LEVERAGING SIMILARITY IN STATISTICAL LEARNING

A DISSERTATION
SUBMITTED TO THE DEPARTMENT OF STATISTICS
AND THE COMMITTEE ON GRADUATE STUDIES
OF STANFORD UNIVERSITY
IN PARTIAL FULFILLMENT OF THE REQUIREMENTS
FOR THE DEGREE OF
DOCTOR OF PHILOSOPHY

Scott Stephen Powers
June 2017

© 2017 by Scott Stephen Powers. All Rights Reserved.
Re-distributed by Stanford University under license with the author.

This work is licensed under a Creative Commons Attribution-Noncommercial 3.0 United States License.
http://creativecommons.org/licenses/by-nc/3.0/us/

This dissertation is online at: http://purl.stanford.edu/fb450fm0815

I certify that I have read this dissertation and that, in my opinion, it is fully adequate in scope and quality as a dissertation for the degree of Doctor of Philosophy.

Robert Tibshirani, Primary Adviser

I certify that I have read this dissertation and that, in my opinion, it is fully adequate in scope and quality as a dissertation for the degree of Doctor of Philosophy.

Jerome Friedman

I certify that I have read this dissertation and that, in my opinion, it is fully adequate in scope and quality as a dissertation for the degree of Doctor of Philosophy.

Trevor Hastie

Approved for the Stanford University Committee on Graduate Studies.

Patricia J. Gumport, Vice Provost for Graduate Education

This signature page was generated electronically upon submission of this dissertation in electronic format. An original signed hard copy of the signature page is on file in University Archives.

Abstract

Machine learning literature has provided a toolbox of techniques for general use by the modern applied statistician. For some data analysis, these methods can be improved to yield better results based on the properties of the data in question. In this manuscript, we develop methodology for three different problems in which, in some sense, similarity can be leveraged to make better predictions, as demonstrated in baseball and epidemiology applications. First, we propose adding a penalty on the nuclear norm of the regression coefficient matrix in multinomial regression to learn which outcomes are similar. Second, we propose adding a clustering step to ℓ_1-penalized regression, to build customized training sets of observations that are similar to the test set. Third, we propose and evaluate several approaches to mining electronic medical records to support physician decision-making with data on patients similar to a patient in question. This last problem is especially challenging because of the observational and high-dimensional nature of the data.

Acknowledgments

My gratitude begins with my parents, Robert and Karen Powers, who truly did give me every possible opportunity to succeed, and then some. I am acutely aware that other students face more obstacles than I did in my education, and that is an awareness I will carry with me. Mom and Dad are really terrific at parenting and instilled good habits in me from a young age. I hope to be half as good a parent to future children as they were to me. I still remember specific life lessons, like the importance of attitude and how to not sweat the small stuff. Anything that I accomplish, I owe in large part to them.

I would also not be the person who I am today without the influence of my brother, Alec. What I admire most about Alec is his resilience and positive attitude in the face of adversity. What I appreciate most about Alec is his belief in me. When I expressed my doubt to him that I could get into Stanford, he responded, "They'd be dumb not to accept you." While I can't agree with the fact of the statement, I do very much appreciate the sentiment behind it.

The reason I came to Stanford was to be with my partner-in-crime, Corinne. We've come a long way since we met almost 13 years ago at the Illinois Math and Science Academy. Corinne has been the one constant in my life for the past six years, the one to whom I've come home for the past four years. The support I have received from her as been invaluable in all aspects of my life.

I had two math teachers at the Illinois Math and Science Academy who were extremely influential in cultivating my early passion for math, and by extension statistics. Steve Condie taught me four semesters of calculus, and Michael Keyton taught me geometry, precalculus and differential equations. Both were coaches on the math team. I will never forget the definition of a *fresnarus*: a bleen sharponz without a swoze.

At the University of North Carolina at Chapel Hill, my undergraduate advisor Douglas Kelly was a very positive influence on me. He was the person from whom I started to learn what it meant to be an academic statistician. When I started my PhD at the University of Chicago, Matthew Stephens was my advisor, an I owe him a debt of gratitude for his willingness to take me on and his support in my decision to leave.

I have never felt more at home in a community than I did at Stanford. This is thanks mostly to the friends I met through Stanford Club Volleyball, Stanford Club Baseball and the Stanford

Sports Analytics Club. I am thankful also to my friends who attended Corinne's and my weekly MargaFriday^{TM}events during my final year. I've never been happier than in my time at Stanford, and I will miss this community.

I am thankful for the support I felt from my fellow statistics PhD students at the University of Chicago, especially Andy Poppick and Serena Rezny. I want to thank my fellow statistics PhD students at Stanford, especially the statistical learning research group. Thanks also to the Stanford statistics staff, especially Ellen Van Stone, Caroline Gates, Susie Ementon and Emily Lauderdale.

I am very grateful to the Stanford statistics faculty for contributing to my education, especially Jonathan Taylor, Balasubramanian Narasimhan, Art Owen and Joe Romano. I am very lucky that Mike Baiocchi stepped in on short notice to serve as the outside chair for my defense. I thank Brad Efron for serving as an examiner, as well as Jerry Friedman for serving as an examiner and as a reader.

Trevor Hastie has been like a second advisor to me, and I have learned a lot while collaborating with him on both research and teaching. His constructive criticism has helped me sharpen my ideas and my writing in each of these chapters. I would not be so prepared for my future career were it not for Trevor.

Finally, I am most grateful to my advisor, Rob Tibshirani, for advising me through this PhD. Rob is an incredibly kind, patient and supportive advisor, and I'm not sure whether I could have completed this work with another advisor. When I decided that I was going to work in baseball after graduation, it would have been very easy to give me less attention and focus on other students, but Rob helped me continue to feel like part of the academic community and to produce a work which I am proud to share with my friends and colleagues.

Chapter 2 is the product of joint work with Trevor and Rob. We thank Hristo Paskov, Reza Takapoui and Lucas Janson for helpful discussions, as well as Balasubramanian Narasimhan for computational assistance. Chapter 3 is the product of joint work with Trevor Hastie and Rob. We thank Livia Eberlin and Richard Zare for allowing us to use the gastric cancer data. Chapter 4 is the product of joint work with Junyang Qian, Kenneth Jung, Alejandro Schuler, Nigam Shah, Trevor and Rob. We thank Jonathan Taylor for helpful discussions.

I was supported financially at various times during my graduate studies by the NSF Graduate Research Fellowship, the McCormick Fellowship from the University of Chicago and the Alan M Abrams Memorial Fellowship.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

One afternoon during my senior year at the University of North Carolina, I sat in the office of Professor Andrew Nobel. We were nearing the end of the second statistics class I had taken under Dr. Nobel, who had a reputation as a tough instructor, offering tough love rather than coddling students. For me it was an effective pedagogy; he liked to teach a simile that sticks with me to this day. Studying for an exam is like training to run a marathon: You can spend time thinking about the race, but at some point you half to lace up your shoes and go for a run. Similarly, when learning new material, at some point you need to close your book and examine what you have learned. Despite pleading, Dr. Nobel offered his students no substitute for figuring things out on their own.

The reason I was at Dr. Nobel's office hour that day was to ask him about machine learning. It sounded like an exciting field of research, but I did not know what "machine learning" is. Dr. Nobel referred me to a book: *The Elements of Statistical Learning* by Trevor Hastie, Robert Tibshirani and Jerome Friedman. This book, as I would later come to learn, has inspired many students to begin careers in machine learning research, myself included. The book covers the basics of statistical learning, like regression, classification and validation, and it describes different methods — linear, nonlinear and nonparametric — for accomplishing these tasks. A fundamental takeaway from *The Elements* is that no one algorithm is the best to use in all situations. That depends on the dimension and distribution of the data, as well as the true relationship between predictors and response. There are other, more subtle factors. For example the lasso is particularly effective in the linear regression setting when the true regression coefficient vector is sparse.

This thesis is about leveraging similarity in statistical learning to make better predictions. That similarity is sometimes in the input space and at other times in the output space. The three chapters address different settings: supervised learning, transductive learning and causal learning, by which I mean the tasks of estimating a treatment effect function when only one response — treated or untreated — can be observed for each patient. All of this work was done in collaboration with others, as described in the acknowledgments chapter, and as such the rest of this manuscript, until

the conclusion, is written in the majestic plural.

In Chapter 2 we introduce Nuclear Penalized Multinomial Regression, which is multinomial logistic regression with a penalty on the nuclear norm of the regression coefficient matrix. This novel method is a convex relaxation of the previously developed reduced-rank multinomial regression, meaning that it is more feasible to solve on large datasets. The upshot is that, unlike standard or ridge multinomial regression, this method can learn which outcome categories are similar to each other, in the sense that when one is likely to occur, the other also tends to be more likely. We apply this method to baseball data, in which (for example) strikeouts are similar to home runs in the sense that batters who hit a lot of home runs tend also to strike out frequently.

In Chapter 3 we introduce customized training for transductive learning, in which the predictor variables in the test set are available of training time. The idea is to leverage knowledge of the test features to choose observations with similar features as the training set. A version of the method applies also to the supervised learning framework, in which no test data are available at time of training. Customized training can be regarded as a type of local regression, but it is much easier computationally than traditional local regression techniques. The method that we localize is ℓ_1-penalized linear and logistic regression, and the framework could just as easily be applied to any regression or classification method. We apply customized training to the task of classifying normal/cancerous cells in surgical resection of gastric cancer cells. We find in this setting it helps when classifying new tissue to create a customized training library of similar patients.

In Chapter 4 we propose and evaluate several methods for the problem of heterogeneous treatment effect estimation, a hot area of current research. Specifically, we seek a method applicable to high-dimensional observational data, motived by the very tangible goal of mining electronic medical records — like the hundreds of millions maintained by Stanford Health Care — to draw insights which can help inform physician's decision-making when considering treatment options for future patients. The relevant question from the patient's perspective is, *What is the effect of this treatment on patients like me?* We introduce three nonparametric ensemble methods: causal boosting, causal MARS and propensity transformed outcome forests, and compare them with the current state-of-the-art: gradient forests. We apply causal MARS and a gradient forest to data from the SPRINT challenge, which come from a large randomized trial comparing an intensive treatment of high blood pressure to the standard one.

Chapter 5 concludes the thesis with a discussion of the results and the outlook for future research in these areas. Together, the techniques presented in the following pages serve as a reminder that despite all that has been invented in this space, there is still room for simple ideas to improve specific applications in modern applied statistics. The toolbox laid out in *The Elements* provides the starting block for today's data scientists, and from these fundamentals they can tailor their analyses to best fit the problems at hand. This is, after all, the motivation of the applied statistician: to best infer real-life insights from real-life data.

Chapter 2

Nuclear penalized multinomial regression

We propose the nuclear norm penalty as an alternative to the ridge penalty for regularized multinomial regression. This convex relaxation of reduced-rank multinomial regression has the advantage of leveraging underlying structure among the response categories to make better predictions. We apply our method, nuclear penalized multinomial regression (NPMR), to Major League Baseball play-by-play data to predict outcome probabilities based on batter-pitcher matchups. The interpretation of the results meshes well with subject-area expertise and also suggests a novel understanding of what differentiates players.

2.1 Introduction

A baseball game comprises a sequence of matchups between one batter and one pitcher. Each matchup, or *plate appearance* (PA), results in one of several outcomes. Disregarding some obscure possibilities, we consider nine categories for PA outcomes: flyout (F), groundout (G), strikeout (K), base on balls (BB), hit by pitch (HBP), single (1B), double (2B), triple (3B) and home run (HR).

A problem which has received a prodigious amount of attention in sabermetric[1] literature is determining the value of each of the above outcomes, as it leads to scoring runs and winning games. But that is only half the battle. Much less developed is the solution to an equally important problem: optimally estimating the probabilities with which each batter and pitcher will produce each PA outcome. Even for "advanced metrics" this second task is usually done by taking simple empirical proportions, perhaps shrinking them toward a population mean using a Bayesian prior.

The state-of-the-art approach, Deserved Run Average [Judge and BP Stats Team, 2015, DRA], is

[1] Sabermetrics is the study of baseball statistics.

similar to the Rasch model used in psychometrics. This models the probability (on the logistic scale) that a student correctly answers an exam question as the difference between the student's skill and the difficulty of the question. DRA models players' skills as random effects and also includes fixed effects like the identity of the ballpark where the PA took place. Each category of the response has its own binomial regression model. Taking HR as an example, each batter B has a propensity β_B^{HR} for hitting home runs, and each pitcher P has a propensity γ_P^{HR} for allowing home runs. Distilling the model to its elemental form, if Y denotes the outcome of a PA between batter B and pitcher P,

$$\mathbb{P}(Y = \text{HR}|B, P) = \frac{e^{\alpha^{\text{HR}}+\beta_B^{\text{HR}}+\gamma_P^{\text{HR}}}}{1 + e^{\alpha^{\text{HR}}+\beta_B^{\text{HR}}+\gamma_P^{\text{HR}}}}.$$

(Actually, in detail DRA uses the probit rather than the logit link function.)

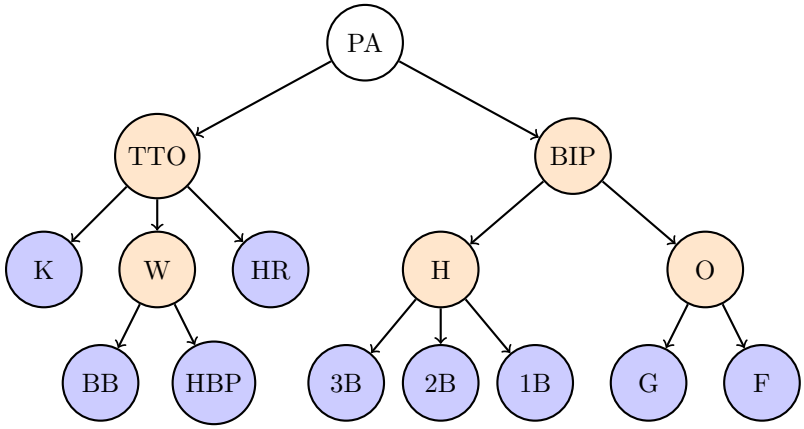
One bothersome aspect of DRA is that the probability estimates do not sum to one; a natural solution is to use a single multinomial regression model instead of several independent binomial regression models. That still leaves about 8,000 parameters to estimate (one for each outcome for each player) on the basis of about 160,000 PAs in a season, bound together only by the restriction that probability estimates sum to one. One may seek to exploit the structure of the problem to obtain better estimates, as in ordinal regression. The categories have an ordering, from least to most valuable to the batting team:

$$\text{K} < \text{G} < \text{F} < \text{BB} < \text{HBP} < \text{1B} < \text{2B} < \text{3B} < \text{HR},$$

with the ordering of the first three categories depending on the game situation. But the proportional odds model used for ordinal regression assumes that when one outcome is more likely to occur, the outcomes close to it in the ordering are also more likely to occur. That assumption is woefully off-base in this setting because as we show below, players who hit a lot of home runs tend to strike out often, and they tend not to hit many triples. The proportional odds model is better suited for response variables on the Likert scale [Likert, 1932], for example.

The actual relationships among the outcome categories are more similar to the hierarchical structure illustrated by Figure 2.1. The outcomes fall into two categories: balls in play (BIP) and the "three true outcomes" (TTO). The three true outcomes, as they have become traditionally known in sabermetric literature, include home runs, strikeouts and walks (which itself includes BB and HBP). The distinction between BIP and TTO is important because the former category involves all eight position players in the field on defense whereas the latter category involves only the batter and the pitcher. Figure 2.1 has been designed (roughly) by baseball experts so that terminal nodes close to each other (by the number of edges separating them) are likely to co-occur. For example, players who hit a lot of home runs tend to strike out a lot, and the outcomes K and HR are only two edges away from each other. Hence the graph reveals something of the underlying structure in the outcome categories.

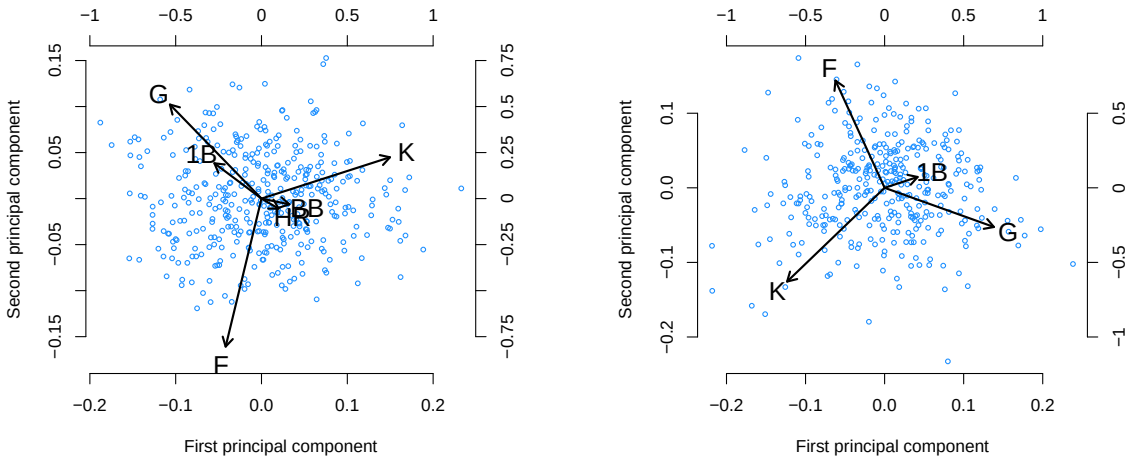
Figure 2.1: Illustration of the hierarchical structure among the PA outcome categories, adapted from Baumer and Zimbalist [2014]. Blue terminal nodes correspond to the nine outcome categories in the data. Orange internal nodes have the following meaning: TTO, three true outcomes; BIP, balls in play; W, walks; H, hits; O, outs. Outcomes close to each other (in terms of number of edges separating them) are likely to co-occur.



This structure is further evidenced by principal component analysis of the player-outcome matrix, illustrated in Figures 2.2 and 2.3. For batters, the principal component (PC) which describes most of the variance in observed outcome probabilities has negative loadings on all of the BIP outcomes and positive loadings on all of the TTO outcomes. For both batters and pitchers, the percentage of variance explained after the first two PCs drops off precipitously.

Principal component analysis is useful for illustrating the relationships between the outcome categories. For example Figure 2.3(a) suggests that batters who tend to hit singles (1B) more than average also tend to ground out (G) more than average. So an estimator of a batter's groundout rate could benefit from taking into account the batter's single rate, and *vice versa*. This is an example of the type of structure in outcome categories that motivates our proposal, which aims to leverage this structure to obtain better regression coefficient estimates in multinomial regression.

In Section 2.2 we review reduced-rank multinomial regression, a first attempt at leveraging this structure. We improve on this in Section 2.3 by proposing nuclear penalized multinomial regression, a convex relaxation of the reduced rank problem. We compare our method with ridge regression in a simulation study in Section 2.4. In Section 2.5 we apply our method and interpret the results on the baseball data, as well as another application. The manuscript concludes with a discussion in Section 2.6.



(a) Batters (b) Pitchers

Figure 2.2: Biplots of the principal component analyses of player outcome matrices, separate for batters and pitchers. The blue dots represent players, and the black arrows (corresponding to the top and right axes) show the loadings for the first two principal components on each of the outcomes. We exclude outcomes for which the loadings of both of the first two principal components are sufficiently small.

2.2 Multinomial logistic regression and reduced rank

Suppose that we observe data $\mathbf{x}_i \in \mathbb{R}^p$ and $Y_i \in \{1, ..., K\}$ for $i = 1, ..., n$. We use $\mathbf{X}$ to denote the matrix with rows $\mathbf{x}_i$, specifically $\mathbf{X} = (\mathbf{x}_1, ..., \mathbf{x}_n)^T$. The multinomial logistic regression model assumes that the Y_i are, conditional on $\mathbf{X}$, independent, and that for $k = 1, ..., K$:

$$\mathbb{P}(Y_i = k|\mathbf{x}_i) = \frac{e^{\alpha_k + \mathbf{x}_i^T \beta_k}}{\sum_{\ell=1}^{K} e^{\alpha_\ell + \mathbf{x}_i^T \beta_\ell}}, \tag{2.1}$$

were $\alpha_k \in \mathbb{R}$ and $\beta_k \in \mathbb{R}^p$ are fixed, unknown parameters. The model (2.1) is not identifiable because an equal increase in the same element of each of the β_k (or in each of the α_k) does not the change the estimated probabilities. That is, for each choice of parameter values there is an infinite set of choices which have the same likelihood as the original choice, for any observed data. This problem may readily be resolved by adopting the restriction for some $k_0 \in \{1, ..., K\}$ that $\alpha_{k_0} = 0$ and $\beta_{k_0} = \vec{0}_p$. However, depending on the method used to fit the model, this identifiability issue may not interfere with the existence of a unique solution; in such a case, we do not adopt this restriction. See the appendix for a detailed discussion.

In contrast with logistic regression, multinomial regression involves estimating not a vector but a matrix of regression coefficients: one for each independent variable, for each class. We denote this

Principal component	1	2	3	4	5	6	7	8	9
F	-0.2	0.7	0.5	-0.1	0.3	0.0	-0.1	0.1	-0.3
G	-0.5	-0.6	0.4	-0.3	0.1	-0.0	-0.1	0.1	-0.3
K	0.8	-0.3	0.3	0.2	0.2	0.1	-0.1	0.1	-0.3
BB	0.1	0.1	-0.6	-0.6	0.4	0.0	-0.1	0.1	-0.3
HBP	0.0	0.0	-0.0	0.0	-0.1	-0.1	0.9	0.1	-0.3
1B	-0.3	-0.0	-0.4	0.7	0.3	-0.1	-0.1	0.1	-0.3
2B	-0.0	0.1	-0.1	0.0	-0.5	0.7	-0.1	0.1	-0.3
3B	-0.0	-0.0	-0.0	0.0	-0.0	0.0	0.0	-0.9	-0.3
HR	0.1	0.1	-0.0	-0.1	-0.6	-0.6	-0.3	0.1	-0.3
% Variance explained	51.1	29.0	8.7	7.2	2.2	1.0	0.6	0.2	0.0

(a) Principal components of batter outcome matrix

Principal component	1	2	3	4	5	6	7	8	9
F	-0.3	-0.7	0.3	0.3	0.3	0.1	0.1	0.1	-0.3
G	0.7	0.2	0.4	0.3	0.1	0.1	0.1	0.1	-0.3
K	-0.6	0.7	0.3	-0.0	0.1	0.1	0.1	0.1	-0.3
BB	-0.0	0.1	-0.8	0.3	0.3	0.1	0.2	0.1	-0.3
HBP	0.0	0.0	-0.0	0.0	-0.0	-0.0	-0.9	0.1	-0.3
1B	0.2	-0.1	-0.0	-0.8	0.3	0.1	0.1	0.1	-0.3
2B	0.0	-0.1	-0.1	-0.1	-0.8	0.4	0.1	0.1	-0.3
3B	0.0	-0.0	-0.0	-0.0	-0.0	-0.0	-0.0	-0.9	-0.3
HR	-0.0	-0.1	-0.0	0.0	-0.2	-0.9	0.2	0.1	-0.3
% Variance explained	52.9	32.7	6.7	4.9	1.5	0.6	0.3	0.2	0.0

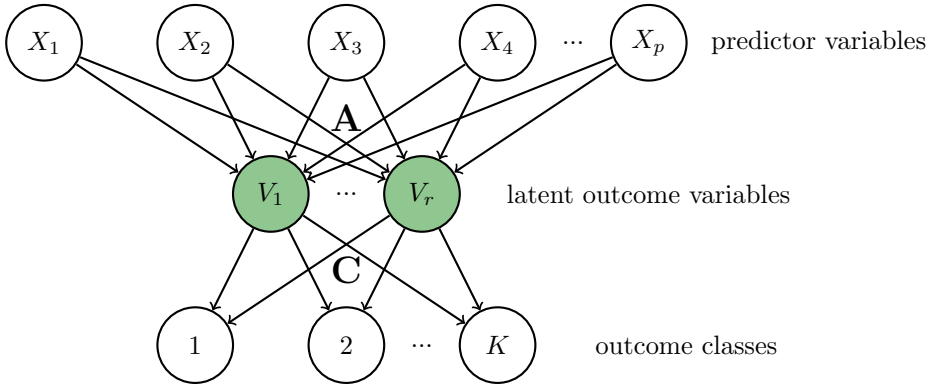
(b) Principal components of pitcher outcome matrix

Figure 2.3: Visualization of principal component analysis of player outcome matrices, separate for batters and for pitchers. The player outcome matrix has a row for each player giving the proportion of PAs which have resulted in each of the nine outcomes in the dataset. The visualization shows the loadings for each PC, along with a green bar plot corresponding to the percentage of variance explained by each PC, which is also printed in the row below the matrix of PC loadings.

matrix by $\mathbf{B} = (\beta_1, ..., \beta_K)$. Motivated by the principal component analysis from Section 2.1, instead of learning a coefficient vector for each class, we might do better by learning a coefficient vector for each of a smaller number r of latent variables, each having a loading on the classes. For $r = 1$, this is the *stereotype model* originally proposed by Anderson [1984], who observed its applicability to multinomial regression problems with structure between the response categories, including ordinal structure. Greenland [1994] argued for the stereotype model as an alternative in medical applications to the standard techniques for ordinal categorical regression: the cumulative-odds and continuation-ratio models.

Yee and Hastie [2003] generalized the model to reduced-rank vector generalized linear models. In detail, the reduced-rank multinomial logistic model (RR-MLM) fits (2.1) by solving, for some

Figure 2.4: Illustration of reduced-rank multinomial regression. If the rank of the regression coefficient matrix $\mathbf{B}$ is $r < K$, then $\mathbf{B} \in \mathbb{R}^{p\times K}$ may be written as $\mathbf{B} = \mathbf{A}\mathbf{C}^T$ with $\mathbf{A} \in \mathbb{R}^{p\times r}$ giving regression coefficients for r latent outcome variables and $\mathbf{C} \in \mathbb{R}^{p\times K}$ defining these latent outcome variables with loadings on each of the outcome classes.



$r \in \{1, ..., K-1\}$, the optimization problem:

$$\begin{aligned} \underset{\alpha\in\mathbb{R}^K,\ \mathbf{B}\in\mathbb{R}^{p\times K}}{\text{minimize}} \; & -\sum_{i=1}^{n} \log\left(\sum_{k=1}^{K} \frac{e^{\alpha_k + \mathbf{x}_i^T\beta_k}}{\sum_{\ell=1}^{K} e^{\alpha_\ell + \mathbf{x}_i^T\beta_\ell}} \mathbb{I}_{\{Y_i=k\}}\right) \\ \text{subject to} \quad & \text{rank}(\mathbf{B}) \leq r \\ & \alpha_1 = 0, \beta_1 = \vec{0}_p. \end{aligned} \tag{2.2}$$

If $\text{rank}(\mathbf{B}) < r$, then there exist $\mathbf{A} \in \mathbb{R}^{p\times r}$, $\mathbf{C} \in \mathbb{R}^{K\times r}$ such that $\mathbf{B} = \mathbf{A}\mathbf{C}^T$. Under this factorization, the r columns of $\mathbf{C}$ can be interpreted as defining latent outcome variables, each with a loading on each of the K outcome classes. The r columns of $\mathbf{A}$ give regression coefficient vectors for these latent outcome variables, rather than the outcome classes. See Figure 2.4.

The optimization problem (2.2) is not convex because $\text{rank}(\cdot)$ is not a convex function. Yee [2010] implemented an alternating algorithm to solve it in the R [R Core Team, 2016] package `VGAM`. However this algorithm is too slow for feasible application to datasets as large as the one motivating Section 2.1 ($n = 176559$, $p = 796$, $K = 9$).

2.3 Nuclear penalized multinomial regression

Because of the computational difficulty of solving (2.2), we propose replacing the rank restriction with a restriction on the nuclear norm $||\cdot||_*$ (defined below) of the regression coefficient matrix. For

some $\rho > 0$, this convex optimization problem is:

$$\begin{aligned} &\underset{\alpha\in\mathbb{R}^K,\ \mathbf{B}\in\mathbb{R}^{p\times K}}{\text{minimize}} \; -\sum_{i=1}^{n}\log\left(\sum_{k=1}^{K}\frac{e^{\alpha_k+\mathbf{x}_i^T\beta_k}}{\sum_{\ell=1}^{K}e^{\alpha_\ell+\mathbf{x}_i^T\beta_\ell}}\mathbb{I}_{\{Y_i=k\}}\right) \\ &\text{subject to} \quad ||\mathbf{B}||_* \leq \rho \end{aligned} \tag{2.3}$$

However we prefer to frame the problem in its equivalent Lagrangian form: For some $\lambda > 0$,

$$\begin{aligned} (\alpha^*, \mathbf{B}^*) &= \underset{\alpha\in\mathbb{R}^K,\ \mathbf{B}\in\mathbb{R}^{p\times K}}{\arg\min} \; -\sum_{i=1}^{n}\log\left(\sum_{k=1}^{K}\frac{e^{\alpha_k+\mathbf{x}_i^T\beta_k}}{\sum_{\ell=1}^{K}e^{\alpha_\ell+\mathbf{x}_i^T\beta_\ell}}\mathbb{I}_{\{Y_i=k\}}\right) + \lambda||\mathbf{B}||_* \\ &\equiv \underset{\alpha\in\mathbb{R}^K,\ \mathbf{B}\in\mathbb{R}^{p\times K}}{\arg\min} \; -\ell(\alpha, \mathbf{B}; \mathbf{X}, Y) + \lambda||\mathbf{B}||_* \end{aligned} \tag{2.4}$$

This optimization problem (2.4) is what we call nuclear penalized multinomial regression (NPMR). We use $\ell(\alpha, \mathbf{B}; \mathbf{X}, Y)$ to denote the log-likelihood of the regression coefficients α and $\mathbf{B}$ given the data $\mathbf{X}$ and Y. The nuclear norm of a matrix is defined as the sum of its singular values, that is, the ℓ_1-norm of its vector of singular values. Explicitly, consider the singular value decomposition of $\mathbf{B}$ given by $\mathbf{U}\mathbf{\Sigma}\mathbf{V}^T$, with $\mathbf{U} \in \mathbb{R}^{p\times p}$ and $\mathbf{V} \in \mathbb{R}^{K\times K}$ orthogonal and $\mathbf{\Sigma} \in \mathbb{R}^{p\times K}$ having values $\sigma_1, ..., \sigma_{\min\{p,K\}}$ along the main diagonal and zeros elsewhere. Then

$$||\mathbf{B}||_* = \sum_{d=1}^{\min\{p,K\}} \sigma_d.$$

In the same way that the lasso induces sparsity of the estimated coefficients in a regression, solving (2.4) drives some of the singular values to exactly zero. Because the number of nonzero singular values is the rank of a matrix, the result is that the estimated coefficient matrix $\mathbf{B}^*$ tends to have less than full rank. Thus (2.4) is a convex relaxation of the reduced-rank multinomial logistic regression problem, in much the same way as the lasso is a convex relaxation of best subset regression [Tibshirani, 1996]. The convexity of (2.4) makes it easier to solve than (2.2), and we discuss algorithms for solving it in Sections 2.3.1 and 2.3.2.

Consider the singular value decomposition $\mathbf{U}^*\mathbf{\Sigma}^*\mathbf{V}^{*T}$ of the $p \times K$ estimated coefficient matrix $\mathbf{B}^*$. Each column of the $K \times K$ orthogonal matrix $\mathbf{V}^*$ represents a latent variable as a linear combination of the K outcome categories. Meanwhile, each row of $\mathbf{U}^*\mathbf{\Sigma}^*$ specifies for each predictor variable a coefficient for each *latent* variable, rather than for each outcome category. By estimating some of the singular values of $\mathbf{B}^*$ (the entries of the diagonal $p \times K$ matrix $\mathbf{\Sigma}^*$) to be zero, we reduce the number of coefficients to be estimated for each predictor variable from (a) one for each of K outcome categories; to (b) one for each of some smaller number of latent variables. These latent variables learned by the model express relationships between the outcomes because two categories for which a latent variable has both large positive coefficients are both likely to occur for large

values of that latent variable. Similarly, if a latent variable has a large positive coefficient for one category and a large negative coefficient for another category, those two categories oppose each other diametrically with respect to that latent variable.

2.3.1 Proximal gradient descent

NPMR relies on solving (2.4). The objective is convex but non-differentiable where any singular values of $\mathbf{B}$ are zero, so we cannot use gradient descent. Generally, when minimizing a function $f : \mathbb{R}^d \to \mathbb{R}$ of a vector $\mathbf{x} \in \mathbb{R}^d$, the gradient descent update of step size s takes the form

$$\mathbf{x}^{(t+1)} = \mathbf{x}^{(t)} - s\nabla f(\mathbf{x}^{(t)}),$$

or equivalently,

$$\mathbf{x}^{(t+1)} = \underset{\mathbf{x}\in\mathbb{R}^d}{\arg\min} \left\{ f(\mathbf{x}^{(t+1)}) + \langle \nabla f(\mathbf{x}^{(t)}), \mathbf{x} - \mathbf{x}^{(t)} \rangle + \frac{1}{2s^{(t)}} ||\mathbf{x} - \mathbf{x}^{(t)}||_2^2 \right\}.$$

Still, if f is non-differentiable, as it is in (2.4), then ∇f is undefined. However, if f is the sum of two convex functions g and h, with g differentiable, we can instead apply the generalized gradient update step [Hastie et al., 2015]:

$$\mathbf{x}^{(t+1)} = \underset{\mathbf{x}\in\mathbb{R}^d}{\arg\min} \left\{ g(\mathbf{x}^{(t+1)}) + \langle \nabla g(\mathbf{x}^{(t)}), \mathbf{x} - \mathbf{x}^{(t)} \rangle + \frac{1}{2s^{(t)}} ||\mathbf{x} - \mathbf{x}^{(t)}||_2^2 + h(\mathbf{x}) \right\}. \tag{2.5}$$

Repeatedly applying this update is known as proximal gradient descent (PGD). In (2.4), we have $\mathbf{x} = (\alpha, \mathbf{B})$, $g = -\ell$ and $h = ||\cdot||_*$. So the PGD update step is:

$$\begin{aligned} \alpha^{(t+1)},\ \mathbf{B}^{(t+1)} = \underset{\alpha,\ \mathbf{B}}{\arg\min} \Big\{ & -\ell(\alpha^{(t)}, \mathbf{B}^{(t)}; \mathbf{X}, \mathbf{Y}) \\ & + \langle \mathbf{X}^T(\mathbf{Y} - \mathbf{P}^{(t)}), \mathbf{B} - \mathbf{B}^{(t)} \rangle + \langle 1_n^T(\mathbf{Y} - \mathbf{P}^{(t)}), \alpha - \alpha^{(t)} \rangle \\ & + \frac{1}{2s} ||\mathbf{B} - \mathbf{B}^{(t)}||_F^2 + \frac{1}{2s}\ ||\alpha - \alpha^{(t)}||_2^2 + \lambda ||\mathbf{B}||_* \Big\}, \end{aligned}$$

where $\mathbf{Y} \in \{0,1\}^{n\times K}$ is the matrix containing the response variable and $\mathbf{P} \in (0,1)^{n\times K}$ is the matrix containing the fitted values. That is, for $i = 1, ..., n$, and $k = 1, ..., K$,

$$\{\mathbf{Y}\}_{ik} = \mathbb{I}_{\{Y_i = k\}}, \quad \text{and} \quad \{\mathbf{P}\}_{ik} = \frac{e^{\alpha_k + \mathbf{x}_i^T \beta_k}}{\sum_{\ell=1}^K e^{\alpha_\ell + \mathbf{x}_i^T \beta_\ell}}. \tag{2.6}$$

The problem is separable in α and $\mathbf{B}$:

$$\begin{aligned}\alpha^{(t+1)} &= \arg\min_{\alpha}\left\{\langle 1_n^T(\mathbf{Y}-\mathbf{P}^{(t)}), \alpha-\alpha^{(t)}\rangle + \frac{1}{2s}||\alpha-\alpha^{(t)}||_2^2\right\} \\ &= \alpha^{(t)} + s1_n^T(\mathbf{Y}-\mathbf{P}^{(t)}),\end{aligned} \tag{2.7}$$

and

$$\begin{aligned}\mathbf{B}^{(t+1)} &= \arg\min_{\mathbf{B}}\left\{\langle \mathbf{X}^T(\mathbf{Y}-\mathbf{P}^{(t)}), \mathbf{B}-\mathbf{B}^{(t)}\rangle + \frac{1}{2s}||\mathbf{B}-\mathbf{B}^{(t)}||_F^2 + \lambda||\mathbf{B}||_*\right\} \\ &= \mathcal{S}^*_{s\lambda}(\mathbf{B}^{(t)} + s\mathbf{X}^T(\mathbf{Y}-\mathbf{P}^{(t)})),\end{aligned} \tag{2.8}$$

where $\mathcal{S}^*_{s\lambda} : \mathbb{R}^{p\times K} \to \mathbb{R}^{p\times K}$ is the soft-thresholding operator on the singular values of a matrix. Explicitly, if a matrix $\mathbf{M} \in \mathbb{R}^{p\times K}$ has singular value decomposition $\mathbf{U\Sigma V}^T$, then $\mathcal{S}^*_{s\lambda}(\mathbf{M}) = \mathbf{U}\mathcal{S}_{s\lambda}(\mathbf{\Sigma})\mathbf{V}^T$, where

$$\{\mathcal{S}_{s\lambda}(\mathbf{\Sigma})\}_{jk} = \text{sign}(\mathbf{\Sigma}_{jk})\max\{|\mathbf{\Sigma}_{jk}| - s\lambda, 0\}.$$

$\mathcal{S}^*_{s\lambda}$ is called the proximal operator of the nuclear norm, and in general solving (2.5) involves the proximal operator of the function h, hence the name proximal gradient descent.

So to solve (2.4), initialize α and $\mathbf{B}$, and iteratively apply the updates (2.7) and (2.8). Due to Nesterov [2007], this procedure converges with step size $s \in (0, 1/L)$ if the log-likelihood ℓ is continuously differentiable and has Lipschitz gradient with Lipschitz constant L. We show in the appendix that the gradient of ℓ is Lipschitz with constant $L = \sqrt{K}||\mathbf{X}||_F^2$, but in practice we recommend starting with step size $s = 0.1$ and halving the step size if any proximal gradient descent step would result in an increase of the objective function (2.4).

2.3.2 Accelerated PGD

In practice, we find that it helps to speed things up considerably to use an accelerated PGD method, also due to Nesterov [2007]. Specifically, we iteratively apply the following updates:

1. $\alpha^{(t+1)} = \alpha^{(t)} + s1_n^T\left(\mathbf{Y}-\mathbf{P}^{(t)}\right)$
2. $\mathbf{A}^{(t+1)} = \mathbf{B}^{(t)} + \frac{t}{t+3}(\mathbf{B}^{(t)} - \mathbf{B}^{(t-1)})$
3. $\mathbf{P}^{(t+1)} = \mathbf{P}(\alpha^{(t+1)}, A^{(t+1)})$
4. $\mathbf{B}^{(t+1)} = \mathcal{S}^*_{s\lambda}\left(\mathbf{A}^{(t+1)} + s\mathbf{X}^T\left(\mathbf{Y}-\mathbf{P}^{(t+1)}\right)\right)$

The function $\mathbf{P}(\cdot)$ in Step 3 returns the matrix of fitted probabilities based on the regression coefficients as described in (2.6). Step 2 is the key to the acceleration because it uses the "momentum" in $\mathbf{B}$ to push it further in the same direction it is heading. We strongly recommend using this accelerated version of PGD as we have found in practice that it leads to much quicker results.

and our implementation of NPMR is available on the Comprehensive R Archive Network as the R package `npmr`.

2.3.3 Software

NPMR is implemented in the R package `npmr`, available on the Comprehensive R Archive Network. The primary function is `cv.npmr`, which uses cross validation to determine the optimal regularization parameter lambda and returns a fitted `cv.npmr` object, which has its own `plot` and `predict` methods. The function call is shown below.

```
cv.npmr(X, Y, lambda = exp(seq(7, -2)), s = 0.1/max(X), eps = 1e-06,
  group = NULL, accelerated = TRUE, B.init = NULL, b.init = NULL,
  foldid = NULL, nfolds = 10)
```

2.3.4 Related work

The idea of using a nuclear norm penalty as a convex relaxation to reduced-rank regression has previously been proposed in the Gaussian regression setting [Chen et al., 2013], but we are not aware of any attempt to do so in the multinomial setting.

The nearest competitor to NPMR that can feasibly be applied to the baseball matchup dataset is multinomial ridge regression, which penalizes the squared Frobenius norm (the sum of the squares of the entries) of the coefficient matrix, instead of the nuclear norm. In detail, ridge regression estimates the regression coefficients by solving the optimization problem

$$(\alpha^*, \mathbf{B}^*) = \underset{\alpha \in \mathbb{R}^K,\ B \in \mathbb{R}^{p \times K}}{\arg\min} - \ell(\alpha^{(t)}, \mathbf{B}^{(t)}; \mathbf{X}, Y) + \lambda ||\mathbf{B}||_F^2. \tag{2.9}$$

This model is very similar to the state of the art in public sabermetric literature for evaluating pitchers on the basis of outcomes while simultaneously controlling for sample size, opponent strength and ballpark effects [Judge and BP Stats Team, 2015]. Software is available to solve this problem very quickly in the R package `glmnet` [Friedman et al., 2010]. This is the standard approach used for regularized multinomial regression problems, so we use it as the benchmark against which to compare the performance of NPMR in Sections 2.4 and 2.5.

2.4 Simulation study

In this section we present the results of two different simulations, one using a full-rank coefficient matrix and the other using a low-rank coefficient matrix. In both settings we vary the training sample size n from 600 to 2000, and we fix the number of predictor variables to be 12 and the number of levels of the response variable to be 8. Given design matrix $\mathbf{X} \in \mathbb{R}^{n \times 12}$ and coefficient matrix

$\mathbf{B} \in \mathbb{R}^{12\times 8}$, we simulate the response according to the multinomial regression model. Explicitly, for $i = 1, ..., n$, and $k = 1, ..., 8$,

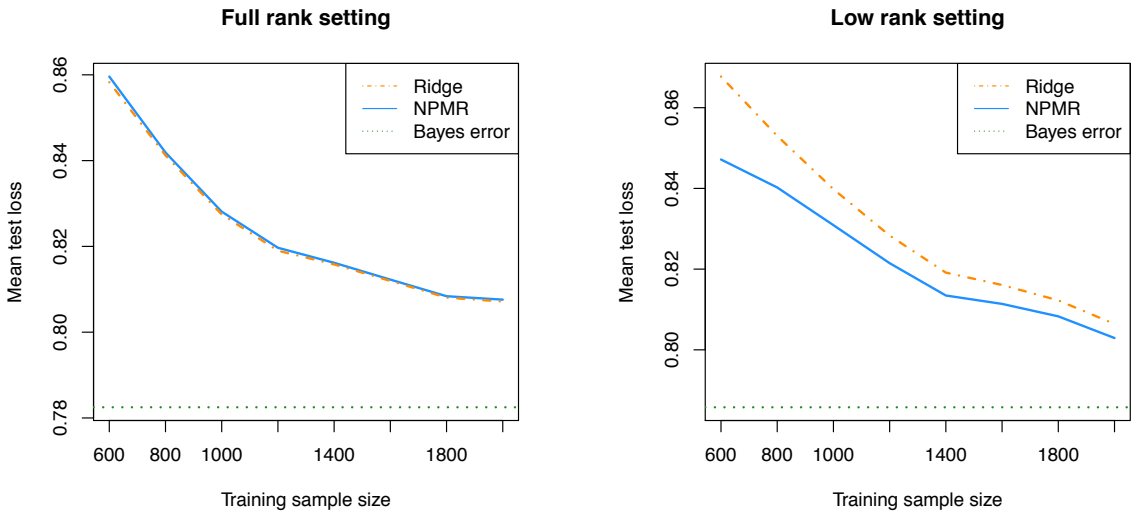
$$\mathbb{P}(Y_i = k) = \frac{e^{\mathbf{X}\beta_k}}{\sum_{\ell=1}^{8} e^{\mathbf{X}\beta_\ell}}.$$

For both simulations the entries of $\mathbf{X}$ follow an i.i.d. standard normal distribution,

$$\mathbf{x}_i \overset{\text{i.i.d.}}{\sim} \text{Normal}(\vec{0}_{12}, I_{12})$$

for $i = 1, ..., n$. However the simulations differ in the generation of the coefficient matrix $\mathbf{B}$. In the *full rank* setting, the entries of $\mathbf{B}$ follow an i.i.d. standard normal distribution: For $k = 1, ..., 8$,

$$\beta_k \overset{\text{i.i.d.}}{\sim} \text{Normal}(\vec{0}_{12}, I_{12}). \tag{2.10}$$



(a) Full-rank coefficient matrix

(b) Low-rank coefficient matrix

Figure 2.5: Simulation results. We plot out-of-sample test error (using log-likelihood loss) against training sample size. The Bayes error is a lower bound on achievable test error. In (a), the full rank setting, ridge regression out-performs NPMR in terms of test error by a slim margin. In (b), the low rank setting, NPMR beats ridge regression, especially for smaller sample sizes.

In the *low rank* setting we first simulate two intermediary matrices $\mathbf{A} \in \mathbb{R}^{12\times 2}$ and $\mathbf{C} \in \mathbb{R}^{8\times 2}$ with i.i.d. standard normal entries, and we then define $\mathbf{B} \equiv \mathbf{A}\mathbf{C}^T$ so that the rank of $\mathbf{B}$ is 2. In each simulation we fit ridge regression and NPMR to the training sample of size n and estimate the out-of-sample error by simulating 10,000 test observations, comparing the model's predictions on those test observations with the simulated response. The results of 3,500 simulations in each setting,

for each training sample size n, are presented in Figure 2.5.

In the full rank setting we expect ridge regression to out-perform NPMR because ridge regression shrinks all coefficient estimates toward zero, which is the mean of the generating distribution for the coefficients in the simulation. If this were a Gaussian regression problem instead of a multinomial regression problem, then the ridge regression coefficient estimates would correspond [Hastie et al., 2009] to the posterior mean estimate under a Bayesian prior of (2.10). In fact ridge regression does beat NPMR in this simulation (for all training sample sizes n), but NPMR's performance is surprisingly close to that of ridge regression.

The low rank setting is one in which NPMR should lead to a lower test error than does ridge regression. NPMR bets on sparsity in the singular values of the coefficient matrix, and in this setting the bet pays off. The simulation results verify that this intuition is correct. NPMR beats ridge regression for all training sample sizes n but especially for smaller sample sizes. By betting (correctly in this case) on the coefficient matrix having less than full rank, NPMR learns more accurate estimates of the coefficient matrix. As the training sample size increases, learning the coefficient matrix becomes easier, and the performance gap between the two methods shrinks but remains evident.

In summary, this simulation demonstrates that each of NPMR and ridge regression is superior in a simulation tailored to its strengths, confirming our intuition. Furthermore, in a simulation constructed in favor of ridge regression, NPMR performs nearly as well. Meanwhile NPMR leads to more significant gains over ridge regression in the low rank setting.

2.5 Results

2.5.1 Implementation details

The 2015 MLB play-by-play dataset from Retrosheet includes an entry for every plate appearance during the six-month regular season. For the purposes of fitting NPMR to predict the outcomes of PAs, the following relevant variables are recorded for the i^{th} PA: the identity (B_i) of the batter; the identity (P_i) of the pitcher; the identity (S_i) of the stadium where the PA took place; an indicator (H_i) of whether the batter's team is the home team; and finally an indicator (O_i) of whether the batter's handedness (left or right) is opposite that of the pitcher.

For each outcome $k \in \mathcal{K} \equiv \{\text{K, G, F, BB, HBP, 1B, 2B, 3B, HR}\}$, the multinomial model fit by both NPMR and ridge regression is specified by

$$\mathbb{P}(Y_i = k) = \frac{e^{\eta_{ik}}}{\sum_{\ell \in \mathcal{K}} e^{\eta_{i\ell}}}, \text{ where}$$
$$\eta_{ik} = \alpha_k + \beta_{B_i k} + \gamma_{P_i k} + \delta_{S_i k} + \zeta_k H_i + \theta_k O_i.$$

The parameters introduced have the following interpretation: α_k is an intercept corresponding to the league-wide frequency of outcome k; $\beta_{B_i k}$ corresponds to the tendency of batter B_i to produce outcome k; $\gamma_{P_i k}$ corresponds to the tendency of pitcher P_i to produce outcome k; $\delta_{S_i k}$ corresponds to the tendency of stadium S_i to produce outcome k; ζ_k corresponds to the increase in likelihood of outcome k due to home field advantage; and θ_k corresponds to the increase in likelihood of outcome k due to the batter having the opposite handedness of the pitcher's.

NPMR and ridge regression fit the same multinomial regression model and differ only in the regularizations used in their objective functions, yielding different results. See Section 2.3 for details. However there is a minor tweak to NPMR for application to these data. Instead of adding to the objective a penalty on the nuclear norm of the whole coefficient matrix, we add penalties on the nuclear norms of the three coefficient sub-matrices corresponding to batters, pitchers and stadiums. The coefficients for home-field advantage and opposite handedness remain unpenalized. The result is that NPMR learns different latent variables for batters than it does for pitchers, instead of learning one set of latent variables for both groups.

We process the PA data before applying NPMR and ridge regression. First, we define a minimum PA threshold separately for batters and pitchers. For batters the threshold is the 390^{th}-largest number of PAs among all batters. This corresponds roughly to the number of rostered batters at any given time during the MLB regular season. Batters who fall below the PA threshold are labelled "replacement level" and within each defensive position are grouped together into a single identity. For example, "replacement-level catcher" is a batter in the dataset just like Mike Trout is, and the former label includes all PAs by a catcher who does not meet the PA threshold. Similarly we define the PA threshold for pitchers to be the 360^{th}-largest number of PAs among all pitchers, and we group all pitchers who fall below that threshold under the "replacement-level pitcher" label. Additionally, we discard all PAs in which a pitcher is batting, and we discard PAs which result in a catcher's interference or an intentional walk. The result is a set of 176,559 PAs featuring 401 unique batters and 362 unique pitchers in 30 unique stadiums.

2.5.2 Validation

We fit NPMR and ridge regression to the baseball data, using a training sample that varied from 5% (roughly 9,000 PAs) to 75% (roughly 135,000 PAs) of the data. In each case we used the remaining data to test the models, with multinomial deviance (twice the negative log-likelihood) as the loss function. For comparison we also include the null model, which predicts for each plate appearance the league average probabilities of each outcome. Figure 2.6 gives the results.

For each training sample size, NPMR outperforms ridge regression though the difference is not statistically significant. At the smallest sample size NPMR, unlike ridge regression, achieves a significantly lower error than the null estimator. There is value in improved estimation of players' skills in small sample sizes because this can inform early-season decision-making. For all other sample

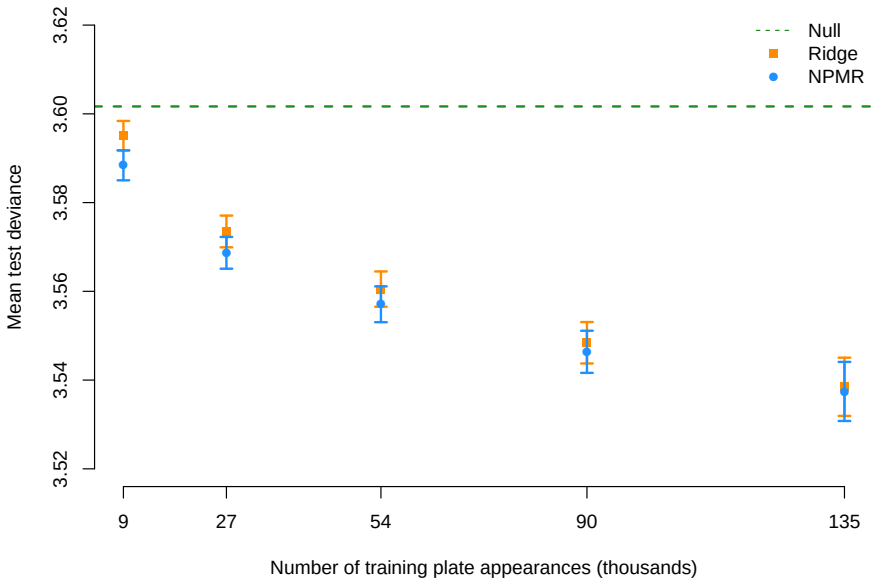


Figure 2.6: Out-of-sample test performance of NPMR, ridge and null estimators on baseball plate appearance result prediction. Each estimator was trained on a fraction of the 2015 regular season data (varying from 5 to 75 percent) and tested on the remaining data. The error bars correspond to one standard error.

sizes, both NPMR and ridge regression achieves errors which are statistically significantly less than the null error. The primary benefit of NPMR relative to ridge regression is the interpretation, as described in the next section.

2.5.3 Interpretation

We focus on the results of fitting NPMR on 5 percent of the training data because there the difference between NPMR and ridge regression is greatest (Figure 2.6). As the sample size increases, the need for a low-rank regression coefficient matrix is reduced, and the NPMR solution becomes more similar to the ridge solution. Figure 2.7 visualizes the singular value decomposition of the fitted regression coefficient submatrices corresponding to batters and pitchers.

We observe that for both batters and pitchers, NPMR identifies three latent variables which differentiate players from one another. By construction, these latent variables are measuring separate aspects of players' skills; across players, expression in each latent skill is uncorrelated with expression in each other latent skill. In that sense, we have identified three separate skills which characterize hitters and three separate skills which characterize pitchers. In baseball scouting parlance, these

skills are called "tools", but unlike the five traditional baseball tools (hitting for power, hitting for contact, running, fielding and throwing), the tools we identify are uncorrelated with one another.

Latent variable	1	2	3	4	5	6	7	8	9
1B	0.38	-0.28	-0.68	0.42	-0.14	-0.07	0.34	-0.03	-0.03
2B	0.03	-0.02	-0.06	-0.46	0.03	-0.77	0.31	0.26	0.17
3B	0.01	-0.00	-0.00	-0.27	0.16	0.09	0.31	0.00	-0.89
BB	-0.16	-0.10	-0.06	-0.45	-0.40	0.31	0.42	-0.52	0.24
F	0.14	0.87	0.09	0.25	-0.12	-0.07	0.35	-0.09	0.02
G	0.43	-0.36	0.72	0.22	-0.12	-0.02	0.33	0.02	0.03
HBP	-0.01	-0.01	-0.03	-0.01	0.85	0.22	0.36	-0.09	0.31
HR	-0.04	0.05	-0.06	-0.14	-0.19	0.47	0.23	0.80	0.14
K	-0.79	-0.15	0.09	0.45	-0.07	-0.17	0.33	0.06	-0.06
Corresponding diagonal	3.66	2.20	1.23	0.00	0.00	0.00	0.00	0.00	0.00

(a) Latent variables for batter regression coefficient matrix

Latent variable	1	2	3	4	5	6	7	8	9
1B	0.16	0.24	-0.34	0.48	-0.46	-0.27	0.42	-0.34	0.05
2B	0.01	0.03	-0.01	0.57	0.71	0.23	0.27	0.00	-0.20
3B	-0.00	-0.01	-0.05	-0.17	-0.12	0.38	-0.14	-0.61	-0.65
BB	0.07	-0.04	-0.69	-0.46	0.12	0.23	0.43	0.22	-0.01
F	0.37	-0.74	0.33	-0.01	-0.14	0.07	0.41	-0.04	0.00
G	0.26	0.62	0.51	-0.27	-0.03	0.19	0.42	0.07	-0.01
HBP	-0.01	0.01	0.00	0.19	-0.31	-0.10	-0.00	0.65	-0.66
HR	0.01	-0.00	0.05	-0.30	0.35	-0.79	0.16	-0.19	-0.31
K	-0.87	-0.09	0.18	-0.03	-0.13	0.05	0.42	-0.05	0.00
Corresponding diagonal	1.98	1.54	0.32	0.00	0.00	0.00	0.00	0.00	0.00

(b) Latent variables for pitcher regression coefficient matrix

Figure 2.7: Visualization of fitted regression coefficient matrices from NPMR on 5% of the baseball data. The matrix displayed is $\mathbf{V}$ in the $\mathbf{U\Sigma V}^T$ decomposition of $\mathbf{B}$ from (2.4), with columns corresponding to latent variables and rows corresponding to outcomes. The bottom row gives the entry in the diagonal matrix $\mathbf{\Sigma}$ corresponding to the latent variable.

The interpretation of Figure 2.7 is very attractive in the context of domain knowledge. In reading the columns of $\mathbf{V}$, note that they are unique only up to a change in sign, so we can taken positive expression of each skill to mean positive or negative values of the corresponding latent variable. We suggest the following interpretation of the first three latent skills for batters:

- *Skill 1: Patience.* The loadings of the first latent variable discriminate perfectly between the TTO outcomes and the BIP outcomes described in Section 2.1. We label this skill as "patience" because when a batter swings at fewer pitches, he is less likely to hit the ball in play.
- *Skill 2: Trajectory.* The second latent variable distinguishes primarily between F and G, corresponding to the vertical launch angle of the ball off the bat.
- *Skill 3: Speed.* The third latent variable distinguishes primarily between 1B and G. Examining the players with strong positive expression of this skill, we find fast players who are more difficult to throw out at first base on a ground ball.

From this interpretation we learn that the primary skill which distinguishes between batters is how often they hit the ball into the field of play. One outcome over which batters have relatively

large control is how often they swing at pitches. Among balls that are put into play, batters have less but still substantial control over whether those are ground balls or fly balls. It is the vertical angle of the batter's swing plane, along with whether he tends to contact the top half or the bottom half of the ball, that determines his trajectory tendency. Finally, given the trajectory of the ball off the bat, the batter has relatively little control over the outcome of the PA. But to the extent that he can influence this outcome, fast runners tend to hit more singles and fewer groundouts.

We suggest the following interpretation of the pitchers' skills based on Figure 2.7:

- *Skill 1: Power.* The first latent variable distinguishes primarily between K and F (and G), thus identifying how the pitcher gets outs. Pitchers who tend to get their outs via the strikeout are known in baseball as "power pitchers".
- *Skill 2: Trajectory.* As with batters, the second latent variable distinguishes primarily between F and G, corresponding to the trajectory of the ball off the bat.
- *Skill 3: Command.* The third latent variable distinguishes primarily between positive outcomes for the pitcher (F, G and K) and negative outcomes for the pitcher (BB and 1B), reflecting how well is able to control the location of his pitches.

The interpretation of the first two skills for pitchers is very similar to the interpretation of the first two skills for batters. Primarily, pitchers can influence how often balls are hit into play against them, but they exhibit less control over this than batters do. Secondarily, as with hitters, pitchers exhibit some control over the vertical launch angle of the ball off the bat. This is based on the location and movement of their pitches. The third skill, distinguishing between positive and negative outcomes, has a relatively very small magnitude.

Table 2.1 lists the top five and bottom five players in each of the three latent batting skills learned by NPMR. These results largely match intuition for the players listed, and to the extent that they do not, it is worth a reminder that they are based on roughly nine days' worth of data from a season which is six months long. The median number of PAs per batter in the training set is 21.

Table 2.1: Top 5 and bottom 5 batters in each of the three latent skills identified by NPMR.

Skill	Patience	Trajectory	Speed
	More K, BB	More F	More 1B
Top 5	Peter Bourjos	Ian Kinsler	Yoenis Cespedes
	Eddie Rosario	Freddie Freeman	Lorenzo Cain
	Carlos Santana	Omar Infante	José Iglesias
	George Springer	Kolten Wong	Kevin Kiermaier
	Mike Napoli	José Altuve	Delino DeShields Jr
Bottom 5	Josh Reddick	Dee Gordon	Evan Longoria
	JT Realmuto	Alex Rodriguez	Ryan Howard
	AJ Pollock	Cameron Maybin	Odubel Herrera
	Kevin Pillar	Shin-Soo Choo	Seth Smith
	Eric Aybar	Francisco Cervelli	Jake Lamb
	More F, G, 1B	More G, 1B	More G

The results in Table 2.2, listing the top and bottom players in each of the three latent pitching skills, are more interesting. The top five power pitchers are all among to top starting pitchers in

the game. All the way on the other side of the spectrum is knuckleball pitcher RA Dickey. The knuckleball is a unique pitch in baseball thrown relatively softly with as little spin as possible to create unpredictable movement. Its goal is not to overpower the opposing batter but to induce weak contact. Another interesting pitcher low on power is Cole Hamels. Two of the leading sabermetric websites, Baseball Prospectus and FanGraphs, disagree greatly on Hamels' value. The discrepancy stems from Baseball Prospectus giving full weight to BIP outcomes while FanGraphs ignores them. Because Hamels tends to get outs via fly balls and ground balls rather than strikeouts, FanGraphs estimates a much lower value for Hamels than Baseball Prospectus does.

Table 2.2: Top 5 and bottom 5 pitchers in each of the three latent skills identified by NPMR.

Tool	Power	Trajectory	Command
	More K	More F	More F, G, K
Top 5	José Quintana	Jesse Chavez	Max Scherzer
	Corey Kluber	Justin Verlander	Masahiro Tanaka
	Madison Bumgarner	Jake Peavy	Jacob deGrom
	Max Scherzer	Johnny Cueto	Rubby de la Rosa
	Clayton Kershaw	Chris Young	Matt Harvey
Bottom 5	John Danks	Dallas Keuchel	Mike Pelfrey
	Dan Haren	Garrett Richards	Chris Tillman
	Cole Hamels	Sam Dyson	Eddie Butler
	Alfredo Simón	Brett Anderson	Gio Gonzalez
	RA Dickey	Michael Pineda	Jeff Samardzija
	More F, G	More G	More BB, 1B

The results for all players are summarized in Figure 2.8. The biplots show players as blue circles and latent skills as black arrows. For batters we observe that (1) high patience corresponds to low G and F rates; (2) high trajectory corresponds to low G rate and high F rate; and (3) high speed corresponds to low G rate and has little relationship with F rate. For pitchers we observe that (a) high power corresponds to low G rate and high K rate; (b) high trajectory corresponds to low G rate and has relatively little influence on K rate; and (c) high command corresponds to high G and F rates, but with very small magnitude.

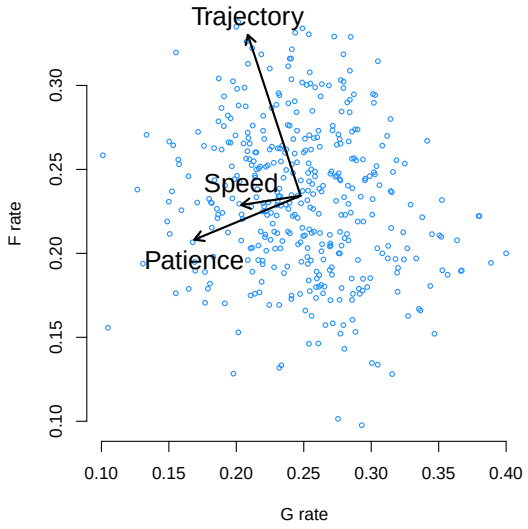
2.5.4 Another application: Vowel data

Consider the problem of vowel classification from Robinson [1989]. The data set comprises 528 training samples and 462 test samples, each classified as one of the 11 vowels listed in Table 2.3, with 10 features extracted from an audio file. The data are grouped by speaker, with 8 subjects in the training set and 7 different subjects in the test set. Each audio clip is split into 6 frames during a duration of steady audio, yielding 6 pseudo-replicates.

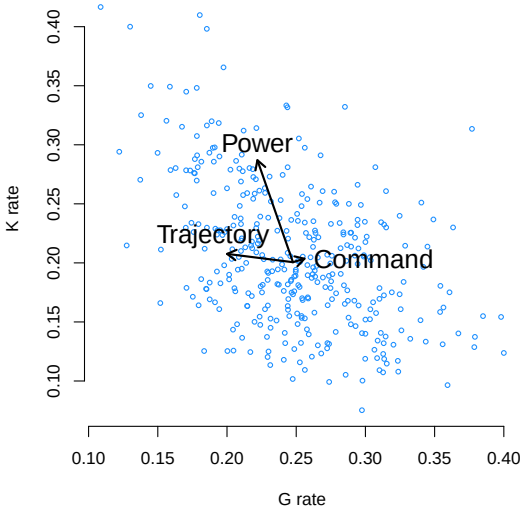
Table 2.3: Symbols and corresponding words for 11 vowels studied in data set from Robinson [1989].

Vowel	Word	Vowel	Word	Vowel	Word	Vowel	Word
i	heed	A	had	O	hod	u:	who'd
I	hid	a:	hard	C:	hoard	3:	heard
E	head	Y	hud	U	hood		

We fit NPMR and ridge regression to the training data over a wide range of regularization



(a) Plot of batters (orange) with latent skills superimposed (blue).



(b) Plot of pitchers (orange) with latent skills superimposed (blue).

Figure 2.8: Biplots for batters and pitchers. Each plot shows the rates of two different outcomes plotted against each other. The black arrows show the relationship between the latent skills and the outcome rates.

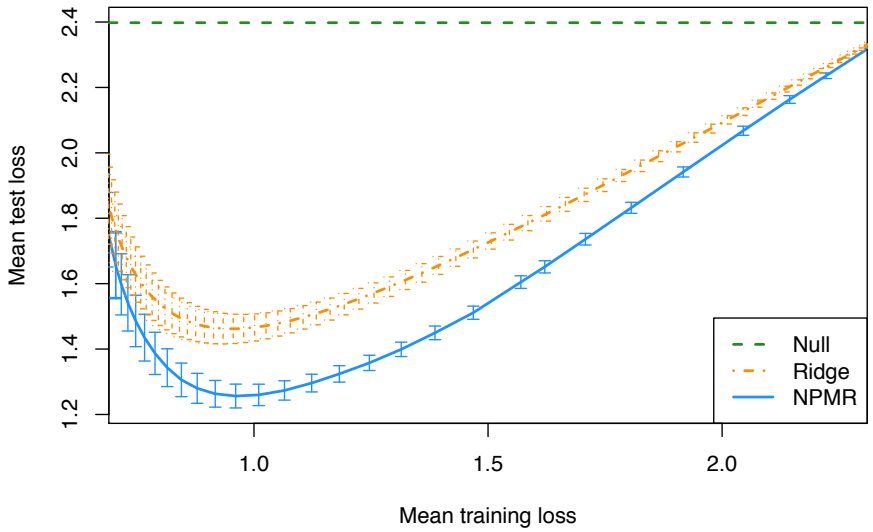
parameters, with the training and test loss (negative log-likelihood) reported in Figure 2.9. As the regularization parameter increases for each method, the training loss increases. The test loss initially decreases and then increases as the model is over-fit. We observe that over the whole solution path, for the same training error NPMR consistently yields a lower test error than ridge regression.

Figure 2.10 reveals a possible explanation why NPMR outperforms ridge regression on the vowel data. For example the results show that when the vowel i is a likely label, the vowel I is also a likely label. The first two latent variables explain a significant portion of the variance in the regression coefficients for the vowels. The first latent variable distinguishes between two groups of vowels, with C:, U and u: having the most negative values and E, A, a: and Y having the most positive values. NPMR has beaten ridge regression in this example because there is a hidden structure between the response categories.

2.6 Discussion

The potential for reduced-rank multinomial regression to leverage the underlying structure among response categories has been recognized in the past. But the computational cost for the state-of-the-art algorithm for fitting such a model is so great as to make it infeasible to apply to a dataset as large as the baseball play-by-play data in the present work. The approach of using a convex relaxation of the problem, by penalizing the nuclear norm of the coefficient matrix instead of its

Figure 2.9: Results of fitting NPMR and ridge regression on vowel data. Test error is plotted against training error, using negative log-likelihood loss. Training error serves as a surrogate for degrees of freedom in the model fit. The null prediction assigns equal probability to all categories. Error bars represent one standard error in estimation of the test loss.



rank, leads to better results.

The interpretation of the results on the baseball data is promising in how it coalesces with modern baseball understanding. Specifically, the NPMR model has quantitative implications on leveraging the structure in PA outcomes to better jointly estimate outcome probabilities. Additional application to vowel recognition in speech shows improved out-of-sample predictive performance, relative to ridge regression. This matches the intuition that NPMR is well-suited to multinomial regression in the presence of a generic structure among the response categories. We recommend the use of NPMR for any multinomial regression problem for which there is some non-ordinal structure among the outcome categories.

Figure 2.10: Visualization of fitted regression coefficient matrices from NPMR applied to the vowel data. The matrix displayed is $\mathbf{V}$ in the $\mathbf{U\Sigma V}^T$ decomposition of the regression coefficient matrix $\mathbf{B}$, with columns corresponding to latent variables and rows corresponding to outcomes. The bottom row gives the entry in the diagonal matrix $\mathbf{\Sigma}$ corresponding to the latent variable.

Latent variable	1	2	3	4	5	6	7	8	9	10
i (heed)	-0.13	0.51	0.66	0.08	-0.41	-0.00	0.09	-0.05	-0.07	0.00
I (hid)	-0.03	0.44	-0.30	-0.44	0.11	0.33	-0.18	0.18	0.17	-0.46
E (head)	0.35	0.32	-0.43	0.18	-0.16	-0.01	0.02	0.20	0.06	0.63
A (had)	0.52	-0.08	-0.14	0.41	-0.08	-0.11	0.22	-0.19	-0.22	-0.55
a: (hard)	0.23	-0.35	0.35	-0.13	0.20	-0.00	0.34	0.51	0.41	0.01
Y (hud)	0.22	-0.14	0.25	0.04	0.37	0.51	-0.32	-0.47	-0.00	0.24
O (hod)	0.02	-0.34	0.06	-0.17	-0.22	-0.17	-0.57	0.36	-0.49	0.00
C: (hoard)	-0.30	-0.41	-0.23	-0.02	-0.58	0.14	0.03	-0.29	0.40	-0.02
U (hood)	-0.34	-0.09	-0.15	-0.21	0.17	0.18	0.58	-0.04	-0.55	0.14
u: (who'd)	-0.53	0.05	-0.07	0.62	0.37	-0.13	-0.18	0.18	0.13	-0.08
3: (heard)	0.01	0.08	-0.01	-0.36	0.24	-0.73	-0.03	-0.40	0.15	0.07
Corresponding diagonal	9.37	7.97	2.65	1.98	1.77	0.78	0.39	0.00	0.00	0.00

Chapter 3

Customized training

We introduce a simple, interpretable strategy for making predictions on test data when the features of the test data are available at the time of model fitting. Our proposal—*customized training*—clusters the data to find training points close to each test point and then fits an ℓ_1-regularized model (lasso) separately in each training cluster. This approach combines the local adaptivity of k-nearest neighbors with the interpretability of the lasso. Although we use the lasso for the model fitting, any supervised learning method can be applied to the customized training sets. We apply the method to a mass-spectrometric imaging dataset from an ongoing collaboration in gastric cancer detection which demonstrates the power and interpretability of the technique. Our idea is simple but potentially useful.

3.1 Introduction

Recent advances in the field of personalized medicine have demonstrated the potential for improved patient outcomes through tailoring medical treatment to the characteristics of the patient [Hamburg and Collins, 2010]. While these characteristics most often come from genetic data, there exist other molecular data on which to distinguish patients. In this paper we propose *customized training*, a very general, simple and interpretable technique for local regression and classification on large amounts of data in high dimension. The method can be applied to any supervised learning or transductive learning task, and it demonstrates value in applications to real-life datasets.

This paper is motivated by a newly proposed medical technique for inspecting the edge of surgically resected tissue for the presence of gastric cancer [Eberlin et al., 2014]. Gastric is the second-most lethal form of cancer, behind lung cancer [World Health Organization, 2013], and the state-of-the-art treatment for gastric cancer is surgery to remove the malignant tissue. With this surgical procedure, removal of the entirety of the diseased tissue is critical to the prognosis for the patient post-surgery. The new medical technique uses mass spectrometric imaging, rather than visual inspection by a

pathologist, to more quickly and more accurately evaluate the surgical margin of the tissue for the presence of cancerous cells. This replaces the procedure wherein the tissue samples are frozen until the pathologist is available to manually label the tissue as cancer or normal (see Figure 3.1).

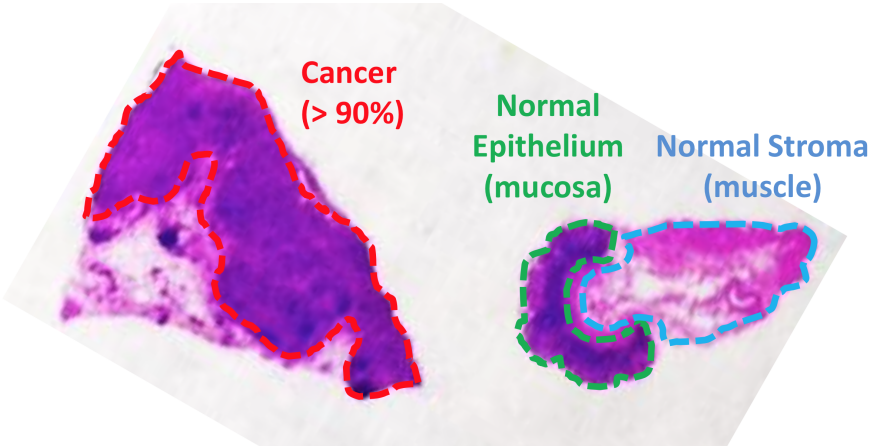


Figure 3.1: Histopathological assessment of a banked tissue example. This hematoxylin and eosin stain has been hand-labelled by a pathologist, marking three regions: gastric adenocarcinoma (cancer), epithelium (normal) and stroma (normal).

The data are images of surgical tissue from a DESI mass spectrometer, which records the intensity of ions at 13,320 mass-to-charge values at each of hundreds of pixels. This results in a mass spectrum for each pixel, illustrated in Figure 3.2.

The 13,320 ion intensities from the mass spectrum for each pixel were averaged across bins of six to yield 2220 features. Each pixel has been labelled by a pathologist (after 2 weeks of sample testing) as epithelial, stromal or cancer, the first two being normal tissue. Each of 20 patients contributed up to three samples, from some or all of the three classes. The training set comprises 28 images from 14 patients, yielding 12,480 pixels, and the test set has 12 images from 6 different patients, for a total of 5696 pixels.

In Eberlin et al. [2014] the authors use the lasso (ℓ_1-regularized multinomial regression) to model the probability that a pixel belongs to each of the three classes on the basis of the ion intensity in each bin of six mass-to-charge values. In that study, the lasso performed favorably in comparison with support vector machines and principal component regression. For a detailed description of the lasso, see Section 3.2.2. For the purposes of the present paper, we collapse epithelial and stromal into one class, "Normal", and we adopt a loss function that assigns twice the penalty to misclassifying a cancer cell as normal, relative to misclassifying a normal cell as cancer. This reflects that the question of interest is whether there are cancer cells in the margin of surgical resection and that a missing cancer cell is more harmul than making an error in the opposite direction.

The lasso classifier fit to the data from the 12,480 pixels in the training set (with the regularization

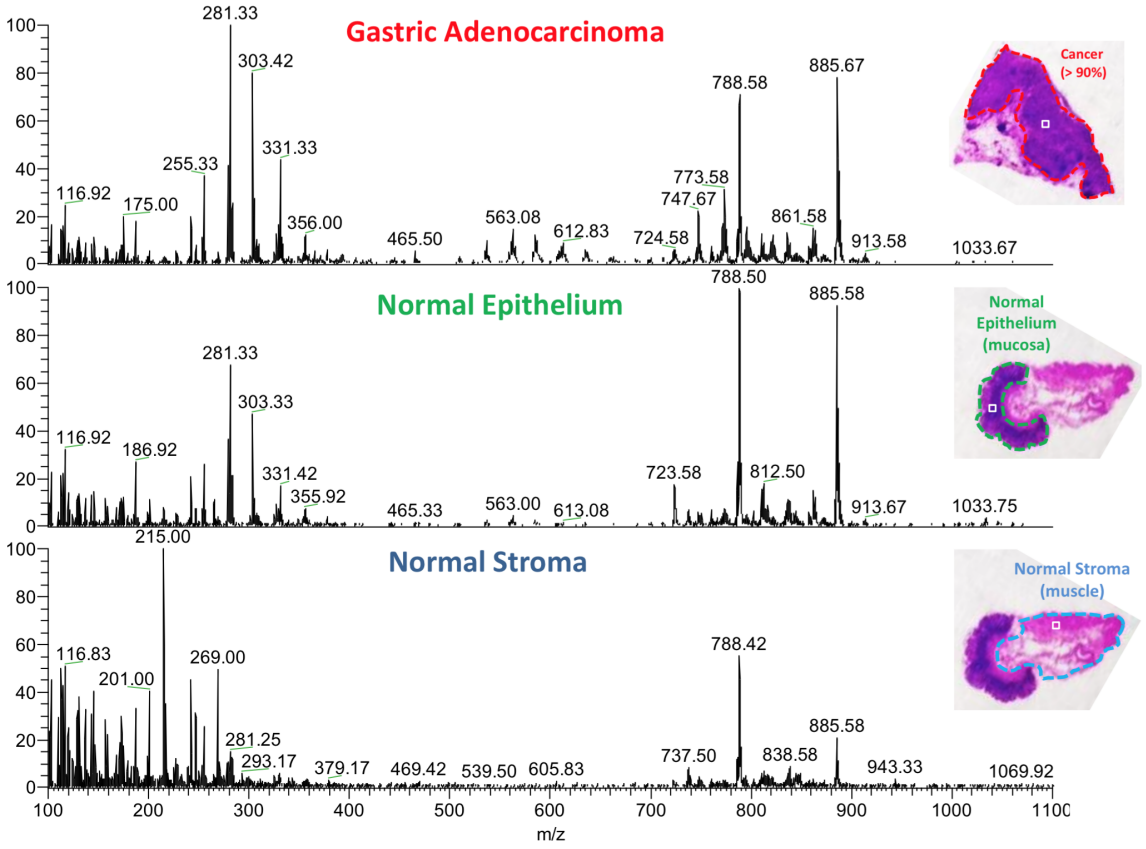


Figure 3.2: DESI mass spectra for one pixel taken from each region in the banked tissue example. The result of DESI mass spectrometric imaging is a 2D ion image with hundreds of pixels. Each pixel has an ion intensity measurement at each of thousands of mass-to-charge values, producing to a mass spectrum. The three mass spectra in the image correspond to one pixel each. The objective is to classify a pixel as cancer or normal on the basis of its mass spectrum.

parameter λ selected via cross validation, see Section 3.2.3) achieves a misclassification rate of 2.97% when used to predict the cancer/normal label of the 5696 pixels in the test set. Among cancer pixels the test error rate is 0.79%, and among normal pixels the test error rate is 4.16%. These results represent a significant improvement over the subjective classifications made by pathologists, which can be unreliable in up to 30% of patients [Eberlin et al., 2014], but the present paper seeks to improve these results further. By using customized training sets, our method fits a separate classifier for each patient, creating a locally linear but globally nonlinear decision boundary. This rich classifier leads to more accurate classifications by using training data most relevant to each patient when modeling his or her outcome probabilities.

3.1.1 Transductive learning

Customized training is best suited for the category of problems known in machine learning literature as transductive learning, in contrast with supervised learning or semi-supervised learning. In all of these problems, there is a training dataset for which both the dependent and the independent variables are observed (we say that the training set is "labelled") and a test dataset on which the objective is to predict the dependent variable. The distinction between the three types of problems is as follows: In supervised learning, the learner does not have access to the independent variables in the test set at the time of model fitting whereas in transductive learning the learner does have access to these data at model fitting. Semi-supervised learning is similar in that the learner has access to unlabelled data in addition to the training set, but these additional data do not belong to the test set on which the learner makes predictions. Customized training leverages information in the test data by choosing the most relevant training data on which to build a model to make better predictions. We have found no review of transductive learning techniques, but for a review of techniques for the related semi-supervised problem, see Zhu [2007].

The rest of the paper is organized as follows. In Section 3.2 we introduce customized training and discuss related methods. Section 3.3 investigates the performance of customized training and competing methods in a simulation study. Results on the motivating gastric cancer dataset are presented, with their interpretation, in Section 3.4. We apply our method and others to a battery of real datasets from the UCI Machine Learning Repository in Section 3.5. The manuscript concludes with a discussion in Section 4.9.

3.2 Customized training

First we introduce some notation. The data we are given are $\mathbf{X}_{\text{train}}$, $\mathbf{Y}_{\text{train}}$, and $\mathbf{X}_{\text{test}}$. $\mathbf{X}_{\text{train}}$ is an $n \times p$ matrix of predictor variables, and $\mathbf{Y}_{\text{train}}$ is an n-vector of response variables corresponding to the n observations represented by the rows of $\mathbf{X}_{\text{train}}$. These response variables may be qualitative or quantitative. $\mathbf{X}_{\text{test}}$ is an $m \times p$ matrix of the same p predictor variables measured on m test observations. The goal is to predict the unobserved random m-vector $\mathbf{Y}_{\text{test}}$ of responses corresponding to the observations in $\mathbf{X}_{\text{test}}$.

Let $\hat{f}_\Lambda(\cdot)$ denote the prediction made by some learning algorithm, as a function of $\mathbf{X}_{\text{train}}$, $\mathbf{Y}_{\text{train}}$, $\mathbf{X}_{\text{test}}$ and an ordered set Λ of tuning parameters. So $\hat{f}_\Lambda(\mathbf{X}_{\text{train}}, \mathbf{Y}_{\text{train}}, \mathbf{X}_{\text{test}})$ is an m-vector. For qualitative $\mathbf{Y}_{\text{train}}$, $\hat{f}_\Lambda$ is a classifier while for quantitative $\mathbf{Y}_{\text{train}}$, $\hat{f}_\Lambda$ fits a regression. We evaluate the performance of $\hat{f}_\Lambda$ with $L(\hat{f}_\Lambda(\mathbf{X}_{\text{train}}, \mathbf{Y}_{\text{train}}, \mathbf{X}_{\text{test}}), \mathbf{Y}_{\text{test}})$, where the loss function L is often taken to be, for example, the number of misclassifications for a qualitative response, or squared error for a quantitative response.

The customized training method partitions the test set into G subsets and fits a separate model $\hat{f}_\Lambda$ to make predictions for each subset. In particular, each subset of the test set uses only its own,

"customized" subset of the training set to fit $\hat{f}_\Lambda$. This leads to a model that is locally linear but rich globally. Next, we propose two methods for partitioning the test set and specifying the customized training subsets.

3.2.1 Clustering

Often there is a grouping inherent to the test data, obviating the need to identify clusters in the data using unsupervised learning techniques. This is especially advantageous on large datasets for which it would be very expensive computationally to cluster the data. For example, in the motivating application for the present manuscript, test data are grouped by patient, so we avoid clustering the 5696 test observations in 2220 dimensions by using patient identity as the cluster membership for each test point.

Given the G "clusters" identified by the grouping inherent to the test data, we identify the customized training set for each test cluster as follows: First, for each observation in the cluster, find the R nearest neighbors in the training set to that observation. This defines many cardinality-R sets of training observations, one for each test point in the cluster. Second, take the union of these sets as the customized training set for the cluster. So the customized training set is the set of all training points that are one of the R nearest neighbors of any test point in the cluster. R is a tuning parameter that could in principal be chosen by cross validation, but we have found that $R = 10$ works well in practice and that results are not particularly sensitive to this choice.

When there is no inherent grouping available for the test data, customized training works by jointly clustering the training and test observations, according to their predictor variables. Any clustering method can be used; here we apply hierarchical clustering with complete linkage to the data $(\mathbf{X}_{\text{train}}^T, \mathbf{X}_{\text{test}}^T)^T$. Then we cut the dendrogram at some height d_G, producing G clusters, and in each cluster we train a classifier on the training observations within that cluster. This model is then used to make predictions for the test observations within the cluster. In this case, G is a tuning parameter to be chosen by cross validation (see Section 3.2.3).

3.2.2 Classification and regression

The key idea behind our method is the selection of a customized training set for each group in the test set. Once these individualized training sets are identified, any supervised classification (or regression, in the case of quantitative outcomes) technique can be used to fit $\hat{f}_\Lambda$ and make predictions for the test set. We suggest using ℓ_1-regularized generalized linear models because of their interpretability. Customized training complicates the model by expanding it into a compilation of G linear models instead of just one. But using ℓ_1 regularization to produce sparse linear models conserves interpretability. For an $n \times p$ predictor matrix $\mathbf{X}$ and corresponding response vector y, an

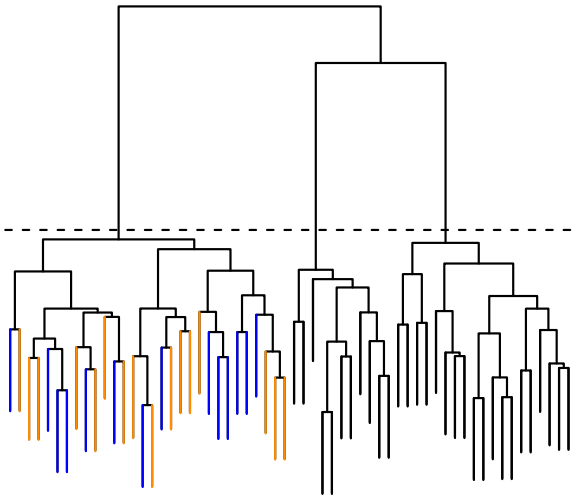


Figure 3.3: A dendrogram depicting joint clustering of training and test data. This is the method proposed for partitioning the test data and identifying customized training sets when there is no grouping inherent to the test data. Here the dendrogram is cut at a height to yield $G = 3$ clusters. Within the left cluster, the training data (blue leaves) are used to fit the model and make predictions for the test data (orange leaves).

ℓ_1-regularized GLM solves the optimization problem

$$\min_{\beta_0,\beta\in\mathbb{R}^p} -\frac{1}{n}\sum \ell(\beta_0,\beta|\mathbf{x}_i,Y_i) + \lambda||\beta||_1 \tag{3.1}$$

where $\ell(\cdot)$ here is the log-likelihood function and depends on the assumed distribution of the response. For example, for linear regression (which we use for quantitative response variables),

$$Y_i|\mathbf{x}_i,\beta_0,\beta \sim \text{Normal}(\beta_0+\beta^T\mathbf{x}_i,\sigma^2)$$

while for logistic regression (which we use for binary response variabes),

$$Y_i|\mathbf{x}_i,\beta_0,\beta \sim \text{Binomial}\left(1,\frac{e^{\beta_0+\beta^T\mathbf{x}_i}}{1+e^{\beta_0+\beta^T\mathbf{x}_i}}\right).$$

For multiclass qualitative reponse variables we use the multinomial distribution in the same framework. The estimated regression coefficient vector $\hat{\beta}$ that solves the optimization problem (3.1) can be interpreted as the contribution of each predictor to the distribution of the response, so by penalizing $||\beta||_1$ in (3.1), we encourage solutions for which many entries in $\hat{\beta}$ are zero, thus simplifying interpretation [Tibshirani, 1996].

Regardless of the $\hat{f}_\Lambda$ chosen, for $g = 1,...,G$, let n_k denote the number of observations in the customized training set for the k^{th} test cluster, and let $\mathbf{X}^k_{\text{train}}$ denote the $n_k \times p$ sub-matrix of $\mathbf{X}_{\text{train}}$ corresponding to these observations, with $\mathbf{Y}^k_{\text{train}}$ denoting the corresponding responses. Similarly,

let m_k denote the number of test observations in the k^{th} cluster, and let $\mathbf{X}^k_{\text{test}}$ denote the $m_k \times p$ sub-matrix of $\mathbf{X}_{\text{test}}$ corresponding to these training observations, with $\mathbf{Y}^k_{\text{test}}$ denoting the corresponding responses. Once we have a partition of the test set into G subsets (some of which may contain no test observations), with tuning parameter Λ we take our prediction for $\mathbf{Y}^k_{\text{test}}$ to be

$$\hat{\mathbf{Y}}^k_{\text{test}} = \hat{f}_\Lambda(\mathbf{X}^k_{\text{train}}, \mathbf{Y}^k_{\text{train}}, \mathbf{X}^k_{\text{test}}) \tag{3.2}$$

Note that if joint clustering is used to partition the test data, the customized training set for the k^{th} test cluster may be empty, in which case $\hat{f}_\Lambda(\mathbf{X}^k_{\text{train}}, \mathbf{Y}^k_{\text{train}}, \mathbf{X}^k_{\text{test}})$ is undefined. There are several ways to get around this issue. We discuss the low prevalence of this problem and offer one solution in Section 3.2.5. Once we have predictions for each subset, they are combined into the m-vector $CT_{G,\Lambda}\left(\mathbf{X}_{\text{train}}, \mathbf{Y}_{\text{train}}, \mathbf{X}_{\text{test}}\right)$, which we take as our prediction for $\mathbf{Y}_{\text{test}}$.

3.2.3 Cross validation

To determine G and Λ, we use standard cross-validation. Because transductive methods have access to test features at training time, we explain carefully in this section what we mean by standard cross-validation.

The training data are randomly partitioned into J approximately equal-sized folds (typically $J = 10$). For $j = 1, ..., J$, $\mathbf{X}^{(j)}_{\text{train}}$ denotes the sub-matrix of $\mathbf{X}_{\text{train}}$ corresponding to the data in the j^{th} fold, and $\mathbf{X}^{(-j)}_{\text{train}}$ denotes the sub-matrix of data *not* in the j^{th} fold. Similarly, $\mathbf{Y}^{(j)}_{\text{train}}$ denotes the responses corresponding to the data in the j^{th} fold, and $\mathbf{Y}^{(-j)}_{\text{train}}$ denotes responses *not* in the j^{th} fold.

We consider $\mathcal{G}$ and A as the sets of possible values for G and Λ, respectively. In practice, we use $\mathcal{G} = \{1, 2, 3, 5, 10\}$. We search over the grid $\mathcal{G} \times A$, and the CV-selected parameters G and Λ are

$$(G^*, \Lambda^*) = \underset{G \in \mathcal{G}, \Lambda \in A}{\arg\min} \sum_{j=1}^{J} L\left(CT_{G,\Lambda}\left(\mathbf{X}^{(-j)}_{\text{train}}, \mathbf{Y}^{(-j)}_{\text{train}}, \mathbf{X}^{(j)}_{\text{train}}\right), \mathbf{Y}^{(j)}_{\text{train}}\right).$$

In more detail, the G clusters for $CT_{G,\Lambda}(\mathbf{X}^{(-j)}_{\text{train}}, \mathbf{Y}^{(-j)}_{\text{train}}, \mathbf{X}^{(j)}_{\text{train}})$ are obtained as described in Section 3.2.1, and the loss for the j^{th} fold is given by

$$L\left(CT_{G,\Lambda}\left(\mathbf{X}^{(-j)}_{\text{train}}, \mathbf{Y}^{(-j)}_{\text{train}}, \mathbf{X}^{(j)}_{\text{train}}\right), \mathbf{Y}^{(j)}_{\text{train}}\right) = \sum_{k=1}^{G} L\left(\hat{f}_\Lambda\left(\mathbf{X}^{(-j)^k}_{\text{train}}, \mathbf{Y}^{(-j)^k}_{\text{train}}, \mathbf{X}^{(j)^k}_{\text{train}}\right), \mathbf{Y}^{(j)^k}_{\text{train}}\right).$$

3.2.4 Software

Customized training for lasso and elastic-net regularized generalized linear models is implemented in the R package `customizedTraining`, available on the Comprehensive R Archive Network. The

primary function is `cv.customizedGlmnet`, which takes as arguments a training covariate matrix $\mathbf{X}$ and response $\mathbf{Y}$, and optionally a test covariate matrix or vector of group memberships for the test set (as in Section 3.4). The function, whose call is given below, uses cross validation to choose the number G of clusters and the regularization parameter λ, and returns a fitted `cv.customizedGlmnet` object which has its own `plot` and `predict` methods.

```
cv.customizedGlmnet(xTrain, yTrain, xTest = NULL, groupid = NULL, Gs = NULL,
  dendrogram = NULL, dendrogramCV = NULL, lambda = NULL,
  nfolds = 10, foldid = NULL, keep = FALSE,
  family = c("gaussian", "binomial", "multinomial"), verbose = FALSE)
```

3.2.5 Out-of-sample rejections

As noted in Section 3.2, when joint clustering is used to partition the test data and identify customized training sets, predictions for a particular test subset may be undefined because the corresponding customized training subsets does not contain any observations. Using the convention of [Bottou and Vapnik, 1992], we refer to this event as a *rejection* (although it might more naturally deemed an *abstention*). The number of rejections, then, is the number of test observations for which our procedure fails to make a prediction due to an insufficient number of observations in the customized training set.

Typically, in the machine learning literature, a rejection occurs when a classifier is not confident in a prediction, but that is not the case here. For customized training, a rejection occurs when there are no training observations close to the observations in the test set. This latter problem has not often been addressed in the literature [Bottou and Vapnik, 1992]. Because the test data lie in a region of the feature space that is poorly represented in the training data, a classifier might make a very confident, incorrect prediction.

We view the potential for rejections as a virtue of the method, identifying situations in which it is best to make no prediction at all because the test data are out-of-sample, a rare feature for machine learning algorithms. In practice, we observe that rejections are rare; Table 3.4 gives a summary of all rejections in the battery of machine learning datasets from Section 3.5.

If a prediction must be made, there are many ways to get around rejections. We propose simply cutting the dendrogram at a greater height $d' > d_G$ so that the test cluster on which the rejections occurred is combined with another test cluster until the joint customized training set is large enough to make predictions. Specifically, we consider the smallest d' for which the predictions are defined. Note that this updates the prediction only for the test observations on which the method previously abstained.

3.2.6 Related work

Local learning in the transductive setting has been proposed before [Zhou et al., 2004, Wu and Schölkopf, 2007]. There are other related methods as well, e.g. transductive regression with elements of local learning [Cortes and Mohri, 2007] or local learning that could be adapted to the transductive setting [Yu et al., 2009]. The main contribution of this paper relative to previous work is the simplicity and interpretability of customized training. By combining only a few sparse models, customized training leads to a much more parsimonious model than other local learning algorithms, easily explained and interpreted by subject-area scientists.

More recently, local learning has come into use in the transductive setting in applications related to personalized medicine. The most relevant example to this paper is evolving connectionist systems [Ma, 2012], but again our proposal for customized training leads to a more parsimonious and interpretable model. Personalized medicine is an exciting area of potential application for customized training.

A host of other methods [Gu and Han, 2013, Ladicky and Torr, 2011, Torgo and DaCosta, 2003] similarly partition the feature space and fit separate classification or regression models in each region. However, in addition to lacking the interpretability of our method, these techniques apply only to the supervised setting and do not leverage the additional information in the transductive setting. Others have approached a similar problem using mixture models [Fu et al., 2010, Shahbaba and Neal, 2009, Zhu et al., 2011], but these methods also come with a great computational burden, especially those which use Gibbs sampling to fit the model instead of an EM algorithm or variational methods.

Variants of customized training could also be applied in the supervised and semi-supervised setting. The method would be semi-supervised if instead of test data, other unlabeled data were used for clustering and determining the customized training set for each cluster. The classifier or regression obtained could be used to make predictions for unseen test data by assigning each test point to a cluster and using the corresponding model. A supervised version of customized training would cluster only the training data and fit a model for each cluster using the training data in that cluster. Again, predictions for unseen test data could be made after assigning each test point to one of these clusters. This approach would be similar to [Jordan and Jacobs, 1994].

Alternative methods

To compare customized training against the state of the art, we try a host of machine learning methods to the datasets in Sections 3.3 and 3.5. We do not apply these methods to the dataset in Section 3.4 because the dataset is too large to get results in a reasonable amount of time.

ST Standard training. This uses the ℓ_1-penalized regression techniques outline in Section 3.2.2, training one model on all of the data in the training set. The regularization parameter λ is chosen through cross validation.

SVM Support vector machine. The cost tuning parameter is chosen through cross-validation.

KSVM K-means + SVM. We cluster the training data into K clusters via the K-means algorithm and fit an SVM to each training cluster. Test data are assigned to the nearest cluster centroid. This is a simpler, special case of the clustered SVMs proposed by Gu and Han [2013].

RF Random forests. At each split we consider $\sqrt{p}$ of the p predictor variables (classification) or $p/3$ of the p predictor variables (regression).

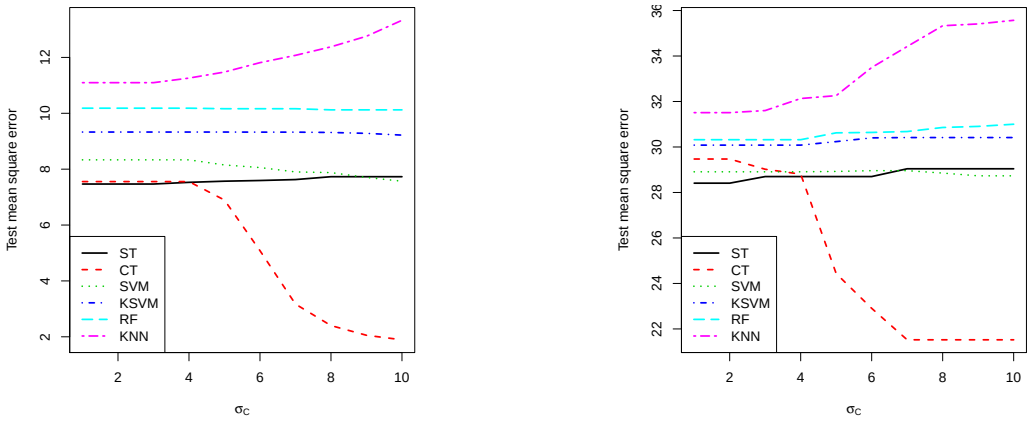
KNN k-nearest neighbors. This simple technique for classification and regression puts the performance of customized training in context with another "local" method. The parameter k is chosen via cross validation.

3.3 Simulation study

We designed a simulation to demonstrate that customized training improves substantially on standard training in situations where one would expect this to occur: when the data belong to several clusters, each with a different relationship between features and responses. We consider real-valued responses (a regression problem) for the sake of variety. We simulated n training observations and m test observations in p dimensions, each observation belonging to one of 3 classes. The frequency of the 3 classes was determined by a Dirichlet$(2, 2, 2)$ random variable. The centers c_1, c_2, c_3 of the 3 classes were generated as i.i.d. p-dimensional normal random variables with covariance $\sigma_c^2 \mathbf{I}_p$.

Given the class membership $z_i \in \{1, 2, 3\}$ of the i^{th} observation, $\mathbf{x}_i$ was generated from a normal distribution with mean $\mathbf{c}_{z_i}$ and covariance matrix $\mathbf{I}_p$. The coefficient vector β^k corresponding to the k^{th} class had $p/10$ entries equal to one, with the rest being zero, reflecting a sparse coefficient vector. The nonzero entries of β^k were sampled uniformly at random, independently for each class k. Given the class membership z_i and coefficient vector $\mathbf{x}_i$ of the i^{th} observation, the response Y_i had a normal distribution with mean $(\beta^{z_i})^T \mathbf{x}_i$ and standard deviation one.

We conduct two simulations, the first with $n = m = 300$, $p = 100$ (the low-dimensional setting), and the second with $n = m = 200$, $p = 300$ (the high-dimensional setting). In each case, we vary σ_C from 0 to 10. Figure 3.4 shows the results. We observe that in both settings, customized training leads to significant improvement in test mean square error as the clusters separate (increasing σ_C). In the high-dimensional setting, the errors are expectedly much larger, but the same pattern is evident. For KSVM in this simulation we fix $K = 3$, thus cheating and giving the algorithm the number of clusters whereas customized training learns the number of clusters from the data. This is why the performance of KSVM does not improve as the clusters separate. In fact, it is because none of the other methods make an attempt to identify the number of clusters that they do not improve as the clusters separate.



(a) $n = m = 300$, $p = 100$ (b) $n = m = 200$, $p = 300$

Figure 3.4: Simulation results. In (a), the low-dimensional setting, as σ_C increases and the clusters separate, the test error for customized training drops while the test error for other methods remains high. In (b), the test errors are much larger overall, but the same pattern persists: Customized training leads to improved results as the clusters separate.

3.4 Results

We applied customized training to the mass-spectrometric imaging dataset of gastric cancer surgical resection margins with the goal of improving of the results obtained by standard training. Because the 5696 pixels in the test set belong to six different patients, the group labels for the test set are given. As described in Section 3.2.1, we obtained a customized training set for each test patient by finding the 10 nearest neighbors of each pixel in that patient's images and using the union of these nearest-neighbor sets. Table 3.1 shows from which training patients the customized training set came, for each test patient. The patient labels have been ordered to make the structure in these data apparent: Test patients 1–3 rely heavily on training patients 1–7 for their customized training sets while test patients 4–6 rely heavily on training patients 9–14 for their customized training sets.

In this setting it is more harmful misclassify cancer tissue as normal than it is to make the opposite error, so we chose to use a loss function that penalizes a false negative (labelling a cancer pixel as normal) twice as much as it does a false positive (labelling a normal pixel as cancer). We compare the results of customized training against the results of standard training for ℓ_1-regularized binomial regression—the method used by Eberlin et al. [2014]. Table 3.2 gives the results.

We observe that customized training leads to a considerable improvement in results. For test patients 3 and 4, the test error is slightly higher for customized training than for standard training, but for all other patients, the test error for customized training is much lower. Overall, customized training cuts the number of misclassifications in half from the results of standard training.

Table 3.1: Source patients for customized training sets. Each column shows, for the corresponding test patient, what percentage of observations in that patient's customized training set came from each of the training patients. Patient labels have been permuted to show the structure in the data: Test patients 1–3 get most of their training sets from patients 1–7 while test patients 4–6 get most of their training sets from patients 9–14.

		Test Patient					
		1	2	3	4	5	6
Training Patient	1	45.5	26.7	9.2	—	—	—
	2	28.9	19.4	4.9	0.2	—	0.6
	3	11.9	11.1	13.5	—	—	0.2
	4	2.6	19.0	14.6	1.1	0.2	0.2
	5	8.2	8.2	7.8	—	—	5.8
	6	1.6	8.7	13.2	—	0.3	—
	7	1.2	6.2	14.8	0.4	0.2	0.3
	8	—	—	—	1.4	1.9	0.3
	9	—	—	5.8	18.1	0.7	1.1
	10	—	0.1	—	20.6	2.1	4.1
	11	—	—	0.9	6.1	14.6	5.6
	12	0.1	0.1	8.8	12.2	10.8	12.3
	13	—	—	1.2	4.7	35.6	22.1
	14	—	0.4	5.4	35.4	33.7	47.5

3.4.1 Interpretation

A key draw for customized training is that, although the decision boundary is more flexible than a linear one, interpretability of the fit is preserved because of the sparsity of the model. In this example, there are 2220 features in the dataset, but the numbers of features selected for test patients 1 through 6 are, respectively, 42, 71, 62, 15, 21 and 54. Figure 3.5 which features are used in each patient's model, along with the features used in the overall model with standard training.

We observe that some pairs of patients have more similar profiles of selected features than other pairs of patients. For example, about 36% of the features selection for test patient 1 are also selected for test patient 2. And about 39% of the features selected for test patient 3 are also selected for test patient 2. This is not surprising because test patients 1 through 3 take much of their customized training sets from the same training patients, as observed above. Similarly, about 40% of the features selected for patient 4 are also selected for patient 6, and about 38% of the features selected for patient 5 are also selected for patient 6.

The third author's subject-area collaborators have suggested that there may actually be two subclasses of cancer present in these data, and given that customized training identifies two different groups of models for predicting cancer presence, this leads to a potentially interesting interpretation of the results.

Table 3.2: Error rates for customized training and standard training on the gastric cancer test data, split by patient and true label of the pixel (cancer or normal). Each error rate is expressed by the percentage of pixels misclassified. Standard training leads to slightly lower errors for patients 3 and 4, but customized training leads to much lower errors for all other patients and roughly half the error rate overall.

Test Patient		1	2	3	4	5	6	All
Standard	Cancer	—	2.67	0.21	—	—	2.70	1.54
training	Normal	13.60	0.81	0.13	0.00	6.37	3.63	3.78
(lasso)	Overall	13.60	1.61	**0.18**	**0.00**	6.37	3.14	2.98
Customized	Cancer	—	1.07	0.11	—	—	1.80	0.74
training	Normal	8.66	0.00	1.44	0.40	0.82	0.66	2.04
(6-CT)	Overall	**8.66**	**0.46**	0.71	0.40	**0.82**	**1.26**	**1.58**

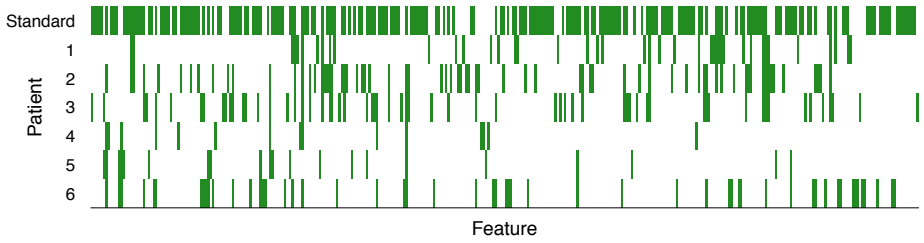


Figure 3.5: Features selected by customized training for each patient (variables not selected by any model are omitted from the x-axis). The first row shows features selected via standard training. Visual inspection suggests that patients 1, 2 and 3 have similar profiles of selected variables, whereas patients 4 and 5 have selected-feature profiles that are more similar to each other than to other patients'. Using hierarchical clustering with Jaccard distance between the sets of selected features to split the patients into two clusters, patients 1, 2 and 3 were in one cluster, with patients 4, 5 and 6 in the other.

3.5 Additional applications

To learn about the value of customized training in practice, we applied customized training (and the alternative methods from Section 3.2.6) to a battery of classification data sets from the UC Irvine Machine Learning Repository [Bache and Lichman, 2013, Gil et al., 2012, Tsanas et al., 2014, Little et al., 2007, Mansouri et al., 2013, Kahraman et al., 2013]. The data sets were selected not randomly but somewhat arbitrarily, covering a wide array of applications and values of n and p, with a bias toward recent data sets. Here we present results on all 16 data sets to which the methods were applied, not just those on which customized training performed well.

Random forests achieve the lowest error on 8 of the 16 datasets, the most of any method. But the method that achieves the lowest error second-most often is customized training, on 7 of the 16 datasets, and customized training beats standard training on 11 datasets, with standard training

Table 3.3: Test error of customized training on 16 benchmark data sets

			ST	CT		SVM	KSVM	RF	KNN		
Data	n	p	Error	Error	G	Error	Error	Error	Error	k	%Imp*
BS	313	4	.112	.099	3	**.086**	.131	.131	.105	20	11.4%
BCW	285	30	**.028**	.035	2	.035	.038	**.028**	.056	63	-25%
C	1598	38	.026	.021	10	.029	.046	**.006**	.085	36	18.5%
CMC	737	18	.485	**.440**	5	.479	.523	.472	.523	32	9.2%
F	50	9	**.160**	**.160**	1	**.160**	**.160**	.180	.180	2	—
FTP	3059	51	.557	.530	5	.489	.444	**.427**	.508	47	4.7%
LSVT	63	310	.126	.142	1	.111	.365	**.095**	.222	15	-12.5%
M	4062	96	**.000**	**.000**	1	.001	.001	**.000**	.001	15	—
ORHD	3823	62	.046	.043	2	.032	.049	**.027**	.055	38	6.0%
P	98	22	.268	.144	3	.154	.144	**.082**	.123	5	46.1%
Q	528	41	.176	**.134**	5	.146	.148	.140	.144	19	23.6%
S	105	7	**.047**	**.047**	2	.066	.114	.104	.066	9	—
SPF	971	27	.321	.278	5	.273	.281	**.246**	.357	57	13.3%
TAE	76	53	.720	**.470**	10	.653	.613	.506	.493	1	34.6%
UKM	258	5	.041	**.013**	1	.103	.213	.068	.565	79	66.6%
V	528	10	.610	.491	2	**.387**	.480	.409	.508	1	19.5%

*%Imp: Percent relative improvement of customized training to standard training

coming out on top for only 2 datasets. We do not expect customized training to provide value on all datasets, but through cross-validation, we can often identify datasets for which standard training is better, mean that $G = 1$ is chosen through cross-validation. The point of this exercise is not to show that customized training is superior to the other methods but rather to show that, despite its simplicity, it is at least competitive with the other methods.

Table 3.4: A listing of all data sets from Section 3.5 for which K-CT_J makes a rejection for some K. The error rates in the last two columns refer to the error rate of standard training.

Data set	Method	Rejections	Error rate on rejections	Error rate overall
First-order theorem proving	3-CT_J	3	1	.518
	5-CT_J	3	1	
	10-CT_J	3	1	
LSVT Voice Rehabilitation	10-CT_J	2	0.5	.142
Parkinsons	10-CT_J	4	0.25	.154
Steel Plates Faults	3-CT_J	1	1	.294
	5-CT_J	1	1	
	10-CT_J	1	1	

Table 3.4 shows all of the rejections that customized training makes on the 16 data sets, for any value of G (not just the values of G chosen by cross-validation). For two of the data sets (`LSVT Voice Rehabilitation` and `Parkinsons`), it is clear that the rejections are just artifacts of using a G that is too large relative to the training sample size n. Such a G is not chosen by cross-validation. However, in the other datasets, `Steel Plates Faults` and `First-order theorem proving`, rejections occur

for moderate values of G. It seems that this rejection is appropriate because the standard training method leads to an error for each test point which is rejected. Overall, we observe that rejections are rare.

Table 3.5: Datasets from UCI Machine Learning Repository [Bache and Lichman, 2013] used in Section 3.5.

Abbrv.	Data set name
BS	Balance scale
BCW	Breast cancer Wisconsin (diagnostic)
C	Chess (king-rook vs king-pawn)
CMC	Contraceptive method choice
F	Fertility
FOTP	First-order theorem proving
LSVT	LSVT voice rehabilitation
M	Mushroom
ORHD	Optical recognition of handwritten digits
P	Parkinsons
QSAR	QSAR biodegration
S	Seeds
SPF	Steel plates faults
TAE	Teaching assistant evaluation
UKM	User knowledge modeling
V	Vowel

3.6 Discussion

The idea behind customized training is simple: for each subset of the test data, identify a customized subset of the training data that is close to this subset, and use this data to train a customized model. We proposed two different clustering methods for finding the customized training sets and use ℓ_1-regularized methods for training the models. Local learning has been used in the transductive setting but not in such a parsimonious, interpretable way. Customized training has the potential to uncover hidden regimes in the data and leverage this discovery to make better predictions. It may be that some classes are over-represented in a cluster, and fitting a model in this cluster effectively customizes the prior to reflect this over-representation.

In this paper we focused on customized training with ℓ_1-regularized methods for the sake of interpretability, but in principle any supervised learning method may be used and this is an area for future work. Another area of future work is the use of different clustering techniques. We use hierarchical clustering, but there may be value in other methods, such as prototype clustering [Bien

and Tibshirani, 2011]. Simulations in Section 3.3 show that the method can struggle in the high-dimensional setting, so it may be worthwhile to consider sparse clustering [Witten and Tibshirani, 2010].

Chapter 4

Patients like me

When devising a course of treatment for a patient, doctors often have little quantitative evidence on which to base their decisions, beyond their medical education and published clinical trials. Stanford Health Care alone has millions of electronic medical records (EMRs) that are only just recently being leveraged to inform better treatment recommendations. These data present a unique challenge because they are high-dimensional and observational. Our goal is to make personalized treatment recommendations based on the outcomes for past patients similar to a new patient. We propose and analyze three methods for estimating heterogeneous treatment effects using observational data. We compare the performance of these methods in simulation with the gradient forest of Athey et al. [2017].

4.1 Introduction

In February 2017, at the Grand Rounds of Stanford Medicine, one of us (NS) unveiled a new initiative — the Informatics Consult. Through this service, clinicians can submit a consultation request online and receive a report based on insights drawn from hundreds of millions of electronic medical records (EMRs) from Stanford Health Care. While the system is in its early stages, a future version will include treatment recommendations: helping a doctor to choose between treatment options for a patient, in cases where there is no randomized controlled trial (RCT) which compares the options. This announcement was met with excitement from the doctors in attendance, considering that they generally need to make decisions without any support from quantitative evidence (about 95% of the time) [Shah, 2016]. Building such a system is a priority in many medical centers in the U.S. and around the world.

The problem setting on which this paper focuses is when a doctor is presented with a patient who has some medical ailment, and the doctor is considering one or more treatment options. A relevant question from the patient's perspective is, *what is the effect of these treatments on patients*

like me? Devising a meaningful definition for "patients like me" is especially difficult given the high-dimensional nature of the problem: We observe thousands of features describing each patients, any of which could be used to describe patient similarity. The other significant complication is that our goal is to infer causal effects from observational data. The task of mining EMRs to support physician decision-making is what motivates this paper. We propose and study methods for estimation and inference of heterogeneous treatment effects, for both randomized experiments and observational studies. We focus on the case of a choice between two treatments, which for the purposes of this manuscript we label as "treatment" and "control".

In detail, we have an $n \times p$ matrix of features $\mathbf{X}$, a treatment indicator vector $\mathbf{T} \in \{0,1\}^n$, and a vector of quantitative responses $\mathbf{Y} \in \mathbb{R}^n$. Let X_i denote the ith row of $\mathbf{X}$, likewise T_i and Y_i. We assume the n observations (X_i, T_i, Y_i) are sampled i.i.d. from some unknown distribution. The number of treated patients is $N_1 = |\{i : T_i = 1\}|$, and the number of control patients is $N_0 = |\{i : T_i = 0\}|$. We adopt the Neyman–Rubin potential outcomes model [Splawa-Neyman et al., 1990, Rubin, 1974]: each patient i has potential outcomes $Y_i^{(1)}$ and $Y_i^{(0)}$, only one of which is observed. $Y_i^{(1)}$ is the response that the patient would have under treatment, and $Y_i^{(0)}$ is the response the patient would have under control. Hence the outcome that we actually observe is $Y_i = Y_i^{(T_i)}$. We consider both randomized controlled trials, where T_i is independent of all pre-treatment characteristics,

$$\left(X_i,\, Y_i^{(0)},\, Y_i^{(1)}\right) \perp\!\!\!\perp T_i, \tag{4.1}$$

and observational studies, where the distribution of T_i is dependent on the covariates. This scenario is discussed in further detail in Section 4.2.1.

We describe four important functions for modelling data of this type. The first is the propensity function, which gives the probability of treatment assignment, conditional on covariates:

$$\pi(x) \equiv \mathbb{P}(T = 1 | X = x). \tag{4.2}$$

The next two functions are the conditional mean functions: the expected response given treatment and the expected response given control.

$$\mu_1(x) \equiv \mathbb{E}[Y | X = x, T = 1] \quad \text{and} \quad \mu_2(x) \equiv \mathbb{E}[Y | X = x, T = 0].$$

The fourth function, and the one of greatest interest, is the treatment effect function, which is the difference between the two conditional means:

$$\tau(x) \equiv \mu_1(x) - \mu_0(x).$$

We seek regions in predictor space where the treatment effect is relatively large or relatively small. This is particularly important for the area of personalized medicine, where a treatment might

have a negligible effect when averaged over all patients but could be beneficial for certain patient subgroups.

An outline of this paper is as follows. Section 4.2 reviews related work. In Section 4.3 we describe the two main high-level approaches to the estimation of heterogeneous treatment effects: transformed outcome regression and conditional mean regression. In Section 4.4 we introduce *pollinated transformed outcome* (PTO) forests, while *causal boosting* is proposed in Section 4.5. *Causal MARS* is the focus of Section 4.6. In Section 4.7 we report the results of a simulation study comparing all of these methods, and a real data application is illustrated in Section 4.8. We end with a discussion.

4.2 Related work

Early work on heterogeneous treatment effect estimation [Gail and Simon, 1985] was based on comparing pre-defined subpopulations of patients in randomized experiments. To characterize interactions between a treatment and continuous covariates, Bonetti and Gelber [2004] formalized the subpopulation treatment effect patter plot (STEPP). Sauerbrei et al. [2007] proposed an efficient algorithm for flexible model-building with multivariable fractional polynomial interaction (MFPI) and compared the empirical performance of MFPI with STEPP.

Identifying subgroups within the patient population is becoming especially problematic in high-dimensional data, as in EMRs. In recent years, a great amount of work has been done to apply methods from machine learning to let the data inform what are the important subgroups in terms of treatment effect. Su et al. [2009] proposed interaction trees for adaptively defining subgroups based on treatment effect. Athey and Imbens [2016] proposed causal trees, which are similar, and constructed valid confidence intervals. Wager and Athey [2015] improved on this line of work by growing random forests [Breiman, 2001] from causal trees. These tree-based methods all use shared-basis conditional mean regression in the framework of Section 4.3. An example of a transformed-outcome estimator is the FindIt method of Imai and Ratkovic [2013] which trains an adapted support vector machine on a transformed binary outcome. Tian et al. [2014] introduced a simple linear model based on transformed covariates and show that it is equivalent to transformed outcome regression in the Gaussian case. In a novel approach, Zhao et al. [2012] used outcome weighted learning to directly determine individualized treatment rules, skipping the step of estimating individualized treatment effects. The problem of estimating heterogeneous treatment effects has also received significant attention in Bayesian literature. Hill [2011] and Green and Kern [2012] approached the problem using Bayesian additive regression trees [Chipman et al., 1998], and Taddy et al. [2016] proposed a method based on Bayesian forests. Chen et al. [2012] developed a Bayesian method for finding qualitative interactions between treatment and covariates, and there are other Bayesian methods for flexible nonlinear modelling of interactive/non-additive relationships between covariates

and response [LeBlanc, 1995, Gustafson, 2000].

What all of the above work (except Hill [2011]) have in common is that they assume randomized treatment assignment. Athey and Imbens [2016] discussed the possibility of adapting their method to observational data but go no further. Wager and Athey [2015] proposed the propensity forest when treatment is not randomized, but this method does not target heterogeneity in the treatment effect. Similarly, Xie et al. [2012] model treatment effect as a function of propensity score, missing out on how it depends on the covariates except through treatment propensity. Crump et al. [2008] devised a nonparametric test for the null hypothesis that the treatment effect is constant across patients, but that is not suited to high-dimensional data. One promising approach which flexibly handles high-dimensional and observational data is the gradient forest of Athey et al. [2017]—we compare the performance of our methods with that of the gradient forest in Section 4.7.

We are particularly interested in flexible, non-parametric approaches that can handle large numbers of observations and predictors, and model interactions between predictors, which none of these papers deal with (except for Zhao et al. [2012]).

4.2.1 Propensity score methods

Much of causal inference is based on the propensity score [Rosenbaum and Rubin, 1983], which is the estimated probability that a patient would receive treatment, conditioned on the patient's covariates. If the estimate of the propensity function (4.2) is $\hat{\pi}(\cdot)$, then the propensity score for a patient with covariate vector x is $\hat{\pi}(x)$. Throughout the present work, we estimate the propensity function using the probability forests of Malley et al. [2012]. We are able to do so quickly using the fast implement in the R package `ranger` [Wright and Ziegler, 2015].

For the estimation of a population-average treatment effect (ATE), propensity score methods for reducing bias in observational studies have been established [Austin, 2011]. *Propensity score matching* emulates a randomized control trial (RCT) by choosing pairs of patients with similar propensity scores, one each in the treatment and control arms, and discards the unmatched patients. *Stratification on the propensity score* groups patients into bins of similar propensity scores to compute the ATE within each bin. The overall ATE is the average of these treatment effects, weighted by the overall frequency of each bin. *Inverse probability weighting* assigns a weight to each patient equal to the inverse of the propensity score if the patient is treated, or else the inverse of one minus the propensity score if the patient is not treated. Hence patients who tend to be under-represented in their arm are given more weight. Propensity score stratification and inverse probability weighting are discussed in more detail in the appendix, along with an additional method: *transformed outcome averaging*.

The assumption that enables these methods to generate causal conclusions from observational

data is known alternatingly across the literature as unconfoundedness, exogeneity or strong ignorability:

$$(Y_i^{(1)}, Y_i^{(0)}) \perp\!\!\!\perp T_i | X_i$$

This is the assumption made in the present work. It means that the relationship between the potential outcomes and treatment must be fully explained by X. There can be no additional unmeasured confounding variable which effects a dependence between potential outcomes and treatment. Note, however, that the outcome itself is not independent of treatment because the treatment determines which potential outcome is observed.

Low et al. [2016] cast doubt on the ability of propensity score methods to adequately account for selection bias in a sophisticated simulation designed to model reality. Nevertheless, we observe in Section 4.7 that propensity score adjustments improve results in non-randomized simulations, which means that they can be used to help doctors make more informed decisions, so we push forward with the application of propensity scores.

4.3 Transformed outcome regression and conditional mean regression

Methods for estimating heterogeneous treatment effects generally fall into one of two categories: *transformed outcome regression* or *conditional mean regression*. In this section we describe the two approaches and explain why we prefer conditional mean regression. The propensity transformed outcome method (Section 4.4) uses a combination of the two approaches, while causal forests (Section 4.2), causal boosting (Section 4.5), and causal MARS (Section 4.6) are all conditional mean regression methods.

Transformed outcome regression is based on the same idea as transformed outcome averaging, which is laid out in detail in the appendix. Given the data described in Section 4.1, we define the *transformed outcome* as

$$Z \equiv T\frac{Y}{\pi(X)} + (1-T)\frac{-Y}{1-\pi(X)}.$$

This quantity is interesting because, as shown in the appendix, for any covariate vector x, $\mathbb{E}[Z|X = x] = \tau(x)$. So the transformed outcome gives us for each patient an unbiased estimate of the personalized treatment effect for that patient. Using this, we can simply use the tools of supervised learning to estimate a regression function for the mean of Z given X. The weakness of this approach is that while Z is unbiased for the treatment effect, its variance can be large due the presence of the propensity score, which can be close to zero or one, in the denominator.

An alternative approach, conditional mean regression is based on the idea that because $\tau(x)$ is defined as the difference between $\mu_1(x)$ and $\mu_0(x)$, if we can get good estimates of these conditional mean functions, then we have a good estimate of the treatment effect function. Estimating the

functions $\mu_1(x)$ and $\mu_0(x)$ are supervised learning problems. If they are both estimated perfectly, then there is no need to bother with propensity scores. The problem is that in practice we never estimate either function perfectly, and differences between the covariate distributions in the two treatment groups can lead to bias in treatment effect estimation if propensity scores are ignored.

We compare these two approaches with a simple example: Consider the task of estimating an ATE using data from a randomized trial. This may seem far removed from heterogeneous treatment effect estimation, but we will describe how two of our methods are based on estimating local ATEs for subpopulations in our data. In this case, the transformed outcome is

$$Z = T\frac{Y}{1/2} + (1-T)\frac{-Y}{1/2} = 2TY - 2(1-T)Y,$$

and the corresponding estimate of the ATE is

$$\hat{\tau}_{\text{TO}} = \frac{1}{n}\sum_{i=1}^{n} Z_i = \frac{2N_1\bar{Y}_1 - 2N_0\bar{Y}_0}{N_1+N_0} = \frac{N_1}{n/2}\bar{Y}_1 - \frac{N_0}{n/2}\bar{Y}_0,$$

where $\bar{Y}_1$ is the average response of patients who received treatment and $\bar{Y}_0$ is the average response of control patients. Meanwhile the conditional mean estimator of the ATE would be

$$\hat{\tau}_{\text{CM}} = \bar{Y}_1 - \bar{Y}_0.$$

Here we are implicitly assuming that neither N_1 nor N_0 is zero. It is worth noting that

$$\hat{\tau}_{\text{TO}} = \hat{\tau}_{\text{CM}} + \frac{N_1 - N_0}{n}(\bar{Y}_1 + \bar{Y}_0),$$

so if $N_1 = N_0$ or $\bar{Y}_1 + \bar{Y}_0 = 0$, then $\hat{\tau}_{\text{TO}} = \hat{\tau}_{\text{CM}}$. However N_1, N_0, $\bar{Y}_1$ and $\bar{Y}_0$ are all random. Given a fixed sample size n, N_1 follows a Binomial$(n, 1/2)$ distribution (truncated to exclude 0 and n), and N_0 is the difference between n and N_1. Suppose $\bar{Y}_1$ and $\bar{Y}_0$ have normal distributions with variances inversely proportional to sample size:

$$\bar{Y}_1 \sim \text{Normal}(\mu_1, \sigma^2/N_1) \qquad \text{and} \qquad \bar{Y}_0 \sim \text{Normal}(\mu_0, \sigma^2/N_0).$$

Note that both $\hat{\tau}_{\text{CM}}$ and $\hat{\tau}_{\text{TO}}$ are unbiased for $\tau \equiv \mu_1 - \mu_0$, but the two estimators have different variances. Conditioning on N_1, the variance of $\hat{\tau}_{\text{CM}}$ is

$$\mathbb{E}[(\hat{\tau}_{\text{CM}} - \tau)^2 | N_1] = \mathbb{V}(\bar{Y}_1 - \bar{Y}_0 | N_1) = \sigma^2/N_1 + \sigma^2/N_0$$

while the variance of $\hat{\tau}_{\text{TO}}$ given N_1 is

$$\mathbb{E}[(\hat{\tau}_{\text{TO}} - \tau)^2 | N_1] = \mathbb{V}(\hat{\tau}_{\text{TO}} | N_1) + (\mathbb{E}[\hat{\tau}_{\text{TO}} - \tau | N_1])^2 = \frac{4}{n}\sigma^2 + \left(\frac{N_1 - N_0}{n}\right)^2 (\mu_1 + \mu_0)^2.$$

So the key is the ratio of the main effect $(\mu_1 + \mu_0)/2$ to the noise level σ. If

$$\left|\frac{\mu_1 + \mu_0}{2\sigma}\right| < \sqrt{\frac{N_1^{-1} + N_0^{-1} - 4n^{-1}}{(N_1 - N_0)^2}},$$

then $\hat{\tau}_{\text{TO}}$ has less variance. If the inequality is reversed, then $\hat{\tau}_{\text{CM}}$ has less variance. Marginalizing over the truncated binomial distribution of N_1 is difficult to do analytically, but we can numerically estimate the marginal variance of each estimator for any $n > 1$. Figure 4.1 illustrates the results for a few different choices of n.

We observe that for small n, $\hat{\tau}_{\text{TO}}$ can have slightly smaller variance than $\hat{\tau}_{\text{CM}}$ if the absolute value of the main effect is close to zero. But this advantage tends to zero as n increases, and $\hat{\tau}_{\text{TO}}$ has much greater variance if the main effect is large. In conclusion, we prefer the conditional mean estimator because of the potentially high variance of the transformed outcome estimator. This is reflected in the following sections as all of our methods use some version of conditional mean regression.

4.3.1 Shared-basis conditional mean regression

In high-dimensional data it is often necessary to choose a subset of variables to include in a model. Beyond that, nonparametric methods adaptively choose transformations of variables. Collectively, we refer to the variables and transformations selected as the basis of the regression. In conditional mean regression it is to be expected that the selected basis be different between the two regression functions. This can cause differences between the conditional means attributable not to evidence of a heterogeneous treatment effect but rather due to chance in basis selection.

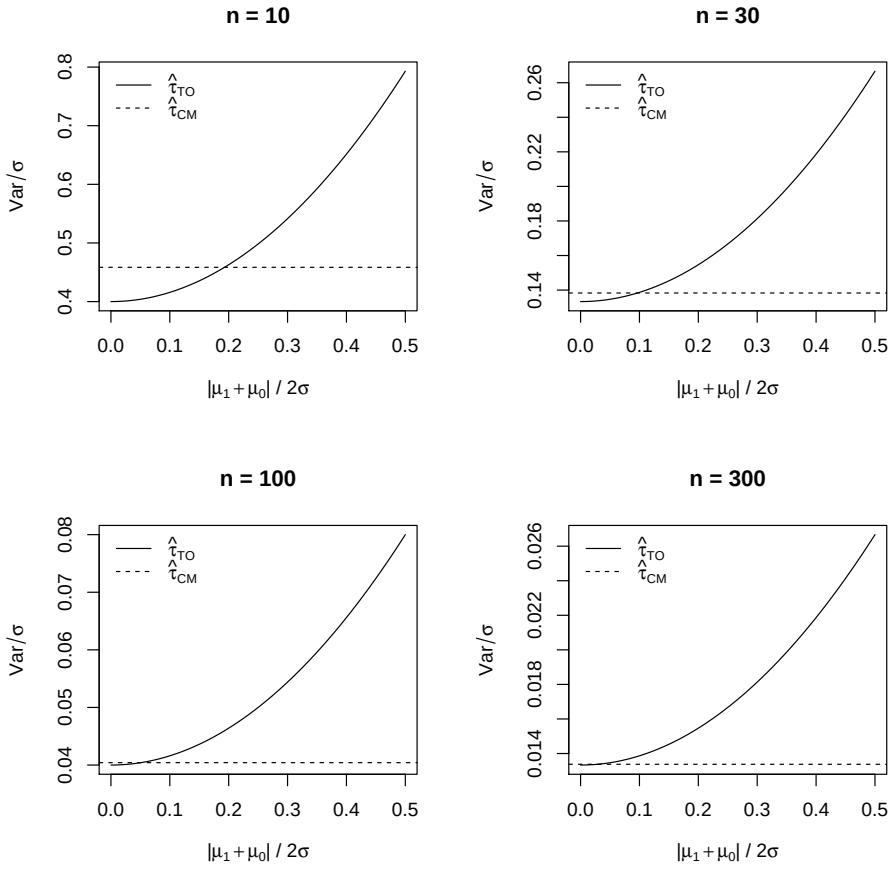
To address this all of our methods jointly choose the same basis for both conditional mean regressions. In detail, this shared basis is chosen adaptively to best explain heterogeneity in the treatment effect, rather than explaining the variance in either treatment group. How exactly this shared basis is determined is different for each method.

4.4 Pollinated transformed outcome (PTO) forests

We first present the idea of a pollinated transformed outcome (PTO) forest in detail and then explain the various components.

In step 1 we compute an unbiased point estimate of the treatment effect for each individual; then in step 2, we fit a random forest using this effect as the outcome. In principal, this should estimate

Figure 4.1: The variance of two ATE estimators for $n = 10$, 30, 100 and 300, as the ratio of the absolute main effect $|\mu_1 + \mu_0|/2$ to the noise level σ increases from 0 to 0.5.



Algorithm 1: PTO forest

1. Build a depth-controlled propensity random (regression) forest $\hat{\pi}$ using the treatment indicator as the response. Use regression trees, so that we estimate the probability of the terminal-node means. If the data are known to have come from a randomized trial, do not build a random forest and instead define $\hat{\pi}$ to be identically equal to the probability of treatment assignment.

2. Define the transformed outcome by

$$Z_i = T_1 \frac{Y_i}{\hat{\pi}(X_i)} + (1 - T_i) \frac{-Y_i}{\hat{\pi}(X_i)}.$$

3. For the randomized treatment setting, define the transformed outcome by

$$\delta_i = (2T_i - 1)Y_i.$$

   Note that if $\hat{\pi}(X_i)$ is the true probability of receiving treatment given covariates X_i, then $E[Z_i|T_i, X_i] = \tau(X_i)$, the true conditional treatment effect (see appendix for details).

4. Grow a depth-controlled random forest G_{TOF} to δ_i.

5. Pollinate G_{TOF} separately with the data in the treated group and the control group to produce two regression forests G_1 and G_0, respectively. This entails sending each observation in the treatment group down each tree in the forest to determine its terminal node and re-estimating the response in that node to be the average of its observations. The same is done for the control group.

6. Compute $\delta_i = G_1(X_i) - G_0(X_i)$.

7. Optionally, fit a random forest S to δ_i and return S, which predicts the treatment effect $\hat{\tau}(x) = S(x)$. This optional layer of regression also helps with the interpretability of the results, yielding importance scores for variables as they relate directly to the estimated treatment effect.

our personalized treatment effect. However, we don't trust these estimates too much, because the outcome can be highly variable. But we will put faith in the trees they produced.

Thus in step 3, we "pollinate" the trees separately with the treated and untreated populations. That is, we send data down each tree and compute new predictions for each terminal node. In step 4, the difference $z_i = G_1(x_i) - G_0(x_i)$ gives us an estimate of the treatment effect. Finally in step 5, we then post-process these predictions by fitting one more forest, primarily for interpretation.

Figure 4.2 illustrates the benefits of cross-pollination. In this example $n = 100, p = 50$ and the response is simulated in each arm according to $Y_i \sim \mathcal{N}(1 - X_{i1} + X_{i2}, 1)$ for treated patients and $Y_i \sim \mathcal{N}(X_{i1} + X_{i2}, 1)$ for untreated patients. Hence the true personalized treatment effect for patient i is $1 + 2X_{i1}$. In the top row the treatment is randomly assigned, while in the bottom row, the probability of treatment assignment is $(1 + e^{X_{i1}+X_{i2}})^{-1}$. The raw estimates correspond to a random forest (as in step 2) grown to predict the transformed outcome. The pollinated estimates correspond to re-estimating (as in step 3) the means of the leaves within each arm. We observe that in each case, the pollination improves the estimates.

4.5 Causal boosting

An alternative to a random forest for least squares regression is boosted trees. Boosting builds up a function approximation by successively fitting weak learners to the residuals of the model at each step. In this section we generalize least squares boosting for regression [Friedman, 2001] to the problem of heterogeneous treatment effect estimation.

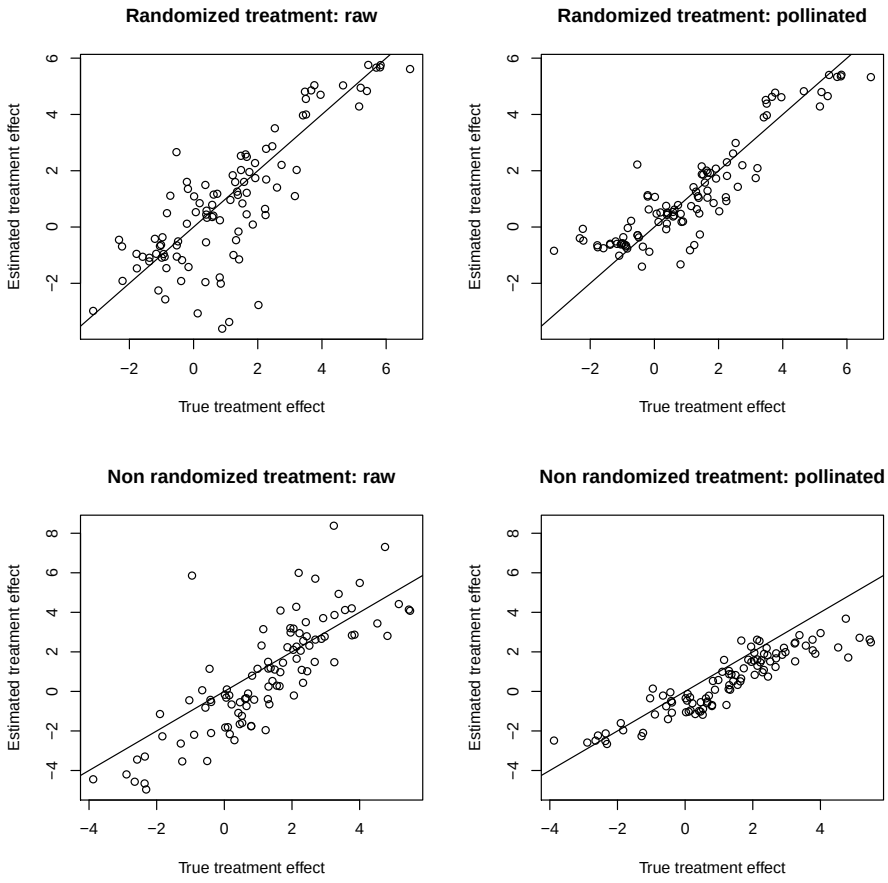
Given data of the form $(X_i, Y_i), i = 1, ..., n$, least squares boosting starts with a regression function $\hat{F}(x) = 0$ and residuals $R_i = Y_i - \hat{F}(x_i)$. We fit a regression tree to R_i, yielding predictions $\hat{f}_1(x)$. Then we update $\hat{F}(x) \leftarrow \hat{F}(x) + \epsilon \cdot \hat{f}_1(x)$, and $R_i \leftarrow R_i - \epsilon \cdot \hat{f}_1(x_i)$ and repeat this (say) a few hundred times. The final prediction is simply $\hat{F}(x)$, a sum of trees shrunk by ϵ.

For our current problem, our data has the form $(X_i, T_i, Y_i), i = 1, ..., n$ with $T_i \in \{0, 1\}$. For now assume randomized treatment assignment. In the next subsection we show to handle the non-randomized case. Here is how we propose to generalize least squares boosting. As with causal forests Wager and Athey [2015], our building block is a causal tree, which returns a function $\hat{g}(x, t)$. The estimated causal effect for an observation $X = x$ is $\hat{\tau}(x) = \hat{g}(x, 1) - \hat{g}(x, 0)$. This is a standard causal tree, except that for each terminal node, we return the pair of treatment-specific means rather than the treatment effect. In other words, if observation $X_i = x$ gets you into terminal node k, where the pair of estimated means are $\hat{\mu}_{1k}$ (treated) and $\hat{\mu}_{0k}$ (untreated), then these are the values returned, respectively, for $\hat{g}(x, 1)$ and $\hat{g}(x, 0)$. The algorithm is summarized in Algorithm 2 below.

The estimated treatment effect for any observation x is $\hat{G}_K(x, 1) - \hat{G}_K(x, 0)$.

Note that this generalizes to loss functions other than squared error. For example, if the causal tree was trained for a binary outcome, then each terminal node would return a pair of logits $\hat{\eta}_{1k} =$

Figure 4.2: A comparison of raw and pollinated transformed outcome forests. Each method is applied to a randomized simulation and a non-randomized simulation, and we visually compare the estimated treatment effect with the true treatment effect We see that in each case, the pollination improves the estimates.



$\text{logit}[\Pr(Y = 1|X = x, T = 1)]$ and $\hat{\eta}_{0k} = \text{logit}[\Pr(Y = 1|X = x, T = 0)]$. Thus $\hat{G}_K(x, T)$ would be a function that returned a pair of logits at x, and hence treatment success probabilities. The treatment effect would be the appropriate function of these (difference, log-odds ratio). Other enhancements to boosting, such as stochastic boosting, are also applicable in the setting.

Note that causal boosting is not strictly a gradient boosting algorithm, because there is no loss function for which we are evaluating the gradient at each step, in order to minimize this loss. Rather, causal boosting is an adaptation of gradient boosting on the observed response, with a different function in each arm of the data. The adaptation is that we use causal trees as our weak learners instead of a standard regression technique. This tweak encourages the learned function to find treatment effect heterogeneities.

Algorithm 2: Causal Boosting

1. Set the outcome $R_i = Y_i$, and define $\hat{G}_0(x,t) = 0$.
2. Do $k = 1, ..., K$
   (a) Fit a causal tree $\hat{g}_k$ to data (X_i, R_i, T_i).
   (b) Set
   $$\begin{aligned} R_i &\leftarrow R_i - \epsilon \cdot \hat{g}_k(X_i, T_i) \\ \hat{G}_k &\leftarrow \hat{G}_{k-1} + \epsilon \cdot \hat{g}_k. \end{aligned}$$
3. Return $\hat{G}_K(x,T)$.

4.5.1 Cross-validation for causal boosting

Unlike random forests, gradient boosting algorithms can over-fit the training data as the number of trees increases [Hastie et al., 2009]. This is because each successive tree is not built independently of the previous ones but rather with the goal of fitting to the residuals of the previous trees. Whereas a random forest will only benefit from using more trees, the number of trees in gradient boosting is itself an important parameter which needs to be tuned.

Complicating matters, the usual cross-validation framework does not apply to the setting of estimating a heterogeneous treatment effect because in this setting each observation does not come with a response corresponding directly to the function we are interested in estimating. We don't observe a response τ_i for the i^{th} patient. What we observe is either $Y_i^{(0)}$ or $Y_i^{(1)}$, depending on whether or not the patient received the treatment.

We describe our approach in the context of a held-out validation set, but this fully specifies our cross-validation procedure. Cross-validation is simply validation done by partitioning the training set into several folds and averaging the results obtained by holding out each fold as a validation set and training on all other folds. The data in this context are a training set $(\mathbf{X}^{tr}, \mathbf{T}^{tr}, \mathbf{Y}^{tr})$ and a validation set $(\mathbf{X}^{v}, \mathbf{T}^{v}, \mathbf{Y}^{v})$. After training causal boosting on $(\mathbf{X}^{tr}, \mathbf{T}^{tr}, \mathbf{Y}^{tr})$, we are left with a sequence of models $G_1(x,T), ..., G_K(x,T)$, and we would like to evaluate the performance of each of these.

To validate the performance of each of these models, we use a pollination of the causal boosting model much like step 3 of the PTO forest. We run through the causal boosting algorithm again, making all the same splits as in the original training. The difference is in how we estimate the value returned in each node of each shallow causal tree. As in causal forests and in step 3 of the transformed outcome forest, we use $(\mathbf{X}^{(tr)}, \mathbf{T}^{(tr)}, \mathbf{Y}^{(tr)})$ to populate the nodes of the constituent causal trees and estimate the ATE within each node. The residuals r_i from the causal boosting algorithm are initialized to be the y_i from the validation set and are updated according to these

re-fitted trees. The result is a new "honest" sequence of models $H_1(x,T),...,H_K(x,T)$.

We are ready to define our validation error for each of the original models $G_1(x,T),...,G_K(x,T)$. The validation error for a causal boosting model with k trees is given by

$$\sum_{x \in v} \left(\{G_k(x,1) - G_k(x,0)\} - \{H_K(x,1) - H_K(x,0)\}\right)^2.$$

We have several remarks to make about this form. $G_k(x,1) - G_k(x,0)$ is the estimated treatment effect at x, for causal boosting with k trees. $H_K(x,1) - H_K(x,0)$ is the estimated treatment effect correspond to the maximum number of trees, *using the responses from the validation set.* For a large number of trees, we can be sure that this is over-fitting to the response, and this is the analog of traditional cross-validation, which compares predictions on the validation set with observed response in the validation set. This observed response, corresponding to the saturated model, is as over-fitted as possible. Intuitively, we are comparing our estimated treatment effect for each validation point against another estimate, which uses the same structure as the model fit to find similar patients and estimate the treatment effect based on those similar patients, some of whom will have received treatment, some of who will have received control. The better the structure is that causal boosting has learned for the heterogeneous treatment effect, the more the local ATE in the training set will mirror the local ATE in the validation set. For the results in Section 4.7, we use this procedure to do cross-validation for causal boosting.

4.5.2 Within-leaf propensity adjustment

When the goal is to estimate not an ATE but rather an individualized treatment effect, the propensity score methods described in Section 4.2.1 and in the appendix do not immediately extend. Consider for example propensity score stratification. Because each patient belongs to only one stratum of propensity score, we can not average treatment effect estimates for a patient across strata. Technically, if we were to fit a causal boosting model within each stratum, each of these models would be able to make a prediction for the query patient. But then all but one of these models would be unwisely extrapolating outside of its training set to make this prediction. An alternative to propensity score stratification, inverse probability weighting is still viable, but the volatility of this method is exacerbated by the attempt to estimate a varying treatment effect, rather than a constant one.

Within each leaf of a causal tree, however, we estimate an ATE. This is where causal boosting adjusts for non-random treatment assignment, using propensity score stratification to reduce the bias in the estimate of the within-leaf ATE. Before initiating the causal boosting algorithm, we begin by evaluating the propensity score for each patient, which is an estimate of probability of being assigned the treatment, conditioned on the observed covariates. Any binomial regression technique could be used here. We fit a probability forest [Malley et al., 2012], which is similar to a random forest for classification [Breiman, 2001] except that each tree returns a probability estimate rather

than a classification. The trees are combined by averaging the probability estimates and not by majority vote. We denote the treatment assignment probability as a function of the covariates by $\pi(x) \equiv \mathbb{P}(T = 1|X = x)$ and the corresponding propensity scores by $\hat{\pi}_i \equiv \hat{\pi}(x_i)$.

We group the patients into S strata of similar propensity scores denoted $1, ..., S$. For example, there could be $S = 10$ strata, with the first comprising $\hat{\pi} \in [0, 0.1)$ and the last comprising $\hat{\pi} \in [0.9, 1]$, with equal-length intervals in between. We use $s_i \in \{1, ..., S\}$ to denote the stratum to which patient i belongs. Hence the data that we observe within each leaf of a causal tree are of the form $(X_i, s_i, T_i, Y_i) \in \mathbb{R}^p \times \{1, ..., S\} \times \{0, 1\} \times \mathbb{R}$. We use n_ℓ to denote the number of patients in leaf ℓ and index these patients by $i = 1, ..., n_\ell$. The propensity-adjusted ATE estimate in leaf ℓ is given by

$$\hat{\tau}_\ell = \frac{\sum_{s=1}^{S} n_{s\ell}(\bar{Y}_{1s\ell} - \bar{Y}_{0s\ell})}{\sum_{s=1}^{S} n_{s\ell}}, \text{ where } \bar{Y}_{ts\ell} = \frac{\sum_{i=1}^{n_\ell} \mathbb{I}_{\{T_i=t \wedge s_i=s\}} Y_i}{n_{ts\ell}} \tag{4.3}$$

is the mean response among the treatment ($t = 1$) or control ($t = 0$) group in stratum s, and $n_{ts\ell} = \sum_{i=1}^{n_\ell} \mathbb{I}_{\{s_i=s\}}$ is the corresponding number of patients in leaf ℓ for $t \in \{0, 1\}, s \in \{1, ..., S\}$. Finally, $n_{s\ell} = n_{1s\ell} + n_{0s\ell}$.

The estimated variance of $\hat{\tau}_\ell$ is

$$\widehat{\text{Var}}(\hat{\tau}_\ell) = \frac{\sum_{s=1}^{S} n_{s\ell}^2 \hat{\sigma}_{s\ell}^2}{(\sum_{s=1}^{S} n_{s\ell})^2}, \text{ where } \hat{\sigma}_{s\ell}^2 = \frac{s_{1s\ell}^2}{n_{1s\ell}} + \frac{s_{0s\ell}^2}{n_{0s\ell}},$$

and $s_{ts\ell}^2$ is the sample variance of the response for arm t of stratum s in leaf ℓ.

Hence, for two candidate daughter leaves ℓ and r of the same parent, The natural extension of the squared T-statistic splitting criterion from Athey and Imbens [2016] is

$$\frac{|\hat{\tau}_\ell - \hat{\tau}_r|}{\sqrt{\widehat{\text{Var}}(\hat{\tau}_\ell) + \widehat{\text{Var}}(\hat{\tau}_r)}}.$$

This is the propensity-stratified splitting criterion used by causal boosting. This criterion could also be used by a causal forest as it applies directly to its constituent causal trees.

We use this propensity adjustment not only for determining the split in a causal tree but also for estimating the treatment effect in the node. Specifically, the causal tree returns two values in each leaf: the propensity-adjusted mean response in the treatment and control groups.

$$\frac{\sum_{s=1}^{S} n_{s\ell} \bar{Y}_{1sl}}{\sum_{s=1}^{S} n_{s\ell}} \quad \text{and} \quad \frac{\sum_{s=1}^{S} n_{s\ell} \bar{Y}_{0sl}}{\sum_{s=1}^{S} n_{s\ell}}.$$

4.6 Causal MARS

One drawback to tree-based methods is that because they use the average treatment effect within each leaf as the prediction for that leaf, there could be high bias in this estimate. This is especially

problematic when it comes to confidence interval construction for personalized treatment effects. The variance of the estimated treatment effect is relatively straightforward to estimate, but the bias presents more of a challenge.

Multivariate adaptive regression splines (MARS, Friedman [1991]) can be thought of as a modification to CART which alleviates this bias problem. MARS starts with the constant function $f(x) = \beta_0$ and considers adding pairs of functions of the form $\{(x_j - c)_+, (c - x_j)_+\}$ and also the products of variables in the model with these pairs, choosing the pair which lead to the greatest drop in training error when they are added to their model, with regression coefficients estimated via OLS. The difference between this and CART is that in CART the pairs of functions considered are of the form $\{\mathbb{I}_{\{x_j - c \geq 0\}}, \mathbb{I}_{\{c - x_j > 0\}}\}$, and when a product with one of the included terms in chosen, it replaces the included term in the model [Hastie et al., 2009].

We propose causal MARS as the adaptation of MARS to the task of treatment effect estimation. We fit two MARS models in parallel in the two arms (treatment and control) of the data, at each step choosing the same basis functions to add to each model. The criterion that we use identifies the best basis in terms of explaining treatment effect: we compare the drop in training error from including the basis in both models with different coefficients to the drop in training error from including the basis in both models with the *same* coefficient in each model. The steps of causal MARS are as follows. The parameter D controls the maximum dimension of the regression basis, and in practice we use 11 in our examples. Algorithm 3 has the details.

To reduce the variance of causal MARS, we perform bagging by taking B bootstrap samples of the original dataset and fitting the causal MARS model to each one. The estimated treatment effect for an individual is the average of the estimates for this individual by the B models.

Note that the algorithm described above applies to the randomized case, not observational data. Given S propensity strata and membership $s \in 1, ..., S$, for each patient, we use the same basis functions within each stratum but different regression coefficients. Within each stratum, the coefficients are estimated separately from the coefficients in other strata. Given the entry criterion $dRSS_s$ and number of patients n_s in each stratum, we combine these into a single criterion $\sum_s n_s dRSS_s$. This is the *propensity-adjusted* causal MARS.

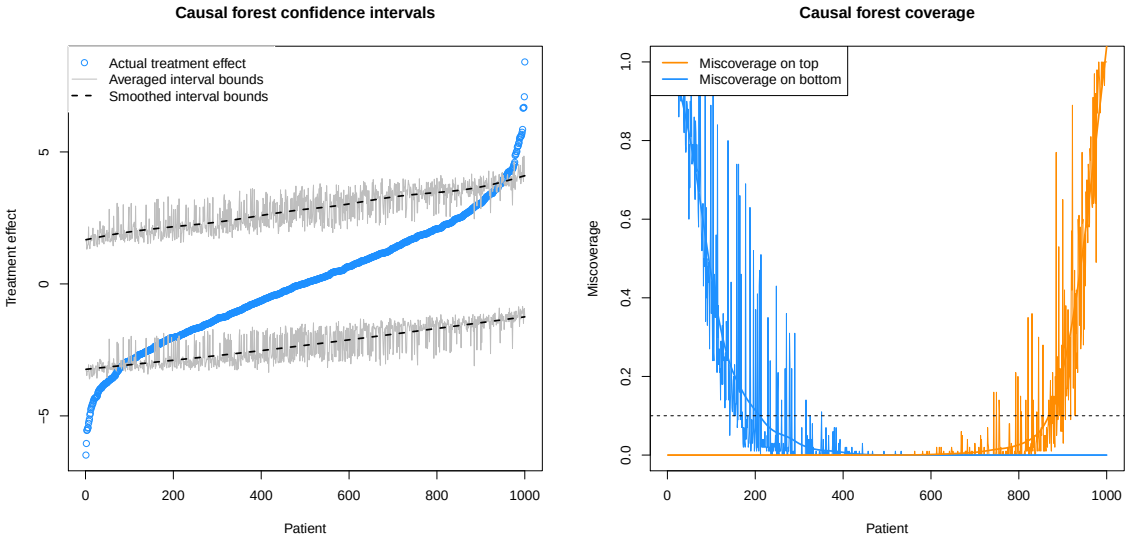
4.6.1 Confidence intervals

One advantage of the bagging-based methods—causal forest and causal MARS—is that in the process of computing the treatment-effect estimates, one gets at no extra cost the computations necessary to estimate the variance of the estimators. Each of the bagged models is based on its own bootstrap re-sampling of the data, so for each patient we have B re-sampled treatment effect estimates, where B is the number of bags. We propose using the quantiles of these estimates as the confidence interval for each patient. To construct a $(1 - \alpha)$ confidence interval, we use the $\alpha/2$ and $1 - \alpha/2$ quantiles of the bootstrapped estimates as lower and upper bounds, respectively.

Algorithm 3: Causal MARS

1. Define $\mathcal{F} = \{\{(x_j - c)_+, (c - x_j)_+\} : c \in \{\mathbf{X}_{ij}\}, j \in \{1, ..., p\}\}$.
2. Initialize $\mathcal{B} = \{1\}$.
3. For d in $1, ..., D$: (growing the model)
   (a) For each pair of functions $\{f, g\} \in \{\{b(x)f^*(x), b(x)g^*(x)\} : b \in \mathcal{B}, \{f^*, g^*\} \in \mathcal{F}\}$:
      i.
$$RSS_\mu = \min_{\beta^1, \beta^0} \sum_{i=1}^{n} \left(y_i - \sum_{b \in \mathcal{B}} (\beta_b^1 b(x_i) \mathbb{I}_{\{t_i=1\}} + \beta_b^0 b(x_i) \mathbb{I}_{\{t_i=0\}}) - \sum_{h \in \{f,g\}} \beta_h h(x_i) \right)^2$$
      ii.
$$RSS_\tau = \min_{\beta^1, \beta^0} \sum_{i=1}^{n} \left(y_i - \sum_{b \in \mathcal{B}} (\beta_b^1 b(x_i) \mathbb{I}_{\{t_i=1\}} + \beta_b^0 b(x_i) \mathbb{I}_{\{t_i=0\}}) - \sum_{h \in \{f,g\}} (\beta_h^1 h(x_i) \mathbb{I}_{\{t_i=1\}} + \beta_h^0 h(x_i) \mathbb{I}_{\{t_i=0\}}) \right)^2$$
      iii.
$$dRSS = RSS_\tau - RSS_\mu$$
   (b) Choose $\{f, g\}$ which maximize $dRSS$ and add them to $\mathcal{B}$.
4. Backward deletion: delete terms one at a time, using the same criterion as in the forward stepwise 3(a). Use the out-of-bag error to estimate the optimal model size.

Figure 4.3: Confidence intervals for causal forest in Scenario 8 from Section 4.7. On the left in blue we plot the true treatment effect for each patient against the index of the patient, sorted by treatment effect. The thin gray lines show the average upper and lower confidence interval bounds for each patient, and the dotted black line smooths over these averages. On the left the thin lines give the miscoverage rate for each patient, and the thick lines smooth over these thin lines. These results reflect 100 simulations using 50 bagged causal trees.



Note that this procedure is targeted at the variability of a single causal tree or a single causal MARS model, but the methods we propose involve averaging these models to reduce their variance. This will make our intervals more conservative because the variance of the bagged models will be lower than the variance of the individual models. However, as the results in this section demonstrate, the conservative nature of these confidence intervals helps with coverage problems due to the inability to fully remove the bias from the treatment effect estimates.

Figure 4.3 shows confidence interval results for causal forest applied to Simulation 8 in Section 4.7. That section describes in detail our simulation scheme, but in this section we use it only as an illustration of the confidence interval results. The left figure shows the average upper and lower bounds of the confidence interval for each patient, across 100 simulations. This demonstrates the difficulty with constructing confidence intervals for random forest predictions: Because of the relatively high bias from using the average as the estimate within each leaf, the confidence intervals do not come close to maintaining $(1 - \alpha)$ coverage for patients with relatively small or relatively large treatment effects.

This problem for causal forests was the motivation for the development of causal MARS. By using piecewise linear models instead of piecewise constant models, MARS can achieve lower bias than regression trees, which is important for bootstrap confidence-interval construction. Figure 4.4 shows

Figure 4.4: Confidence intervals for causal MARS in Scenario 8 from Section 4.7. On the left in blue we plot the true treatment effect for each patient against the index of the patient, sorted by treatment effect. The thin gray lines show the average upper and lower confidence interval bounds for each patient across 100 simulations, and the dotted black line smooths over these averages. On the left the thin lines give the miscoverage rate for each patient, and the thick lines smooth over these thin lines. These results reflect 100 simulations using 50 bagged causal MARS models.

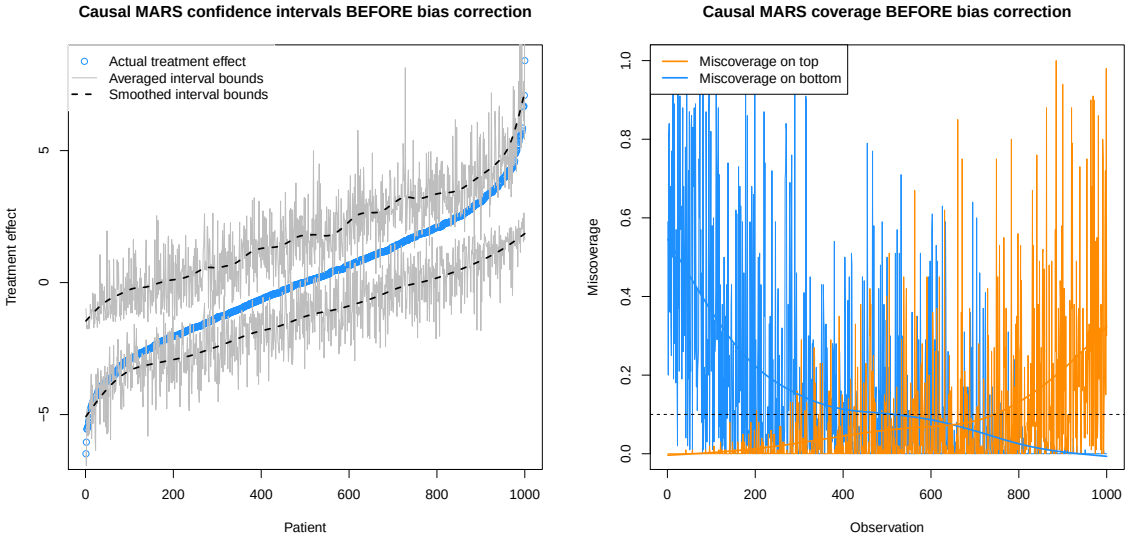
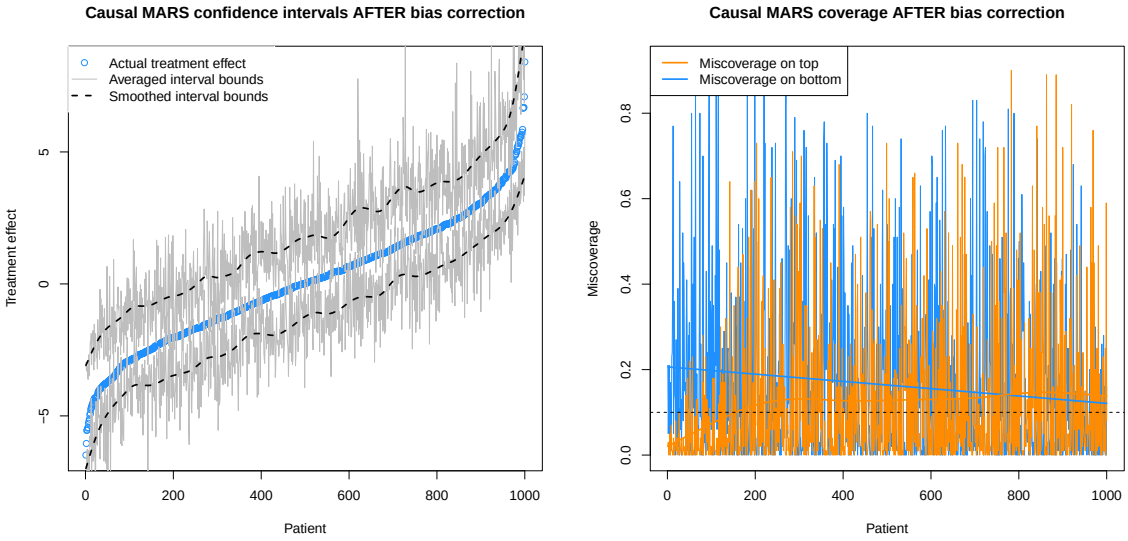


Figure 4.5: Bias-corrected confidence intervals for causal MARS in Scenario 8 from Section 4.7. On the left in blue we plot the true treatment effect for each patient against the index of the patient, sorted by treatment effect. The thin gray lines show the average upper and lower confidence interval bounds for each patient across 100 simulations, and the dotted black line smooths over these averages. On the left the thin lines give the miscoverage rate for each patient, and the thick lines smooth over these thin lines. These results reflect 100 simulations using 50 bagged causal MARS models.



the results of constructing confidence intervals for the causal MARS estimates in a single simulation. The average confidence intervals are more volatile in Figure 4.4 than in Figure 4.3 because causal MARS is a higher-variance method. But we see that the confidence intervals adhere more closely to the true treatment effect for this method than for the causal forest. Examining the coverage, we see that there is still a bias problem for treatment effects near the edges of the range of values, but the miscoverage is closer to 0.5, an improvement of the coverage which approaches 1 for causal forest.

Still, bagged causal MARS has not fully mitigated the bias problem. We see that the miscoverage on bottom is decreasing with the true treatment effect, and the miscoverage on top is increasing with the true treatment effect. We attempted to address this with a bias correction. We bootstrapped residuals from the fitted model and applied a standard bootstrap bias correction. The results of this correction are shown in Figure 4.5. Here the confidence intervals adhere even more closely to the true treatment effect, and the coverage is improved. The miscoverage on either side of the confidence interval is capped at 0.2 when smoothed, though the target miscoverage rate is 0.1. We have taken steps toward constructing confidence intervals for personalized treatment effects, but it remains an area for future research.

4.7 Simulation study

In the design of our simulations to evaluate performance of methods for heterogeneous treatment effect estimation, there are four elements to the generation of synthetic data:

1. The number n of patients in the training set, and the number p of features observed for each patient.

2. The distribution $\mathcal{D}_X$ of the feature vectors X_i. Across all scenarios, we draw odd-numbered features independently from a standard Gaussian distribution. We draw even-numbered features independently from a standard Bernoulli distribution.

3. The propensity function $\pi(\cdot)$, the mean effect function $\mu(\cdot)$ and the treatment effect function $\tau(\cdot)$. We take the conditional mean effect functions to be $\mu_1(x) = \mu(x) + \tau(x)/2$ and $\mu_0(x) = \mu(x) - \tau(x)/2$.

4. The conditional variance σ_Y^2 of Y_i given X_i and T_i. This corresponds to the noise level, and we choose is to make the percentage of null variance explained of the true model to be roughly 20-25%. This ensures we are comparing the methods on relevant simulations.

Given the elements above, our data generation model is, for $i = 1, ..., n$:

$$X_i \overset{\text{i.i.d.}}{\sim} \mathcal{D}_X$$

$$T_i \overset{\text{ind.}}{\sim} \text{Bernoulli}(\pi(X_i))$$

$$Y_i \overset{\text{ind.}}{\sim} \text{Normal}\left(\mu(X_i) + (T_i - 1/2)\tau(X_i), \sigma_Y^2\right)$$

The third element above, encompassing $\pi(\cdot)$, $\mu(\cdot)$ and $\tau(\cdot)$, is most interesting. Note that $\pi(\cdot)$ and $\mu(\cdot)$ are nuisance functions, and $\tau(\cdot)$ is the function we are interested in estimating. In this section, we present two batches of simulations, the first of which represent randomized experiments. The second batch of simulations represent observational studies. Within each set of simulations, we make eight different choices of mean effect function and treatment effect function, meant to represent a wide variety of functional forms: both univariate and multivariate; both additive and interactive;

Table 4.1: Specifications for the 16 simulation scenarios. The four rows of the table correspond, respectively, to the sample size, dimensionality, mean effect function, treatment effect function and noise level. Simulations 1 through 8 use randomized treatment assignment, meaning $\pi(x) = 1/2$. Simulations 9 through 16 have a bias in treatment assignment, specified by (4.4).

Scenarios	1, 9	2, 10	3, 11	4, 12	5, 13	6, 14	7, 15	8, 16
n	200	200	300	300	400	400	1000	1000
p	400	400	300	300	200	200	100	100
$\mu(x)$	$f_8(x)$	$f_5(x)$	$f_4(x)$	$f_7(x)$	$f_3(x)$	$f_1(x)$	$f_2(x)$	$f_6(x)$
$\tau(x)$	$f_1(x)$	$f_2(x)$	$f_3(x)$	$f_4(x)$	$f_5(x)$	$f_6(x)$	$f_7(x)$	$f_8(x)$
σ^2_Y	1	1/4	1	1/4	1	1	4	4

both univariate and multivariate. The eight functions that we chose are:

$$f_1(x) = 0 \qquad f_2(x) = 5\mathbb{I}_{\{x_1>1\}} - 5 \qquad f_3(x) = 2x_1 - 4$$

$$f_4(x) = x_2x_4x_6 + 2x_2x_4(1-x_6) + 3x_2(1-x_4)x_6 + 4x_2(1-x_4)(1-x_6) + 5(1-x_2)x_4x_6$$
$$+6(1-x_2)x_4(1-x_6) + 7(1-x_2)(1-x_4)x_6 + 8(1-x_2)(1-x_4)(1-x_6)$$

$$f_5(x) = x_1 + x_3 + x_5 + x_7 + x_8 + x_9 - 2$$

$$f_6(x) = 4\mathbb{I}_{\{x_1>1\}}\mathbb{I}_{\{x_3>0\}} + 4\mathbb{I}_{\{x_5>1\}}\mathbb{I}_{\{x_7>0\}} + 2x_8x_9$$

$$f_7(x) = \frac{1}{2}\left(x_1^2 + x_2 + x_3^2 + x_4 + x_5^2 + x_6 + x_7^2 + x_8 + x_9^2 - 11\right)$$

$$f_8(x) = \frac{1}{\sqrt{2}}\left(f_4(x) + f_5(x)\right)$$

Each of the eight functions above is centered and scaled so that with respect to the distribution $\mathcal{D}_X$, each has mean close to zero and all have roughly the same variance. Table 4.1 gives the mean and treatment effect functions for the eight randomized simulations, in terms of the eight functions above. In these simulations $\pi(x) = 1/2$ for all $x \in \mathbb{R}^p$. In addition to the methods described in Sections 4.4, 4.5 and 4.6, we include results for two additional estimators for comparison. The null estimator is simply the difference $\bar{Y}_1 - \bar{Y}_0$ in mean response between treated and untreated patients. This provides a naive baseline. The other competitor is the gradient forest of Athey et al. [2017], using the `gradient.forest` R package made available online by the authors. The results of the fist batch of simulations are shown in Figure 4.6.

If we pick "winners" in each of the simulation scenario based on which method has the lowest distribution of errors, causal MARS would win Scenarios 5, 7 and 8, tying with the pollinated transformed outcome forest in Scenario 4. The PTO forest would win Scenarios 2 and 3, tying with causal boosting in Scenario 6. In general all of the methods outperform the null estimator except in Scenario 1, when the treatment effect is constant, and in Scenario 6, when the gradient forest perform worst.

The second batch of simulations matches the parameters listed in Table 4.1: Scenario 9 is like Scenario 1; Scenario 10 is like Scenario 2; and so on. The difference is in the propensity function. For this second batch of simulations, we use

$$\pi(x) = \frac{e^{\mu(x)-\tau(x)/2}}{1+e^{\mu(x)-\tau(x)/2}}. \tag{4.4}$$

The interpretation of this propensity function is that patients with greater mean effect are more likely to receive the treatment. This resembles a situation in which greater values of the outcome are worse for the patient, and only patients who have need for treatment will receive it. There are many possible forms for the propensity function, but we focus on this one because it is particularly troublesome, and a good estimator of the treatment effect needs to avoid the pitfall of estimating to great an effect because the treated patients have greater mean effect. This is exactly the kind of bias we are most concerned about in observational studies. The results of this second batch of simulations are shown in Figure 4.7.

In the batch of simulations with biased treatment assignments, propensity-adjusted causal boosting shines. In six of the eight simulations, causal boosting as either the lowest error distribution or is one of the two methods with the lowest error distribution. Curiously, in Scenario 13, unadjusted causal MARS performs very well, but the propensity adjustment ruins this performance. In Scenario 15, PTO forest and gradient forest produce the best results though all of the methods perform well. Overall, across the 16 simulation scenarios, causal boosting and causal MARS stand out as having the best performance.

4.8 Application

In September 2016, *New England Journal of Medicine* opened The SPRINT Data Analysis Challenge, based on the complete dataset from a randomized trial of a novel intervention for the treatment of high blood pressure [SPRINT Research Group, 2015]. The goal was open-ended: to draw novel or clinically useful insights from the SPRINT dataset, possibly in tandem with other publicly available data.

The intervention in the randomized trial [SPRINT Research Group, 2015] was a more intensive control of systolic blood pressure (target 120 mm Hg) than is standard (target 140 mm Hg). The

Figure 4.6: Results across eight simulated randomized experiments. For details of the generating distributions, see Table 4.1. The seven estimators being evaluated are: NULL = the null prediction, GF = gradient forest, PTO0 = pollinated transformed outcome forest (using propensity = 1/2), CB0 = causal boosting, CM0 = causal MARS. The vertical blue bar shows the standard deviation of the response, for assessing the practical significance of the difference between the methods' performances.

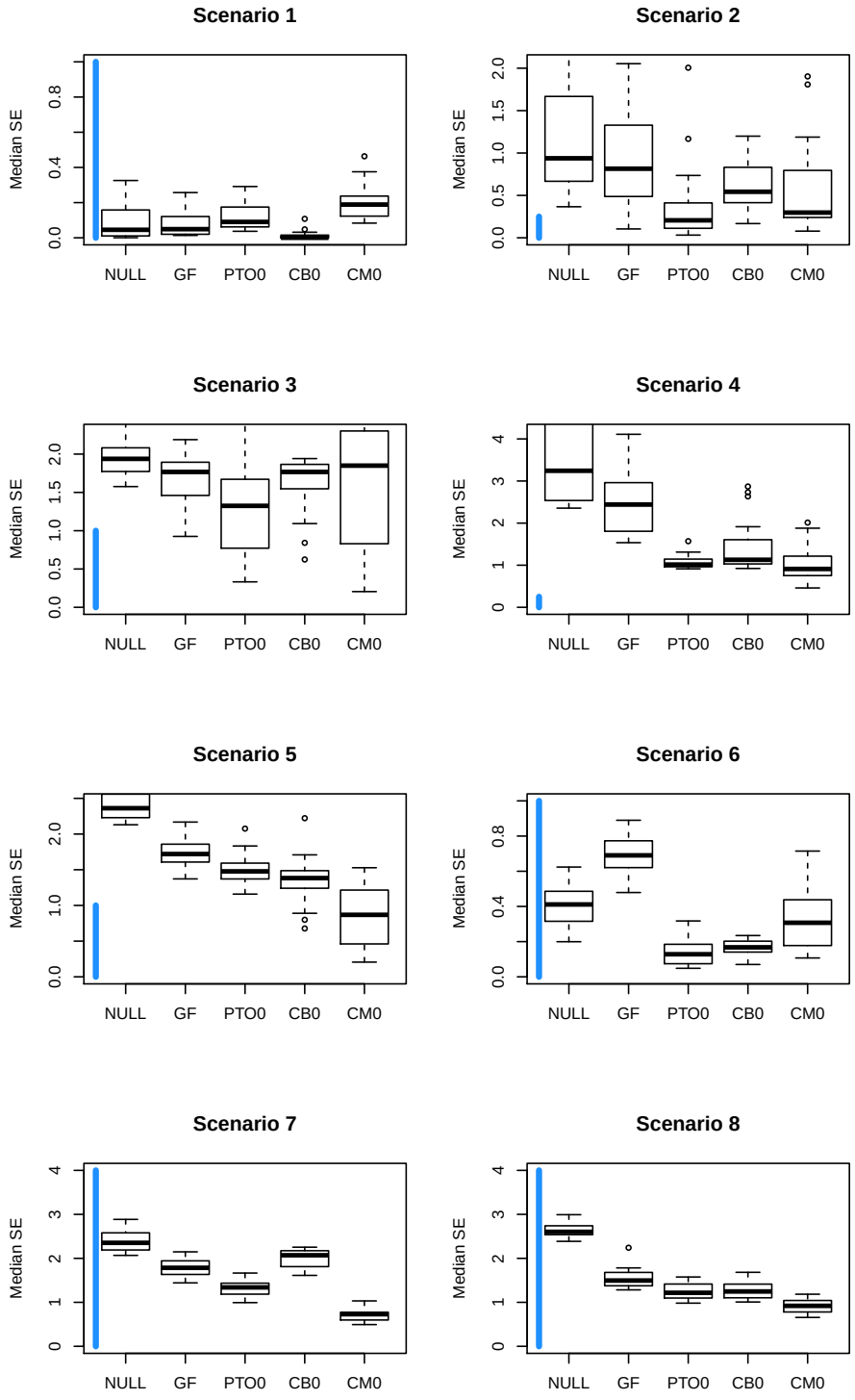
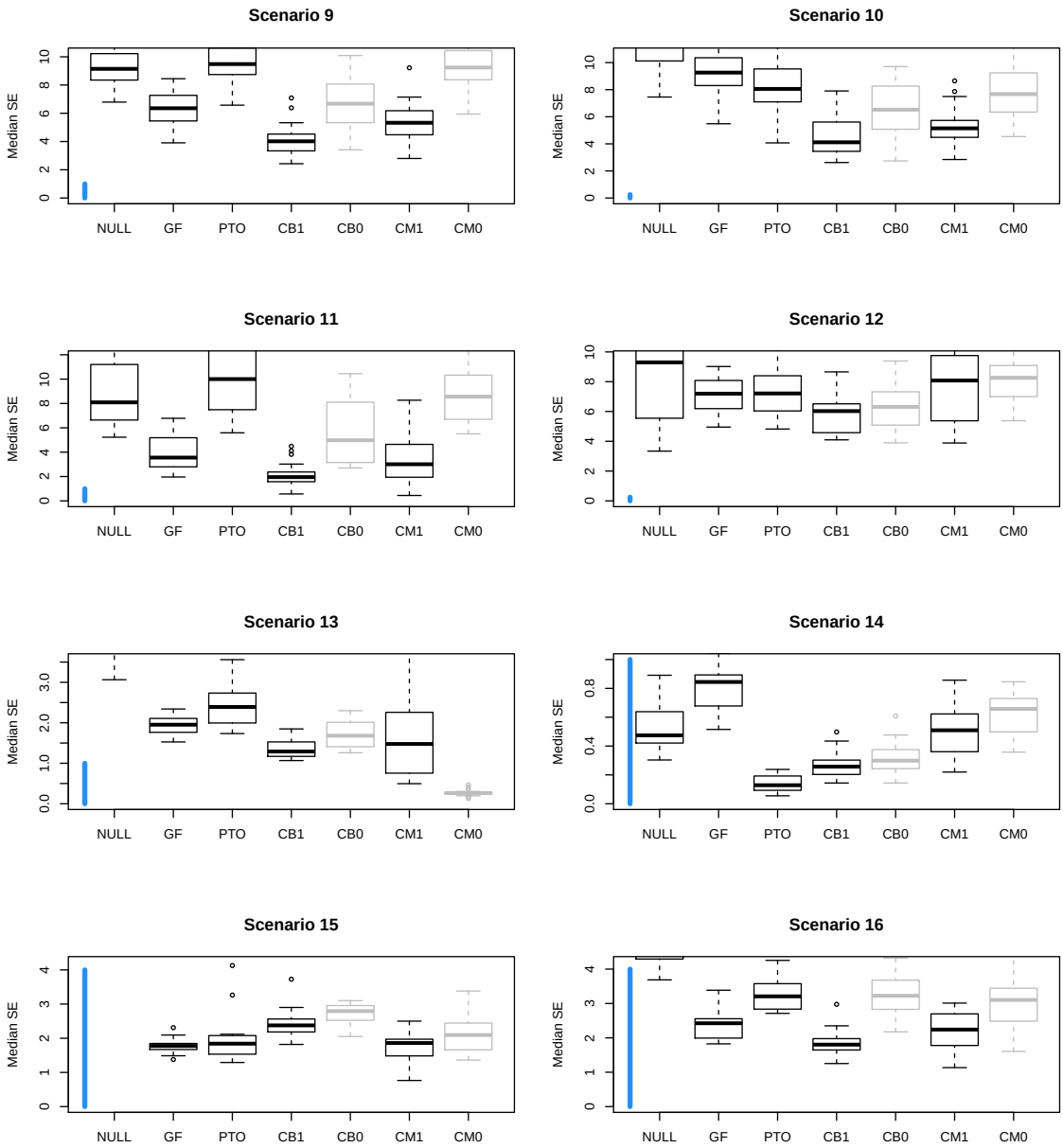


Figure 4.7: Results across eight simulated observational studies, in which treatment is more likely to be assigned to those with a greater mean effect. The seven estimators being evaluated are: NULL = the null prediction, GF = gradient forest, PTO = pollinated transformed outcome forest, CB1 = causal boosting (propensity adjusted), CB0 = causal boosting, CM1 = causal MARS (propensity adjusted), CM0 = causal MARS. CB0 and CM0 are in gray because they would not be used in this setting. They are provided for reference to assess the effect of the propensity adjustment. The vertical blue bar shows the standard deviation of the response, for assessing the practical significance of the difference between the methods' performances.



primary outcome of interest was whether the patient experienced any of the following events: myocardial infarction (heart attack), other acute coronary syndrome, stroke, heart failure or death from cardiovascular causes. The trial, which enrolled 9361 patients, ended after a median follow-up period of 3.26 years, when researchers determined at a pre-planned checkpoint that the population-average outcome for the intensive treatment group (1.65% incidence per year) was significantly better than that of the standard treatment group (2.19% incidence per year).

In addition to the primary event, for each patient researchers tracked several other adverse events, as well as 20 baseline covariates recorded at the moment of treatment assignment randomization: 3 demographic variables, 6 medical history variables and 11 lab measurements. The question that we seek to answer in this section is whether we can use these variables to give personalized estimates of treatment effect which are more informative than the population-level average treatment effect. To answer this question, we apply the gradient forest and causal MARS to these data.

Of the 9361 patients who underwent randomization, 1172 (12.5%) died, discontinued intervention, withdrew consent or were lost to follow-up before the conclusion of the trial. There is little evidence (χ^2 p-value = 31%) that this censorship was more common in either arm of the trial. To extract a binary outcome from these survival data, we use as our response the indicator that a patient experiences the primary outcome within 1000 days of beginning treatment, ignoring patients who were censored before 1000 days. Additionally, we dropped the 1.8% of patients who have at least one lab measure missing. This leaves us with a sample of 7344 patients, which we split into equally sized training and validation sets.

The results of fitting causal boosting and causal MARS on the training sample of 3672 patients are shown in Figure 4.8. We observe that the two methods yield very different distributions of estimated personalized treatment effects in the aggregate. Causal boosting produces estimates resembling a normal distribution with a standard deviation of about 3.5% risk. In contrast, causal MARS estimates almost all patients to have a treatment effect between -5% risk and $+0\%$ risk, but for a small percentage of patients the treatment effect is much greater or much lesser. The tails of this distribution are much heavier than that of a normal distribution. In fact, a very small number of patients (0.4% of the training sample) are not included in this figure because their treatment effect estimate from causal MARS falls outside of the plotted region.

Figure 4.9 depicts decision trees which summarize the key inferences made by causal boosting and causal MARS. Each leaf gives the average estimated treatment effect for patients who belong to that leaf. Such a decision could be reported to a physician to explain the basis for these personalized treatment effect estimates. According to causal boosting, for example, older patients with high triglycerides stand to gain more from the intensive blood pressure treatment than younger patients with high triglycerides. Among patients with low triglycerides and high glucose, those with low creatinine stand to benefit more from the intensive treatment than those with high creatinine.

Figure 4.8: Personalized treatment effect estimates from causal boosting and causal MARS. Each circle represents a patient, who gets a personalized estimate from each method. The dashed line represents the diagonal, along which the two estimates are the same.

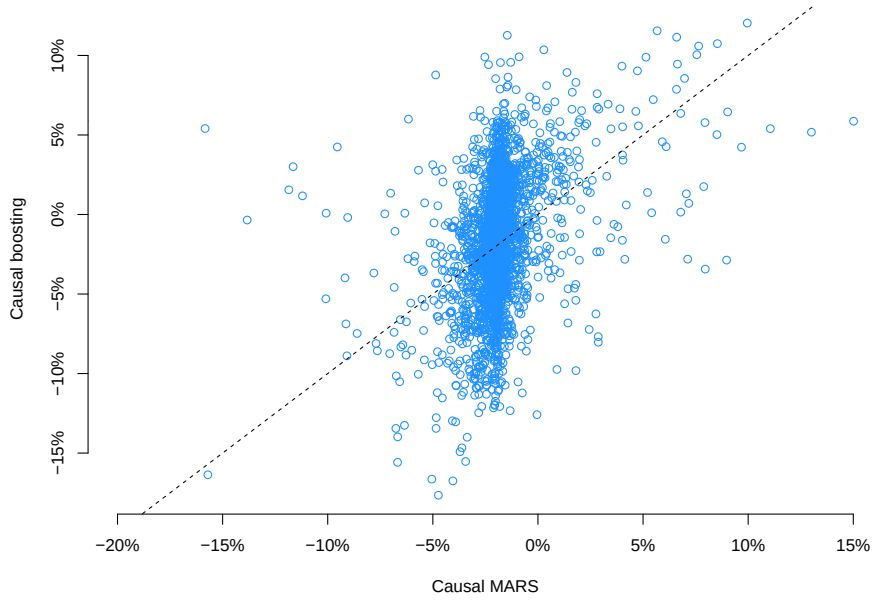


Figure 4.9: Decision trees summarizing with broad strokes the inferences of causal boosting and causal MARS. The variables are: `trr` triglcerides (mg/dL) from blood draw; `age` (years) age at beginning of trial; `glur` glucose (mg/dL) from blood draw; `screat` creatinine (mg/dL) from blood draw; `umalcr` albumin/creatinine ratio from urine sample; `dbp` diastolic blood pressure (mm Hg); `egfr` estimated glomerular filtration rate (mL/min/1.73m^2). If the inequality at a split is true for a patient, then that patient belongs to the left daughter node.

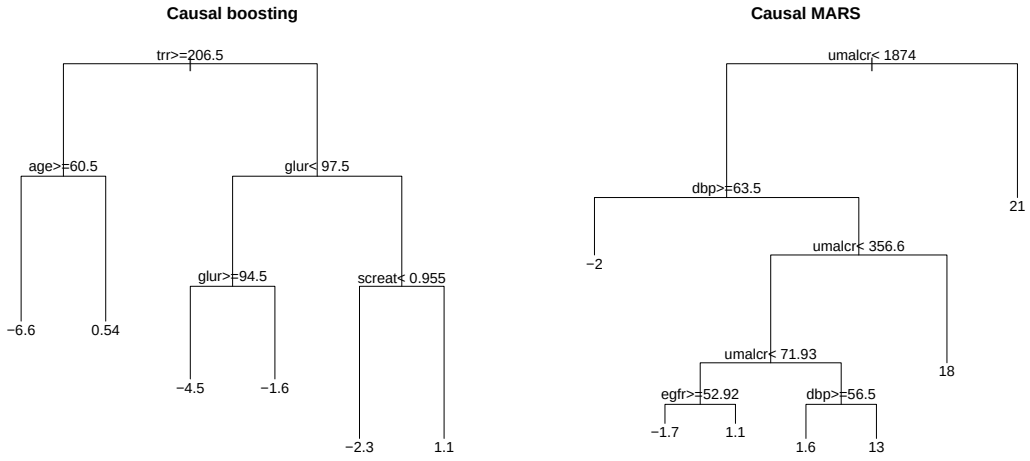
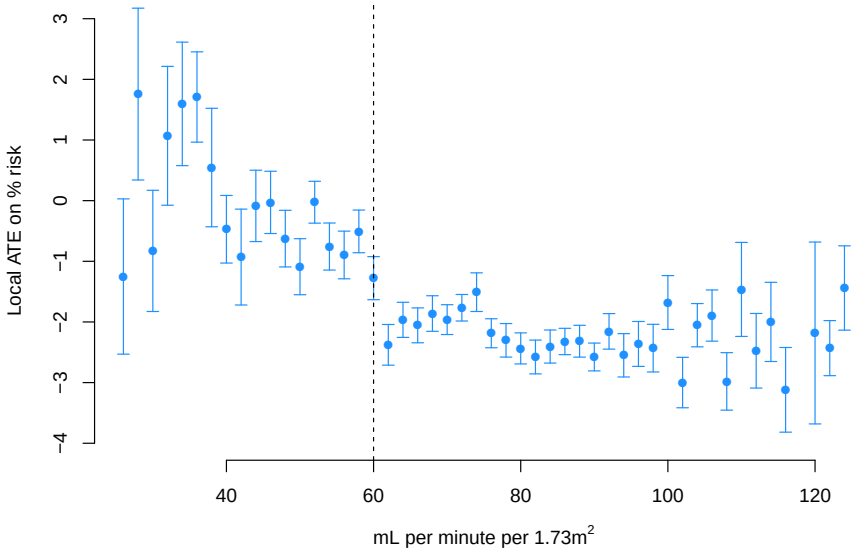


Figure 4.10: Training set personalized treatment effects, estimated via causal boosting, versus estimated glomerular filtration rate (eGFR). Patients are stratified according to eGFR on the x-axis, and each point gives the average personalized treatment effect among patients in that stratum. Error bars correspond to one standard error for the mean personalized treatment effect. The vertical dashed line represents a medical cutoff, below which patients are considered to suffer from chronic kidney disease.

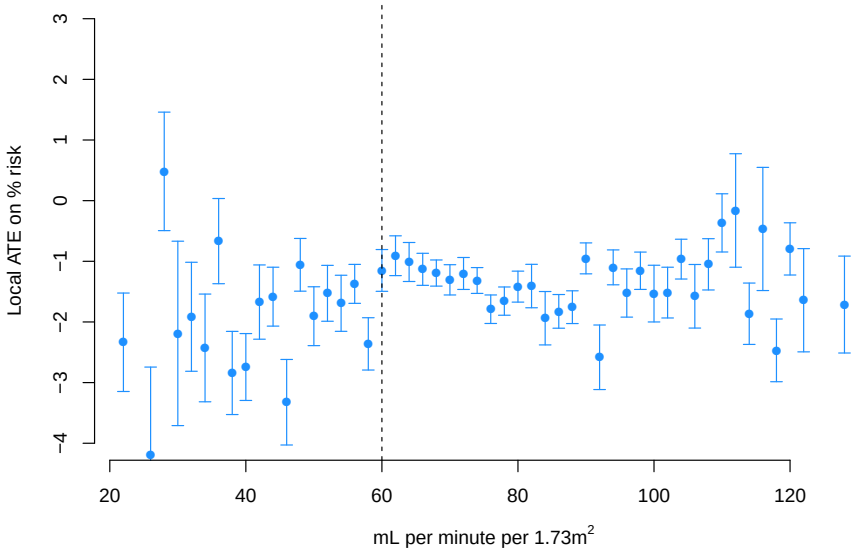


The decision tree for causal MARS makes the extreme claim that for patients with urine albumin/creatinine ratio above 1874, the average treatment effect is a 21% increase in risk. Discussions with practitioners suggest that the distribution of personalized treatment effects estimated by causal boosting is more plausible than that of causal MARS. As such, we focus our interpretation on the results of causal boosting for the reminder of this section.

To simplify the results even more than the decision tree does, we note that for both causal boosting and causal MARS, the two features which correlate most to the personalized treatment effect estimates are estimated glomerular filtration rate (eGFR) and creatinine. These two variables are highly correlated with each other, as creatinine is one of the variables used to estimate GFR. Both are used to assess kidney health, and patients with eGFR below 60 are considered to have chronic kidney disease. Figure 4.10 shows the relationship between eGFR and the estimated personalized treatment effect from causal boosting. Despite there being no manual notation in the data that there is something special about an eGFR of 60, we have learned from causal boosting that patients below this cutoff have less to gain from the intensive blood pressure treatment than patients above this cutoff.

Note that we are not only interested in whether a patient's personalized treatment effect is positive or negative. Intensive control of blood pressure comes with side effects and should only

Figure 4.11: Validation set personalized treatment effects, estimated via causal boosting, versus estimated glomerular filtration rate (eGFR). Patients are stratified according to eGFR on the x-axis, and each point gives the average personalized treatment effect among patients in that stratum. Error bars correspond to one standard error for the mean personalized treatment effect. The vertical dashed line represents a medical cutoff, below which patients are considered to suffer from chronic kidney disease.



be assigned to patients for whom the benefit of reducing the risk of an adverse coronary event is substantial. The results of causal boosting on the training set would suggest that patients with chronic kidney disease have less to gain from this treatment than do other patients.

4.8.1 Validation

The results above tell an interesting story: If you are a patient with chronic kidney disease (eGFR < 60), you are expected to benefit less from intensive blood pressure control. As discussed in Section 4.5.1, validating treatment effect estimates is challenging because we do not observe the treatment effect for any individual patient. In this section, we make an attempt to validate the more general conclusion from the previous section: that the treatment has less benefit for patients with chronic kidney disease.

Figure 4.11 shows the results of fitting causal boosting on the held-out validation set of 3672 patients. We see that the relationship between eGFR and estimated treatment effect does not tell the same story as in the training set. In fact, there is no clear relationship between these two variables in the validation set.

It is possible that we have insufficient power in the validation set to identify the relationship between eGFR and treatment effect and that with a larger sample of patients, we would have

validated our conclusions from the training set. It is worth noting that the team from Boston University which placed second in the SPRINT Data Analysis Challenge made the same finding as shown in the causal boosting results. They found that intensive blood pressure management does not improve primary outcomes for patients with chronic kidney disease. Something that the authors do not address is why they chose to analyze patients with chronic kidney disease. Presumably they used some combination of prior medical knowledge and manual hypothesis selection. In our training set, we came to the same conclusion using causal boosting without the benefit of either of these steps. The dissimilar results on the validation set could be explained by insufficient power.

4.9 Discussion

We have proposed and compared a number of different methods for estimating heterogeneous treatment effects from high-dimensional covariates. The causal boosting and causal MARS approaches seem particularly promising. More work is needed in refining and testing these methods, and in the construction of reliable confidence intervals for the estimated effects.

Chapter 5

Discussion

The reason I got into statistics is because there was something that felt magical about reaching your hand into a pool of numbers and pulling out a secret about how the world works or how the world will work. As I've learned about techniques for data analysis and developed my own, that magical feeling has worn off. Probability distributions are models for describing what we don't know about the world. We can make assumptions, wagers or educated guesses about these probability distributions, changing our conclusions accordingly. Sometimes these wagers will pay off; other times they will not. The last word belongs to validation: How accurate are the predictions, compared with the truths we observe?

This manuscript describes three settings in which we can leverage our understanding of the world to make wagers that are likely to pay off. In Chapter 2, we wager that the likelihood of a multinomial outcome A tells us something about how likely another outcome B is, relative to the other possible outcomes. We show in simulations that our method for leveraging this assumption works well when the assumption is correct and not so badly when it is false. On real data we show that this can lead to better predictions for the results of matchups in baseball and better identification of spoken vowels from audio data.

In Chapter 3, we wager that there are different types of gastric cancer patients, and the features used to identify cancer should be different between the different types. This wager pays off when applied to the data from Dr. Tibshirani's collaborators, yielding a test error roughly half that of standard training. We apply standard and customized training to a large battery of real-life datasets and show that customized training can sometimes lead to large improvements in test error. Because we use cross validation to choose the number of clusters, we can usually identify situations in which standard training would be better.

In Chapter 4, our implicit wager is that similar patients have similar personalized treatment effects. The challenge is learning what "similar" means in high-dimensional data when the treatment effect is not actually observed for any individual. Our three methods — causal boosting,

causal MARS and propensity transformed outcome forests — all use different estimators for average treatment effects across subsets of patients, and each searches for splits along variables to define patient similarity. Simulations show that we have taken steps toward addressing this problem, and application to real data shows that there is still work to be done.

Appendix A

Appendix

A.1 Identifiability of multinomial logistic regression model

We observe in Section 2.2 that the model (2.1) is not identifiable: For any $a \in \mathbb{R}$ and $\mathbf{c} \in \mathbb{R}^p$,

$$\frac{e^{\alpha_k - a + \mathbf{x}_i^T(\beta_k - \mathbf{c})}}{\sum_{\ell=1}^K e^{\alpha_\ell - a + \mathbf{x}_i^T(\beta_\ell - \mathbf{c})}} = \frac{e^{-a - x_i^T \mathbf{c}} e^{\alpha_k + \mathbf{x}_i^T \beta_k}}{e^{-a - x_i^T \mathbf{c}} \sum_{\ell=1}^K e^{\alpha_\ell + \mathbf{x}_i^T \beta_\ell}} = \frac{e^{\alpha_k + \mathbf{x}_i^T \beta_k}}{\sum_{\ell=1}^K e^{\alpha_\ell + \mathbf{x}_i^T \beta_\ell}}$$

Hence $(\alpha, \mathbf{B})$ and $(\alpha - a1_K, \mathbf{B} - \mathbf{c}1_K^T)$ have the same likelihood. The ridge penalty in (2.9) provides a natural resolution. Any solution to this problem must satisfy

$$||\mathbf{B}||_F^2 = \min_{\mathbf{c} \in \mathbb{R}^p} ||\mathbf{B} - \mathbf{c}1_K^T||_F^2 \tag{A.1}$$

because otherwise $\mathbf{B}$ can be replaced by $\mathbf{B} - \mathbf{c}1_K^T$ with a smaller norm but the same likelihood and hence a lesser objective. Note that the optimization problem on the right-hand side of (A.1) is separable in the entries of $\mathbf{c}$ and has the unique solution $\mathbf{c}^* = \frac{1}{K}\mathbf{B}1_K$, meaning that the rows of $\mathbf{B}$ in the solution must have mean zero. The unpenalized intercept α stil lacks identifiability, but we may take it to have mean zero as well.

Similarly, the NPMR solution must satisfy

$$||\mathbf{B}||_* = \min_{\mathbf{c} \in \mathbb{R}^p} ||\mathbf{B} - \mathbf{c}1_K^T||_*. \tag{A.2}$$

Whether this optimization problem always (for any $\mathbf{B} \in \mathbb{R}^{p \times K}$) has a unique solution is an open question. We speculate that it does and that the unique solution is $\mathbf{c}^* = \frac{1}{K}\mathbf{B}1_K$. As evidence, each fit of NPMR in the present manuscript has a solution with zero-mean rows. As further evidence, we have used the MATLAB software CVX [Grant et al., 2008] to solve (A.2) for several randomly generated matrices $\mathbf{B}$, and each time the solution has been $\mathbf{c}^* = \frac{1}{K}\mathbf{B}1_K$.

Note that $\mathbf{c}^* = \frac{1}{K}\mathbf{B}1_K$ must always be *a* solution to (A.2). To see this, note that

$$\mathbf{B} - \mathbf{c}^* 1_K^T = \mathbf{B} - \frac{1}{K}\mathbf{B}1_K 1_K^T = \mathbf{B}\left(\mathbf{I} - \frac{1}{K}1_K 1_K^T\right) = \mathbf{B}(\mathbf{I} - \mathbf{H}),$$

where $\mathbf{H} = 1_K(1_K^T 1_K)^{-1}1_K^T$ is a projection matrix. Hence $\mathbf{I} - \mathbf{H}$ is also a projection matrix and has spectral norm (maximum singular value) $||\mathbf{I} - \mathbf{H}||_\infty = 1$. By Hölder's inequality for Schatten p-norms [Bhatia, 1997],

$$||\mathbf{B}(\mathbf{I} - \mathbf{H})||_* \leq ||\mathbf{B}||_* ||\mathbf{I} - \mathbf{H}||_\infty = ||\mathbf{B}||_*,$$

so for any $\mathbf{B} \in \mathbb{R}^{p\times K}$,

$$||\mathbf{B} - \frac{1}{K}\mathbf{B}1_K 1_K^T||_* \leq ||\mathbf{B}||_*.$$

In order words, the nuclear norm can always be decreased, or at least not increased, by centering the rows to have mean zero.

The problem with a lack of identifiability in the multimonial regression model comes in the interpretation of the regression coefficients. When comparing coefficients across variables for the same outcome class, it is concerning that an arbitrary increase in either coefficient can corresond to the same fitted probabilities (if that same increase applys to all other coefficients for the same variable). This does not apply to any of the interpretation in Section 2.5.3, but in the absence of certainty that there is a unique solution to (A.2), we take the NPMR solution to be the one for which the mean of α and the row means of $\mathbf{B}$ are zero.

A.2 Proof of Lipschitz condition for multinomial log likelihood

We prove that the multinomial log-likelihood $\ell(\alpha, \mathbf{B}; \mathbf{X}, Y)$ from (2.4) has Lipschitz gradient with constant $L = \sqrt{K}||\mathbf{X}||_F^2$. Assume (without loss of generality) that the covariate matrix $\mathbf{X}$ has a column of 1s encoding the intercept, so $\alpha = 0$. The gradient of $\ell(\mathbf{B}; \mathbf{X}, Y)$ with respect to $\mathbf{B}$ is given by $\mathbf{X}^T(\mathbf{Y} - \mathbf{P})$, where $\mathbf{Y}$ and $\mathbf{P}$ are defined as in (2.6). What we must show is that, for any $\mathbf{B}, \mathbf{B}' \in \mathbb{R}^{p\times K}$:

$$||\mathbf{X}^T(\mathbf{Y} - \mathbf{P}) - \mathbf{X}^T(\mathbf{Y} - \mathbf{P}')||_F \leq \sqrt{K}||\mathbf{X}||_F^2 ||\mathbf{B} - \mathbf{B}'||_F. \tag{A.3}$$

Recall that $\mathbf{P}$ is a function of $\mathbf{B}$, so $\mathbf{P}'$ corresponds to $\mathbf{B}'$.

Consider a single entry $\mathbf{P}_{ik}$ of $\mathbf{P}$. Note that the gradient of $\mathbf{P}_{ik}$ with respect to $\mathbf{B}$ is given by $x_i w_{ik}^T$, where $w_{ik} \in \mathbb{R}^p$ and

$$(w_{ik})_j = \begin{cases} -\mathbf{P}_{ik}\mathbf{P}_{ij} & j \neq k \\ \mathbf{P}_{ik}(1 - \mathbf{P}_{ik}) & j = k \end{cases}.$$

For any $\mathbf{P} \in (0,1)^{n \times K}$,

$$||w_{ik}||_2 \leq ||w_{ik}||_1 = \mathbf{P}_{ik}(1 - \mathbf{P}_{ik}) + \mathbf{P}_{ik} \sum_{j \neq k} \mathbf{P}_{jk} = 2\mathbf{P}_{ik}(1 - \mathbf{P}_{ik}) \leq \frac{1}{2}.$$

This implies that the norm of the gradient of $\mathbf{P}_{ik}$ is bounded above by the inequality $||x_i w_{ik}^T||_F \leq ||x_i||_2 ||w_{ik}^T||_F \leq ||x_i||_2$. So for any $\mathbf{B}, \mathbf{B}' \in \mathbb{R}^{p \times K}$:

$$|\mathbf{P}_{ik} - \mathbf{P}'_{ik}| \leq ||x_i||_2 ||\mathbf{B} - \mathbf{B}'||_F. \tag{A.4}$$

Now we are ready to prove (A.3).

$$\begin{aligned}
||\mathbf{X}^T(\mathbf{Y} - \mathbf{P}) - \mathbf{X}^T(\mathbf{Y} - \mathbf{P}')||_F &= ||\mathbf{X}^T(\mathbf{P} - \mathbf{P}')||_F \\
&\leq ||\mathbf{X}||_F ||\mathbf{P} - \mathbf{P}'||_F \\
&= ||\mathbf{X}||_F \sqrt{\sum_{i=1}^{n} \sum_{k=1}^{K} (\mathbf{P}_{ik} - \mathbf{P}'_{ik})^2} \\
&\leq ||\mathbf{X}||_F \sqrt{\sum_{i=1}^{n} \sum_{k=1}^{K} ||x_i||_2^2 ||\mathbf{B} - \mathbf{B}'||_F^2} \qquad \text{from (A.4)} \\
&= ||\mathbf{X}||_F \sqrt{K ||\mathbf{B} - \mathbf{B}'||_F^2 \sum_{i=1}^{n} ||x_i||_2^2} \\
&= ||\mathbf{X}||_F \sqrt{K ||\mathbf{B} - \mathbf{B}'||_F^2 ||\mathbf{X}||_F^2} \\
&= \sqrt{K} ||\mathbf{X}||_F^2 ||\mathbf{B} - \mathbf{B}'||_F \qquad \blacksquare
\end{aligned}$$

A.3 Propensity score methods

In this appendix we outline the already-established techniques for using propensity score to adjust for bias in treatment assignment for observational studies in which the goal is to estimate a population-average treatment effect (ATE). Define $f(x)$ the marginal feature density, $f_1(x)$ the conditional density of X given $T = 1$ (and likewise $f_0(x)$), where T is binary treatment indicator, and let $\pi_1 = \mathbb{P}(T = 1)$ be the marginal proportion of treated. Let $\mu_1(X) = \mathbb{E}[Y|T = 1, X]$, and likewise $\mu_0(X)$, and $\tau(X) = \mu_1(X) - \mu_0(X)$. Finally, let $\pi(X) = \mathbb{P}(T = 1|X)$ be the treatment propensity.

Transformed outcome averaging

Note that the *transformed outcome*

$$Z \equiv T\frac{Y}{\pi(X)} + (1-T)\frac{-Y}{1-\pi(X)}$$

satisfies

$$\begin{aligned}\mathbb{E}[Z|X] &= \mathbb{P}(T=1|X)\frac{1}{\pi(x)}\mathbb{E}[Y|T=1,X] - \mathbb{P}(T=0|X)\frac{1}{1-\pi(x)}\mathbb{E}[Y|T=1,X] \\ &= \mathbb{E}[Y|T=1,X] - \mathbb{E}[Y|T=0,X] = \mu_1(X) - \mu_0(X) = \tau(X).\end{aligned}$$

Hence if the expectation of Z is evaluated with respect to the distribution of X,

$$\mathbb{E}_X[Z] = \mathbb{E}_X[\mathbb{E}[Z|X]] = \mathbb{E}_X[\tau(X)].$$

In other words, the transformed outcome is unbiased for the ATE. So a natural estimator for the ATE in a sample of patients would be the sample mean of the transformed outcome. This justifies for example using Z as a response to grow a random forest in our pollinated transformed outcome forest.

Propensity score stratification

Note that it is not necessarily the case that $E[Y|T=1] = E[\mu_1(X)|T=1]$ and $E_X[\mu_1(X)]$ are the same; it is possible that conditioning on T changes the distribution of X and consequently the distribution of $\mu_1(X)$. This is the essence of why we cannot ignore non-randomized treatment assignment in observational studies. However, it is the case that

$$\mathbb{E}[Y|T=1,\pi(X)] = \mathbb{E}[\mu_1(X)|\pi(X)].$$

To see this, note that $X \perp\!\!\!\perp T|\pi(X)$ because by assumption $T \sim \text{Binomial}(1,\pi(X))$. Hence the conditional distribution of X given $\pi(X)$ and T is the same as the conditional distribution of X given $\pi(X)$. This implies that

$$\mathbb{E}[Y|T=1,\pi(X)] = \mathbb{E}\left\{\mathbb{E}[Y|T=1,X] \;\middle|\; T=1,\pi(X)\right\} = \mathbb{E}[\mu_1(X)|\pi(X)].$$

What this says is that for fixed $\pi(X)$, the mean response under treatment is unbiased for the conditional expectation of $\mu_1(X)$. This equality holds for any value of X, so the expectations of these two quantities are the same with respect to the distribution of $\pi(X)$:

$$\mathbb{E}_{\pi(X)}\left[\mathbb{E}[Y|T=1,\pi(X)]\right] = \mathbb{E}_{\pi(X)}\left[\mathbb{E}[\mu_1(X)|\pi(X)]\right] = \mathbb{E}_X[\mu_1(X)].$$

This leads to the following estimator for $\mathbb{E}_X[\mu_1(X)]$: Compute the average response for all treated patients for each value of the propensity, and integrate with respect to the distribution of the propensity. In practice, we approximate this by using a rough approximation to the distribution of $\pi(X)$: Define strata (or bins) of the propensity score, for example (0, 0.1], ..., (0.9, 1). Within each stratum, find the average response among treated patients. Then combine these values in a weighted average, weighting according to the frequency of each stratum. This is our estimate of $\mathbb{E}_X[\mu_1(X)]$. We follow the same procedure in the control arm to estimate $\mathbb{E}_X[\mu_0(X)]$, and the difference is our estimate of $\mathbb{E}_X[\tau(X)]$.

Inverse probability weighting

From Bayes' theorem, $f_1(x) = f(x)\pi(x)/\pi_1$. Consider weighting this density with weights proportional to $1/\pi(x)$. The density of this weighted distribution is given by

$$\tilde{f}_1(x) = \frac{\frac{1}{\pi(x)} f(x)\pi(x)/\pi_1}{\int_{\mathbb{R}} \frac{1}{\pi(x)} f(x)\pi(x)/\pi_1} dx = \frac{f(x)/\pi_1}{\int_{\mathbb{R}} f(x)/\pi_1 dx} = \frac{f(x)/\pi_1}{1/\pi_1} = f(x).$$

Hence the weighted conditional distribution of X given $T = 1$ is the same as the marginal distribution of X. So the expectation of any function of X with respect to this distribution is the same as with respect to the marginal distribution of X. Specifically, using $\tilde{X}$ to denote the random variable following the weighted density $\tilde{f}_1(x)$,

$$\mathbb{E}_{\tilde{X}}[\mu_1(\tilde{X})] = \mathbb{E}_X[\mu_1(X)].$$

Based on this result, we use the sample mean of the response in the treatment arm, with weights proportional to the inverse of the propensity, as an unbiased estimator for $\mathbb{E}_X[\mu_1(X)]$. Similarly, in the control arm we use weights proportional to $1/(1 - \pi(x))$ to get an unbiased estimate for $\mathbb{E}_X[\mu_0(x)]$. The difference between these two is our estimate for $\mathbb{E}_X[\tau(X)]$

Bibliography

John A Anderson. Regression and ordered categorical variables. *Journal of the Royal Statistical Society B*, 46(1):1–30, 1984.

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

Susan Athey, Jule Tibshirani, and Stefan Wager. Solving heterogeneous estimating equations with gradient forests. 2017.

Peter C Austin. An introduction to propensity score methods for reducing the effects of confounding in observational studies. *Multivariate Behavioral Research*, 46:399–424, 2011.

K Bache and M Lichman. UCI Machine Learning Repository. UC Irvine School of Information and Computer Science, 2013.

Benjamin Baumer and Andrew Zimbalist. *The Sabermetric Revolution*. University of Pennsylvania Press, Philadelphia, 2014.

Rajendra Bhatia. *Matix Analysis*. Springer, New York, 1997.

Jacob Bien and Robert J Tibshirani. Hierarchical clustering with prototypes via minimax linkage. *Journal of the American Statistical Association*, 106(495):1075–1084, 2011.

Marco Bonetti and Richard D Gelber. Patterns of treatment effects in subsets of patients in clinical trials. *Biostatistics*, 5(3):465–481, 2004.

Léon Bottou and Vladimir Vapnik. Local learning algorithms. *Neural Computation*, 4(6):888–900, 1992.

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

Kun Chen, Hongbo Dong, and Kung-Sik Chan. Reduced rank regression via adaptive nuclear norm penalization. *Biometrika*, 100(4):901–920, 2013.

Wei Chen, Debashis Ghosh, Trivellore E Raghunathan, Maxim Norkin, Daniel J Sargent, and Gerold Bepler. On Bayesian methods of exploring qualitative interactions for targeted treatment. *Statistics in Medicine*, 31(28):3693–3707, 2012.

Hugh A Chipman, Edward I George, and Robert E McCulloch. Bayesian CART model search. *Journal of the American Statistical Assocation*, 93(443):935–948, 1998.

Corinna Cortes and Mehryar Mohri. On transductive regression. In *Advances in Neural Information Processing Systems*, volume 19, 2007.

Richard K Crump, V Joseph Hotz, Guido W Imbens, and Oscar A Mitnik. Nonparametric tests for treatment effect heterogeneity. *The Review of Economics and Statistics*, 90(3):389–405, 2008.

Livia S Eberlin, Robert J Tibshirani, J Zhang, T A Longacre, G J Berry, D B Bingham, J A Norton, Richard N Zare, and G A Poultsides. Molecular assessment of surgical-resection margins of gastric cancer by mass-spectrometric imaging. *Proceedings of the National Academy of Sciences*, 111(7): 2436–2441, 2014.

Jerome Friedman. Multivariate adaptive regression splines. *The Annals of Statistics*, 19(1):1–67, 1991.

Jerome Friedman. Greedy function approximation: A gradient boosting machine. *The Annals of Statistics*, 29(5):1189–1232, 2001.

Jerome Friedman, Trevor J Hastie, and Robert J Tibshirani. Regularization paths for generalized linear models via coordinate descent. *Journal of Statistical Software*, 33(1):1–22, 2010.

Zhouyu Fu, Antonio Robles-Kelly, and Jun Zhou. Mixing linear SVMs for nonlinear classification. *IEEE Transactions on Neural Networks*, 21(12):1963–1975, 2010.

M Gail and Richard Simon. Testing for qualitative interactions between treatment effects and patient subsets. *Biometrics*, 41(2):361–372, 1985.

David Gil, Jose Luis Girela, Joaquin De Juan, M Jose Gomez-Torres, and Magnus Johnsson. Predicting seminal quality with artificial intelligence methods. *Expert Systems With Applications*, 39 (16):12564–12573, 2012.

Michael Grant, Stephen Boyd, and Yinyu Ye. *CVX: A Language and Environment fo Statistical Computing*. CVX Research, 2008. URL http://www.cvxr.com/.

Donald P Green and Holger L Kern. Modeling heterogeneous treatment effects in survey experiments with Bayesian additive regression trees. *Public Opinion Quarterly*, 76(3):491–511, 2012.

Sander Greenland. Alternative models for ordinal logistic regression. *Statistics in Medicine*, 13(16): 1665–1677, 1994.

Quanquan Gu and Jiawei Han. Clustered support vector machines. In *Proceedings of the 16th International Conference on Artificial Intelligence and Statistics*, pages 307–315, 2013.

Paul Gustafson. Bayesian regression modeling with interactions and smooth effects. *Journal of the American Statistical Association*, 95(451):795–806, 2000.

Margaret A. Hamburg and Francis S. Collins. The path to personalized medicine. *The New England Journal of Medicine*, 363(4):301–304, 2010.

Trevor J Hastie, Robert Tibshirani, and Jerome Friedman. *The Elements of Statistical Learning: Data mining, inference and prediction*. Springer Series in Statistics. Springer, 2nd edition, 2009.

Trevor J Hastie, Robert J Tibshirani, and Martin Wainwright. *Statistical Learning with Sparsity: The Lasso and its Generalizations*. Monographs on Statistics and Applied Probability. CRC Press, 1st edition, 2015.

Jennifer L Hill. Bayesian nonparametric modelling for causal inference. *Journal of Computational and Graphical Statistics*, 20(1):217–240, 2011.

Kosuke Imai and Marc Ratkovic. Estimating treatment effect heterogeneity in randomized program evaluation. *The Annals of Applied Statistics*, 7(1):443–470, 2013.

Michael I Jordan and Robert A Jacobs. Hierarchical mixtures of experts and the EM algorithm. *Neural computation*, 6(2):181–214, 1994.

Jonathan Judge and BP Stats Team. DRA: An in-depth discussion. http://www.baseballprospectus.com/article.php?articleid=26196, April 2015.

H Tolga Kahraman, Seref Sagiroglu, and Ilhami Colak. The development of intuitive knowledge classifier and the modeling of domain dependent data. *Knowledge-Based Systems*, 37:283–295, 2013.

Lubor Ladicky and Phillip Torr. Locally linear support vector machines. In *Proceedings of the 28th International Conference on Machine Learning*, pages 985–992, 2011.

Michael LeBlanc. An adaptive expansion method for regression. *Statistica Sinica*, 5(2):737–748, 1995.

Rensis Likert. A technique for the measurement of attitudes. *Archives of Psychology*, 140(22):1–55, 1932.

Max A Little, Patrick E McSharry, Stephen J Roberts, Declan AE Costello, and Irene M Moroz. Exploiting nonlinear recurrence and fractal scaling properties for voice disorder detection. *BioMedical Engineering OnLine*, 6(1):23, 2007.

Yen Sia Low, Blanca Gallego, and Nigam Haresh Shah. Comparing high-dimensional confounder control methods for rapid cohort studies from electronic health records. *Journal of Comparative Effectiveness Research*, 5(2):179–192, 2016.

Tian Min Ma. *Local and personalised modelling for renal medical decision support system.* PhD thesis, Auckland University of Technology, 2012.

James D Malley, Jochen Kruppa, Abhijit Dasgupta, Karen G Malley, and Andreas Ziegler. Probability machines: consistent probability estimation using nonparametric learning machines. *Methods of Information in Medicine*, 51(1):74–81, 2012.

Kamel Mansouri, Tine Ringsted, Davide Ballabio, Roberto Todeschini, and Viviana Consonni. Quantitative structure–activity relationship models for ready biodegradability of chemicals. *Journal of Chemical Information and Modeling*, 53(4):867–878, 2013.

Yu Nesterov. Gradient methods for minimizing composite objective function. Technical Report 2007076, Université Catholique de Louvain, Center for Operations Research and Econometrics (CORE), 2007.

R Core Team. *R: A Language and Environment fo Statistical Computing.* R Foundation for Statistical Computing, Vienna, Austria, 2016. URL http://www.R-project.org/.

Anthony John Robinson. Dynamic error propagation networks. *PhD dissertation, University of Cambridge*, 1989.

Paul R Rosenbaum and Donald B Rubin. The central role of the propensity score in observational studies for causal effects. *Biometrika*, 70(1):41–55, 1983.

Donald B Rubin. Estimating causal effects of treatments in randomized and nonrandomized studies. *Journal of Educational Psychology*, 66(5):688–701, 1974.

Willi Sauerbrei, Patrick Royston, and Karina Zapien. Detecting an interaction between treatment and a continuous covariate: A comparison of two approaches. *Computational Statistics and Data Analysis*, 51(8):4054–4063, 2007.

Nigam Haresh Shah. Performing an informatics consult, May 2016. URL http://bigdata.stanford.edu/pastevents/2016-presentations.html. Big Data in Biomedicine Conference — Stanford Medicine.

Babak Shahbaba and Radford Neal. Nonlinear models using Dirichlet process mixtures. *Journal of Machine Learning Research*, 10:1829–1850, 2009.

Jerzy Splawa-Neyman, Dorota M Dabrowska, and Terence P Speed. On the application of probability theory to agricultural experiments. Essay on principles. Section 9. *Statistial Science*, 5(4):465–472, 1990.

SPRINT Research Group. A randomized trial of intensive versus standard blood-pressure control. *New England Journal of Medicine*, 373(22):2103–2116, 2015.

Xiaogang Su, Chih-Ling Tsai, Hansheng Wang, David M Nickerson, and Bogong Li. Subgroup analysis via recursive partitioning. *Journal of Machine Learning Research*, 10:141–158, 2009.

Matt Taddy, Matt Gardner, Liyun Chen, and David Draper. A nonparametric Bayesian analysis of heterogenous treatment effects in digital experimentation. *Journal of Business & Economic Statistics*, 34(4):661–672, 2016.

Lu Tian, Ash A Alizadeh, Andrew J Gentles, and Robert Tibshirani. A simple method for estimating interactions between a treatment and a large number of covariates. *Journal of the American Statistical Association*, 109(508):1517–1532, 2014.

Robert Tibshirani. Regression shrinkage and selection via the lasso. *Journal of the Royal Statistical Society B*, 58(1):267–288, 1996.

Luis Torgo and Joaquim Pinto DaCosta. Clustered partial linear regression. *Machine Learning*, 50 (3):303–319, 2003.

Athanasios Tsanas, Max A Little, Cynthia Fox, and Lorraine O Ramig. Objective automatic assessment of rehabilitative speech treatment in Parkinson's disease. *IEEE Transactions on Neural Systems and Rehabilitation Engineering*, 22(1):181–190, 2014.

Stefan Wager and Susan Athey. Estimation and inference of heterogeneous treatment effects using random forests. 2015.

Daniela M Witten and Robert J Tibshirani. A framework for feature selection in clustering. *Journal of the American Statistical Association*, 105(490):713–726, 2010.

World Health Organization. Cancer. WHO Fact Sheet No. 297, 2013. Available at http://www.who.int/mediacentre/factsheets/fs297/en/index.html.

Marvin N Wright and Andreas Ziegler. ranger: A fast implementation of random forests for high dimensional data in C++ and R. 2015.

Mingrui Wu and Bernhard Schölkopf. Transductive classification via local learning regularization. In *Proceedings of the 11th International Conference on Artificial Intelligence and Statistics*, pages 628–635, 2007.

Yu Xie, Jennie E Brand, and Ben Jann. Estimating heterogeneous treatment effects with observational data. *Sociological Methodology*, 42(1):314–347, 2012.

Thomas W Yee. The VGAM package for categorical data analysis. *Journal of Statistical Software*, 32(10):1–34, 2010.

Thomas W Yee and Trevor J Hastie. Reduced-rank vector generalized linear models. *Statistical Modelling*, 3(1):15–41, 2003.

Kai Yu, Tong Zhang, and Yihong Gong. Nonlinear learning using local coordinate coding. In *Advances in Neural Information Processing Systems*, volume 21, pages 2223–2231, 2009.

Yingqi Zhao, Donglin Zeng, A John Rush, and Michael R Kosorok. Estimating individualized treatment rules using outcome weighted learning. *Journal of the American Statistical Association*, 107(499):1106–1118, 2012.

Dengyong Zhou, Olivier Bousquet, Thomas Navin Lal, Jason Weston, and Bernhard Schölkopf. Learning with local and global consistency. In *Advances in Neural Information Processing System*, volume 16, pages 321–328, 2004.

Jun Zhu, Ning Chen, and Eric P Xing. Infinite SVM: a Dirichlet process mixture of large-margin kernel machines. In *Proceedings of the 18th International Conference on Machine Learning*, pages 617–624, 2011.

Xiaojin Zhu. Semi-supervised learning literature survey. Technical Report 1530, University of Wisconsin-Madison Department of Computer Science, December 2007.