART	C4.5	C5.0	Fast Boxes
pima	0.00	0.00	0.00	0.07
castle	0.00	0.00	0.00	0.10
corner	0.00	**0.60**	**0.70**	0.00
diamond	0.00	0.00	0.00	0.00
square	0.00	0.00	0.00	0.00
flooded	0.00	**0.70**	**0.80**	0.00
fourclass	0.00	0.00	0.00	0.03
castle3D	0.00	0.00	0.00	0.10
corner3D	0.00	**0.50**	**0.50**	0.07
diamond3D	0.00	**1.00**	**1.00**	0.06
square3D	0.00	**0.90**	**0.80**	0.10
flooded3D	0.05	**1.00**	**1.00**	0.09
breast	0.00	0.00	0.00	0.37
abalone19	0.43	**1.00**	**1.00**	0.35
yeast6	0.00	0.00	0.00	0.02
yeast5	0.00	0.00	0.00	0.36
yeast1289	0.16	**0.70**	**0.60**	0.35
yeast4	0.00	0.20	0.20	0.30
yeast28	0.39	**0.90**	**0.90**	0.00
yeast1458	0.24	**0.70**	**0.90**	0.19
abalone918	0.00	0.10	0.10	0.40
pageblocks134	0.00	0.00	0.00	0.39
ecoli4	0.00	0.00	0.00	0.32
yeast17	0.03	0.20	0.30	0.21
shuttle04	0.00	0.00	0.00	0.00
glass2	0.09	0.40	**0.70**	0.28
vehicle3	0.00	0.00	0.00	0.01
vehicle1	0.00	0.00	0.00	0.02
vehicle2	0.00	0.00	0.00	0.27
haberman	0.00	0.40	**0.70**	0.13
yeast1	0.00	0.00	0.00	0.08
glass0	0.00	0.00	0.00	0.08
iris0	0.00	0.00	0.00	0.03
wisconsin	0.00	0.00	0.00	0.34
ecoli01	0.00	0.00	0.00	0.21
glass1	0.00	0.00	0.00	0.16
breast tissue	0.00	0.00	0.00	0.08

Table 2.3: Fraction of the time we get a trivial model. Bold indicates values over 0.5.

Comparison of Fast Boxes and Exact Boxes

Since we know that Fast Boxes is competitive with other baselines for handling imbalanced data, we would like to know whether Exact Boxes has the potential to yield

significant performance gains over Fast Boxes and other methods. We implemented the MIP using GUROBI on a quad core Intel i7 860 2.8 GHz, 8GB cache, processor with 4 cores with hyperthreading and 16GM of RAM. We first ran the Exact Boxes algorithm for 30 minutes, and if the AUH performance was not competitive and the optimality gap was above 1%, we ran it up to 90 minutes for each instance. We did not generally allow the MIP to solve to provable optimality. This has the potential to hinder performance, but as we were performing repeated experiments we needed to be able to solve the method repeatedly.

Table 2.4 shows results from Exact Boxes for several of the smaller data sets, along with the results from Fast Boxes for comparison. Bold font in this table summarizes results from the other baseline algorithms as well: if the entry is in bold, it means that the result is not statistically significantly different than the best out of *all* of the algorithms. Thus, for 5 out of 8 datasets we tried, the MIP was among the top performers. Further, the AUH value was substantially improved for some of the data sets. Thus, restricting the algorithm to produce a box drawing classifier does not generally seem to hinder performance.

Note that it is time-consuming to perform cross-validation on the MIP, so the cluster number that we found using cross-validation for Fast Boxes was used for Exact Boxes.

Interpretability demonstration

We provide a classifier we learned from the glass2 data set that predicts whether a particular glass is a building window that is non-float processed. The other types of glasses are building windows that are float processed, vehicle windows, containers, tableware, and headlamps. The attributes include the refraction index as well as various indices for metals. These metals include Sodium, Magnesium, Aluminum, Silicon, Potassium, Calcium, Barium, and Iron.

One of the predictive models from Fast Boxes is as follows. To be a particular glass of a building window that is non-float processed:
1) The refractive index should be above 1.5161.

Data	Best Performance	Fast Boxes	Exact Boxes	Exact Boxes ranking
vehicle2	0.9496 (0.015)	0.9191 (0.0242)	0.9496 (0.015)	1
haberman	0.6699 (0.0276)	0.5290 (0.0265)	**0.6632** (0.0303)	2
yeast1	0.7641 (0.0133)	0.5903 (0.0286)	0.7392 (0.0172)	2
glass0	0.8312 (0.0345)	0.7937 (0.0212)	0.7977 (0.0421)	2
iris0	1 (0)	**1** (0)	**1** (0)	1
wisconsin	0.9741 (0.0075)	0.8054 (0.1393)	**0.9726** (0.0079)	2
ecoli01	0.9840 (0.0105)	0.9433 (0.0300)	**0.9839** (0.0109)	2
glass1	0.7922 (0.0377)	0.6654 (0.0356)	**0.7922** (0.0337)	1

Table 2.4: Comparison of test data AUH of interpretable methods with Exact Boxes. Bold font includes results from non-interpretable methods.

2) Magnesium index must be above 3.3301.

3) Aluminum should be below 1.7897.

4) Silicon should be below 73.0199.

5) Potassium should be below 0.6199.

6) Calcium should be between 8.3101 and 2.3741.

7) Barium should be below 2.6646.

8) Sodium and iron are not important factors.

We believe that this simple form of model would appeal to practitioners because of the natural threshold structure of the box drawing classifiers.

2.5 Theoretical guarantee on performance

Statistical learning theory will allow us to provide a probabilistic guarantee on the performance of our algorithms. We will construct a uniform generalization bound,

which holds over all box drawing classifiers with K boxes anchored at M_j different fixed values for each dimension, where K is fixed. We might choose M_j as the count of numbers with at most a certain number of decimal places (say 2 decimal places) in between the largest and smallest possible values for a particular feature. (Often in practice only 2 decimal places are used.) The main step in our proof is to count the number of possible box drawing classifiers. The set of all box drawing classifiers with up to K boxes, with l_j and u_j attaining the M_j values, will be called F.

Define the empirical risk to be the objective of Exact Boxes with no regularization,

$$R^{emp}(f) = \sum_{i:y_i=1} \mathbf{1}_{[f(\mathbf{x}_i)=1]} + C_I \sum_{i:y_i=-1} \mathbf{1}_{[f(\mathbf{x}_i)=-1]},$$

and let the true risk $R^{true}(f)$ be the expectation of this taken over the distribution that the data are drawn iid from.

Proposition 3 *For all $\delta > 0$ with probability at least $1 - \delta, \forall f \in F$,*

$$R^{true}(f) \leq R^{emp}(f) + \sqrt{\frac{K \sum_{j=1}^{n} \log(\frac{M_j(M_j-1)}{2}) - logK! + \log \frac{1}{\delta}}{2m}}.$$

To outline the proof, there are $\prod_{j=1}^{n} \begin{pmatrix} M_j \\ 2 \end{pmatrix}$ ways to construct a single box, since for each dimension, we select 2 values, namely the lower boundary l_j and upper boundary u_j. To construct multiple boxes, there are at most $\prod_{j=1}^{n} \begin{pmatrix} M_j \\ 2 \end{pmatrix}^K$ ways if the order of construction of the boxes matter. Since the order does not matter, we need to divide the term by $K!$. Note that this is an upper bound which is not tight since some boxes can be a proper subset or equal to another box. Although we are considering the set of all box drawing classifiers up to K boxes, it suffices to consider box drawing classifiers with exactly K boxes. This can be seen by supposing we constructed a classifier with $l < K$ boxes, and noting the same classifier can be constructed using K boxes by duplicating some boxes. We apply Hoeffding's inequality and the union bound to complete the proof.

2.6 Making the MIP more practical

From the experimental outcome, it is clear that Exact Boxes is indeed a competitive solution. The main challenge lies in its computational complexity. There are several ways one might make the MIP more practical: first, one could limit computation to focus only a neighborhood of the positive data, and use the solution to this problem to warm start the MIP on the full problem. In that case we would consider only negative points that are close to the positive points in at least one dimension, which can be identified in a single pass through the negative examples. Alternatively, one can perform clustering first as in the Fast Boxes approach, and solve the MIP on each cluster. For each cluster, we would scan through each feature of the data in a single pass and keep only the data that are close to the mean of the cluster center to use in the MIP.

2.7 Discussion and Conclusion

We have presented two new approaches to designing interpretable predictive models for imbalanced data settings. Exact Boxes is formulated as a mixed integer program, and acts as a gold standard interpretable modeling technique to compare with. It can be used for small to moderately sized problems. Fast Boxes uses a characterize-then-discriminate approach, and tends to work well when the minority class is naturally clustered (for instance when the clusters represent different failure modes of mechanical equipment). We illuminated the benefits and limitations of our approaches, and hope that these types of models will be able to provide alternative explanations and insights into imbalanced problems. In comparing Fast Boxes with gold standard interpretable techniques like Exact Boxes, and with many other methods, we can now judge the power of the class of interpretable models: it is interesting that such simple approaches can achieve comparable performance with even the best state-of-the-art techniques.

Data	Logistic	SVM	CART	C4.5	Ada-Boost	RF	C5.0	HDDT	Fast Boxes
pima	**0.8587** (0.0112)	**0.8468** (0.0126)	0.7738 (0.0123)	0.6579 (0.0347)	0.6810 (0.0218)	0.6942 (0.0126)	0.6574 (0.0353)	0.6642 (0.0274)	0.7298 (0.0241)
castle	0.5 (0)	**1** (0)	0.9941 (0.0068)	0.9947 (0.0060)	**0.9949** (0.0046)	**0.9922** (0.0079)	0.9941 (0.0060)	**0.9949** (0.0062)	**1** (0)
corner	**0.9871** (0.0129)	**0.9948** (0.0005)	0.9488 (0.2717)	0.5997 (0.1482)	0.6984 (0.0449)	0.6828 (0.0265)	0.5612 (0.1110)	0.6865 (0.0365)	0.9891 (0.0001)
diamond	0.5 (0)	**0.9980** (0.0004)	0.9585 (0.0129)	0.9328 (0.0181)	0.9460 (0.0117)	0.9433 (0.0121)	0.9311 (0.0208)	0.9364 (0.0180)	0.9744 (0.0062)
square	0.5404 (0.0718)	0.9944 (0.0001)	0.9949 (0.0051)	0.9949 (0.0043)	0.9939 (0.0033)	0.9947 (0.0033)	0.9949 (0.0043)	0.9949 (0.0027)	**0.9984** (0.0015)
flooded	0 (0)	**0.9831** (0.0010)	0.9466 (0.0157)	0.5488 (0.1074)	0.7017 (0.0231)	0.7036 (0.0252)	0.5482 (0.1077)	0.6992 (0.0208)	09638 (0.0091)
fourclass	0.8122 (0.0195)	**0.9957** (0.0176)	0.9688 (0.0176)	0.9916 (0.0296)	0.9670 (0.0265)	**0.9920** (0.0053)	0.9670 (0.0130)	0.9698 (0.0116)	0.9546 (0.0174)
castle3D	0.5449 (0.0324)	**1** (0)	0.9532 (0.0347)	0.9530 (0.0374)	0.9272 (0.0499)	**0.9455** (0.0563)	0.9439 (0.0615)	0.9530 (0.0374)	**1** (0)
corner3D	0.8448 (0.0316)	**0.9225** (0.0463)	0.8481 (0.0504)	0.5596 (0.0729)	0.6245 (0.03927)	0.5657 (0.0309)	0.5622 (0.0778)	0.6413 (0.0457)	**0.9736** (0.0091)
diamond3D	0.5449 (0.0324)	**0.7962** (0.0917)	0.7372 (0.0347)	0.5 (0.0374)	0.5492 (0.0499)	0.5957 (0.0309)	0.5622 (0.0778)	0.6883 (0.0542)	**0.9516** (0.0119)
square3D	0.5 (0)	**0.9626** (0.0156)	**0.9106** (0.0306)	0.5387 (0.1224)	0.8703 (0.01451)	0.8790 (0.0234)	0.5811 (0.1712)	0.9034 (0.0322)	**0.9578** (0.0090)
flooded3D	0.5 (0)	0.7912 (0.0781)	0.7724 (0.0902)	0.5 (0)	0.5471 (0.0329)	0.5489 (0.0440)	0.5 (0)	0.6422 (0.0749)	**0.9233** (0.0307)
breast	0.9297 (0.0230)	**0.9801** (0.0079)	0.9516 (0.0173)	0.9251 (0.0138)	0.9457 (0.0329)	0.9609 (0.0102)	0.9281 (0.0135)	0.9231 (0.0180)	0.8888 (0.0313)
abalone19	0.5188 (0.0182)	0.5 (0)	0.5382 (0.0261)	0.5 (0)	0.5 (0)	0.5 (0)	0.5 (0)	0.5116 (0.0164)	**0.6882** (0.0583)
yeast6	**0.8503** (0.0341)	**0.8649** (0.0246)	0.7995 (0.0624)	0.7129 (0.0829)	0.7126 (0.0536)	0.7277 (0.0581)	0.7129 (0.0853)	0.7064 (0.0772)	**0.8609** (0.0585)
yeast5	**0.9499** (0.0479)	0.9229 (0.0339)	**0.9197** (0.0575)	0.8280 (0.1159)	0.8305 (0.0859)	0.8061 (0.0616)	0.8241 (0.1157)	0.7931 (0.1126)	**0.9767** (0.0092)
yeast1289	**0.6319** (0.0433)	0.5618 (0.0332)	**0.7076** (0.0665)	0.5088 (0.0322)	0.5152 (0.0288)	0.5067 (0.0141)	0.5156 (0.0342)	0.5531 (0.0436)	0.5932 (0.0557)
yeast4	0.8001 (0.0309)	0.7836 (0.0480)	0.7595 (0.0410)	0.6115 (0.0902)	0.6131 (0.0326)	0.5922 (0.0326)	0.6210 (0.07899)	0.6289 (0.0471)	**0.8794** (0.0274)
yeast28	**0.7907** (0.0525)	0.6596 (0.0565)	0.6402 (0.0893)	0.5100 (0.0316)	0.5 (0)	0.6489 (0.0472)	0.5248 (0.0784)	0.6126 (0.0606)	**0.7366** (0.0467)
yeast1458	**0.6164** (0.0510)	0.5420 (0.0322)	**0.6032** (0.0281)	0.5 (0)	0.5023 (0.0088)	0.5095 (0.0154)	0.5 (0)	0.5340 (0.0467)	**0.6090** (0.0431)

Continued on next page

abalone918	**0.8849**	0.6780	0.7427	0.5904	0.6117	0.5580	0.5725	0.6310	0.7171
	(0.0270)	(0.0391)	(0.0517)	(0.0581)	(0.0456)	(0.03213)	(0.0470)	(0.0418)	(0.0603)
pageblocks	0.9461	0.7874	**0.9945**	**0.9908**	**0.9908**	**0.9500**	**0.9908**	**0.9551**	0.9500
134	(0.0444)	(0.1184)	(0.0109)	(0.0219)	(0.0449)	(0.0345)	(0.0219)	(0.0487)	(0.0359)
ecoli4	**0.8926**	**0.9176**	**0.8809**	0.7759	**0.7965**	**0.8494**	0.8471	**0.8430**	**0.9202**
	(0.0615)	(0.0424)	(0.0593)	(0.07756)	(0.0775)	(0.0775)	(0.0532)	(0.0743)	(0.0622)
yeast17	**0.7534**	**0.6905**	**0.7481**	0.5841	0.5382	0.5529	0.5721	0.6070	**0.7033**
	(0.0611)	(0.0386)	(0.0713)	(0.0698)	(0.0225)	(0.0359)	(0.0699)	(0.0509)	(0.0547)
shuttle04	**0.9965**	**0.9828**	**1**	**0.9994**	**1**	**1**	**1**	**0.9994**	**0.9967**
	(0.0045)	(0.0105)	(0)	(0.0008)	(0)	(0)	(0)	(0.0008)	(0.0042)
glass2	**0.7609**	0.6128	**0.7112**	0.5541	0.5324	0.5479	0.5200	0.5892	**0.7334**
	(0.0726)	(0.0941)	(0.1090)	(0.0640)	(0.0417)	(0.0597)	(0.0415)	(0.0573)	(0.0904)
vehicle3	**0.8397**	**0.8524**	0.7733	0.6515	0.6591	0.6484	0.6621	0.6823	0.7003
	(0.0079)	(0.0169)	(0.0255)	(0.0401)	(0.0212)	(0.0232)	(0.02117)	(0.0300)	(0.0267)
vehicle1	**0.8587**	0.8468	0.7738	0.6579	0.6810	0.6942	0.6574	0.6719	0.7298
	(0.0112)	(0.0126)	(0.0123)	(0.0347)	(0.0218)	(0.0126)	(0.0353)	(0.0265)	(0.0241)
vehicle2	0.9632	**0.9837**	0.9437	0.9351	0.9677	**0.9775**	0.9365	0.9248	0.9191
	(0.0134)	(0.0072)	(0.01880)	(0.0133)	(0.0097)	(0.0106)	(0.0129)	(0.0243)	(0.0242)
haberman	**0.6589**	**0.6898**	**0.6699**	**0.5733**	0.6004	0.6130	**0.5420**	0.5604	0.5290
	(0.1713)	(0.0427)	(0.0276)	(0.0748)	(0.0323)	(0.0318)	(0.6780)	(0.0231)	(0.0265)
yeast1	0.7836	**0.7991**	0.7641	0.6672	0.6859	0.6130	0.5420	0.6369	0.5903
	(0.0184)	(0.0150)	(0.0133)	(0.0372)	(0.0219)	(0.0318)	(0.0678)	(0.0128)	(0.0286)
glass0	0.7951	**0.8636**	0.8312	0.7687	0.7998	**0.8572**	0.7690	0.7569	0.7937
	(0.0437)	(0.0336)	(0.0345)	(0.0619)	(0.0381)	(0.0281)	(0.0595)	(0.0424)	(0.0212)
iris0	**1**	**0.998**	**1**	**0.978**	**1**	**1**	**0.972**	**0.9880**	**1**
	(0)	(0.0063)	(0)	(0.0175)	(0)	(0)	(0.0169)	(0.0193)	(0)
wisconsin	**0.9746**	**0.9735**	**0.9741**	0.9455	0.9611	0.9672	0.9416	0.9249	0.8054
	(0.0093)	(0.0073)	(0.0075)	(0.0124)	(0.0122)	(0.0072)	(0.0121)	(0.0203)	(0.1393)
ecoli01	**0.9728**	**0.9850**	**0.9840**	**0.9806**	**0.9828**	**0.9855**	**0.9806**	0.9806	0.9433
	(0.0140)	(0.0091)	(0.0105)	(0.0107)	(0.0063)	(0.0097)	(0.0107)	(0.0107)	(0.0300)
glass1	0.7247	**0.8057**	0.7598	0.7050	0.6997	0.7833	0.6822	0.7189	0.6654
	(0.0363)	(0.0340)	(0.0490)	(0.0358)	(0.0478)	(0.0274)	(0.0320)	(0.0586)	(0.0356)
breast tis-	0.9411	0.9908	0.9417	0.9450	0.9632	0.9531	0.9630	0.9314	**0.9953**
sue	(0.0394)	(0.0064)	(0.0747)	(0.0602)	(0.0297)	(0.0505)	(0.0403)	(0.0550)	(0.0042)

Table 2.5: Comparison of test data AUH of Fast Boxes with other algorithms

Chapter 3

Cascaded Bayesian Histograms For Density Estimation

3.1 Introduction

A histogram is a piecewise constant density estimation model. There are good reasons that the histogram is among the first techniques taught to any student dealing with data [14]: (i) histograms are easy to display, (ii) they are accurate as long as there are enough data in each bin. A downside of the conventional histogram is that all of these properties fail in more than 2 or 3 dimensions, particularly for binary or categorical data. One cannot easily display or write down a conventional histogram in more than 3 dimensions. For binary data this would require us to display a multidimensional dimensional hypercube. In terms of accuracy, there may not be enough data in each bin, so the estimates would cease to be accurate. In terms of interpretability, for a > 4 dimensional histogram, enumerating a large set of bin values ceases to be an interpretable representation of the data density, and can easily obscure important properties of the data distribution. Considering marginals is often useless for binary variables, since there are only two bins (0 and 1). The question is how to construct a piecewise constant density estimation model (like a histogram) for categorical data that has the properties mentioned above: (i) it can be displayed, (ii) it is an accurate estimate of the underlying density.

In this chapter we present three cascaded (tree- or list- structured) density estimation models. These are similar to variable bin-width histograms, (e.g., see [91, 84]), though our approaches use only a subset of the variables. A leaf (that is, a histogram bin) is defined by conditions on a subset of variables (e.g. "the second component of x is 0" and "the first component of x is 1"), and the density is estimated to be constant within each leaf.

Let us give an example to illustrate how each bin is modeled to be of constant density. Let us say we are modeling the population of burglaries (housebreaks) in a city, namely Cambridge, MA. We might want to create a density model to understand how common or unusual the particular details of a crime might be (e.g., do we see crimes like this every month, or is this relatively uncommon?). A leaf (histogram bin) in our model might be the following: if *premise* is `residence`, *owner present* is `false`, *location of entry* is `window`, then p(situation) is 0.20. This means that the total density in the bin where these conditions hold is 0.20, that is, for 20% of burglaries, the three conditions are met. Let us say we have an additional variable, *means of entry*, with outcomes `pried`, `forced`, and `unlocked`, indicating how the criminal entered the premise. Each of these outcomes would be equally probable in the leaf, each with probability $0.20/3 = .067$ since we assume the density in the leaf to be uniform. We described just one bin above, whereas a full tree could be that of Figure 3-1.

Bayesian priors control the shape of the tree in our methods. This helps with both generalization and interpretability. For the first method, the prior parameter controls the number of leaves. For the second method, the prior controls the desired number of branches for nodes of the tree. For the third method, which creates lists (one-sided trees), the prior controls the desired number of leaves and also the length of the leaf descriptions. Domain knowledge might be required to know how complex the tree structure to be, in the absent of such knowledge, we might assume uniform prior over a finite domain.

This generative structure aims to fix the issues with conventional histograms: (i) display: we need only write down the conditions we used in the tree- or list-

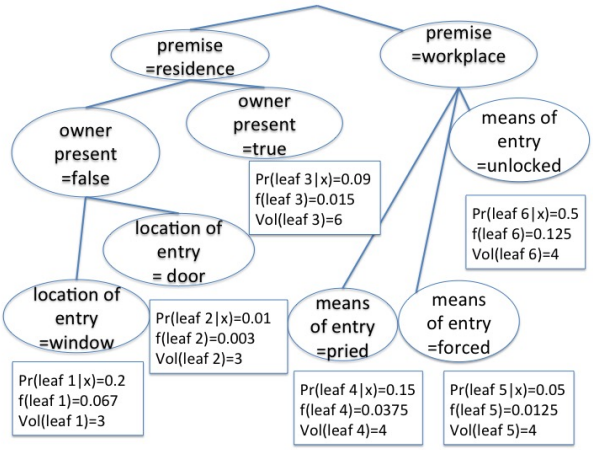


Figure 3-1: A sparse tree to represent the density of housebreaks in Cambridge MA. Probability of belonging to the leaf, the densities (f) and volume (Vol) are specified in the sparse tree.

shaped cascade to understand the model. It is transparent. (ii) accuracy: the prior encourages the cascade to be smaller, which means the bins are larger, and generalize better.

Density estimation is a classic topic in statistics and machine learning. Density estimation is much harder than classification, regression, or clustering; even kernel density estimation (KDE) methods do not provide high fidelity density estimates in more than a few dimensions. However, even between 4 and $\sim$ 10 dimensions the KDE models can be queried but not easily displayed. Another challenge is that since there is no formal criterion for defining what is the correct presentation of a data set, there is no clear evaluation function to determine which of your presentation methods is best. One frequently-used metric is the log likelihood. Without using domain-specific generative assumptions, the most useful techniques have been nonparametric, mainly variants of KDE [3, 77, 63, 13, 56, 59, 73, 92, 87, 24, 64]. KDE is highly tunable, not domain dependent, and can generalize well, but does not have the logical structure of histograms and cannot be easily displayed. In other words, one can query the model at specific points, but there is no easy way to display or convey the global model. Alternative methods with the same drawback include mixtures of Gaussians [51, 103, 61, 62, 19, 85], forest density estimation [53], RODEO [52] and other nonparametric Bayesian methods [58] which have been proposed for general

purpose density estimation. [44] provides a Bayesian treatment of latent directed graph structure for non-iid data, but does not focus on sparsity. Pólya trees are generated probabilistically for real valued features and could be used as priors [95]. [28] uses a projection pursuit method to perform density estimation. However, such an approach is highly sensitive to the projection index used. Another task related to density estimation is the level set estimation problem, where the question of interest is whether, given a parameter γ, the density at a leaf is higher than γ. [94] addresses the problem using a tree representation, like we do. In [38], a discretized kernel is used to construct level set trees. The most similar paper to ours is on density estimation trees (DET) [69]. DETs are constructed in a top-down greedy way. This gives them a disadvantage in optimization, often leading to lower quality trees. They do not have a generative interpretation, and their parameters do not have a physical meaning in terms of the shape of the trees (unlike the methods defined in this work). Other top-down greedy approaches include [100, 101, 99] where discrepancy is used, and [60] where negative log-likelihood and MISE are used as splitting criteria. Distinctions between our work and existing literature are that *we place a Bayesian prior directly on the shape of the trees* that we desire. Also *instead of greedy splitting, we aim to globally maximize the posteriors of Bayesian models.* This is the first method that we know of that aims to globally optimize accuracy and sparsity of density trees.

3.2 Models

For all the three models, we will need the following notation. There are p features. We express the path to a leaf as the set of conditions on each feature along the path. For instance, for a particular leaf (leaf t in Figure 3-2), we might see conditions that require the first feature $x_{\cdot 1} \in \{4, 5, 6\}$ and the second feature $x_{\cdot 2} \in \{100, 101\}$. Thus the leaf is defined by the set of all outcomes that obey these conditions, that is, the leaf could be

$$x \in \{x_{\cdot 1} \in \{4, 5, 6\}, x_{\cdot 2} \in \{100, 101\}, x_{3\cdot}, x_{\cdot 4}, \ldots, x_{\cdot p} \text{ are any allowed values}\}.$$

This implies there is no restriction on $x_{3.}, x_{.4}, \ldots, x_{.p}$ for observations within the leaf. Notationally, a condition on the j^{th} feature is denoted $x_{.j} \in \sigma_j(l)$ where $\sigma_j(l)$ is the set of allowed values for feature j along the path to leaf l. If there are no conditions on feature j along the path to l, then $\sigma_j(l)$ includes all possible outcomes for feature j. Thus, leaf l includes outcomes x obeying:

$$x \in \{x_{.1} \in \sigma_1(l), x_{.2} \in \sigma_2(l), \ldots, x_{.p} \in \sigma_p(l)\}.$$

For categorical data, the volume of a leaf l is defined to be $\vec{v}_l = \prod_{j=1}^{p} |\sigma_j(l)|$. We give an example of this computation next.

Volume Computation Example

The data are categorical. Possible outcomes for $x_{.1}$ are $\{1, 2, 3, 4, 5, 6, 7\}$. Possible outcomes for $x_{.2}$ are $\{100, 101, 102, 103\}$. Possible outcomes for $x_{.3}$ are $\{10, 11, 12, 13, 14, 15\}$. Possible outcomes for $x_{.4}$ are $\{8, 9, 10\}$.

Consider the tree in Figure 3-2. We compute the volume for leaf l. Here, $\sigma_1(l) =$

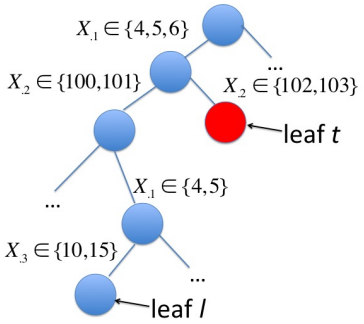


Figure 3-2: Example of computation of volume.

$\{4, 5\}$ since l requires both $x_{.1} \in \{4, 5, 6\}$ and $x_{.1} \in \{4, 5\}$. $\sigma_2(l) = \{100, 101\}$, $\sigma_3(l) = \{10, 15\}$, and $\sigma_4(l) = \{8, 9, 10\}$ because there is no restriction on $x_{.4}$. So

$$\vec{v}_l = \prod_j |\sigma_j(l)| = 2 \cdot 2 \cdot 2 \cdot 3 = 24.$$

Our notation handles only categorical data for ease of exposition but can be ex-

tended to handle ordinal and continuous data. For ordinal data, the definition is the same as for categorical but σ_j can (optionally) include only continguous values (e.g. $\{3, 4, 5\}$ but not $\{3, 4, 6\}$). For continuous variables, σ_j is the "volume" of the continuous variables, for example, for node condition $x_{\cdot j} \in (0, 0.5), \sigma_j = 0.5 - 0$.

In the next three subsections, we present the leaf-based modeling approach, branch-based modeling approach, and an approach to construct density rule lists.

3.2.1 Model I: Leaf-based Cascade Model

We define prior and likelihood for the tree-based model. To create the tree we will optimize the posterior over possible trees.

Prior:

For this model, the main prior on tree T is on the number of leaves K_T. This prior is Poisson (since its domain is the positive integers and it is chosen for computational elegance purpose) with a particular scaling (which will make sense later on), where the Poisson is centered at a user-defined parameter λ. Notation N_{K_T} is the number of trees with K_T leaves. The prior is:

$$\begin{aligned} &P(\text{Number of leaves in } T = K_T|\lambda) \\ &\propto N_{K_T} \cdot \ \text{Poisson}(K_T, \lambda) = N_{K_T} e^{-\lambda} \frac{\lambda^{K_T}}{K_T!}. \end{aligned}$$

Thus λ allows the user to control the number of leaves in the tree. The number of possible trees is finite, thus the distribution can be trivially normalized.

Among trees with K_T leaves, tree T is chosen uniformly, with probability $1/N_{K_T}$. This means the probability to choose a particular tree T is Poisson:

$$\begin{aligned} P(T|\lambda) \propto P(T|K_T)P(K_T|\lambda) \ &\propto \ \frac{1}{N_{K_T}} N_{K_T} e^{-\lambda} \frac{\lambda^{K_T}}{K_T!} = e^{-\lambda} \frac{\lambda^{K_T}}{K_T!} \\ &\propto \ \text{Poisson}(K_T, \lambda). \end{aligned}$$

We place a uniform prior over the probabilities for a data point to land in each of

the leaves. To do this, we start from a Dirichlet distribution with equal parameters $\alpha_1 = \ldots = \alpha_{K_T} = \alpha \in \mathbb{Z}^+$ where hyperparameter $\alpha > 1$. We denote the vector with K_T equal entries $[\alpha, \ldots, \alpha]$ as $\boldsymbol{\alpha}_{K_T}$. We draw multinomial parameters $\boldsymbol{\theta} = [\theta_1, \ldots, \theta_{K_T}]$ from $\text{Dir}(\boldsymbol{\alpha}_{K_T})$.

Thus, the first part of our model is as follows, given hyperparameters λ and α: Number of leaves in T: $K_T \propto$ scaled Poisson(λ), i.e., $N_{K_T} \cdot$ Poisson(K_T, λ), the tree shape, $T \propto$ Uniform over trees with K_T leaves, and Prior distribution over leaves: $\boldsymbol{\theta} \propto \text{Dir}(\boldsymbol{\alpha}_{K_T})$. As usual, the prior can be overwhelmed with a large amount of data.

Likelihood:

Let n_l denote the number of points captured by the l-th leaf, and denote $\vec{v}_l$ to be the volume of that leaf, defined above. The probability to land at any specific value within leaf l is $\frac{\theta_l}{\vec{V}_l}$. The likelihood for the full data set is thus

$$P(X|\boldsymbol{\theta}, T) = \prod_{l=1}^{K_T} \left(\frac{\theta_l}{\vec{v}_l}\right)^{n_l}.$$

Posterior:

The posterior can be written as follows, where we have substituted the distributions from the prior into the formula. Here, $B(\boldsymbol{\alpha}_{K_T}) = \frac{\prod_{l=1}^{K_T} \Gamma(\alpha_l)}{\Gamma(\sum_{l=1}^{K_T} \alpha_l)} = \frac{(\Gamma(\alpha))^{K_T}}{\Gamma(K_T \alpha)}$ is the multinomial beta function which is also the normalizing constant for the Dirichlet

distribution.

$$
\begin{aligned}
&P(T|\lambda, \boldsymbol{\alpha}, X) \\
&\propto \int_{\boldsymbol{\theta}:\text{simplex}} P(K_T|\lambda) \cdot P(T|K_T) \cdot P(\boldsymbol{\theta}|\boldsymbol{\alpha}_{K_T}) \cdot P(X|\boldsymbol{\theta}, T) d\boldsymbol{\theta} \\
&\propto \int_{\boldsymbol{\theta}:\text{simplex}} P(T|\lambda) \left[\frac{1}{B(\boldsymbol{\alpha}_{K_T})} \left(\prod_{l=1}^{K_T} \theta_l^{\alpha-1} \right) \right] \left[\prod_{l=1}^{K_T} \left(\frac{\theta_l}{\vec{v}_l} \right)^{n_l} \right] d\boldsymbol{\theta} \\
&\propto P(T|\lambda) \frac{1}{B(\boldsymbol{\alpha}_{K_T})} \left(\prod_{l=1}^{K_T} \left(\frac{1}{\vec{v}_l} \right)^{n_l} \right) \int_{\boldsymbol{\theta}:\text{simplex}} \prod_{l=1}^{K_T} \theta^{n_l+\alpha-1} d\boldsymbol{\theta} \\
&\propto P(T|\lambda) \frac{B(n_1+\alpha, \ldots, n_{K_T}+\alpha)}{B(\boldsymbol{\alpha}_{K_T})} \prod_{l=1}^{K_T} \frac{1}{\vec{v}_l^{n_l}} \\
&\propto P(T|\lambda) \frac{\Gamma(K_T\alpha)}{\Gamma(n+K_T\alpha)} \prod_{l=1}^{K_T} \frac{(n_l+\alpha-1)!}{(\alpha-1)!} \vec{v}_l^{-n_l},
\end{aligned}
$$

where $P(T|\lambda)$ is simply Poisson(K_T, λ) as discussed earlier. For numerical stability, we maximize the log-posterior which is equivalent to maximizing the posterior.

For the purposes of prediction, we are required to estimate the density that is being assigned to leaf l. This is calibrated to the data, simply as:

$$\hat{f} = \frac{n_l}{n\vec{v}_l}$$

where n is the total number of training data points and n_l is the number of training data points that reside in leaf l. The formula implicitly states that the density in the leaf is uniformly distributed over the features whose values are undetermined within the leaf (features for which σ_j contains all outcomes for feature j).

3.2.2 Model II: Branch-based Cascade Model

In the previous model, a Dirichlet distribution is drawn only over the leaves. In this model, a Dirichlet distribution is drawn at every internal node to determine branching. Similar to the previous model, we choose the tree that optimizes the posterior.

Prior:

The prior is comprised of two pieces: the part that creates the tree structure, and the part that determines how data propagates through it.

Tree Structure Prior: For tree T, we let $B_T = \{b_i | i \in I\}$ be a multiset, where each element is the count of branches from a node of the tree. For instance, if in tree T, the three nodes have 3 branches, 2 branches, and 2 branches respectively, then $B_T = \{3, 2, 2\}$. We let N_{B_T} denote the number of trees with the same multiset B_T. Note that B_T is unordered, so $\{3, 2, 2\}$ is the same multiset as $\{2, 3, 2\}$ or $\{2, 2, 3\}$.

Let I denote the set of internal nodes of tree T and let L denote the set of leaves. We let $\vec{v}_l$ denote the volume at leaf l.

In the generative model, a Poisson distribution with parameter λ is used at each internal node in a top down fashion to determine the number of branches. Iteratively, for node i, the number of branches, b_i, obeys $b_i \sim \text{Poisson}(\lambda)$. Hence, at any node i, with probability $\exp(-\lambda)\frac{\lambda^{b_i}}{b_i!}$, there are b_i branches from node i. This implies that with probability $\exp(-\lambda)$, the node is a leaf. In summary,

$$P(\text{Multiset of branches } = B|\lambda) \propto N_B \left[\prod_{i \in I} e^{-\lambda} \frac{\lambda^{b_i}}{b_i!}\right] \left[\prod_{l \in L} e^{-\lambda}\right].$$

Among trees with multiset B, tree T is chosen uniformly, with probability $\frac{1}{N_B}$. This means the probability to choose a particular tree is:

$$\begin{aligned} P(T|\lambda) \propto P(T|B_T)P(B_T|\lambda) &\propto \frac{1}{N_{B_T}} N_{B_T} \left[\prod_{i \in I} e^{-\lambda} \frac{\lambda^{b_i}}{b_i!}\right] \left[\prod_{l \in L} e^{-\lambda}\right] \\ &= \left[\prod_{i \in I} e^{-\lambda} \frac{\lambda^{b_i}}{b_i!}\right] \left[\prod_{l \in L} e^{-\lambda}\right]. \end{aligned} \tag{3.1}$$

Tree Propagation Prior: After the tree structure is determined, we need a generative process for how the data propagate through each internal node. We denote θ_l as the probability to land in leaf l. We denote $\widetilde{\theta}_{ij}$ as the probability to traverse to node j from internal node i. Notation $\boldsymbol{\theta}$ is the vector of leaf probabilities (the θ_l's), $\widetilde{\boldsymbol{\theta}}$ is the set of all $\widetilde{\theta}_{ij}$'s, and $\widetilde{\boldsymbol{\theta}}_i$ is the set of all internal node transition probabilities

from node i (the $\widetilde{\theta}_{ij}$'s).

We compute $P(\widetilde{\boldsymbol{\theta}}_i|\alpha, T)$ for all internal nodes i of tree T. At each internal node, we draw a sample from a Dirichlet distribution with parameter $[\alpha, \ldots, \alpha]$ (of size equal to the number of branches b_i of i) to determine the proportion of data, $\widetilde{\theta}_{i,j}$, that should go along the branch leading to each child node j from the internal parent node i. Thus, $\widetilde{\boldsymbol{\theta}}_i \sim \text{Dir}(\boldsymbol{\alpha})$ for each internal node i, that is:

$$P(\widetilde{\boldsymbol{\theta}}_i|\boldsymbol{\alpha}, T) = \frac{1}{B_{b_i}(\boldsymbol{\alpha})} \prod_{j \in C_i} \widetilde{\theta}_{ij}^{\alpha-1},$$

where $B_k(\boldsymbol{\alpha})$ is the normalizing constant for the Dirichlet distribution with parameter α and k categories, and C_i are the indices of the children of i. Thus,

$$P(\widetilde{\boldsymbol{\theta}}|\boldsymbol{\alpha}, T) = \prod_i P(\widetilde{\boldsymbol{\theta}}_i|\boldsymbol{\alpha}, T) = \prod_i \frac{1}{B_{b_i}(\boldsymbol{\alpha})} \prod_{j \in C_i} \widetilde{\theta}_{ij}^{\alpha-1}. \tag{3.2}$$

Thus, the prior is $P(T|\lambda) \cdot P(\widetilde{\boldsymbol{\theta}}|\alpha, T)$, where $P(T|\lambda)$ is in (3.1) and $P(\widetilde{\boldsymbol{\theta}}|\alpha, T)$ is in (3.2).

In summary, the prior of our model is as follows, given hyperparameters λ and α:

Multiset of branches, $B_T \propto N_{B_T} \left[\prod_{i \in I} e^{-\lambda} \frac{\lambda^{b_i}}{b_i!}\right] \left[\prod_{l \in L} e^{-\lambda}\right]$, the tree shape, $T \sim$ Uniform over trees with branches B_T, and the prior distribution over each branch, $\widetilde{\boldsymbol{\theta}}_i \sim \text{Dir}(\alpha)$.

$$e^{-\lambda(|I|+|L|)} \lambda^{\sum_{i \in I} b_i} \left(\prod_{i \in I} \frac{1}{b_i!} \frac{B_{b_i}(\alpha + n_{c_1}, \ldots, \alpha + n_{c_{b_i}})}{B_{b_i}(\alpha, \ldots, \alpha)} \right) \prod_{l \in L} \left(\frac{1}{\vec{v}_l^{n_l}} \right).$$

Possible Extension: We can include an upper layer of the hierarchical Bayesian Model to control (regularize) the number of features d that are used in the cascade out of a total of p dimensions. This would introduce an extra multiplicative factor within the posterior of $\begin{pmatrix} p \\ d \end{pmatrix} \gamma^d (1-\gamma)^{p-d}$, where γ is a parameter between 0 and 1,

where a smaller value favors a simpler model.

$$\left(\begin{array}{c} p \\ d \end{array} \right) \gamma^d (1-\gamma)^{p-d} e^{-\lambda(|I|+|L|)} \lambda^{|I|+|L|-1}$$
$$\left(\prod_{i \in I} \frac{1}{b_i!} \frac{B_{b_i}(\alpha + n_{c_1}, \ldots, \alpha + n_{c_{b_i}})}{B_{b_i}(\alpha, \ldots, \alpha)} \right) \prod_{l \in L} \left(\frac{1}{\vec{v}_l^{n_l}} \right).$$

3.2.3 Model III: Leaf-based Density Rule List

Rather than producing a general tree, an alternative approach is to produce a rule list. A rule list is a one-sided tree. Rule lists are easier to optimize than trees. Any tree can be expressed as a rule list; however, some trees may be more complicated to express as a rule list. By using lists, we implicitly hypothesize that more complicated trees may not be necessary.

An example of a density rule list is as follows:

if x obeys a_1 **then** density$(x) = f_1$
else if x obeys a_2 **then** density$(x) = f_2$
$\vdots$
else if x obeys a_m **then** density$(x) = f_m$
else density$(x) = f_0$.

The antecedents a_1,...,a_m are chosen from a large pre-mined collection of possible antecedents, called A.

We define A to be the set of all possible antecedents of size at most H, where the user chooses H. The size of A is: $|A| = \sum_{j=0}^{H} A_j$, where A_j is the number of antecedents of size j,

$$A_j = \sum_{\left[\begin{array}{c} t_1, t_2, \ldots, t_j \in \{1, \ldots, p\} \\ \text{s.t. } t_1 > t_2 \ldots > t_j \end{array} \right]} \prod_{i=1}^{j} q_{t_i},$$

where feature i consists of q_i categories.

Generative Process:

We now sketch the generative model for the tree from the observations x and antecedents A. Prior parameters λ and η are used to indicate preferences over the length of density list and the number of conjunctions in each sub-rule a_i.

Define $a_{<j}$ as the antecedents before j in the rule list if there are any. For example $a_{<3} = \{a_1, a_2\}$. Similarly, let c_j be the cardinalities of the antecedents before j in the rule list. Let d denote the rule list. The generative model is as follows, following exposition of [50]:

1. Sample a decision list length $m \sim P(m|A, \lambda)$.

2. For decision list rule $j = 1, \ldots, m$:
   Sample the cardinality of antecedent a_j in d as $c_j \sim P(c_j|c_{<j}, A, \eta)$.
   Sample a_j of cardinality c_j from $P(a_j|a_{<j}, c_j, A)$.

3. For observation $i = 1, \ldots, n$: Find the antecedent a_j in d that is the first that applies to x_i. If no antecedents in d applies, set $j = 0$.

4. Sample parameter $\boldsymbol{\theta} \sim$ Dirichlet $(\boldsymbol{\alpha})$ for the probability to be in each of the leaves, where $\boldsymbol{\alpha}$ is a user-chosen vector of size $m+1$, usually where all elements are the same. $f_i = \frac{\theta_i}{\vec{v}_i}$, where $\vec{v}_i$ is the volume.

Prior:

The distribution of m is the Poisson distribution, truncated at the total number of preselected antecedents:

$$P(m|A, \lambda) = \frac{\lambda^m/m!}{\sum_{j=0}^{|A|}(\lambda^j/j!)}, m = 0, \ldots, |A|.$$

When $|A|$ is huge, we can approximate $P(m|A, \lambda) \approx \lambda^m/m!$, since the denominator of the expression above would be close to 1.

We let $R_j(c_1, \ldots, c_j, A)$ be the set of antecedent cardinalities that are available after drawing antecedent j, and we let $P(c_j|c_{<j}, A, \eta)$ be a Poisson truncated to remove

values for which no rules are available with that cardinality:

$$P(c_j|c_{<j}, A, \eta) = \frac{(\eta^{c_j}/c_j!)}{\sum_{k \in R_{j-1}(c_{<j}, A)}(\eta^k/k!)}, \;\; c_j \in R_{j-1}(c_{<j}, A).$$

We use a uniform distribution over antecedents in A of size c_j excluding those in a_j,

$$P(a_j|a_{<j}, c_j, A) \propto 1, \qquad a_j \in \{a \in A \setminus a_{<k} : |a| = c_j\}.$$

The cascaded prior for the antecedent lists is thus:

$$P(d|A, \lambda, \eta) = P(m|A, \lambda) \cdot \prod_{j=1}^{m} P(c_j|c_{<j}, A, \eta) \cdot P(a_j|a_{<j}, c_j, A).$$

The prior distribution over the leaves $\boldsymbol{\theta} = [\theta_1, \ldots, \theta_m, \theta_0]$ is drawn from $\text{Dir}(\boldsymbol{\alpha}_{m+1})$:

$$P(\boldsymbol{\theta}|\alpha) = \frac{1}{B_{m+1}(\alpha, \cdots, \alpha)} \prod_{l=0}^{m} \theta_l^{\alpha-1}.$$

It is straightforward to sample an ordered antecedent list d from the prior by following the generative model that we just specified, generating rules from the top down.

Likelihood:

Similar to the first model, the probability to land at any specific value within leaf l is $\frac{\theta_l}{\vec{v}_l}$. Hence, the likelihood for the full data set is thus

$$P(X|\boldsymbol{\theta}, d) = \prod_{l=0}^{m} \left(\frac{\theta_l}{\vec{v}_l}\right)^{n_l}.$$

Posterior:

The posterior can be written as

$$
\begin{aligned}
& P(d|A,\lambda,\eta,\alpha,X) \\
& \quad \propto \int_{\boldsymbol{\theta}\in\text{simplex}} P(d|A,\lambda,\eta)\cdot P(\boldsymbol{\theta}|\alpha)\cdot P(X|\boldsymbol{\theta},d)d\boldsymbol{\theta} \\
& \quad = P(d|A,\lambda,\eta)\int_{\boldsymbol{\theta}\in\text{ simplex}} \frac{1}{B_{m+1}(\alpha,\cdots,\alpha)}\prod_{l=0}^{m}\theta_l^{\alpha-1}\left(\frac{\theta_l}{\vec{v}_l}\right)^{n_l} d\boldsymbol{\theta} \\
& \quad = P(d|A,\lambda,\eta)\frac{\prod_{l=0}^{m}\Gamma(n_l+\alpha)\vec{v}_l^{-n_l}}{\Gamma(\sum_{l=0}^{m}(n_l+\alpha))},
\end{aligned}
$$

where the last equality uses the standard Dirichlet-multinomial distribution derivation.

For these objective functions, there is an optimization algorithm that iteratively maximizes the posterior, of which the details can be found in [31] and [50]. We use a simulated annealing method to optimize the objective. Recently, some works [82, 5] have achieved provable optimality on minimization of models over a set of pre-mined rules. They have also noted that randomized methods, such as that of [98] or those considered in the present work, tend to produce models that are close to these optimal solutions. Thus, we have reason to believe that our MAP solutions for the rule list optimization are approximately globally optimal over the space of rule lists.

3.3 Experiments

Our experimental setup is as follows. We considered five models: the leaf-based cascaded histograms, the branch-based cascaded histograms, the leaf-based density list, regular histograms and density estimation trees (DET) [69]. To our knowledge, this essentially represents the full set of logical, high dimensional density estimation methods. To ascertain uncertainty, we split the data in half 5 times randomly and assessed test log-likelihood and sparsity of the trees for each method. A model with fewer bins and higher test likelihood is a better model.

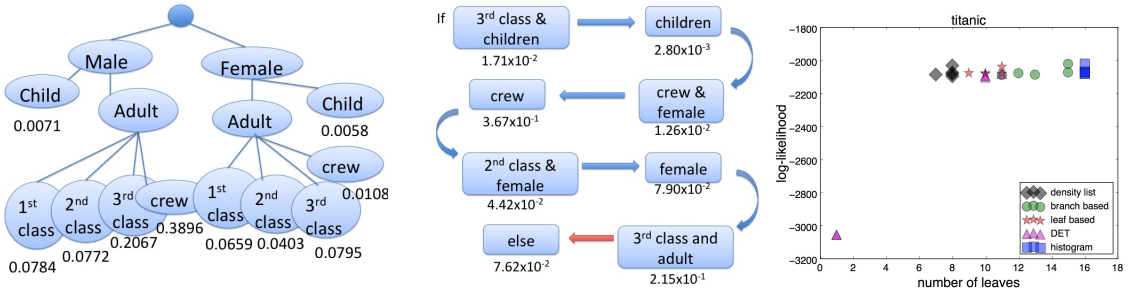
For the histogram, we treated each possible configuration as a separate bin. DET was designed for continuous data, which meant that the computation of volume

needed to be adapted – it is the number of configurations in the bin (rather than the lengths of each bin multiplied together). The DET method has two parameters, the minimum allowable support in a leaf, and the maximum allowable support. We originally planned to use a minimum of 0 and a maximum of the size of the full dataset, but the algorithm often produced trivial models when we did this, so we needed to resort to heuristics. We then tried values $\{0, 3, 5\}$ for the minimum values and $\left\{10, n, \lfloor \frac{n}{2} \rfloor\right\}$ where n is the number of training data points, and reported results for the best of these. For the leaf-based cascade model, the mean of the Poisson prior was chosen from the set $\{5, 8\}$ using nested cross validation. For the branch-based cascade model, the parameter to control the number of branches was chosen from the set $\{2, 3\}$. γ was fixed to be 0.5, and α was set to be 2 for the experiment. For the leaf-based density list model, the parameters λ, η and α were chosen to be 3, 1, and 1 respectively.

3.3.1 Illustration: Titanic Dataset

The Titanic dataset has an observation for each of the 2201 people aboard the Titanic, and even on this simple dataset, we can see where other methods go wrong. There are 3 features: gender, whether someone is an adult, and the class of the passenger (first class, second class, third class, or crew member). A cascade would help us understand the set of people on board the Titanic.

Figure 3-3c shows the results, both in out-of-sample likelihood and sparsity, for each model, for each of the 5 folds. The histogram method had high likelihood, but also the most leaves (by design). The other methods performed similarly, arguably the leaf-based density list method performed slightly better in the likelihood-sparsity tradeoff. DET produced a trivial tree for one of the splits. In general, we will see similar results on other datasets: the histogram produces too many bins, the leaf-based density list model and leaf-based cascade performs well, and DET has inconsistent performance (possibly due to its top-down greedy nature, or the fact that DET approximately optimizes Hellinger distance rather than likelihood.) Figure 3-3a shows one of the density cascades generated by the leaf-based method. The



(a) (b) (c)

Figure 3-3: (a) Tree representing titanic. (b) List representing titanic. Each arrow represents an "else if" statement. This can be directly compared to the cascade in Figure (b). Slight differences in estimates between the two models occurred because we used different splits of data for the two figures. The estimates were robust to the change in data. (c) the scatter plot for titanic.

reason for the split is clear: the distributions of the males and females were different, mainly due to the fact that the crew was mostly male. There were fewer children than adults, and the volume of crew members was very different than the volume of $1^{\text{st}}, 2^{\text{nd}}$, and 3^{rd} class passengers. Figure 3-3b shows one of the density lists generated by our model. It shows that male crew and third class male adults have higher density on the ship.

3.3.2 Crime Dataset

We were interested in an applied problem that resulted as part of a collaboration with the Cambridge Police Department in Massachusetts. The motivation is to understand the common types of modus operandi (M.O.) characterizing housebreaks, which is important in crime analysis. The data consist of all housebreaks occurring in Cambridge between 1997 and 2012 inclusive. We used 6 categorical features.

1. Location of entry: "window," "door," "wall," and "basement."

2. Means of entry: "forceful" (cut, broke, cut screen, etc.), "open area," "picked lock," "unlocked," and "other."

3. Whether the resident is inside.

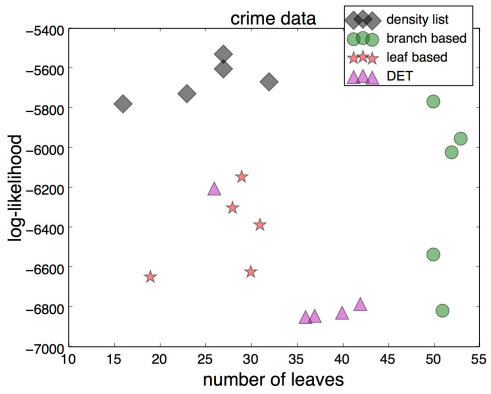


Figure 3-4: the scatter plot for the Cambridge Police Department dataset.

4. Whether the premise is judged to be ransacked by the reporting officer.

5. "Weekday" or "Weekend."

6. Type of premise. The first category is "residence" (including apartment, residence/unk., dormitory, single-family house, two-family house, garage (personal), porch, apartment hallway, residence unknown, apartment basement, condominium). The second category is "non-medical, non-religious work place" (commercial unknown, accounting firm, research, school). The third group consists of halfway houses, nursing homes, medical buildings, and assisted living. The fourth group consists of parking lots and parking garages, and the fifth group consists of YWCAs, YMCAs, and social clubs. The last groups are "storage," "construction site," "street," and "church" respectively.

The experiments show that DET and our approaches are competitive for the crime data set. The standard multi-dimensional histogram's results were not reported since they involve too many bins to fit on the figure 3-4.

These types of results can be useful for crime analysts to assess whether a particular modus operandi is unusual.

Figure 3-5 shows markers at crime locations on the Cambridge map. Each subfigure displays crimes within a different time range. Each crime is colored according to its leaf from the crime tree from Figure 3-8. These plots can be useful for crime series identification: If crimes are close to each other geographically, temporally, and have

a similar M.O., then those crimes may be part of a series. If we had used a regular histogram, it would have required 1440 different markers (discrete states). Instead, our crime tree groups the crimes into just 25 bins.

3.4 Empirical Performance Analysis

Each subsection below is designed to provide insight into how the models operate.

3.4.1 Sparse Tree Dataset

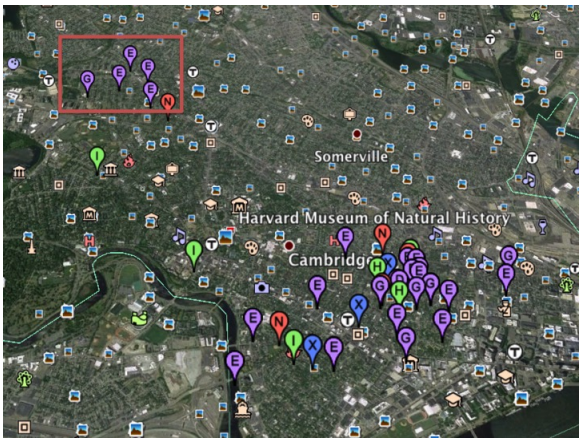
We generated a dataset that arises from a tree with 6 leaves, involving 3 features. The data consists of 1000 data points, where 100 points are tied at value (1,2,1), 100 points are at (1,2,2), 100 points are at (2,1,1), 400 points are at (2,1,2), and 300 points are at (2,2,2).

We trained the models on half of the dataset and tested on the other half. Figure 3-6 shows the scatter plot of out-of-sample performance and sparsity. This is a case where the DET failed badly to recover the true model. It produced a model that was too sparse, with only 4 leaves. The leaf-based cascade method recovered the full tree.

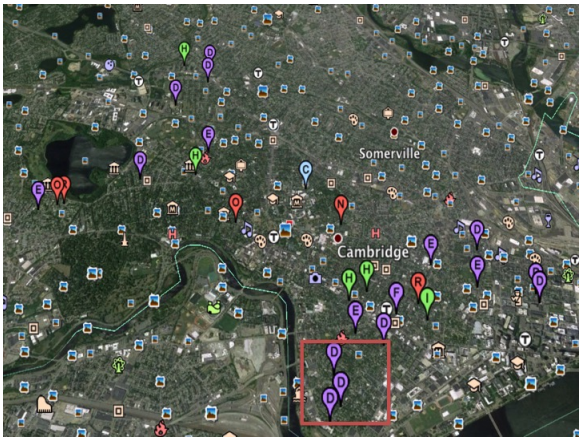
3.4.2 Extreme Uniform Dataset

We generated a 1-dimensional data set that consists of 100 data points. The data are simply all unique integers from 1 to 100. This is a case where the histogram badly fails to generalize. Figure 3-7 shows the result.

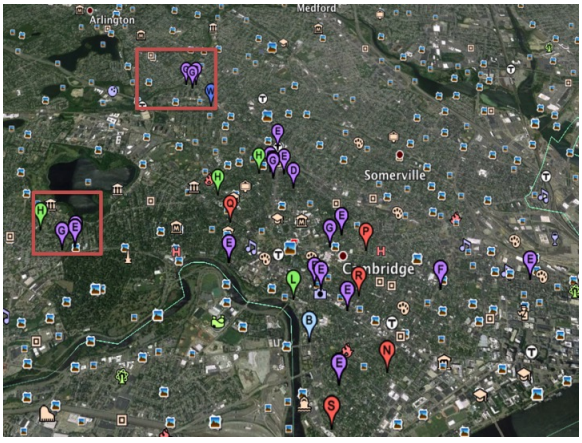
The leaf-based and branch-based models both return the solution that consists of a single root node, implying that the data are in fact uniformly distributed, or at least that we do not have evidence to further split on the single node. The density list output is close to uniform as well. DET is competitive as well, though it does not return the trivial tree. The histogram totally fails, since the test data and training data do not overlap at all.



(a)



(b)



(c)

Figure 3-5: Timeshot picture of crime events. Events in the boxes might be related to each other as they are close to each other geographically, temporally, and have a similar M.O.

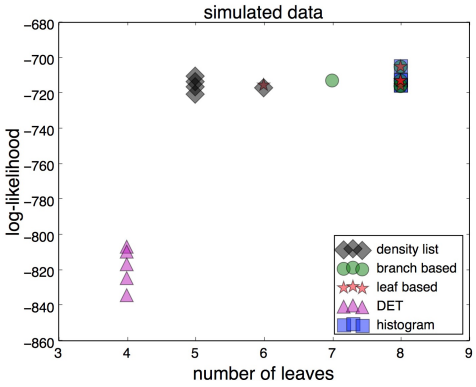


Figure 3-6: Performance vs. sparsity on a simulated data set.

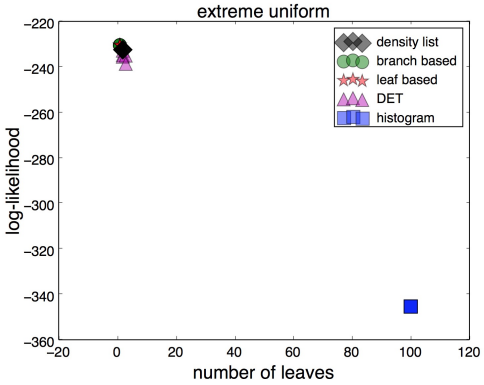


Figure 3-7: Performance vs. sparsity on uniform data set.

3.5 Consistency

A consistent model has estimates that converge to the real densities as the size of the training set grows. Consistency of conventional histograms is well studied [for example 2, 25]. More generally, consistency for general rectangular partitions has been studied by [102, 55]. Typical consistency proofs [e.g., 23, 69] require the leaf diameters to become asymptotically smaller as the size of the data grows. In our case if the ground truth density is a tree, we do not want our models to asymptotically produce smaller and smaller bin sizes, we would rather they reproduce the ground truth tree. This means we require a new type of consistency proof.

Definition 1: Trees have a single root and there are conditions on each branch. A density value, f_l is associated with each leaf l of the tree.

Definition 2: Two trees, T_1 and T_2 are equivalent with respect to density f if they assign the same density values to every data point on the domain, that is $f_{T_1}(x) = f_{T_2}(x)$, for all x. We denote the class of trees that are equivalent to T as $[T]_f$.

Theorem 1: *Let Θ be the set of all trees. Consider these conditions:*

1. $T_n \in \arg\max_T Obj(T)$, and the objective function can be decomposed into $\text{Obj}(T) = \ln q_n(T|X) + \ln g_n(T|X)$ where $\arg\max_T [\ln q_n(T|X) + \ln g_n(T|X)] \equiv \arg\max_T \ln g_n(T|X)$ as $n \to \infty$.

2. $\ln g_n(T|X)$ converges in probability, for any tree T, to the empirical log-likelihood that is obtained by the maximum likelihood principle, $\hat{l}_n(T|X) = \frac{1}{n}\sum_{i=1}^n \ln \hat{f}_n(x_i|T)$.

3. $\sup_{T\in\Theta} |\hat{l}_n(T|X) - l(T)| \xrightarrow{P} 0$ where $l(T) = \mathbb{E}_x(\ln(f(x|T)))$.

4. $T^*_{\text{MLE}} \in \arg\max_T l(T)$ is unique up to equivalence among elements of $[T^*_{\text{MLE}}]_f$.

If these conditions hold, then the trees T_n that we learned, $T_n \in \arg\max_T \text{Obj}(T)$, obey $T_n \in [T^*_{\text{MLE}}]_f$ for $n > M$ for some M.

The first condition and the second condition are true any time we use a Bayesian model. They are also true any time we use regularized empirical likelihood where the regularization term's effect fades with the number of observations. $q_n(T|X)$ refers to the part that does not vanish while $g_n(T|X)$ is the part that remains when the number of data points is sufficiently large. Note that the third condition is automatically true by the law of large numbers. The last condition is not automatically true, and requires regularity conditions for identifiability.

The result states that our learned trees are equivalent to maximum likelihood trees when there are enough data.

3.6 Conclusion

We have presented a Bayesian approach to density estimation using cascaded piecewise constant models. These estimators have nice properties: their prior encourages them to be sparse, which permits interpretability. In many cases, the models can be displayed easily on a piece of paper. They do not have the pitfalls of other nonparametric density estimation methods like density estimation trees, which are top-down greedy. They are consistent, without needing to asymptotically produce infinitesimally small leaves. Practically, the approaches presented here have given us insight into a real data set (the housebreak dataset from the Cambridge Police) that we could not have obtained reliably in any other way that we know of.

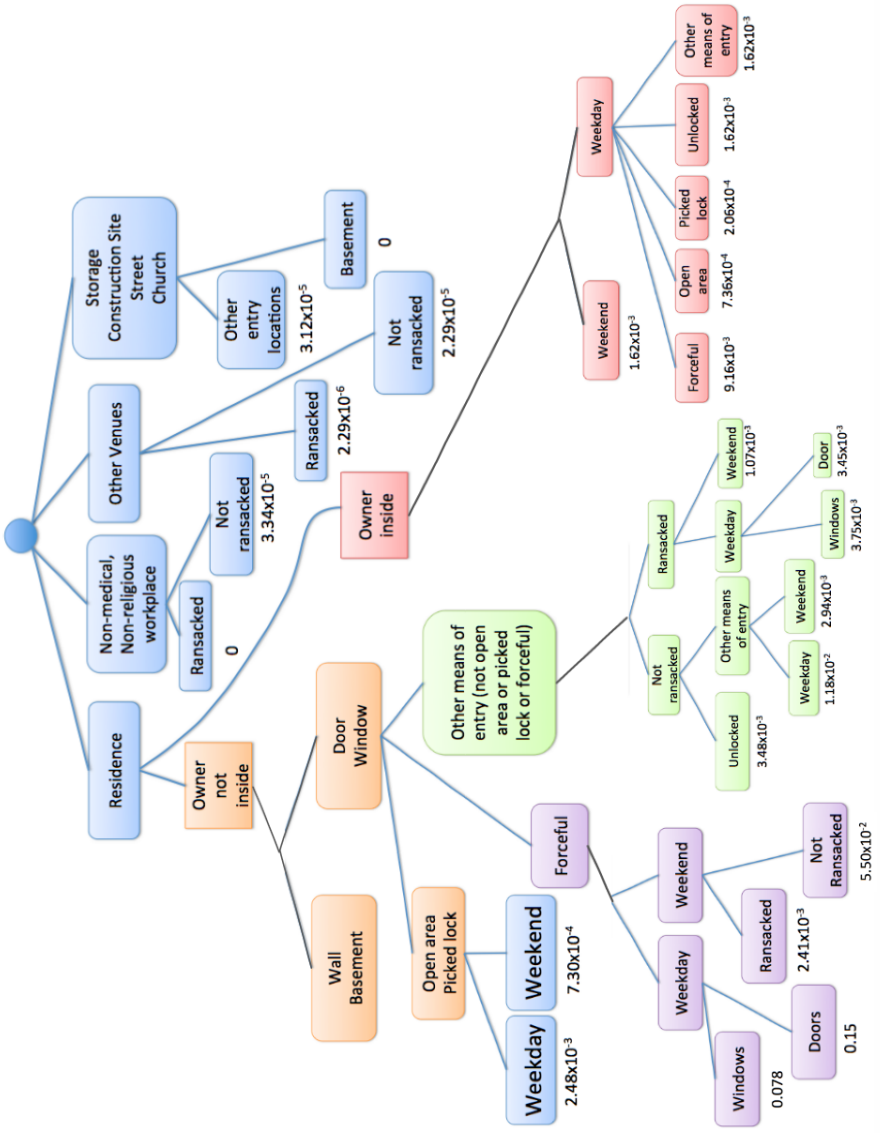


Figure 3-8: Tree representing the crime data set.

Chapter 4

A Minimax Surrogate Loss Approach to Conditional Difference Estimation

4.1 Introduction

Many data-driven decisions, such as whether to prescribe a particular pharmaceutical drug or whether to launch a particular marketing campaign, are problems of causal inference that require conditional difference estimation. Causal inference considers the effects of interventions, which is the basis for policy-making. It is well-known that standard machine learning methods are not designed to handle questions of causal inference; they are designed only for prediction and not for estimation of conditional differences or causal effects. A key reason that supervised machine learning does not usually handle causal inference problems is that by the nature of these problems, we do not observe counterfactuals (e.g., what would have happened if the same unit had not received the treatment), which means we are missing half of each label of a supervised learning problem. On the other hand, machine learning can handle powerful nonlinear modeling problems, which traditional causal inference methods cannot. Ideally, we would leverage the strengths of modern machine learning to create powerful models for conditional differences that could, in the right settings, be used for causal inference.

This work provides an approach to nonlinear treatment effect estimation using

machine learning, where the outcomes are binary (yes/no), and the goal is to predict whether treatment effects are positive, neutral, or negative. Given a new sample of units that are not in the training set, our goal is to decide which units would have a positive treatment effect, and which units would have a negative treatment effect. In this setting, we assume we can personalize who receives the treatment. Since the treatment is not globally launched across the population, it does not make sense to investigate average treatment effects. Since the outcome is binary, we are not interested in the estimated size of the treatment effect, but the simpler question of whether we have correctly determined whether the treatment effect is positive or negative for each individual. This is related to policy questions such as what fraction of the population did we correctly assign to the treatment.

We present a single formulation that handles treatment group and control group data simultaneously, and outputs a single function f whose thresholds at -1 and 1 provide decision boundaries between positive, neutral, and negative treatment effects. We provide a formulation as a type of "minimax" support vector machine. This handles either linear or non-linear treatment responses in a computationally efficient manner (via the kernel trick). By changing the kernel we create nonparametric models, and if we use the natural linear kernel, we create linear decision boundaries like regression models.

A large body of work in the causal inference community has focused on estimating average treatment effects (ATE) through linear models [79, 80, 81, 37, 74, 83, 10, 11], where the coefficient for the treatment variable provides an estimate for the ATE. As discussed earlier, ATE estimation is not relevant for determining who should receive a (personalized) treatment. Recently there has been a lot of work on subgroup identification [40, 42, 72] and other types of nonlinear predictors of causal effects [8, 45, 7, 90, 86]. Our work also focuses on personalized predictions of treatment effects, but differs from those listed above in several ways:

1) Our methods construct a single model with a tighter bound on the surrogate loss than a sum of treatment and control losses. This means that theoretically, our method should create more accurate single models than if one created separate treatment and

control models and subtracted them to obtain an estimate for the treatment effect. This arises from our formulation as a single regularized minimax problem.

2) The algorithm does not rely on greedy splitting and pruning heuristics or other non-convex optimization procedures, such as decision trees, random forests, matching, neural networks, etc. Our formulation is a single convex quadratic optimization problem that has known fast solution methods. Model complexity depends on the choice of kernel and regularization parameters, not on splitting or pruning parameters.

One work that seems similar to ours on the surface but is not, is that of Ratkovic and Tingley [71], who use support vector machines (SVM's) only to determine the largest balanced subset of data, by classifying which units are likely to have high density according to the treatment population distribution. From there, a traditional method is used to estimate conditional differences. Conversely, in our work, we use a traditional inverse propensity score model [89, 36] or other method to estimate the ratio of densities, and propose a single support vector machine formulation to estimate treatment effects.

Another work that is more relevant to ours is that of [41] who use a regularized squared hinge loss over all observations to estimate a single model that predicts outcomes for both treatment and control. The model includes two sets of covariates to predict outcomes, one that does not depend on the treatment and the other that does. For predicting conditional differences, the second set of covariates would not be used. Usually hinge loss is chosen to be a convex proxy for a given 0-1 loss that is hard to minimize, but it is not clear what the 0-1 loss is in their case, as it is not discussed. The 0-1 loss implicit in their formulation seems to be the sum of 0-1 losses for predictions of both treatment and control *outcomes* rather than a 0-1 loss for *treatment effects*. In this work, we prove that the 0-1 loss for prediction of outcomes is an upper bound on a relevant 0-1 loss for treatment effects, motivating the use of their 0-1 loss, but showing that it is a loose upper bound. In our formulation we use a tighter upper bound than the sum of 0-1 losses for prediction. The method of [41] is similar to estimating treatment and control outcomes with two separate models, because the estimated control outcomes can depend on one set of covariates (forming

one model), whereas the treatment outcomes can depend on the other (treatment-related) set of covariates (forming the second model). [1]

Let us discuss the estimation of density ratios in order to perform inverse propensity score weighting. We would use inverse propensity score weighting to correct for the fact that the control and/or treatment data does not come directly from the target population of interest. In general, if the (conditional) treatment effects are estimated correctly, then it is irrelevant whether or not the density ratios are poorly estimated. As an extreme case, if both the treatment and control losses are zero, the density ratio estimate is completely irrelevant. Thus, if we focus directly on accurate estimates of treatment effects, we may avoid problems faced by other methods that use looser surrogate loss functions.

In cases where conditional treatment effects are not able to be perfectly estimated, our method still can provide high quality treatment effect estimates without accurate estimation of the density ratio. Our predictor function minimizes the larger of the treatment loss and the control loss. If the target population is the treatment population, then the control loss involves density ratio estimation but not the treatment loss. Hence, if the treatment loss is always higher than the re-weighted control loss, then regardless of whether the density ratio is poorly estimated, the method will still produce the same answer. Its result is robust to poor density estimates when this happens.

Because support vector machines with radial basis functions are nonparametric, they are related to matching approaches. Historically, matching methods [76, 75, 22, 104, 46, 47] are different in that the matching is done prior to the modeling, with some exceptions [78]; here we use a single modeling approach.

[1] In fact their method would be identical to the two-separate-models approach if their model is chosen to be an indicator for control times a linear combination of variables plus an indicator for treatment times a linear combination of variables.

4.2 Problem Setting

We work in a standard potential outcomes setting, with observational data. Each observation possesses covariates and is assigned to either treatment or control groups, and an outcome is observed for each individual. The potential outcomes for observation i are denoted by Y_i^T or Y_i^C, where the superscript T denotes membership in the treatment group, and C denotes the control group. We cannot observe instances of $Y_i^T - Y_i^C$ since Y_i^T and Y_i^C are not simultaneously observable. Hence, this is a missing data problem where exactly half of the data are not observable. We define a binary causal exposure variable W, taking value 1 if the corresponding sample point belongs to the treatment group and 0 for control points. The outcome variable Y_i^{obs} is thus:

$$Y_i^{obs} = WY_i^T + (1 - W)Y_i^C.$$

We assume outcomes Y_i^T and Y_i^C depend on features (covariates) of the data. The features follow distributions $\mu_{X|T}$ and $\mu_{X|C}$ for treatment and control, respectively. Covariates are denoted by X. We often use upper case for random variables and lower case for draws of random variables. In notation, $Y^T \sim \mu_{Y^T|x}$, and $Y^C \sim \mu_{Y^C|x}$. Since we assume underlying treatment and control populations differ on the covariate space, they are notated as $\mu_{X|T}$ and $\mu_{X|C}$, where treatment observations follow $X \sim \mu_{X|T}$, and control observations follow $X \sim \mu_{X|C}$. By using the standard Radon-Nikodym derivative, we can effectively transform one distribution to another. The method we introduce can be trivially adapted to any target distribution, so for ease of notation we chose the treatment distribution $\mu_{X|T}$ to be our target population.

In this paper, we consider binary (yes/no) outcomes $y^T, y^C \in \{-1, 1\}$, e.g., whether or not someone had a heart attack. We let h denote our predictor function of $Y^T - Y^C$, which is a function of the covariates.

If we were given a predictor function *and the ground truth*, we might measure the quality of our predictor function using the following two conditional-difference loss

functions. The first one is:

$$l_{0-1}(x, y^T, y^C, h) = \begin{cases} \mathbb{1}_{|h(x)|\geq 1}, & y^T = y^C \\ \mathbb{1}_{h(x)\leq 0}, & y^T > y^C \\ \mathbb{1}_{h(x)\geq 0}, & y^T < y^C. \end{cases}.$$

This loss is 1 if there is no treatment effect and h predicts either a positive or negative treatment effect (top condition). The loss is 1 also when h predicts a treatment effect that is opposite from the true treatment effect. This loss function does not consider the average or magnitude of the treatment effect, it counts the number of individuals for whom the treatment effect was incorrectly predicted. This is a relevant loss when we aim to correctly assign treatment to individual members of a population: e.g., optimally assigning advertisements to individuals visiting a website, or optimally assigning a pharmaceutical drug to individuals who would benefit from it.

The second loss function that we consider is

$$l_\theta(x, y^T, y^C, h) = \begin{cases} \mathbb{1}_{|h(x)|\geq \theta}, & y^T = y^C \\ \mathbb{1}_{h(x)\leq -\theta}, & y^T > y^C \\ \mathbb{1}_{h(x)\geq \theta}, & y^T < y^C. \end{cases}$$

The first loss function is an upper bound for the second loss function, for margin $\theta > 1$. For the second loss function, l_θ, the set of possible inputs x for which $h(x)$ is within $(-\theta, \theta)$ can be interpreted as the region with no large predicted treatment effect. Provided that the predictions of treatment effect are real-valued, $h(x)$ can be rescaled, and thus to minimize l_θ it suffices to consider l_1 by letting $\theta = 1$.

$$l_1(x, y^T, y^C, h) = \begin{cases} \mathbb{1}_{|h(x)|\geq 1}, & y^T = y^C \\ \mathbb{1}_{h(x)\leq -1}, & y^T > y^C, \text{(false negative)} \\ \mathbb{1}_{h(x)\geq 1}, & y^T < y^C, \text{(false positive)}. \end{cases}$$

To see this, note that suppose we have $h_1 \in \arg\min_h l_1(.)$, then $\theta h_1 \in \arg\min_h l_\theta(.)$.

Hence, we will focus on l_θ where $\theta = 1$ when we derive our algorithm.

Since y^T and y^C are not observed simultaneously, it is not possible to compute the quantity above. This motivated us to use a surrogate function that separates y^T and y^C. Ideally, a surrogate function is an upper bound to the 0-1 loss, with a minimizer that can be easily computed. We dedicate the next section to an upper bound of the conditional-difference 0-1 loss function which in turn motivates the surrogate function explored in Section 4.4.

4.3 A Surrogate Conditional-Difference Loss Function

The following theorem defines sufficient conditions under which a surrogate loss function is valid for l_1.

Theorem 1 *If a function $l(.)$ satisfies $l(z) \geq \mathbb{1}_{z\geq 0} + \mathbb{1}_{z\geq 1}$, then we have*

$$\begin{aligned}
&\mathbb{E}_{X\sim\mu_{X|T},Y^T\sim\mu_{Y^T|X},Y^C\sim\mu_{Y^C|X}} l_1(X,Y^T,Y^C,h) \\
&\leq \max\left(\mathbb{E}_{X\sim\mu_{X|T},Y^T\sim\mu_{Y^T|X}} l(-h(X)Y^T), \mathbb{E}_{X\sim\mu_{X|C},Y^C\sim\mu_{Y^C|X}} \frac{l(h(X)Y^C)}{\mu_{X|C}(X)/\mu_{X|T}(X)}\right).
\end{aligned}$$

The proof of the theorem is in the appendix. It involves finding a lower bound for each loss term in the maximum function, symmetry arguments, and using properties of indicator variables. It is broken down in 5 subsections in the proof to facilitate the reader's understanding.

We have a similar bound for the l_{0-1} loss function,

Theorem 2 *If a function $l(.)$ satisfies $l(z) \geq \mathbb{1}_{z\geq 0} + \mathbb{1}_{z\geq 1}$, then we have*

$$\begin{aligned}
&\mathbb{E}_{X\sim\mu_{X|T},Y^T\sim\mu_{Y^T|X},Y^C\sim\mu_{Y^C|X}} l_{0-1}(X,Y^T,Y^C,h) \\
&\leq \max\left(\mathbb{E}_{X\sim\mu_{X|T},Y^T\sim\mu_{Y^T|X}} l(-h(X)Y^T), \mathbb{E}_{X\sim\mu_{X|C},Y^C\sim\mu_{Y^C|X}} \frac{l(h(X)Y^C)}{\mu_{X|C}(X)/\mu_{X|T}(X)}\right).
\end{aligned}$$

The proof of this theorem in [32] is similar to that of Theorem 1.

The subscript of the expectation includes the generative model for the data. Here, $\mathbb{E}_{X\sim\mu_X, Y^T\sim\mu_{Y^T|X}, Y^C\sim\mu_{Y^C|X}}$ means that, first, X follows distribution μ_X, then Y^T is drawn from distribution $\mu_{Y^T|X}$, and Y^C is drawn from distribution $\mu_{Y^C|X}$.

The significance of this inequality is that the quantity on the right can be estimated using empirical averages, without imputation for counterfactuals.

The right-hand sides of the theorems are our surrogate loss functions. Since the max of the two terms is less than or equal to their sum, our surrogate losses are strictly tighter than using the sum of treatment and control losses. That sum would lead to separate modeling for treatment and control groups.

One corollary of the theorem is a remark on the importance of accurate density ratio estimation. The following corollary shows that in some cases, it is not crucial to obtain an accurate estimate of the density ratio.

Corollary 1 *If for all functions h,*

$$\begin{aligned} R(h) :=& \max\left(\mathbb{E}_{X\sim\mu_{X|T}, Y^T\sim\mu_{Y^T|X}} l(-h(X)Y^T), \mathbb{E}_{X\sim\mu_{X|C}, Y^C\sim\mu_{Y^C|X}} \frac{l(h(X)Y^C)}{\mu_{X|}(X)/\mu_{X|T}(X)}\right) \\ &= \mathbb{E}_{X\sim\mu_{X|T}, Y^T\sim\mu_{Y^T|X}} l(-h(X)Y^T), \ \textit{and also} \end{aligned}$$

$$\begin{aligned} \hat{R}(h) :=& \max\left(\mathbb{E}_{X\sim\mu_{X|T}, Y^T\sim\mu_{Y^T|X}} l(-h(X)Y^T), \mathbb{E}_{X\sim\mu_{X|C}, Y^C\sim\mu_{Y^C|X}} \frac{l(h(X)Y^C)}{\hat{\mu}_{X|C}(X)/\hat{\mu}_{X|T}(X)}\right) \\ &= \mathbb{E}_{X\sim\mu_{X|T}, Y^T\sim\mu_{Y^T|X}} l(-h(X)Y^T), \end{aligned}$$

then the minimizers $\min_h R(h)$ *and* $\min_h \hat{R}(h)$ *do not depend on how close the estimates* $\hat{\mu}_{X|C}(X)/\hat{\mu}_{X|T}(X)$ *are to the true ratios* $\mu_{X|C}(X)/\mu_{X|T}(X)$.

The corollary implies that flaws in density ratio estimation have no effect in a special case where the treatment loss is higher than the re-weighted control loss. This is different from using the sum of losses for treatment and control groups, where problems with density ratio estimation can affect the loss regardless of which achieves the max. Thus, our method is sometimes robust to poor density estimation methods,

and where it is not, the same problem is present in traditional methods; our method is no worse.

The sufficient condition to construct a surrogate upper bound for the 0-1 loss function is used in both theorems. It is:

$$l(z) \geq \mathbb{1}_{z \geq 0} + \mathbb{1}_{z \geq 1}.$$

This condition is easy to satisfy, and we list some valid losses below.

1. $\mathbb{1}_{z \geq 0} + \mathbb{1}_{z \geq 1}$, which is a type of 0-1 loss function. Clearly, this is trivially an upper bound, and it is non-smooth and it is difficult to optimize directly.

2. $\lfloor 1 + z \rfloor_+$, the hinge loss function. We will use this loss function to construct an SVM-based algorithm.

3. $(1 + z)^2$, the squared loss function.

4. $\frac{2\ln(1+e^z)}{\ln(1+e)}$, a scaled logistic loss function.

5. e^z, the exponential loss, used by AdaBoost.

4.4 Conditional Difference SVM

In this section, we use the regularized hinge loss to formulate a quadratic programming problem that is similar to classical SVM (except that it is for potential outcomes data where we have only "half" of the label for each observation).

For this section, we assume that the ratio $\mu_C(x_i)/\mu_T(x_i)$ is either known or has been estimated previously (or is irrelevant according to the theorems above). We will discuss this more later. These density ratios act as importance weights on the control group terms for *cost-sensitive* learning. The formulation below is kernelized, meaning that each x is replaced with a transformation $\phi(x)$, such that $\langle \phi(p), \phi(q) \rangle = K(p, q)$ can be evaluated efficiently as a kernel function. There are several standard conditions for k to be a valid kernel (an inner product of a Reproducing Kernel Hilbert space

– RKHS). Trivially, if ϕ is chosen to be the identity, then the kernel is linear. The model for $h(x)$ is $w_0 + \langle \phi(w), \phi(x) \rangle$.

The optimization problem suggested by Theorem 1, with the hinge loss as an upper bound on the 0-1 loss, is below. We added an RKHS norm regularization term with regularization parameter γ.

$$\begin{aligned} & R(w, w_0, \gamma) \\ & = \max \Big(\tfrac{1}{n^T} \textstyle\sum_{i \in T} \lfloor 1 - (w_0 + \langle \phi(w), \phi(x_i) \rangle) y_i^T \rfloor_+, \\ & \qquad \tfrac{1}{n^C} \textstyle\sum_{i \in C} \frac{\lfloor 1 + (w_0 + \langle \phi(w), \phi(x_i) \rangle) y_i^C \rfloor_+}{\mu_{X|C}(x_i)/\mu_{X|T}(x_i)} \Big) \\ & \qquad + \gamma \langle \phi(w), \phi(w) \rangle. \end{aligned}$$

Rewriting the inner product as a kernel, this is equivalent to:

$$\begin{aligned} & R(w, w_0, \gamma) \\ & = \max \left(\tfrac{1}{n^T} \textstyle\sum_{i \in T} \left\lfloor 1 - (w_0 + K(w, x_i)) y_i^T \right\rfloor_+, \right. \\ & \left. \tfrac{1}{n^C} \textstyle\sum_{i \in C} \frac{\left\lfloor 1 + (w_0 + K(w, x_i)) y_i^C \right\rfloor_+}{\mu_{X|C}(x_i)/\mu_{X|T}(x_i)} \right) + \gamma K(w, w). \end{aligned}$$

This minimax problem can be reformulated as a constrained optimization problem as follows:

Primal Problem:

$$\begin{aligned} & \min_{w, w_0, z, r, \forall i \; s_i, \forall i \; r_i} z + \gamma K(w, w) \quad \text{subject to} \\ z & \geq \frac{1}{n^T} \sum_{i \in T} r_i \\ z & \geq \frac{1}{n^C} \sum_{i \in C} \frac{s_i}{\mu_{X|C}(x_i)/\mu_{X|T}(x_i)} \\ r_i & \geq 1 - (w_0 + K(w, x_i)) y_i^T, \forall i \in T \\ s_i & \geq 1 + (w_0 + K(w, x_i)) y_i^C, \forall i \in C \\ r_i & \geq 0, \forall i \in T \\ s_i & \geq 0, \forall i \in C. \end{aligned}$$

We let K^* be the Gram matrix where $K^*(i,j) = K(x_i, x_j)$ where we order the vectors such that $x_1^T, \ldots, x_{n_T}^T, x_1^C, \ldots, x_{n_C}^C$.

The corresponding dual optimization problem is as follows.

Dual Problem:

$$\max_{\alpha,\beta,\{\lambda_i\}_i,\{\eta_i\}_i} -\frac{1}{4\gamma} \begin{bmatrix} \lambda \\ \eta \end{bmatrix}^T diag(y_1^T, \ldots, y_{n^T}^T, -y_1^C, \ldots, -y_{n^C}^C) K^* \\ diag(y_1^T, \ldots, y_{n^T}^T, -y_1^C, \ldots, -y_{n^C}^C) \begin{bmatrix} \lambda \\ \eta \end{bmatrix} + \sum_{i \in T} \lambda_i + \sum_{i \in C} \eta_i,$$

subject to

$$\begin{aligned} \alpha + \beta &= 1 \\ \forall i \in T, \quad 0 \leq \lambda_i &\leq \tfrac{1}{n^T}\alpha \\ \forall i \in C, \quad 0 \leq \eta_i &\leq \tfrac{1}{n^C(\mu_{X|C}(x_i)/\mu_{X|T}(x_i))}\beta \\ \sum_{i \in T} \lambda_i y_i^T &= \sum_{i \in C} \eta_i y_i^C \\ \alpha, \beta, \lambda, \eta &\geq 0 \end{aligned}$$

which is a quadratic programming problem that resembles the regular SVM problem. Its computational scaling properties are essentially identical to standard SVM.

Recovering the Intercept w_0

After solving for λ and η, we are able to theoretically recover an expression for $\phi(w)$ in the primal formulation that can be used to obtain values of $K(w, x)$ for any given x. To make prediction possible, we need to evaluate $h(x)$ for any x, thus we need to recover w_0, the intercept term. The complementary slackness conditions are

as follows:

$$\begin{aligned}
\lambda_i(r_i - 1 + (w_0 + K(w, x_i^T))y_i^T) &= 0 \ \ \forall i \in T \\
\eta_i(s_i - 1 - (w_0 + K(w, x_i^C))y_i^C) &= 0 \ \ \forall i \in C \\
r_i \left(\frac{\alpha}{n^T} - \lambda_i\right) &= 0 \ \ \forall i \in T \\
s_i \left(\frac{\beta}{n^C(\mu_{X|C}(x_i)/\mu_{X|T}(x_i))} - \eta_i\right) &= 0 \ \ \forall i \in C.
\end{aligned}$$

By solving the dual optimization problem, we know the value of $\{\lambda_i\}_i, \{\eta_i\}_i, \alpha, \beta$. We can use these to analytically recover w_0 from the primal problem using one of the "support vectors." Support vectors are data points that determine the separating hyperplane in regular SVM. In our context, we are minimizing the maximum of two hinge losses, and the maximum value will be attained by at least one of the control or treatment group. The hyperplane is chosen such that it minimizes loss (and maximizes the margin) in one of those groups, and since the larger of the two losses is being minimized, the loss in the other group will be upper bounded as well. Similar to the regular SVM, the points that fully determine the positions of the hyperplane (support vectors — SV's) are those with active constraints in the primal formulation. Figure 4-1 shows the support vectors for the Causal SVM on the spiral dataset discussed below. SV's from both treatment and control points can be present simultaneously. As usual, as long as the problem is not ill-conditioned (meaning at least one λ_i is between 0 and α/n^T, or at least one η_i is between 0 and β/n^C), we are able to recover the primal solution from the dual solution as follows: for $i \in T$ if $\lambda_i < \frac{\alpha}{n^T}$, we can conclude that $r_i = 0$ and similarly if $\lambda_i > 0$, we can conclude that $-1 + (w_0 + K(w, x_i))y_i^T = 0$. Using that y_i is binary:

$$w_0 = y_i^T - K(w, x_i^T).$$

Similarly, for $i \in C$ if $\eta_i < \frac{\beta}{n^C(\mu_{X|C}(x_i)/\mu_{X|T}(x_i))}$, we conclude that $s_i = 0$, and if for the same $i \in C$, $\eta_i > 0$, we have $w_0 = -y_i^C - K(w, x_i^C)$. Also, using optimization methods that use a primal dual approach, it is possible to obtain w_0 numerically.

Let us switch gears to discuss learning theory bounds.

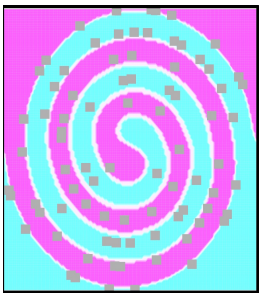


Figure 4-1: Causal SVM with RBF kernel on spiral data. The circular points are the support vectors, pink indicates predictions of positive treatment effect, and blue indicates negative predictions.

4.5 Generalization Bound

The bound in this section provides a theoretical foundation for minimizing the maximum of treatment and control empirical errors.

Definition 1 *Growth Function: [12] Let $\mathcal{F}$ be a function class (also known as hypothesis class). Given data points $z_1, \ldots, z_m$, we consider $\mathcal{F}_{z_1,\ldots,z_m} = \{f(z_1), \ldots, f(z_m)\}$, the set of ways the data $z_1, \ldots, z_m$ are classified by functions from $\mathcal{F}$. The growth function is the maximum number of ways into which m points can be classified by the function class. $S_{\mathcal{F}}(m) = \sup_{(z_1,\ldots,z_m)} |\mathcal{F}_{z_1,\ldots,z_m}|$.*

Let $R^{true}(f) = \mathbb{P}_{(x,y)\sim D}(f(X) \neq Y) = \mathbb{E}_{(X,Y)\sim D}[\mathbb{1}_{f(X)\neq Y}]$ and $R^{emp}(f) = \frac{1}{m}\sum_{i=1}^{m} \mathbb{1}_{[f(x_i)\neq y_i]}$. Using the Hoeffding and union bounds, a classical result shows that for any $\delta > 0$, with probability at least $1-\delta$ with respect to a random draw of the data,

$$\forall f \in \mathcal{F}, R^{true}(f) \leq R^{emp}(f) + 2\sqrt{2\frac{\log S_{\mathcal{F}}(2m) + \log\frac{4}{\delta}}{m}}.$$

We will derive an analogous bound for the causal inference estimation framework. Since we deal with both treatment and control groups, we need to handle weighted data points with Radon-Nikodym derivatives. More definitions follow.

Suppose $M = \sup_x l(h(x))$. We define a new loss function $l^M(.) = \frac{1}{M} l(.)$.

$$R_T(h) = \mathbb{E}_{X\sim\mu_{X|T}, Y^T\sim\mu_{Y^T|X}} l^M(-h(X)Y^T).$$

The corresponding empirical estimator for the expectation above would be

$$\hat{R}_T(h) = \frac{1}{n_T} \sum_{i \in T} l^M(-h(x_i)Y^T).$$

For the control group, we have

$$R_C(h) = \mathbb{E}_{X \sim \mu_{X|C}, Y^C \sim \mu_{Y^C|X}} l^M(-h(X)Y^C).$$

Using estimation of $\mu_{X|T}$ and $\mu_{X|C}$, the corresponding empirical estimator would be:

$$\hat{R}_C(h) = \frac{1}{n_C} \sum_{i \in C} \frac{\mu_{X|T}(x_i)}{\mu_{X|C}(x_i)} l^M(-h(x_i)Y^C).$$

Theorem 3 *Let $\mathcal{F}$ be a function class, and suppose we have n data points, let $p = Pdim(\mathcal{F})$, the standard pseudo-dimension of $\mathcal{F}$ (see [32] for precise definition). Let*

$$\Delta_T(\delta) = 2\sqrt{2\frac{\log S_{\mathcal{F}}(2n_T) + \log \frac{4}{\delta}}{n_T}} \text{ and}$$

$$\Delta_C(\delta) = 2^{\frac{5}{4}} \sqrt{d_2(\mu_T||\mu_C)} \sqrt[3/8]{\frac{p \log \frac{2n_C e}{p} + \log \frac{4}{\delta}}{n_C}}.$$

Then $\forall h \in \mathcal{F}$, with probability at least $1 - \delta$,

$$\begin{aligned} &\mathbb{E}_{X \sim \mu_X, Y^T \sim \mu_{Y^T|X}, Y^C \sim \mu_{Y^C|X}} l_1(X, Y^T, Y^C) \\ &\leq M \left(\max(\hat{R}_T(h), \hat{R}_C(h)) + \max \left(\Delta_T \left(\frac{\delta}{2} \right), \Delta_C \left(\frac{\delta}{2} \right) \right) \right). \end{aligned}$$

In the theorem, $d_2(P||Q) = 2^{D_{KL}(P||Q)}$ where $D_{KL}(P||Q)$ is the usual KL divergence between distributions P and Q.

Proof of the generalization bound is provided in the appendix. As usual, the bound is algorithm independent.

4.6 Experiments

We cannot observe both treatment and control outcomes for the same observation in real data (this is not standard supervised learning), so ground truth treatment effects must be obtained another way for the purpose of evaluation. In these experiments, the goal is to test the most basic potential outcomes setting. We randomly assign observations to either the treatment group or control group. We choose distributions μ_T and μ_C for generating the x_i's, and choose distributions to generate both potential outcomes y_i^T and y_i^C for each i, $y_i^T, y_i^C \in \{-1, 1\}$. We observe either y_i^T or y_i^C, depending on whether the observation is in the treatment or control group. The treatment effect $y_i^T - y_i^C$ is thus never observed for any i, and takes three possible values: positive, neutral, or negative. We then split the data uniformly into a training set and a test set. The training data were used to build a model that predicts conditional treatment effect given a new test point x. We then predict treatment effects for the test data and evaluate our predictions with respect to the ground truth using the conditional difference loss as a performance measure.

For Causal SVM, we use linear, quadratic, cubic, and radial basis function (RBF) kernels. We compare with matching-based algorithms, such as GenMatch [26] and nearest neighbor matching, followed by ridge regression or kernel ridge regression on the matched groups to create a predictive model. We also compare with algorithms that fit two distinct classification or regression models and take the difference; this includes the difference of ridge regression models, difference of kernel ridge regression models, difference of logistic regression models, difference of SVM models using RBF kernels, and difference of random forests. Note that the methods where two regression are fitted for different groups are commonly used in meta-algorithms [27, 48]. Also, we compare our algorithms with causal random forests [90]. For methods that involve Genmatch, the pop.size parameter was chosen as $n/2$, and after matching, cross validation was performed for tuning the regularization parameter for the regression methods. For methods involving the difference of two models, cross validation for parameter tuning was performed on the treatment and control data separately.

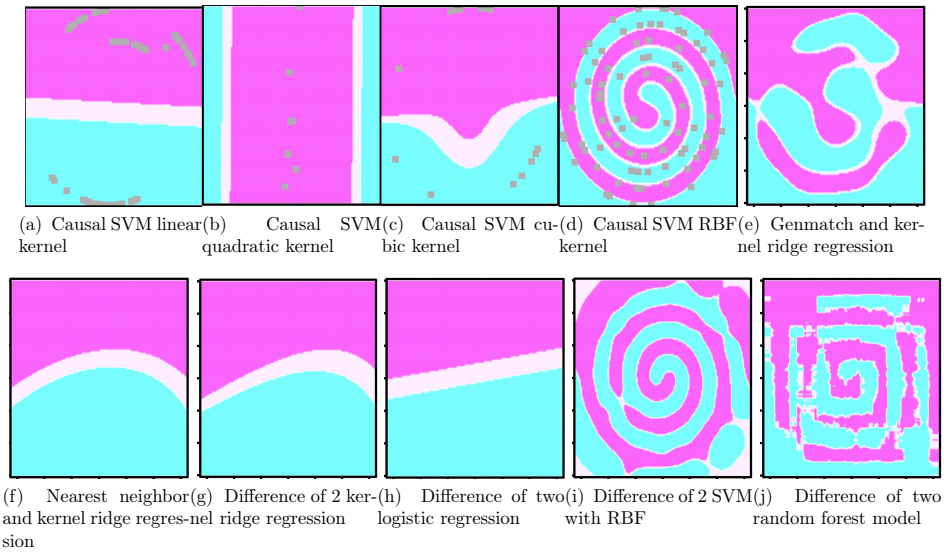
As discussed earlier, the difference of two distinct classification or regression models is similar to our approach but uses a looser upper bound to the 0-1 loss function: a sum of the terms for treatment and control (as obtained through the triangle inequality), rather than a maximum of the two terms. However, using a difference of two models would mean using a strictly richer family of functions to learn the treatment effect. Intuitively one might expect that difference of complex models (e.g. random forests or SVM) would potentially overfit. The bounds are loose enough that it is unclear why the sign of the difference of SVM models would necessarily produce useful models, as usually the sign of each single SVM model is used for predicting outcomes.

Our results for each dataset are reported in 2-column tables. The column heading is the value of θ used in the loss, where θ is the fraction of data predicted to be neutral. For example when θ=0.1, the 10% of data with smallest absolute predicted difference are assigned to be neutral. The first 15 rows of each table are the output of our Causal SVM algorithm. For these methods, the number appended at the end (e.g., 1e-8) indicates the parameter γ used. For the RBF kernel, the other number is the inverse kernel width. These are followed by matching based methods, difference of two supervised learning methods, and causal random forests. The two numbers for the causal random forest methods are the α and λ parameters in that algorithm. The mean and the standard deviation (in braces) are reported in the table. *The superscript index indicates the rank of the algorithm for the top algorithms.*

Noisy Spirals

We show the result of a simple but challenging experiment on a causal inference version of the two spiral dataset from [49], which has two covariates. The goal is to predict a positive treatment effect on one spiral and a negative treatment effect on the other spiral. Half of the training points were randomly assigned to be treated and the other half assigned to control. On one of the two spirals, the treatment effect is positive (treated points had outcome "yes" and control points had outcome "no"), whereas on the other spiral the treatment effect is negative (treated points had outcome "no", and control points had outcome "yes"). We introduce label noise: with probability 20%, a data point that should have a positive treatment effect is assigned

a negative treatment effect and vice versa. We fit models on the training data, and predicted on out-of-sample test data. Ideally, all points from one spiral should have "yes" predictions and the other should have all "no" predictions.



(a) Causal SVM linear kernel (b) Causal SVM quadratic kernel (c) Causal SVM cubic kernel (d) Causal SVM RBF kernel (e) Genmatch and kernel ridge regression

(f) Nearest neighbor and kernel ridge regression (g) Difference of 2 kernel ridge regression (h) Difference of two logistic regression (i) Difference of 2 SVM with RBF (j) Difference of two random forest model

Figure 4-2: Contour plots showing the predicted treatment effect for spiral data with 20% label noise. Support vectors are noted with gray dots for the SVM models.

As we can see from Table 4.1, the linear, quadratic, and cubic methods all perform poorly, because spirals cannot be modeled accurately using linear models or low dimensional polynomials; this would have been clear before performing the experiment, but provides useful baselines. The best performers are RBF SVM models and difference of 2 random forests. The matching methods GenMatch and nearest neighbor seem to have consistently poor performance, as does causal random forests. In fact the performance of these methods is as bad as results obtained from modeling the spirals with linear models. Figure 4-2 shows models from several machine learning methods.

More experimental results are in the supplementary materials.

θ	0.01	0.1
linear Causal SVM 1e-8	57.91(1.94)	52.21(1.91)
linear Causal SVM 1e-6	57.91(1.94)	52.21(1.91)
linear Causal SVM 1e-4	57.91(1.94)	52.21(1.91)
quadratic Causal SVM 1e-8	58.77(0.95)	53.51(0.86)
quadratic Causal SVM 1e-6	58.77(0.95)	53.51(0.86)
quadratic Causal SVM 1e-4	58.81(1)	53.52(0.87)
cubic Causal SVM 1e-8	56.02(1.72)	50.51(1.28)
cubic Causal SVM 1e-6	56.02(1.72)	50.51(1.28)
cubic Causal SVM 1e-4	56.23(1.81)	50.42(1.25)
RBF Causal SVM 0.05, 1e-8	41.4(2.63)	36.53(2.46)
RBF Causal SVM 0.05, 1e-6	45.52(2.02)	40.65(1.75)
RBF Causal SVM 0.05, 1e-4	55.57(1.7)	50.28(1.17)
RBF Causal SVM 0.1, 1e-8	$23.57(0.99)^2$	$19.45(0.88)^2$
RBF Causal SVM 0.1, 1e-6	41.81(2.15)	37.03(2.13)
RBF Causal SVM 0.1, 1e-4	52.47(1.2)	47.16(1.24)
GenMatch, Ridge	57.44(1.11)	52.31(1.1)
Nearest, Ridge	56.94(1.57)	51.8(0.93)
Genmatch, kernel ridge	51.7(3.41)	46.9(3.28)
Nearest, kernel ridge	52.64(3.2)	47.27(2.96)
2 ridge	58.12(1.25)	52.52(1.09)
2 kernel ridge	53.04(3.03)	48.03(2.65)
2 logistic	58.26(0.98)	52.72(0.92)
2 SVM	$19.91(0.85)^1$	$17.17(1.17)^1$
2 RF	$25.61(1.32)^3$	$21.29(1.11)^3$
causal_rf 0.05 0	46.61(1.8)	41.5(1.6)
causal_rf 0.01 0	55.37(1.65)	50.14(1.86)
causal_rf 0.05 0.1	46.67(1.54)	41.64(1.79)
causal_rf 0.01 0.1	54.63(2.22)	49.43(2.05)

Table 4.1: Loss values $l_{.01}$ and $l_{.1}$ for spiral data with noise. The best three performers in each column are indicated with superscripts 1, 2 and 3.

4.7 Breaking the Cycle of Drugs and Crime

Next, we apply our method to data from a social program in the United States, known as Breaking the Cycle (BTC)[34], which studies the effect of intervention on the reduction of crime and drug use. These data were chosen for their relevance to treatment programs for the current opioid epidemic in the U.S. As far as we know, these data have not been previously studied using machine learning techniques. We focus on estimating the effect of the program on reducing non-drug-related crime in Birmingham, Alabama, between years 1997 and 2001.The BTC strategy was to

screen offenders shortly after arrest and require those found to use drugs to participate in a drug intervention while under criminal justice supervision. The control group consisted of similar defendants arrested in the year prior to the implementation of BTC. BTC targeted all adult felony defendants and was not limited to those charged with drug offenses. Defendants were ordered to report to BTC for drug screening as a condition of pretrial release. Those who reported drug use, tested positive for drugs, or were arrested on drug felony charges were placed in drug testing and, when appropriate, referred to drug treatment or drug education classes.

We chose categorical features whose data seemed reliable, that had no missing values, and that had a correlation with the outcome of non-drug-related crime of at least 0.1. We did not use data recorded during the time period over which the outcome was generated, as we intended to build a prediction model for the outcome during that same time period. The features include: whether the defendant has a driver's license, whether the defendant has access to an automobile, whether an SSI benefit is being received, whether the defendant lives with anyone with an alcohol problem or takes nonprescription drugs, whether the defendant has problems getting along with their father, whether they have suffered for depression within the past 30 days, whether they have had depression or anxiety for a long period of time, and whether they have trouble understanding. In this dataset, some participants were subsequently dropped from the study as they were later determined to be ineligible, leaving us with 382 participants.

Our algorithm requires a choice of regularization parameter and kernel parameter. In regular supervised learning, nested cross validation would be the natural method to tune parameters. Typically for causal inference applications, since ground truth is not known, parameters cannot be tuned. In the case of our method, the objective function does *not* require the ground truth to be known, hence, we *can* perform nested cross validation to select parameters using our objective function.

After running our algorithm, we wish to examine the results by determining which subgroups benefit from the BTC program. Grouping together the observations with neutral and negative estimated treatment effect, to distinguish them from the obser-

vations for which the treatment was effective, we used interpretable modeling methods to understand the result.

We generated association rules using the Apriori algorithm [33]. The rules indicate that those with access to an automobile seem to benefit from the BTC program in terms of reduction of non-drug related crime. (Having a license could be an indicator of competence of several forms.) We list some of the rules in Table 4.2.

Antecedent	Effective?	Support	Confidence	Lift
have-automobile=1, prob-getting-along-father=0, serious-depression-30-days=0	Y	0.2173	1.0000	2.2209
have-license=1, serious-depression-30-days=0, serious-depression-life=0	Y	0.2539	0.9899	2.1983
have-license=1, prob-getting-along-father=0, serious-depression-30-days=0	Y	0.2539	0.9898	2.1983
have-automobile=1, serious-depression-30-days=0, serious-depression-life=0	Y	0.2173	0.9881	2.1945
have-license=0, SSI-benefit=0, prob-getting-along-father=0	N	0.4424	0.9037	1.6440
have-license=0, prob-getting-along-father=0, trouble-understanding-life=0	N	0.3953	0.8935	1.6253
have-license=0, prob-getting-along-father=0, serious-depression-life=1	N	0.1414	0.8852	1.6103
have-license=0, live-w-anyone-alcohol=0, prob-getting-along-father=0	N	0.4398	0.8660	1.5753
have-license=0, SSI-benefit=0, trouble-understanding-life=0	N	0.4031	0.8652	1.5738

Table 4.2: Association rules for the estimated (Causal SVM) treatment effect of the BTC program on the reduction of non-drug related offenses.

We then created a one-sided decision tree (decision list, or rule list) as an interpretable approximation to the Causal SVM output. We used the CORELS algorithm to produce this rule list [6], which also certified that this model is optimal according to the objective of accuracy and sparsity on the training set. The rule list is below.

```
if (have_drivers_license) then (effective) (87%/13%)
else if (long_term_serious_depression) then (not_effective) (20%/80%)
else if (long_term_trouble_understanding) then (effective) (92%/8%)
else if (SSI_benefit) then (effective) (100%/0%)
else if (prob_getting_along_with_father) then (effective) (91%/9%)
else (not_effective) (3%/97%)
```

Let us explain the expressions such as (87%/13%) on the right of each rule: in the first rule, 87% of observations captured by this rule have an estimated positive treatment effect from Causal SVM. 13% is the percentage of units captured with a negative or neutral estimated treatment effect.

4.8 Discussion

We presented a framework for estimating personalized treatment effects with theoretically appealing properties. Its surrogate loss bound is tighter than the sum of losses for treatment and control groups. Since it uses global convex optimization, it is easier to troubleshoot and tune than methods that involve greedy splitting, pruning, and averaging (e.g., random forests). Its high-quality experimental results seem to be robust to different datasets, unlike several other methods, meaning that it might be more more trustworthy across domains. Our experiments indicate that it could be useful to include the Causal SVM algorithm in experimental studies, in addition to the algorithms based on separate treatment and control models. The principles used to derive the Causal SVM framework are its surrogate loss definition and bounds, which are of independent interest for other causal inference problems. The generalization bounds are algorithm-independent, and can be applied to any surrogate for the minimax conditional difference loss introduced in this work.

Code: < `https://github.com/shangtai/githubcausalsvm`>.

Appendix for Chapter 4

Proof of Theorem 1

Proof 1 *(Of Theorem 1). We break down the proofs into five steps for readability.*

1. Obtaining lower bounds for $\mathbb{E}_{X\sim\mu_{X|T},Y^T\sim\mu_{Y^T|X}} l(-h(X)Y^T)$ and $\mathbb{E}_{X\sim\mu_{X|C},Y^C\sim\mu_{Y^C|X}} \frac{l(h(X)Y^C)}{\mu_{X|C}(X)/\mu_{X|T}(X)}$.

$$
\begin{aligned}
&\mathbb{E}_{X\sim\mu_{X|T},Y^T\sim\mu_{Y^T|X},Y^C\sim\mu_{Y^C|X}} l(-h(X)Y^T) \\
&= \mathbb{E}_{X\sim\mu_{X|T},Y^T\sim\mu_{Y^T|X},Y^C\sim\mu_{Y^C|X}:Y^T>Y^C} l(-h(X)Y^T) \\
&+ \mathbb{E}_{X\sim\mu_{X|T},Y^T\sim\mu_{Y^T|X},Y^C\sim\mu_{Y^C|X}:Y^T<Y^C} l(-h(X)Y^T) \\
&+ \mathbb{E}_{X\sim\mu_{X|T},Y^T\sim\mu_{Y^T|X},Y^C\sim\mu_{Y^C|X}:Y^T=Y^C} l(-h(X)Y^T) \\
&= \mathbb{E}_{X\sim\mu_{X|T},Y^T\sim\mu_{Y^T|X},Y^C\sim\mu_{Y^C|X}:Y^T>Y^C} l(-h(X)) \\
&+ \mathbb{E}_{X\sim\mu_{X|T},Y^T\sim\mu_{Y^T|X},Y^C\sim\mu_{Y^C|X}:Y^T<Y^C} l(h(X)) \\
&+ \mathbb{E}_{X\sim\mu_{X|T},Y^T\sim\mu_{Y^T|X},Y^C\sim\mu_{Y^C|X}:Y^T=Y^C} l(-h(X)Y^T) \\
&\geq \mathbb{E}_{X\sim\mu_{X|T},Y^T\sim\mu_{Y^T|X},Y^C\sim\mu_{Y^C|X}:Y^T>Y^C} \mathbb{1}_{h(X)\leq 0} \\
&+ \mathbb{E}_{X\sim\mu_{X|T},Y^T\sim\mu_{Y^T|X},Y^C\sim\mu_{Y^C|X}:Y^T<Y^C} \mathbb{1}_{h(X)\geq 0} \\
&+ \mathbb{E}_{X\sim\mu_{X|T},Y^T\sim\mu_{Y^T|X},Y^C\sim\mu_{Y^C|X}:Y^T=Y^C} l(-h(X)Y^T),
\end{aligned}
$$

where for the second equation, the reasoning is if $Y^T > Y^C$ then $Y^T = 1$, and if $Y^T < Y^C$ then $Y^T = -1$. The last inequality makes use of the property that if $h(X) \leq 0$, then $l(-h(X)) \geq 1$. Similarly, if $h(X) \geq 0$, then $l(h(X)) \geq 1$.

By similar reasoning, we have

$$
\begin{aligned}
&\mathbb{E}_{X\sim\mu_{X|C},Y^C\sim\mu_{Y^C|X}}\frac{l(h(X)Y^C)}{\mu_{X|C}(X)/\mu_{X|T}(X)}\\
&=\mathbb{E}_{X\sim\mu_{X|T},Y^T\sim\mu_{Y^T|X},Y^C\sim\mu_{Y^C|X}}l(h(X)Y^C)\\
&=\mathbb{E}_{X\sim\mu_{X|T},Y^T\sim\mu_{Y^T|X},Y^C\sim\mu_{Y^C|X}:Y^T>Y^C}l(h(X)Y^C)\\
&+\mathbb{E}_{X\sim\mu_{X|T},Y^T\sim\mu_{Y^T|X},Y^C\sim\mu_{Y^C|X}:Y^T<Y^C}l(h(X)Y^C)\\
&+\mathbb{E}_{X\sim\mu_{X|T},Y^T\sim\mu_{Y^T|X},Y^C\sim\mu_{Y^C|X}:Y^T=Y^C}l(h(X)Y^C)\\
&=\mathbb{E}_{X\sim\mu_{X|T},Y^T\sim\mu_{Y^T|X},Y^C\sim\mu_{Y^C|X}:Y^T>Y^C}l(-h(X))\\
&+\mathbb{E}_{X\sim\mu_{X|T},Y^T\sim\mu_{Y^T|X},Y^C\sim\mu_{Y^C|X}:Y^T<Y^C}l(h(X))\\
&+\mathbb{E}_{X\sim\mu_{X|T},Y^T\sim\mu_{Y^T|X},Y^C\sim\mu_{Y^C|X}:Y^T=Y^C}l(h(X)Y^C)\\
&\geq\mathbb{E}_{X\sim\mu_{X|T},Y^T\sim\mu_{Y^T|X},Y^C\sim\mu_{Y^C|X}:Y^T>Y^C}\mathbb{1}_{h(X)\leq 0}\\
&+\mathbb{E}_{X\sim\mu_{X|T},Y^T\sim\mu_{Y^T|X},Y^C\sim\mu_{Y^C|X}:Y^T<Y^C}\mathbb{1}_{h(X)\geq 0}\\
&+\mathbb{E}_{X\sim\mu_{X|T},Y^T\sim\mu_{Y^T|X},Y^C\sim\mu_{Y^C|X}:Y^T=Y^C}l(h(X)Y^C).
\end{aligned}
$$

2. Finding a lower bound for $\max\Big(\mathbb{E}_{X\sim\mu_{X|T},Y^T\sim\mu_{Y^T|X}} l(-h(X)Y^T),$ $\mathbb{E}_{X\sim\mu_{X|C},Y^C\sim\mu_{Y^C|X}}\frac{l(h(X)Y^C)}{\mu_C(X)/\mu_T(X)}\Big)$

We use the property that $a \geq b$ *and* $c \geq d$ *imply* $\max(a,c) \geq \max(b,d)$*. Taking the maximum of the previous two inequalities in the previous part of the proof, we have*

$$
\begin{aligned}
&\max\left(\mathbb{E}_{X\sim\mu_{X|T},Y^T\sim\mu_{Y^T|X}} l(-h(X)Y^T), \mathbb{E}_{X\sim\mu_{X|C},Y^C\sim\mu_{Y^C|X}}\frac{l(h(X)Y^C)}{\mu_{X|C}(X)/\mu_{X|T}(X)}\right)\\
&\geq \max\Big(\mathbb{E}_{X\sim\mu_{X|T},Y^T\sim\mu_{Y^T|X},Y^C\sim\mu_{Y^C|X}:Y^T>Y^C}\mathbb{1}_{h(X)\leq 0}\\
&+\mathbb{E}_{X\sim\mu_{X|T},Y^T\sim\mu_{Y^T|X},Y^C\sim\mu_{Y^C|X}:Y^T<Y^C}\mathbb{1}_{h(X)\geq 0}\\
&+\mathbb{E}_{X\sim\mu_{X|T},Y^T\sim\mu_{Y^T|X},Y^C\sim\mu_{Y^C|X}:Y^T=Y^C} l(-h(X)Y^T)\\
&,\mathbb{E}_{X\sim\mu_{X|T},Y^T\sim\mu_{Y^T|X},Y^C\sim\mu_{Y^C|X}:Y^T>Y^C}\mathbb{1}_{h(X)\leq 0}\\
&+\mathbb{E}_{X\sim\mu_{X|T},Y^T\sim\mu_{Y^T|X},Y^C\sim\mu_{Y^C|X}:Y^T<Y^C}\mathbb{1}_{h(X)\geq 0}\\
&+\mathbb{E}_{X\sim\mu_{X|T},Y^T\sim\mu_{Y^T|X},Y^C\sim\mu_{Y^C|X}:Y^T=Y^C} l(h(X)Y^C)\Big)\\
&\geq \mathbb{E}_{X\sim\mu_{X|T},Y^T\sim\mu_{Y^T|X},Y^C\sim\mu_{Y^C|X}:Y^T>Y^C}\mathbb{1}_{h(X)\leq 0}\\
&+\mathbb{E}_{X\sim\mu_{X|T},Y^T\sim\mu_{Y^T|X},Y^C\sim\mu_{Y^C|X}:Y^T<Y^C}\mathbb{1}_{h(X)\geq 0}\\
&+\max\Big(\mathbb{E}_{X\sim\mu_{X|T},Y^T\sim\mu_{Y^T|X},Y^C\sim\mu_{Y^C|X}:Y^T=Y^C} l(-h(X)Y^T),\\
&\mathbb{E}_{X\sim\mu_{X|T},Y^T\sim\mu_{Y^T|X},Y^C\sim\mu_{Y^C|X}:Y^T=Y^C} l(h(X)Y^C)\Big),
\end{aligned}
$$

where the second inequality is because the first two terms within the maximum are exactly the same.

For the next part, we focus on

$$
\begin{aligned}
&\max\Big(\mathbb{E}_{X\sim\mu_{X|T},Y^T\sim\mu_{Y^T|X},Y^C\sim\mu_{Y^C|X}:Y^T=Y^C} l(-h(X)Y^T),\\
&\mathbb{E}_{X\sim\mu_{X|T},Y^T\sim\mu_{Y^T|X},Y^C\sim\mu_{Y^C|X}:Y^T=Y^C} l(h(X)Y^C)\Big),
\end{aligned}
$$

** that is the case where* $Y^T = Y^C$*.*

3. Lower bounds for $\mathbb{E}_{X\sim\mu_{X|T},Y^T\sim\mu_{Y^T|X},Y^C\sim\mu_{Y^C|X}:Y^T=Y^C} l(-h(X)Y^T)$ and $\mathbb{E}_{X\sim\mu_{X|T},Y^T\sim\mu_{Y^T|X},Y^C\sim\mu_{Y^C|X}:Y^T=Y^C} l(h(X)Y^C)$.

Since

$$l(-h(X)Y^T) \geq 0$$

because it is an upper bound for an indicator function, we have the following implications

$$Y^T = Y^C = -1 \text{ and } h(X) \geq 1 \implies l(-h(X)Y^T) = (l(h(X))) \geq 2\mathbb{1}_{\{h(x)\geq 1\}}$$
$$Y^T = Y^C = 1 \text{ and } h(X) \leq -1 \implies l(-h(X)Y^T) = (l(-h(X))) \geq 2\mathbb{1}_{\{h(x)\leq -1\}}.$$

We have

$$\begin{aligned}
&\mathbb{E}_{X\sim\mu_{X|T},Y^T\sim\mu_{Y^T|X},Y^C\sim\mu_{Y^C|X}:Y^T=Y^C} l(-h(X)Y^T) \\
&\geq 2\mathbb{P}_{X\sim\mu_{X|},Y^T\sim\mu_{Y^T|X},Y^C\sim\mu_{Y^C|X}}(Y^T = Y^C = -1, h(X) \geq 1) \\
&+ 2\mathbb{P}_{X\sim\mu_{X|T},Y^T\sim\mu_{Y^T|X},Y^C\sim\mu_{Y^C|X}}(Y^T = Y^C = 1, h(X) \leq -1).
\end{aligned}$$

We have a similar result for the control group,

$$\begin{aligned}
&\mathbb{E}_{X\sim\mu_{X|T},Y^T\sim\mu_{Y^T|X},Y^C\sim\mu_{Y^C|X}:Y^T=Y^C} l(h(X)Y^C) \\
&\geq 2\mathbb{P}_{X\sim\mu_{X|T},Y^T\sim\mu_{Y^T|X},Y^C\sim\mu_{Y^C|X}}(Y^T = Y^C = 1, h(X) \geq 1) \\
&+ 2\mathbb{P}_{X\sim\mu_{X|T},Y^T\sim\mu_{Y^T|X},Y^C\sim\mu_{Y^C|X}}(Y^T = Y^C = -1, h(X) \leq -1).
\end{aligned}$$

4. Lower Bound for the maximum between $\mathbb{E}_{X\sim\mu_{X|T},Y^T\sim\mu_{Y^T|X},Y^C\sim\mu_{Y^C|X}:Y^T=Y^C} l(-h(X)Y^T)$ and $\mathbb{E}_{X\sim\mu_{X|T},Y^T\sim\mu_{Y^T|X},Y^C\sim\mu_{Y^C|X}:Y^T=Y^C} l(h(X)Y^C)$

By using the fact that if $a \geq b$ and $c \geq d$, then we have $\max(a,c) \geq \max(b,d)$ and the result from the previous subsection, we have the first inequality below. The second

inequality below is due to $2max(a,b) \geq a+b$.

$$
\begin{aligned}
&\max(\mathbb{E}_{X\sim\mu_{X|T},Y^T\sim\mu_{Y^T|X},Y^C\sim\mu_{Y^C|X}:Y^T=Y^C} l(-h(X)Y^T),\\
&\mathbb{E}_{X\sim\mu_{X|T},Y^T\sim\mu_{Y^T|X},Y^C\sim\mu_{Y^C|X}:Y^T=Y^C} l(h(X)Y^C))\\
&\geq 2\max\Big(\mathbb{P}_{X\sim\mu_{X|T},Y^T\sim\mu_{Y^T|X},Y^C\sim\mu_{Y^C|X}}(Y^T=Y^C=-1,h(X)\leq 1)\\
&+\mathbb{P}_{X\sim\mu_{X|T},Y^T\sim\mu_{Y^T|X},Y^C\sim\mu_{Y^C|X}}(Y^T=Y^C=1,h(X)\leq -1),\\
&\mathbb{P}_{X\sim\mu_{X|T},Y^T\sim\mu_{Y^T|X},Y^C\sim\mu_{Y^C|X}}(Y^T=Y^C=1,h(X)\geq 1)\\
&+\mathbb{P}_{X\sim\mu_{X|T},Y^T\sim\mu_{Y^T|X},Y^C\sim\mu_{Y^C|X}}(Y^T=Y^C=-1,h(X)\leq -1)\ )\\
&\geq \mathbb{P}_{X\sim\mu_{X|T},Y^T\sim\mu_{Y^T|X},Y^C\sim\mu_{Y^C|X}}(Y^T=Y^C=-1,h(X)\geq 1)\\
&+\mathbb{P}_{X\sim\mu_{X|T},Y^T\sim\mu_{Y^T|X},Y^C\sim\mu_{Y^C|X}}(Y^T=Y^C=1,h(X)\leq -1)\\
&+\mathbb{P}_{X\sim\mu_{X|T},Y^T\sim\mu_{Y^T|X},Y^C\sim\mu_{Y^C|X}}(Y^T=Y^C=1,h(X)\geq 1)\\
&+\mathbb{P}_{X\sim\mu_{X|T},Y^T\sim\mu_{Y^T|X},Y^C\sim\mu_{Y^C|X}}(Y^T=Y^C=-1,h(X)\leq -1)\\
&=\mathbb{P}_{X\sim\mu_{X|T},Y^T\sim\mu_{Y^T|X},Y^C\sim\mu_{Y^C|X}}(Y^T=Y^C,|h(X)|\geq 1).
\end{aligned}
$$

5. Lower Bound for $\max(\mathbb{E}_{X\sim\mu_{X|T},Y^T\sim\mu_{Y^T|X}} l(-h(X)Y^T),$ $\mathbb{E}_{X\sim\mu_{X|C},Y^C\sim\mu_{Y^C|X}} \frac{l(h(X)Y^C)}{\mu_C(X)/\mu_T(X)})$

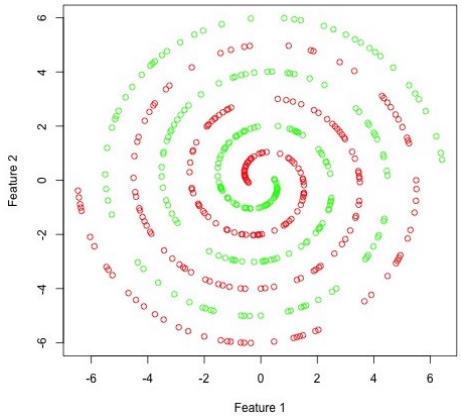
Combining the result from the seocnd step and fifth step, we have

$$
\begin{aligned}
&\max(\mathbb{E}_{X\sim\mu_{X|T},Y^T\sim\mu_{Y^T|X}} l(-h(X)Y^T), \mathbb{E}_{X\sim\mu_{X|C},Y^C\sim\mu_{Y^C|X}} \frac{l(h(X)Y^C)}{\mu_{X|C}(X)/\mu_{X|T}(X)} \\
&\geq \mathbb{E}_{X\sim\mu_{X|T},Y^T\sim\mu_{Y^T|X},Y^C\sim\mu_{Y^C|X}:Y^T>Y^C} \mathbb{1}_{h(X)\leq 0} \\
&+ \mathbb{E}_{X\sim\mu_{X|T},Y^T\sim\mu_{Y^T|X},Y^C\sim\mu_{Y^C|X}:Y^T<Y^C} \mathbb{1}_{h(X)\geq 0} \\
&+ \max\Big(\mathbb{E}_{X\sim\mu_{X|T},Y^T\sim\mu_{Y^T|X},Y^C\sim\mu_{Y^C|X}:Y^T=Y^C} l(-h(X)Y^T) \\
&, \mathbb{E}_{X\sim\mu_{X|T},Y^T\sim\mu_{Y^T|X},Y^C\sim\mu_{Y^C|X}:Y^T=Y^C} l(h(X)Y^C)\Big) \\
&\geq \mathbb{E}_{X\sim\mu_{X|T},Y^T\sim\mu_{Y^T|X},Y^C\sim\mu_{Y^C|X}:Y^T>Y^C} \mathbb{1}_{h(X)\leq 0} \\
&+ \mathbb{E}_{X\sim\mu_{X|T},Y^T\sim\mu_{Y^T|X},Y^C\sim\mu_{Y^C|X}:Y^T<Y^C} \mathbb{1}_{h(X)\geq 0} \\
&+ \mathbb{P}_{X\sim\mu_{X|T},Y^T\sim\mu_{Y^T|X},Y^C\sim\mu_{Y^C|X}}(Y^T = Y^C, |h(X)| \geq 1) \\
&= \mathbb{E}_{X\sim\mu_{X|T},Y^T\sim\mu_{Y^T|X},Y^C\sim\mu_{Y^C|X}:Y^T>Y^C} \mathbb{1}_{h(X)\leq 0} \\
&+ \mathbb{E}_{X\sim\mu_{X|T},Y^T\sim\mu_{Y^T|X},Y^C\sim\mu_{Y^C|X}:Y^T<Y^C} \mathbb{1}_{h(X)\geq 0} \\
&+ \mathbb{E}_{X\sim\mu_{X|T},Y^T\sim\mu_{Y^T|X},Y^C\sim\mu_{Y^C|X}:Y^T=Y^C} \mathbb{1}_{|h(X)|\geq 1} \\
&= \mathbb{E}_{X\sim\mu_{X|T},Y^T\sim\mu_{Y^T|X},Y^C\sim\mu_{Y^C|X}} l_{0-1}(X, Y^T, Y^C, h) \\
&\geq \mathbb{E}_{X\sim\mu_{X|T},Y^T\sim\mu_{Y^T|X},Y^C\sim\mu_{Y^C|X}} l_1(X, Y^T, Y^C, h).
\end{aligned}
$$

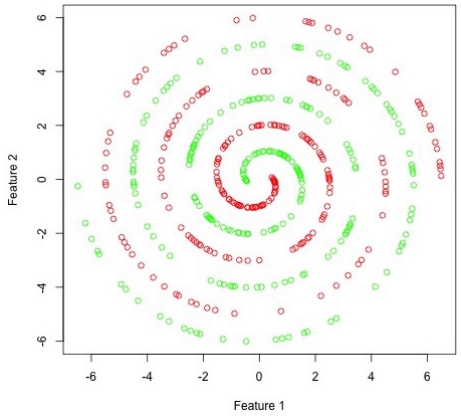
That is we have proven that the expectation of the loss function is upper bounded by

$$
\max\left(\mathbb{E}_{X\sim\mu_{X|T},Y^T\sim\mu_{Y^T|X}} l(-h(X)Y^T),\ \mathbb{E}_{X\sim\mu_{X|C},Y^C\sim\mu_{Y^C|X}} \frac{l(h(X)Y^C)}{\mu_{X|C}(X)/\mu_{X|T}(X)}\right).
$$

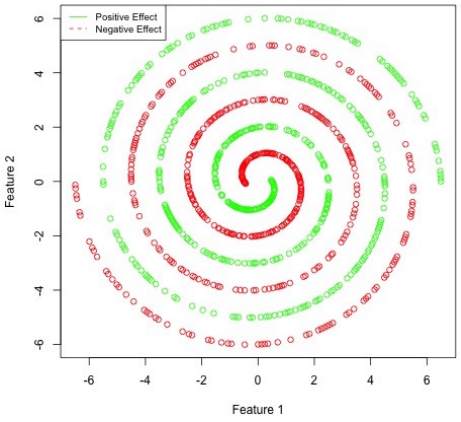
Remark: The proof of Theorem 2 is actually included where we stop just before the final inequality.



(a) Observed treatment output. The red data points indicate $y^T = 1$ while the green data points indicate $y^T = -1$.



(b) Observed control output. The red data points indicate $y^C = 1$ while the green data points indicate $y^C = -1$.



(c) The ground truth treatment effect (not observed). The green data points indicate positive treatment effect and the red data points indicate negative treatment effect.

Figure 4-3: The ground truth and the observed outcome for the spiral data set

Proof of Generalization Bound

From [12], we have $\forall \delta > 0$, with probability at least $1 - \delta$

$$\forall h \in \mathcal{F}, R_T(h) \leq \hat{R}_T(h) + \Delta_T(\delta),$$

where

$$\Delta_T(\delta) = 2\sqrt{2\frac{\log S_{\mathcal{F}}(2n_T) + \log \frac{4}{\delta}}{n_T}}.$$

Unlike conventional statistical learning bounds, recall that we are working with two different distributions, μ_T and μ_C of which we have chosen μ_T to be our target distribution. The use of Radon-Nikodym derivatives to transform μ_C to μ_T corresponds to importance weighting.

We will build our result on Theorem 3 in [21] which states the following:

Let F be a hypothesis set such that $Pdim\left(\{L_h(x) : h \in \mathcal{F}\}\right) = p < \infty$. Let X denote the input space and let Y be the label set. We let $L : Y \times Y \to [0, 1]$ be a loss function. We let $f : X \to Y$ be the target labeling function. We let $L_h(x)$ denote $L(h(x), f(x))$ in the absence of ambiguity about the target function f.

For any hypothesis $h \in \mathcal{F}$, we denote by $R(h)$ its loss and by $\hat{R}_w(h)$ its weighted empirical loss:

$$R(h) = \mathbb{E}_{x \sim P}[L(h(x)), f(x)]$$
$$\hat{R}_w(h) = \frac{1}{m}\sum_{i=1}^{m} w(x_i)L(h(x_i), f(x_i))$$

Assume that $d_2(P||Q) = 2^{D_{KL}(P||Q)} < +\infty$ and $w(x) \neq 0$ for all x. Then, for any $\delta > 0$, with probability at least $1 - \delta$, the following holds:

$$\forall h \in \mathcal{F}, R(h) \leq \hat{R}_w(h) + 2^{\frac{5}{4}}\sqrt{d_2(P||Q)}\sqrt[3/8]{\frac{p\log\frac{2ne}{p} + \log\frac{4}{\delta}}{n}}. \tag{4.1}$$

From Equation 4.1, we can conclude that

$\forall h \in \mathcal{F}$, with probability at least $1-\delta$,

we have

$$R_C(h) \leq \hat{R}_C(h) + \Delta_C(\delta),$$

where

$$\Delta_C(\delta) = 2^{\frac{5}{4}} \sqrt{d_2(\mu_T||\mu_C)} \sqrt[3/8]{\frac{p \log \frac{2n_C e}{p} + \log \frac{4}{\delta}}{n_C}}.$$

Hence, we can combine these two inequalities using union bound and obtain the following:

$\forall h \in \mathcal{F}$, with probability at least $1-\delta$,

$$\begin{aligned}
\max(R_T(h), R_C(h)) &\leq \max\left(\hat{R}_T(h) + \Delta_T\left(\frac{\delta}{2}\right), \hat{R}_C(h) + \Delta_C\left(\frac{\delta}{2}\right)\right) \\
&\leq \max(\hat{R}_T(h), \hat{R}_C(h)) + \max\left(\Delta_T\left(\frac{\delta}{2}\right), \Delta_C\left(\frac{\delta}{2}\right)\right)
\end{aligned}$$

We complete the proof by noticing that from definition of l^M and linearity of expectation that we have

$$\begin{aligned}
&\mathbb{E}_{X\sim\mu_{X|T}, Y^T\sim\mu_{Y^T|X}, Y^C\sim\mu_{Y^C|X}} l_1(X, Y^T, Y^C, h) \\
&= M\mathbb{E}_{X\sim\mu_{X|T}, Y^T\sim\mu_{Y^T|X}, Y^C\sim\mu_{Y^C|X}} l_1^M(X, Y^T, Y^C, h) \\
&\leq M \max(R_T(h), R_C(h)) \\
&\leq \max(\hat{R}_T(h), \hat{R}_C(h)) + \max\left(\Delta_T\left(\frac{\delta}{2}\right), \Delta_C\left(\frac{\delta}{2}\right)\right).
\end{aligned}$$

Additional Experimental Results

Due to the page limit constraint in the main paper, here are results on some additional data sets.

Spiral Dataset without noise: The first data set that we present is the spiral data set as shown in Figure 4-3 without any noise. The support vectors are noted in the figure for causal SVM. Causal SVM and 2 SVM perform comparably, and the

determination of which one performs better depends on the kernel bandwidth – here the 2 SVM is slightly better. Numerical results are in Table 4.3.

	0.01	0.1
linear causal SVM 1e-8	47.33(1.82)	43.47(1.29)
linear causal SVM 1e-6	47.33(1.82)	43.47(1.29)
linear causal SVM 1e-4	47.34(1.81)	43.47(1.29)
quadratic causal SVM 1e-8	51(0.97)	46.13(0.81)
quadratic causal SVM 1e-6	51(0.97)	46.13(0.81)
quadratic causal SVM 1e-4	50.95(1.01)	46.16(0.79)
cubic causal SVM 1e-8	46.23(2.16)	42.3(0.82)
cubic causal SVM 1e-6	46.23(2.16)	42.3(0.82)
cubic causal SVM 1e-4	46.21(2.34)	42.31(0.82)
rbf causal SVM 0.05, 1e-8	25.27(2.9)	20.95(2.28)
rbf causal SVM 0.05, 1e-6	31.26(2.81)	26.98(2.98)
rbf causal SVM 0.05, 1e-4	47.69(2.42)	42.94(2.16)
rbf causal SVM 0.1, 1e-8	$4.32(0.78)^3$	$2.03(0.27)^2$
rbf causal SVM 0.1, 1e-6	26.39(2.63)	22.3(2.04)
rbf causal SVM 0.1, 1e-4	39.25(2.17)	34.65(1.76)
GenMatch, Ridge	48.34(2.94)	43.84(1.91)
Nearest, Ridge	49.33(1.66)	44.88(1.47)
Genmatch, kernel ridge	43.64(4.94)	38.69(4.24)
Nearest, kernel ridge	44.93(4.95)	40.44(5.02)
2 ridge	48.52(2.33)	44.27(1.63)
2 kernel ridge	43.11(3.52)	38.73(3.22)
2 logistic	48.74(2.32)	44.3(1.75)
2 SVM	$0.05(0.05)^1$	$0.01(0.03)^1$
2 RF	$4.24(0.87)^2$	$1.51(0.76)^2$
causal_rf 0.05 0	34.06(3.32)	29.72(2.89)
causal_rf 0.01 0	47.04(2.64)	42.14(2.36)
causal_rf 0.05 0.1	32.92(2.67)	28.85(2.6)
causal_rf 0.01 0.1	45.35(2.15)	41.37(2.56)

Table 4.3: Numerical output for the spiral data set. As we can see, our method is the best method without using the difference of two supervised classifiers.

A Dataset Where the Treatment Effect Changes a Few Times

We construct a 2-dimensional data set as follows. The features are distributed uniformly between 0 and 1. We denote $x_{i,j}$ as i is the index for the i-th data point and j is the index for the feature. If $x_{i,1} < 0.6$, $y_i^T = 1$ with probability 0.4 and $y_i^C = 1$ with probability 0.6; If $x_{i,1}$ is between 0.6 and 0.8, $y_i^T = 1$ with probability 0.3 and $y_i^C = 1$ with probability 0.7; Otherwise, $y_i^T = 1$ with probability 0.8 and $y_i^C = 1$ with probability 0.2;

Causal SVM outperforms other algorithms for this data set. Numerical results are in Table 4.4.

	0.01	0.1
linear causal SVM 1e-8	62.1(3.18)	55.3(3.3)
linear causal SVM 1e-6	62.1(3.18)	55.3(3.3)
linear causal SVM 1e-4	61.5(3.34)	54.6(3.06)
quadratic causal SVM 1e-8	61.2(4.54)	53.9(4.33)
quadratic causal SVM 1e-6	60.6(4.55)	53.5(4.33)
quadratic causal SVM 1e-4	61.7(5.06)	55(5.25)
cubic causal SVM 1e-8	59.6(3.78)[1]	53.2(4.02)[2]
cubic causal SVM 1e-6	61.8(6)	55.3(5.29)
cubic causal SVM 1e-4	61.5(5.74)	54.9(5.2)
rbf causal SVM 0.05, 1e-8	60.5(3.84)	53.7(3.59)
rbf causal SVM 0.05, 1e-6	60.1(4.65)	53.4(4.86)[3]
rbf causal SVM 0.05, 1e-4	60.7(3.56)	53.7(3.33)
rbf causal SVM 0.1, 1e-8	61.6(3.84)	54.8(3.88)
rbf causal SVM 0.1, 1e-6	60(4.22)[2]	53.1(4.09)[1]
rbf causal SVM 0.1, 1e-4	60.1(3.98)[3]	53.6(3.06)
GenMatch, Ridge	65.2(3.71)	58.4(3.31)
Nearest, Ridge	66.2(2.97)	59.5(2.59)
Genmatch, kernel ridge	65.6(6.19)	58.5(5.56)
Nearest, kernel ridge	65.2(3.52)	58.9(3.93)
2 ridge	63(3.5)	56.4(3.57)
2 kernel ridge	63.9(3.96)	57.4(3.81)
2 logistic	63.3(3.43)	57.3(2.83)
2 SVM	66(3.46)	59.5(3.24)
2 RF	64.5(1.51)	59(1.89)
causal_rf 0.05 0	67.3(6.5)	59.8(5.81)
causal_rf 0.01 0	64.7(23.28)	63.4(22.81)
causal_rf 0.05 0.1	68.4(6.02)	61.1(4.61)
causal_rf 0.01 0.1	70.4(4.9)	70.4(4.9)

Table 4.4: The output for a data set where the treatment effect changes a few times. Our method seems to be more suited for this type of data set.

We can see from Table 4.4 that for this particular data set, our approaches outperform the other algorithms.

Imbalanced when it is more likely to belong to the control group

This is a simulated data set that consists of 1000 data points. Each data point has 30 features. The first 20 features are independently generated from a normal distribution with mean 0 and variance 1, while the remaining 10 features are uniformly distributed between -1 and 1. The treatment effect is determined by the 2-norm of

the feature. If the feature has norm that is bigger than 3, then y_i^T takes value 1 with probability 0.8 ; Otherwise, y_i^T takes value -1 with probability 0.2. y_i^C always take value 1 with probability 0.2. Each data point has a probability of 0.3 of being assigned to the control group. Table 4.5 shows that causal SVM is comparable to 2-SVM, and matching-based methods have worse performance.

	0.01	0.1
linear causal SVM 1e-8	57.8(2.92)	51.6(2.91)
linear causal SVM 1e-6	57.8(2.92)	51.6(2.91)
linear causal SVM 1e-4	57.82(2.98)	51.68(2.91)
quadratic causal SVM 1e-8	63.36(1.83)	57.02(1.61)
quadratic causal SVM 1e-6	63.4(1.81)	57.02(1.6)
quadratic causal SVM 1e-4	59.6(1.35)	53.3(1.46)
cubic causal SVM 1e-8	62.58(1.87)	56.1(1.81)
cubic causal SVM 1e-6	62.58(1.87)	56.12(1.83)
cubic causal SVM 1e-4	55.38(2.07)	49.58(1.9)
rbf causal SVM 0.05, 1e-8	55.02(1.6)	48.82(1.55)
rbf causal SVM 0.05, 1e-6	55.02(1.6)	48.82(1.55)
rbf causal SVM 0.05, 1e-4	55.02(1.6)	48.82(1.55)
rbf causal SVM 0.1, 1e-8	50.82(1.74)	44.2(1.62)
rbf causal SVM 0.1, 1e-6	50.82(1.74)	44.16(1.58)[2]
rbf causal SVM 0.1, 1e-4	50.82(1.74)	44.16(1.58)[2]
GenMatch, Ridge	52.84(2.56)	47.22(2.28)
Nearest, Ridge	50.88(2.23)	46.38(2.51)
Genmatch, kernel ridge	56.76(2.44)	50.76(2.2)
Nearest, kernel ridge	53.52(2.09)	47.96(1.82)
2 ridge	51.02(1.97)	46(2.19)
2 kernel ridge	53.44(1.94)	47.7(2.05)
2 logistic	57.4(1.8)	50.88(1.53)
2 SVM	50.4(2.1)[1]	43.78(2.07)[1]
2 RF	50.86(2.12)	45.84(1.97)
causal_rf 0.05 0	50.68(2.02)[3]	45.1(2.22)
causal_rf 0.01 0	50.82(2.15)	45.46(2.04)
causal_rf 0.05 0.1	50.66(2.16)[2]	45.16(2.04)
causal_rf 0.01 0.1	50.78(2.11)	45.66(2.5)

Table 4.5: The output for a data set that simulate a scenario where it is more likely to be assigned to the control group. It is shown that our RBF-based methods beat matching-based methods.

A dataset with treatment effect that changes a few times in high dimensions

This is a data set which consists of 1000 data points where each data point consists

of 120 features. 60 of the features follows independent normal distribution with mena 0 and standard deviation 1 and 60 features follows uniform distribution between -1 and 1. The treatment effect is a function of the 2-norm of each data point. If the norm is less than 3, y_i^T takes value 1 with probability 0.4 while y_i^C takes value 1 with probability 0.6; If the norm is between 3 and 4, y_i^T takes value 1 with probability 0.3 while y_i^C takes value 1 with probability 0.7; Otherise, y_i^T and y_i^C independently take value 1 with probability 0.2. It is equally likely for a data to be assigned to a treatment group or a control group.

	0.01	0.1
linear causal SVM 1e-8	69.52(2.34)	62.98(2.26)
linear causal SVM 1e-6	69.5(2.38)	62.98(2.26)
linear causal SVM 1e-4	69.48(2.2)	62.9(2.11)
quadratic causal SVM 1e-8	67.3(1.33)	60.56(1.32)
quadratic causal SVM 1e-6	67.2(1.38)	60.66(1.3)
quadratic causal SVM 1e-4	67.18(1.36)	60.66(1.3)
cubic causal SVM 1e-8	61.74(1.28)	55.1(1.44)
cubic causal SVM 1e-6	61.7(1.32)	55.08(1.51)[2]
cubic causal SVM 1e-4	61.7(1.32)	55.08(1.51)[2]
rbf causal SVM 0.05, 1e-8	61.12(1.35)	55.54(1.05)
rbf causal SVM 0.05, 1e-6	61.12(1.35)	55.54(1.05)
rbf causal SVM 0.05, 1e-4	61.12(1.35)	55.54(1.05)
rbf causal SVM 0.1, 1e-8	61.2(1.38)	55.82(1.07)
rbf causal SVM 0.1, 1e-6	61.2(1.38)	55.8(1.06)
rbf causal SVM 0.1, 1e-4	61.2(1.38)	55.8(1.06)
GenMatch, Ridge	61.74(1.59)	55.9(1.66)
Nearest, Ridge	61.6(1.87)	55.68(1.72)
Genmatch, kernel ridge	62.98(1.53)	56.62(1.43)
Nearest, kernel ridge	62.94(1.78)	56.12(1.67)
2 ridge	61.1(1.54)[3]	55.66(1.82)
2 kernel ridge	63.5(1.6)	56.76(1.39)
2 logistic	74.98(5.98)	70.18(9.18)
2 SVM	61.08(1.41)[1]	56.72(2.62)
2 RF	62.26(1.35)	56.18(1.16)
causal_rf 0.05 0	61.24(1.41)	55.7(1.4)
causal_rf 0.01 0	61.16(1.38)	55.06(1.38)[1]
causal_rf 0.05 0.1	61.3(1.62)	55.66(1.69)
causal_rf 0.01 0.1	61.1(1.36)[2]	55.52(1.57)

Table 4.6: The output table for a high dimensional data set where a data point is equally likely to be assigned to the treatment group or control group.

For this data set, Table 4.6 shows that our method ,without using matching,

is highly competitive compared to the approaches that using the difference of two classification or regression methods as well as causal random forest approach.

Red Wine data set

We provided experiment with the Red Wine data set in the main paper. We also perform similar experiment under different assignment mechanism settings for this data set.

Setting 1: Equally likely to be assigned to be treatment or control group.

	0.01	0.1
linear causal SVM 1e-8	39.64(1.03)	34.29(1.18)
linear causal SVM 1e-6	39.64(1.03)	34.34(1.25)
linear causal SVM 1e-4	39.62(1.07)	34.34(1.22)
quadratic causal SVM 1e-8	48.78(1.6)	43.71(1.53)
quadratic causal SVM 1e-6	48.74(1.59)	43.75(1.74)
quadratic causal SVM 1e-4	49.08(2.53)	43.65(2.57)
cubic causal SVM 1e-8	43.84(1.33)	38.96(1.12)
cubic causal SVM 1e-6	40.2(1.28)	35.39(1.16)
cubic causal SVM 1e-4	40.8(2.72)	35.59(2.51)
rbf causal SVM 0.05, 1e-8	45.66(1.56)	40.75(1.26)
rbf causal SVM 0.05, 1e-6	43.32(1.95)	38.02(1.86)
rbf causal SVM 0.05, 1e-4	38.18(1.3)[2]	32.81(1.49)[2]
rbf causal SVM 0.1, 1e-8	45.99(1.87)	40.81(1.65)
rbf causal SVM 0.1, 1e-6	44.51(1.45)	39.51(1.33)
rbf causal SVM 0.1, 1e-4	39.19(1.78)	33.85(1.44)
GenMatch, Ridge	39.42(1.02)	34.38(0.9)
Nearest, Ridge	39.39(1.22)	34.41(1.28)
Genmatch, kernel ridge	39.1(1.63)	34.31(1.59)
Nearest, kernel ridge	42.09(2.31)	37.03(2.25)
2 ridge	39.01(1.39)	33.88(0.97)
2 kernel ridge	39.02(1.93)	34.01(1.66)
2 logistic	39.08(1.12)	33.88(0.88)
2 SVM	42.85(1.75)	37.6(1.73)
2 RF	36.1(1.6)[1]	31.55(1.62)[1]
causal_rf 0.05 0	38.31(1.01)[3]	33.22(1.06)
causal_rf 0.01 0	40.61(3.44)	35.44(3.52)
causal_rf 0.05 0.1	38.55(0.86)	33.1(1.07)[3]
causal_rf 0.01 0.1	41.91(3.2)	36.81(3.19)

Table 4.7: Output table for the red wine data set where each data point is equally likely to be assigned to the treatment or control group.

For this setting, from Table 4.7, the difference of two-random forest model seems to perform better than other algorithms. Causal SVM with RBF kernels performs

	0.01	0.1
linear causal SVM 1e-8	40.39(1.42)	35.29(1.17)
linear causal SVM 1e-6	40.39(1.42)	35.29(1.17)
linear causal SVM 1e-4	40.4(1.35)	35.25(1.15)
quadratic causal SVM 1e-8	48.02(2.32)	42.9(2.43)
quadratic causal SVM 1e-6	47.92(2.09)	42.78(2.28)
quadratic causal SVM 1e-4	48.99(2.16)	43.64(1.9)
cubic causal SVM 1e-8	44.52(1.28)	39.41(1.35)
cubic causal SVM 1e-6	41.7(1.36)	36.54(1.42)
cubic causal SVM 1e-4	44.25(3.07)	38.61(2.65)
rbf causal SVM 0.05, 1e-8	45.19(2.02)	40.02(1.73)
rbf causal SVM 0.05, 1e-6	42.98(1.71)	38.19(1.64)
rbf causal SVM 0.05, 1e-4	39.39(1.44)[2]	34.49(1.48)[2]
rbf causal SVM 0.1, 1e-8	45.41(1.27)	40.24(1.54)
rbf causal SVM 0.1, 1e-6	44.78(1.65)	39.6(1.6)
rbf causal SVM 0.1, 1e-4	40.65(1.47)	35.36(1.35)
GenMatch, Ridge	40.11(1)[3]	34.96(0.98)
Nearest, Ridge	41.41(1.43)	36.12(1.52)
Genmatch, kernel ridge	41.3(1.33)	36.3(1.21)
Nearest, kernel ridge	43.15(1.4)	37.84(1.26)
2 ridge	40.32(1.04)	34.92(1.01)[3]
2 kernel ridge	40.49(1.64)	35.29(1.53)
2 logistic	40.34(0.84)	35.11(1)
2 SVM	43.02(2.21)	37.71(2.05)
2 RF	38.26(1.02)[1]	32.99(1.08)[1]
causal_rf 0.05 0	41.45(1.41)	36.19(1.38)
causal_rf 0.01 0	50.98(4.76)	44.55(3.53)
causal_rf 0.05 0.1	41.48(1.45)	36.18(1.62)
causal_rf 0.01 0.1	51.51(4.62)	45.19(3.81)

Table 4.8: Output table for the red wine data where the assignment mechanism is based on Bernoulli $\left(0.75\left(\frac{1-\exp(-x_c^2)}{1+\exp(-x_c^2)}\right)\right)$

similarly.

Setting 2: The assignment mechanism is based on Bernoulli $\left(0.75\left(\frac{1-\exp(-x_c^2)}{1+\exp(-x_c^2)}\right)\right)$

In this setting, we let our assignment mechanism be depending on the reading of citric acid. We let x_c denotes the reading of the citric acid and we let the probability that one is being assigned to the treatment group follows Bernoulli $\left(0.75\left(\frac{1-\exp(-x_c^2)}{1+\exp(-x_c^2)}\right)\right)$. Table 4.8 shows that the 2-random forest method achieves the best performance, but the performance for most methods is very similar.

Setting 3: The assignment mechanism is based on Bernoulli $\left(0.5\left(\frac{1-\exp(-x_c^2)}{1+\exp(-x_c^2)}\right)\right)$

In this setting, we let our assignment mechanism be depending on the reading of

	0.01	0.1
linear causal SVM 1e-8	40.62(1.53)	35.5(1.42)
linear causal SVM 1e-6	40.62(1.53)	35.51(1.39)
linear causal SVM 1e-4	40.64(1.44)	35.56(1.32)
quadratic causal SVM 1e-8	49.99(0.99)	44.44(1.23)
quadratic causal SVM 1e-6	49.71(0.99)	44.36(1.12)
quadratic causal SVM 1e-4	50.21(1.93)	44.78(2.02)
cubic causal SVM 1e-8	44.69(1.92)	39.7(2)
cubic causal SVM 1e-6	42.46(1.39)	37.08(1.16)
rbf causal SVM 0.05, 1e-8	46.48(1.49)	41.38(1.32)
rbf causal SVM 0.05, 1e-6	44.12(1.64)	39.06(1.59)
rbf causal SVM 0.05, 1e-4	39.21(0.94)[2]	34.05(0.92)[2]
rbf causal SVM 0.1, 1e-8	46.42(1.22)	41.61(1.61)
rbf causal SVM 0.1, 1e-6	45.66(1.39)	40.59(1.37)
rbf causal SVM 0.1, 1e-4	39.86(0.84)[3]	34.74(0.86)[3]
GenMatch, Ridge	40.89(1.05)	35.41(0.61)
Nearest, Ridge	41.59(1.63)	36.21(1.56)
Genmatch, kernel ridge	41.96(1.59)	36.62(1.39)
Nearest, kernel ridge	43.18(1.47)	37.84(1.42)
2 ridge	40.48(0.82)	35.19(0.56)
2 kernel ridge	41.59(1.18)	36.12(1.33)
2 logistic	41.06(0.73)	35.48(0.67)
2 SVM	41.82(2.6)	36.55(2.75)
2 RF	38.26(1.11)[1]	33.4(1.18)[1]
causal_rf 0.05 0	41.3(2.1)	35.94(1.96)
causal_rf 0.01 0	50.89(4.61)	44.71(4.16)
causal_rf 0.05 0.1	41.4(2.52)	36.11(2.52)
causal_rf 0.01 0.1	48.75(5.93)	42.88(5.05)

Table 4.9: Output table for the red wine data where the assignment mechanism is based on Bernoulli $\left(0.5\left(\frac{1-\exp(-x_c^2)}{1+\exp(-x_c^2)}\right)\right)$

citric acid. We let x_c denotes the reading of the citric acid and we let the probability that one is being assigned to the treatment group follows Bernoulli $\left(0.5\left(\frac{1-\exp(-x_c^2)}{1+\exp(-x_c^2)}\right)\right)$.

From Table 4.9, the two-random forest method again outperform all algorithm, however, our algorithm achieve similar result using a simpler model.

Summary of Experiments

Methods based on differences of two predictive models often use a richer class of functions than methods using a single model. Our experiments tend to favor the more complex model classes, such as difference of 2-SVM, despite the fact that there is no real theoretical principle underlying the use of 2-SVM. There are some

advantages to using a single model, beyond the tighter bound on the 0-1 loss, and generalization bounds, in particular, better control over the complexity of the model, a single global optimization problem to solve with a guarantee of optimality, which is easier to troubleshoot and trust.

Chapter 5

Conclusion

In this thesis, we have attempted to address three challenging problems from different settings. In the first problem, the number of data points from each class is unbalanced, and we seek to understand the data from the minority class. In the second problem, we try to understand a complicated histogram, trying to describe a high dimensional histogram in a high dimension using just the right amount of description. The last problem that we consider is the fundamental problem of causal inference where we try to find the treatment effect but we do not have access to the ground truth. In each of these problems, the nature of information available to us differs. However, a thesis doesn't mark the end of its author's research, and there are more things to work on in these area.

In the first two topics of the research, we cover interpretable models using techniques that use boxes to characterize interesting patterns. The attempt is to make the patterns interpretable and the intuition is interpretable, following the intuition that sparsity improves interpretability. We thus seek to use as few rules as possible to parsimoniously describe a few patterns. Without focusing on a particular domain, sparsity seems like a reasonable correlate of interpretability. However, with domain knowledge, interpretability might come in different forms. For example, in the vision research community, a transformation to convert a set of pixels into a common, easily identifiable object is certainly more interpretable than the number of pixels in the set. It is a challenging topic to even characterize the meaning of interpretability. What

matters most is whether we can use such conclusions to help in decision making for the benefit of society. While it is hard to come up with a measure that works across all domains, we hope that the technique we have used can be adapted to a particular domain — for example, rather than regularizing on the number of rules, convert that to the number of interpretable entities relevant to that research area. Furthermore, even if we obtain a tree, it might still require a domain expert and a machine learning person to explain the conclusions of the final model in layman's terms. Any attempts to reduce the communication gap can be of great interest.

While models that are highly complex but accurate are necessary, I believe simple models that are interpretable can help humans make more informed decisions as well.

Also, a part of this thesis is dedicated to the fundamental problem of causal inference, which arises in many real-life problems in which the ground truth is not known. As such, alternative models may serve as a reference for the ground truth. The conventional way is to perform matching or perform regression separately to get an estimate of causal inference. We provided a framework which is SVM-like and which enables us to use the kernel trick. We hope that such a framework can supplement existing methods. A possible generalization to this framework is to use statistics to enable our model to conclude that we might not have sufficient data to make a conclusion. Also, an extension from binary-valued outcomes to real-valued outcomes would be interesting.

Bibliography

[1] N. Abe. Sampling approaches to learning from imbalanced datasets: Active learning, cost senstive learning and beyond, 2003. Proc. ICML, Workshop Learning from Imbalanced Data Sets II.

[2] Saab Abou-Jaoude. Conditions nécessaires et suffisantes de convergence l1 en probabilité de l'histogramme pour une densité. *Annales de l'IHP Probabilités et statistiques*, 12(3):213–231, 1976.

[3] Hirotugu Akaike. An approximation to the density function. *Annals of the Institute of Statistical Mathematics*, 6(2):127–132, 1954.

[4] J. Alcala-Fdez, A. Fernandez, J. Luengo, J. Derrac, and S. Garcia. Keel data-mining sofware tool: Data set repository, integration of algorithms and experimental analysis framework. 2011. `http://sci2s.urg.es/keel/imbalanced.php#sub3`.

[5] Elaine Angelino, Nicholas Larus-Stone, Daniel Alabi, Margo Seltzer, and Cynthia Rudin. Certifiably optimal rule lists for categorical data. In *Proceedings of the 23rd ACM SIGKDD Conference of Knowledge, Discovery, and Data Mining (KDD)*, 2017.

[6] Elaine Angelino, Nicholas Larus-Stone, Daniel Alabi, Margo Seltzer, and Cynthia Rudin. Learning certifiably optimal rule lists. In *Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '17, pages 35–44, New York, NY, USA, 2017. ACM.

[7] Susan Athey and Guido Imbens. Recursive partitioning for heterogeneous causal effects. *Proceedings of the National Academy of Sciences*, 113(27):7353–7360, 2016.

[8] Susan Athey and Guido W Imbens. Machine learning methods for estimating heterogeneous causal effects. *stat*, 1050(5), 2015.

[9] K. Bache and M. Lichman. UCI machine learning repository. 2013. `http://archive.ics.uci.edu/ml`.

[10] Heejung Bang and James M Robins. Doubly robust estimation in missing data and causal inference models. *Biometrics*, 61(4):962–973, 2005.

[11] Alexandre Belloni, Victor Chernozhukov, and Christian Hansen. High-dimensional methods and inference on structural and treatment effects. *The Journal of Economic Perspectives*, 28(2):29–50, 2014.

[12] Olivier Bousquet, Stéphane Boucheron, and Gábor Lugosi. Introduction to statistical learning theory. In *Advanced lectures on machine learning*, pages 169–207. Springer, 2004.

[13] Theophilos Cacoullos. Estimation of a multivariate density. *Annals of the Institute of Statistical Mathematics*, 18(1):179–189, 1966.

[14] Soumen Chakrabarti, Martin Ester, Usama Fayyad, Johannes Gehrke, Jiawei Han, Shinichi Morishita, Gregory Piatetsky-Shapiro, and Wei Wang. Data mining curriculum: A proposal (version 1.0). *Intensive Working Group of ACM SIGKDD Curriculum Committee*, 2006.

[15] Chih-Chung Chang and Chih-Jen Lin. LIBSVM: A library for support vector machines. *ACM Transactions on Intelligent Systems and Technology*, 2:27:1–27:27, 2011. Software available at `http://www.csie.ntu.edu.tw/~cjlin/libsvm`.

[16] Nitesh V. Chawla. C4.5 and imbalanced data sets: investigating the effect of sampling method, probabilistic estimate, and decision tree structure. In *In Proceedings of the ICMLo03 Workshop on Class Imbalances*, 2003.

[17] Nitesh V. Chawla, Kevin W. Bowyer, Lawrence O. Hall, and W. Philip Kegelmeyer. Smote: Synthetic minority over-sampling technique. *Journal of Artificial Intelligence Research*, 16:321–357, 2002.

[18] Nitesh V. Chawla, Nathalie Japkowicz, and Aleksander Kotcz. Editorial: Special issue on learning from imbalanced data sets. *SIGKDD Explor. Newsl.*, 6(1):1–6, June 2004.

[19] Tao Chen, Julian Morris, and Elaine Martin. Probability density estimation via an infinite gaussian mixture model: application to statistical process monitoring. *Journal of the Royal Statistical Society: Series C (Applied Statistics)*, 55(5):699–715, 2006.

[20] DavidA. Cieslak, T.Ryan Hoens, NiteshV. Chawla, and W.Philip Kegelmeyer. Hellinger distance decision trees are robust and skew-insensitive. *Data Mining and Knowledge Discovery*, 24(1):136–158, 2012.

[21] Corinna Cortes, Yishay Mansour, and Mehryar Mohri. Learning bounds for importance weighting. In *Advances in neural information processing systems*, pages 442–450, 2010.

[22] Rajeev H Dehejia and Sadek Wahba. Propensity score-matching methods for nonexperimental causal studies. *The review of economics and statistics*, 84(1):151–161, 2002.

[23] L. Devroye, L. Györfi, and G. Lugosi. *A Probabilistic Theory of Pattern Recognition*. Springer, 1996.

[24] Luc Devroye. Exponential inequalities in nonparametric estimation. In *Nonparametric functional estimation and related topics*, pages 31–44. Springer, 1991.

[25] Luc Devroye and László Györfi. Distribution-free exponential bound on the l1 error of partitioning estimates of a regression function. *Probability and statistical decision theory, Vol. A*, 67:76, 1983.

[26] Alexis Diamond and Jasjeet S. Sekhon. Genetic matching for estimating causal effects: A general multivariate matching method for achieving balance in observational studies. *The Review of Economics and Statistics*, 95(3):932–945, 2013.

[27] Jared C Foster. Subgroup identification and variable selection from randomized clinical trial data. 2013.

[28] Jerome H Friedman, Werner Stuetzle, and Anne Schroeder. Projection pursuit density estimation. *Journal of the American Statistical Association*, 79(387):599–608, 1984.

[29] JeromeH. Friedman and NicholasI. Fisher. Bump hunting in high-dimensional data. *Statistics and Computing*, 9(2):123–143, 1999.

[30] Siong Thye Goh and Cynthia Rudin. 2014. `http://web.mit.edu/stgoh/www/imbalanceddatafolder/`.

[31] Siong Thye Goh and Cynthia Rudin. Cascaded high dimensional histograms: A generative approach to density estimation. 2016.

[32] Siong Thye Goh and Cynthia Rudin. Supplementary document to a minimax surrogate loss approach to conditional difference estimation, 2017. `http://web.mit.edu/stgoh/www/mypage/causalsupplement.pdf`.

[33] Michael Hahsler, Bettina Grün, and Kurt Hornik. arules - a computational environment for mining association rules and frequent item sets. *Journal of Statistical Software, Articles*, 14(15):1–25, 2005.

[34] Douglas Marlowe Harrell, Adele V. and Jeffrey Merrill. Breaking the cycle of drugs and crime in birmingham, alabama, jacksonville, florida, and tacoma, wwashington, 1997-2001., 2004. `https://doi.org/10.3886/ICPSR03928.v1`.

[35] Haibo He and E.A. Garcia. Learning from imbalanced data. *Knowledge and Data Engineering, IEEE Transactions on*, 21(9):1263–1284, Sept 2009.

[36] Shohei Hido, Yuta Tsuboi, Hisashi Kashima, Masashi Sugiyama, and Takafumi Kanamori. Statistical outlier detection using direct density ratio estimation. *Knowledge and information systems*, 26(2):309–336, 2011.

[37] Paul W Holland. Statistics and causal inference. *Journal of the American statistical Association*, 81(396):945–960, 1986.

[38] Lasse Holmström, Kyösti Karttunen, and Jussi Klemelä. Estimation of level set trees using adaptive partitions. 2015.

[39] Robert C. Holte. Very simple classification rules perform well on most commonly used datasets. In *Machine Learning*, pages 63–91, 1993.

[40] Kosuke Imai, Luke Keele, Teppei Yamamoto, et al. Identification, inference and sensitivity analysis for causal mediation effects. *Statistical science*, 25(1):51–71, 2010.

[41] Kosuke Imai and Marc Ratkovic. Estimating treatment effect heterogeneity in randomized program evaluation. *Ann. Appl. Stat.*, 7(1):443–470, 03 2013.

[42] Kosuke Imai, Dustin Tingley, and Teppei Yamamoto. Experimental designs for identifying causal mechanisms. *Journal of the Royal Statistical Society: Series A (Statistics in Society)*, 176(1):5–51, 2013.

[43] N. Japkowicz. Class imbalances: Are we focusing on the right issue? In *Notes from the ICML Workshop on Learning from Imbalanced Data Sets II*, 2003.

[44] Tony S Jebara. Bayesian out-trees. *arXiv preprint arXiv:1206.3269*, 2012.

[45] Fredrik Johansson, Uri Shalit, and David Sontag. Learning representations for counterfactual inference. In *International Conference on Machine Learning*, pages 3020–3029, 2016.

[46] Luke Keele, Rocío Titiunik, and José R Zubizarreta. Enhancing a geographic regression discontinuity design through matching to estimate the effect of ballot initiatives on voter turnout. *Journal of the Royal Statistical Society: Series A (Statistics in Society)*, 178(1):223–239, 2015.

[47] Gary King, Christopher Lucas, and Richard A Nielsen. The balance-sample size frontier in matching methods for causal inference. *American Journal of Political Science*, 61(2):473–489, 2017.

[48] S. R. Künzel, J. S. Sekhon, P. J. Bickel, and B. Yu. Meta-learners for Estimating Heterogeneous Treatment Effects using Machine Learning. *ArXiv e-prints*, June 2017.

[49] Kevin J Lang. Learning to tell two spirals apart. In *Proc. of 1988 Connectionist Models Summer School*, 1988.

[50] Benjamin Letham, Cynthia Rudin, Tyler H. McCormick, and David Madigan. Interpretable classifiers using rules and bayesian analysis: Building a better stroke prediction model. *Annals of Applied Statistics*, 9(3):1350–1371, 2015.

[51] Jonathan Q. Li and Andrew R. Barron. Mixture density estimation. In *NIPS 12*, pages 279–285, 1999.

[52] Han Liu, John D. Lafferty, and Larry A. Wasserman. Sparse nonparametric density estimation in high dimensions using the rodeo. In *Proc. AISTATS-07*, volume 2, pages 283–290, 2007.

[53] Han Liu, Min Xu, Haijie Gu, Anupam Gupta, John Lafferty, and Larry Wasserman. Forest density estimation. *Journal of Machine Learning Research*, 12:907–951, July 2011.

[54] Xu-Ying Liu and Zhi-Hua Zhou. The influence of class imbalance on cost-sensitive learning: An empirical study. In *Data Mining, 2006. ICDM '06. Sixth International Conference on*, pages 970–974, Dec 2006.

[55] Gábor Lugosi, Andrew Nobel, et al. Consistency of data-driven histogram methods for density estimation and classification. *The Annals of Statistics*, 24(2):687–706, 1996.

[56] Ravi Mahapatruni and Alexander G Gray. Cake: Convex adaptive kernel density estimation. In *Proc. AISTATS*, pages 498–506, 2011.

[57] Kate McCarthy, Bibi Zabar, and Gary Weiss. Does cost-sensitive learning beat sampling for classifying rare classes? In *Proceedings of the 1st International Workshop on Utility-based Data Mining*, UBDM '05, pages 69–77, New York, NY, USA, 2005. ACM.

[58] Peter Müller and Fernando A Quintana. Nonparametric bayesian data analysis. *Statistical science*, pages 95–110, 2004.

[59] É A Nadaraya. Remarks on non-parametric estimates for density functions and regression curves. *Theory of Probability & Its Applications*, 15(1):134–137, 1970.

[60] Hong Ooi. Density visualization and mode hunting using trees. *Journal of Computational and Graphical Statistics*, 2012.

[61] Dirk Ormoneit and Volker Tresp. Improved gaussian mixture density estimates using bayesian penalty terms and network averaging. In *Proc. NIPS*, 1995.

[62] Dirk Ormoneit and Volker Tresp. Averaging, maximum penalized likelihood and bayesian estimation for improving gaussian mixture probability density estimates. *IEEE Transactions on Neural Networks*, 9(4):639–650, 1998.

[63] Emanuel Parzen. On estimation of a probability density function and mode. *The Annals of Mathematical Statistics*, pages 1065–1076, 1962.

[64] Benjamin Peherstorfer, Dirk Pflüge, and Hans-Joachim Bungartz. Density estimation with adaptive sparse grids for large data sets. In *Proceedings of the 2014 SIAM international conference on data mining*, pages 443–451. SIAM, 2014.

[65] RonaldoC. Prati, GustavoE.A.P.A. Batista, and MariaCarolina Monard. Class imbalances versus class overlapping: An analysis of a learning system behavior. In Ra?l Monroy, Gustavo Arroyo-Figueroa, LuisEnrique Sucar, and Humberto Sossa, editors, *MICAI 2004: Advances in Artificial Intelligence*, volume 2972 of *Lecture Notes in Computer Science*, pages 312–321. Springer Berlin Heidelberg, 2004.

[66] Foster Provost and Pedro Domingos. Tree induction for probability-based ranking, 2002.

[67] Foster Provost and Tom Fawcett. Robust classification for imprecise environments. *Machine Learning*, 42(3):203–231, 2001.

[68] Y. Qi. A brief literature review of class imbalanced problem. IR-Lab Project of Yanjun Qi, 2004.

[69] Parikshit Ram and Alexander G. Gray. Density estimation trees. In *KDD '11*, pages 627–635, 2011.

[70] Bhavani Raskutti and Adam Kowalczyk. Extreme re-balancing for svms: A case study. *SIGKDD Explor. Newsl.*, 6(1):60–69, June 2004.

[71] Marc Ratkovic, Kyle Marquardt, and Jasjeet Sekhon. Balancing within the margin: Causal effect estimation with support vector machinesâĹŮ. 2014.

[72] Marc Ratkovic and Dustin Tingley. Sparse estimation and uncertainty with application to subgroup analysis. *Political Analysis*, 25(1):1–40, 2017.

[73] L Rejtö and P Révész. Density estimation and pattern classification. *Problems of Control and Information Theory*, 2(1):67–80, 1973.

[74] James M Robins, Andrea Rotnitzky, and Lue Ping Zhao. Analysis of semiparametric regression models for repeated outcomes in the presence of missing data. *Journal of the american statistical association*, 90(429):106–121, 1995.

[75] Paul R Rosenbaum. The consquences of adjustment for a concomitant variable that has been affected by the treatment. *Journal of the Royal Statistical Society. Series A (General)*, pages 656–666, 1984.

[76] Paul R Rosenbaum and Donald B Rubin. Assessing sensitivity to an unobserved binary covariate in an observational study with binary outcome. *Journal of the Royal Statistical Society. Series B (Methodological)*, pages 212–218, 1983.

[77] Murray Rosenblatt et al. Remarks on some nonparametric estimates of a density function. *The Annals of Mathematical Statistics*, 27(3):832–837, 1956.

[78] Sudeepa Roy, Cynthia Rudin, Alexander Volfovsky, and Tianyu Wang. Flame: A fast large-scale almost matching exactly approach to causal inference. *arXiv*, 2017.

[79] Donald B Rubin. Estimating causal effects of treatments in randomized and nonrandomized studies. *Journal of educational Psychology*, 66(5):688, 1974.

[80] Donald B Rubin. Assignment to treatment group on the basis of a covariate. *Journal of Educational and Behavioral statistics*, 2(1):1–26, 1977.

[81] Donald B Rubin. Estimation in parallel randomized experiments. *Journal of educational and behavioral statistics*, 6(4):377–401, 1981.

[82] Cynthia Rudin and Seyda Ertekin. Learning optimized lists of rules. 2015.

[83] Daniel O Scharfstein, Andrea Rotnitzky, and James M Robins. Adjusting for nonignorable drop-out using semiparametric nonresponse models. *Journal of the American Statistical Association*, 94(448):1096–1120, 1999.

[84] David W Scott. On optimal and data-based histograms. *Biometrika*, 66(3):605–610, 1979.

[85] Thomas Seidl, Ira Assent, Philipp Kranen, Ralph Krieger, and Jennifer Herrmann. Indexing density models for incremental learning and anytime classification on data streams. In *In 12th EDBT/ICDT*, pages 311–322, 2009.

[86] U. Shalit, F. D. Johansson, and D. Sontag. Estimating individual treatment effect: generalization bounds and algorithms. *ArXiv e-prints*, jun 2016.

[87] Bernard W Silverman. *Density estimation for statistics and data analysis*, volume 26. CRC press, 1986.

[88] J. Sniadecki. Bump hunting with sas: A macro approach to employing prim. SAS Global Forum 2011, Data Mining and Text Analytics, 2011.

[89] Masashi Sugiyama, Shinichi Nakajima, Hisashi Kashima, Paul V Buenau, and Motoaki Kawanabe. Direct importance estimation with model selection and its application to covariate shift adaptation. In *Advances in neural information processing systems*, pages 1433–1440, 2008.

[90] Stefan Wager and Susan Athey. Estimation and inference of heterogeneous treatment effects using random forests. *Journal of the American Statistical Association*, (just-accepted), 2017.

[91] MP Wand. Data-based choice of histogram bin width. *The American Statistician*, 51(1):59–64, 1997.

[92] Larry Wasserman. *All of nonparametric statistics*. Springer Science & Business Media, 2006.

[93] Gary M. Weiss. Mining with rarity: A unifying framework. *SIGKDD Explor. Newsl.*, 6(1):7–19, June 2004.

[94] RM Willett and Robert D Nowak. Minimax optimal level-set estimation. *IEEE Transactions on Image Processing*, 16(12):2965–2979, 2007.

[95] Wing H Wong and Li Ma. Optional pólya tree and bayesian inference. *The Annals of Statistics*, 38(3):1433–1459, 2010.

[96] Gang Wu and Edward Y. Chang. Class-boundary alignment for imbalanced dataset learning. In *In ICML 2003 Workshop on Learning from Imbalanced Data Sets*, pages 49–56, 2003.

[97] Xindong Wu, Vipin Kumar, Ross, Joydeep Ghosh, and et al. Top 10 algorithms in data mining. *Knowledge and Information Systems*, 14(1):1–37, 2008.

[98] Hongyu Yang, Cynthia Rudin, and Margo Seltzer. Scalable bayesian rule lists. 2016.

[99] Kun Yang, Hao Su, and Wing Hung Wang. Density estimation via discrepancy. *arXiv preprint arXiv:1509.06831*, 2015.

[100] Kun Yang and Wing Hung Wong. Density estimation via adaptive partition and discrepancy control. *arXiv preprint arXiv:1404.1425*, 2014.

[101] Kun Yang and Wing Hung Wong. Discovering and visualizing hierarchy in the data. *arXiv preprint arXiv:1403.4370*, 2014.

[102] Lin Cheng Zhao, Paruchuri R Krishnaiah, and Xi Ru Chen. Almost sure ℓ_r-norm convergence for data-based histogram density estimates. *Theory of Probability & Its Applications*, 35(2):396–403, 1991.

[103] Xinhua Zhuang, Yan Huang, Kannappan Palaniappan, and Yunxin Zhao. Gaussian mixture density modeling, decomposition, and applications. *IEEE Transactions on Image Processing*, 5(9):1293–1302, 1996.

[104] José R Zubizarreta. Using mixed integer programming for matching in an observational study of kidney failure after surgery. *Journal of the American Statistical Association*, 107(500):1360–1371, 2012.