STATISTICAL LEARNING WITH MISSING DATA

Thomas Gordon Stewart

A dissertation submitted to the faculty at the University of North Carolina at Chapel Hill in partial fulfillment of the requirements for the degree of Doctor of Philosophy in the Department of Biostatistics.

Chapel Hill
2015

Approved by:

Michael Wu

Donglin Zeng

Amy Herring

Yufeng Liu

Neil Hayes

ABSTRACT

Thomas Gordon Stewart: Statistical Learning with Missing Data
(Under the direction of Michael Wu and Donglin Zeng)

Statistical learning is a popular family of data analysis methods which has been successfully employed in biomedical research, the social sciences, public safety applications, and most data dependent areas of research. A major goal of statistical learning methods is to construct rules which predict an outcome y from a set of predictors $\mathbf{x}$, for example, predicting treatment response from a set of pre-treatment biomarkers. Accurate prediction rules of treatment response can guide health care providers to select the best treatment options. The support vector machine (SVM) is a statistical learning method profitably employed in a number of research areas such as biomedical computer vision tasks, drug design, and genetics. Because SVMs admit nonlinear prediction rules, it is a natural choice for analyzing data with potentially complex relationships. One drawback to SVMs is the limited means of handling missing data in the training set, yet missing data is ubiquitous in studies of health-related outcomes. In this research, we review the literature on missing data, and we summarize those scenarios when missing data may bias statistical analysis. We also provide an overview of supervised classification methods, especially those methods which accommodate missing data. We pay special attention to SVMs as this family of methods is the focus of our proposed contributions to this body of work. We propose three methods involving SVMs and missing data. The first paper proposes an EM-based solution for constructing SVMs when the training set includes observations with missing covariates. We present the method for continuous covariates but the method is applicable to discrete covariates as well. The second paper proposes weighting methods inspired by

weighted estimating equations, also for the purpose of constructing SVMs when the training set includes observations with missing covariates. The third paper considers scenarios in which class labels are missing or are partially observed, an area of study commonly called semi-supervised learning. We propose an EM-type solution for the semi-supervised learning scenario, and we apply the method to both two-class and multi-class SVMs. In each paper, the proposed methods will be demonstrated in the context of a large multi-center observational study of Hepatitis C patients.

To my parents.

ACKNOWLEDGEMENTS

I could not have completed this dissertation without the patient and kind help of Michael Wu and Donglin Zeng. I am indebted to them for their mentoring, encouragement, and support. To each, I offer sincere appreciation and gratitude. Thank you Michael and Donglin.

My time at UNC was enriched with many interesting projects with talented collaborators. Foremost of these projects is my work with Paul Stewart and HCV-TARGET. I consider my association with Paul and each member of the HCV-TARGET family as one of the great, unforeseen experiences of my graduate school career. Not only did HCV-TARGET support me financially, it gave me a chance to develop into a consulting statistician. Thank you to Paul Stewart, Monica Vainorius, Lucy Akushevich, Ken Bergquist, John Baron, and Michael Fried.

Veronica Stallings and Melissa Hobgood are the cheerful core of the Biostatistics Department. My time in the department has been improved by their laugh, encouragement, and administrative finesse. Thank you Veronica and Melissa.

So much of the good in my life is due to my goodly parents, Anne and Monte. I am incapable of unwinding and untangling all the ways in which their love and example have inspired me, encouraged me, and made me. An eternal thank you Mom and Dad.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

CHAPTER 1: LITERATURE REVIEW

1.1 Introduction

Missing data are ubiquitous. Despite continuing advances in data collection, missing data are likely to remain a permanent feature of statistical analysis. While many missing data methods exist for multivariate normal models [47], general likelihood models [89, 28], survey sampling models [90], and weighted estimating equation models [86], all the developments are either parametric or semi-parametric methods, so they are sensitive to model miss-specification and may not be applicable in high dimensional settings.

Statistical learning is a popular family of data analysis methods particularly suited for high dimensional settings. Statistical learning methods have been successfully employed in biomedical research, the social sciences, public safety applications, and most data dependent areas of research. The goal of statistical learning methods is to construct rules which predict an outcome y from a potentially large set of predictors $\mathbf{x}$, for example, predicting treatment response from a set of pre-treatment biomarkers. However, methods in statistical learning for missing data are, in many cases, ad-hoc. The scant attention to the topic in statistical learning texts, like Devroye et al. [29] and Hastie et al. [50], points to this issue and the need to close this gap in current statistical learning methodology. The collective examples of Ghahramani and Jordan [42], Nigam et al. [75], and Williams et al. [108] indicate that many of the principled approaches which grew from Rubin [89] and Dempster et al. [28] can also be extended to statistical learning. The goal of this literature review is to provide an overview of the approaches in statistical learning to handle missing data. We give

special attention to support vector machines (SVM) because it is the focus of methods for missing data proposed in the following chapters. The support vector machine is a statistical learning method introduced in Boser et al. [15], Cortes and Vapnik [25] and Vapnik [106]. The method has been successfully employed in both classification and regression tasks, and it is particularly useful in computer vision applications [77, 23]. It is a basis expansion method which provides the user with considerable modeling flexibility.

The literature review is organized into four sections. The first is a review of Rubin [89] and many of the parametric or semi-parametric methods that followed. We introduce likelihood with EM, imputation, weighted estimating equations, and Bayesian methods for missing data. The second section is an overview of binary supervised classification methods and their associated methods for missing data. The third section particularly focuses on SVM and describes its methods for missing data. Lastly, we describe a research plan which provides three principled missing data methods.

1.2 Parametric and semi-parametric methods for missing data

1.2.1 Missing data mechanism

Consider a training data set of n observations with outcome y_i and covariate vector $\mathbf{x}_i$ of dimension d for each patient. Depending on the scenario, outcomes or covariates may be missing. To indicate which variables are missing, let $\mathbf{z}_i = (y_i, \mathbf{x}_i)$ and define vector $\mathbf{r}_i$ to indicate if the corresponding data element in $\mathbf{z}_i$ is observed or not,

$$\mathbf{r}_i = (r_{i0}, r_{i1}, r_{i2}, \ldots, r_{id}) \qquad r_{ij} = \begin{cases} 1 & \text{if } z_{ij} \text{ is observed} \\ 0 & \text{if } z_{ij} \text{ is missing.} \end{cases}$$

Often data are partitioned into subvectors of missing and observed elements, for example $\mathbf{z}_i = (\mathbf{z}_i^m, \mathbf{z}_i^o)$. The processes which generates missing data are grouped into

three types. The simplest type of missing data mechanism is missing completely at random (MCAR). MCAR describes situations when the missing data mechanism is independent of the data. In terms of the missing data indicator, MCAR mechanisms are characterized as

$$P(\mathbf{r}_i \mid \mathbf{z}_i^m,\ \mathbf{z}_i^o) = P(\mathbf{r}_i).$$

In other words, the mechanisms which lead to missing data are unrelated to either outcome or predictors. The second type of missing data mechanism is missing at random (MAR). It occurs if, conditional on the observed data $\mathbf{z}^o$, the missing data mechanism is independent of the missing data $\mathbf{z}^m$. That is,

$$P(\mathbf{r}_i \mid \mathbf{z}_i^m,\ \mathbf{z}_i^o) = P(\mathbf{r}_i \mid \mathbf{z}_i^o).$$

Lastly, not missing at random (NMAR) occurs if the missing data model is a function of the unobserved missing value. The missing value is unobserved for reasons related to the value. In the notation of missing data models, NMAR is

$$P(\mathbf{r}_i \mid \mathbf{z}_i^m,\ \mathbf{z}_i^o) = P(\mathbf{r}_i \mid \mathbf{z}_i^m,\ \mathbf{z}_i^o),$$

or any distribution which depends on $\mathbf{z}_i^m$. The three types of missing data represent a hierarchy of modeling assumptions. MCAR is the strongest assumption while NMAR is the weakest.

Rubin [89] studied scenarios when the missing data model can be ignored in likelihood based analyses. Specifically, if (a) the missing mechanism is MAR or MCAR and (b) the missing data model parameters are distinct from the response model parameters, then maximum likelihood estimates based on the observed data likelihood which ignores the missing indicator model are valid. That is to say, data analysis can proceed without the cumbersome burden of modeling the missing data

mechanism under these two assumptions. In contrast, if the missing data are NMAR, likelihood and Bayesian methods must incorporate the unverifiable assumptions of a missing indicator model. In the sections that follow, we will discuss methods and identify them as being applicable to MAR data or MCAR data.

1.2.2 Complete case analysis and imputation methods

Perhaps the earliest and easiest method for handling missing data is to omit observations with missing values, a method known as complete case analysis. Complete case analysis is valid in MCAR situations, but is not valid with MAR data.

Imputation is a two-stage method and is popular because it works with a wide variety of models and estimation techniques. One of its earliest forms was developed and used extensively with the Current Population Survey by the US Census Bureau, despite its poorly developed theory [4]. In the context of large scale surveys, Rubin [90] introduced multiple imputation and provided conditions for the method's application to unbiased estimation, also see Schafer [93], Harel and Zhou [46] and Rubin [91]. There are two general imputation types: single and multiple. The single imputation procedure is to replace missing values with values drawn from an imputation model. The filled-in or complete data set is analyzed with the desired method. As noted in Rubin [90], single imputation standard error estimates are generally too small because uncertainty from the imputations is not incorporated into the standard error calculations.

Multiple imputation improves on single imputation by producing more accurate standard error estimates. The procedure is to: (a) generate several, say Q, filled-in data sets, (b) generate an estimate from each data set, and finally (c) combine the Q estimates by taking the average. The standard error of the estimate is calculated as $\sqrt{U + (1 + 1/Q)B}$ where U is mean standard error of the Q estimates and B is the imputation error or the variance of the Q estimates. If the imputed data sets all generate

very similar estimates, the imputation error leads to a small increase in standard error. Conversely, disparate estimates lead to a larger standard error.

Depending on the analysis method and its attendant assumptions, the imputation model can be chosen so that resulting estimates are unbiased in cases of MCAR or MAR data [90]. However, when missing values are drawn from a convenience distribution instead of the proper imputation distribution, the imputation model is improper. Estimates calculated from improper imputation can be reasonable, but there is no assurance that the estimates are unbiased.

1.2.3 Parametric and semi-parametric methods

Here we consider the methods for missing data in three broad families of parameter estimation: likelihood estimation, estimating equations, and Bayesian estimation. The methods described here can be generalized to a wide variety of classification and regression methods. In each method, we consider estimation of some population parameter $\boldsymbol{\beta}$.

Likelihood estimation and EM

In the context of likelihood estimation of a regression model, say of $y = \mathbf{x}^t\boldsymbol{\beta} + \epsilon$, with MAR missingness in the covariates, Rubin [89] noted that estimation only requires maximization of the observed data likelihood. Thus, if $P(y|\mathbf{x}, \boldsymbol{\beta})$ is the likelihood and $P(\mathbf{x}^m|\mathbf{x}^o, \boldsymbol{\alpha})$ is the covariate distribution, then the observed data likelihood to be maximized is

$$\mathcal{L}(\boldsymbol{\beta}, \boldsymbol{\alpha}) = \prod_i^n \int P(y_i|\mathbf{x}_i, \boldsymbol{\beta})\, P(\mathbf{x}_i^m|\mathbf{x}_i^o, \boldsymbol{\alpha})\, d\mathbf{x}_i^m. \tag{1.1}$$

Because equation 1.1 can be difficult to maximize directly, the EM algorithm [28] provided a computationally feasible solution to maximizing the observed data likelihood. The algorithm avoids the computational difficulties of the observed data likelihood by working with the complete data log-likelihood for which maximization algorithms

are already available. One starts the algorithm by postulating values for parameters $\boldsymbol{\beta}^{(0)}$ and $\boldsymbol{\alpha}^{(0)}$. Then, the iterations of the algorithm consist of two steps. In the first, one replaces the unobserved complete data log-likelihood with its expectation conditional on the observed data, and postulated values of model parameters,

$$E\{\ell(\boldsymbol{\beta}, \boldsymbol{\alpha}) \mid \mathbf{X}^o, \mathbf{y}, \boldsymbol{\beta}^{(0)}, \boldsymbol{\alpha}^{(0)}\} = \sum_{i=1}^{n} E\{\log[P(y_i|\mathbf{x}_i, \boldsymbol{\beta})] \mid \mathbf{x}_i^o, y_i, \boldsymbol{\beta}\} + \sum_{i=1}^{n} E\{\log[P(\mathbf{x}_i^m|\mathbf{x}_i^o)] \mid \mathbf{x}_i^o, y_i, \boldsymbol{\alpha}\}.$$

At the second step and with the expectation in hand, one maximizes the log likelihood as if the data were fully observed. The estimates $\hat{\boldsymbol{\beta}}$ and $\hat{\boldsymbol{\alpha}}$ update the previously postulated model parameters, $\boldsymbol{\beta}^{(1)} = \hat{\boldsymbol{\beta}}$ and $\boldsymbol{\alpha}^{(1)} = \hat{\boldsymbol{\alpha}}$. The expectation and maximization steps repeat, each time updating the postulated parameter values with the estimates. The repeated two steps give rise to the name EM: expectation and maximization. The EM iterations continue until the model parameters converge.

Louis [70] provided formulas to estimate variance and covariance via the observed information matrix. Wu [109] defined regularity conditions and provided convergence properties for the EM algorithm. Although EM is guaranteed to converge under mild conditions, the rate of convergence can be slow. In light of the slow convergence, several researchers have proposed modifications to improve speed; these include Meng and van Dyk [73], Meng and van Dyk [72], Berlinet and Roland [13], and Liu and Rubin [67].

The EM algorithm has been successfully employed in a number of parametric models involving missing data. Fuchs [39] applied it to missing data in log linear models. Ibrahim [55] applied it to generalized linear models with missingness in discrete covariates. Lipsitz and Ibrahim [64] addressed missingness in categorical covariates in survival analysis; Herring and Ibrahim [52] developed a weighted EM in order to estimate Cox model parameters when predictor values are missing. Herring

and Ibrahim [53] applies EM to cure rate models with random effects when there is non-ignorable missingness in the predictors. The successful application of the EM algorithm to parametric and semi-parametric models is substantial, and it highlights the method's utility.

Weighted estimating equations

Likelihood estimation can be seen as a member of a broader family of estimation methods known as estimating equations. Introduced in Godambe [43] and Godambe and Thompson [44], an estimating equation for parameter $\boldsymbol{\beta}$ is a function $\psi(y, \mathbf{x}, \boldsymbol{\beta})$ which satisfies the expression

$$E_P[\psi(y, \mathbf{x}, \boldsymbol{\beta})] = 0. \tag{1.2}$$

Estimation of $\boldsymbol{\beta}$ is based on the empirical expectation; the estimate $\hat{\boldsymbol{\beta}}$ is selected so that

$$\sum_i^n \psi(y_i, \mathbf{x}_i, \hat{\boldsymbol{\beta}}) = 0.$$

The estimate is unbiased, and the method is free of any specific distributional assumptions. Estimating equations are a framework often used in causal inference and robust estimation research. Robins et al. [86] and Bang and Robins [8] introduced weighted estimating equations as a missing data method. The method, known as the doubly robust estimator, builds on earlier ideas known as inverse probability weighting. We summarize both. Let $\dot{r}_i$ indicate if patient i is a complete case.

The inverse probability weighted (IPW) method starts with the selection probability, $p_i = P(\dot{r}_i = 1|\mathbf{x}_i^o, y_i)$. The estimate $\hat{\boldsymbol{\beta}}$ is selected so that

$$\sum_i^n \frac{\dot{r}_i}{p_i} \psi(y_i, \mathbf{x}_i, \hat{\boldsymbol{\beta}}) = 0.$$

Only complete case observations contribute to the estimating equation, but value $1/p_i$ weights each observation so that the expectation of the estimating equation is the desired output. For example, if missingness is a function of gender and responses are less likely from males, the inverse selection probability up-weights males and down-weights females so that the resulting average reflects the entire population and not the complete case sample.

The doubly robust (DR) estimator builds on the IPW. The key improvement of the DR estimator over the IPW estimator is that incomplete cases contribute to the estimating equation. This is achieved with a surrogate function, $\phi(Y, X^o, \theta)$, which takes the place of ψ for incomplete cases. The DR estimating equation is

$$\sum_i^n \frac{\dot{r}_i}{p_i}\psi(y_i, \mathbf{x}_i, \boldsymbol{\beta}) - \left(\frac{\dot{r}_i}{p_i} - 1\right)\phi(y_i, \mathbf{x}_i^o, \boldsymbol{\beta}).$$

Bang and Robins [8] shows that the resulting estimator is consistent if either (a) the estimates of p_i are correct or (b) the surrogate function ϕ approximates well the function ψ. For optimal efficiency, the surrogate function is selected as

$$\phi(y_i, \mathbf{x}_i^o, \boldsymbol{\beta}) = E[\psi(y_i, \mathbf{x}_i, \boldsymbol{\beta})|y_i, \mathbf{x}_i^o],$$

which in practice is often approximated with imputation techniques. See Carpenter et al. [18], Vansteelandt et al. [105], and Ibrahim et al. [57] for reviews of DR estimators. Note that likelihood based estimation can be framed within estimating equations. The score function $S(\boldsymbol{\beta}) = \frac{\partial \ell(\boldsymbol{\beta})}{\partial \boldsymbol{\beta}}$ can be an estimating equation ψ. Thus, the IPW and DR missing data methods can apply to likelihood estimation as well.

Bayesian estimation

Briefly, consider the Bayesian estimation of $\boldsymbol{\beta}$ with missing data. Ibrahim et al. [56] showed that the Bayesian paradigm offers a straightforward approach to missing

data. Like likelihood methods, it begins with a likelihood function $P(y|\mathbf{x}, \boldsymbol{\beta})$, a data model $P(\mathbf{x}^m|\mathbf{x}^o, \boldsymbol{\alpha})$, and possibly a missing data model $P(\mathbf{r}|\phi)$. One also assumes a prior distribution $P(\boldsymbol{\beta}, \boldsymbol{\alpha}, \boldsymbol{\phi})$ for the model parameters. Estimates of $\boldsymbol{\beta}$ are based on the observed data posterior distribution $P(\boldsymbol{\beta}|\mathbf{x}^o, y)$. Operationally, Bayesian analyses usually involve sequential sampling methods, like Gibbs, to draw from the complete data posterior distribution. Without missing data, the sampling sequence at each iteration draws from (a) $P(\boldsymbol{\beta}|\boldsymbol{\alpha}, \mathbf{x}^o, \mathbf{x}^m, y)$ then (b) $P(\boldsymbol{\alpha}|\boldsymbol{\beta}, \mathbf{x}^o, \mathbf{x}^m, y)$. With missing data, the sequence also includes draws from (c) $P(\mathbf{x}^m|\boldsymbol{\alpha}, \boldsymbol{\beta}, x^o, y)$. Inference about $\boldsymbol{\beta}$ is based on summaries of the resulting sample.

1.3 Statistical learning methods for missing data

In this section, we discuss a general framework for supervised classification methods and properties of optimal classifiers. Additionally, we discuss several statistical learning classifiers and their associated missing data methods. This summary covers binary classification, though the concepts can be generalized to classification into several groups.

To begin, we introduce notation specific to classification in the context of a patient population where the outcome is cure status. Those that are cured, label $y_i = +1$; label all others $y_i = -1$. Let y and $\mathbf{x}$ denote the outcome and covariates of a generic patient not in the training set $\mathcal{T}_n$. The classification task is to construct a classification rule $f : \mathbb{R}^d \to \{\pm 1\}$ which predicts cure status y from inputs $\mathbf{x}$. To measure classification performance, consider the four possible outcomes of a classifier for a single patient:

	y	
$f(\mathbf{x})$	+1	-1
+1	correct	error A
-1	error B	correct

The relative importance of error A to error B depends on the specific application; when both errors are penalized equally, classification error is captured by the c-loss function $L_c[y_o, f(\mathbf{x})] = I[y_o \neq f(\mathbf{x})]$ or equivalently $I[y_o f(\mathbf{x}) < 0]$. The performance of any classification rule is measured in terms of average classification error (or risk) which is defined as

$$\mathcal{R}_{L_c,P}(f) = E_P\Big\{L_c[y_o, f(\mathbf{x})]\Big\} \tag{1.3}$$

where P denotes the distribution $P(y, \mathbf{x})$. Optimal classifiers minimizes this quantity. Such classifiers are known as Bayes classifiers, and the minimum classification risk is the Bayes risk, i.e.,

$$f_{bayes}(x) = \arg\min_f \mathcal{R}_{L_c,P}(f) \quad \text{and} \quad \mathcal{R}_{L_c,P}(f_{bayes}) = \min_f \mathcal{R}_{L_c,P}(f).$$

If the distribution $P(y, \mathbf{x})$ is known, then the Bayes classifier can be calculated directly as

$$f_{bayes}(x) = \text{sign}\left[P(y = +1 \,|\, \mathbf{x}) - \frac{1}{2}\right]. \tag{1.4}$$

Many of the methods discussed in this section are plug-in estimates, in that the primary endeavor of several methods is to generate an estimate of $P(y = +1 \,|\, \mathbf{x})$. In fact, the methods that follow can be classified into four groups. The first group assumes a distribution, usually some form of the Bernoulli distribution $P(y|\theta(\boldsymbol{\beta}, \mathbf{x}))$ where $\theta(\boldsymbol{\beta}, \mathbf{x})$ is the odds parameter and a function of $\mathbf{x}$ and $\boldsymbol{\beta}$. The second endeavors to estimate the conditional distribution without parametric assumptions. The third group is the distribution free k-nearest neighbors. And, the last group approaches the problem with empirical risk minimization.

1.3.1 Plug-in method

In the plug-in method, one estimates $p(y = +1 \,|\, \mathbf{x})$ directly via parametric or semi-parametric methods. Logistic regression is an extremely popular method and is

a standard likelihood and Bayesian model. In the context of classification performance and finding a Bayes classifier, the logistic regression model starts with the assumption that $y|\mathbf{x}$ is a Bernoulli random variable with odds parameter $\theta(\boldsymbol{\beta}, \mathbf{x})$. Thus, once $\theta(\boldsymbol{\beta}, \mathbf{x})$ is estimated, the model $P[y|\theta(\mathbf{x})]$ can be plugged into equation (1.4) as an estimate of the Bayes rule. If the distribution and modeling assumptions are correct, the resulting classifier is asymptotically a Bayes classifier.

The simplest model of log odds is the linear model. It is:

$$\log[\theta(\mathbf{x})] = \mathbf{x}^t\boldsymbol{\beta}$$

where $\boldsymbol{\beta}$ is a vector of size d, and the model parameters are interpreted in terms of odds ratios. Estimates of $\boldsymbol{\beta}$ are computed by maximizing the likelihood, a task usually achieved with the iterative Newton-Raphson algorithm or with iteratively reweighted least squares. Logistic regression is a low variance estimator in the sense that repeated application of the logistic regression model to data generated in the same way will lead to similar estimates from one dataset to the next. This stability is an important benefit. The drawback, however, is that the model only captures linear relationships between the outcome and predictors. As a likelihood based model, methods for missing data include the four discussed in detail earlier: imputation, maximum likelihood via EM, weighted estimating equations, and Bayesian estimation.

When the predictor $\mathbf{x}$ is high dimensional, one goal is to find which predictors are important. The LASSO, introduced in Tibshirani [99], and Elastic Net, introduced in Zou and Hastie [119], are regularized forms of logistic regression. In the case of LASSO,

$$\hat{\boldsymbol{\beta}} = \arg\min_{\beta} \ell(\boldsymbol{\beta}) \quad \text{such that } ||\boldsymbol{\beta}||_1 \leq c;$$

and elastic net,

$$\hat{\boldsymbol{\beta}} = \arg\min_{\beta} \ell(\boldsymbol{\beta}) + \lambda_1||\boldsymbol{\beta}||_1 + \lambda_2||\boldsymbol{\beta}||_2,$$

where in both cases $\ell(\boldsymbol{\beta})$ is similar to the linear log odds model. The parameters λ_1 and λ_2 control the degree of regularization. The estimates are found using algorithms specifically suited the task, such as least angle regression [33] for LASSO and LARS-EN for elastic net.

In case of missing data, Hastie et al. [50] recommends multiple imputation, and De Ruyck et al. [27] applied such a strategy to a lung cancer cohort. In a related study, Sabbe et al. [92] proposed an EM solution and applied it to the analysis of 273 lung cancer patients and 345 predictor variables. While no publication for the IPW solution and LASSO or Elastic Net was found, current software implementations of both methods allow for weighted observations [37]. It follows that the IPW solution could easily be implemented. Park and Casella [78] introduces a Bayesian LASSO; though, there is no mention of missing data.

Another type of plug-in method is the basis expansion model. Basis expansion models in the logistic regression family estimate the log odds with a function of the form

$$\log[\theta(\mathbf{x})] = \sum_{k=1}^{m} \beta_k h_k(\mathbf{x}).$$

The functions h_k are transformations of the data and are called basis functions. They can be polynomial-, exponential-, log-, indicator-functions, or combinations of all four. Popular choices of the basis functions include the natural cubic spline and wavelets. In these two setups, each predictor is modeled by a set of smoothing functions. The overall function is

$$\log[\theta(\mathbf{x})] = \sum_{k=1}^{d} \mathbf{h}_k(x_k)^t \boldsymbol{\beta}_k$$

where $\mathbf{h}_k(x_k)$ is a vector of basis functions for the k^{th} predictor. The advantage of this setup over the linear case is the very large class of potentially non-linear functions that result from transformations of the data. The disadvantage is that the method can be high variance in the sense that estimates calculated from one dataset to another

similarly generated dataset can be markedly different. The estimation algorithms for basis expansion models are the same as the linear model or regularized linear model because the basis expansion model is linear in terms of the transformed variables. Missing data methods for the linear model of log odds all apply to basis expansion methods because the model is linear in transformed variables. Thus, likelihood, weighting, Bayesian, and imputation methods are available.

The last type of plug-in method we discuss is the generalized additive model. The generalized additive log odds model has a similar setup to the basis expansion model, except that basis functions are not specified before hand. Rather, the model

$$\log[\theta(\mathbf{x})] = \sum_{k=1}^{d} \beta_k h_k(x_k)$$

is composed of nonlinear functions h_k estimated along with the coefficients. The function $h_k(x_k)$ takes a single predictor as input and is estimated as a scatterplot smoother. Again, the contribution of the generalized additive model is a very flexible model of log odds which can reveal non-linear relationships between y and $\mathbf{x}$. Computation of the coefficients and functions is achieved through a back fitting algorithm described in Yee [115]. Hastie [48] suggests mean imputation as a missing data method, and French and Wand [36] provides an EM solution for missing data in an application of estimating spatially correlated cancer incidence rates.

1.3.2 Discriminant analysis

Discriminant analysis does not assume a distribution for $P(y|\mathbf{x})$; rather, it assumes a distribution for $P(\mathbf{x}|y)$. Specifically, $\mathbf{x}|y \sim N_d(\mu_y, \Sigma_y)$. The quadratic classifier

(QD) is constructed by calculating the log likelihood ratio

$$QD(\mathbf{x}) = \log \frac{P(\mathbf{x}|y=+1)}{P(\mathbf{x}|y=-1)}$$
$$= (\mathbf{x}-\mu_{+1})^t \Sigma_{+1}^{-1}(\mathbf{x}-\mu_{+1}) + \log \Sigma_{+1}^{-1} - (\mathbf{x}-\mu_{-1})^t \Sigma_{-1}^{-1}(\mathbf{x}-\mu_{-1}) - \log \Sigma_{-1}^{-1}.$$

The linear classifier (LD) assumes a common covariance among groups, $\Sigma_{-1} = \Sigma_{+1} = \Sigma$, which simplifies the ratio to

$$LD(\mathbf{x}) = \mathbf{x}^t \Sigma^{-1}(\mu_{+1} - \mu_{-1}).$$

In linear and quadratic versions, the classifier predicts $y = +1$ if the ratio exceeds a constant c,

$$f_{LDA}(\mathbf{x}) = \text{sign}[LD(\mathbf{x}) - c] \qquad f_{QDA}(\mathbf{x}) = \text{sign}[QD(\mathbf{x}) - c].$$

Note that both classifiers are based on estimates of means and covariances from the normal distribution. Thus, all five general types of missing data methods — likelihood EM, estimating equations, Bayesian estimation, imputation, and complete case — are available. The work in Little [65], Beale and Little [10] and Chan and Dunn [19] address the imputation and likelihood type solutions. Their contributions represent a number of pre-Rubin 1976 solutions, and represent ad-hoc recommendations. Shortly after the introduction of the EM algorithm, an EM solution for constructing discriminant functions with missing data was introduced in Little [65] as one among several proposed missing data solutions; in a small comparison the EM solution performed as well as other single imputation type options.

The fact that the classifier is based on the multivariate normal model highlights its advantages and disadvantages. First, the method does not perform well when the underlying distribution deviates significantly from the normal. Second, the method requires estimation of covariance parameters, which is an issue (a) when the data

contain outliers or (b) in high dimensional settings. However, when the underlying assumptions are true, discriminant analysis can be a Bayes classifier if the cutoff is properly selected.

1.3.3 Trees based methods

In this section, we consider a family of methods which estimate $P(y|\mathbf{x})$ with what is called a tree. Unlike the estimation in the previous section, tree methods attack the conditional probability directly and non-parametrically. We first present a single tree classifier, and then we present random forest classifiers which combine information from several single tree classifiers.

Single classification tree

A classification tree introduced in Breiman et al. [17] is a series of splits which partition the predictor space ($\mathbb{R}^d$). Figure 1.1 provides a small example. In the figure, capital letters refer to predictors and lower case letters refer to thresholds. The tree starts at the top; the first partition sends observations with $A > a$ to the right and all others to the left. At the second level, both partitions are split again. The left partition is split so that observations with $B > b$ are sent to the right and the remainder are sent to the left. Likewise, the right partition is split with variable C. The recursive partitioning continues for a prespecified number of levels. In Figure 1.1, the tree has 8 terminal groups labeled $P1$, $P2$, etc.

A classification tree is constructed so that the terminal groups are homogenous in terms of the outcome. There are two phases to the tree's construction: growing and pruning. During the iterations of the growing phase, the algorithm selects a predictor variable and threshold which minimizes variance in the resulting groups. During the pruning phase, splits are eliminated which fail a complexity-benefit criterion.

Estimates of $P(y = +1|\mathbf{x})$ are calculated by finding the terminal partition for $\mathbf{x}$ and

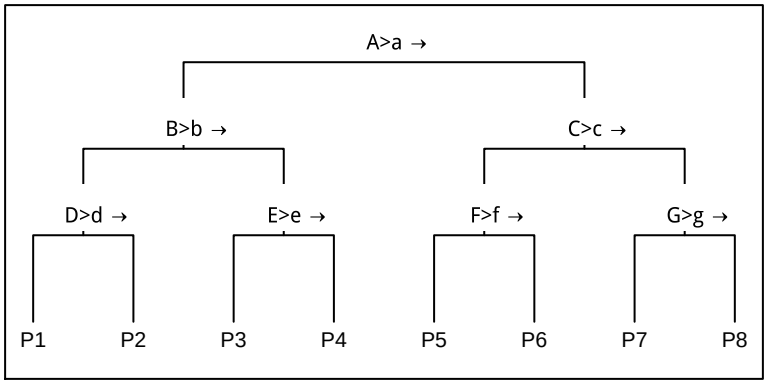


Figure 1.1: Example of classification tree

calculating the proportion of $y = +1$ outcomes in the group. The estimates are plugged into the Bayes rule to generate the classification rule. Because of the sequential nature of selecting splits, trees are sensitive to the earliest splits. Errors because of outliers are perpetuated. This instability is a drawback of the classification tree.

Random forests

Random forests [16] are an ensemble method: a single classifier is constructed by averaging the predictions of several single trees. The individual trees are constructed with a slight modification from before. Rather than considering all predictors as candidates for a split, only a random subset of predictors are considered. A new random subset is selected at each opportunity to make a split. The trees are not pruned.

Thus, if $\hat{P}_k(y = +1|\mathbf{x})$ is the k^{th} random tree generated, then the random forest (RF) estimate is

$$\hat{P}_{RF}(y = +1|\mathbf{x}) = \frac{1}{B}\sum_{k=1}^{B} \hat{P}_k(y = +1|\mathbf{x}).$$

The number of trees B is usually several hundred ($B > 200$); and the number of predictors selected at each node is usually $\sqrt{d}$. Both parameters can be tuned. Because the random forest classifier is an ensemble of single trees, the method avoids the

instability issues inherent in single trees. Also, the sharp divisions between partitions in a single tree are smoothed in a random forest. Lastly, the random forest classifier outperforms a single tree [16].

There are several tree-specific missing data methods, along with EM and imputation. Any single tree missing data method can be applied to random forests. The tree-specific methods are described in [9, 31, 50]. The following descriptions are summarized from those references.

Surrogate split

Consider the step of selecting a split in the tree. Each predictor is considered individually; if a predictor value is missing, then the observation is momentarily omitted in computations related to the threshold and homogeneity of the outcome. Once the best predictor-threshold split is selected, the algorithm determines a sequence of predictors-thresholds which best replicate the first choice. The sequence is ordered by the quality of the replication. Thus, each split is represented by a sequence of predictor-thresholds, $(B < b, E < e, \ldots, W < w)$, instead of just one. At the moment of classification, an observation is directed to the left or right partition based on the first applicable predictor-threshold pair.

Probability split

Consider the step of sending an observation to the left or right. If the observation is missing the value of the needed predictor, the observation is sent to both sides. However, the contribution to the right partition is weighted to reflect the probability of being assigned right, likewise for the contribution to the left partition. Computations downstream of the split are altered to reflect observation weights. Depending on the tree and the predictors that are missing, an observation may appear in several terminal groups, weighted so that the total contribution sums to one. Generally, probability

weights are naively calculated. The weight in the right partition is the proportion of observed observations in the right partition, likewise for the left weights.

Missing category

In the case of a nominal categorical predictor with missing values, missing is treated as a category. In the case of ordinal data, both discrete and continuous, missing values are replaced with a large value far outside the normal range of values. This allows the algorithm to create partitions which separate missing and non-missing observations. Of course, this strategy can be sensitive to replacing missing values with a large positive value or a large negative value.

A simulation study [31] compared single-tree performance of these three tree-specific methods (with caveats for surrogate split) along with mean imputation and complete case. The probability split performed well in situations when the validation set did not have missing values. The missing category worked well in situations when the missingness mechanism was a function of the outcome y. This is not surprising because the missing category can capture that relationship well.

1.3.4 k-nearest neighbor

The methods presented up to this point have all approached the classification task via a distribution, either by explicitly assuming a distribution (logistic, LDA, and QDA classifiers) or by estimating a distribution (tree-based classifiers). In this section, we consider the nearest neighbor classifier which does not necessarily assume or estimate a distribution.

The k-nearest neighbors (KNN) classifier predicts y from $\mathbf{x}$ by taking a poll of the k nearest neighbors in the training set $\mathcal{T}_n$. In the case of 3-nearest neighbor, the predicted outcome is the majority outcome of the 3 nearest training points. The k-nearest neighbors classifier has been successfully used in a number of computer

vision tasks [50], and is especially useful when the probability of group membership changes abruptly at the boundary. Further, because KNN does not require any distributional assumptions, it can be applied in nearly any situation. However, the resulting model is not easily interpreted, and calculating distances in large databases requires considerable memory.

Imputation is the most prevalent missing data method for KNN classifiers [2]. KNN itself is used widely as an imputation technique [51], and it has been widely used. Missing values $\mathbf{x}_i^m$ are predicted from KNN applied to $\mathbf{x}_i^o$. Say for $k = 5$, the replacement of $\mathbf{x}_i^m$ is the average of the 5 other training values closest to $\mathbf{x}_i^o$. In multiple imputation setups, replacement values are drawn from the set of nearest neighbors.

1.4 Support Vector Machines

As noted earlier, SVMs are a statistical learning classifier introduced in Boser et al. [15], Cortes and Vapnik [25] and Vapnik [106]. The method has historical basis in linear separating hyperplanes, and the following description builds on that basis. For the sake of demonstration, suppose that that $\mathbf{x}$ is two-dimensions, and that we can plot the data as in Figure 1.2. In this case, the two groups are separable in the sense that there is a lines (planes in higher dimensions) in the $\mathbf{x}-$space which separates the groups. In fact there are infinitely many such lines; two are shown in the figure. Such lines can be used as classifiers for future, unseen points by labeling points on one side of the line as +1 and labeling the others as -1. If we parametrize each line as

$$\{\mathbf{x} \,|\, w^x\mathbf{x} + b = 0\},$$

then the classifier is $f(\mathbf{x}_o) = \text{sign}(w^t\mathbf{x}_o + b)$, and the signed distance between a point $\mathbf{x}_o$ and the line is $||w||^{-1}(w^t\mathbf{x}_o + b)$. For each separable line, the margin is the perpendicular

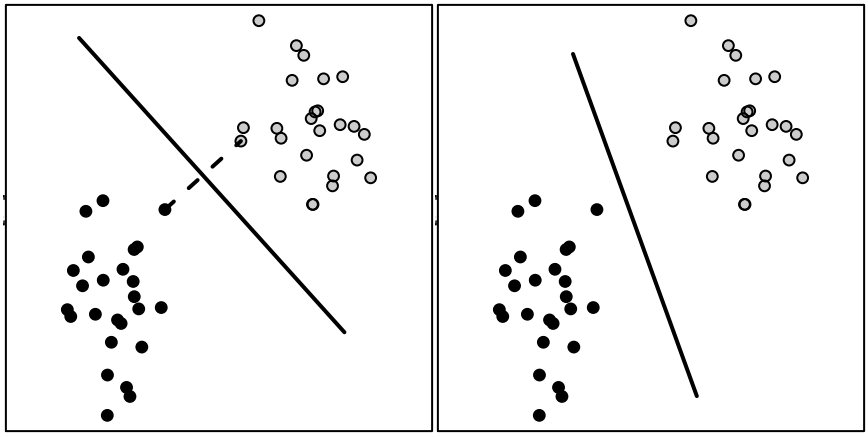


Figure 1.2: SVM Demonstration, toy data

distance from the line to the nearest point,

$$\text{margin(w,b)} = \min_i \ \|w\|^{-1}|w^t\mathbf{x}_i + b|.$$

The motivating, geometric interpretation of a linear SVM is to select the line, of all separating lines, which maximizes the margin. This can be expressed as

$$\min_{w,b} \|w\| \quad \text{such that} \quad y_i(w^t\mathbf{x}_i + b) \geq 1, \quad i = 1, \ldots, n.$$

The constraint enforces separability. Note in the simple example in Figure 1.2, the line on the left (the SVM solution) does maximize the margin, while the one on the right does not. In the case of unseparable data, the geometric interpretation of the SVM introduces the concept of a slack variable ξ_i, which provides a penalty if a point is

misclassified. Specifically,

$$\begin{aligned} &\min_{w,b} \|w\| + C \sum_{i}^{n} \xi_i \\ &\text{such that} \quad y_i(w^t \mathbf{x}_i + b) \geq 1 - \xi_i \qquad i = 1, \ldots, n \\ &\qquad\qquad \xi_i \geq 0 \qquad i = 1, \ldots, n. \end{aligned} \tag{1.5}$$

Thus, in the unseparable case, some points will always incur a penalty because the definition of unseparable implies $y_i(w^t \mathbf{x}_i + b) < 0$ for at least one point for every w and b. The parameter C controls the tradeoff between the penalty and the complexity of the classifier. Maximization of the SVM is a quadratic programming problem which can be expressed as

$$\begin{aligned} &\min_{\alpha} \sum_{i=1}^{n} \alpha_i - \frac{1}{2} \sum_{i=1}^{n} \sum_{j=1}^{n} \alpha_i \alpha_j y_i y_j \mathbf{x}_i^t \mathbf{x}_j \\ &\text{such that} \quad 0 \leq \alpha_i \leq C \\ &\qquad\qquad \sum_{i=1}^{n} \alpha_i y_i = 0, \end{aligned}$$

where the solution is $w = \sum_{i=1}^{n} \alpha_i y_i \mathbf{x}_i$. The key contribution of Cortes and Vapnik [25] was to identify that the data $\mathbf{x}_i$ enters the objective function through the dot-product $\mathbf{x}_i^t \mathbf{x}_j$, and that general forms of dot-products in a Hilbert space, $\phi(\mathbf{x}_i) \cdot \phi(\mathbf{x}_j) = \kappa(\mathbf{x}_i, \mathbf{x}_j)$, could replace the euclidean dot-product without affecting the computational burden. General dot-product forms represent a large family of transformations on the data, $\phi(\mathbf{x}_i)$, but they only enter into the computation via a kernel function κ which satisfies certain conditions. Such transformations admit flexible, non-linear functions of the form

$$f(\mathbf{x}) = b + \sum_{\mathbf{x}_i \in \mathcal{T}_n} c_i \kappa(\mathbf{x}, \mathbf{x}_i). \tag{1.6}$$

This important link between the geometric beginnings of SVM and the kernel transformation recast the method into a framework of empirical risk minimization within a Reproducing Kernel Hilbert Space (RKHS) of functions. To see the connection, return to (1.5) and rewrite $(w^t \mathbf{x}_i + b)$ as $f(\mathbf{x}_i)$

$$\min_{f \in \mathcal{H}} \|f\| + C \sum_i^n \xi_i$$

$$\text{such that} \quad \xi_i \geq 1 - y_i f(\mathbf{x}_i) \qquad i = 1, \ldots, n$$

$$\xi_i \geq 0 \qquad i = 1, \quad i = 1, \ldots, n.$$

Allow $L_h[y_i, f(\mathbf{x}_i)] = \max[0, 1 - y_i f(\mathbf{x}_i)]$, and the objective function is identical to the following objective:

$$f_{svm} = \arg\min_{f \in \mathcal{H}} \underbrace{\lambda \|f\|_{\mathcal{H}}^2}_{\text{penalty}} + \underbrace{\mathcal{R}_{L_h,D}(f)}_{\text{empirical h-risk}} , \tag{1.7}$$

$$\text{empirical h-risk: } \mathcal{R}_{L_h,D}(f) = \frac{1}{n} \sum_i^n \underbrace{\max[0, 1 - y_i f(\mathbf{x}_i)]}_{L_h}.$$

which in an empirical and regularized approximation of the Bayes objective function,

$$f_{bayes}(\mathbf{x}) = \arg\min_f \mathcal{R}_{L_c,P}(f).$$

Note the subscript D to denote the empirical distribution instead of P for the true distribution. Regularized empirical risk minimization over a RKHS provides a framework for exploring the statistical properties of the method, including asymptotic properties. It is also in this framework that we develop our proposed methods for missing data.

Asymptotic properties

The SVM solution is unique, stable, consistent, and ensures generalization [94, 98, 34, 50]. Consistency is defined in the statistical sense that finite sample h-risk converges in probability to the minimum h-risk as n gets large.,

$$\mathcal{R}_{L_h,P}(f_{svm}) \xrightarrow{p} \min_f \mathcal{R}_{L_h,P}(f).$$

Generalization refers to fact that the training risk converges in probability to true risk,

$$\mathcal{R}_{L,D}(f_{svm}) \xrightarrow{p} \mathcal{R}_{L,P}(f_{svm}),$$

and it suggests that the SVM solution does not over-fit the data as a non-regularized empirical risk solution would.

Steinwart and Christmann [98] and Schölkopf and Smola [94] discuss a number of computational advantages to this setup. The h-loss function seen in equation (1.7) is convex, which allows the implementation of efficient optimization algorithms. As noted above, the SVM solution converges in probability to the minimum h-risk; however, the primary goal is to find a solution that minimizes the c-risk. In fact, as $n \to \infty$, classification risk of f_{svm} converges to the Bayes risk,

$$\mathcal{R}_{L_c,P}(f_{svm}) \to \min_f \mathcal{R}_{L_c,P}(f).$$

Kernel function and tuning parameters

The RKHS $\mathcal{H}$ is characterized by a kernel function $\kappa : \mathbb{R}^d \times \mathbb{R}^d \to \mathbb{R}$, and functions in $\mathcal{H}$ are of the form in equation (1.6), and are a linear combination of basis functions $h_i(\mathbf{x}) = \kappa(\mathbf{x}, \mathbf{x}_i)$. Computationally, the construction of $f_{svm}(\mathbf{x})$ centers on finding values for $c_1, \ldots, c_n$ and b. Predictions from the SVM classifier are calculated as sign$[f_{svm}(\mathbf{x})]$. The choice of kernel function affects the form of the classification function. Two

common choices of κ are

$$\underbrace{\kappa(\mathbf{u}, \mathbf{v}) = \mathbf{u}^t\mathbf{v}}_{\text{Linear kernel}} \qquad \text{and} \qquad \underbrace{\kappa(\mathbf{u}, \mathbf{v}) = e^{-\gamma\|\mathbf{u}-\mathbf{v}\|^2}}_{\text{Guassian kernel}}.$$

The linear kernel generates decision rules with a linear boundary; the Gaussian kernel generates both linear and nonlinear boundaries. The details regarding the inherent transformation of Gaussian kernel functions is discussed in Steinwart and Christmann [98].

Note that equation (1.7) includes the parameter λ and the definition of the Gaussian kernel function includes parameter γ. These parameters are tuning parameters. The cost parameter λ (also reparameterized as C) controls the impact of the complexity penalty relative to the empirical h-risk; the parameter γ is a scale or bandwidth parameter. In practice, the values of tuning parameters are selected from a list of potential values on the basis of cross validation.

Computational details

Equation (1.7) can be re-expressed as a constrained quadratic programming problem for n-vector $\boldsymbol{\alpha}$:

$$\begin{aligned} \min_{\boldsymbol{\alpha}} \ & \frac{1}{2}\boldsymbol{\alpha}^t\mathbf{W}\boldsymbol{\alpha} - \mathbb{1}^t\boldsymbol{\alpha} \\ \text{such that } & \mathbf{y}^t\boldsymbol{\alpha} = 0 \\ & 0 \le \boldsymbol{\alpha}_i \le C, \quad i = 1, \ldots, n \end{aligned} \tag{1.8}$$

Note the conventions: (a) $\mathbb{1}$ is a vector of ones, (b) $C = 1/(2\lambda n)$, (c) the kernel matrix $\mathbf{K}_{ij} = \kappa(\mathbf{x}_i, \mathbf{x}_j)$, and (d) $W_{ij} = y_i\mathbf{K}_{ij}y_j$. With optimal $\boldsymbol{\alpha}$, the SVM classifier is defined in equation (1.6) with $c_i = y_i\alpha_i$. One result of the computation is that $\alpha_i = 0$ for observations outside the margin, that is those with $y_i f(x_i) > 1$. This means the corresponding

component of the classifier in equation (1.6) can be omitted. Only those observations inside the margin, $y_i f(x_i) \leq 1$, contribute to the classifier and are call 'the support vectors'. As such, the SVM solution represents data reduction in the sense that all non-support vectors can be omitted from the training set without affecting the resulting classifier. This, especially in large databases, is an advantage of the SVM model.

A wide variety of computational algorithms exist for solving this quadratic programming problem; most SVM software packages implement a type of sequential minimal optimization (SMO). This specific algorithm can solve (1.8) even when the memory required for the $n \times n$ matrix $\mathbf{W}$ exceeds available computer memory; thus, the SMO algorithm can generate a solution when the sample size is large. For a discussion of computational details see [20, 94].

SVM example with toy data

For completeness, we end this section on support vector machines with a simple demonstration of a SVM classifier constructed from a training set of 50 patients. Plots of the constructed classifier along with details of the demonstration are reported in Figure 1.3. Both the linear and non-linear boundaries are demonstrated. The toy example highlights a number of advantages and disadvantages of the SVM model. In the linear case, the SVM is interpretable in terms of the maximum margin plane described earlier. In the non-linear case, the interpretation of the SVM solution is not straight forward. The solution represents the maximum margin plane in a vaguely specified transformation of the data. In the two dimensional toy example, the boundary can be plotted and interpreted. In higher dimensions, no descriptive interpretation is available.

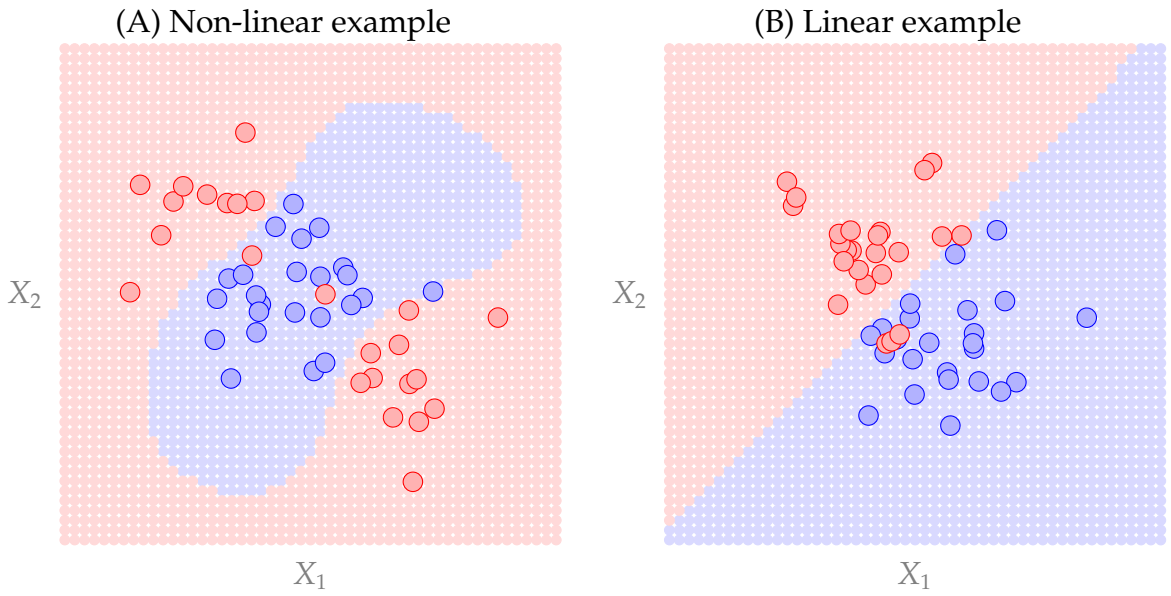


Figure 1.3: Examples of Two SVM Classifiers

Panel (A) is a non-linear example fit with Gaussian kernel; panel (B) is a linear example fit with linear kernel. Training set data is plotted as larger, darker circles. The regions defined by the SVM classifier are denoted with the red or blue background colors. The tuning parameters selected with cross validation are $C = 0.125$.

SVM methods for missing data

Improper Imputation

There are a number of improper imputation methods posited for SVMs and other statistical learning methods. Many of the methods rely on statistical learning methods in order to generate the underlying imputation distribution. For example, [2] encouraged the use of KNN imputation and [41] suggested Naive Bayes type imputation, while neither provided examples specific to SVMs. In Farhangfar et al. [35], the authors consider the Gaussian SVM with hot deck imputation, Naive Bayes, mean imputation, and regression-based imputation. To the author's credit, they consider these imputations over a very large range of missingness levels, ranging from 5% to 50%. However, the missing data mechanism was MCAR. The authors reported very mild accuracy improvements over the complete case classifier. Dick et al. [30]

suggests an imputation scheme in which parameters of the imputation distribution are jointly selected with the classification function. That is, it searches for the classifier and the missing data distribution which minimizes loss. The performance of the method is compared to single imputation with MCAR missing data. On balance, results comparing this imputation method to single imputation showed 1% - 2% improvement in accuracy, though some example datasets showed around 4% improvement.

Kernel Completion

Kernel completion is an SVM specific missing data method suggested in Tsuda et al. [100] and Anderson and Gupta [3]. The method centers on equation (1.8), the quadratic programming problem which solves the SVM objective function. Note that data from the predictors $\mathbf{x}_i$ enter the expression solely through the kernel matrix $\mathbf{K}$. Thus, when the outcomes y_i are fully observed and missing data is restricted to the predictors, the kernel completion method for missing data is to reconstruct $\mathbf{K}$. With $\mathbf{K}$ in hand, a solution for c_is in (1.6) is available. When $\mathbf{K}$ is the linear kernel, no additional steps are needed. When $\mathbf{K}$ is a non-linear kernel, one must also specify $\kappa(\mathbf{x}, \mathbf{x}_i)$ in the same equation.

In Tsuda et al. [100], authors propose a kernel completion method in which one uses auxiliary information to fill-in the missing portions of the kernel matrix. The argument goes something like this: Let $\mathbf{P}$ be the partially observed kernel matrix. Let $\mathbf{M}$ be a kernel matrix derived from fully observed auxiliary data. Think of $\mathbf{P}$ and $\mathbf{M}$ as covariance matrices from zero mean multivariate normal distributions. Then, the Kullback-Leibler distance is

$$KL(\mathbf{P}, \mathbf{M}) = tr(\mathbf{M}^{-1}\mathbf{P}) + \log\det\mathbf{M} - \log\det\mathbf{P} - n$$

where n is the number of observations. At a basic level, the objective is to complete $\mathbf{P}$

in a way that (a) $\mathbf{P}$ remains positive semi-definite and (b) minimizes $KL(\mathbf{P}, \mathbf{M})$. Because this method assumes an auxiliary data set, it is only applicable to situations when such data is available and when one can argue that the auxiliary data is a reasonable surrogate. Thus, this method is quite limited in its application.

In Anderson and Gupta [3], researchers proposed kernel completion method in which one specifies distributions for the predictor vector $\mathbf{x}_i$ and then calculates the expected kernel,

$$E[\mathbf{K}_{ij}] = \int \int \kappa(\mathbf{x}_i, \mathbf{x}_j) p(\mathbf{x}_i) p(\mathbf{x}_j) \, d\mathbf{x}_i \, d\mathbf{x}_j.$$

The authors are not clear if the assumed distributions should be conditional on the observed components. This framework is similar to multiple imputation methods: rather than averaging several classifier functions, one averages several kernels and then finds a single classifier. Like imputation, this method is easy to implement. But like imputation, it relies on convenience distributions.

Missing data loss function

Smola et al. [97] provides a principled missing data method based on a connection between SVM and exponential family distributions. Specifically, if $\frac{y}{2}\phi(\mathbf{x})$ is the sufficient statistic of $p(y|\mathbf{x})$, then the SVM solution with kernel $\kappa(\mathbf{x}_i, \mathbf{x}_j) = \phi(\mathbf{x}_i) \cdot \phi(\mathbf{x}_j)$ is also the solution which maximizes the log-likelihood ratio. The method proposes that for missing data, one replace the loss function with one constructed with sufficient statistics from the distribution $p(y|\mathbf{x}^o)$. The solution requires a number of "thorny" [97] computational issues, and it is limited in workable size.

Probability Constraint

Shivaswamy et al. [95] takes advantage of equation 1.5. The authors propose a

missing data method in which the first constraint is replaced by a probability statement:

$$P(1 - y_i f(\mathbf{x}_i) \leq \xi_i | \mathbf{x}_i^o, y_i) \geq 1 - v_i \quad v_i \in (0, 1]; \;\; i = 1, \ldots, n.$$

In essence, this method replaces uncertain observations with distributions. Thinking geometrically, the standard constraint penalized each misclassified point. The probability constraint penalizes uncertain points if $1 - v_i$ percent of the distribution is misclassified. The computation of this solution is non-trivial especially with any non-linear kernel. In the case of the linear kernel, the resulting objective function is a second order cone program. This limits the solvable sample size and increases computation time. The computational issues are exacerbated in the case of non-linear kernels like the Gaussian. Despite the computational issues, the probability constraint method is applicable to MCAR and MAR missing data situations.

Geometric Max Margin

Chechik et al. [22] proposed a missing data method built on the geometric perspective of SVMs; the basic idea is to define the margin in terms of non-missing predictor variables. That is, to define the margin within a subspace. Recall that the geometric linear SVM objective function is

$$\max_{w,b} \min_i \underbrace{||w||^{-1} |w^t \mathbf{x}_i + b|}_{\text{margin}}.$$

The geometric max margin method redefines the margin uniquely for each observation. If $\mathbf{r}_i$ is the indicator vector (as defined in section 1.2.3) then the margin within the observed subspace is

$$\text{margin}(w, b, \mathbf{x}_i, y_i) = \left[\sum_{k=1}^{d} r_{ik} w_k^2 \right]^{-1} \left| b + \sum_{k=1}^{d} w_k r_{ik} x_{ik} \right|,$$

and the objective function is

$$\max_{w,b} \min_{i} \text{ margin}(w, b, \mathbf{x}_i, y_i).$$

This geometric solution performs reasonably well when the missing data are MCAR, but it does not work well when the missing data are MAR. Further, the computation requires non-convex optimization when the two groups are not separable. This limits the computational speed and the size of the training set.

1.5 Multi-class Support Vector Machines

Due to the popularity of SVMs in the two-class setting, a natural extension of the method is the multi-class setting in which one wants to construct a classifier that distinguishes between more than two classes. There are two families of multi-class SVMs, which we consider in turn.

1.5.1 Composite-of-binary SVMs

. The first family, which we call the composite-of-binary family, constructs a set of two-class SVM rules from which a single multi-class rule is constructed. Procedures of this type are widely used in many classification settings and are not specific to SVMs [38, 49]. The three most popular composite-of-binary SVMs are one-versus-one SVM, the one-versus-many SVM, and the one-versus-one DAGSVM [81]. In a K-class setting, the one-versus-one multi-class SVM is constructed by generating all $K(K-1)/2$ binary rules that separate generic class i from generic class j. To classify a point, the covariate vector is input to each binary classifier which outputs a vote for one of two classes. The overall multi-class rule outputs the class which receives the most votes. The one-versus-one DAGSVM is a similar alternative in which classification is based on a decision tree with a one-versus-one classifier at each node. The decision tree

is constructed so that one potential class is eliminated at each decision node, and the terminal node represents the class prediction. Lastly, we describe the one-versus-many classifier which generates K binary rules which separate generic class i from all other classes. To classify a point, the covariate vector is input to each two-class rule; the final prediction corresponds to the class which generated the largest signed distance, i.e., $\hat{y} = \arg\max_{i=1,\ldots,K} f_i(\mathbf{x})$. Variations of the these composite-of-binary SVMs exist, and generally vary by how each component classifier casts a vote for a potential class. For example, using the component SVMs to generate probability estimates which in turn are combined in a final decision rule. See [49, 80, 20, 110] for a discussion of generating and combining multi-class probabilities from SVMs.

Because of their operational efficiency, versions of the composite-of-binary multi-class SVM are implemented widely in statistical software packages [20]. Both in simulation settings and applied settings, the composite-of-binary family has demonstrated its usefulness [54, 32]. Much of the operational efficiency of the composite-of-binary SVM stems from the fact that it takes advantage of specialized but widely distributed algorithms for the two-class SVM. The draw back to this type of multi-class SVM is that voting does not always generate a clear winning class. For example, a one-versus-one multi-class rule in a $K = 3$ class setting is built on 3 two-class rules. For some covariate combinations, the 3 two-class rules may generate a vote for each class. The same scenario can occur with one-versus-all multi-class rules as well. The primary advantage of the DAGSVM is that it avoids the ambiguities like ties that can arise in the standard voting scheme. Beyond the issue of ties is the issue of consistency, for which the author in [68] shows examples when one-versus-all is not Fisher consistent. Despite this issue, the composite-of-binary approach continues to be a popular multi-class SVM method [84].

1.5.2 Simultaneously Trained SVMs

The second family of multi-class SVM builds a decision rule by simultaneously training K functions where each function corresponds to a single class [107, 26, 62, 69]. Distinguished by specific multi-class loss functions, the various flavors of multi-class SVMs simultaneously construct the K functions so that the predicted class corresponds to the function with the largest value. The simultaneous estimation of the K functions ensures that the estimated multi-class rule targets the multi-class Bayes rule.

The general setup is very similar to the two-class setting, and we describe it here. Consider a training set, $\mathcal{T}_n$, of n observations, each consisting of a d-vector of covariates, $\mathbf{x} \in \mathbb{R}^d$, and multi-class outcome, $y \in \{1, \ldots, k\}$. Each observation is an iid draw from an unknown distribution $P(\mathbf{x}, y)$. Consider functions $\mathbf{f}(\mathbf{x}) = \{f_1(\mathbf{x}), \ldots, f_k(\mathbf{x})\}$ so that the class label of $\mathbf{x}$ can be predicted as

$$\hat{y} = \arg\max_i f_i(\mathbf{x}).$$

The classifier which minimizes the average classification error over $P(\mathbf{x}, y)$ is the Bayes classifier,

$$\mathbf{f}^{bayes} = \arg\min_{\mathbf{f}} E_P[y \neq \arg\max_i f_i(\mathbf{x})].$$

The average classification error of the Bayes classifier is called the Bayes risk. The goal is to construct a classifier from the training set $\mathcal{T}_n$ which is asymptotically a Bayes classifier but also performs well in finite sample situations.

The SVM solution frames the task within empirical risk minimization; specifically, the SVM solution is

$$\hat{\mathbf{f}} = \arg\min_{\mathbf{f} \in \mathcal{H}} \lambda \|\mathbf{f}\|^2_{\mathcal{H}} + \frac{1}{n} \sum_{i=1}^{n} L[y_i, \mathbf{f}(\mathbf{x}_i)] \tag{1.9}$$

$$\text{such that} \qquad \sum_{i}^{k} f_i(\mathbf{x}) = 0$$

where $\|\mathbf{f}\|_{\mathcal{H}} = \sum_i^k \|f_i\|_{\mathcal{H}}^2$ and $L[y_i, \mathbf{f}(\mathbf{x}_i)]$ is a loss function which penalizes misclassification. The multi-class SVM methods discussed here build on the reinforced multi-class SVM (RMSVM) proposed in [69] because it provides a multi-class loss function which unifies the earlier work of [62] and [107] as special cases of a general multi-class loss function. The RMSVM loss function is

$$L[y, \mathbf{f}(\mathbf{x})] = \gamma[(k-1) - f_y(\mathbf{x})]_+ + (1-\gamma)\sum_{j \neq y}[1 + f_j(\mathbf{x})]_+$$

where the function $[t]_+ = \max\{0, t\}$ and γ is a tuning parameter which calibrates the loss. The set of solutions, $\mathcal{H}$, is constructed so that each component of solution, $\hat{\mathbf{f}}$, is of the form

$$f_i(\mathbf{x}) = b + \sum_{j=1}^{n} c_j K(\mathbf{x}, \mathbf{x}_j) \qquad \mathbf{x}_j \in \mathcal{T}_n$$

where $K(\mathbf{u}, \mathbf{v})$ is a kernel function. The linear kernel, $K(\mathbf{u}, \mathbf{v}) = \mathbf{u}^t\mathbf{v}$, and the Gaussian kernel, $K(\mathbf{u}, \mathbf{v}) = \exp\{-\sigma\|\mathbf{u} - \mathbf{v}\|^2\}$, are commonly used.

As the solution to the empirical risk minimization problem, the SVM targets the conditional expectation of the loss function, $E\{\, L[y, \mathbf{f}(\mathbf{x})] \,|\mathbf{x}\}$. When $\gamma \leq 1/2$, the RMSVM solution is Fisher consistent in the sense that the minimizer of $E\{L[y, \mathbf{f}(\mathbf{x})|\mathbf{x}]\}$ is also the multi-class Bayes rule [69]. Simulation examples in [69] show that the RMSVM performs better when $\gamma = 1/2$ than when $\gamma = 0$ or 1.

1.6 Semi-supervised Learning and Support Vector Machines

Examples of collected data in which some class labels are missing are increasingly common. Any situation where determining the class label is expensive or overly invasive can generate data with fully-observed covariates but few class labels. For example, data for which class labels must be assigned by a human expert, as with

immunohistochemistry data, or data for which disease subtype must be ascertained by biopsy or expensive blood-assay, as with rare forms of hepatitis. Often, the research goals associated with these types of datasets is to construct a classifier which predicts class labels because the gold-standard method is inaccessible. Classifiers built with some missing class labels are often called semi-supervised classification rules [87].

There are existing methods for constructing semi-supervised two-class or multi-class rules in statistical learning contexts, many of which are based on missing data ideas like imputation and the expectation-maximization (EM) algorithm while other methods are based on statistical learning ideas like boosting. For example, [42] applied EM as a way to estimate Gaussian mixture models in an unsupervised, clustering context and to missing data in a supervised context. In [76], researchers perform semi-supervised text classification with a naive Bayes classifier and the EM algorithm. Imputation type semi-supervised classification include co-training [14] and ASSEMBLE [12]. In [103], authors provide a general purpose semi-supervised algorithm for any multi-class classifier based on boosting.

1.6.1 S^3VM

One semi-supervised method specific to two-class SVMs is the method introduced in [11] called S^3VM. While standard SVM methodology chooses the rule which minimizes the empirical hinge risk, S^3VM chooses the rule which minimizes the empirical hinge risk calculated as if unlabeled points are in fact correctly labeled. Because unlabeled points are treated as correctly classified, the unlabeled points in the empirical hinge risk act as a 'high density' penalty. That is to say, the S^3VM setup prefers rules with boundaries in low density regions. Computationally, the S^3VM objective function is not convex and has potentially many local minima; however, there are a number of algorithms developed to find the S^3VM solution. See [118, 117, 21] for a discussion of the various non-convex optimization techniques which have been proposed

for constructing S^3VM rules.

Multi-class extensions of S^3VM were proposed in [113] for specific types of kernels (sparse and full rank) with a specific multi-class loss function. Using a sparse Laplacian kernel matrix (a kernel to which the method applies), the algorithm employs gradient descent to find a solution. In contrast to the limited setting of sparse and full rank kernels, a more accessible multi-class extensions of S^3VM have been developed with the composite-of-binary multi-class SVM, such as the methods proposed first in [21] and adapted in [6]. In [21], the composite-of-binary multi-class SVM applies S^3VM methods to each of the two-class rules. The method requires all unlabeled points to be included when training each two-class rule, even with a one-versus-one rule. In [6], authors adapted the composite-of-binary S^3VM method of [21] by introducing participation weights of unlabeled observations. Based on a generic similarity measure, the participation weights will up-weight or down-weight the contribution of unlabeled observations when training individual two-class S^3VMs as part of the larger composite-of-binary rule. Thus, unlabeled observations which are similar to class 1 observations are given greater weight when constructing rules involving class 1. Likewise, observations not similar to class 1 are given less weight. As demonstrated in a number of examples, using participation weights in the construction of semi-supervised composite-of-binary multi-class SVMs appears to improve classifier performance.

1.6.2 Maximum Margin Clustering

Another approach to semi-supervised SVMs was introduced in [112] in the context of clustering with SVMs with a method known as maximum margin clustering [111]. The objective of maximum margin clustering is to find a set of class labels which minimizes the SVM empirical risk under the constraint that the class balance in the proposed solution is within prescribed bounds. Once the optimal set of class labels is

found, an SVM classifier is constructed from a training set composed of the new class labels and covariates. The key extension of max margin clustering to semi-supervised learning is that the max margin clustering objective function can incorporate information from observations with known outcomes by introducing constraints. Often called the semi-definite SVM (SD-SVM) because the objective function is a semi-definite programing problem, the SD-SVM is similar to S^3VM in the two-class setting but has the additional advantage of also providing a multi-class solution. However, the SD-SVM as a semi-supervised method is limited in application by the computationally expensive algorithm, its sensitivity to tuning parameters, and its assumption that the intercept term (sometimes called the bias term) is zero [102]. While [102, 116] proposed unsupervised clustering algorithms based on SD-SVM which minimize these drawbacks, the resulting methods do not admit a semi-supervised solution.

The SD-SVM can be framed as a two-step procedure in which the first step clusters the data and the second step constructs an SVM based on the predicted class labels. A cluster-then-construct solution is one in which any clustering procedure provides class labels in the first step and an SVM is constructed in the second step. SD-SVM differs from this ad-hoc procedure because the clustering criteria minimized in the first step is the SVM empirical risk. Note that the semi-supervised composite-of-binary multi-class SVM from [6] which is described above is another flavor of the cluster-then-construct type solution. The first step generates fuzzy-cluster labels in the form of participation weights and the second step constructs a composite-of-binary, one-versus-one S^3VM rules. Because this method relies on a composite-of-binary SVM, its computational burden is much less than SD-SVM.

As evidenced by the several types of semi-supervised SVMs in both the two-class and multi-class settings, there is considerable interest in developing algorithms which efficiently generate semi-supervised SVM classifiers. Despite the interest in the topic, the value of semi-supervised multi-class SVMs over complete-case multi-class

SVMs has been questioned, as a number of authors have provided examples where complete-case multi-class SVMs out perform semi-supervised multi-class SVMs, such as [120] and [63]. Further, the computation required for many semi-supervised methods can be substantial, though recent algorithms such as those described in [96] improve on earlier algorithms for two-class SVMs. In the multi-class setting, however, computation time may still be prohibitively expensive, such as the SD-SVM solution which requires estimation of $n + n_u^2$ parameters where n is the number of observations and n_u is the number of unlabeled observations. Multi-class methods which are computationally expensive like the multi-class S^3VM methods are restricted to specific application areas (sparse, full rank kernels) or rely on composite-of-many SVMs. In our judgment, there is a need for a stable, multi-class semi-supervised SVM.

1.7 Dissertation topics

The missing data methods for SVMs, with a few exceptions, are ad-hoc techniques. Those that are based on statistical reasoning are limited in sample size, only apply to MCAR missing data, or require specialized software. In the first paper, we will propose an EM-type missing data method which can be performed with standard SVM software. As such, this method will provide SVM users an important tool to address a common issue. The performance of the method is examined in a simulation study and in application to a subset of an observational study database of patients treated for hepatitis C (HCV-TARGET). The simulations cover a variety of missing data scenarios, including situations in which missingness in the covariates is a function of the outcome.

In the second paper, we will propose a weighted estimating equation missing data method. A unique feature of SVMs makes this route possible; specifically, the empirical risk function can be an estimating equation. This approach is promising because it avoids two important draw backs of the EM-type solution. First, the EM

solution requires estimation of data distribution parameters, and in high dimensional settings, the number of parameters can be very large. Second, for missingness in continuous covariates, it requires MCMC methods for sampling from the conditional EM distribution which can be time consuming. Thus, as the method developed in the second paper avoids these issues, it may be particularly helpful for high dimensional settings, a common setting for SVMs.

In the third paper, we propose an EM-type SVM to address situations involving missing class labels, an area of study commonly called semi-supervised learning. We apply the method to both two-class and multi-class SVMs. We also show how the method can be applied to settings when some information about missing class labels is available, for example when observations with missing labels are known to be of class 2 or 3 but not of class 1. We examine the performance of our proposed method with both simulated data and with data from HCV-TARGET.

CHAPTER 2: AUGMENTED AND WEIGHTED SUPPORT VECTOR MACHINES FOR MISSING COVARIATES

2.1 Introduction

Hepatitis C is the most prevalent blood born infection in the United States [24] with 3.4 to 4.9 million infected US residents [5]. The mortality burden of hepatitis C continues to grow, and in 2007 the number of deaths from hepatitis C exceeded those from HIV [71]. In the last five years, new drug therapies such as Telaprevir have offered physicians an effective means for treating hepatitis C infection [1]. However, the cost of these new treatments is substantial. In the United States, the cost per cure for Telaprevir ranges between $150,000 and $250,000 [101]. The prevalence of hepatitis C infection coupled with the cost of treatment are motivating factors for developing classification rules that identify patients which are more likely to respond to treatment. Such rules could improve the cost effectiveness of treatment and lower the risk of unnecessary therapy. HCV-TARGET is a multi-center longitudinal observational study which enrolls a diverse population of patients receiving treatment for hepatitis C. Researchers collected relevant demographic, clinical, and outcome data during treatment and follow-up. A scientific objective is to develop a classifier that can accurately predict treatment efficacy using baseline variables.

In the HCV-TARGET study, the potential complex relationship between treatment efficacy and baseline biomarkers requires a statistical approach that provides modeling flexibility. Support vector machine classification (SVM) is a statistical method that offers modeling flexibility by allowing for both linear and non-linear relationships between the predictors and outcomes. SVMs have been used within a wide range of different applications, including gene expression analysis [40], fMRI

analysis [83], and other biomedical computer vision tasks. SVMs represent a natural approach for building a decision rule because of the modeling flexibility. However, SVMs do not easily accommodate missing covariate information, yet this is of prime importance within the HCV-TARGET study as the researchers were unable to collect all baseline data for all patients for a wide variety of reasons. For example, physicians sometimes collected certain chemistry measures and not others depending on the patient's health or the presence of comorbidities. In other instances, the cause of missing baseline chemistry measures is unknown. It is necessary to accommodate the missing data in developing SVM based classifiers.

In section 1.4, we discussed a number of published SVM with missing data methods. We noted that most of those solutions were ad-hoc and validated via simulation studies in very limited missing data scenarios. And those methods motivated by statistical reasoning are plagued by computational challenges or are limited to specialized missing data scenarios. Given the limitations of existing methods and motivated by the HCV-TARGET study, we propose the augmented and weighted support vector machine (AWSVM) classifier, an EM-motivated solution to the missing data problem for SVMs which maintains the convex objective function and which allows the researcher to use the same software as the complete case solution. The method is iterative and involves replacing incomplete observations in the training set with several draws from the EM conditional distribution. The observations in the training set are weighted so that the draws corresponding to a single incomplete observation contribute the same as a single, fully observed observation. We will show that in certain situations, the AWSVM asymptotically minimizes classification error, i.e., is a Bayes classifier. Further, we will show through simulation that the AWSVM has good finite sample properties as well. In contrast to existing approaches, the AWSVM is built on statistical principles which have been effective in other missing data methods.

In addition to advancing core biostatistical and machine learning methodology

through the development of the AWSVM method, we also make an important contribution to the field of hepatology. In particular, we apply the AWSVM method to the motivating HCV-TARGET study to develop a novel classification rule for determining whether or not treatment will be efficacious using baseline factors. The rule is sensitive and while further validations are required, it represents an important step towards the development of personalized treatment rules that can reduce the health risks and costs associated with treatment of hepatitis C.

2.2 Method

Our proposed method to accommodate incomplete observations is motivated by EM-principles as described in section 1.2.3. The focus of our analysis is the regularized empirical risk function which defines the SVM solution (section 1.4). Here, we make use of the weighted form of the objective function derived in Yang et al. [114]. With observation weights w_i, the weighted SVM objective function is

$$\hat{f}_{\mathcal{T}_n} = \arg\min_{f \in \mathcal{H}} \frac{1}{2}\|f\|^2_{\mathcal{H}} + \sum_{i}^{n} w_i \max(0,\, 1 - y_i f(\mathbf{x}_i)). \tag{2.1}$$

We use the same notation as chapter 1; recall that outcomes and covariates are denoted as y_i and $\mathbf{x}_i$ for the n observations of the training set $\mathcal{T}_n$. Building on the weighted objective function, we assume that the sampling model of $y_i|\mathbf{x}_i$ can be approximated by a quasi-probability model which is a function of the loss L_h and decision boundary f. We denote this quasi-probability model as $\tilde{p}(y|\mathbf{x},\, f)$. Second, we assume a distribution $p(\mathbf{x};\, \boldsymbol{\theta})$. Together, these assumptions imply a conditional quasi-probability model for the missing covariates $\mathbf{x}^m$ conditional on the observed data and parameters, i.e.,

$$\tilde{p}(\mathbf{x}_i^m|\mathbf{x}_i^o, y_i, f, \boldsymbol{\theta}) = \frac{\tilde{p}(y_i|\mathbf{x}_i, f)p(\mathbf{x}_i^m|\mathbf{x}_i^o, \boldsymbol{\theta})}{\int \tilde{p}(y_i|\mathbf{x}_i, f)p(\mathbf{x}_i^m|\mathbf{x}_i^o, \boldsymbol{\theta})\, d\mathbf{x}_i^m}. \tag{2.2}$$

The proposed method is to replace $\max[0, 1 - y_i f(\mathbf{x}_i)]$ in equation (2.1) with its conditional expectation $\tilde{E}\left\{\max[0, 1 - y_i f(\mathbf{x}_i)] \,|\, \mathbf{x}_i^o, y_i, f, \boldsymbol{\theta}\right\}$ for incomplete observations. Further, we propose that the expectation be replaced as a sum of finite draws:

$$\tilde{E}\left\{\max[0, 1 - y_i f(\mathbf{x}_i)] \,|\, \cdots\right\} = \frac{1}{r}\sum_{k=1}^{r} \max[0, 1 - y_i f(\mathbf{x}_i^{(k)})]$$

where $\mathbf{x}_i^{(k)}$ is $\mathbf{x}_i$ with missing values replaced with a draw from $\tilde{p}(\mathbf{x}_i^m | \mathbf{x}_i^o, y_i, f, \boldsymbol{\theta})$.

The key observation is that this setup essentially amounts to replacing each incomplete observation with r draws from the conditional distribution and then adjusting the corresponding weight for each observation. Thus the algorithm begins with a postulated rule, usually the complete case solution, and then proceeds between drawing replacement observations for missing values and constructing a new decision rule which is then used in the next step of drawing replacement values. The iterative solution is outlined in Table 2.1 for the standard case when each observation is initially weighted the same, i.e., $w_i = C/n$.

In many situations, sampling from the conditional distribution of $\tilde{p}(\mathbf{x}_i^m | \mathbf{x}_i^o, y_i, f, \boldsymbol{\theta})$ will require MCMC methods. As such, we describe a Metropolis-Hastings algorithm which can be used in the augment step which uses a normal proposal distribution. For an observation $\mathbf{x}$ with missing values at an arbitrary augment step $(t + 1)$ of the AWSVM algorithm, start the sampling chain with an initial, prior value of $\mathbf{x}_{(1)}$. Let $\mathbf{x}_{(1)}^o = \mathbf{x}^o$. Then generate a proposal value from the normal distribution centered at $\mathbf{x}_{(1)}$, call it $\mathbf{x}_{(1)}^p$. Again, ensure the observed elements of $\mathbf{x}$ match the corresponding elements in $\mathbf{x}_{(1)}^p$. (The proposal distribution my be tuned via the covariance if needed.) Let u be a uniform random variable, and let ϕ be the normal distribution function. If

$$\frac{\tilde{p}(y_i | \mathbf{x}_{(1)}^p, f^{(t)})\phi(\mathbf{x}_{(1)}^p; \theta^{(t)})}{\tilde{p}(y_i | \mathbf{x}_{(1)}, f^{(t)})\phi(\mathbf{x}_{(1)}; \theta^{(t)})} \geq u$$

then we set $\mathbf{x}_{(1)}^p$ as the second link in the chain, $\mathbf{x}_{(2)} = \mathbf{x}_{(1)}^p$. Otherwise, we set $\mathbf{x}_{(2)} = \mathbf{x}_{(1)}$.

Table 2.1: AWSVM Algorithm

Step	Procedure
0	Choose a distribution for $P(x;\boldsymbol{\theta})$, say normal. Let $f^{(0)}$ and $\theta^{(0)}$ be initial, starting values of f and $\boldsymbol{\theta}$.
1	Let t index the iterations. Start $t = 0$.
1a	(Augment) For each observation with missing values, construct r replicates (indexed by k) of $\mathbf{x}_i^{(k)}$ with draws from $\tilde{p}(\mathbf{x}_i^m \vert \mathbf{x}_i^o, y_i, f^{(t)}, \boldsymbol{\theta}^{(t)})$. Augment the data set with these draws, call it $\mathcal{T}_{aw}$.
1b	(Weight) For entries corresponding to complete cases, set the weight to C/n. For entries of incomplete cases, set the weight to $C/(rn)$.
1c	Find $\hat{f}_{\mathcal{T}_{aw}}(\mathbf{x})$ and weighted maximum likelihood estimates of $\boldsymbol{\theta}$. Set $f^{(t+1)} = \hat{f}_{\mathcal{T}_{aw}}(\mathbf{x})$ and $\boldsymbol{\theta}^{(t+1)} = \hat{\boldsymbol{\theta}}$
2	Repeat step 1 until $f^{(t+1)}$ converges.

The procedure continues with $\mathbf{x}_{(2)}$ acting as the prior value in order to generate $\mathbf{x}_{(3)}$, and so forth. The first 1000 values are discarded, and then every 20th draw is kept. The set of r values drawn from this procedure are the replicates $\mathbf{x}_i^{(k)}$ described in step 1a of the AWSVM algorithm reported in Table 2.1. These draws are appended to the training set with the corresponding weights set to $1/r$ of the complete case weight.

Users of AWSVM must select the cost parameter C and any kernel specific parameters just as they might in a situation without missing data. We propose two options: the first is standard cross-validation in which one uses the same proposed value of the tuning parameters at every iteration. The second is a cross validation step at the start of each iteration, thus allowing the tuning parameters to change. Depending on the computational burden of (a) generating draws from the quasi-conditional distribution and (b) cross-validation, one option may be more time-effective. If (b) is more time intensive, then the standard method is likely time-effective. Conversely, if

(a) is more time intensive, then allowing the parameters to change at each iteration is preferred. Our simulations have not identified either one to be better/worse in terms of prediction error.

2.2.1 Properties of the AWSVM

We frame the AWSVM in terms of a quasi-likelihood; it is in that context that we derive the method's properties. When weights are uniform each observation contributes

$$\frac{1}{C}\|f\|_H + \max(0, 1 - y_i f(\mathbf{x}_i))$$

to the penalized empirical risk. The loss function is such that properly classified observations contribute less while misclassified observations contribute more. We build the proposed quasi-conditional probability, $\tilde{p}(y_i|\mathbf{x}_i, f)$, with this risk contribution. Let D be a normalizing constant, and let $\Delta L(\mathbf{x}_i) = \max[0, 1 + f(\mathbf{x}_i)] - \max[0, 1 - f(\mathbf{x}_i)]$. We define the quasi-conditional probability model as

$$\begin{aligned}\tilde{p}(y_i|\mathbf{x}_i, f) &= D\exp\{-\text{empirical risk contribution}\} \\ &= \left[1 + \exp\left\{-y_i \Delta L(\mathbf{x}_i)\right\}\right]^{-1}.\end{aligned}$$

If $\mathbf{x}_i$ is on the boundary of the decision rule f, the induced conditional probability is $\frac{1}{2}$. Otherwise, $\tilde{p}(y_i|\mathbf{x}_i, f)$ is larger in regions of smaller loss, and conversely, is smaller in regions of larger loss.

The proposed method has important asymptotic properties which we state here and prove section A.1 (page 104) and section A.2 (page 108) of the appendix.

Proposition 1. *The AWSVM solution maximizes the observed data quasi-likelihood.*

Proposition 2. *If the data model $p(x;\theta)$ is specified correctly, then then decision function that maximizes the expected observed quasi likelihood is asymptotically a Bayes classifier.*

The first proposition and its proof indicate that the AWSVM algorithm does converge to a meaningful classifier. Similar to other EM methods, the AWSVM algorithm generates a rule which maximizes the observed data quasi-likelihood. The second proposition indicates that under certain conditions, the AWSVM solution is consistent in the sense that the AWSVM solution is a Bayes classifier.

2.3 Simulation Study

In this section, we compare the performance of the proposed AWSVM method to a set of commonly used competitors. We consider both linear and non-linear boundaries, both linear and Gaussian kernels, and two types of missingness models. For each data set, we built a decision rule with AWSVM and with the following competitors: Complete Case (CC), Mean Imputation (Imp), Multiple Imputation (MI), K-Nearest Neighbor (KNN), and Probability constraint (PC). For reference, we also built the Oracle (O) rule, the SVM classifier built with no missing data. The SVM competitors are described in section 1.4. We built the AWSVM decision rule using the complete case solution as the initial decision rule. We selected $p(\mathbf{x}; \boldsymbol{\theta})$ to be the normal distribution, and we sampled from $\tilde{p}(\mathbf{x}^m|\mathbf{x}^o, y, f^{(t)}, \boldsymbol{\theta}^{(t)})$ using the MCMC method described in section 2.2 with $r = 30$ draws.

2.3.1 Results

We start with simulations involving $d = 2$ predictors. There are two boundary models (rows) and two missingness models (columns). Thus, the results are reported in four panels in Figure 2.1, and each panel represents a different combination of boundary and missingness model. The top row is the linear boundary, and the bottom row is the non-linear boundary. The left column is missingness dependent on only predictors, and the right column is missingness dependent on predictors and the

outcome. The linear boundary (data in the top row) was generated with normally distributed predictors centered at 10, and the outcome was generated from the linear log odds model

$$\log \frac{p(y = +1|\mathbf{x})}{p(y = -1|\mathbf{x})} = \gamma \mathbb{1}_d^t (x - 10 \cdot \mathbb{1}_d).$$

The parameter γ was calibrated to achieve a consistent signal strength, specifically 15% error rate for the oracle classifier, and in the case of $d = 2$ predictors, $\gamma = 2.4$. The non-linear boundary (data in the bottom row) was generated with normally distributed predictors centered at 10, and the outcome was generated from a non-linear boundary,

$$\log \frac{p(y = +1|\mathbf{x})}{p(y = -1|\mathbf{x})} = \gamma \left(\delta + \mathbf{x}_d - \sum_{i=1}^{d-1} (\mathbf{x}_i - 10)^2 \right). \tag{2.3}$$

Similarly, γ calibrates signal strength of the missingness model and was set at 2.5. The parameter δ controls the proportion of observations in each group and was set for uniform group sizes at $\delta = 0.7$. Missingness dependent on only the predictors (left column) is generated as

$$\log \frac{p(\text{Missing } x_{ik}|x_{i,k+1})}{p(\text{Observed } x_{ik}|x_{i,k+1})} = \alpha + \beta(x_{i,k+1} - 10). \tag{2.4}$$

The parameter β controls the signal strength of the missingness model. The parameter α calibrates the overall missingness percentage. When $\beta = 0$, the missing data are MCAR. For values of β away from zero, the missing data are MAR. The simulations consider several values of β: -10, -6, -2, 0, 2, 6, and 10. Missingness dependent on predictors and the outcome (right column) was generated as

$$\log \frac{p(\text{Missing } x_{ik}|x_{i,k+1}, y_i)}{p(\text{Observed } x_{ik}|x_{i,k+1}, y_i)} = \alpha + \beta y_i (x_{i,k+1} - 10) \tag{2.5}$$

with the same interpretation and values for α and β. In each panel and for each level of β, we generated 100 datasets. In these simulations, a linear kernel is constructed

for the linear boundary scenarios and a Gaussian kernel is constructed for the non-linear boundary. While the true boundary is not known to the user, the boundaries in this simulation are sufficiently linear or non-linear that cross-validation would have indicated to the user the right kernel to use. The results are presented on that basis, but we revisit this point in a discussion of Figure 2.3.

In the simulations involving missingness dependent on predictors (left column), AWSVM performs better than the competitors. In some regions of β the improvement is 5% less prediction error. When missingness is dependent on the predictors and the the outcome (right column), the AWSVM performs better than competitors in some regions of β and performs comparably well in others. Further, in the right column, no method is clearly superior.

The results involving $d = 20$ predictors are reported in Figure 2.2. The boundary and missingness models are the same, except the signal strength parameters are calibrated to account for the additional predictors ($\gamma = 0.75$ and $\delta = 16.5$). We also performed simulations for $d = 100$ predictors, but those results are omitted because they closely mimic the $d = 20$ case. In the linear boundary scenario, regardless of missingness model, all the methods performed well except for complete case. This scenario represents less information loss than the $d = 2$ scenario because only 3.5% of predictor values are missing in this scenario compared to 35% in the $d = 2$ scenario. Thus, the fact that all methods perform well is to be expected. In the non-linear boundary scenarios, complete case and probability constraint are markedly poor performers. The probability constraint method is a linear solution and the poor performance is expected in the non-linear case. Interestingly, AWSVM performs 2-3% worse than the competitors in this situation.

In practice, the true decision boundary is not known. Further, decision rules built with the linear kernel can perform well even when the decision boundary is mildly non-linear, especially when the number of predictors is large. The non-linear

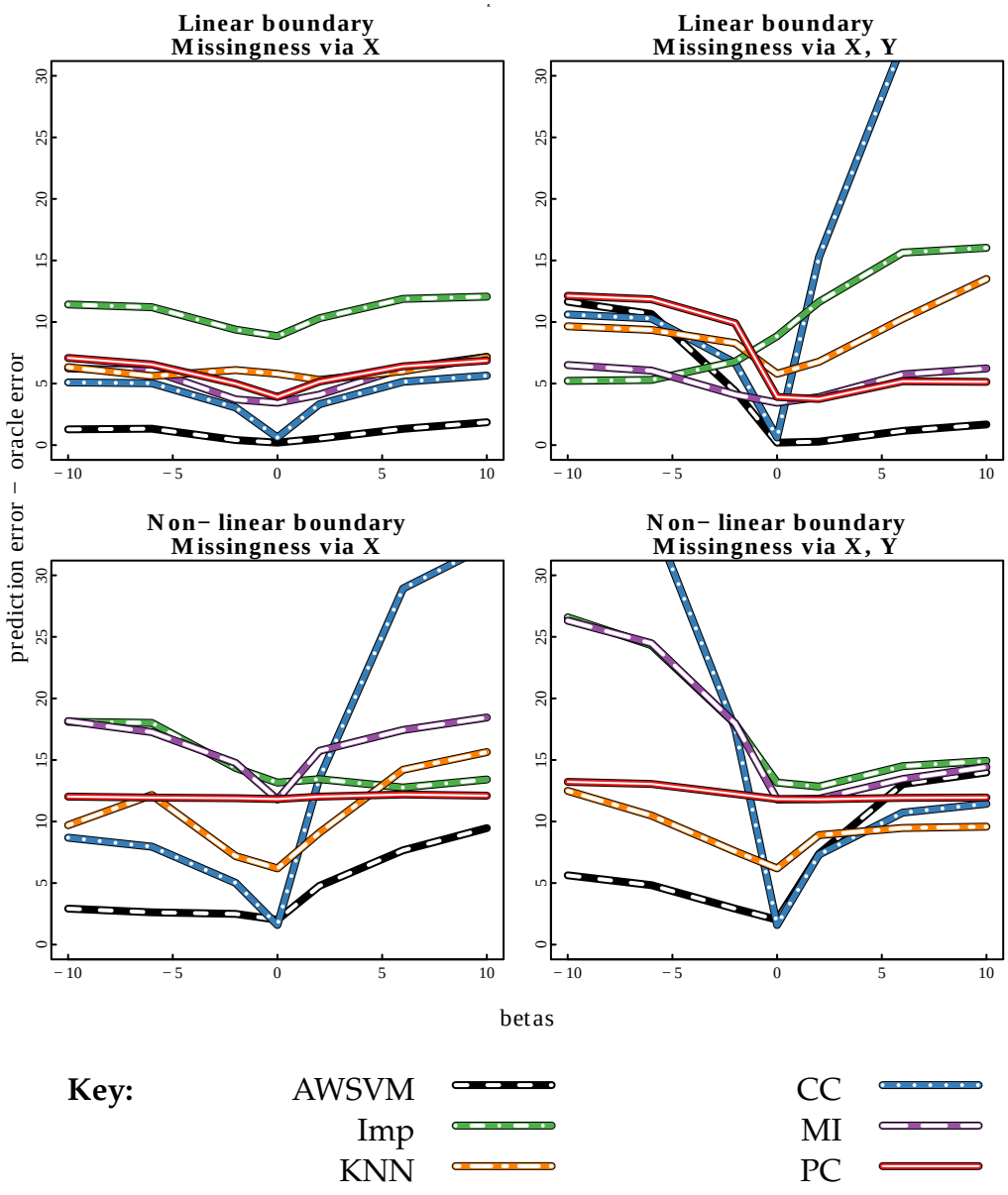


Figure 2.1: Comparison of AWSVM to competitors, simulation results for $d = 2$ predictors.

The X-axis is β in equations (2.5) and (2.4). The Y-axis is prediction error above oracle error. The missingness percentage is 70%.

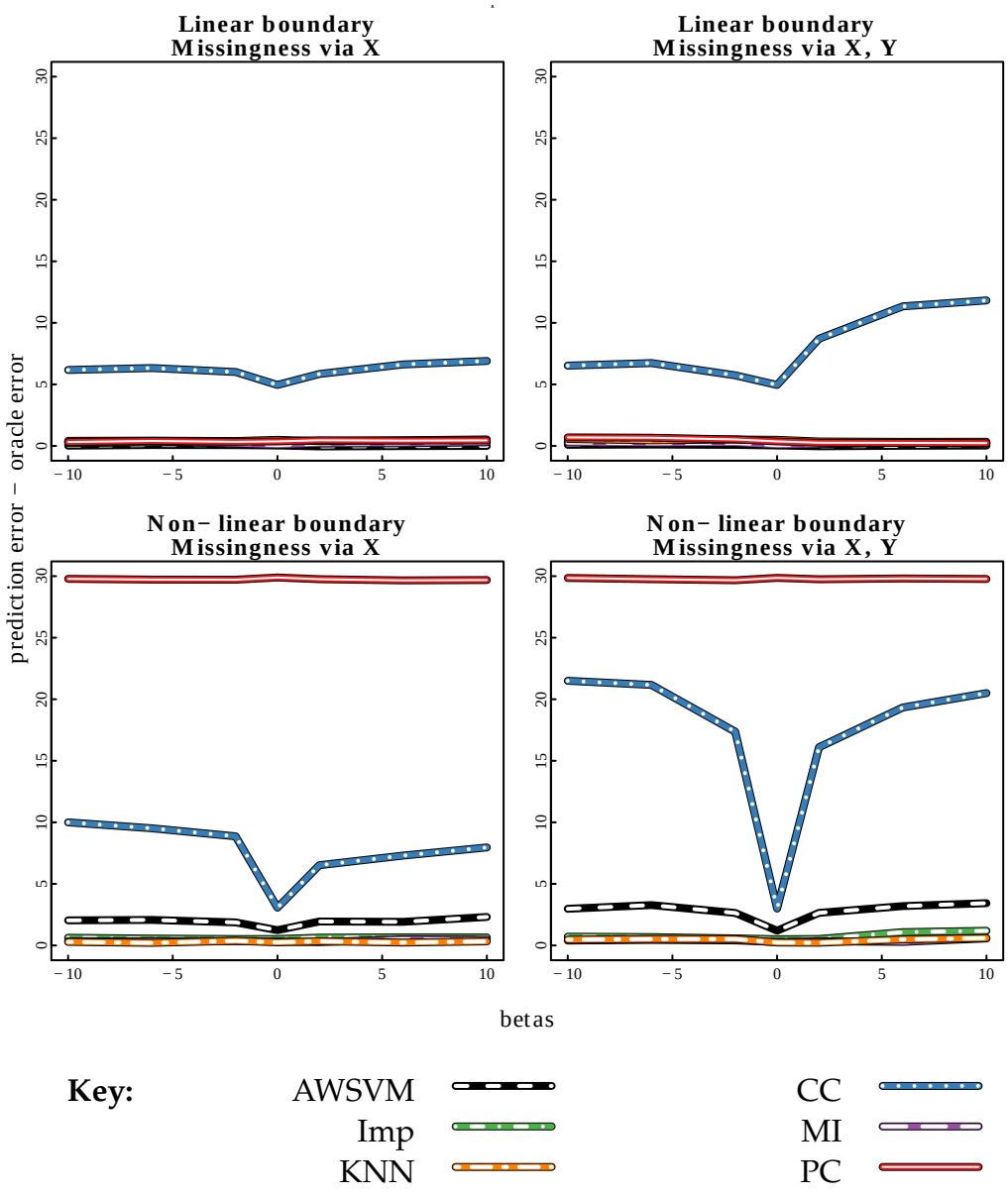


Figure 2.2: Comparison of AWSVM to competitors, simulation results for $d = 20$ predictors.

The X-axis is β in equations (2.5) and (2.4). The Y-axis is prediction error above oracle error. The missingness percentage is 70%.

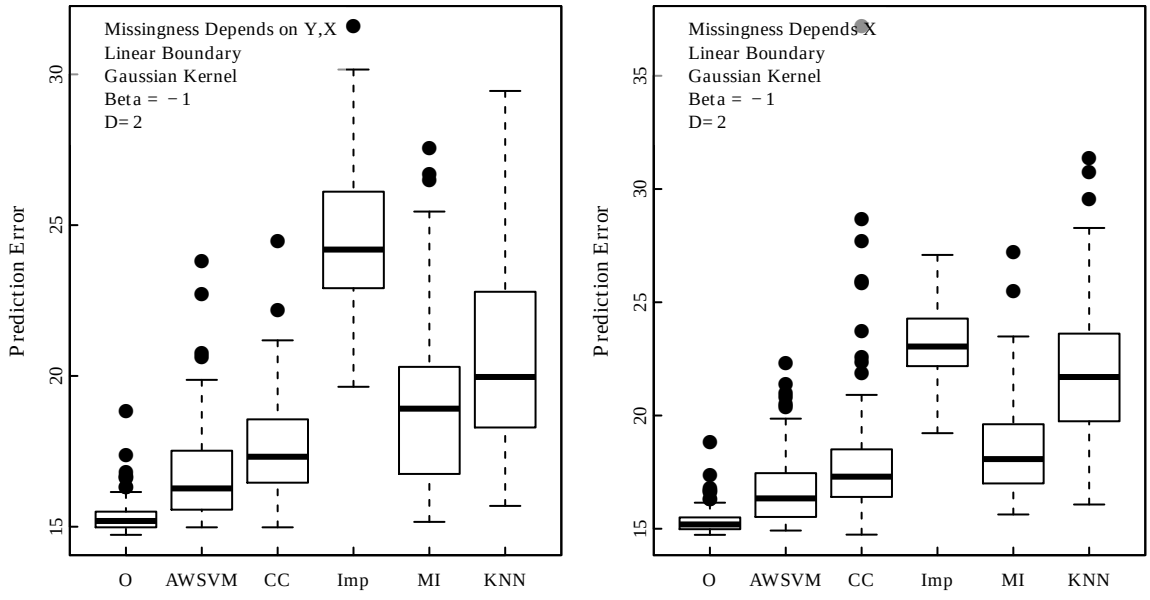


Figure 2.3: Prediction error when the boundary is linear and the decision rule is constructed with a Gaussian kernel

decision boundary used in this simulation is sufficiently non-linear so that linear rules trained with the non-linear data perform poorly. Conversely, rules constructed with the Gaussian kernel even when the boundary is linear did perform well. (See Figure 2.3.)

Figure 2.4 provides a graphical display of the variability of the simulation results. The simulation variability of AWSVM was better or comparable to its competitors in this and many other scenarios.

2.4 Application to HCV-TARGET Data

We apply the AWSVM method to a subset of patients from the HCV-TARGET database. HCV-TARGET is a consortium of North American academic and community medical centers performing a longitudinal observational study of patients undergoing treatment for hepatitis C. Specifically, we consider previously treated, non-cirrhotic, female patients treated with Telaprevir. One reason for choosing this subset is that its cure rate is lower than some of its counterparts.

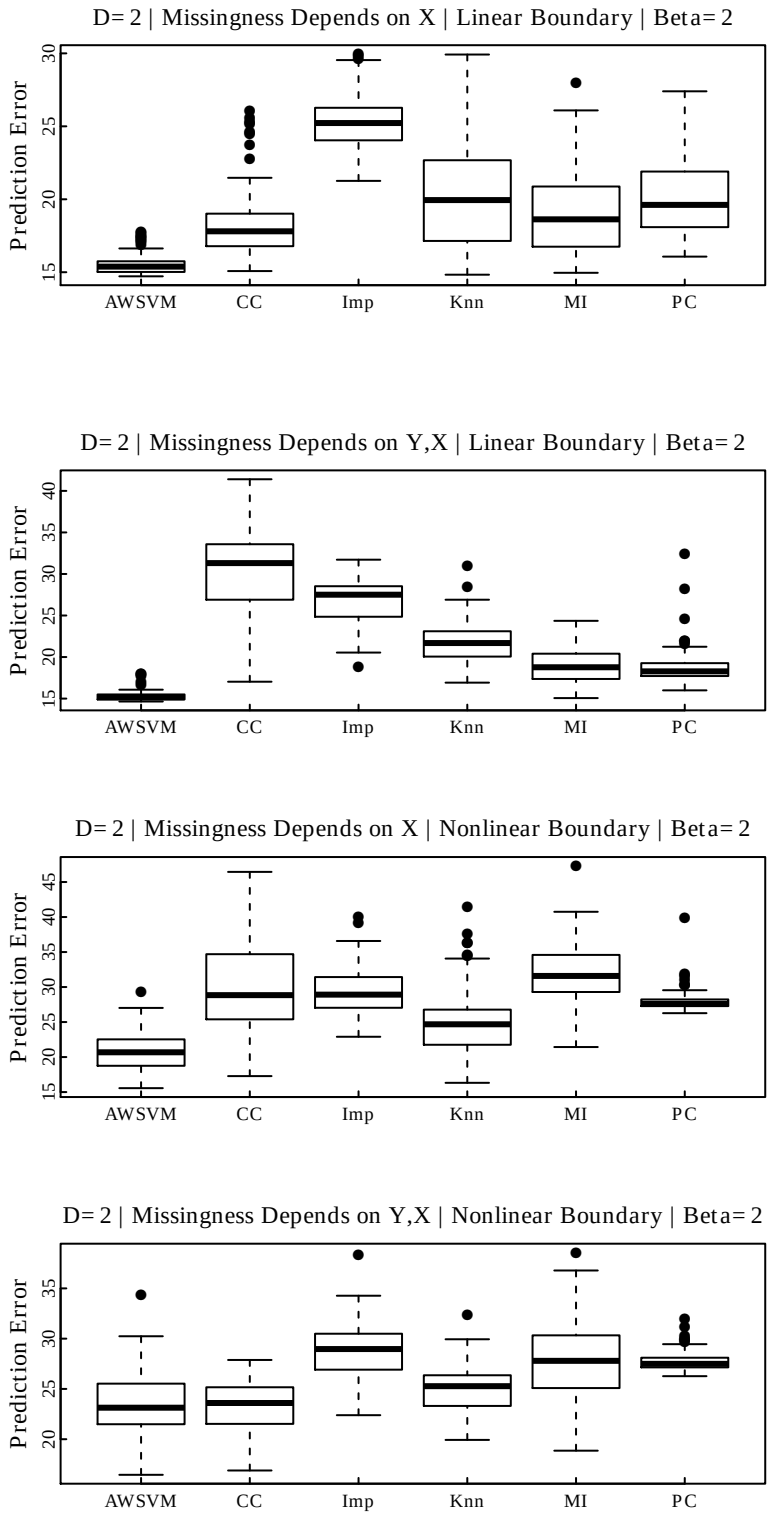


Figure 2.4: Comparison of Prediction Error Variability of Commonly Used Missing Data Methods

Cure, the outcome of interest, is defined as undetected hepatitis C virus 12 weeks after ending treatment. The training set (N=112) are those patients treated prior to 1 January 2012. The validation set (N=38) are those treated after. The predictors of interest include age and five blood assays at baseline: total bilirubin (BILI), creatinine (CRE), hemoglobin (HGB), hepatitis C viral load (LOGHCV), and absolute neutrophil count (ANC). We selected these predictors from the set of baseline measures in consultation with members of the HCV-TARGET team. In the training set, 37% of patients have incomplete data. Of those with incomplete data, 71% are missing one predictor, 20% are missing two predictors, and the remaining 9% are missing three predictors.

We applied the AWSVM method along with complete case, mean imputation, multiple imputation, *k*-nearest neighbor, and probability constraint methods to the construction of decision rules with linear and Gaussian kernels. For comparison purposes, we also considered the logistic regression with EM method with the other linear classifiers. The out-of-sample prediction error for each classification rule is reported in Table 2.2. There may be non-linear relationships between the predictors and the outcome; each kernel method performed better with the Gaussian kernel than the linear kernel. Our proposed method performs best among non-linear decision rules (prediction error 18.4%); the multiple imputation method was second best (23.7%). Interestingly, logistic regression with EM performed poorly in the linear case (50%) while the probability constraint method did well (31.5%). AWSVM was second best among linear methods (36.8%).

The out-of-sample performance of the AWSVM predictor indicates that personalized treatment-assignment rules may exist which identify patients likely to respond to treatment. The Gaussian AWSVM rule constructed with the HCV-TARGET data is reported in Figure 2.5. The pairwise plot of BILI and ANC, for example, is created by setting the value of all other predictors to the respective conditional mean. Only values within two standard deviations of the mean are plotted in order to restrict the

Table 2.2: Prediction Error of Competing Classification Methods Applied to HCV-TARGET Data

Method	Linear Kernel	Gaussian Kernel
AWSVM	36.84	18.42
Complete Case	44.74	31.58
KNN	44.74	31.58
Logistic Regression EM	50.00	
Mean Imputation	47.37	28.95
Multiple Imputation	39.47	23.68
Probability Constraint	31.58	

regions to areas in which training data are observed. While SVM decision rules are difficult to interpret in terms of a single predictor or even two predictors, the plots do demonstrate potentially important non-linear boundaries between outcome groups in the predictor variables. Note, for example, that a number of relationships indicate one group near the mean and the other group near both tails. AGE is one such predictor with this trend. Others like BILI tend to have a more linear-type relationship, where higher values of BILI do not favor cure. The Gaussian AWSVM rule admits both non-linear and linear relationships, and as this example demonstrates, such rules represent potentially significant cost savings and reduced risk of unhelpful exposure to toxic treatment.

2.5 Conclusion

We have proposed an SVM classifier for situations when the training data includes incomplete observations. We compared the AWSVM classifier to common competitors with simulated data; those simulations suggested that the AWSVM decision rule had fewer out-of-sample prediction errors in many situations. The simulations also showed that the advantages of AWSVM are largest when the number of predictors is small. In the analysis of hepatitis C data from HCV-TARGET, we used age, total bilirubin, creatinine, hemoglobin, hepatitis C viral load, and ANC to predict

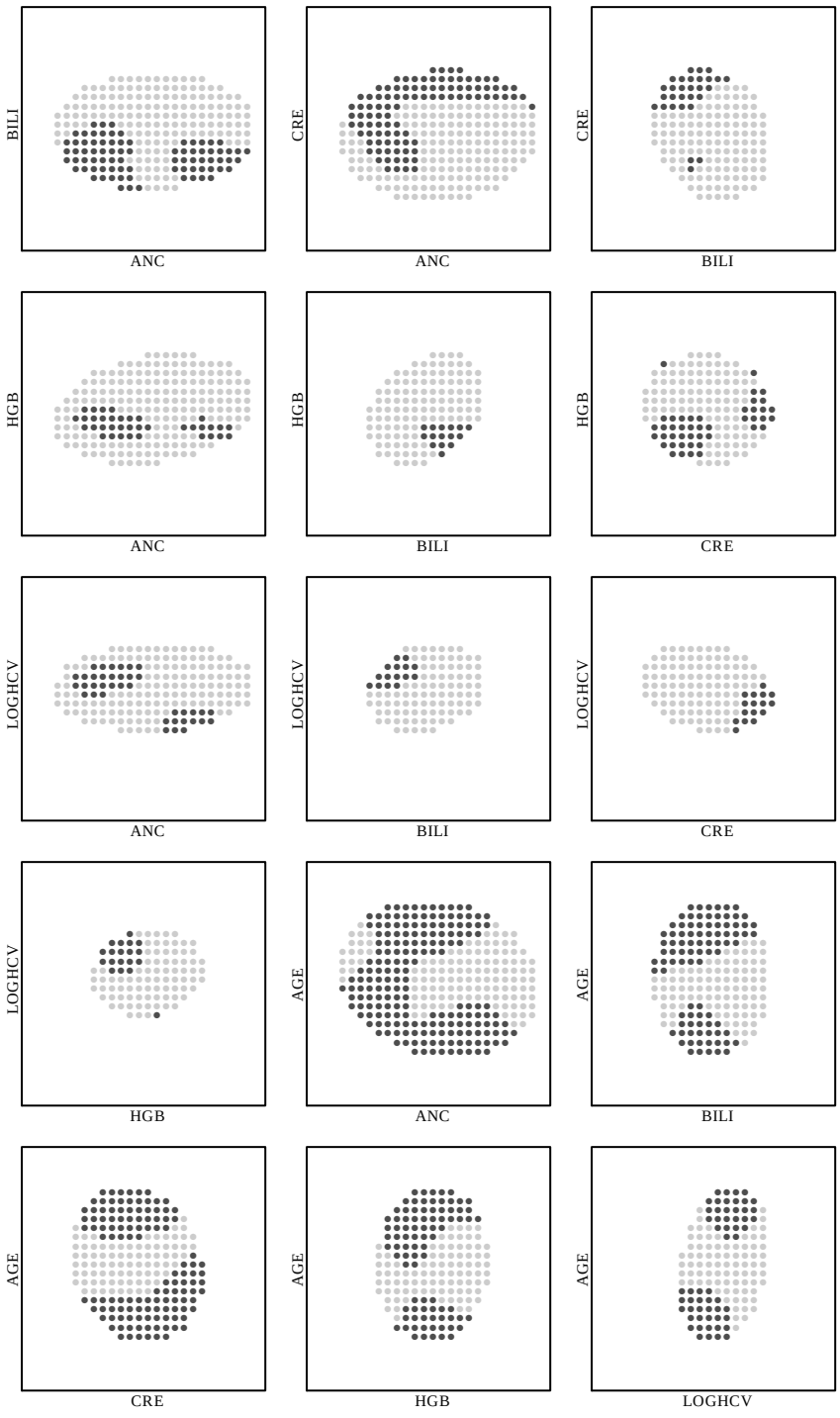


Figure 2.5: Plots of AWSVM decision rule constructed with HCV-TARGET data. Dark gray indicates cure. Values have been centered and scaled. The center of each axis is the mean. Each pairwise plot is created by setting all other predictor variables to the conditional mean.

cure among a specific subset of patients. The proposed method provided the lowest validation set prediction error.

There are several ways in which AWSVM can be modified or extended to fit specific data analyses. First, consider the replicates with which one augments the training data set. We proposed generating the replicates with an MCMC sampler. In our simulations, we drew 30 replicates for each missing observation. Depending on the size of the training set and the number of incomplete observations, the size of the augmented training set can become prohibitively large. One possible extension is to augment the training set with a smaller number of observations but weighted in a way that the expectation is reasonably preserved.

Second, consider the data model $p(X;\theta)$. In the current setup, the user must specify the data model. In our simulations and in the analysis of hepatitis C data, we choose a normal distribution. One area of future work is finding ways which relax this requirement.

The AWSVM provides a principled and usable solution to the problem of missing data and SVM classifiers. It does not require specialized software beyond that used for the non-missing data case. In the analysis of clinical data in HCV-TARGET database, where missing data in the training set is expected, the AWSVM classification rule provided a personalized treatment rule with smaller classification error than its competitors.

Acknowledgments

We thank Dr. Michael Fried and HCV-TARGET for providing the data analyzed in section 2.4.

CHAPTER 3: DOUBLY ROBUST SUPPORT VECTOR MACHINES FOR MISSING COVARIATES

3.1 Introduction

The Support Vector Machine (SVM) is a popular tool in a number of biomedical applications in which researchers must predict a binary outcome from a potentially large set of predictors. For example, SVMs were used in gene expression analysis [40], fMRI analysis [83], and other biomedical computer vision tasks. SVMs are successful in computer vision applications because the method can accommodate a large number of predictors and because the method constructs with relative ease both linear and non-linear models. Despite the method's success in applications of machine-generated biomedical data (e.g., fMRI data), SVMs have not experienced the same success in applications which regularly involve missing data because the SVM does not easily accommodate missing data. Because applications which generate datasets with missing data are prevalent, we propose a weighted SVM classifier which accommodates missing data in the covariates. The proposed SVM is based on ideas from weighted estimating equations and inverse probability weighting, and the classifier can be constructed with standard SVM software. Because this method can be widely implemented and eliminates missing data barriers, this method has the potential to expand SVM tools to a broader set of application areas.

One such application area which motivates this work is HCV-TARGET, an observational and longitudinal study of Hepatitis C patients. As is expected in such studies, researchers were unable to collect all baseline data for all patients. For example, baseline data like blood assay measures Albumin may not have been collected for some patients. Despite missing data issues, researchers intend to analyze the HCV-TARGET

database in order to identify patient sub-populations which respond differently to current treatment regimens. SVMs may be a valuable tool for this research task because the SVM offers considerable modeling flexibility. The method proposed in this paper accommodates the missing values in the covariates.

There is substantial literature related to missing data within parametric and semi-parametric modeling approaches [28, 85, 56, 90]. Notably, [89] introduced a general framework for understanding missing data and the scenarios when missing data can bias data analysis results. Only limited work has been done within the context of SVMs despite their importance for accommodating the complexity of many real data sets. One popular SVM method for missing data includes complete case analysis which operates by omitting observations with missing data and constructing a decision rule with only fully observed observations. The method can be seriously biased when missingness depends on the outcome and can be inefficient even when valid. Another popular missing data method with SVMs is single imputation [50]. The method replaces missing values with values from an imputation model (e.g. via regression or k-nearest neighbors). The method is simple but its validity rests on the quality of the imputation model. A poorly specified imputation model may introduce bias. Similarly, multiple imputation for SVMs, which requires imputing multiple data sets and averaging classification rules derived from each data set, relies on the imputation model. Because single and multiple imputation are accessible and widely used, there is some research about various imputation models and SVMs. For example, [2] encouraged the use of KNN imputation and [41] suggested Naive Bayes type imputation. In [35], the authors consider the Gaussian SVM with hot deck imputation, Naive Bayes, mean imputation, and regression-based imputation. Other imputation methods which are popular in parametric analyses but have not, to our knowledge, been considered with SVMs include Sequences of Regression Models (SRM) [82] and Multivariate Imputation by Chained Equations (MICE) [104].

Other methods for SVMs that are more sophisticated than complete case analysis and imputation include probability constraint techniques (PC) [95], methods which recast the SVM objective as estimation in the exponential family (EF) [97], and a geometric max-margin approach (MM) [22]. Although better motivated from a statistical standpoint, such methods require the use of nonstandard optimization techniques leading to computational challenges. These challenges are exacerbated by modest sample sizes and non-linearity, thereby losing one of the key advantages of using SVMs.

The method proposed in this paper is motivated by missing data methods known as doubly-robust estimators [8] which were developed in the context of parametric and semi-parametric models. The method combines the ideas of re-weighting and multiple imputation in order to correct for potential bias introduced by missing data [57, 61, 105, 88]. Furthermore, the objective function of doubly-robust estimators combines the contributions of imputation and re-weighting in such a way that if the imputation model or the re-weighting model are correctly specified, then the resulting estimator is unbiased. The method gets its name from the fact that only one of the two models needs to be correctly specified for the method to generate unbiased estimates.

In the context of SVMs, we apply re-weighting and imputation so that the empirical risk component of the objective function is unbiased. The proposed method is easily implemented with standard software. The method is developed in section 3.2 and a simulation study of these methods and their alternatives is reported in section 3.3. We compare the performance of the proposed method to its alternatives when applied the HCV-TARGET database in section 3.4, and we conclude in section 3.5.

3.2 Methods

3.2.1 Support Vector Machine Classifiers

Consider the task of constructing a classification rule $f_{\mathcal{T}_n}(\mathbf{x})$ from training set $\mathcal{T}_n$ to predict binary outcomes $y \in \{-1, +1\}$ from predictors $\mathbf{x} \in \mathbb{R}^d$. Specifically, construct the rule so that the predicted outcome is $\hat{y} = \text{sign}[f_{\mathcal{T}_n}(\mathbf{x})]$, which means an outcome is correctly predicted if $y f_{\mathcal{T}_n}(\mathbf{x}) > 0$ and is incorrectly predicted otherwise. The SVM classification rule built from $\mathcal{T}_n$ minimizes this complexity-penalized empirical risk function:

$$\hat{f}_{\mathcal{T}_n} = \arg\min_{f \in \mathcal{H}_{\mathcal{T}_n}} \underbrace{\lambda \|f\|^2_{\mathcal{H}_{\mathcal{T}_n}}}_{\text{penalty}} + \underbrace{\frac{1}{n}\sum_i^n \max[0,\, 1 - y_i f(\mathbf{x}_i)]}_{\text{empirical risk}}. \tag{3.1}$$

The restricted set of functions, denoted $\mathcal{H}$, is a Reproducing Kernel Hilbert Space of functions. Member functions of $\mathcal{H}$ are of the form

$$f_{\mathcal{T}_n}(\mathbf{z}) = b + \sum_{\mathbf{x}_i \in \mathcal{T}_n} c_i \kappa(\mathbf{z}, \mathbf{x}_i)$$

where κ is a kernel function. Thus, the SVM rule is a type of linear basis expansion model; the rule is a linear combination of basis functions $h_i(\mathbf{z}) = \kappa(\mathbf{z}, \mathbf{x}_i)$. The construction of $f_{\mathcal{T}_n}(\mathbf{x})$ centers on finding values for $c_1, \ldots, c_n$ and b.

With regards to the asymptotic properties of SVMs, two key results are that (a) the empirical risk converges to the hinge risk,

$$\frac{1}{n}\sum_i^n \max[0,\, 1 - y_i f(\mathbf{x}_i)] \to E_P\{\max[0,\, 1 - y f(\mathbf{x})]\},$$

and (b) the hinge risk minimizer also minimizes the classification risk. The combination of these statements implies that the SVM decision rule enjoys an important property: asymptotically, it is a Bayes classifier. See [98] for details and proofs.

3.2.2 Doubly Robust Support Vector Machine

Now consider the specific situation when the training set $\mathcal{T}_n = \{(\mathbf{y}_i, \mathbf{x}_i)|i = 1 \ldots n\}$ from which the classifier is to be built includes observations with some missing predictor values. Let $\mathbf{x}_i^m$ and $\mathbf{x}_i^o$ denote the components of $\mathbf{x}_i$ which are missing and observed, respectively. Also, let $\mathbf{x}_i^a$ denote the subset of predictors that are fully observed for all observations. To denote which observations include missing predictor values, construct the variable r_i. If an observation is missing a predictor value, let $r_i = 0$. Otherwise, let $r_i = 1$. The subset of observations with $r_i = 1$ are called the complete cases and are denoted as $\mathcal{T}_{cc}$ ($|\mathcal{T}_{cc}| = n_{cc}$). The compliment set are the incomplete cases and are denoted as $\mathcal{T}_{ic}$ ($|\mathcal{T}_{ic}| = n_{ic}$).

Incomplete data is problematic when the population of fully observed subjects (where $r_i = 1$) differs in important ways from the population of both fully observed and incompletely observed subjects, i.e., issues with missing data occur when $P(r = 1, y, \mathbf{x})$ differs from $P(y, \mathbf{x})$. In the context of SVMs built with fully observed subjects, the empirical risk portion of the objective function no longer approximates the hinge risk with respect to $P(y, \mathbf{x})$. Rather, the empirical risk portion approximates the hinge risk with respect to $P(r = 1, y, \mathbf{x})$. The motivating idea of our proposed method is to re-weight fully observed subjects so that the empirical risk approximates the risk with respect to $P(y, \mathbf{x})$.

As noted earlier, the concept of adjusting observation weights to preserve an expectation is used in survey sampling, causal inference, epidemiology, and several other areas of data analysis. Because the method proposed in this paper is based on the Doubly Robust estimator and combines the ideas of imputation and re-weighting, we call the method the Doubly Robust Support Vector Machine (DRSVM). As a preliminary step in developing the ideas of the DRSVM, suppose that $\hat{p}_i \in (0, 1)$ could be

selected so that

$$\frac{1}{n_{cc}} \sum_i \frac{r_i}{\hat{p}_i} \max[0,\, 1 - y_i f(\mathbf{x}_i)] \to E_{Y \times \mathbf{X}}\{\max[0,\, 1 - Yf(\mathbf{X})]\}$$

as n_{cc} gets large. Called inverse probability weighting in semi-parametric estimating equation frameworks, $\hat{p}_i$ is chosen to approximate $P(r_i = 1|\mathbf{x}_i,\, y_i)$. If the expectation is preserved, an SVM constructed with the weights $r_i/\hat{p}_i$ would be, asymptotically, a Bayes classifier. The draw back of stopping here is that only complete case observations contribute to the resulting classifier. As such, the efficiency of such an SVM may be poor.

We augment the inverse probability weighted empirical risk function with a surrogate loss function, $\phi(\mathbf{y}_i, \mathbf{x}_i^a)$, so that observations with missing values contribute to the empirical risk calculation. The surrogate loss function incorporates the multiply imputed values of the missing covariates, and the objective function for the DRSVM is

$$\begin{aligned} \hat{f}_{\text{DR}} = \arg\min_{f \in \mathcal{H}} \; & \lambda \|f\|_{\mathcal{H}}^2 \\ & + \frac{1}{n} \sum_i \frac{r_i}{\hat{p}_i} \max[0,\, 1 - y_i f(\mathbf{x}_i)] - \left(\frac{r_i}{\hat{p}_i} - 1 \right) \phi(y_i, \mathbf{x}_i^a). \end{aligned} \tag{3.2}$$

The advantage of such a setup is that the solution is asymptotically a Bayes classifier if one of the following holds:

1. The surrogate loss function captures the true loss, $\phi(y_i, \mathbf{x}_i^a) = E\{\max[0,\, 1 - y_i f(\mathbf{x}_i)]|y_i, \mathbf{x}_i^a\}$, for all observations.

2. The estimated probabilities $\hat{p}_i$ are unbiased in the sense that $\hat{p}_i = P(r_i = 1|\mathbf{x}_i,\, y_i)$.

The proofs of these propositions are relegated to section B.2 (page 111) in the appendix. The key advantage of this method compared to the commonly used imputation is that properly specified inverse probability weights protect against poorly specified imputation models. Likewise, this method compared to the inverse probability weights on just

the complete cases has the advantage of incorporating some information from incomplete cases. Lastly, this method highlights the key role of the empirical classification risk in the SVM framework; so long as the empirical classification risk approximates the true classification risk, the resulting SVM will be a Bayes classifier.

3.2.3 Computation of the DRSVM

In practice, the probabilities $\hat{p}_i$ are estimated by specifying a model for $r_i = 1|\mathbf{x}_i, y_i$. Logistic regression, classification trees, or even SVMs and other statistical learning classifiers can be used for this task. Whatever the tool, estimates of $\hat{p}_i$ which are relatively very small require extra attention as the corresponding observation may overly influence the estimate.

The surrogate loss, $\phi(y_i, \mathbf{x}_i^a)$, incorporates information from the observations with missing covariates. However, it is calculated for all observations, including those which are fully observed. Based on the subset of covariates that are observed for all observations, x^a, the surrogate loss is calculated via imputation. One chooses an imputation model conditional on $\mathbf{x}_i^a$ and y_i, and then samples the remaining covariates for each observation. Each observation is imputed K times. Denote filled-in draws as $\dot{\mathbf{x}}_i^{(k)}$. The surrogate loss is calculated as

$$\phi(y_i, \mathbf{x}_i^a) = \frac{1}{K} \sum_{k=1}^{K} \max[0, 1 - y_i f(\dot{\mathbf{x}}_i^{(k)})].$$

Operationally, however, the surrogate loss need not be computed directly. Rather, note that computation of the DRSVM relies on three types of observations: (a) covariates with imputed values from observations with missing values, (b) covariates with imputed values from observations without missing values, and (c) covariates with fully observed values. Consider a matrix $\overset{*}{X}$ in which all three types of covariates are stacked to form a single covariate matrix. Let $\overset{*}{y}$ record the corresponding outcomes for each

row in $\overset{*}{X}$. Suppose the vector v indicates the corresponding observation type for each row in $\overset{*}{X}$. And, lastly, depending on the observation type, let the vector w record the corresponding weight for each row in $\overset{*}{X}$. That is,

$$w_i = \begin{cases} \frac{1}{K} & \text{observation of type (a)} \\ \frac{1}{K}\left(1 - \frac{1}{\hat{p}_i}\right) & \text{observation of type (b)} \\ \frac{1}{\hat{p}_i} & \text{observation of type (c)} \end{cases}$$

The DRSVM objective function can be re-expressed with $\overset{*}{X}$, $\overset{*}{y}$ and w as

$$\hat{f}_{\text{DR}} = \arg\min_{f \in \mathcal{H}} \ \lambda \|f\|^2_{\mathcal{H}} + \frac{1}{n} \sum_i w_i \max[0,\, 1 - \overset{*}{y}_i f(\overset{*}{\mathbf{x}}_i)].$$

Computationally, this means the DRSVM can be expressed as a weighted SVM [114] with an augmented covariate matrix and outcome vector.

The DRSVM objective function is not convex. However, it can be approximated with convex functions in a way that allows users to compute the DRSVM with standard SVM software. The issue is that for observations of type (b), the corresponding weight, w_i, is negative. In section B.1 (page 110) of the appendix, we show that

$$\hat{f}_{\text{DR}} = \arg\min_{f \in \mathcal{H}} \ \lambda \|f\|^2_{\mathcal{H}} + \sum_i |w_i| \max[0,\, 1 - \text{sign}(w_i)\overset{*}{y}_i f(\overset{*}{\mathbf{x}}_i)]. \tag{3.3}$$

is an asymptotically equivalent solution which preserves convexity of the objective function. Because it is an approximate solution, situations involving heavily weighted observations of type (c) can lead to classifiers which poorly approximate the DRSVM classifier. However, such situations can be diagnosed in the following way. Starting with the most influential fully-observed observation, reconstruct $\overset{*}{X}$, $\overset{*}{y}$ and w as if the observation had missing covariates. Operationally, reconstructing the data simplifies to adjusting the corresponding values in w. For the potentially influential observation,

change the weights of observations of type (b) as if the observations were of type (a). Further, set the corresponding weight of the observation of type (c) to zero. With the new weights, re-fit the classifier. Large changes in the resulting classifier suggest that the heavily weighted observation led to a classifier which poorly approximated the DRSVM solution. If such is the case, the re-fit solution is preferred. If the re-fit classifier is similar to the first, one can continue to check subsequent influential observations.

3.2.4 Doubly Weighted SVM Classifier

Despite the complications of approximating the DRSVM, the motivating ideas behind DRSVM are appealing, specifically (a) weighting fully observed data to estimate a proper empirical risk and (b) increasing efficiency by incorporating incomplete observations via imputation and surrogate loss. In this section we propose a weighted SVM classifier motivated by the same ideas of DRSVM but which avoids the computational complications associated when approximating the DRSVM.

Recall $\mathcal{T}_{cc}$ ($|\mathcal{T}_{cc}| = n_{cc}$) is the set of complete case observations, and $\mathcal{T}_{ic}$ ($|\mathcal{T}_{ic}| = n_{ic}$) is the set of incomplete observations. The proposed empirical risk is

$$\eta\left[\frac{1}{n_{cc}}\sum\frac{r_i}{\hat{p}_i}\max[0,1-y_i f(\mathbf{x}_i)]\right]+(1-\eta)\left[\frac{1}{n_{ic}}\sum\frac{1-r_i}{1-\hat{p}_i}\phi(y_i,\mathbf{x}_i^o)\right].$$

The first part of the empirical risk mirrors the inverse probability weighted empirical risk and captures contributions from complete case observations. The second part is a weighted surrogate loss which captures contributions from incomplete cases. Moreover, the surrogate loss is defined in terms of $\mathbf{x}_i^o$ instead of $\mathbf{x}_i^a$, which means the imputations which make up the surrogate loss calculation incorporate more of the observed information. The parameter $\eta \in (0,1)$ calibrates the relative contribution of each part to the overall estimation. Because both complete and incomplete observations are reweighted, we call this the doubly weighted SVM (DWSVM).

DWSVM does not enjoy the doubly robust property and the surrogate loss may introduce bias similar to all improper imputation methods; however, DWSVM differs from other improper imputation methods because the analyst can control the relative importance of the surrogate loss and imputation model with the parameter η. It is an important contribution to SVMs and missing data research because it provides the analyst one way to gauge how sensitive an SVM model may be to a particular choice of imputation distribution. By calculating a classifier at several values of η, the researcher may examine how the classifier changes as more weight is given to the imputed portion. Further, the user may choose η via cross-validation as any other tuning parameter if the right level of η is in question.

Computation of the DWSVM involves similar steps of augmenting the dataset and computing inverse probability weights. By arguments similar to those in statement 2 of section B.2 (page 111) in the appendix, if the estimated probabilities, $\hat{p}_i$, are unbiased, then the DWSVM is asymptotically a Bayes classifier.

3.3 Simulation Study

3.3.1 Simulation Scenarios

The finite sample performance of the proposed estimators was evaluated through a series of simulation scenarios. The simulation study was organized as a factorial experiment in which we considered combinations of the following factors: (a) the distribution of the underlying covariate data, (b) the shape of the boundary between groups, (c) the number of predictors, and (d) the missing data model. We considered two distributions for the underlying data. In half of the scenarios, we generated d covariates from a multivariate random normal distribution with mean zero, unit variance, and pairwise correlation among covariates as 0.3. In the other half of scenarios, we generated d covariates from a χ^2 distribution with no pairwise correlation among covariates. We generated the outcome, y, in two ways. The first creates a linear

boundary between $y = +1$ and $y = -1$:

$$y \sim BIN[1, P(y = 1|\mathbf{x})] \text{ where } \log \frac{P(y = +1|\mathbf{x})}{P(y = -1|\mathbf{x})} = \gamma(\mathbb{1}_d^t \mathbf{x} + \delta). \tag{3.4}$$

The parameter γ is calibrated to achieve a consistent signal strength which we define as a 15% error rate for the oracle classifier. The parameter δ is calibrated to maintain consistent group proportions. We set the parameter so that $P(y = 1) = .5$. We also generated the outcome so that a non-linear boundary exists between groups, Specifically,

$$y \sim BIN[1, P(y = 1|\mathbf{x})] \text{ where } \log \frac{P(y = +1|\mathbf{x})}{P(y = -1|\mathbf{x})} = \gamma\,(\,\delta + x_d - \sum_{i=1}^{d-1} x_j^2\,). \tag{3.5}$$

Like the linear case, γ is set to achieve consistent signal strength, and δ is set to maintain consistent group proportions. The number of covariates, d, was set to 2 or 10.

Missing data was generated in two ways. In both missing data scenarios, half of the covariates (indexed as $k = 1, \ldots, d/2$) were eligible to be missing. In the first scenario, the probability of missingness is a function of both the outcome and covariates:

$$\log \frac{P(\text{Missing } x_{ik}|x_{i,d/2+k}, y_i)}{P(\text{Observed } x_{ik}|x_{i,d/2+k}, y_i)} = \alpha + \beta y_i x_{i,d/2+k}. \tag{3.6}$$

In the second scenario, the probability of missingness is a function of only the covariates:

$$\log \frac{P(\text{Missing } x_{ik}|x_{i,d/2+k})}{P(\text{Observed } x_{ik}|x_{i,d/2+k})} = \alpha + \beta x_{i,d/2+k}. \tag{3.7}$$

In both missing data models, the parameter β controls the missing-model signal strength. When $\beta = 0$, missingness does not depend on the covariates or the outcome. That scenario represents data missing completely at random (MCAR). As β deviates from 0, the association between missingness and the corresponding covariates (and outcome) is more pronounced. Scenarios with $\beta \neq 0$ represent data missing

at random (MAR). Note that negative and positive values of β represent considerably different missing data patterns, and methods for missing data can have non-symmetric performance along the range of βs. To evaluate the performance of the missing data methods, we consider β = -6, -2, 0, 2, 6. The parameter α in both missing data models controls the overall proportion of observations with missing values. This parameter is set so that 60% of observations have at least on missing covariate.

With 2 levels in each of the 4 experimental factors, there are 2^4 simulation scenarios in this simulation experiment. Within each scenario, the missingness signal is set to 5 levels, for a total of $2^4 \cdot 5$ simulation settings. In each of these settings, we generated 100 training data sets along with a single out-of-sample validation data set of 10,000 observations. Unlike the training set, the validation set did not have missing covariates. For each training set, we constructed classification rules with the methods considered in this study. The prediction error for each rule was calculated as the percentage of observations in the validation set which were missclassified.

We compared the performance of the proposed DRSVM and DWSVM methods to a set of commonly used competitors. For each data set, we built a decision rule with DRSVM, DWSVM, and with the following competing missing data methods: Complete Case (CC), Mean Imputation (Imp), Multiple Imputation (MI), K-Nearest Neighbor (KNN), and Probability constraint (PC). For reference, we also built the Oracle (O) rule, the SVM classifier built with no missing data. In each simulation setting, we constructed the SVM classifiers with both linear and Gaussian kernels (except the PC classifier).

The PC method [95], as noted earlier, takes advantage of the constrained optimization problem which characterizes computation of the SVM solution. Without

missing data, the constrained optimization problem is

$$f_{svm} = \arg\min_{f \in \mathcal{H}} \ \lambda \|f\|^2_{\mathcal{H}} + \frac{1}{n}\sum_{i}^{n} \xi_i$$

$$\text{Such that } 1 - y_i f(\mathbf{x}_i) \leq \xi_i \text{ and } \xi_i \geq 0 \text{ for } \ i = 1, \ldots, n$$

The PC method replaces the constraints involving missing covariates with

$$P(1 - y_i f(\mathbf{x}_i) \leq \xi_i | \mathbf{x}_i^o, y_i) \geq 1 - v_i \quad v_i \in (0, 1]; \ \ i = 1, \ldots, n.$$

The authors show that Chebyshev inequalities applied to the constraint allow the optimization problem to be recast as a second order cone programming problem. We construct linear classification rules with this method.

We constructed both DRSVM and DWSVM with $K = 5$ imputed draws to construct the surrogate loss. The imputations were drawn from a 10-nearest-neighbor imputation model. That is, replacement values for missing covariates were selected by drawing with replacement from the nearest 10 observations.

The cost parameter and the scale parameter (when using the Gaussian kernel) of each SVM method were selected with 2-fold cross validation. The grid of possible tuning parameter values was the sequence $\{2^{-15}, 2^{-14}, \ldots, 2^{14}, 2^{15}\}$. In the case of DWSVM, the parameter η, the relative weight of complete case observations to incomplete observations, was also selected with two-fold cross validation from the sequence $\{.5, .6, .7, .8, .9\}$.

3.3.2 Simulation results

Let oracle error be the out-of-sample prediction error achieved when the training set is fully observed. We define the above oracle prediction error (AOPE) as the out-of-sample prediction error minus the oracle error. The AOPE represents the loss in

accuracy due to missing data in the covariates. Tables B.1 – B.4 in section B.4 (page 117) in the appendix report the AOPE for each competing method in each of the simulation settings.

The number of predictors is an important factor in the performance of each of the missing data methods. With more covariates, the stability of the resulting classifier improves. Consider the inter quartile range (IQR) of AOPE within each simulation setting. With two covariates, mean imputation has the smallest median IQR across all settings at 2.4 percentage points. DRSVM generates the largest median IQR at 3.5 percentage points. Thus, all methods exhibit comparable variability in AOPE. With ten covariates, the median IQR drops for each method with the smallest median IQR at 1 percentage point (mean imputation) and the largest is 1.9 percentage point (DRSVM). Roughly speaking, each method experiences a 50% decrease in IQR between two and ten predictors. The complete case method, however, is the exception. Its median IQR remained essentially unchanged between two and ten predictors. For all but the complete case method, the improved stability as the number of covariates increases is expected. Note that in these simulations, the number of observations and number of observations with missing data remain constant while the number of covariates increases. A single missing covariate represents much greater information loss in the setting with two covariates than the setting with ten covariates. Continuing with this heuristic thinking, the stability of the complete case method does not improve because a single missing covariate represents the same percent information loss in the two covariate setting as the ten covariate setting because the entire observation is removed.

Along with stability, the number of predictors affects the overall accuracy of each of the methods. Averaging over all simulation settings, the median prediction error improved from 3.6 percentage points (in the case KNN imputation) to 0.6 percentage points (in the case of mean imputation) when increasing the number of predictors from

two to ten. The complete case solution is the only method which did not improve with more covariates.

Focusing on specific classifiers, the clearest (and expected) result from the simulation study is that complete case classifier only performs well in MCAR situations ($\beta = 0$). When missingness depends on the covariates and outcome ($\beta \neq 0$), the complete case has an average increase in prediction error of 5 percentage points compared to when missingness depends on the covariates but not the outcome. On average, the other classification methods did not perform differently under one missing data model than the other.

Considering the settings where kernel choice matches boundary type, most missing data methods performed better in the linear boundary settings. The difference was most pronounced with fewer predictors. One exception to this observation is the DRSVM classifier, which performed worse by about 1 percentage point in the non-linear settings even with 10 predictors.

The performance of some classifiers did not vary when the covariate distribution changed from normal to χ^2, when averaging over the other factors. However, complete case performed worse by 2.5 percentage points in the χ^2 setting. Similarly, the accuracy of DRSVM decreased by 1.5 percentage points when the underlying data was χ^2 while the performance of the other competitors, including DWSVM, stayed within 0.5 percentage points.

The simulation results highlight situations when the DRSVM performs well and situations when it does not. Note that in the first scenario of Table 3.1, the DRSVM performs best at all levels of β except for the extreme case when $\beta = -6$. The DRSVM is most beneficial in situations with few covariates and considerable missing data. The method seems to work much better with the linear kernel than with the Gaussian kernel. The DRSVM struggles with the Gaussian kernel in large part because of issues related to the approximate objective function, over-fitting, and tuning parameter

selection. Recall that the objective function, in order to maintain convexity, is approximated as in equation (3.3) in which the data has been reorganized into an augmented set of observations of types (a), (b), and (c) as described above. Operationally, the approximation re-labels observations of type (b). Thus each complete case observation contributes to the augmented data a single observation of type (c) and K observations of type (b). The weighted, average loss of those contributed observations is the loss contribution of the complete case observation to the standard (not augmented) objective function (3.2) so long as the rule classifies all of the contributed observations to the same class. The linear kernel will classify each contributed observation to the same class except for contributed observations close to the classifier boundary. The Gaussian kernel, because it can generate complex boundaries, can generate boundaries so that contributed observations in the augmented dataset from a single complete case are classified into different classes. In such situations, the loss contribution from the complete cases is poorly approximated by equation (3.3). The poor approximation issue is an over-fitting issue and can be remedied by selecting appropriate tuning parameters. On a dataset-by-dataset basis, the user can inspect the outcomes to ensure over-fitting does not occur. This simulation study highlights the need for an automated solution for this issue for situations when dataset-by-dataset review is not feasible. Potential ideas for an automated solution are discussed in the conclusions.

3.4 Application to HCV-TARGET Study

Hepatitis C is a common blood born infection which "affects about 2.35% of the worldwide population" [74]. The prevalence of hepatitis C infection coupled with the cost of treatment are motivating factors for developing classification rules that identify patients which are more likely to respond to treatment. Such rules could improve the cost effectiveness of treatment and lower the risk of unnecessary therapy. HCV-TARGET is a multi-center longitudinal observational study which enrolls a diverse

Table 3.1: Subset of Simulation Results (Full results are in the Appendix)

						Median AOPE [IQR]						
D	Missingness	X distr	Boundary	Kernel	β	cc	dr	dw	mi	knn	mi3	socp
2	Y and X	Normal	Linear	Linear	-6	13.0 [3.8]	2.8 [5.4]	6.7 [7.7]	13.3 [4.5]	12.2 [5.6]	2.3 [3.2]	14.2 [3.1]
					-2	5.7 [3.7]	0.6 [1.4]	1.4 [2.0]	7.0 [3.9]	9.4 [5.6]	1.8 [2.3]	10.8 [4.4]
					0	0.3 [0.9]	0.1 [0.6]	0.3 [0.7]	0.3 [0.7]	3.3 [2.9]	1.9 [2.0]	2.5 [2.9]
					2	12.4 [8.3]	0.5 [1.1]	1.7 [2.0]	3.5 [2.2]	5.3 [2.3]	3.2 [2.5]	3.6 [1.2]
					6	32.8 [7.1]	1.8 [2.3]	4.8 [3.4]	4.6 [2.2]	8.0 [2.4]	3.8 [3.4]	5.2 [1.1]
				Non-linear	-6	13.6 [3.6]	20.2 [10.4]	25.9 [6.9]	16.4 [5.1]	13.1 [6.6]	8.9 [10.6]	
					-2	6.4 [3.8]	8.6 [10.8]	20.2 [5.8]	7.8 [3.8]	9.0 [5.0]	3.6 [5.9]	
					0	0.6 [1.3]	1.6 [3.7]	12.9 [3.7]	0.7 [1.1]	4.2 [2.9]	1.8 [1.8]	
					2	13.1 [6.8]	2.4 [4.1]	8.9 [3.7]	5.2 [3.3]	5.9 [3.2]	4.0 [2.9]	
					6	31.7 [7.4]	5.1 [4.4]	9.4 [4.2]	8.4 [3.2]	8.7 [3.9]	9.2 [5.6]	
			Non-linear	Non-linear	-6	8.8 [1.9]	12.4 [6.9]	19.2 [5.8]	10.3 [3.5]	9.9 [2.9]	11.6 [5.6]	
					-2	5.1 [3.9]	6.4 [7.6]	15.7 [6.6]	6.9 [4.7]	7.7 [4.5]	10.1 [4.0]	
					0	0.9 [1.8]	2.4 [3.1]	11.3 [4.1]	2.6 [2.4]	4.7 [3.6]	10.4 [4.1]	
					2	11.2 [5.7]	2.8 [4.1]	8.3 [3.7]	7.6 [7.7]	5.5 [3.6]	13.3 [6.0]	
					6	26.4 [8.4]	3.7 [4.5]	7.7 [4.1]	17.1 [5.5]	6.4 [3.2]	16.4 [5.0]	
2	X	χ^2	Linear	Linear	-6	1.5 [3.7]	2.6 [4.5]	1.3 [2.9]	1.1 [1.9]	6.5 [5.7]	6.0 [5.3]	5.1 [2.6]
					-2	0.7 [2.2]	2.0 [3.0]	0.5 [1.6]	0.8 [2.7]	8.2 [3.4]	5.8 [4.2]	5.5 [2.7]
					0	0.6 [1.8]	3.4 [4.2]	0.7 [1.4]	1.2 [2.2]	7.9 [2.1]	3.5 [2.4]	6.4 [3.2]
					2	0.7 [2.2]	5.1 [4.9]	1.0 [1.5]	2.1 [2.9]	9.3 [2.9]	2.6 [2.9]	9.3 [3.6]
					6	4.1 [5.2]	4.1 [4.6]	2.5 [2.8]	7.4 [7.0]	10.2 [2.8]	2.8 [3.7]	11.1 [2.7]
				Non-linear	-6	1.7 [4.4]	13.6 [13.2]	15.2 [13.9]	1.5 [3.4]	4.2 [4.1]	5.9 [4.3]	
					-2	0.9 [2.3]	4.4 [12.4]	19.5 [7.8]	1.4 [2.8]	4.7 [2.8]	5.3 [3.8]	
					0	1.2 [1.9]	2.4 [3.5]	16.8 [6.9]	1.4 [2.1]	5.6 [3.0]	3.4 [2.7]	
					2	8.6 [11.3]	3.4 [3.8]	6.7 [6.4]	2.1 [2.4]	6.2 [3.1]	2.2 [3.4]	
					6	25.1 [22.9]	4.4 [4.1]	3.8 [3.6]	3.9 [6.1]	8.6 [5.0]	2.4 [3.6]	
			Non-linear	Non-linear	-6	5.3 [9.5]	16.3 [6.7]	14.9 [6.3]	6.7 [9.6]	8.6 [13.1]	12.0 [7.2]	
					-2	1.2 [2.4]	11.2 [9.7]	14.1 [5.8]	2.2 [3.7]	6.5 [5.4]	7.5 [4.1]	
					0	0.9 [1.5]	3.0 [4.1]	9.8 [4.6]	1.7 [1.4]	2.3 [2.7]	4.8 [3.3]	
					2	1.8 [2.6]	2.2 [2.5]	4.1 [4.0]	2.0 [1.8]	2.8 [2.6]	4.1 [3.9]	
					6	2.4 [2.9]	2.3 [2.5]	1.8 [3.2]	1.9 [2.2]	2.8 [2.5]	8.7 [6.4]	

population of patients receiving treatment for hepatitis C in order to assess efficacy and safety in non-clinical settings [45]. Researchers collected relevant demographic, clinical, and outcome data during treatment and follow-up. Because recently developed treatments will slowly gain adoption in areas outside the US and Europe, one objective is to use US and European data to develop a classifier that can accurately predict which patients will respond to earlier generation treatment.

We apply DRSVM, DWSVM, and commonly used competitors using a subset of patients from the HCV-TARGET database.

Cure, the outcome of interest, is defined as undetected hepatitis C virus 12 weeks after ending treatment. The training set (N=112) are those patients treated prior to 1 January 2012. The validation set (N=38) are those treated after. The predictors of interest include age and five blood assays at baseline: total bilirubin (BILI), creatinine (CRE), hemoglobin (HGB), hepatitis C viral load (LOGHCV), and absolute neutrophil count (ANC). We selected these predictors from the set of baseline measures in consultation with members of the HCV-TARGET team. In the training set, 37% of patients have incomplete data. Of those with incomplete data, 71% are missing one predictor, 20% are missing two predictors, and the remaining 9% are missing three predictors.

We constructed decision rules with linear and Gaussian kernels. The out-of-sample prediction error for each classification rule is reported in Table 3.2. There may be non-linear relationships between the predictors and the outcome; each kernel method performed better with the Gaussian kernel than the linear kernel. Our proposed DRSVM method performs best among non-linear decision rules (prediction error 20.4%); the DWSVM method was second best (21.1%). Interestingly, logistic regression with EM performed poorly in the linear case (50%) while the probability constraint method did well (31.5%). The out-of-sample performance of the DRSVM and DWSVM predictor indicates that personalized treatment-assignment rules may exist which identify patients likely to respond to treatment.

Table 3.2: Prediction Error of Competing Classification Methods Applied to HCV-TARGET Data

	Prediction Error (%)	
Method	Linear Kernel	Gaussian Kernel
DRSVM	34.21	20.42
DWSVM	44.78	21.05
Complete Case	44.74	31.58
KNN	44.74	31.58
Logistic Regression EM	50.00	
Mean Imputation	47.37	28.95
Multiple Imputation	39.47	23.68
Probability Constraint	31.58	

3.5 Conclusion

We have proposed an SVM classifier for situations when the training data includes incomplete observations. We compared the DRSVM classifier to common competitors with simulated data; those simulations suggested that the DRSVM decision rule had better performance when the number of predictors is small and a linear kernel is appropriate. In the analysis of hepatitis C data from HCV-TARGET, we used age, total bilirubin, creatinine, hemoglobin, hepatitis C viral load, and ANC to predict cure among a specific subset of patients. The proposed method provided the lowest validation set prediction error.

We are currently exploring ways in which DRSVM can be modified to avoid over-fitting so that the method works without user inspection when constructing classifiers with the Gaussian kernel. As noted earlier, the issue is that the approximate surrogate loss in equation (3.3) for complete case observations,

$$\phi(y_i, \mathbf{x}_i^a) = \left|1 - \frac{1}{\hat{p}_i}\right| \frac{1}{K} \sum_{k=1}^{K} \max[0, 1 + y_i f(\dot{\mathbf{x}}_i{}^{(k)})],$$

performs poorly if f is so flexible that the observed $\mathbf{x}_i$ are classified differently than its associated imputed values, $\dot{\mathbf{x}}_i^{(k)}$. In other words, the approximation is poor when

$\text{sign} f(\mathbf{x}_i) \neq \text{sign} f(\dot{\mathbf{x}}_i^{(k)})$. One modification of DRSVM to avoid over-fitting in the Gaussian setting is to limit the flexibility of f to classify complete case points differently than the associated imputed values. This limitation is achieved by restricting the set of solutions, $\mathcal{H}$, to functions of the form

$$f(\mathbf{z}) = b + \sum_{\mathbf{x}_i \in \mathcal{T}_{cc}} c_i \kappa(\mathbf{z}, \mathbf{x}_i),$$

where $\mathcal{T}_{cc}$ is the set of complete cases. Only complete case observations are potential support vectors. If $\mathcal{T}_a$ is the augmented data, then the proposed objective function is

$$\hat{f} = \arg \min_{f \in \mathcal{H}_{\mathcal{T}_{cc}}} \frac{1}{2} \|f\|^2_{\mathcal{H}_{\mathcal{T}_{cc}}} + \sum_{i \in \mathcal{T}_a} w_i \max[0, 1 - y_i f(\mathbf{x}_i)].$$

In the absence of other complete case observation, the classifier cannot generate localized regions where $\text{sign} f(\mathbf{x}_i) \neq \text{sign} f(\dot{\mathbf{x}}_i^{(k)})$ because the imputed values are not support vectors. The construction of the quadratic programming problem for this modification is reported in the appendix.

In this paper, we considered missing covariates, but the ideas that motivate the proposed method can be applied to situations when the outcome is missing or when both outcome and some covariates are missing. Both situations are avenues for future research. Furthermore, linear SVMs are often used in variable selection contexts when the number of predictors is large. The performance of the DRSVM in the variable selection setting is also an area of future study.

In conclusion, the DRSVM provides a usable solution to the problem of missing data and SVM classifiers. While some care is needed for Gaussian kernel classifiers, the method does not require specialized software beyond that used for the non-missing data case. In certain simulation scenarios, the method performed well, and in the analysis of clinical data in HCV-TARGET database, where missing data in the training set is expected, the DRSVM classification rule provided a personalized treatment rule

with smaller classification error than its competitors.

3.6 Acknowledgments

We thank Dr. Michael Fried and HCV-TARGET for providing the data analyzed in Section 3.4.

CHAPTER 4: SUPPORT VECTOR MACHINES FOR PARTIALLY OBSERVED OUTCOMES

4.1 Introduction

A common complication of liver disease is the development of varices in the gastrointestinal tract. Varices are large, swollen blood vessels leading to the liver that have a higher-than-normal likelihood of rupturing. Rupture of the varices is an adverse event which requires emergency medical attention because the mortality rate associated with variceal hemorrhage is high [59]. When treating diseases of the liver, such as Hepatitis C, researchers are often interested in the differential treatment response that may occur when treating patients with and without varices. Furthermore, in accessing the safety profile of treatments, another important question is differential rates of adverse events between patients with and without varices. However, health care providers do not always perform the procedure to identify varices. In the context of HCV-TARGET, an observational and longitudinal study of patients undergoing treatment for Hepatitis C, information regarding varices is missing from 62% of patient records. Because of the interest within HCV-TARGET regarding differential treatment effects and differential safety profiles between patients with and without varices, researchers want to develop classification rules which utilize baseline covariate data in order to predict if a patient has varices.

In this paper, we propose to apply missing data methods to SVMs in order to accommodate missing outcome data. We propose a method which works for two-class and multi-class SVMs and which works for any type of kernel. Our proposed method is based on EM principles which we can extend to settings which go beyond the standard missing-class-information problem to include a setting when some information about

class labels is known. For example, we develop a multi-class SVM which incorporates information that a training observation is in class 2 or class 3 but not in class 1. Thus, the method which we propose will be applicable to standard missing-class-information settings but also to settings when class information is partially observed. In the remainder of the introduction, we describe the literature related to two-class SVMs, multi-class SVMs, and the issue of missing class information which is commonly called called semi-supervised learning. In section 4.2, we describe our proposed method. In section 4.3, we demonstrate the performance of our method in a simulation study. An important contribution of this work is to demonstrate that in many situations, the complete case solution performs very well. We apply the method to HCV-TARGET data in section 4.4.1. In the final section, we discuss extensions for this work.

4.1.1 Two-class and Multi-class Support Vector Machines

The Support Vector Machine (SVM) introduced in [25, 106] is a method for binary classification which has been successfully implemented in a number of biomedical applications such as gene expression analysis [40], fMRI analysis [83], and other biomedical computer vision tasks. The appeal of SVMs stems from the method's capability to (a) model non-linear relationships between covariates and the outcome classes, and (b) generate a solution when the number of covariates exceed the number of observations. Further, the method differs from other popular classification methods like logistic regression, LASSO, trees, and random forests because the method does not make an assumption about or attempt to estimate the distribution of the outcome class conditional on the covariates. Rather, the SVM approach works to approximate the optimal boundary between classes. As the number of observations increases, the SVM classifier achieves the lowest possible expected classification error, regardless of the underlying distribution of outcomes and covariates [94]. Because of its success as a binary classifier, a number of researchers have adapted the SVM to multi-class

settings.

There are two families of multi-class SVMs. The first, which we call the composite-of-binary family, constructs a set of two-class SVM rules from which a single multi-class rule is constructed. Procedures of this type are widely used in many classification settings and are not specific to SVMs [38, 49]. The three most popular composite-of-binary SVMs are one-versus-one SVM, the one-versus-many SVM, and the one-versus-one DAGSVM [81]. In a K-class setting, the one-versus-one multi-class SVM is constructed by generating all $K(K-1)/2$ binary rules that separate generic class i from generic class j. To classify a point, the covariate vector is input to each binary classifier which outputs a vote for one of two classes. The overall multi-class rule outputs the class which receives the most votes. The one-versus-one DAGSVM is a similar alternative in which classification is based on a decision tree with a one-versus-one classifier at each node. The decision tree is constructed so that one potential class is eliminated at each decision node, and the terminal node represents the class prediction. Lastly, we describe the the one-versus-many classifier which generates K binary rules which separate generic class i from all other classes. To classify a point, the covariate vector is input to each two-class rule; the final prediction corresponds to the class which generated the largest signed distance, i.e., $\hat{y} = \arg\max_{i=1,\ldots,K} f_i(\mathbf{x})$. Variations of the these composite-of-binary SVMs exist, and generally vary by how each component classifier casts a vote for a potential class. For example, using the component SVMs to generate probability estimates which in turn are combined in a final decision rule. See [49, 80, 20, 110] for a discussion of generating and combining multi-class probabilities from SVMs.

Because of their operational efficiency, versions of the composite-of-binary multi-class SVM are implemented widely in statistical software packages [20]. The draw back to this type of multi-class SVM is that voting does not always generate a clear winning class. For example, a one-versus-one multi-class rule in a $K = 3$ class

setting is built on 3 two-class rules. For some covariate combinations, the 3 two-class rules may generate a vote for each class. The same scenario can occur with one-versus-all multi-class rules as well. The primary advantage of the DAGSVM is that it avoids the ambiguities like ties that can arise in the standard voting scheme. Beyond the issue of ties is the issue of consistency, and the author in [68] shows examples when one-versus-all is not Fisher consistent. Despite this issue, the composite-of-binary approach continues to be a popular multi-class SVM method. While members of the composite-of-binary family of SVMs are similar in performance and popularity, in a comparison of the composite-of-binary multi-class rules reported in [32], the authors concluded that probability based multi-class SVMs performed best, especially with sparse training data. Before that report, the authors in [84] demonstrated with a number of real-life datasets the desirable performance of the one-versus-all SVM with proper tuning.

The second type of multi-class SVM builds a decision rule by simultaneously training K functions where each function corresponds to a single class [107, 26, 62, 69]. Distinguished by specific multi-class loss functions, the various flavors of multi-class SVMs simultaneously construct the K functions so that the predicted class corresponds to the function with the largest value. The simultaneous estimation of the K functions ensures that the estimated multi-class rule targets the multi-class Bayes rule. We provide greater detail of the simultaneous, direct multi-class SVM in section 4.2

4.1.2 Missing class labels

Examples of collected data in which some class labels are missing are increasingly common. Any situation where determining the class label is expensive or overly invasive can generate data with fully-observed covariates but few class labels. For example, data for which class labels must be assigned by a human expert, as with immunohistochemistry data, or data for which disease subtype must be ascertained

by biopsy or expensive blood-assay. Often, the research goals associated with these types of datasets is to construct a classifier which predicts class labels because the gold-standard method is inaccessible. Classifiers built with some missing class labels are often called semi-supervised classification rules [87].

There are existing methods for constructing semi-supervised two-class or multi-class rules in statistical learning contexts, many of which are based on missing data ideas like imputation and the expectation-maximization (EM) algorithm while other methods are based on statistical learning ideas like boosting. For example, [42] applied EM as a way to estimate Gaussian mixture models in an unsupervised, clustering context and to missing data in a supervised context. In [76], researchers perform semi-supervised text classification with a naive Bayes classifier and the EM algorithm. Imputation type semi-supervised classification include co-training [14] and ASSEMBLE [12]. In [103], authors provide a general purpose semi-supervised algorithm for any multi-class classifier based on boosting.

One semi-supervised method specific to two-class SVMs is the method introduced in [11] called S^3VM. While standard SVM methodology chooses the rule which minimizes the empirical hinge risk, S^3VM chooses the rule which minimizes the empirical hinge risk calculated as if unlabeled points are in fact correctly labeled. Because unlabeled points are treated as correctly classified, the unlabeled points in the empirical hinge risk act as a 'high density' penalty. That is to say, the S^3VM setup prefers rules with boundaries in low density regions. Computationally, the S^3VM objective function is not convex and has potentially many local minima; however, there are a number of algorithms developed to find the S^3VM solution. See [118, 117, 21] for a discussion of the various non-convex optimization techniques which have been proposed for constructing S^3VM rules.

Multi-class extensions of S^3VM were proposed in [113] for specific types of kernels (sparse and full rank) with a specific multi-class loss function. Using a sparse

Laplacian kernel matrix (a kernel to which the method applies), the algorithm employs gradient descent to find a solution. In contrast to the limited setting of sparse and full rank kernels, a more accessible multi-class extensions of S^3VM have been developed with the composite-of-binary multi-class SVM, such as the methods proposed first in [21] and adapted in [6]. In [21], the composite-of-binary multi-class SVM applies S^3VM methods to each of the two-class rules. The method requires all unlabeled points to be included when training each two-class rule, even with a one-versus-one rule. In [6], authors adapted the composite-of-binary S^3VM method of [21] by introducing participation weights of unlabeled observations. Based on a generic similarity measure, the participation weights will up-weight or down-weight the contribution of unlabeled observations when training individual two-class S^3VMs as part of the larger composite-of-binary rule. Thus, unlabeled observations which are similar to class 1 observations are given greater weight when constructing rules involving class 1. Likewise, observations not similar to class 1 are given less weight. As demonstrated in a number of examples, using participation weights in the construction of semi-supervised composite-of-binary multi-class SVMs appears to improve classifier performance.

Another approach to semi-supervised SVMs was introduced in [112] in the context of clustering with SVMs with a method known as maximum margin clustering [111]. The objective of maximum margin clustering is to find a set of class labels which minimizes the SVM empirical risk under the constraint that the class balance in the proposed solution is within prescribed bounds. Once the optimal set of class labels is found, an SVM classifier is constructed from a training set composed of the new class labels and covariates. The key extension of max margin clustering to semi-supervised learning is that the max margin clustering objective function can incorporate information from observations with known outcomes by introducing constraints. Often called the semi-definite SVM (SD-SVM) because the objective function is a semi-definite pro-

graming problem, the SD-SVM is similar to S^3VM in the two-class setting but has the additional advantage of also providing a multi-class solution. However, the SD-SVM as a semi-supervised method is limited in application by the computationally expensive algorithm, its sensitivity to tuning parameters, and its assumption that the intercept term (sometimes called the bias term) is zero [102]. While [102, 116] proposed unsupervised clustering algorithms based on SD-SVM which minimize these drawbacks, the resulting methods do not admit a semi-supervised solution.

The SD-SVM can be framed as a two-step procedure in which the first step clusters the data and the second step constructs an SVM based on the predicted class labels. A cluster-then-construct solution is one in which any clustering procedure provides class labels in the first step and an SVM is constructed in the second step. SD-SVM differs from this ad-hoc procedure because the clustering criteria minimized in the first step is the SVM empirical risk. Note that the semi-supervised composite-of-binary multi-class SVM from [6] which is described above is another flavor of the cluster-then-construct type solution. The first step generates fuzzy-cluster labels in the form of participation weights and the second step constructs a composite-of-binary, one-versus-one S^3VM rule. Because this method relies on a composite-of-binary SVM, its computational burden is much less than SD-SVM.

As evidenced by the several types of semi-supervised SVMs in both the two-class and multi-class settings, there is considerable interest in developing algorithms which efficiently generate semi-supervised SVM classifiers. Despite the interest in the topic, the value of semi-supervised multi-class SVMs over complete-case multi-class SVMs has been questioned, as a number of authors have provided examples where complete-case multi-class SVMs out perform semi-supervised multi-class SVMs, such as [120] and [63]. Further, the computation required for many semi-supervised methods can be substantial, though recent algorithms such as those described in [96] improve on earlier algorithms for two-class SVMs. In the multi-class setting, however,

computation time may still be prohibitively expensive, such as the SD-SVM solution which requires estimation of $n + n_u^2$ parameters where n is the number of observations and n_u is the number of unlabeled observations. Multi-class methods which are not computationally expensive like the multi-class S^{3}VM methods are restricted to specific application areas (sparse, full rank kernels) or rely on composite-of-many SVMs. In this paper, we propose a method that is computationally feasible for large numbers of observations, is applicable to all application areas, and is applicable to both two-class and multi-class settings (particularly composite-of-many SVMs). The method proposed in this paper is an EM-type algorithm, and the method's primary advantage is that it can easily be computed with standard SVM software. Further, our proposed EM solution is particularly helpful when some information about the class label is known. It is this setting where the proposed method has the potential to be valuable, hence our focus on the setting in the simulation scenarios.

4.2 Methods

4.2.1 The Reinforced Multi-class Support Vector Machine

While the method applies to two-class and multi-class SVMs, we develop the method in the multi-class context. Consider a training set, $\mathcal{T}_n$, of n observations, each consisting of a d-vector of covariates, $\mathbf{x} \in \mathbb{R}^d$, and multi-class outcome, $y \in \{1, \ldots, k\}$. Each observation is an iid draw from an unknown distribution $P(\mathbf{x}, y)$. Consider functions $\mathbf{f}(\mathbf{x}) = \{f_1(\mathbf{x}), \ldots, f_k(\mathbf{x})\}$ so that the class label of $\mathbf{x}$ can be predicted as

$$\hat{y} = \arg\max_i f_i(\mathbf{x}).$$

The classifier which minimizes the average classification error over $P(\mathbf{x}, y)$ is the Bayes classifier,

$$\mathbf{f}^{bayes} = \arg\min_{\mathbf{f}} E_P[y \neq \arg\max_i f_i(\mathbf{x})].$$

The average classification error of the Bayes classifier is called the Bayes risk. The goal is to construct a classifier from the training set $\mathcal{T}_n$ which is asymptotically a Bayes classifier but also performs well in finite sample situations.

The SVM solution frames the task within empirical risk minimization; specifically, the SVM solution is

$$\hat{\mathbf{f}} = \arg\min_{\mathbf{f}\in\mathcal{H}} \lambda\|\mathbf{f}\|^2_{\mathcal{H}} + \frac{1}{n}\sum_{i=1}^{n} L[y_i, \mathbf{f}(\mathbf{x}_i)] \tag{4.1}$$

$$\text{such that} \qquad \sum_{i}^{k} f_i(\mathbf{x}) = 0$$

where $\|\mathbf{f}\|^2_{\mathcal{H}} = \sum_i^k \|f_i\|^2_{\mathcal{H}}$ and $L[y_i, \mathbf{f}(\mathbf{x}_i)]$ is a loss function which penalizes misclassification. The multi-class SVM methods proposed in this paper builds on the reinforced multi-class SVM (RMSVM) proposed in [69] because it provides a multi-class loss function which unifies the earlier work of [62] and [107] as special cases of a general multi-class loss function. The RMSVM loss function is

$$L[y, \mathbf{f}(\mathbf{x})] = \gamma[(k-1) - f_y(\mathbf{x})]_+ + (1-\gamma)\sum_{j\neq y}[1 + f_j(\mathbf{x})]_+$$

where the function $[t]_+ = \max\{0, t\}$ and γ is a tuning parameter which calibrates the loss. The set of solutions, $\mathcal{H}$, is constructed so that each component of solution, $\hat{\mathbf{f}}$, is of the form

$$f_i(\mathbf{x}) = b + \sum_{j=1}^{n} c_j K(\mathbf{x}, \mathbf{x}_j) \qquad \mathbf{x}_j \in \mathcal{T}_n$$

where $K(\mathbf{u}, \mathbf{v})$ is a kernel function. The linear kernel, $K(\mathbf{u}, \mathbf{v}) = \mathbf{u}^t\mathbf{v}$, and the Gaussian kernel, $K(\mathbf{u}, \mathbf{v}) = \exp\{-\sigma\|\mathbf{u} - \mathbf{v}\|^2\}$, are commonly used.

As the solution to the empirical risk minimization problem, the SVM targets the conditional expectation of the loss function, $E\{\, L[y, \mathbf{f}(\mathbf{x})] \,|\mathbf{x}\}$. When $\gamma \leq 1/2$, the RMSVM solution is Fisher consistent in the sense that the minimizer of $E\{L[y, \mathbf{f}(\mathbf{x})|\mathbf{x}]\}$ is

also the multi-class Bayes rule [69]. Further, because simulation examples in [69] show that the RMSVM performs better when $\gamma = 1/2$ than when $\gamma = 0$ or 1, we proceed with γ always 1/2, thus, ensuring a Fisher consistent rule.

4.2.2 Proposed Method for Missing Class Labels

Our proposed method for handling missing class labels is based on EM principles described in [28] which have been successfully applied in other missing data situations in both likelihood based contexts [66] and statistical learning contexts [42]. In the likelihood context, the EM algorithm is an iterative method for finding maximum likelihood (ML) estimates when the likelihood includes missing data. The algorithm begins with an initial value of the likelihood parameter, say $\theta = \theta^{(0)}$. Then, each iteration of the EM algorithm involves two steps which generate an update of the estimate of θ. In the first step, also called the expectation- or E-step, one replaces the log-likelihood with its expectation conditional on the covariates and the current value of θ. Thus, in the r^{th} iteration, if the log-likelihood is $\ell(\theta) = \sum_i^n \log[p(y_i|\mathbf{x}_i, \theta)]$, then

$$Q(\theta|\theta^{(r-1)}) = \sum_i^n E\{\, \ell(\theta) \,|\, \mathbf{x}_i, \theta^{(r-1)}\}.$$

In the second step, called the maximization- or M-step, one updates the value of θ with the maximizer of $Q(\theta|\theta^{(r-1)})$. The two steps taken together are the basis of the EM algorithm:

$$\begin{aligned} \text{E Step:} &\quad \text{Calculate } Q(\theta|\theta^{(r-1)}) \\ \text{M Step:} &\quad \text{Solve } \theta^{(r)} = \arg\min_\theta Q(\theta|\theta^{(r-1)}). \end{aligned}$$

The algorithm ends when $|\theta^{(r)} - \theta^{(r+1)}|$ is smaller than some prespecified threshold. As shown in [109], the EM algorithm converges to a local maximum under certain regularity conditions.

The EM algorithm can be applied to situations when some outcomes or covariates or both are missing. In the specific situation of missing categorical outcomes, note that the log-likelihood in the E-step can be reduced to a weighted average of all possible outcomes:

$$E\{\log[p(y_i|\mathbf{x}_i,\theta)]\,|\,x_i,\theta^r\} = \sum_{j=1}^{k} w(j,r)\log p(y_i=j|\mathbf{x}_i,\theta)$$

where $w(j,r) = p(y_i = j|\mathbf{x}_i,\theta^{(r)})$. As was observed in [55], the missing outcome situation can be recast into a weighted likelihood procedure. In the procedure, observations with missing outcomes are replaced with k observations. In the first replacement observation, the outcome is recorded as $y_i = 1$. In the second, the outcome is $y_i = 2$, and so on for all k replacement observations. The k new observations are weighted as $w(j,r)$. In this setup, the E-step of the EM algorithm involves updating the weights, and the M-step involves fitting the weighted log-likelihood. This procedure is called EM-by-method-of-weights.

Our proposed method is to apply similar concepts to the objective function of the RMSVM or the composite-of-binary counterpart. The resulting procedure will be iterative and similar to the EM-by-method-of-weights. In order to incorporate weights, we need to re-express the two-class SVM and RMSVM with observation weights. In [114], authors provided a weighted version of the two-class SVM. In section C.3 (page 126) of the appendix, we show that the RMSVM can be re-expressed as

$$\hat{\mathbf{f}} = \arg\min_{\mathbf{f}\in\mathcal{H}} \|\mathbf{f}\|^2_{\mathcal{H}} + \sum_{i=1}^{n} w_i L[y_i,\mathbf{f}(\mathbf{x}_i)] \tag{4.2}$$

where $w_i = 1/(\lambda n)$ in the standard setup.

For observations with missing class labels, we propose that the loss be replaced with $\widetilde{E}\{L[y_i,\mathbf{f}(\mathbf{x}_i)]|\mathbf{x},\mathbf{f}^{(r)}\}$, the expected loss over a quasi-distribution $\tilde{p}(y|\mathbf{x},\mathbf{f}^{(r)})$. Because

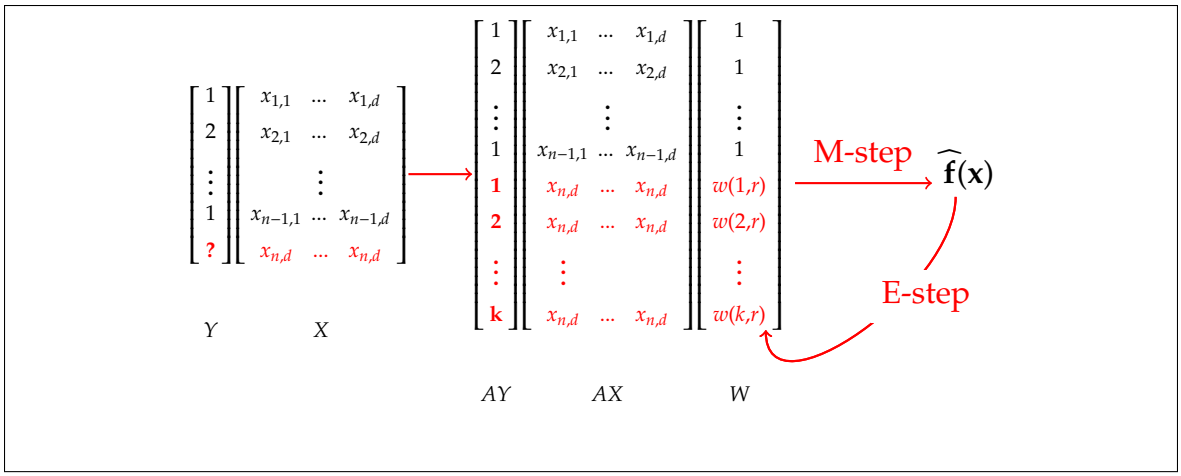


Figure 4.1: Schematic of EM algorithm for RMSVM with missing class labels

the expected loss can be expressed as a weighted sum, the loss then becomes

$$\widetilde{L}[y_i, \mathbf{f}(\mathbf{x}_i), \mathbf{f}^{(r)}] = \begin{cases} L[y_i, \mathbf{f}(\mathbf{x}_i)] & \text{label known} \\ \sum_{j=1}^{k} w(j,r) L[y_i = j, \mathbf{f}(\mathbf{x}_i)] & \text{label unknown} \end{cases} \tag{4.3}$$

The key observation is that this loss function essentially amounts to replacing each observation with missing class label with k draws from the conditional distribution and then adjusting the corresponding weight for each observation. Figure 4.1 is a schematic of the algorithm where Y denotes class labels and X denotes the covariates in the training set. The vector AY and matrix AX are the augmented outcomes and covariate in which observations with missing labels have been replaced with k observations, one for each possible class. The vector W records the corresponding weights. Like EM-by-method-of-weights, the E-step involves calculating weights $w(j,r)$ from the quasi-distribution $\tilde{p}(y|\mathbf{x}, \mathbf{f}^{(r)})$. The M-step involves solving the RMSVM objective function with the usual algorithm. The iterations continue until $\|\mathbf{f}^{(r)} - \mathbf{f}^{(r+1)}\|$ is smaller than a convergence threshold.

An important aspect of this method is the quasi-distribution $\tilde{p}(y|\mathbf{x}, \mathbf{f}^{(r)})$ from which the weights are calculated. As noted earlier, there are methods for extracting multi-class probabilities from SVMs [49, 80, 20, 110]. Here, we implement a simplified

version of the method proposed in [80]. Specifically,

$$\tilde{p}(\, y = c \,|\, \mathbf{x}, \mathbf{f}^{(r)}\,) = \frac{\exp\{-L[y = c, \mathbf{f}^{(r)}(\mathbf{x})]\}}{\sum_{j=1}^{k} \exp\{-L[y = j, \mathbf{f}^{(r)}(\mathbf{x})]\}}.$$

This quasi-distribution offers a number of desirable properties. First, larger values of $f_c(\mathbf{x})$ correspond to larger values of $\tilde{p}(y = c|\mathbf{x}, \mathbf{f}^{(r)})$. Second, points on the boundary between classes ($f_i(\mathbf{x}) = f_j(\mathbf{x})$) have equal quasi-probability of being in those classes ($\tilde{p}(y = i|\mathbf{x}, \mathbf{f}^{(r)}) = \tilde{p}(y = j|\mathbf{x}, \mathbf{f}^{(r)})$).

4.2.3 Partially Observed Outcomes

Up to this point, the proposed method treats each missing outcome as if the label is completely missing. However, in some settings, some information about class labels may be known. For example, the data is structured so that the class label is known to come from a subset of size two out of four possible outcomes. This information is easily incorporated into the EM solution by using the added information to construct the weights. Let PY be the set of potential class labels, then the updated quasi-probabilities are:

$$\tilde{p}(\, y = c \,|\, \mathbf{x}, \mathbf{f}^{(r)}, y \in PY\,) = \frac{\exp\{-L[y = c, \mathbf{f}^{(r)}(\mathbf{x})]\} \cdot I(c \in PY)}{\sum_{j \in PY} \exp\{-L[y = j, \mathbf{f}^{(r)}(\mathbf{x})]\}}.$$

It follows that for $c \notin PY$, $\tilde{p}(y = c|\mathbf{x}, \mathbf{f}^{(r)}) = 0$. Observations with zero weight can be omitted from the augmented training set.

4.2.4 Properties

In this section, we examine the properties of the proposed method by framing it in terms of a quasi-likelihood. The proposed method has important asymptotic properties which we state here and prove in section C.1 (page 124) and section C.2 (page 124) the appendix.

Proposition 1. *The algorithm converges in the sense that each iteration of the algorithm constructs a classifier which increases the observed data quasi-likelihood.*

Proposition 2. *The solution is Fisher consistent.*

Proposition 1 indicates that the algorithm converges and indicates that the solution converges to a meaningful quantity. The proof of proposition 2 shows Fisher consistency. Taken together, these propositions indicate that the solution is an unbiased method when the quasi-distribution, $\tilde{p}$, is specified correctly. That is, the method constructs a Bayes classifier under correct assumptions about the distribution. Similar to the other EM-type algorithms, the first proposition and its proof indicate that the algorithm creates a sequence of classifiers where each new classifier improves upon the previous classifier in terms of the observed data quasi-likelihood. Because the observed data quasi-likelihood is bounded, the sequence corresponding to the classifiers does converge. The second proposition indicates that under certain conditions, the function which maximizes the observed quasi-likelihood also minimizes the SVM hinge risk. Because the minimizer of the SVM hinge risk is a Bayes classifier, the solution of the proposed method is also a Bayes classifier.

4.2.5 Computation

One of the important advantages of the proposed method is that the objective function is convex, unlike S^3VM and its multi-class extension. As such, the proposed method can utilize software already widely distributed for SVMs or quadratic programing programs. One drawback is that the method requires the training set to be augmented. Because the size of the training set determines the size of the quadratic programming problem which solves the SVM objective function, the computation time may be increased. However, because our method uses software algorithms developed and optimized for SVMs (like [79]), the computation time is not noticeably longer than

the competitor's computation time. In comparison to the multi-class SD-SVM which requires semi-definite programing, our EM approach requires less memory and, in our experience, is must faster.

As with all SVM methods, our method requires the user to specify tuning parameters related to the complexity penalty (λ in equation (4.1)) and possible kernel-specific scale parameters (like σ in the Gaussian kernel). In developing and testing this method, we have used a 2-fold cross validation grid search approach when selecting values for the tuning parameters. The performance of this approach has been satisfactory. In some situations where fitting the RMSVM is relatively slow, we have found that using a using a one-versus-all multi-class SVM in the tuning parameter grid search phase can speed up the overall fitting process without any noticeable decrease in performance.

4.2.6 Improvements for two-class and multi-class settings

In situations where there is a large number of missing outcomes, the method requires adding the same covariate observation for each possible outcome. We have found that the method works best when each covariate is represented only once in the resulting SVM classifier, that is, when SVM derived from the slightly different formulation. Let the full training set be $\mathcal{T}_n$, and let $\mathcal{T}_m \subset \mathcal{T}_n$ be a subset of the training set. Let observations in $\mathcal{T}_m$ serve a basis functions of the SVM solution and calculate the empirical risk portion from all observations in $\mathcal{T}_m$, as in

$$\hat{\mathbf{f}} = \arg\min_{\mathbf{f}\in\mathcal{H}_{\mathcal{T}_m}} \|\mathbf{f}\|^2_{\mathcal{H}_{\mathcal{T}_m}} + \sum_{i\in\mathcal{T}_n}^{n} w_i L[y_i, \mathbf{f}(\mathbf{x}_i)]$$

$$\text{where } \mathbf{f}(\mathbf{z}) = [f_1(\mathbf{z}) \dots f_k(\mathbf{z})] \text{ and } f_j(\mathbf{z}) = b + \sum_{i\in\mathcal{T}_m} c_{ij}\kappa(\mathbf{z}, \mathbf{x}_i).$$

The computational details for the multi-class version are reported in section refoutcomeem:mn-multi-class (page 131). Computational details for the two-class version are similar to the multi-class version.

4.3 Simulation Study

4.3.1 Classification with Partially-Observed Outcomes

We examined the performance of our proposed method in a setting where class labels were partially observed. We considered the performance of the method under several experimental conditions such as (a) the number of covariates, (b) the number of observations and the relative balance of class proportions, (c) the percent of labels fully observed, and (d) the underlying level of classification risk. With these four factors, we constructed a $2 \times 2 \times 3 \times 4$ factorial experiment in which we generated data sets with 2 or 10 covariates, with four levels of class balance and class size, with 5% or 25% observations fully observed, and with .05, .15, or .35 underlying classification risk. This last factor refers to the relative ease or difficulty of separating the classes. The value is the proportion of errors for the best possible classifier, and lower values represent settings that are inherently easier to classify. For each of the combination of settings, we generated 100 datasets with $k = 4$ classes at the specified balance and class size along with each of the other settings. Partially observed outcomes were constructed by randomly selecting another class to which the observation may belong. Thus, for fully observed observations, the label was know exactly. For partially observed observations, the class label was known up to a set of size two. Along with the training data, we generated a validation data set 50 times the size of the training set in order to measure out-of-sample prediction accuracy. For each data set we constructed 8 classifiers of which 4 were linear SVMs and 4 were nonlinear Gaussian kernel SVMs. We constructed the complete case SVM (CC) by constructing an SVM without observations with missing class labels. We constructed a two-step cluster-then-classify SVM, labeled the Naive classifier. In the first step, an observation was labeled as the class most likely of the two possible based on the complete case SVM solution. In the second step, the SVM was constructed with the full training set with newly labeled observations. Along with constructing the SVM proposed in this paper,

we constructed the Oracle SVM, or the SVM constructed with fully observed data. In each case, tuning parameters were selected by grid search 2-fold cross validation. The methods searched over the same grid. With SVM rules in hand, we calculated the out of sample prediction accuracy.

4.3.2 Results of Partially-Observed Outcomes Simulation

The full results are reported in section C.5 (page 136) of the appendix. Here we briefly note a number of observations. First, the factor affecting performance was, expectedly, the level of classification risk. At the 0.05 level, nearly all methods were able to achieve around 90% median accuracy. The complete case SVM, however, is the exception. Particularly with smaller sample sizes, both the linear and nonlinear complete case methods scored poorly for nearly all levels of classification risk. Along with the level of classification risk, decreasing the percentage of fully observed class labels (25% to 5%) or decreasing the sample size (100 to 40 observations per class) decreased the performance of each of the non-oracle classifiers by about 5 percentage points of accuracy.

We now note a handful of setting in which the proposed method performed well in comparison to the non-oracle competitors. As a matter of full disclosure, the proposed method and the cluster-then-classify method we've labeled Naive, on average, perform about the same. However, there are settings when the proposed EM solution does seem to give better results. Table 4.1 provides a selection of simulation results. It features one of the more difficult settings: fewer observations and even fewer class labels. In both the linear set and the non-linear set, the proposed EM solution generated highly accurate decision rules. This setting also highlights that the proposed EM method generally did well in the $D = 10$ covariate simulation scenarios.

The appendix includes a series of graphical displays of the results which offer

Table 4.1: Selection of Simulation Results, Partially Observed Outcomes

					Prediction Accuracy [IQR]			
N	D	SIZE	% OBSD	RISK	Oracle-L	CC-L	Naive-L	EM-L
40	10	Equal	5	0.05	93.5 [01]	57.3 [17]	64.9 [08]	91.2 [03]
				0.15	82.3 [01]	50.2 [15]	56.6 [07]	80.1 [03]
				0.35	60.5 [02]	38.2 [08]	41.4 [07]	52.8 [03]
		Unequal	5	0.05	93.8 [00]	65.6 [23]	90.4 [06]	92.1 [04]
				0.15	82.5 [01]	54.0 [10]	70.5 [12]	81.0 [04]
				0.35	61.5 [01]	37.8 [06]	45.1 [06]	54.4 [06]

N	D	SIZE	% OBSD	RISK	Oracle-N	CC-N	Naive-N	EM-N
40	10	Equal	5	0.05	93.3 [01]	47.2 [18]	66.5 [06]	90.8 [08]
				0.15	81.7 [02]	43.5 [12]	54.6 [11]	74.7 [08]
				0.35	59.5 [03]	30.6 [10]	39.3 [08]	44.3 [09]
		Unequal	5	0.05	93.8 [01]	62.6 [11]	87.8 [12]	91.8 [02]
				0.15	82.4 [01]	50.7 [13]	63.5 [12]	78.5 [04]
				0.35	60.6 [02]	37.2 [11]	42.9 [08]	51.0 [06]

a detailed look at the distribution of prediction accuracy scores. The displays indicate that there are a handful of settings were the proposed method did not fare well. Overall, the performance of the method suggests that the EM solution is a reasonable approach when class labels are only partially observed.

4.3.3 Semi-supervised classification

We also considered a simulation experiment in standard semi-supervised learning in which class labels are missing with no other partial information. Using the same experimental factors as the previous simulation study, we altered the data generation process so that all class information was lacking for observations selected to be missing. In addition, we considered $K = 2$ and $K = 4$ settings, along with two types of missing mechanisms. The first was simple random selection, i.e., MCAR data. The second mechanism generated MAR data by censoring class labels from observations at an extreme end of the covariate distribution. As a set of competitors for this simulation study, we considered the complete case SVM, S^3VM, and the SD-SVM. However, the computational memory required by the SD-SVM method often outran our available

resources, and as such, we had to discontinue it from the simulation study. For some very modest problem sizes, the algorithm called for 8 or more gigabytes of RAM. The version of S^3VM we implemented is commonly called SVM^{light} [60].

As before, we generated 100 training datasets along with a single validation dataset 50 times the size of the training set for each combination of factors. We tuned the EM and complete case in the same way as the previous simulation study. For S^3VM, the cost parameter was tuned via grid search. For the nonlinear S^3VM, the kernel parameter was selected as the inverse of the average distance between covariates.

4.3.4 Results

The complete results are reported in section C.6 (page 147) of the appendix. As expected, the same factors that were important in the previous experiment are important in this one. Notably, the underlying classification risk is the most important factor in classifier performance. It reflects the level of separation between classes, and larger classification risk is the result of less separable classes. Table C.6 clearly shows the reduction is prediction accuracy as the the risk goes up. Table C.5 highlights the increase in prediction performance for each method as fewer class labels are missing.

Considering only linear classifiers and averaging over all experimental factors, the EM solution performed about 1 percentage point better than the complete case solution, and about 5 percentage points better than S^3VM. Stratified by underlying classification risk, the performance of all three methods is poor in the high risk setting. In the low risk setting, the EM solution outperforms the complete case by 3 percentage points and S^3VM by 10 percentage points. In the medium risk setting (risk=0.15) the differences were less, with a 1.5 and 5 percentage point difference, respectively. For the nonlinear classifiers, the pattern was the same with regards to classification risk.

Because each method performed poorly in the high risk setting, we set aside that setting and discuss the results restricted to low and medium risk. Table 4.2 provides

Table 4.2: Summary of Semi-supervised Learning Simulation Restricted to Low and Medium Classification Risk Settings

				Accuracy Difference (95% CI)			
	K	Factor		EM - S^3VM		EM - CC	
Linear	2	Class Sizes	Equal	11.1	(10, 12)	2.7	(2, 4)
			Unbalanced	6.9	(6, 8)	1.4	(1, 2)
		Missingness	MAR	7.8	(7, 9)	1.3	(0, 2)
			MCAR	10.1	(9, 11)	2.6	(2, 4)
		N	100	9.3	(8, 10)	0.5	(-0, 1)
			40	8.5	(8, 9)	3.4	(2, 4)
		OBSD	25	10.7	(10, 11)	0.9	(0, 2)
			5	6.6	(6, 8)	3.2	(2, 4)
	4	Class Sizes	Equal			-2.3	(-3, -2)
			Unbalanced			-6.0	(-7, -5)
		Missingness	MAR			-2.8	(-4, -2)
			MCAR			-5.3	(-6, -5)
		N	100			-6.7	(-7, -6)
			40			-1.2	(-2, -0)
		OBSD	25			-4.6	(-5, -4)
			5			-3.3	(-4, -2)
Nonlinear	2	Class Sizes	Equal	14.2	(13, 15)	3.5	(3, 5)
			Unbalanced	9.6	(9, 10)	1.1	(0, 2)
		Missingness	MAR	10.7	(10, 12)	1.4	(0, 2)
			MCAR	13.0	(12, 14)	3.0	(2, 4)
		N	100	12.1	(11, 13)	0.8	(-0, 2)
			40	11.4	(11, 12)	3.5	(3, 4)
		OBSD	25	13.0	(12, 14)	0.8	(0, 2)
			5	10.2	(9, 11)	3.8	(3, 5)
	4	Class Sizes	Equal			0.0	(-1, 1)
			Unbalanced			-3.8	(-5, -3)
		Missingness	MAR			-2.8	(-4, -2)
			MCAR			-0.8	(-2, 0)
		N	100			-5.9	(-7, -5)
			40			2.6	(2, 4)
		OBSD	25			-2.2	(-3, -2)
			5			-1.1	(-2, -0)

a summary of differences between the proposed method and the two competitors. Most noticeable from the table is that the proposed method works well for two-class settings. It out performs the S^3VM and the complete case method in nearly every setting. However, in the multi-class setting, the method is out performed by the complete case solution. The version of S^3VM implemented in this study performed poorly, with both linear and nonlinear kernels. The poor performance suggests that we consider other implementations of the S^3VM algorithms.

4.4 Application to Real Datasets

4.4.1 Application to HCV-TARGET data

Hepatitis C (HCV) is the most prevalent blood born infection in the United States [24] with 3.4 to 4.9 million infected US residents [5]. Recently, a string of new treatments for HCV have been approved for use in the United States and Europe which has expanded the pool of patients that can be safely treated. Because the patient pool is very large and diverse, it is possible that the population exhibits heterogeneity of treatment effectiveness particularly between patients with and without pre-existing medical conditions. One condition particularly relevant to the treatment of the liver is the presence of varices. Not only is the presence of varices an important condition by itself, it is also potentially a marker for the underlying health of the patient's liver. Because clinical trials for new generations of HCV treatments included patient populations with fewer cirrhotic patients than the currently population undergoing treatment, additional study of these patients is needed.

HCV-TARGET is a multi-center longitudinal observational study in North America and Europe which enrolls a diverse population of patients receiving treatment for hepatitis C. Researchers collected relevant demographic, clinical, and outcome data during treatment and follow-up. However, information about the presence of varices was often missing from patient records. A scientific objective is to develop a classifier

Table 4.3: Prediction Error of Semi-supervised SVM Methods Applied to HCV-TARGET Varices Data

	Prediction Error (%)	
Method	Linear Kernel	Gaussian Kernel
Proposed	24.76	24.76
CC	24.76	24.76
S^3VM	24.76	24.76

that can accurately predict the presence of varices using baseline variables; however, the potential complex relationship between liver function and baseline biomarkers requires a statistical approach that provides modeling flexibility. We apply our proposed method and its competitors to a subset of patients from the HCV-TARGET database.

The training set (N=800) are those patients that have started treatment from 1 December 2014 through 30 March 2015. Of the total, the presence or absence of varices is reported for 179 patients; the patient records for the remaining 621 subjects are missing information about varices. The validation set is N=200 patients for which the presence or absence of varices is known. Using a forward selection process, the predictors included in the model are: cirrhosis status, sex, and baseline hepatitis C viral load. These variables were selected from a list of predictors developed in consultation with members of the HCV-TARGET team. The baseline covariates are all fully observed.

We constructed decision rules with the Gaussian and linear kernels within the proposed method. The out-of-sample prediction error for each classification rule is reported in Table 4.3. This particular data example resulted in similarly effective classifiers, and both classifier performed well. Each classified correctly 75% of patients in the validation group. Despite the similar performance, this data example highlights that our proposed method can easily implement a non-linear kernel.

4.5 Conclusion

In this paper, we proposed a semi-supervised SVM algorithm which provides a computationally simpler solution than existing approaches, like S^3VM. The key idea was to employ the EM algorithm so that the SVM objective function remains convex, thus allowing a solution which can use standard and optimized software. In our simulations, we showed that the algorithm constructed a solution noticeably faster than S^3VM. Furthermore, the proposed method can accommodate non-linear kernel functions without any modification. Also, an important contribution of the paper was an exploration of partially-observed outcomes, or settings where the class labels are known to belong to a subset of possible outcomes, but the exact class label is unknown.

There are a number of areas related to these methods that deserve continued study. For example, we hope to find computational improvements that leverage the fact that the covariate vectors of observations with missing outcomes appear multiple times in the augmented training set. Another related area is the quasi-probability function which uses the classifier as a parameter. In this paper, we proposed a rather simple model. Because the method relies on the probability distribution, we suspect that future research will generate better performing alternatives.

4.6 Acknowledgments

We thank Michael Fried and HCV-TARGET for providing the data analyzed in section 4.4.1.

CHAPTER 5: FUTURE RESEARCH TOPICS

Each paper in this dissertation ends with a discussion about specific ways in which the proposed methods might be improved. This chapter describes a number of research topics which follow from or are inspired by the methods developed in the previous three papers. The common thread which links these proposed research topics and the dissertation is that each applies concepts developed in a parametric regression setting in order to address a question in the statistical learning framework.

5.1 Re-weighting Instead of Tuning

An important task when constructing an SVM is selecting reasonable values for the cost parameter along with any kernel-specific parameters like the scale parameter, σ, in the Gaussian kernel. Cross validation is a popular method for selecting the tuning parameters in which the cross validation error is calculated at each combination of pre-specified cost and kernel parameter values. The list of all possible combinations can be large and the search can be time intensive.

As noted in the papers of this dissertation, the cost parameter is closely related to the weights in the weighted formulation of the support vector machine. Specifically, we noted that the relationship $w_i = C/n$ links the weighted SVM to the standard SVM, and we have taken advantage of treating the empirical risk as an estimating-equation in order to handle missing data. The estimating-equation approach to SVMs also suggests one way for tuning the SVM which is analogous to iteratively re-weighted least squares and robust regression methods that re-weight overly influential observations. The algorithm is proposed as follows:

- Set $w_i^{(0)} = 1$.
- Repeat until convergence:
  - Solve $f^{(k)} = \arg\min \frac{1}{2}\|f\|^2 + \sum_i w_i^{(k)} L[y_i, f(x_i)]$.
  - Assign $w_i^{(k+1)} \propto \exp\{-L[y_i, f^{(k)}(x_i)]\}$.

The cost parameter attempts to balance the flexibility of SVMs with the potential risk of over-fitting the data. Much like its counterpart in the regression setting, SVM re-weighting minimizes the influence of outliers which, in turn, reduces the potential to over-fit the data.

We performed an initial proof-of-concept of this tuning method, and the simulation results look promising. We generated 100 datasets in which 240 observations were drawn from class-specific mixture distributions. (120 observations were drawn from each class-specific distribution.) Then, we constructed two SVMs with Gaussian kernel: one was constructed using grid search 10-fold cross validation, and the other was constructed using the re-weighting algorithm described above. The out-of-sample prediction accuracy was calculated from a validation set 50 times larger than the training set. The results are reported in Figure 5.1. The results are promising because in a large majority ($> 80\%$) of the the simulation datasets, the re-weighted tuning method performed better than grid search cross validation. The median improvement was 2 percentage points and the 75^{th} quantile was 5 percentage points. In some datasets, the improvement in out-of-sample prediction accuracy by the proposed method was greater than 10 percentage points.

Given the initial results, this avenue of research appears promising and worthy of continued study. The topic fits with the ideas developed in this dissertation because it utilizes the weighted formulation of the SVM in order to incorporate concepts developed for parametric models.

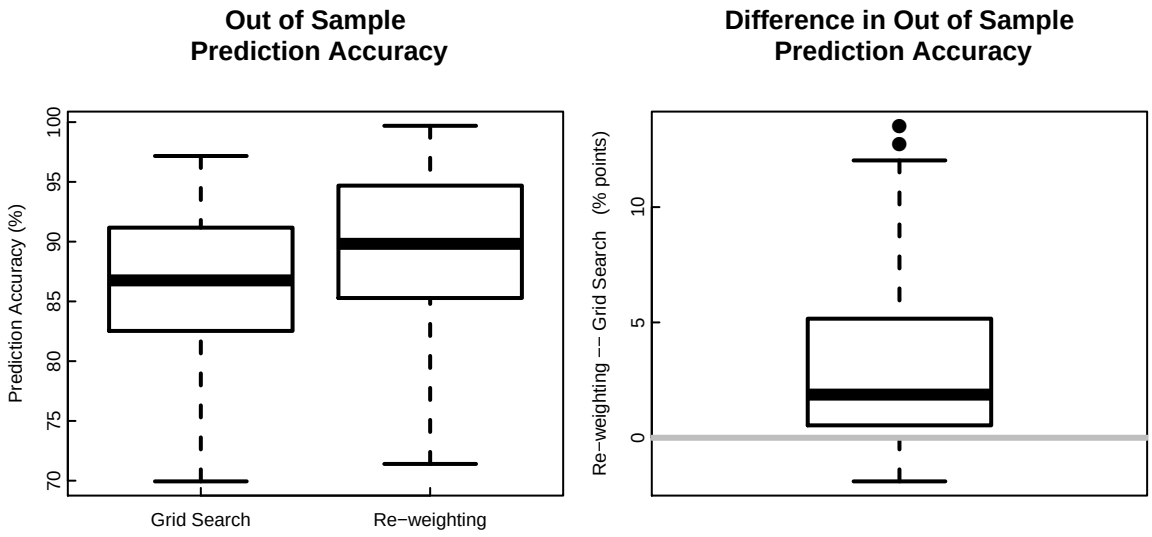


Figure 5.1: Simulation Results Comparing the Prediction Accuracy of Re-weighted Tuning and Cross Validation Tuning of the Cost Parameter

5.1.1 SVMs and NMAR Data

There are a number of research questions related to SVMs and missing data that deserve further consideration. The methods described in chapter 2 and chapter 3 were developed in the context of covariates Missing at Random (MAR). A natural next step is to consider situations when the covariates are Not Missing at Random (NMAR). In parametric settings, one way to analyze NMAR data is to explicitly model the missingness mechanism. The outcome/prediction model and the missingness model are estimated jointly such that if both models are specified correctly, then the estimated parameters are unbiased. A similar approach with SVMs is a reasonable starting point for this research question. As a statistical learning approach, the missingness model could be non-parametric.

In addition to SVM methods for NMAR missing data, there is also the more general question of whether one can even determine whether missing data are MAR or NMAR. In parametric settings, some procedures exist to test this question. One area of future research is to consider these procedures in high dimensional settings when

most of the covariates are not predictive of the outcome.

5.1.2 Causal Inference and Statistical Learning

A key issue in observational, comparative effectiveness studies is to account for potential confounding due to treatment choice. Current practices to adjust for covariates related to treatment choice include doubly robust type estimators, stratification, or matching. Each of these methods can and often use propensity scores, and researchers have recently implemented propensity scores constructed from statistical learning methods [7]. Such research is an important first step towards implementing statistical learning ideas as solutions to traditional statistical issues like confounding. There is still ample work to be done in this area. For example, recent work with covariate balancing propensity scores [58] improves on earlier parametric propensity score methods, and extending such methods with statistical learning methods is potentially valuable to causal inference and comparative effectiveness research.

APPENDIX A: AWSVM PROPOSITIONS AND ADDITIONAL RESULTS

A.1 Proposition 1

Define the quasi-sampling model as

$$\tilde{p}(y_i|x_i, f) = \left[1 + \exp\{y_i(\max[0, 1 + f(x_i)] - \max[0, 1 - f(x_i)])\}\right]^{-1};$$

Then, the quasi-likelihood is

$$\mathcal{L}(y_i, x_i, f, \theta) = \tilde{p}(y_i|x_i, f)p(x; \theta),$$

and the observed data quasi-likelihood is

$$\mathcal{L}^o(y_i, x_i^o, f, \theta) = \int \mathcal{L}(y_i, x_i, f, \theta)\, dx_i^m.$$

Thus, for a sample of size n, the observed data quasi-likelihood is

$$\mathcal{L}^o(f, \theta) = \prod_i^n \mathcal{L}^o(y_i, x_i^o, f, \theta).$$

The AWSVM solution maximizes the observed data quasi-likelihood.

Proof. We accomplish this task using the sequence of decision rules and model parameters estimated in each iteration of the AWSVM algorithm: $(f^{(0)}, \theta^{(0)})$, $(f^{(1)}, \theta^{(1)})$, $(f^{(2)}, \theta^{(2)})$, …. For notational ease, allow t to denote $(f^{(t)}, \theta^{(t)})$ when t is a function argument.

We will show

$$\sum \log\left[\mathcal{L}^o(x_i^o, y_i, t)\right] \leq \sum \log\left[\mathcal{L}^o(x_i^o, y_i, t+1)\right].$$

This indicates that $\mathcal{L}^o(t)$ is monotonically increasing with respect to t, and that each subsequent estimate of $(f^{(t)}, \theta^{(t)})$ improves this quantity.

The quasi-log-likelihood in the usual EM notation is

$$\widetilde{\ell}(f, \theta) = \sum \log \left[\tilde{p}(y_i|x_i, f)\right] + \sum \log \left[p(x_i, \theta)\right].$$

Its expectation is

$$\begin{aligned}
&\widetilde{Q}\left(f, \theta \,|\, f^{(t)}, \theta^{(t)}, y_i, x_i^o\right) \\
&\quad = \sum \widetilde{E}\left\{\log \left[\tilde{p}(y_i|x_i, f)\right] \,|\, f^{(t)}, \theta^{(t)}, y_i, x_i^o\right\} \\
&\qquad + \sum \widetilde{E}\left\{\log \left[p(x_i, \theta)\right] \,|\, f^{(t)}, \theta^{(t)}, y_i, x_i^o\right\}.
\end{aligned}$$

Denote the first sum as $\widetilde{Q}_1$ and the second sum as $\widetilde{Q}_2$. We begin the argument by noting that the following:

$$\widetilde{Q}_1(f^{(t+1)} \,|\, t, y_i, x_i^o) \geq \widetilde{Q}_1(f^{(t)}, \,|\, t, y_i, x_i^o)$$

and

$$\widetilde{Q}_2(\theta^{(t+1)} \,|\, t, y_i, x_i^o) \geq \widetilde{Q}_2(\theta^{(t)} \,|\, t, y_i, x_i^o).$$

So,

$$\widetilde{Q}(f^{(t+1)}, \theta^{(t+1)} \,|\, t, y_i, x_i^o) \geq \widetilde{Q}(f^{(t)}, \theta^{(t)} \,|\, t, y_i, x_i^o)$$

which we rewrite as

$$\sum E\left\{\log\left[\mathcal{L}(y_i, x_i, t+1)\right] \mid t, y_i, x_i^o\right\} \geq \sum E\left\{\log\left[\mathcal{L}(y_i, x_i, t)\right] \mid t, y_i, x_i^o\right\}.$$

Multiply the inner expression by one to get:

$$\sum E\left\{\log\left[\frac{\mathcal{L}(y_i, x_i, t+1)}{\mathcal{L}^o(y_i, x_i^o, t+1)}\mathcal{L}^o(y_i, x_i^o, t+1)\right] \mid t, y_i, x_i^o\right\} \geq \sum E\left\{\log\left[\frac{\mathcal{L}(y_i, x_i, t)}{\mathcal{L}^o(y_i, x_i^o, t)}\mathcal{L}^o(y_i, x_i^o, t)\right] \mid t, y_i, x_i^o\right\}.$$

Rewrite as

$$\begin{aligned}
&\text{(A)} \quad \sum E\left[\log\left(\mathcal{L}^o(y_i, x_i^o, t+1)\right) \mid t, y_i, x_i^o\right] + \\
&\text{(B)} \qquad E\left[\log\left(\tilde{\mathcal{L}}(x_i^m | x_i^o, y_i, t+1)\right) \mid t, y_i, x_i^o\right] \\
&\text{(C)} \qquad \geq \sum E\left[\log\left(\mathcal{L}^o(y_i, x_i^o, t)\right) \mid t, y_i, x_i^o\right] + \\
&\text{(D)} \qquad E\left[\log\left(\tilde{\mathcal{L}}(x_i^m | x_i^o, y_i, t)\right) \mid t, y_i, x_i^o\right].
\end{aligned}$$

Note, in lines (B) and (D) that

$$\tilde{\mathcal{L}}(x_i^m | x_i^o, y_i, t+1) = \frac{\mathcal{L}(y_i, x_i, t+1)}{\mathcal{L}^o(y_i, x_i^o, t+1)}$$

is the conditional distribution of x_i^m given observed data and model parameters. Recall the property of Kullback-Leibler divergence: $E_P[\log(dP)] \geq E_P[\log(dQ)]$. Applied to lines (B) and (D), we note

$$\begin{aligned}
&\text{(B)} \qquad E\left\{\log\left[\tilde{\mathcal{L}}(x_i^m | x_i^o, y_i, t+1)\right] \mid t, y_i, x_i^o\right\} \\
&\text{(D)} \qquad \leq E\left\{\log\left[\tilde{\mathcal{L}}(x_i^m | x_i^o, y_i, t)\right] \mid t, y_i, x_i^o\right\}.
\end{aligned}$$

This means for lines (A) and (C),

$$\begin{aligned} \text{(A)} \qquad & \sum E\left\{\log\left[\mathcal{L}^o(y_i, x_i^o, t+1)\right] \mid t, y_i, x_i^o\right\} \\ \text{(C)} \qquad & \geq \sum E\left\{\log\left[\mathcal{L}^o(y_i, x_i^o, t)\right] \mid t, y_i, x_i^o\right\}. \end{aligned}$$

Note that $\mathcal{L}^o$ is not a function of x_i^m, so

$$\sum \log\left[\mathcal{L}^o(y_i, x_i^o, t+1)\right] \geq \sum \log\left[\mathcal{L}^o(y_i, x_i^o, t)\right].$$

This is the desired result. The observed-quasi-likelihood is non-decreasing with respect to the sequence of quasi-EM classifiers $f^{(t)}$ and data model parameters $\theta^{(t)}$.

A.2 Proposition 2

Let $\mathcal{L}^o$ be the observed quasi likelihood defined in proposition 1. If the data model $P(x;\theta)$ is specified correctly, then decision function that maximizes the expected observed quasi likelihood is asymptotically a Bayes classifier.

Proof. Consider the case when $w(x;\theta)$ is the correct data distribution. Let $p^o = p(y = 1|x^o)$.

$$\begin{aligned} & E\left[\log \mathcal{L}^o(y, x^o, t)\right] \\ & \quad = E_{x^o}\left\{E_y\left[\log \mathcal{L}^o(y, x^o, t) \,|\, x^o\right]\right\} \\ & \quad = \int E_y\left[\log \mathcal{L}^o(y, x^o, t) \,|\, x^o\right] \, w(x^o)\, dx^o \end{aligned}$$

Note

$$\begin{aligned} & E_y\left[\log \mathcal{L}^o(y, x^o, t) \,|\, x^o\right] \\ & \quad = p^o \log\left[\mathcal{L}^o(1, x^o, t)\right] + (1 - p^o)\log\left[\mathcal{L}^o(-1, x^o, t)\right] \\ & \quad = p^o \log\left[\frac{\mathcal{L}^o(1, x^o, t)}{w(x^o)}\right] + (1 - p^o)\log\left[\frac{\mathcal{L}^o(-1, x^o, t)}{w(x^o)}\right] \\ & \qquad + \log[w(x^o)] \end{aligned}$$

Let $\tilde{p}^o = \tilde{p}(y = 1|x^o) = \frac{\mathcal{L}^o(1,x^o,t)}{w(x^o)}$.

$$\begin{aligned} & E\left[\log \mathcal{L}^o(y, x^o, t)\right] \\ & \quad = \int \left[p^o \log\left(\tilde{p}^o\right) + (1 - p^o)\log(1 - \tilde{p}^o)\right] w(x^o)\, dx^o + C \end{aligned}$$

Note that for each x, the value $\tilde{p}^o$ that maximizes the inner quantity is $\tilde{p}^o = p^o$. This

means the maximizer, f^*, satisfies

$$\frac{\mathcal{L}^o(1, x^o, f^*)}{w(x^o)} = p(y = 1|x^o),$$

and it implies

$$\tilde{p}(y = 1|x^o, f^*) = p(y = 1|x^o).$$

Finally, we note

$$\tilde{E}\{L_h[y, f(x)]|x^o\} = E\{L_h[y, f(x)]|x^o\}$$

is the hinge-loss classification risk for (y, x^o). The maximizer of the hinge-loss classification risk is a Bayes classifier.

APPENDIX B: RESULTS AND PROPOSITIONS FOR WEIGHTED SUPPORT VECTOR MACHINES

B.1 Proposition 1

Consider a sample of size N of covariates, outcomes, and weights: (x_1, y_1, w_1), (x_2, y_2, w_2), . . ., (x_n, y_n, w_n). *If* $w_i < 0$ *for a subset of observations, then empirical risk minimization*

$$f = \arg\min_{f \in \mathcal{H}} \sum_{i=1}^{n} w_i I[y_i \neq f(x_i)]$$

can be re-expressed as

$$f = \arg\min_{f \in \mathcal{H}} \sum_{i=1}^{n} |w_i| \, I[sign(w_i) y_i \neq f(x_i)].$$

Proof.

$$\begin{aligned}
\sum_{i=1}^{n} w_i I[y_i \neq f(x_i)] &= \sum_{w_i>0} w_i I[y_i \neq f(x_i)] + \sum_{w_i<0} w_i I[y_i \neq f(x_i)] \\
&= \sum_{w_i>0} w_i I[y_i \neq f(x_i)] + \sum_{w_i<0} w_i \{1 - I[y_i = f(x_i)]\} \\
&= \sum_{w_i>0} w_i I[y_i \neq f(x_i)] + \sum_{w_i<0} -w_i I[y_i = f(x_i)] + \sum_{w_i<0} w_i \\
&= \sum_{w_i>0} |w_i| \, I[\text{sign}(w_i) y_i \neq f(x_i)] + \sum_{w_i<0} |w_i| \, I[\text{sign}(w_i) y_i \neq f(x_i)] + C \\
&= \sum_{i=1}^{n} |w_i| \, I[\text{sign}(w_i) y_i \neq f(x_i)] + C
\end{aligned}$$

B.2 Proposition 2

Recall equation (3.2)*, the objective function for the DRSVM,*

$$\hat{f}_{DR} = \arg\min_{f \in \mathcal{H}} \ \lambda \|f\|^2_{\mathcal{H}} + \frac{1}{n} \sum_i \frac{r_i}{\hat{p}_i} max[0,\ 1 - y_i f(x_i)] - \left(\frac{r_i}{\hat{p}_i} - 1 \right) \phi(y_i, x_i^a).$$

The solution is Fisher consistent if one of the following holds:

1. *The surrogate loss function captures the true loss,* $\phi(y_i, x_i^a) = E\{max[0,\ 1 - y_i f(x_i)] | y_i, x_i^a\}$*, for all observations.*

2. *The estimated probabilities* $\hat{p}_i$ *are unbiased in the sense that* $\hat{p}_i = p(r_i = 1 | x_i,\ y_i)$*.*

Proof of statement 1. Let $\phi(y, x^a) = E\{\max[0,\ 1 - y f(x)] | y, x^a\}$. We show that expectation of the DRSVM loss is equal to the expectation of the standard SVM hinge loss. Thus, the population minimizer of the DRSVM risk is also a population minimizer of the

standard hinge risk which is a Bayes classifier.

$$E_{y,x}\left\{\frac{r}{\hat{p}}\max[0,\,1-yf(x)]-\left(\frac{r}{\hat{p}}-1\right)E\{\max[0,\,1-yf(x)]|y,x^a\}\right\}$$

$$=\frac{r}{\hat{p}}\,E_{y,x}\left\{\max[0,\,1-yf(x)]\right\}$$
$$-\left(\frac{r}{\hat{p}}-1\right)E_{y,x}\left\{E\{\max[0,\,1-yf(x)]|y,x^a\}\right\}$$

$$=\frac{r}{\hat{p}}\,E_{y,x}\left\{\max[0,\,1-yf(x)]\right\}$$
$$-\left(\frac{r}{\hat{p}}-1\right)E_{y,x}\left\{\max[0,\,1-yf(x)]\right\}$$

$$=\left(\frac{r}{\hat{p}}-\frac{r}{\hat{p}}+1\right)E_{y,x}\left\{\max[0,\,1-yf(x)]\right\}$$

$$=E_{y,x}\left\{\max[0,\,1-yf(x)]\right\}$$

Proof of statement 2. Let $\hat{p}=p(r=1|y,x)$. We show that expectation of the DRSVM loss is equal to the expectation of the standard SVM hinge loss. Thus, the population minimizer of the DRSVM risk is also a population minimizer of the standard hinge

risk which is a Bayes classifier.

$$E_{r,y,x}\left\{\frac{r}{\hat{p}}\max[0,\,1-yf(x)]-\left(\frac{r}{\hat{p}}-1\right)\phi(y,x^a)\right\}$$

$$\begin{aligned}
&= E_{y,x}\,E_{r|y,x}\left\{\frac{r}{\hat{p}}\max[0,\,1-yf(x)]\right\}\\
&\quad - E_{y,x}\,E_{r|y,x}\left\{\left(\frac{r}{\hat{p}}-1\right)\phi(y,x^a)\right\}
\end{aligned}$$

$$\begin{aligned}
&= E_{y,x}\,\left\{p(r=1|y,x)\frac{1}{\hat{p}}\max[0,\,1-yf(x)]\right\}\\
&\quad - E_{y,x}\,\left\{p(r=1|y,x)\left(\frac{1}{\hat{p}}-1\right)\phi(y,x^a)-p(r=0|y,x)\phi(y,x^a)\right\}
\end{aligned}$$

$$\begin{aligned}
&= E_{y,x}\,\{\max[0,\,1-yf(x)]\}\\
&\quad - 0
\end{aligned}$$

B.3 Proposition 3

If either of the following conditions hold

1. $E\left[\frac{r_i}{\hat{p}_i}\right] = 1$
2. $E\left[L_h[y_i, f(x_i)] - \phi(y_i, x_i^a)\right] = 0$

then, as n gets large, the DRSVM classifier achieves the Bayes risk.

Proof. The setup of this proof builds on the consistency results for standard SVMs discussed in [94]. Let $R(f) = E\{L_h[y, f(x)]\}$ denote the risk, and let $R_{emp}(f) = \frac{1}{n}\sum L_h[y_i, f(x_i)]$ denote the empirical risk. Denote the Bayes classifier as $f_{\text{bayes}} = \arg\min_f R(f)$. Denote the standard svm classifier as $\hat{f}_{svm} = \arg\min_f R_{emp}(f)$. Likewise, let $R^{dr}_{emp}(f)$ be the empirical loss of the DRSVM, and let $\hat{f}_{\text{dr}} = \arg\min_f R^{dr}_{emp}(f)$. The following statements follow:

$$R(\hat{f}_{\text{dr}}) - R(f_{\text{bayes}}) \geq 0,$$
$$R^{dr}_{emp}(f_{\text{bayes}}) - R^{dr}_{emp}(\hat{f}_{\text{dr}}) \geq 0$$

The sum of these inequalities satisfies:

$$\begin{aligned}
\text{(P3.1)} \qquad 0 &\leq R(\hat{f}_{\text{dr}}) - R(f_{\text{bayes}}) + R^{dr}_{emp}(f_{\text{bayes}}) - R^{dr}_{emp}(\hat{f}_{\text{dr}}) \\
&= R(\hat{f}_{\text{dr}}) - R^{dr}_{emp}(\hat{f}_{\text{dr}}) + R^{dr}_{emp}(f_{\text{bayes}}) - R(f_{\text{bayes}}) \\
&\leq \underbrace{\sup_f [R(f) - R^{dr}_{emp}(f)]}_{A} + \underbrace{R^{dr}_{emp}(f_{\text{bayes}}) - R(f_{\text{bayes}})}_{B}
\end{aligned}$$

The basic idea of this proof is to leverage results for SVMs to show that the right hand side of the last line of the above inequality converges to 0. Two results which we

will leverage are (proved elsewhere such as [94]),

$$\sup_f [R(f) - R_{emp}(f)] \to_p 0,$$

$$|R_{emp}(f_{\text{bayes}}) - R(f_{\text{bayes}})| \to_p 0$$

as n gets large.

To proceed, suppose that all covariates are fully observed, but that construction of the DRSVM and the calculation of the DR empirical risk proceeds as if some covariates are missing. Define

$$\begin{aligned} Q(f) &= R^{dr}_{emp}(f) - R_{emp}(f) \\ &= \frac{1}{n} \sum_{i=1}^{n} \left[\frac{r_i}{\tilde{p}_i} - 1 \right] \left[L_h[y_i, f(x_i)] - \phi(y_i, x_i^a) \right]. \end{aligned}$$

Note that by the strong law of large numbers,

$$Q(f) \to_{as} E\left\{ \left[\frac{r_1}{\tilde{p}_1} - 1 \right] \left[L_h[y_1, f(x_1)] - \phi(y_1, x_1^a) \right] \right\},$$

and it is clear to see that

$$E\left\{ \left[\frac{r_1}{\tilde{p}_1} - 1 \right] \left[L_h[y_1, f(x_1)] - \phi(y_1, x_1^a) \right] \right\} = 0$$

if either $E[r_1/p_1] = 1$ (condition 1) or $E[L_h[y_1, f(x_1)] - \phi(y_1, x_1^a)] = 0$ (condition 2) hold. Returning back to expression A of (P3.1),

$$\begin{aligned} \sup_f [R(f) - R^{dr}_{emp}(f)] &= \sup_f [R(f) - R_{emp}(f) - Q(f)] \\ &\leq \sup_f [R(f) - R_{emp}(f)] + \sup_f |Q(f)|. \end{aligned}$$

Because $\sup_f[R(f) - R_{emp}(f)] \to_p 0$ and $\sup_f |Q(f)| \to 0$, if follows that

$$\sup_f[R(f) - R^{dr}_{emp}(f)] \to_p 0.$$

We now proceed to expression B of (P3.1).

$$\begin{aligned} R^{dr}_{emp}(f_{\text{bayes}}) - R(f_{\text{bayes}}) &= R_{emp}(f_{\text{bayes}}) + Q(f_{\text{bayes}}) - R(f_{\text{bayes}}) \\ &\leq |R_{emp}(f_{\text{bayes}}) - R(f_{\text{bayes}})| + |Q(f_{\text{bayes}})| \end{aligned}$$

As noted earlier, $|R_{emp}(f_{\text{bayes}}) - R(f_{\text{bayes}})|$ converges to 0 as n gets large. Likewise for $|Q(f_{\text{bayes}})|$, if condition 1 or condition 2 holds.

Thus, both expressions A and B in (P3.1) converge to zero. Because

$$0 \leq R(\hat{f}_{\text{dr}}) - R(f_{\text{bayes}}) \leq A + B$$

and $(A + B) \to_p 0$, it follows that

$$|R(\hat{f}_{\text{dr}}) - R(f_{\text{bayes}})| \to_p 0$$

if either condition 1 or condition 2 holds.

B.4 Simulation Results for Doubly Robust SVM

The following tables report simulation results involving the DRSVM. Each method is labeled as follows:

Label	Method
cc	Complete Case
dr	Doubly Robust
dw	Doubly Weighted
mi	Mean Imputation
mi3	Multiple Imputation
knn	*k*-Nearest Neighbor
socp	Probability Constraint

Table B.1: Simulation Results (2 Covariates, Missingness depends on Y and X)

						Median AOPE [IQR]						
D	Missingness	X distr	Boundary	Kernel	β	cc	dr	dw	mi	knn	mi3	socp
2	Y and X	Normal	Linear	Linear	-6	13.0 [3.8]	2.8 [5.4]	6.7 [7.7]	13.3 [4.5]	12.2 [5.6]	2.3 [3.2]	14.2 [3.1]
					-2	5.7 [3.7]	0.6 [1.4]	1.4 [2.0]	7.0 [3.9]	9.4 [5.6]	1.8 [2.3]	10.8 [4.4]
					0	0.3 [0.9]	0.1 [0.6]	0.3 [0.7]	0.3 [0.7]	3.3 [2.9]	1.9 [2.0]	2.5 [2.9]
					2	12.4 [8.3]	0.5 [1.1]	1.7 [2.0]	3.5 [2.2]	5.3 [2.3]	3.2 [2.5]	3.6 [1.2]
					6	32.8 [7.1]	1.8 [2.3]	4.8 [3.4]	4.6 [2.2]	8.0 [2.4]	3.8 [3.4]	5.2 [1.1]
				Non-linear	-6	13.6 [3.6]	20.2 [10.4]	25.9 [6.9]	16.4 [5.1]	13.1 [6.6]	8.9 [10.6]	
					-2	6.4 [3.8]	8.6 [10.8]	20.2 [5.8]	7.8 [3.8]	9.0 [5.0]	3.6 [5.9]	
					0	0.6 [1.3]	1.6 [3.7]	12.9 [3.7]	0.7 [1.1]	4.2 [2.9]	1.8 [1.8]	
					2	13.1 [6.8]	2.4 [4.1]	8.9 [3.7]	5.2 [3.3]	5.9 [3.2]	4.0 [2.9]	
					6	31.7 [7.4]	5.1 [4.4]	9.4 [4.2]	8.4 [3.2]	8.7 [3.9]	9.2 [5.6]	
			Non-linear	Linear	-6	-0.3 [0.7]	0.5 [1.6]	-0.2 [0.9]	-0.1 [0.5]	-0.1 [0.6]	2.2 [7.2]	0.0 [0.7]
					-2	-0.3 [0.8]	0.1 [0.8]	-0.1 [0.8]	-0.1 [0.5]	-0.1 [0.5]	0.9 [4.2]	0.0 [0.7]
					0	0.2 [1.2]	0.1 [0.7]	0.1 [0.7]	0.1 [0.4]	0.0 [0.5]	0.6 [2.3]	0.1 [0.6]
					2	21.4 [8.0]	0.1 [0.5]	0.3 [0.8]	1.6 [2.8]	-0.1 [0.4]	0.7 [2.4]	0.1 [0.6]
					6	43.4 [2.2]	0.1 [0.5]	1.0 [1.8]	2.6 [2.7]	-0.0 [0.6]	0.8 [2.4]	0.1 [0.9]
				Non-linear	-6	8.8 [1.9]	12.4 [6.9]	19.2 [5.8]	10.3 [3.5]	9.9 [2.9]	11.6 [5.6]	
					-2	5.1 [3.9]	6.4 [7.6]	15.7 [6.6]	6.9 [4.7]	7.7 [4.5]	10.1 [4.0]	
					0	0.9 [1.8]	2.4 [3.1]	11.3 [4.1]	2.6 [2.4]	4.7 [3.6]	10.4 [4.1]	
					2	11.2 [5.7]	2.8 [4.1]	8.3 [3.7]	7.6 [7.7]	5.5 [3.6]	13.3 [6.0]	
					6	26.4 [8.4]	3.7 [4.5]	7.7 [4.1]	17.1 [5.5]	6.4 [3.2]	16.4 [5.0]	
		χ^2	Linear	Linear	-6	13.1 [2.5]	6.6 [7.2]	7.1 [3.7]	11.5 [1.7]	11.8 [1.8]	4.9 [5.0]	12.3 [2.1]
					-2	6.5 [4.3]	2.9 [5.3]	4.1 [2.5]	10.0 [1.8]	11.4 [1.9]	5.1 [3.7]	11.7 [2.5]
					0	0.4 [1.7]	3.0 [3.7]	0.5 [1.3]	1.2 [2.2]	8.6 [2.9]	3.7 [3.3]	6.6 [3.0]
					2	6.4 [3.6]	2.2 [3.6]	0.6 [1.7]	3.1 [1.6]	6.0 [2.5]	5.1 [2.3]	5.1 [2.3]
					6	13.7 [7.7]	1.9 [3.3]	3.5 [4.0]	4.0 [1.3]	6.2 [1.8]	7.7 [4.4]	5.6 [1.7]
				Non-linear	-6	12.5 [2.2]	15.6 [11.5]	23.6 [7.0]	11.6 [4.1]	11.3 [3.4]	12.6 [4.7]	
					-2	7.2 [3.9]	8.9 [11.8]	21.1 [7.6]	7.9 [3.5]	9.7 [3.5]	8.8 [5.1]	
					0	1.0 [2.1]	2.1 [3.6]	16.5 [9.0]	1.8 [2.5]	5.7 [3.5]	4.0 [2.4]	
					2	12.0 [8.4]	2.1 [2.5]	6.2 [7.6]	3.1 [2.3]	4.9 [2.5]	5.2 [3.8]	
					6	24.7 [10.0]	3.5 [3.6]	7.5 [7.9]	4.6 [2.6]	5.2 [2.4]	8.2 [5.9]	
			Non-linear	Linear	-6	8.6 [2.2]	2.0 [4.1]	3.9 [3.4]	4.6 [1.5]	5.4 [1.5]	9.8 [6.0]	5.5 [1.2]
					-2	5.3 [3.1]	1.9 [3.7]	2.2 [3.6]	2.5 [3.1]	5.0 [2.3]	10.1 [5.9]	5.1 [1.7]
					0	0.3 [1.2]	2.0 [2.1]	0.2 [1.0]	0.3 [0.9]	2.9 [2.8]	7.2 [3.7]	3.3 [2.3]
					2	24.7 [1.7]	1.3 [1.8]	-0.1 [0.8]	1.1 [1.3]	4.6 [1.6]	0.6 [2.5]	3.4 [2.1]
					6	25.1 [1.2]	1.0 [2.2]	-0.0 [0.8]	2.1 [1.5]	5.2 [1.0]	-0.1 [1.2]	4.5 [1.8]
				Non-linear	-6	5.3 [3.1]	11.5 [6.4]	13.2 [5.4]	5.9 [3.8]	5.7 [6.0]	4.5 [3.4]	
					-2	2.2 [2.3]	8.5 [7.3]	12.3 [5.5]	3.9 [2.6]	4.0 [5.2]	4.2 [2.8]	
					0	0.7 [1.6]	2.2 [3.4]	9.9 [5.1]	1.9 [1.8]	2.8 [3.5]	4.3 [4.6]	
					2	9.6 [5.4]	2.2 [3.1]	4.9 [5.2]	9.3 [5.1]	5.3 [3.4]	8.9 [5.9]	
					6	25.3 [6.9]	3.8 [6.2]	3.5 [6.3]	16.0 [7.2]	6.4 [2.3]	11.8 [5.7]	

Table B.2: Simulation Results (2 Covariates, Missingness depends on X)

						Median AOPE [IQR]						
D	Missingness	X distr	Boundary	Kernel	β	cc	dr	dw	mi	knn	mi3	socp
2	X	Normal	Linear	Linear	-6	0.8 [3.0]	2.3 [3.5]	2.8 [3.9]	1.9 [3.0]	4.1 [4.4]	3.0 [3.5]	3.6 [3.9]
					-2	0.5 [1.2]	0.7 [1.1]	0.5 [1.5]	1.4 [2.6]	3.4 [4.9]	2.6 [2.3]	2.8 [3.1]
					0	0.2 [0.9]	0.3 [0.6]	0.2 [0.6]	0.3 [0.8]	3.4 [3.3]	2.0 [2.0]	2.0 [2.3]
					2	0.4 [1.1]	0.7 [1.2]	0.8 [1.3]	1.3 [1.6]	4.7 [5.1]	2.1 [2.5]	2.6 [2.2]
					6	0.9 [2.2]	2.2 [3.7]	2.4 [2.9]	2.1 [2.9]	2.8 [5.4]	2.9 [3.2]	3.7 [4.0]
				Non-linear	-6	4.5 [12.9]	9.0 [6.9]	14.8 [6.0]	2.4 [4.5]	5.7 [7.4]	3.2 [3.5]	
					-2	1.1 [2.7]	5.3 [7.0]	13.5 [4.6]	2.2 [2.8]	5.2 [4.5]	2.4 [2.1]	
					0	0.4 [1.5]	1.5 [2.7]	12.6 [4.2]	0.7 [1.5]	4.2 [4.5]	1.8 [1.8]	
					2	1.3 [3.0]	4.5 [5.4]	12.8 [3.8]	2.1 [2.8]	5.0 [5.0]	2.1 [2.3]	
					6	4.4 [12.5]	8.0 [7.1]	13.3 [6.1]	2.1 [3.5]	6.0 [7.3]	2.7 [3.6]	
			Non-linear	Linear	-6	20.9 [1.3]	0.1 [0.7]	0.7 [1.0]	-0.1 [0.4]	0.2 [0.7]	0.2 [1.7]	0.0 [0.7]
					-2	0.5 [17.8]	0.1 [0.8]	0.4 [0.9]	0.0 [0.5]	0.0 [0.6]	0.3 [1.0]	0.0 [0.7]
					0	0.2 [1.0]	0.2 [0.7]	0.1 [0.6]	0.0 [0.4]	0.0 [0.5]	0.4 [2.7]	-0.0 [0.4]
					2	5.7 [21.5]	0.1 [0.6]	0.0 [0.7]	0.6 [1.6]	0.0 [0.6]	0.9 [4.4]	0.1 [0.4]
					6	22.7 [14.9]	0.2 [0.5]	0.7 [1.9]	1.8 [2.7]	0.7 [1.8]	4.0 [6.2]	0.3 [0.7]
				Non-linear	-6	20.3 [19.8]	7.0 [7.8]	11.4 [5.4]	7.5 [7.4]	10.0 [7.8]	19.4 [7.7]	
					-2	4.9 [5.3]	4.2 [5.5]	11.6 [5.5]	3.8 [4.9]	7.8 [4.1]	17.1 [6.1]	
					0	0.9 [1.7]	2.6 [3.3]	11.1 [3.3]	2.7 [2.2]	5.0 [5.0]	10.6 [4.3]	
					2	1.5 [4.2]	4.2 [6.1]	12.2 [5.1]	2.8 [2.5]	3.3 [3.6]	11.8 [6.4]	
					6	3.8 [12.7]	7.9 [6.7]	11.9 [6.2]	3.8 [4.9]	5.9 [8.5]	16.1 [6.6]	
		χ^2	Linear	Linear	-6	1.5 [3.7]	2.6 [4.5]	1.3 [2.9]	1.1 [1.9]	6.5 [5.7]	6.0 [5.3]	5.1 [2.6]
					-2	0.7 [2.2]	2.0 [3.0]	0.5 [1.6]	0.8 [2.7]	8.2 [3.4]	5.8 [4.2]	5.5 [2.7]
					0	0.6 [1.8]	3.4 [4.2]	0.7 [1.4]	1.2 [2.2]	7.9 [2.1]	3.5 [2.4]	6.4 [3.2]
					2	0.7 [2.2]	5.1 [4.9]	1.0 [1.5]	2.1 [2.9]	9.3 [2.9]	2.6 [2.9]	9.3 [3.6]
					6	4.1 [5.2]	4.1 [4.6]	2.5 [2.8]	7.4 [7.0]	10.2 [2.8]	2.8 [3.7]	11.1 [2.7]
				Non-linear	-6	1.7 [4.4]	13.6 [13.2]	15.2 [13.9]	1.5 [3.4]	4.2 [4.1]	5.9 [4.3]	
					-2	0.9 [2.3]	4.4 [12.4]	19.5 [7.8]	1.4 [2.8]	4.7 [2.8]	5.3 [3.8]	
					0	1.2 [1.9]	2.4 [3.5]	16.8 [6.9]	1.4 [2.1]	5.6 [3.0]	3.4 [2.7]	
					2	8.6 [11.3]	3.4 [3.8]	6.7 [6.4]	2.1 [2.4]	6.2 [3.1]	2.2 [3.4]	
					6	25.1 [22.9]	4.4 [4.1]	3.8 [3.6]	3.9 [6.1]	8.6 [5.0]	2.4 [3.6]	
			Non-linear	Linear	-6	23.9 [22.4]	1.1 [3.1]	0.4 [3.0]	3.6 [2.5]	4.8 [2.4]	8.5 [4.9]	5.1 [1.5]
					-2	1.0 [2.9]	1.3 [1.8]	0.4 [1.3]	1.7 [2.4]	4.9 [1.4]	8.4 [4.0]	4.7 [1.7]
					0	0.5 [1.0]	1.8 [2.1]	0.2 [1.0]	0.4 [0.9]	3.3 [2.5]	6.8 [4.7]	3.2 [2.4]
					2	0.4 [2.0]	1.2 [1.4]	0.0 [0.9]	0.1 [0.8]	2.5 [2.4]	4.8 [5.4]	2.0 [2.7]
					6	2.9 [14.1]	0.8 [1.0]	0.2 [1.0]	-0.1 [0.7]	0.7 [2.2]	2.1 [6.7]	1.6 [2.8]
				Non-linear	-6	5.3 [9.5]	16.3 [6.7]	14.9 [6.3]	6.7 [9.6]	8.6 [13.1]	12.0 [7.2]	
					-2	1.2 [2.4]	11.2 [9.7]	14.1 [5.8]	2.2 [3.7]	6.5 [5.4]	7.5 [4.1]	
					0	0.9 [1.5]	3.0 [4.1]	9.8 [4.6]	1.7 [1.4]	2.3 [2.7]	4.8 [3.3]	
					2	1.8 [2.6]	2.2 [2.5]	4.1 [4.0]	2.0 [1.8]	2.8 [2.6]	4.1 [3.9]	
					6	2.4 [2.9]	2.3 [2.5]	1.8 [3.2]	1.9 [2.2]	2.8 [2.5]	8.7 [6.4]	

Table B.3: Simulation Results (10 Covariates, Missingness depends on Y and X)

						Median AOPE [IQR]						
D	Missingness	X distr	Boundary	Kernel	β	cc	dr	dw	mi	knn	mi3	socp
10	Y and X	Normal	Linear	Linear	-6	17.3 [4.3]	3.8 [2.5]	3.2 [2.3]	2.3 [2.2]	2.4 [1.7]	1.2 [1.4]	2.1 [1.8]
					-2	10.7 [3.6]	2.9 [2.5]	1.8 [2.0]	1.7 [1.5]	1.3 [1.7]	0.7 [1.4]	1.4 [1.6]
					0	1.6 [1.5]	1.5 [1.3]	0.6 [1.0]	0.6 [1.0]	0.4 [1.1]	0.6 [0.9]	0.5 [1.1]
					2	13.6 [6.7]	1.8 [1.9]	2.2 [1.7]	1.0 [1.4]	0.4 [1.0]	1.0 [1.4]	0.7 [1.5]
					6	22.0 [5.2]	2.1 [1.7]	3.2 [2.4]	1.5 [1.7]	0.9 [1.1]	1.3 [1.9]	1.3 [1.7]
				Non-linear	-6	16.6 [4.7]	4.2 [3.4]	5.1 [3.8]	3.7 [2.6]	2.8 [2.0]	1.4 [1.9]	
					-2	11.1 [5.5]	3.7 [2.4]	3.0 [2.7]	2.1 [1.9]	1.5 [1.7]	0.8 [1.4]	
					0	1.8 [2.2]	2.2 [2.7]	1.0 [1.9]	0.5 [1.1]	0.4 [1.3]	0.5 [0.9]	
					2	14.5 [7.0]	2.1 [2.4]	3.2 [2.4]	1.2 [1.6]	0.6 [1.4]	1.0 [1.7]	
					6	21.5 [6.1]	2.6 [2.6]	4.2 [3.0]	1.6 [2.0]	0.8 [1.6]	1.0 [1.6]	
			Non-linear	Linear	-6	-3.8 [2.9]	-0.2 [2.2]	-3.8 [3.5]	-0.1 [1.8]	0.0 [1.1]	0.2 [1.1]	0.0 [1.4]
					-2	-4.6 [3.6]	-0.2 [1.9]	-4.6 [3.3]	-0.0 [1.5]	0.0 [1.3]	0.2 [1.3]	-0.1 [1.4]
					0	0.2 [3.1]	0.0 [1.3]	0.0 [1.9]	0.0 [0.5]	0.0 [0.5]	0.0 [1.0]	0.0 [1.1]
					2	2.1 [3.6]	0.4 [1.2]	0.8 [4.1]	0.5 [1.4]	0.0 [0.7]	0.3 [1.5]	0.0 [1.0]
					6	3.5 [4.6]	0.3 [2.1]	1.1 [3.4]	0.3 [1.8]	0.0 [0.7]	0.3 [1.5]	0.0 [0.8]
				Non-linear	-6	17.2 [7.1]	3.9 [3.1]	2.5 [1.9]	1.6 [2.1]	1.0 [1.2]	1.4 [1.8]	
					-2	9.8 [7.8]	4.2 [3.4]	2.2 [2.1]	0.9 [2.0]	0.5 [1.4]	1.1 [1.4]	
					0	2.8 [2.5]	3.8 [3.9]	1.1 [1.6]	0.7 [1.4]	0.5 [1.3]	0.8 [1.7]	
					2	11.0 [4.8]	4.3 [2.4]	2.0 [2.2]	1.0 [1.5]	0.6 [1.2]	1.0 [1.5]	
					6	17.5 [6.1]	3.7 [2.6]	2.1 [1.9]	1.5 [1.7]	0.6 [1.4]	1.4 [1.7]	
		χ^2	Linear	Linear	-6	9.6 [3.9]	1.2 [1.6]	1.3 [1.4]	0.4 [0.8]	1.5 [1.9]	0.5 [0.8]	1.7 [2.1]
					-2	6.5 [2.9]	1.1 [1.6]	1.0 [1.2]	0.3 [0.7]	1.0 [1.3]	0.4 [0.8]	1.0 [1.5]
					0	1.3 [2.0]	0.9 [1.7]	0.4 [0.8]	0.2 [0.5]	0.4 [0.9]	0.3 [0.8]	0.3 [0.9]
					2	8.4 [5.6]	1.1 [1.7]	0.9 [1.0]	0.3 [0.7]	0.7 [1.2]	0.5 [1.0]	0.8 [1.7]
					6	12.1 [5.0]	1.0 [1.3]	0.7 [1.4]	0.2 [0.6]	0.9 [0.9]	0.4 [1.1]	0.9 [1.2]
				Non-linear	-6	12.2 [5.2]	1.9 [2.3]	1.8 [2.2]	0.5 [1.1]	1.3 [1.8]	0.6 [1.3]	
					-2	8.6 [4.7]	1.4 [1.6]	1.5 [2.0]	0.5 [1.3]	1.0 [1.6]	0.4 [1.3]	
					0	1.8 [2.3]	1.2 [2.3]	0.7 [1.9]	0.1 [0.9]	0.4 [1.2]	0.4 [1.1]	
					2	11.3 [6.4]	1.5 [2.3]	1.1 [2.2]	0.3 [1.3]	0.8 [1.4]	0.6 [1.6]	
					6	15.7 [7.4]	0.9 [1.8]	0.9 [1.8]	0.2 [1.0]	0.8 [1.5]	0.4 [1.6]	
			Non-linear	Linear	-6	5.6 [2.6]	0.9 [1.6]	1.1 [1.4]	0.2 [0.8]	0.8 [0.9]	0.8 [1.8]	0.7 [1.3]
					-2	4.0 [2.3]	1.2 [1.6]	1.1 [1.4]	0.3 [0.8]	0.6 [0.9]	0.7 [1.4]	0.6 [1.6]
					0	1.7 [1.7]	1.1 [1.3]	0.6 [1.1]	0.2 [0.6]	0.4 [0.7]	0.4 [1.2]	0.6 [1.6]
					2	14.2 [2.6]	0.8 [1.2]	1.7 [1.9]	-0.0 [0.6]	0.4 [0.9]	0.4 [1.4]	0.3 [0.9]
					6	18.5 [4.5]	0.9 [1.3]	2.5 [1.9]	0.1 [0.6]	0.4 [0.9]	1.2 [1.9]	0.5 [1.0]
				Non-linear	-6	3.5 [2.8]	1.1 [1.7]	1.6 [1.6]	0.8 [1.7]	0.7 [1.1]	2.3 [2.2]	
					-2	1.6 [1.8]	0.9 [1.9]	1.4 [1.8]	0.5 [1.2]	0.4 [1.0]	1.7 [2.0]	
					0	1.2 [1.9]	1.3 [1.8]	1.0 [1.7]	0.4 [0.9]	0.4 [1.1]	0.9 [1.5]	
					2	5.6 [6.3]	2.0 [1.9]	0.7 [1.9]	0.1 [0.8]	0.5 [0.9]	0.6 [1.5]	
					6	12.2 [8.2]	2.2 [2.2]	0.8 [1.9]	0.3 [0.9]	0.6 [1.0]	0.6 [1.4]	

Table B.4: Simulation Results (10 Covariates, Missingness depends on X)

						Median AOPE [IQR]						
D	Missingness	X distr	Boundary	Kernel	β	cc	dr	dw	mi	knn	mi3	socp
10	X	Normal	Linear	Linear	-6	2.1 [2.4]	1.4 [1.5]	0.9 [1.0]	0.6 [0.9]	0.8 [1.1]	0.8 [1.2]	0.5 [1.3]
					-2	1.6 [2.1]	1.3 [1.4]	0.6 [1.2]	0.4 [0.9]	0.7 [1.1]	0.8 [1.2]	0.5 [1.1]
					0	1.6 [1.5]	1.7 [1.8]	0.5 [1.1]	0.3 [0.8]	0.5 [1.0]	0.6 [0.8]	0.5 [1.1]
					2	1.7 [1.8]	1.4 [1.6]	0.7 [0.9]	0.6 [0.9]	0.6 [0.9]	0.6 [1.3]	0.5 [1.1]
					6	2.0 [2.9]	1.5 [1.7]	0.8 [1.1]	0.8 [0.9]	1.0 [1.3]	1.1 [1.3]	0.5 [1.1]
				Non-linear	-6	3.4 [3.0]	2.4 [2.9]	1.3 [2.1]	0.8 [1.4]	1.1 [1.2]	1.2 [2.3]	
					-2	2.5 [3.2]	2.5 [2.8]	1.2 [2.1]	0.8 [1.7]	0.9 [1.5]	0.8 [1.7]	
					0	1.8 [2.1]	2.3 [1.8]	1.0 [1.7]	0.4 [1.1]	0.6 [0.9]	0.6 [1.0]	
					2	2.7 [2.8]	2.3 [2.5]	1.3 [2.5]	0.6 [1.3]	0.8 [1.3]	1.0 [1.4]	
					6	2.9 [2.6]	2.2 [2.0]	1.6 [2.2]	0.8 [1.3]	1.2 [1.5]	1.6 [2.0]	
			Non-linear	Linear	-6	-6.1 [3.5]	-0.8 [2.0]	-3.9 [5.1]	-0.0 [1.8]	0.0 [1.0]	0.1 [1.9]	0.0 [0.9]
					-2	-5.4 [3.5]	-0.1 [1.8]	-2.9 [3.9]	-0.1 [1.3]	0.0 [0.8]	0.0 [1.1]	0.0 [1.2]
					0	0.6 [2.8]	0.1 [1.3]	0.0 [1.8]	0.0 [0.7]	0.0 [0.7]	0.0 [1.1]	-0.1 [0.9]
					2	-2.7 [4.0]	0.7 [2.2]	-0.1 [4.2]	0.6 [1.7]	0.2 [1.1]	0.0 [1.3]	0.0 [1.2]
					6	-3.9 [3.3]	0.5 [1.9]	-0.7 [3.6]	0.4 [1.5]	0.0 [1.1]	0.5 [1.6]	0.0 [1.1]
				Non-linear	-6	7.0 [3.8]	3.8 [2.8]	1.0 [2.2]	0.8 [1.3]	0.8 [1.4]	1.4 [2.6]	
					-2	4.3 [2.4]	3.4 [2.9]	1.2 [1.7]	0.8 [1.3]	0.5 [1.3]	1.1 [1.8]	
					0	2.4 [1.9]	3.8 [3.0]	1.3 [1.7]	0.7 [1.4]	0.4 [1.3]	0.8 [1.4]	
					2	3.2 [2.5]	4.2 [2.9]	0.8 [2.0]	0.7 [1.3]	0.4 [1.5]	1.0 [1.6]	
					6	3.8 [3.7]	4.6 [2.5]	1.1 [2.0]	0.5 [1.4]	0.4 [1.3]	1.4 [2.3]	
		χ^2	Linear	Linear	-6	1.3 [1.9]	1.3 [1.7]	0.4 [1.0]	0.1 [0.6]	0.4 [1.0]	0.5 [0.8]	0.5 [1.2]
					-2	1.4 [2.0]	1.2 [1.6]	0.4 [0.9]	0.2 [0.6]	0.4 [0.8]	0.3 [0.9]	0.6 [1.2]
					0	1.3 [1.3]	0.8 [1.3]	0.4 [0.9]	0.1 [0.7]	0.3 [0.9]	0.4 [0.9]	0.4 [1.0]
					2	2.6 [2.9]	0.8 [1.4]	0.5 [0.9]	0.2 [0.7]	0.6 [0.9]	0.5 [1.1]	0.5 [1.2]
					6	4.0 [4.2]	1.0 [1.3]	0.5 [1.0]	0.1 [0.8]	0.7 [1.4]	0.6 [1.0]	0.8 [1.4]
				Non-linear	-6	1.5 [2.4]	1.6 [2.3]	0.7 [1.9]	0.2 [1.1]	0.3 [1.1]	0.4 [1.3]	
					-2	1.5 [2.2]	1.5 [1.9]	0.6 [1.5]	0.2 [1.0]	0.4 [1.1]	0.5 [1.2]	
					0	1.7 [2.3]	1.5 [2.2]	0.7 [1.7]	0.2 [0.9]	0.4 [1.1]	0.3 [1.0]	
					2	5.3 [6.8]	1.7 [1.9]	0.6 [1.7]	0.4 [1.1]	1.1 [1.4]	0.7 [1.6]	
					6	7.8 [7.1]	1.6 [2.0]	0.6 [1.3]	0.4 [1.0]	1.0 [1.3]	0.9 [1.7]	
			Non-linear	Linear	-6	1.3 [1.7]	1.2 [1.4]	0.5 [1.0]	0.1 [0.7]	0.4 [0.8]	0.2 [1.0]	0.4 [1.0]
					-2	1.4 [2.0]	1.0 [1.5]	0.5 [1.2]	0.1 [0.7]	0.4 [1.1]	0.3 [0.9]	0.4 [1.0]
					0	1.8 [1.9]	1.0 [1.5]	0.5 [0.8]	0.1 [0.6]	0.2 [0.8]	0.3 [0.7]	0.4 [1.1]
					2	8.6 [4.0]	1.0 [1.3]	0.7 [1.4]	0.1 [0.6]	0.5 [1.2]	0.8 [1.8]	0.6 [2.0]
					6	12.0 [5.6]	0.9 [1.0]	0.7 [1.3]	0.2 [0.6]	0.4 [1.0]	1.8 [2.4]	0.6 [1.3]
				Non-linear	-6	1.2 [1.8]	1.0 [2.1]	0.9 [1.5]	0.4 [0.9]	0.3 [1.2]	1.2 [1.8]	
					-2	1.1 [1.8]	1.3 [1.9]	1.0 [1.9]	0.3 [0.9]	0.4 [1.2]	1.3 [1.9]	
					0	1.1 [2.1]	1.1 [1.9]	1.1 [2.1]	0.3 [1.2]	0.2 [1.0]	1.2 [1.6]	
					2	1.9 [2.4]	2.0 [2.3]	0.8 [1.4]	0.3 [0.8]	0.6 [1.1]	1.2 [1.7]	
					6	3.0 [2.8]	2.4 [2.0]	0.4 [1.8]	0.2 [1.0]	0.5 [1.2]	1.8 [2.2]	

APPENDIX C: PROPOSITIONS AND RESULTS FOR PARTIALLY OBSERVED OUTCOME SUPPORT VECTOR MACHINES

C.1 Proposition 1

Define

$$\tilde{p}(k, \mathbf{f}) = \tilde{p}\left(Y = k \,|\, \mathbf{X}, \mathbf{f}\right) = \frac{\exp\{-L[Y = k, \mathbf{f}(\mathbf{X})]\}}{\sum_{j=1}^{K} \exp\{-L[Y = j, \mathbf{f}(\mathbf{X})]\}}$$

as a quasi-probability distribution over Y as a function of the SVM classifier and loss. The function

$$g(\mathbf{f}) = \sum_{i=1}^{n} \begin{cases} \ln \tilde{p}\left(y_i, \mathbf{f}\right) & \textit{known label} \\ 0 & \textit{unknown label} \end{cases}$$

is the observed data likelihood. For successive solutions of the EM algorithm, the following holds:

$$g\left(\mathbf{f}^{(m)}\right) \leq g\left(\mathbf{f}^{(m+1)}\right).$$

Because g is bounded above, the sequence

$$g\left(\mathbf{f}^{(1)}\right), g\left(\mathbf{f}^{(2)}\right), \ldots$$

converges to a local maximum.

Proof. Let

$$h\left(\mathbf{f}|\mathbf{f}^{(m)}\right) = \sum_{i=1}^{n} \begin{cases} \ln \tilde{p}\left(y_i, \mathbf{f}\right) & \text{known label} \\ \sum_{j=1}^{K} \tilde{p}\left(j, \mathbf{f}^{(m)}\right) \ln \tilde{p}\left(j, \mathbf{f}\right) & \text{unknown label.} \end{cases}$$

Further, let $a\left(\mathbf{f}|\mathbf{f}^{(m)}\right) = h\left(\mathbf{f}|\mathbf{f}^{(m)}\right) + g\left(\mathbf{f}^{(m)}\right) - h\left(\mathbf{f}^{(m)}|\mathbf{f}^{(m)}\right)$. By construction, the following

inequality/equality hold:

$$(A) \qquad a\left(\mathbf{f}^{(m+1)}|\mathbf{f}^{(m)}\right) \geq a\left(\mathbf{f}^{(m)}|\mathbf{f}^{(m)}\right)$$
$$(B) \qquad a\left(\mathbf{f}^{(m)}|\mathbf{f}^{(m)}\right) = g\left(\mathbf{f}^{(m)}\right).$$

We will show

$$(C) \qquad g\left(\mathbf{f}\right) \geq a\left(\mathbf{f}|\mathbf{f}^{(m)}\right) \text{ for all } \mathbf{f}.$$

We derive (C) in the following way:

$$g\left(\mathbf{f}\right) - a\left(\mathbf{f}|\mathbf{f}^{(m)}\right) = \sum_{i=1}^{n} \begin{cases} 0 & \text{label known} \\ \sum_{j=1}^{K} \tilde{p}\left(j, \mathbf{f}^{(m)}\right) \ln \frac{\tilde{p}\left(j, \mathbf{f}^{(m)}\right)}{\tilde{p}\left(j, \mathbf{f}\right)} & \text{label unknown} \end{cases}$$

Because KL divergence $E_P\left[\ln\left(\frac{P}{Q}\right)\right] \geq 0$ it follows that

$$\sum_{j=1}^{K} \tilde{p}\left(j, \mathbf{f}^{(m)}\right) \ln \frac{\tilde{p}\left(j, \mathbf{f}^{(m)}\right)}{\tilde{p}\left(j, \mathbf{f}\right)} \geq 0$$

which implies that $g\left(\mathbf{f}\right) - h\left(\mathbf{f}|\mathbf{f}^{(m)}\right) \geq 0$.

Using (A), (B), and (C), the following inequality holds:

$$g\left(\mathbf{f}^{(m)}\right) = a\left(\mathbf{f}^{(m)}|\mathbf{f}^{(m)}\right) \leq a\left(\mathbf{f}^{(m+1)}|\mathbf{f}^{(m)}\right) \leq g\left(\mathbf{f}^{(m+1)}\right).$$

Because $g\left(\mathbf{f}^{(m)}\right) \leq g\left(\mathbf{f}^{(m+1)}\right)$ and because g is bounded above, the sequence

$$g\left(\mathbf{f}^{(1)}\right), g\left(\mathbf{f}^{(2)}\right), \ldots$$

converges to a local maximum.

C.2 Proposition 2

Let $P(Y, X)$ denote a distribution over $Y \times X$ where $Y \in \{\pm 1\}$ and $X \in \mathbb{R}^d$. Let R be a random variable which is one if Y is observed and zero otherwise. Define

$$\tilde{p}(k, \mathbf{f}) = \tilde{p}\left(Y = k \,|\, \mathbf{X}, \mathbf{f}\right) = \frac{\exp\{-L[Y = k, \mathbf{f}(\mathbf{X})]\}}{\sum_{j=1}^{K} \exp\{-L[Y = j, \mathbf{f}(\mathbf{X})]\}}$$

as a quasi-probability distribution over Y as a function of the SVM classifier and loss. If

(a) $\widetilde{P}(Y = c|\mathbf{x}) = P(Y = c|\mathbf{x})$*, and*

(b) the multi-class loss function generates a Fisher consistent classifier,

$$\mathbf{f}_{bayes} = \arg\min_{\mathbf{f}} E\{L[Y, \mathbf{f}(\mathbf{X})]\},$$

then the EM multi-class loss function,

$$\widetilde{L}[y_i, \mathbf{f}(\mathbf{x}_i)] = \begin{cases} L[y_i, \mathbf{f}(\mathbf{x}_i)] & \textit{class label known} \\ \sum_{j=1}^{k} \widetilde{P}(Y = c|\mathbf{x}) L[y_i = j, \mathbf{f}(\mathbf{x}_i)] & \textit{class label unknown} \end{cases},$$

also generates a Fisher consistent classifier.

Proof. We show that $E\{\widetilde{L}[Y, \mathbf{f}(\mathbf{X})] \,|\, \mathbf{X} = \mathbf{x}\} = E\{L[Y, \mathbf{f}(\mathbf{X})] \,|\, \mathbf{X} = \mathbf{x}\}$. Because $L[Y, \mathbf{f}(\mathbf{X})]$

generates a Fisher consistent classifier, so does $\widetilde{L}[Y, \mathbf{f}(\mathbf{X})]$.

$$E\{\widetilde{L}[Y, \mathbf{f}(\mathbf{X})] \,|\, \mathbf{X} = \mathbf{x}\} = P(R = 1) \underbrace{E\{\widetilde{L}[Y, \mathbf{f}(\mathbf{X})] \,|\, R = 1, \mathbf{X} = \mathbf{x}\}}_{A} + P(R = 0) \underbrace{E\{\widetilde{L}[Y, \mathbf{f}(\mathbf{X})] \,|\, R = 0, \mathbf{X} = \mathbf{x}\}}_{B}$$

$$A = E\{L[Y, \mathbf{f}(\mathbf{X})] \,|\, R = 1, \mathbf{X} = \mathbf{x}\}$$

$$\begin{aligned} B &= E\left\{\sum_{k=1}^{K} \widetilde{P}(Y = k|\mathbf{x}) L[Y = k, \mathbf{f}(\mathbf{X})] \,|\, R = 0, \mathbf{X} = \mathbf{x}\right\} \\ &= \sum_{k=1}^{K} \widetilde{P}(Y = k|\mathbf{x}) E\{L[Y = k, \mathbf{f}(\mathbf{X})] \,|\, R = 0, \mathbf{X} = \mathbf{x}\} \\ &= \sum_{k=1}^{K} P(Y = k|\mathbf{x}) E\{L[Y = k, \mathbf{f}(\mathbf{X})] \,|\, R = 0, \mathbf{X} = \mathbf{x}\} \\ &= E\{L[Y, \mathbf{f}(\mathbf{X})] \,|\, R = 0, \mathbf{X} = \mathbf{x}\} \end{aligned}$$

Thus,

$$\begin{aligned} E\{\widetilde{L}[Y, \mathbf{f}(\mathbf{X})] \,|\, \mathbf{X} = \mathbf{x}\} &= P(R = 1) E\{L[Y, \mathbf{f}(\mathbf{X})] \,|\, R = 1, \mathbf{X} = \mathbf{x}\} \\ &\quad + P(R = 0) E\{L[Y, \mathbf{f}(\mathbf{X})] \,|\, R = 0, \mathbf{X} = \mathbf{x}\} \\ &= E\{L[Y, \mathbf{f}(\mathbf{X})] \,|\, \mathbf{X} = \mathbf{x}\} \end{aligned}$$

C.3 Constructing a weighted multi-class SVM classifier

Consider a training set $\mathcal{T}_n = \{(\mathbf{x}_1, y_1, w_1), \ldots, (\mathbf{x}_n, y_n, w_n)\}$ where $\mathbf{x}_i \in \mathbb{R}^d$, $y_i \in \{1, \ldots, k\}$, and $w_i \in [0, \infty)$. Here we show how the the reinforced multi-class support vector machine (RMSVM) introduced in [69] can be expressed in terms of a weighted empirical risk in which each observation is weighted by w_i. Constructed from $\mathcal{T}_n$, the RMSVM classifier with kernel function κ is of the form $\mathbf{f}(\mathbf{z}) = [f_1(\mathbf{z}) \ldots f_k(\mathbf{z})]$ where

$$f_j(\mathbf{z}) = b_j + \sum_{i=1}^{n} \kappa(\mathbf{z}, \mathbf{x}_i) v_{ij}.$$

The function is characterized by the matrix of coefficients V and intercept vector $\mathbf{b}$. Let K be the kernel matrix with $K_{ij} = \kappa(\mathbf{x}_i, \mathbf{x}_j)$. For notational ease, let $K_i = \text{row}_i(K)$ and $\mathbf{v}_j = \text{col}_j(V)$. In the weighted setting, the RMSVM is the solution to the following objective function:

$$\hat{\mathbf{f}} = \arg\min_{\mathbf{f}} \sum_{j=1}^{k} \frac{1}{2} \mathbf{v}_j^t K \mathbf{v}_j + \sum_{i=1}^{n} w_i \left\{ \gamma [z - b_{y_i} - K_i \mathbf{v}_{y_i}]_+ + (1 - \gamma) \sum_{j \neq y_i}^{k} [1 + b_j + K_j \mathbf{v}_j]_+ \right\}$$

$$s.t. \qquad \sum_{j=1}^{k} \left[b_j + K_i \mathbf{v}_j \right] = 0 \quad \text{for } i = 1, \ldots, n.$$

Note that $w_i = \frac{1}{\lambda n}$ is the standard, uniformly weighted objective function.

Here we show that the solution can be computed from a quadratic programing problem involving the dual of the objective function. Following the steps similar to [69], we introduce slack variables:

$$\sum_{j=1}^{k} \frac{1}{2} \mathbf{v}_j^t K \mathbf{v}_j + \sum_{i=1}^{n} w_i \left\{ \gamma \xi_{i,y_i} + (1 - \gamma) \sum_{j \neq y_i}^{k} \xi_{ij} \right\}$$

$$
\begin{aligned}
s.t. \qquad & \sum_{j=1}^{k} \left[b_j + K_i \mathbf{v}_j \right] = 0 && i = 1, \ldots, n \\
& \xi_{ij} \geq 0 && i = 1, \ldots, n \\
& \xi_{i,y_i} \geq z - b_{y_i} - K_i \mathbf{v}_{y_i} && i = 1, \ldots, n \\
& \xi_{i,y_i} \geq 1 + b_i + K_i \mathbf{v}_i && i \neq j,\ i = 1, \ldots, n.
\end{aligned}
$$

Then construct the Lagrangian:

$$
\begin{aligned}
\mathcal{L} = & \sum_{j=1}^{k} \frac{1}{2} \mathbf{v}_j^t K \mathbf{v}_j \\
& + \sum_{i=1}^{n} w_i \left\{ \gamma \xi_{i,y_i} + (1-\gamma) \sum_{j \neq y_i}^{k} \xi_{ij} \right\} \\
& - \sum_{i=1}^{n} \sum_{j=1}^{k} \tau_{ij} \xi_{ij} \\
& + \sum_{i=1}^{n} \delta_i \sum_{j=1}^{k} \left[b_j + K_i \mathbf{v}_j \right] \\
& - \sum_{i=1}^{n} \alpha_{i,y_i} \left[\xi_{i,y_i} - z + b_{y_i} + K_i \mathbf{v}_{y_i} \right] \\
& - \sum_{i=1}^{n} \sum_{j \neq y_i}^{k} \alpha_{ij} \left[\xi_{ij} - 1 - b_j - K_i \mathbf{v}_j \right].
\end{aligned}
$$

Define the matrices A and B so that

$$
A_{ij} = \begin{cases} \alpha_{ij} & y_i = j \\ -\alpha_{ij} & y_i \neq j \end{cases} \qquad B_{ij} = \begin{cases} \gamma & y_i = j \\ 1-\gamma & y_i \neq j. \end{cases}
$$

Group terms in Lagrangian by ξ_{ij}, b_j, and $\mathbf{v}_j$:

$$\begin{aligned}\mathcal{L} = &\sum_{j=1}^{k} \frac{1}{2}\mathbf{v}_j^t K \mathbf{v}_j + \sum_{i=1}^{n}\sum_{j=1}^{k}\left[\delta_i - A_{ij}\right] K_i \mathbf{v}_j \\ &+ \sum_{i=1}^{n}\left[w_i B_{ij} - \tau_{ij} - \alpha_{ij}\right]\xi_{ij} \\ &+ \sum_{i=1}^{n}\sum_{j=1}^{k}\left[\delta_i - A_{ij}\right] b_j \\ &+ \sum_{i=1}^{n} A_{ij}\left[z+1\right] \\ &- \sum_{i=1}^{n}\sum_{j=1}^{k} A_{ij}\end{aligned}$$

The derivatives of the Lagrangian with respect to the primal variables ξ_{ij}, b_j, and $\mathbf{v}_j$ are:

$$\begin{aligned}\frac{\partial \mathcal{L}}{\partial \xi_{ij}} &= w_i B_{ij} - \alpha_{ij} - \tau_{ij} \\ \frac{\partial \mathcal{L}}{\partial b_j} &= \sum_{i=1}^{n} \delta_i - A_{ij} \\ \frac{\partial \mathcal{L}}{\partial \mathbf{v}_j} &= \mathbf{v}_j^t K + \sum_{i=1}^{n}\left[\delta_i - A_{ij}\right] K_i\end{aligned}$$

We set derivatives to zero and solve to get

$$\begin{aligned}\tau_{ij} &= w_i B_{ij} - \alpha_{ij} \qquad\qquad i = 1,\ldots,n \;\; j = 1,\ldots,k \\ \delta &= \frac{1}{k} A \mathbb{1}_k \\ V &= A\left[I_k - \frac{1}{k}\mathbb{1}_k \mathbb{1}_k^t\right],\end{aligned}$$

along with the conditions that

$$\begin{aligned} 0 \le \alpha_{ij} \le w_i B_{ij} \qquad & i = 1, \ldots, n \;\; j = 1, \ldots, k \\ \sum_i^n A_{ij} = \sum_i^n A_{i,1} \qquad & j = 2, \ldots, k. \end{aligned}$$

Note that the solution of V includes the residual projection matrix

$$R = I_k - \frac{1}{k} \mathbb{1}_k \mathbb{1}_k^t,$$

and that $\mathbf{v}_j = A \operatorname{col}_j(R)$. Plugging the solutions back into the Lagrangian, the dual problem is

$$\begin{aligned} \max_{\alpha_{ij}} & -\frac{1}{2} \sum_{j=1}^k R_j^t A^t K A R_j + \sum_{i=1}^n A_{ij} [z+1] - \sum_{i=1}^n \sum_{j=1}^k A_{ij} \\ s.t. \quad & 0 \le \alpha_{ij} \le w_i B_{ij} \qquad i = 1, \ldots, n \;\; j = 1, \ldots, k \\ & \sum_i^n A_{ij} = \sum_i^n A_{i,1} \qquad j = 2, \ldots, k. \end{aligned}$$

With the goal of expressing the problem as a quadratic programing problem, note that

$$\begin{aligned} \sum_{j=1}^k R_j^t A^t K A R_j &= \operatorname{vec}(AR)^t (I_k \otimes K) \operatorname{vec}(AR) \\ &= \operatorname{vec}(A)^t \underbrace{(R^t \otimes I_n^t)(I_k \otimes K)(R \otimes I_n)}_{Q} \operatorname{vec}(A). \end{aligned}$$

Also define vector $\mathbf{g}$,

$$\mathbf{g} = \operatorname{vec}(G) \text{ where } G_{ij} = \begin{cases} 1 & j \neq y_i \\ -z & j = y_i \end{cases}$$

So the quadratic programing problem is

$$\min_{A_{ij}} \frac{1}{2} \operatorname{vec}(A)^t Q \operatorname{vec}(A) + \mathbf{g}^t \operatorname{vec}(A)$$

$$\begin{aligned} s.t. \quad & 0 \le A_{ij} \le w_i B_{ij} && i = 1, \ldots, n \;\; j = y_i \\ & -w_i B_{ij} \le A_{ij} \le 0 && i = 1, \ldots, n \;\; j \neq y_i \\ & \sum_i^n A_{ij} = \sum_i^n A_{i,1} && j = 2, \ldots, k. \end{aligned}$$

And the solution $V^* = A^* R$ where A^* is the solution to the quadratic programing problem. The parameter b can be found in the way described in [69].

C.4 Constructing a multi-class SVM classifier with fewer basis functions

Consider a training set of size n, $\mathcal{T}_n = \{(\mathbf{x}_1, y_1, w_1), \ldots, (\mathbf{x}_n, y_n, w_n)\}$ where $\mathbf{x}_i \in \mathbb{R}^d$, $y_i \in \{1, \ldots, k\}$, and $w_i \in [0, \infty)$. The RMSVM classifier constructed from $\mathcal{T}_n$ with kernel function κ is of the form $\mathbf{f}(\mathbf{z}) = [f_1(\mathbf{z}) \ldots f_k(\mathbf{z})]$ where

$$f_j(\mathbf{z}) = b_j + \sum_{\mathbf{x}_i \in \mathcal{T}_n} \kappa(\mathbf{z}, \mathbf{x}_i) v_{ij}.$$

Each observation acts as a basis function, $\kappa(\mathbf{z}, \mathbf{x}_i)$, in the RMSVM classifier. Here we show how to compute a weighted RMSVM in which only the first m observations of the training set act as basis functions but all observations contribute to the weighted empirical risk. Specifically, we show how to find

$$\hat{\mathbf{f}} = \arg\min_{\mathbf{f} \in \mathcal{H}_{\mathcal{T}_m}} \|\mathbf{f}\|_{\mathcal{H}_{\mathcal{T}_m}} + \sum_{i \in \mathcal{T}_n} w_i L[y_i, \mathbf{f}(\mathbf{x}_i)].$$

Let $m < n$, and let $\mathcal{T}_m$ denote the first m observations of $\mathcal{T}_n$. Let $K_{ij} = \kappa(\mathbf{x}_i, \mathbf{x}_j)$ denote the kernel matrix constructed from $\mathcal{T}_n$ and kernel function κ. Let K^{nm} denote the first m columns of K, and let K^{mm} denote the first m columns and first m rows. The objective function is:

$$\hat{\mathbf{f}} = \arg\min_{\mathbf{f} \in \mathcal{H}_{\mathcal{T}_m}} \sum_{j=1}^{k} \frac{1}{2} \mathbf{v}_j^t K^{mm} \mathbf{v}_j + \sum_{i=1}^{n} w_i \left\{ \gamma [z - b_{y_i} - K_i^{nm} \mathbf{v}_{y_i}]_+ + (1 - \gamma) \sum_{j \neq y_i}^{k} [1 + b_j + K_j^{nm} \mathbf{v}_j]_+ \right\}$$

$$s.t. \qquad \sum_{j=1}^{k} \left[b_j + K_i^{nm} \mathbf{v}_j \right] = 0 \quad \text{for } i = 1, \ldots, n.$$

As before, the notation $K_i^{nm} = \text{row}_i(K^{nm})$ and $\mathbf{v}_j = \text{col}_j(V)$.

Here we show that the solution can be computed from a quadratic programing problem involving the dual of the objective function. Following the steps similar to

[69] and proposition, we introduce slack variables:

$$\sum_{j=1}^{k} \frac{1}{2} \mathbf{v}_j^t K^{mm} \mathbf{v}_j + \sum_{i=1}^{n} w_i \left\{ \gamma \xi_{i,y_i} + (1-\gamma) \sum_{j \neq y_i}^{k} \xi_{ij} \right\}$$

$$\begin{aligned}
s.t. \qquad & \sum_{j=1}^{k} \left[b_j + K_i^{nm} \mathbf{v}_j \right] = 0 && i = 1, \ldots, n \\
& \xi_{ij} \geq 0 && i = 1, \ldots, n \\
& \xi_{i,y_i} \geq z - b_{y_i} - K_i^{nm} \mathbf{v}_{y_i} && i = 1, \ldots, n \\
& \xi_{i,y_i} \geq 1 + b_i + K_i^{nm} \mathbf{v}_i && i \neq j,\ i = 1, \ldots, n.
\end{aligned}$$

Then construct the Lagrangian:

$$\begin{aligned}
\mathcal{L} = & \sum_{j=1}^{k} \frac{1}{2} \mathbf{v}_j^t K^{mm} \mathbf{v}_j \\
& + \sum_{i=1}^{n} w_i \left\{ \gamma \xi_{i,y_i} + (1-\gamma) \sum_{j \neq y_i}^{k} \xi_{ij} \right\} \\
& - \sum_{i=1}^{n} \sum_{j=1}^{k} \tau_{ij} \xi_{ij} \\
& + \sum_{i=1}^{n} \delta_i \sum_{j=1}^{k} \left[b_j + K_i^{nm} \mathbf{v}_j \right] \\
& - \sum_{i=1}^{n} \alpha_{i,y_i} \left[\xi_{i,y_i} - z + b_{y_i} + K_i^{nm} \mathbf{v}_{y_i} \right] \\
& - \sum_{i=1}^{n} \sum_{j \neq y_i}^{k} \alpha_{ij} \left[\xi_{ij} - 1 - b_j - K_i^{nm} \mathbf{v}_j \right].
\end{aligned}$$

Define the matrices A and B so that

$$A_{ij} = \begin{cases} \alpha_{ij} & y_i = j \\ -\alpha_{ij} & y_i \neq j \end{cases} \qquad B_{ij} = \begin{cases} \gamma & y_i = j \\ 1-\gamma & y_i \neq j. \end{cases}$$

Group terms in Lagrangian by ξ_{ij}, b_j, and $\mathbf{v}_j$:

$$\begin{aligned}\mathcal{L} = &\sum_{j=1}^{k} \frac{1}{2}\mathbf{v}_j^t K^{mm}\mathbf{v}_j + \sum_{i=1}^{n}\sum_{j=1}^{k}\Big[\delta_i - A_{ij}\Big]K_i^{nm}\mathbf{v}_j \\ &+ \sum_{i=1}^{n}\Big[w_i B_{ij} - \tau_{ij} - \alpha_{ij}\Big]\xi_{ij} \\ &+ \sum_{i=1}^{n}\sum_{j=1}^{k}\Big[\delta_i - A_{ij}\Big]b_j \\ &+ \sum_{i=1}^{n} A_{ij}\,[z+1] \\ &- \sum_{i=1}^{n}\sum_{j=1}^{k} A_{ij}\end{aligned}$$

The derivatives of the Lagrangian with respect to the primal variables ξ_{ij}, b_j, and $\mathbf{v}_j$ are:

$$\begin{aligned}\frac{\partial \mathcal{L}}{\partial \xi_{ij}} &= w_i B_{ij} - \alpha_{ij} - \tau_{ij} \\ \frac{\partial \mathcal{L}}{\partial b_j} &= \sum_{i=1}^{n} \delta_i - A_{ij} \\ \frac{\partial \mathcal{L}}{\partial \mathbf{v}_j} &= \mathbf{v}_j^t K^{mm} + \sum_{i=1}^{n}\Big[\delta_i - A_{ij}\Big]K_i^{nm}\end{aligned}$$

We set derivatives to zero and solve to get

$$\begin{aligned}\tau_{ij} &= w_i B_{ij} - \alpha_{ij} \qquad\qquad i = 1,\ldots,n \;\; j = 1,\ldots,k \\ \delta &= \frac{1}{k} A \mathbb{1}_k \\ V &= [K^{mm}]^{-}[K^{nm}]^t A\left[I_k - \frac{1}{k}\mathbb{1}_k\mathbb{1}_k^t\right],\end{aligned}$$

along with the conditions that

$$0 \le \alpha_{ij} \le w_i B_{ij} \qquad i = 1, \ldots, n \;\; j = 1, \ldots, k$$
$$\sum_i^n A_{ij} = \sum_i^n A_{i,1} \qquad j = 2, \ldots, k.$$

Let $P = [K^{mm}]^{-}[K^{nm}]^t$. Note that the solution of V includes the residual projection matrix

$$R = I_k - \frac{1}{k}\mathbb{1}_k\mathbb{1}_k^t,$$

and that $\mathbf{v}_j = PA\,\mathrm{col}_j(R)$. Plugging the solutions back into the Lagrangian, the dual problem is

$$\max_{\alpha_{ij}} -\frac{1}{2}\sum_{j=1}^{k} R_j^t A^t P^t K P A R_j + \sum_{i=1}^{n} A_{ij}\,[z+1] - \sum_{i=1}^{n}\sum_{j=1}^{k} A_{ij}$$
$$s.t. \quad 0 \le \alpha_{ij} \le w_i B_{ij} \qquad i = 1, \ldots, n \;\; j = 1, \ldots, k$$
$$\sum_i^n A_{ij} = \sum_i^n A_{i,1} \qquad j = 2, \ldots, k.$$

With the goal of expressing the problem as a quadratic programing problem, note that

$$\begin{aligned}\sum_{j=1}^{k} R_j^t A^t P^t K P A R_j &= \mathrm{vec}(AR)^t (I_k \otimes P^t K^{mm} P)\,\mathrm{vec}(AR) \\ &= \mathrm{vec}(A)^t \underbrace{(R^t \otimes I_n^t)(I_k \otimes P^t K^{mm} P)(R \otimes I_n)}_{Q}\,\mathrm{vec}(A).\end{aligned}$$

The matrix $P^t K^{mm} P$ reduces to $K^{nm}[K^{mm}]^{-}[K^{nm}]^t$. Also define vector $\mathbf{g}$,

$$\mathbf{g} = \mathrm{vec}(G) \text{ where } G_{ij} = \begin{cases} 1 & j \ne y_i \\ -z & j = y_i \end{cases}$$

So the quadratic programing problem is

$$\min_{A_{ij}} \frac{1}{2} \operatorname{vec}(A)^t Q \operatorname{vec}(A) + \mathbf{g}^t \operatorname{vec}(A)$$

$$\begin{aligned} s.t. \quad & 0 \le A_{ij} \le w_i B_{ij} && i = 1, \ldots, n \;\; j = y_i \\ & -w_i B_{ij} \le A_{ij} \le 0 && i = 1, \ldots, n \;\; j \neq y_i \\ & \sum_i^n A_{ij} = \sum_i^n A_{i,1} && j = 2, \ldots, k. \end{aligned}$$

And the solution $V^* = PA^*R$ where A^* is the solution to the quadratic programing problem. The parameter **b** can be found in the way described in [69].

C.5 Simulation Study Results for Partially-observed Outcomes

The following tables report simulation study results for the partially-observed settings. The competing methods are labeled with a -L or -N to denote linear kernel or nonlinear Gaussian kernel, respectively. The methods are:

Label	Method
Oracle	SVM trained with no missing data
CC	Complete Case SVM
Naive	Cluster-then-classify SVM
EM	The method proposed in this paper

Table C.1: Simulation results of methods for partially-observed outcomes, $N = 40$ for smallest class

				Prediction Accuracy [IQR]							
D	BALANCE	% OBSD	BAYES RISK	Oracle-L	CC-L	Naive-L	EM-L	Oracle-N	CC-N	Naive-N	EM-N
2	Equal	5	0.05	94.8 [00]	89.8 [23]	93.9 [11]	89.1 [15]	94.6 [01]	86.0 [43]	93.3 [18]	92.7 [04]
			0.15	84.5 [00]	76.5 [19]	82.8 [13]	81.6 [06]	84.2 [01]	77.6 [26]	79.0 [16]	82.7 [04]
			0.35	64.6 [01]	55.2 [13]	56.5 [12]	61.7 [04]	63.6 [02]	51.5 [16]	54.3 [11]	59.0 [05]
		25	0.05	94.7 [00]	94.2 [02]	94.7 [00]	91.5 [14]	94.6 [01]	93.7 [07]	94.4 [01]	93.0 [04]
			0.15	84.5 [00]	83.6 [02]	84.2 [01]	82.5 [04]	84.2 [01]	81.8 [06]	83.8 [02]	82.6 [02]
			0.35	64.5 [01]	63.0 [04]	63.9 [01]	63.0 [02]	63.7 [02]	59.2 [08]	62.0 [03]	61.9 [03]
	Unequal	5	0.05	94.7 [00]	91.3 [19]	94.5 [01]	90.0 [07]	94.6 [01]	91.9 [07]	93.9 [01]	93.6 [02]
			0.15	84.6 [00]	79.3 [09]	83.6 [03]	80.5 [04]	84.4 [01]	78.4 [16]	82.3 [06]	82.7 [03]
			0.35	64.6 [01]	59.4 [08]	62.0 [04]	62.3 [02]	63.7 [02]	56.6 [11]	59.6 [08]	60.7 [04]
		25	0.05	94.8 [00]	94.3 [01]	94.7 [00]	91.5 [07]	94.6 [01]	94.1 [01]	94.5 [01]	93.7 [02]
			0.15	84.6 [01]	84.2 [02]	84.5 [01]	82.3 [04]	84.3 [01]	83.7 [02]	84.1 [01]	83.6 [01]
			0.35	64.6 [01]	64.1 [02]	64.2 [01]	63.7 [02]	63.6 [02]	61.4 [04]	62.5 [03]	63.2 [03]
10	Equal	5	0.05	93.9 [01]	84.1 [27]	90.7 [27]	92.0 [03]	93.9 [01]	77.1 [41]	91.6 [25]	91.6 [04]
			0.15	83.2 [01]	56.5 [20]	75.5 [23]	81.0 [03]	82.9 [02]	48.9 [19]	75.8 [23]	77.3 [08]
			0.35	62.1 [03]	47.0 [06]	48.5 [14]	57.5 [04]	61.2 [03]	31.5 [10]	44.6 [14]	49.8 [10]
		25	0.05	94.0 [01]	91.9 [03]	93.6 [01]	92.7 [02]	93.9 [01]	92.0 [04]	93.5 [01]	92.7 [02]
			0.15	83.1 [01]	79.9 [04]	81.9 [03]	82.1 [03]	82.7 [02]	78.0 [08]	81.6 [03]	80.7 [04]
			0.35	61.8 [03]	55.4 [08]	58.9 [05]	59.3 [05]	60.8 [03]	55.0 [06]	57.3 [05]	58.0 [07]
	Unequal	5	0.05	94.0 [01]	83.9 [21]	92.2 [03]	93.1 [03]	94.1 [01]	69.1 [27]	92.4 [04]	92.9 [02]
			0.15	83.5 [01]	65.7 [18]	79.2 [09]	81.9 [03]	83.1 [01]	56.4 [17]	78.2 [16]	79.8 [05]
			0.35	62.6 [02]	46.1 [09]	52.3 [12]	60.0 [04]	62.0 [02]	40.7 [13]	50.3 [13]	52.8 [07]
		25	0.05	94.1 [01]	92.7 [03]	93.9 [01]	93.1 [02]	94.0 [01]	92.4 [03]	93.8 [01]	93.2 [01]
			0.15	83.4 [02]	80.1 [04]	82.6 [02]	82.8 [02]	83.3 [01]	80.1 [04]	82.4 [03]	82.1 [03]
			0.35	62.1 [02]	56.3 [06]	59.6 [05]	60.5 [05]	61.3 [03]	55.4 [06]	58.6 [05]	58.5 [05]

Table C.2: Simulation results of methods for partially-observed outcomes, $N = 100$ for smallest class

				Prediction Accuracy [IQR]							
D	BALANCE	% OBSD	BAYES RISK	Oracle-L	CC-L	Naive-L	EM-L	Oracle-N	CC-N	Naive-N	EM-N
2	Equal	5	0.05	94.8 [00]	89.8 [23]	93.9 [11]	89.1 [15]	94.6 [01]	86.0 [43]	93.3 [18]	92.7 [04]
			0.15	84.5 [00]	76.5 [19]	82.8 [13]	81.6 [06]	84.2 [01]	77.6 [26]	79.0 [16]	82.7 [04]
			0.35	64.6 [01]	55.2 [13]	56.5 [12]	61.7 [04]	63.6 [02]	51.5 [16]	54.3 [11]	59.0 [05]
		25	0.05	94.7 [00]	94.2 [02]	94.7 [00]	91.5 [14]	94.6 [01]	93.7 [07]	94.4 [01]	93.0 [04]
			0.15	84.5 [00]	83.6 [02]	84.2 [01]	82.5 [04]	84.2 [01]	81.8 [06]	83.8 [02]	82.6 [02]
			0.35	64.5 [01]	63.0 [04]	63.9 [01]	63.0 [02]	63.7 [02]	59.2 [08]	62.0 [03]	61.9 [03]
	Unequal	5	0.05	94.7 [00]	91.3 [19]	94.5 [01]	90.0 [07]	94.6 [01]	91.9 [07]	93.9 [01]	93.6 [02]
			0.15	84.6 [00]	79.3 [09]	83.6 [03]	80.5 [04]	84.4 [01]	78.4 [16]	82.3 [06]	82.7 [03]
			0.35	64.6 [01]	59.4 [08]	62.0 [04]	62.3 [02]	63.7 [02]	56.6 [11]	59.6 [08]	60.7 [04]
		25	0.05	94.8 [00]	94.3 [01]	94.7 [00]	91.5 [07]	94.6 [01]	94.1 [01]	94.5 [01]	93.7 [02]
			0.15	84.6 [01]	84.2 [02]	84.5 [01]	82.3 [04]	84.3 [01]	83.7 [02]	84.1 [01]	83.6 [01]
			0.35	64.6 [01]	64.1 [02]	64.2 [01]	63.7 [02]	63.6 [02]	61.4 [04]	62.5 [03]	63.2 [03]
10	Equal	5	0.05	93.9 [01]	84.1 [27]	90.7 [27]	92.0 [03]	93.9 [01]	77.1 [41]	91.6 [25]	91.6 [04]
			0.15	83.2 [01]	56.5 [20]	75.5 [23]	81.0 [03]	82.9 [02]	48.9 [19]	75.8 [23]	77.3 [08]
			0.35	62.1 [03]	47.0 [06]	48.5 [14]	57.5 [04]	61.2 [03]	31.5 [10]	44.6 [14]	49.8 [10]
		25	0.05	94.0 [01]	91.9 [03]	93.6 [01]	92.7 [02]	93.9 [01]	92.0 [04]	93.5 [01]	92.7 [02]
			0.15	83.1 [01]	79.9 [04]	81.9 [03]	82.1 [03]	82.7 [02]	78.0 [08]	81.6 [03]	80.7 [04]
			0.35	61.8 [03]	55.4 [08]	58.9 [05]	59.3 [05]	60.8 [03]	55.0 [06]	57.3 [05]	58.0 [07]
	Unequal	5	0.05	94.0 [01]	83.9 [21]	92.2 [03]	93.1 [03]	94.1 [01]	69.1 [27]	92.4 [04]	92.9 [02]
			0.15	83.5 [01]	65.7 [18]	79.2 [09]	81.9 [03]	83.1 [01]	56.4 [17]	78.2 [16]	79.8 [05]
			0.35	62.6 [02]	46.1 [09]	52.3 [12]	60.0 [04]	62.0 [02]	40.7 [13]	50.3 [13]	52.8 [07]
		25	0.05	94.1 [01]	92.7 [03]	93.9 [01]	93.1 [02]	94.0 [01]	92.4 [03]	93.8 [01]	93.2 [01]
			0.15	83.4 [02]	80.1 [04]	82.6 [02]	82.8 [02]	83.3 [01]	80.1 [04]	82.4 [03]	82.1 [03]
			0.35	62.1 [02]	56.3 [06]	59.6 [05]	60.5 [05]	61.3 [03]	55.4 [06]	58.6 [05]	58.5 [05]

Figure C.1: Simulation results of methods for partially-observed outcomes, linear SVMs, $N = 40$ per class

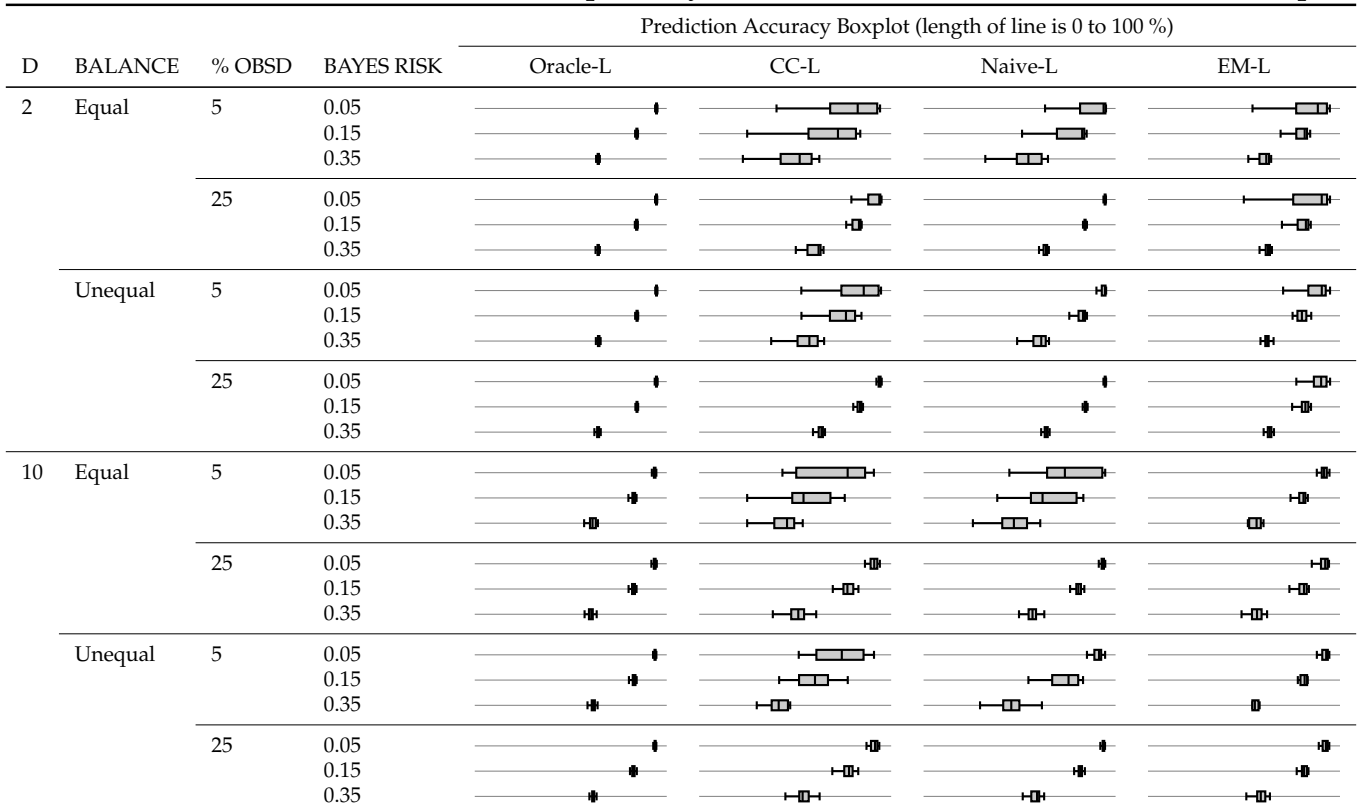


Figure C.2: Simulation results of methods for partially-observed outcomes, linear SVMs, $N = 100$ per class

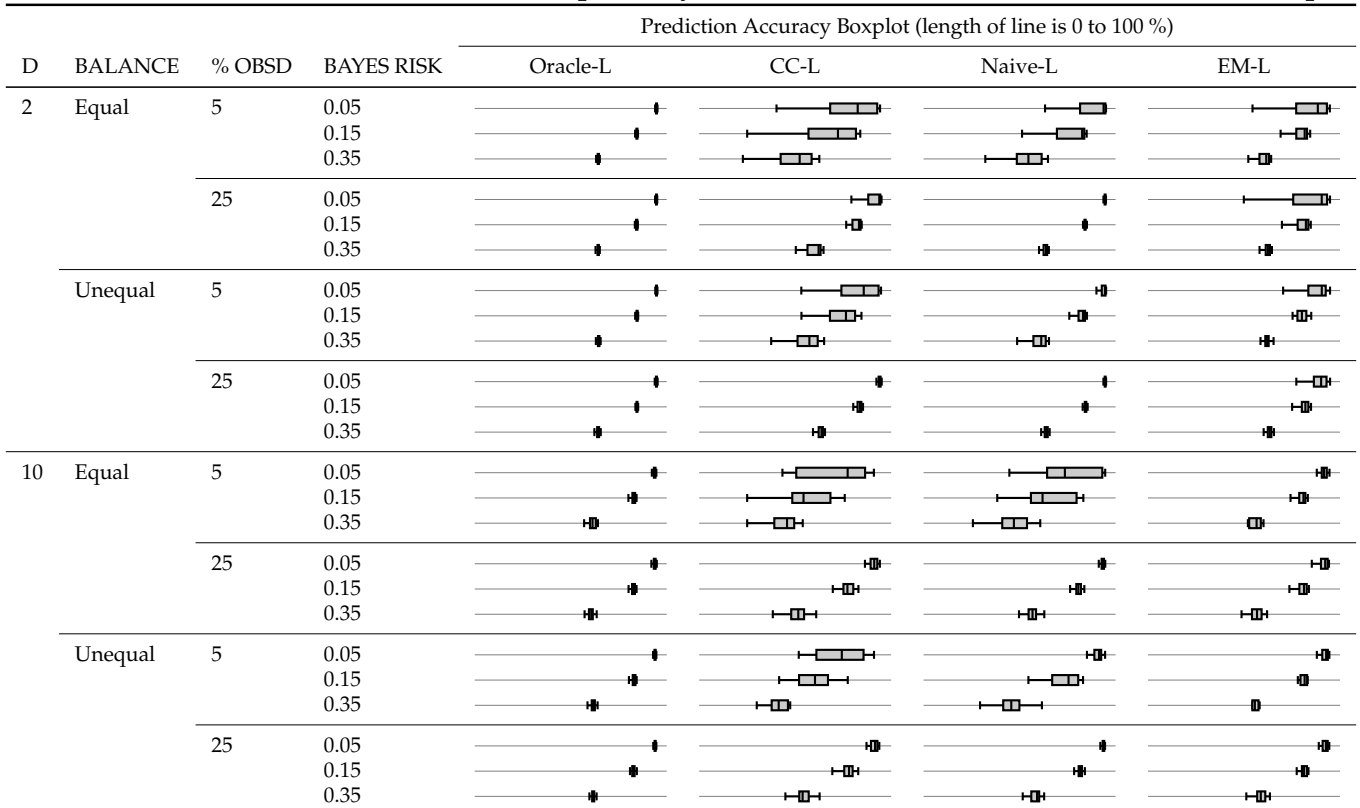


Figure C.3: Simulation results of methods for partially-observed outcomes, nonlinear SVMs, $N = 40$ per class

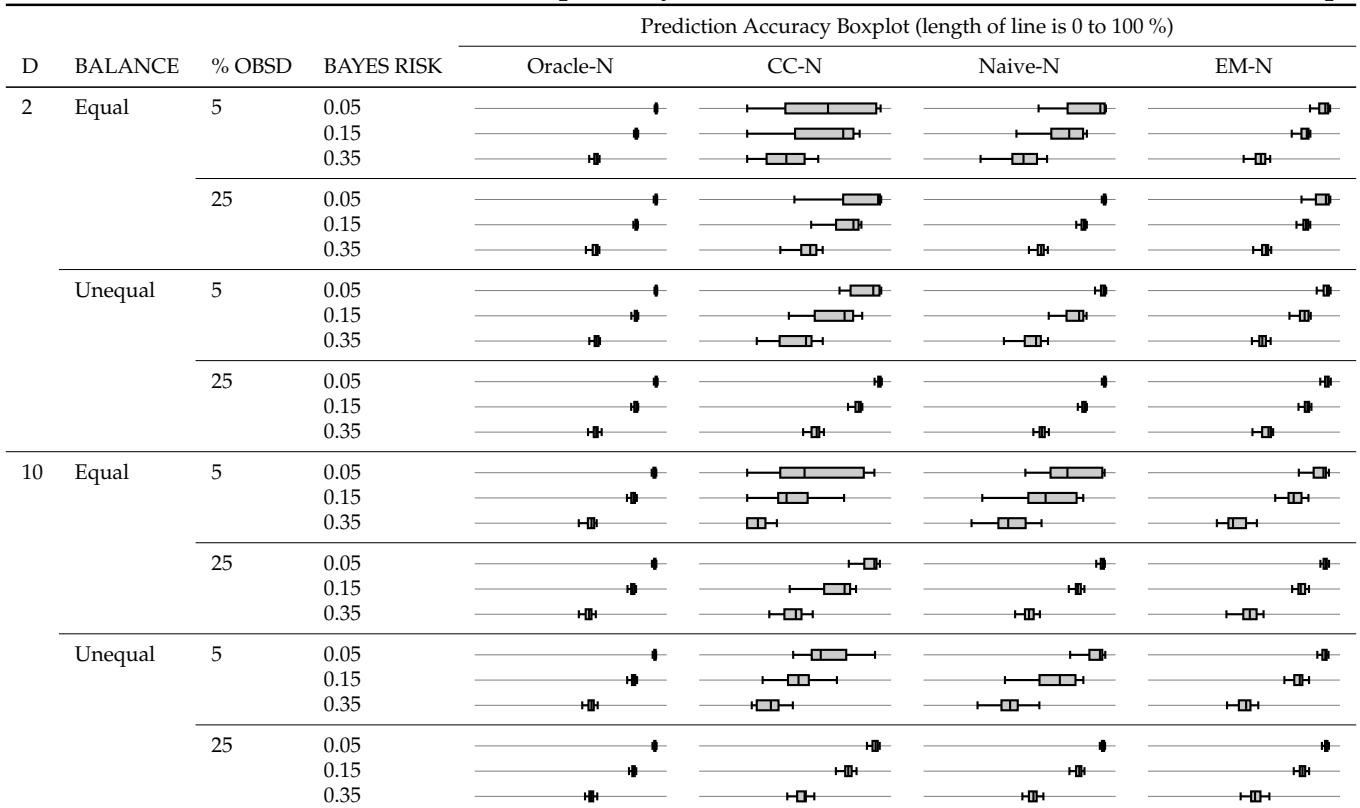


Figure C.4: Simulation results of methods for partially-observed outcomes, nonlinear SVMs, $N = 100$ per class

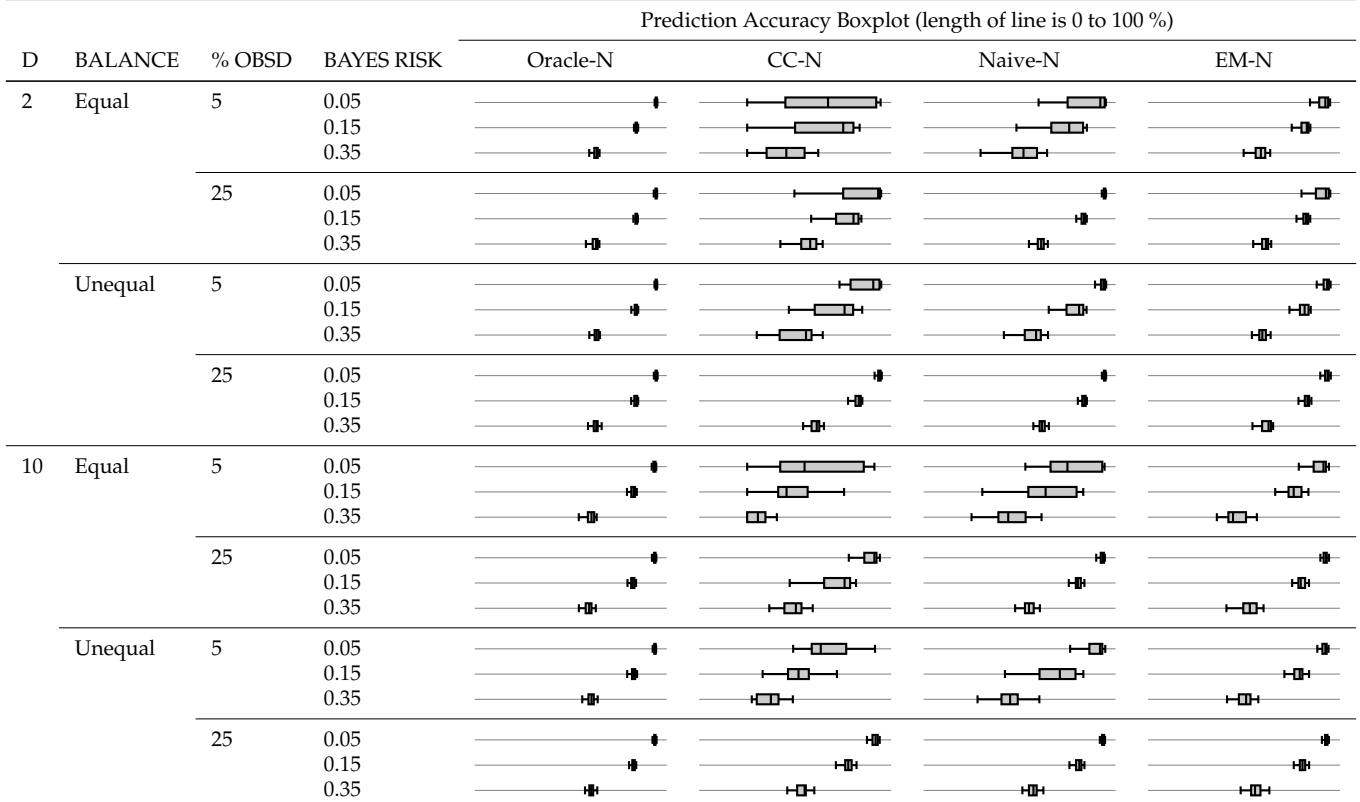


Figure C.5: Simulation results of methods for partially-observed outcomes, side-by-side comparison, linear SVMs, $N = 40$ per class

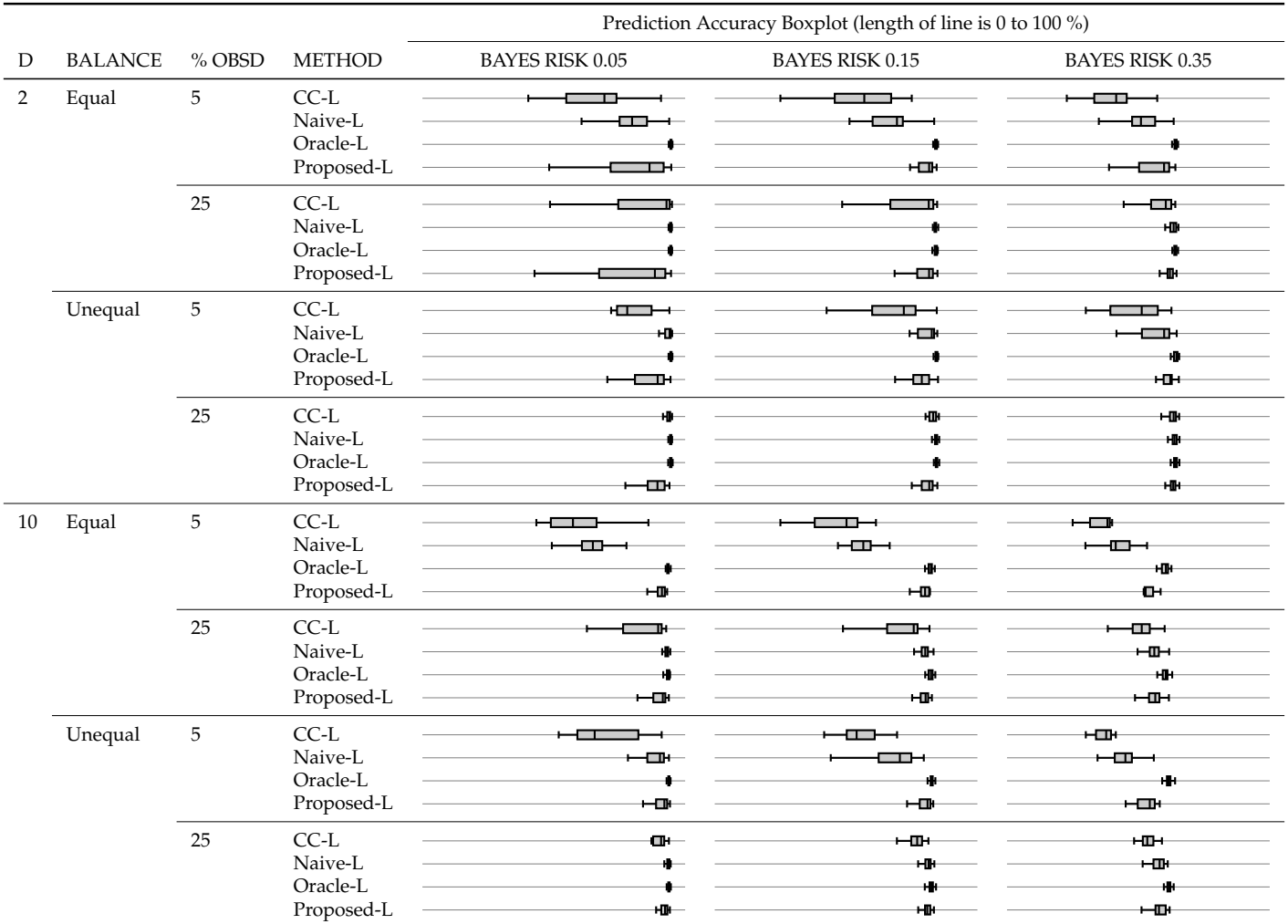


Figure C.6: Simulation results of methods for partially-observed outcomes: side-by-side comparison, nonlinear SVMs, $N = 40$ per class

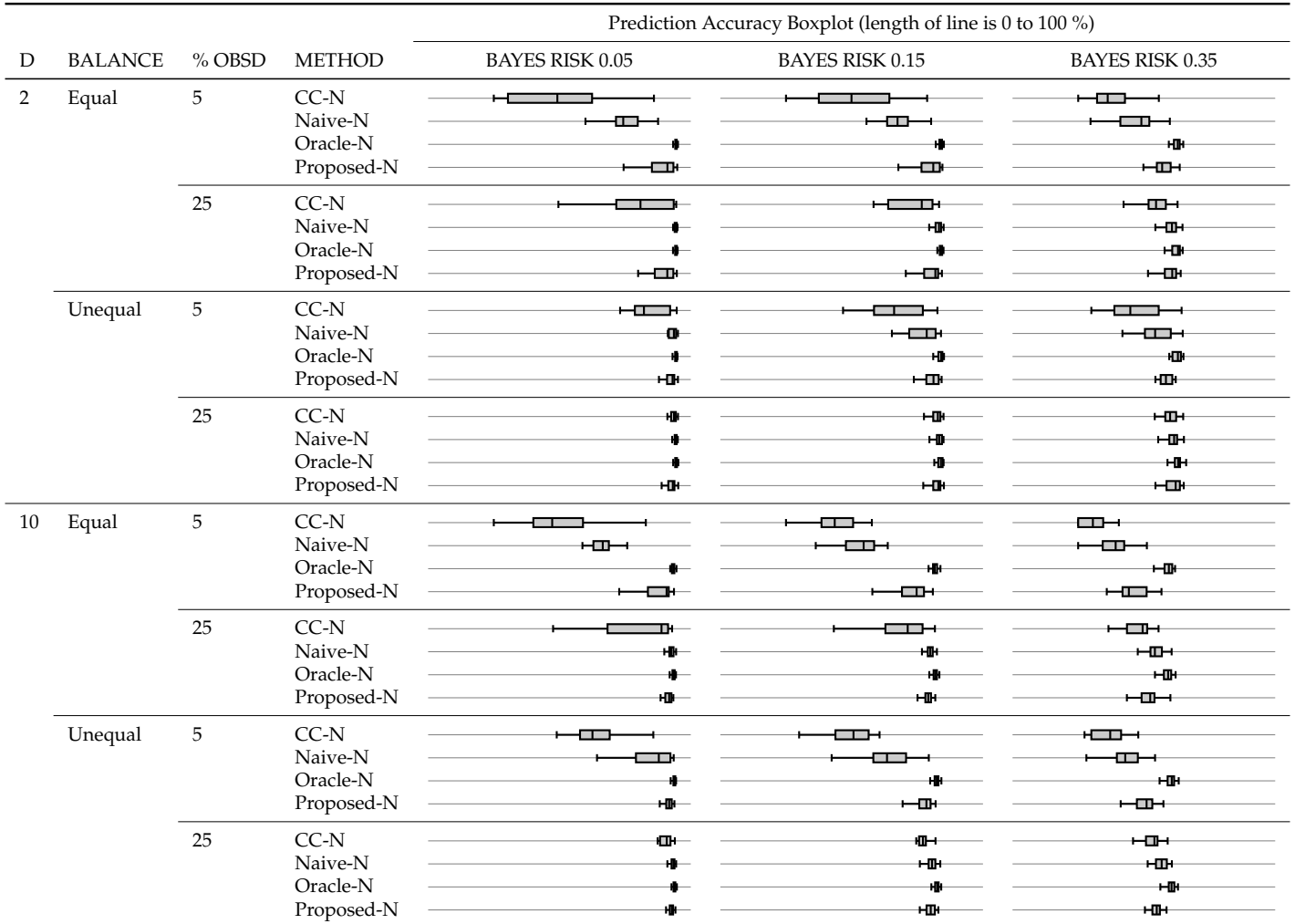


Figure C.7: Simulation results of methods for partially-observed outcomes, side-by-side comparison, linear SVMs, $N = 100$ per class

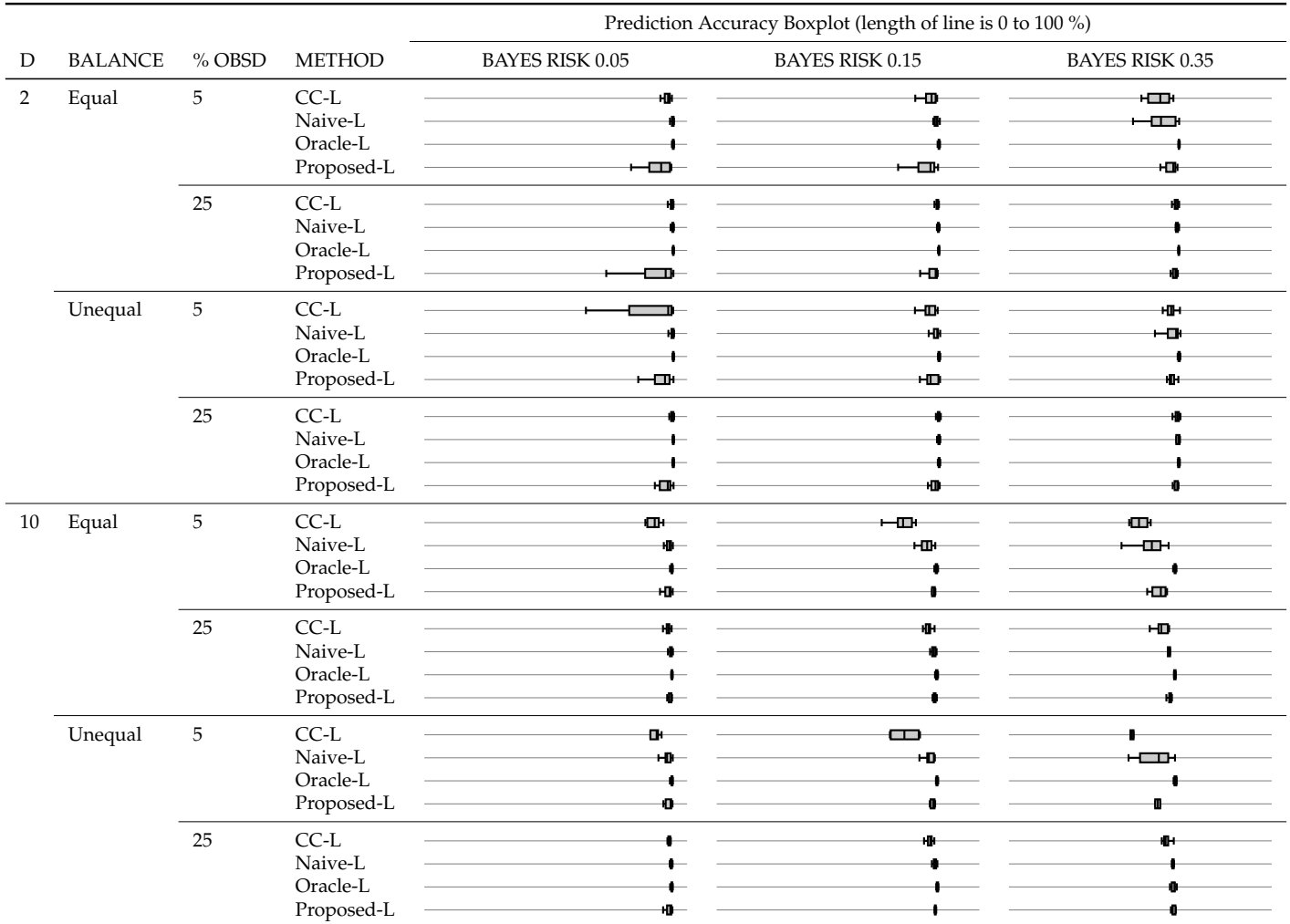
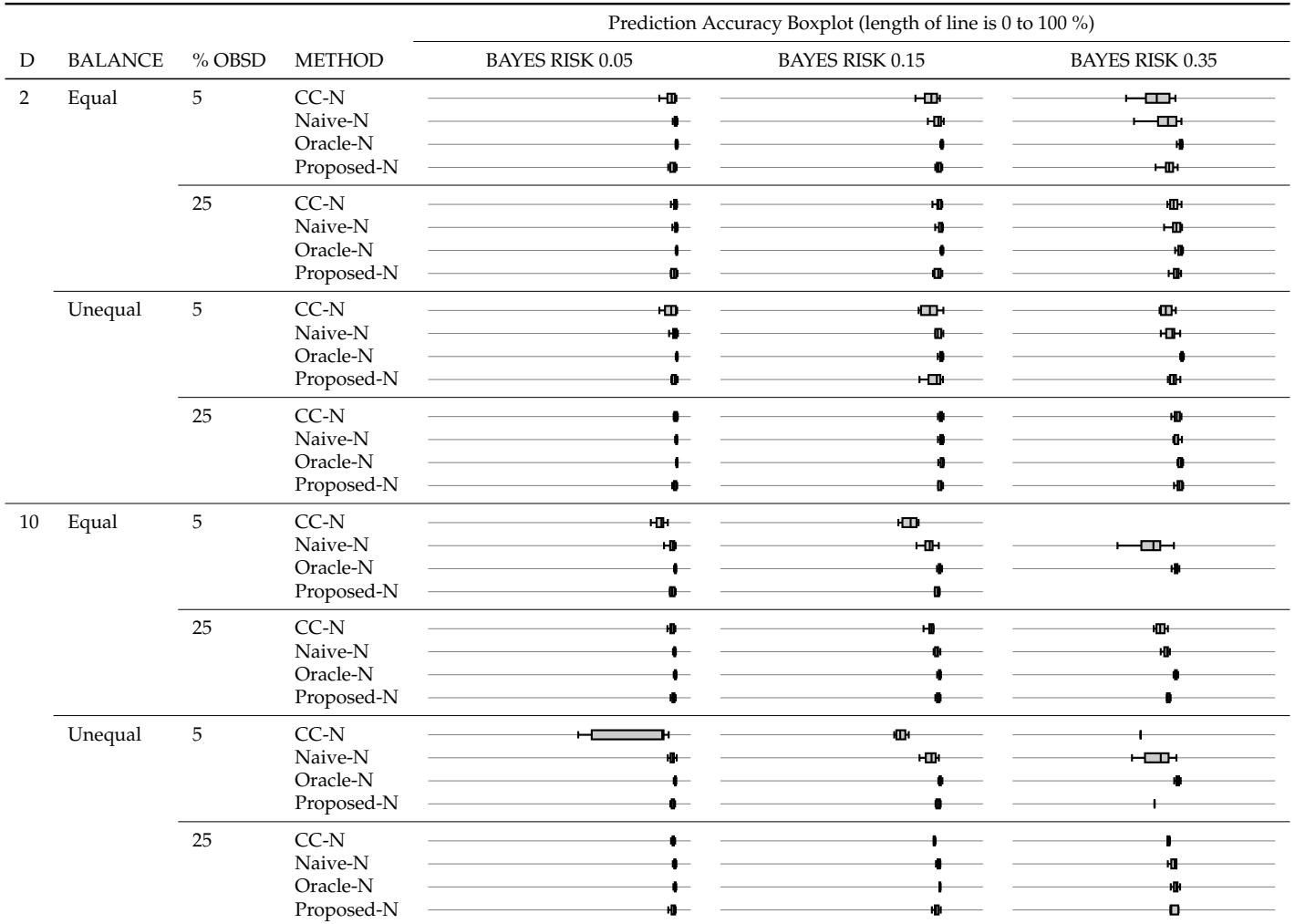


Figure C.8: Simulation results of methods for partially-observed outcomes: side-by-side comparison, nonlinear SVMs, $N = 100$ per class



C.6 Simulation Study Results for Semi-supervised Learning

The following tables report simulation study results for the semi-supervised settings. The competing methods are labeled with a -L or -N to denote linear kernel or nonlinear Gaussian kernel, respectively. The methods are:

Label	Method
SD	SD-SVM
CC	Complete Case SVM
S^3VM	S^3VM as implemented in [60]. Commonly called SVM^{light}.
EM	The method proposed in this paper

Table C.3: Simulation results of semi-supervised methods by Missing Data Model

				Prediction Accuracy [IQR]							
K	N	D	MCAR / MAR	EM-L	EM-N	SD-L	SD-N	S3VM-L	S3VM-N	CC-L	CC-N
2	40	2	MAR	82.6 [30]	77.4 [29]			66.5 [33]	60.0 [33]	80.3 [30]	75.7 [31]
			MCAR	84.3 [28]	81.3 [29]			81.4 [34]	63.2 [38]	83.4 [29]	80.6 [32]
		10	MAR	70.5 [29]	69.0 [27]			60.0 [29]	60.0 [16]	69.2 [26]	65.8 [25]
			MCAR	73.8 [30]	71.9 [27]			60.0 [31]	60.0 [16]	65.1 [25]	60.2 [23]
	100	2	MAR	78.9 [27]	65.1 [27]			66.8 [25]	62.4 [25]	77.4 [24]	70.3 [27]
			MCAR	84.5 [29]	84.2 [28]			84.0 [30]	64.5 [34]	84.7 [28]	84.2 [29]
		10	MAR	73.0 [29]	73.0 [30]			60.0 [32]	59.4 [24]	73.7 [26]	73.6 [27]
			MCAR	80.1 [31]	79.1 [33]			60.0 [34]	60.0 [30]	79.6 [28]	78.2 [29]
4	40	2	MAR	58.6 [23]	57.6 [26]					62.4 [28]	58.7 [29]
			MCAR	61.9 [20]	67.2 [25]					69.1 [25]	63.8 [31]
		10	MAR	69.1 [30]	62.8 [30]					61.7 [27]	57.6 [28]
			MCAR	68.8 [28]	62.1 [29]					60.1 [29]	55.8 [32]
	100	2	MAR	62.2 [21]	58.8 [27]					72.1 [27]	69.4 [28]
			MCAR	69.0 [20]	76.8 [23]					83.8 [29]	83.2 [30]
		10	MAR	77.0 [31]	71.9 [30]					75.5 [31]	75.2 [32]
			MCAR	80.7 [28]	79.1 [29]					80.8 [28]	80.6 [31]

Table C.4: Simulation results of semi-supervised methods by Class Balance

				Prediction Accuracy [IQR]							
K	N	D	BALANCE	EM-L	EM-N	SD-L	SD-N	S3VM-L	S3VM-N	CC-L	CC-N
2	40	2	Equal class sizes	84.2 [29]	78.2 [33]			68.7 [44]	63.3 [43]	80.9 [32]	63.5 [36]
			Unequal class sizes	83.9 [28]	79.8 [28]			81.3 [34]	60.0 [32]	83.0 [27]	81.4 [28]
		10	Equal class sizes	71.6 [29]	70.4 [29]			52.3 [30]	50.0 [17]	65.8 [30]	58.3 [28]
			Unequal class sizes	72.8 [30]	71.3 [25]			60.0 [21]	60.0 [02]	69.3 [22]	68.2 [21]
	100	2	Equal class sizes	84.7 [26]	72.1 [27]			84.0 [31]	82.7 [43]	81.8 [25]	75.1 [29]
			Unequal class sizes	83.2 [30]	78.1 [26]			76.9 [26]	63.2 [25]	82.6 [23]	81.7 [25]
		10	Equal class sizes	75.9 [29]	75.2 [32]			52.7 [32]	50.0 [26]	75.9 [27]	75.5 [28]
			Unequal class sizes	78.7 [33]	78.0 [32]			60.0 [24]	60.0 [19]	77.7 [28]	77.4 [29]
4	40	2	Equal class sizes	59.7 [23]	61.4 [27]					61.7 [27]	55.0 [30]
			Unequal class sizes	61.3 [20]	62.6 [26]					72.7 [26]	71.0 [26]
		10	Equal class sizes	67.7 [29]	59.2 [30]					55.2 [27]	52.2 [31]
			Unequal class sizes	70.6 [29]	64.3 [29]					67.0 [26]	64.2 [28]
	100	2	Equal class sizes	64.3 [22]	64.4 [28]					78.0 [26]	76.4 [27]
			Unequal class sizes	66.3 [20]	69.7 [27]					80.0 [27]	77.7 [28]
		10	Equal class sizes	79.4 [30]	72.6 [28]					78.7 [29]	77.7 [30]
			Unequal class sizes	78.0 [27]	76.1 [27]					77.1 [27]	76.5 [29]

Table C.5: Simulation results of semi-supervised methods by Different Quantities of Missing Data

				Prediction Accuracy [IQR]							
K	N	D	% OBSD	EM-L	EM-N	SD-L	SD-N	S3VM-L	S3VM-N	CC-L	CC-N
2	40	2	5	81.7 [28]	75.8 [28]			73.7 [33]	60.0 [33]	76.6 [27]	63.7 [29]
			25	84.4 [29]	82.4 [30]			80.8 [34]	64.5 [33]	84.3 [28]	82.9 [29]
		10	5	66.2 [27]	62.4 [23]			60.0 [27]	60.0 [10]	60.0 [21]	60.0 [19]
			25	77.0 [31]	75.5 [31]			60.0 [29]	60.0 [25]	76.1 [30]	75.2 [30]
	100	2	5	83.2 [33]	67.6 [34]			64.8 [34]	60.0 [25]	78.1 [30]	75.9 [27]
			25	83.9 [26]	80.0 [22]			83.1 [23]	80.7 [25]	83.2 [21]	83.2 [22]
		10	5	70.1 [28]	70.9 [30]			60.0 [32]	58.4 [22]	71.1 [25]	70.2 [25]
			25	80.6 [32]	79.6 [33]			60.0 [31]	60.0 [29]	79.8 [30]	79.1 [31]
4	40	2	5	55.8 [22]	55.6 [25]					57.4 [29]	49.3 [30]
			25	63.2 [20]	67.1 [25]					72.2 [24]	70.1 [26]
		10	5	59.8 [30]	51.3 [27]					49.9 [25]	44.0 [24]
			25	74.2 [32]	70.5 [32]					71.9 [30]	70.2 [27]
	100	2	5	58.8 [22]	59.6 [27]					74.2 [27]	71.6 [28]
			25	69.7 [20]	71.5 [24]					82.5 [26]	80.0 [27]
		10	5	69.6 [33]	63.9 [30]					70.6 [34]	69.8 [33]
			25	81.2 [21]	80.0 [25]					80.7 [20]	80.5 [21]

Table C.6: Simulation results of semi-supervised methods by Underlying Risk

				Prediction Accuracy [IQR]							
K	N	D	BAYES RISK	EM-L	EM-N	SD-L	SD-N	S3VM-L	S3VM-N	CC-L	CC-N
2	40	2	0.05	93.0 [04]	90.8 [11]			93.9 [35]	92.9 [34]	91.9 [06]	90.3 [14]
			0.15	82.1 [09]	77.5 [14]			81.4 [25]	65.2 [23]	80.4 [11]	75.7 [22]
			0.35	61.0 [12]	56.6 [11]			59.9 [12]	58.5 [10]	60.0 [10]	57.4 [10]
		10	0.05	91.0 [07]	90.5 [09]			89.0 [33]	60.5 [28]	87.4 [12]	86.5 [19]
			0.15	73.9 [13]	73.2 [13]			66.1 [19]	60.0 [18]	72.6 [16]	71.8 [17]
			0.35	54.6 [06]	55.5 [06]			54.9 [10]	55.8 [10]	56.0 [08]	56.4 [08]
	100	2	0.05	93.6 [02]	92.4 [11]			94.3 [04]	94.1 [35]	92.8 [07]	92.1 [09]
			0.15	83.6 [03]	83.3 [10]			83.6 [08]	83.2 [25]	83.0 [07]	83.5 [10]
			0.35	61.9 [10]	57.9 [11]			63.1 [05]	60.0 [08]	61.2 [09]	59.5 [10]
		10	0.05	92.9 [05]	93.1 [04]			91.9 [44]	82.0 [42]	90.7 [06]	90.0 [06]
			0.15	78.3 [10]	77.5 [09]			74.5 [32]	60.0 [28]	77.3 [10]	76.3 [09]
			0.35	56.3 [06]	55.9 [07]			56.3 [10]	55.4 [10]	56.7 [06]	56.5 [06]
4	40	2	0.05	72.5 [24]	78.7 [24]					84.3 [21]	79.7 [26]
			0.15	67.9 [17]	68.0 [18]					72.0 [21]	68.2 [25]
			0.35	54.0 [12]	49.3 [13]					53.9 [16]	50.2 [17]
		10	0.05	86.0 [13]	83.6 [19]					81.9 [22]	74.8 [26]
			0.15	71.5 [14]	67.2 [17]					67.7 [20]	66.2 [22]
			0.35	47.2 [12]	44.6 [12]					47.3 [13]	45.5 [15]
	100	2	0.05	76.3 [19]	82.1 [19]					92.4 [07]	91.5 [08]
			0.15	70.4 [15]	71.2 [20]					80.1 [10]	78.4 [12]
			0.35	56.1 [09]	51.1 [15]					58.0 [11]	55.7 [12]
		10	0.05	90.3 [06]	86.8 [10]					89.9 [06]	89.8 [06]
			0.15	77.5 [09]	73.0 [14]					75.4 [09]	74.9 [09]
			0.35	51.7 [11]	48.1 [13]					49.8 [10]	50.0 [10]

REFERENCES

[1] AALSD (2014), "Recommendations for Testing, Managing, and Treating Hepatitis C," Tech. rep., American Association for the Study of Liver Diseases.

[2] Acuña, E. and Rodriguez, C. (2004), "The treatment of missing values and its effect in the classifier accuracy," in *Classification, Clustering, and Data Mining Applications*, (eds.) D. Banks, F. R. McMorris, P. Arabie, and W. Gaul, Springer Berlin Heidelberg, pp. 639–647.

[3] Anderson, H. S. and Gupta, M. R. (2011), "Expected kernel for missing features in support vector machines," in *Statistical Signal Processing Workshop (SSP), 2011 IEEE*, IEEE, pp. 285–288.

[4] Andridge, R. R. and Little, R. J. A. (2010), "A Review of Hot Deck Imputation for Survey Non-response," *International Statistical Review*, 78(1), 40–64.

[5] Armstrong, G. L., Wasley, A., Simard, E. P., Mcquillan, G. M., Kuhnert, W. L., and Alter, M. J. (2006), "The Prevalence of Hepatitis C Virus Infection in the United States, 1999 through 2002," *Annals of Internal Medicine*, 144(10), 705–716.

[6] Arora, A. and Namboodiri, A. M. (2011), "A Semi-supervised SVM Framework for Character Recognition," in *2011 International Conference on Document Analysis and Recognition*, IEEE, pp. 1105–1109.

[7] Austin, P. C. (2011), "An Introduction to Propensity Score Methods for Reducing the Effects of Confounding in Observational Studies." *Multivariate Behavioral Research*, 46(3), 399–424.

[8] Bang, H. and Robins, J. M. (2005), "Doubly robust estimation in missing data and causal inference models." *Biometrics*, 61(4), 962–73.

[9] Batista, G. E. A. P. A. and Monard, M. C. (2003), "An analysis of four missing data treatment methods for supervised learning," *Applied Artificial Intelligence*, 17, 519–533.

[10] Beale, E. M. L. and Little, R. J. A. (1975), "Missing Values in Multivariate Analysis," *Journal of the Royal Statistical Society: Series B (Methodological)*, 37(1), 129–145.

[11] Bennett, K. P. and Demiriz, A. (1999), "Semi-Supervised Support Vector Machines," *Advances in Neural Information Processing Systems*, 368–374.

[12] Bennett, K. P., Demiriz, A., and Maclin, R. (2002), "Exploiting unlabeled data in ensemble methods," in *Proceedings of the Eighth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining - KDD '02*, New York, NY: ACM Press, pp. 289–297.

[13] Berlinet, A. F. and Roland, C. (2012), "Acceleration of the EM algorithm: P-EM versus epsilon algorithm," *Computational Statistics & Data Analysis*, 56(12), 4122–4137.

[14] Blum, A. and Mitchell, T. (1998), "Combining labeled and unlabeled data with co-training," in *Proceedings of the Eleventh Annual Conference on Computational Learning Theory - COLT' 98*, New York, NY: ACM Press, pp. 92–100.

[15] Boser, B. E., Guyon, I. M., and Vapnik, V. N. (1992), "A Training Algorithm for Optimal Margin Classifiers," in *Proceedings of the Fifth Annual Workshop on Computational Learning Theory*, Pittsburgh, PA: ACM, pp. 144–152.

[16] Breiman, L. (2001), "Random Forests," *Machine Learning*, 45(1), 5–32.

[17] Breiman, L., Friendman, J. H., Olshen, R. A., and Stone, C. J. (1993), *Classification and Regression Trees*, New York, NY: Chapman & Hall.

[18] Carpenter, J. R., Kenward, M. G., and Vansteelandt, S. (2006), "A comparison of multiple imputation and doubly robust estimation for analyses with missing data," *Journal of the Royal Statistical Society: Series A (Statistics in Society)*, 169(3), 571–584.

[19] Chan, L. S. and Dunn, O. J. (1974), "A Note on the Asymptotic Aspect of the Treatment of Missing Values in Discriminant Analysis," *Journal of the American Statistical Association*, 69(347), 672–673.

[20] Chang, C.-C. and Lin, C.-J. (2011), "LIBSVM: A library for support vector machines," *ACM Transactions on Intelligent Systems and Technology*, 2(3), 27:1–27:27.

[21] Chapelle, O., Chi, M., and Zien, A. (2006), "A continuation method for semi-supervised SVMs," in *Proceedings of the 23rd International Conference on Machine Learning - ICML '06*, New York, NY: ACM Press, pp. 185–192.

[22] Chechik, G., Heitz, G., Elidan, G., Abbeel, P., and Daphne Koller (2007), "Max-margin classification of incomplete data," in *Advances in Neural Information Processing Systems 19*, Cambridge, MA: MIT Press, pp. 233–240.

[23] Chen, D., Bourlard, H., and Thiran, J.-P. (2001), "Text Identification in Complex Background Using SVM," in *Proceedings of the 2001 IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pp. II–621–II–626 vol.2.

[24] Chou, R., Bottrell Barth, E. K., Wasson, N., Rahman, B., and Guise, J.-M. (2012), "Screening for Hepatitis C Infection in Adults. Comparative Effectiveness Review No. 69. AHRQ Publication No. 12-EHC090-EF." Tech. Rep. 69, Agency for Healthcare Research and Quality, U.S. Department of Health and Human Services, Rockville, MD.

[25] Cortes, C. and Vapnik, V. (1995), "Support-Vector Networks," *Machine Learning*, 20(3), 273–297.

[26] Crammer, K. and Singer, Y. (2001), "On the Algorithmic Implementation of Multiclass Kernel-based Vector Machines," *Journal of Machine Learning Research*, 2, 265–292.

[27] De Ruyck, K., Sabbe, N., Oberije, C., Vandecasteele, K., Thas, O., De Ruysscher, D., Lambin, P., Van Meerbeeck, J., De Neve, W., and Thierens, H. (2011), "Development of a multicomponent prediction model for acute esophagitis in lung cancer patients receiving chemoradiotherapy." *International Journal of Radiation Oncology, Biology, Physics*, 81(2), 537–44.

[28] Dempster, A. P., Laird, N. M., and Rubin, D. B. (1977), "Maximum Likelihood from Incomplete Data via the EM Algorithm," *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 39(1), 1–38.

[29] Devroye, L., Gyorfi, L., and Lugosi, G. (1996), *A probabilistic theory of pattern recognition*, New York, NY: Springer.

[30] Dick, U., Haider, P., and Scheffer, T. (2008), "Learning from incomplete data with infinite imputations," in *Proceedings of the 25th International Conference on Machine Learning - ICML '08*, New York, NY: ACM Press, pp. 232–239.

[31] Ding, Y. and Simonoff, J. S. (2010), "An Investigation of Missing Data Methods for Classification Trees Applied to Binary Response Data," *Journal of Michine Learning Research*, 11, 131–170.

[32] Duan, K.-B. and Keerthi, S. S. (2005), "Which Is the Best Multiclass SVM Method? An Empirical Study," in *Multiple Classifier Systems, Lecture Notes in Computer Science*, (eds.) N. C. Oza, R. Polikar, J. Kittler, and F. Roli, Springer Berlin Heidelberg, pp. 3541:278–285.

[33] Efron, B., Hastie, T., Johnstone, I., and Tibshirani, R. (2004), "Least Angle Regression," *Annals of Statistics*, 32(2), 407–499.

[34] Evgeniou, T., Pontil, M., and Poggio, T. (2000), "Regularization Networks and Support Vector Machines," *Advances in Computation Mathematics*, 13(1), 1–50.

[35] Farhangfar, A., Kurgan, L., and Dy, J. (2008), "Impact of imputation of missing values on classification error for discrete data," *Pattern Recognition*, 41(12), 3692–3705.

[36] French, J. L. and Wand, M. P. (2004), "Generalized additive models for cancer mapping with incomplete covariates." *Biostatistics*, 5(2), 177–191.

[37] Friedman, J., Hastie, T., and Tibshirani, R. (2010), "Regularization Paths for Generalized Linear Models via Coordinate Descent," *Journal of Statistical Software*, 33(1), 1–22.

[38] Friedman, J. H. (1996), "Another Approach to Polychotomous Classification," Tech. rep., Stanford University.

[39] Fuchs, C. (1982), "Maximum Likelihood Estimation and Model Selection in Contingency Tables with Missing Data," *Journal of the American Statistical Association*, 77(378), 270–278.

[40] Furey, T. S., Cristianini, N., Duffy, N., Bednarski, D. W., Schummer, M., and Haussler, D. (2000), "Support vector machine classification and validation of cancer tissue samples using microarray expression data," *Bioinformatics*, 16(10), 906–914.

[41] Garcia, A. J. T. and Hruschka, E. R. (2005), "Naïve Bayes as an Imputation Tool for Classification Problems," in *Proceedings of the Fifth International Conference on Hybrid Intelligent Systems (HIS'05)*, pp. 3–5.

[42] Ghahramani, Z. and Jordan, M. I. (1994), "Supervised learning from incomplete data via an EM approach," in *Advances in Neural Information Processing Systems 6*, (eds.) J. Cowan, G. Tesauro, and J. Alspector, Morgan-Kaufmann, pp. 120–127.

[43] Godambe, V. P. (1960), "An Optimum Property of Regular Maximum Likelihood Estimation," *The Annals of Mathematical Statistics*, 31(4), 1208–1211.

[44] Godambe, V. P. and Thompson, M. E. (1978), "Some aspects of the theory of estimating equations," *Journal of Statistical Planning and Inference*, 2, 95–104.

[45] Gordon, S. C., Muir, A. J., Lim, J. K., Pearlman, B., Argo, C. K., Ramani, A., Maliakkal, B., Alam, I., Stewart, T. G., Vainorius, M., Peter, J., Nelson, D. R., Fried, M. W., and Reddy, K. R. (2015), "Safety profile of boceprevir and telaprevir in chronic hepatitis C: Real world experience from HCV-TARGET," *Journal of Hepatology*, 62(2), 286–293.

[46] Harel, O. and Zhou, X.-H. (2007), "Multiple imputation: Review of theory, implementation and software," *Statistics in Medicine*, 26, 3057–3077.

[47] Hartley, H. O. and Hocking, R. R. (1971), "The Analysis of Incomplete Data," *Biometrics*, 27(4), 783–823.

[48] Hastie, T. (2015), "gam: Generalized Additive Models. R package version 1.12." .

[49] Hastie, T. and Tibshirani, R. (1998), "Classification by Pairwise Coupling," *The Annals of Statistics*, 26(2), 451–471.

[50] Hastie, T., Tibshirani, R., and Friedman, J. (2013), *The Elements of Statistical Learning*, Springer, 2^{nd} ed.

[51] Hastie, T., Tibshirani, R., Sherlock, G., Eisen, M., Brown, P., and Botstein, D. (1999), "Imputing Missing Data for Gene Expression Arrays Imputation," Tech. rep., Division of Biostatistics, Standford University.

[52] Herring, A. H. and Ibrahim, J. G. (2001), "Likelihood-Based Methods for Missing Covariates in the Cox Proportional Hazards Model," *Journal of the American Statistical Association*, 96(453), 292–302.

[53] ——— (2002), "Maximum likelihood estimation in random effects cure rate models with nonignorable missing covariates." *Biostatistics*, 3(3), 387–405.

[54] Hsu, C.-W. and Lin, C.-J. (2002), "A comparison of methods for multiclass support vector machines." *IEEE transactions on Neural Networks*, 13(2), 415–425.

[55] Ibrahim, J. G. (1990), "Incomplete Data in Generalized Linear Models," *Journal of the American Statistical Association*, 85(411), 765–769.

[56] Ibrahim, J. G., Chen, M.-h., and Lipsitz, S. R. (2002), "Bayesian Methods for Generalized Linear Models with Covariates Missing at Random," *Canadian Journal of Statistics*, 30(1), 55–78.

[57] Ibrahim, J. G., Chen, M.-H., Lipsitz, S. R., and Herring, A. H. (2005), "Missing-Data Methods for Generalized Linear Models," *Journal of the American Statistical Association*, 100(469), 332–346.

[58] Imai, K. and Ratkovic, M. (2014), "Covariate balancing propensity score," *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 76(1), 243–263.

[59] Jensen, D. M. (2002), "Endoscopic screening for varices in cirrhosis: Findings, implications, and outcomes," *Gastroenterology*, 122(6), 1620–1630.

[60] Joachims, T. (1999), "Making Large-Scale SVM Learning Practical," in *Advances in Kernel Methods - Support Vector Learning*, (eds.) B. Schölkopf, C. J. C. Burges, and A. J. Smola, MIT Press, chap. 11, p. 1999.

[61] Kang, J. D. Y. and Schafer, J. L. (2007), "Demystifying Double Robustness: A Comparison of Alternative Strategies for Estimating a Population Mean from Incomplete Data," *Statistical Science*, 22(4), 523–539.

[62] Lee, Y., Lin, Y., and Wahba, G. (2004), "Multicategory Support Vector Machines," *Journal of the American Statistical Association*, 99(465), 67–81.

[63] Li, Y.-f. and Zhou, Z.-h. (2015), "Towards Making Unlabeled Data Never Hurt," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 37(1), 175–188.

[64] Lipsitz, S. R. and Ibrahim, J. G. (1996), "Using the EM-algorithm for survival data with incomplete categorical covariates." *Lifetime Data Analysis*, 2(1), 5–14.

[65] Little, R. J. A. (1978), "Consistent Regression Methods for Discriminant Analysis with Incomplete Data," *Journal of the American Statistical Association*, 73(362), 319–322.

[66] Little, R. J. A. and Rubin, D. B. (2002), *Statistical Analysis with Missing Data*, Hoboken, New Jersey: John Wiley & Sons, 2^{nd} ed.

[67] Liu, C. and Rubin, D. B. (2014), "The ECME Algorithm: A Simple Extension of EM and ECM with Faster Monotone Convergence," *Biometrika*, 81(4), 633–648.

[68] Liu, Y. (2007), "Fisher Consistency of Multicategory Support Vector Machines," in *Proceedings of the Eleventh International Workshop on Artificial Intelligence and Statistics*, Madison, WI: Omnipress, pp. 289–296.

[69] Liu, Y. and Yuan, M. (2011), "Reinforced Multicategory Support Vector Machines," *Journal of Computational and Graphical Statistics*, 20(4), 901–919.

[70] Louis, T. A. (1982), "Finding the Observed Information Matrix when Using the EM Algorithm," *Journal of the Royal Statistical Society. Series B (Methodological)*, 44(2), 226–233.

[71] Ly, K. N., Xing, J., Klevens, R. M., Jiles, R. B., Ward, J. W., and Homberg, S. D. (2012), "The Increasing Burden of Mortality From Viral Hepatitis in the United States Between 1999 and 2007," *Annals of Internal Medicine*, 156(4), 271–280.

[72] Meng, X.-L. and van Dyk, D. (1997), "The EM Algorithm–An Old Folk-Song Sung to a Fast New Tune," *Journal of the Royal Statistical Society. Series B (Methodological)*, 59(3), 511–567.

[73] ——— (1998), "Fast EM-Type Implementations for Mixed Effects Models," *Journal of the Royal Statistical Society. Series B (Statistical Methodology)*, 60(3), 559–578.

[74] Negro, F. and Alberti, A. (2011), "The global health burden of hepatitis C virus infection," *Liver International*, 31 Suppl 2(July), 1–3.

[75] Nigam, K., McCallum, A. K., Thrun, S., and Mitchell, T. (2000), "Text Classification from Labeled and Unlabeled Documents using EM," *Machine Learning*, 39, 103–134.

[76] ——— (2000), "Text Classification from Labeled and Unlabeled Documents using EM," *Machine Learning*, 39(2), 103–134.

[77] Osuna, E., Freund, R., and Girosi, F. (1997), "Training Support Vector Machines: an Application to Face Detection," in *IEEE Computer Society Conference on Computer Vision and Pattern Recognition, 1997*, pp. 130–136.

[78] Park, T. and Casella, G. (2008), "The Bayesian Lasso," *Journal of the American Statistical Association*, 103(482), 681–686.

[79] Platt, J. C. (1998), "Fast Training of Support Vector Machines Using Sequential Minimal Optimization," in *Advances in Kernel Methods: Support Vector Learning*, (eds.) B. Schölkopf, C. J. C. Burges, and A. J. Smola, MIT Press, chap. 12, pp. 185–208.

[80] ——— (1999), "Probabilistic Outputs for Support Vector Machines and Comparisons to Regularized Likelihood Methods," in *Advances in Large Margin Classifiers*, MIT Press, pp. 61–74.

[81] Platt, J. C., Cristianini, N., and Shawe-Taylor, J. (2000), "Large Margin DAGs for Multiclass Classification," in *Advances in Neural Information Processing Systems 12*, (eds.) S. A. Solla, T. K. Leen, and K.-R. Müller, Cambridge, MA: MIT Press, pp. 547–553.

[82] Raghunathan, T. E., Lepkowski, J. M., Hoewyk, J. V., and Solenberger, P. (2001), "A Multivariate Technique for Multiply Imputing Missing Values Using a Sequence of Regression Models," *Survey Methodology*, 27(1), 85–95.

[83] Rehme, A. K., Volz, L. J., Feis, D.-L., Bomilcar-Focke, I., Liebig, T., Eickhoff, S. B., Fink, G. R., and Grefkes, C. (2014), "Identifying Neuroimaging Markers of Motor Disability in Acute Stroke by Machine Learning Techniques." *Cerebral Cortex*.

[84] Rifkin, R. and Klautau, A. (2004), "In Defense of One-Vs-All Classification," *Journal of Machine Learning Research*, 5, 101–141.

[85] Robins, J. (1994), "Estimation of regression coefficients when some regressors are not always observed," *Journal of the American Statistical Association*, 89(427), 846–866.

[86] Robins, J. M., Rotnitzky, A., and Zhao, L. P. (1994), "Estimation of Regression Coefficients When Some Regressors Are Not Always Observed," *Journal of the American Statistical Association*, 89(427), 846–866.

[87] Roli, F. (2005), "Semi-supervised Multiple Classifier Systems: Background and Research Directions," in *Multiple Classifier Systems, Lecture Notes in Computer Science*, (eds.) N. Oza, R. Polikar, J. Kittler, and F. Roli, Springer-Verlag Berlin Heidelberg, pp. 3541:1–11.

[88] Rotnitzky, A., Lei, Q., Sued, M., and Robins, J. M. (2012), "Improved double-robust estimation in missing data and causal inference models." *Biometrika*, 99(2), 439–456.

[89] Rubin, D. B. (1976), "Inference and Missing Data," *Biometrika*, 63(3), 581–592.

[90] ——— (1987), *Multiple Imputation for Nonresponse in Surveys*, New York, NY: John Wiley & Sons.

[91] ——— (1996), "Multiple Imputation After 18+ Years," *Journal of the American Statistical Association*, 91(434), 473–489.

[92] Sabbe, N., Thas, O., and Ottoy, J.-P. (2013), "EMLasso: logistic lasso with missing data." *Statistics in Medicine*, 32(18), 3143–3157.

[93] Schafer, J. L. (1999), "Multiple imputation: a primer." *Statistical Methods in Medical Research*, 8(1), 3–15.

[94] Schölkopf, B. and Smola, A. J. (2002), *Learning with Kernels*, Cambridge, MA: MIT Press.

[95] Shivaswamy, P. K., Bhattacharyya, C., and Smola, A. J. (2006), "Second Order Cone Programming Approaches for Handling Missing and Uncertain Data," *Journal of Machine Learning Research*, 7, 1283–1314.

[96] Sindhwani, V. and Keerthi, S. (2007), "Newton Methods for Fast Solution of Semi- supervised Linear SVMs," in *Large Scale Kernel Machines*, (eds.) L. Bottou, O. Chapelle, D. DeCoste, and J. Weston, MIT Press, chap. 7, pp. 155–174.

[97] Smola, A., Vishwanathan, S., and Hoffman, T. (2005), "Kernel methods for missing variables," in *Artificial Intelligence and Statistics (AISTATS'05)*, (eds.) Z. Ghahramani and R. Cowell, Society for Artificial Intelligence and Statistics, pp. 325–332.

[98] Steinwart, I. and Christmann, A. (2008), *Support Vector Machines*, Spinger.

[99] Tibshirani, R. (1996), "Regression Shrinkage and Selection via the Lasso," *Journal of the Royal Statistical Society: Series B (Methodological)*, 58(1), 267–299.

[100] Tsuda, K., Akaho, S., and Asai, K. (2003), "The em Algorithm for Kernel Matrix Completion with Auxiliary Data," *Journal of Machine Learning Research*, 4, 67–81.

[101] Tucker, M. E. (2013), "Costs for Hepatitis C Treatment Skyrocket," .

[102] Valizadegan, H. and Jin, R. (2006), "Generalized maximum margin clustering and unsupervised kernel learning," in *Advances in Neural Information Processing Systems 19*, (eds.) B. Schölkopf, J. Platt, and T. Hofmann, Cambridge, MA: MIT Press, pp. 1417–1424.

[103] Valizadegan, H., Jin, R., and Jain, A. K. (2008), "Semi-Supervised Boosting for Multi-Class," in *Machine Learning and Knowledge Discovery in Databases*, (eds.) W. Daelemans, B. Goethals, and K. Morik, Springer Berlin Heidelberg, pp. 522–537.

[104] van Burren, S. and Oudshoorn, K. (1999), "Flexible multivariate imputation by MICE," Tech. rep., TNO Prevention and Health, Leiden, Netherlands.

[105] Vansteelandt, S., Carpenter, J., and Kenward, M. G. (2010), "Analysis of Incomplete Data Using Inverse Probability Weighting and Doubly Robust Estimators," *Methodology*, 6(1), 37–48.

[106] Vapnik, V. N. (1998), *Statistical Learning Theory*, New York, NY: John Wiley & Sons.

[107] Weston, J. and Watkins, C. (1999), "Support Vector Machines for Multi-Class Pattern Recognition," in *Proceedings of the 7th European Symposium on Artificial Neural Networks (ESANN-99)*, pp. 219–224.

[108] Williams, D., Liao, X., Xue, Y., Carin, L., and Krishnapuram, B. (2007), "On classification with incomplete data." *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 29(3), 427–436.

[109] Wu, J. C. F. (1983), "On the Convergence Properties of the EM Algorithm," *Annals of Statistics*, 11(1), 95–103.

[110] Wu, T.-F., Lin, C.-J., and Weng, R. C. (2004), "Probability Estimates for Multi-class Classification by Pairwise Coupling," *Journal of Machine Learning Research*, 5, 975–1005.

[111] Xu, L., Neufeld, J., Larson, B., and Schuurmans, D. (2005), "Maximum Margin Clustering," in *Advances in Neural Information Processing Systems 17*, (eds.) L. Saul, Y. Weiss, and L. Bottou, MIT Press, pp. 1537–1544.

[112] Xu, L. and Schuurmans, D. (2005), "Unsupervised and Semi-supervised Multi-class Support Vector Machines," in *Proceedings of the National Conference on Artificial Intelligence*, pp. 904–910.

[113] Yajima, Y. and Kuo, T.-F. (2006), "Optimization Approaches for Semi-Supervised Multiclass Classification," in *ICDMW '06: Proceedings of the Sixth IEEE International Conference on Data Mining - Workshops*, Washington, DC: IEEE, pp. 863–867.

[114] Yang, X., Song, Q., and Wang, Y. (2007), "A Weighted Support Vector Machine for Data Classification," *International Journal of Pattern Recognition and Artificial Intelligence*, 21(05), 961–976.

[115] Yee, T. W. (2010), "The VGAM Package for Categorical Data Analysis," *Journal of Statistical Software*, 32(10).

[116] Zhang, K., Tsang, I. W., and Kwok, J. T. (2007), "Maximum margin clustering made practical," in *ICML '07: Proceedings of the 24th International Conference on Machine Learning*, (ed.) Z. Ghahramani, New York, NY: ACM, pp. 1119–1126.

[117] Zhao, B., Wang, F., and Zhang, C. (2008), "CutS3VM : A Fast Semi-Supervised SVM Algorithm," in *Proceedings of the 14th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, Las Vegas, NV: ACM, pp. 830–838.

[118] Zhu, X. and Goldberg, A. B. (2009), "Introduction to Semi-Supervised Learning," in *Synthesis Lectures on Artificial Intelligence and Machine Learning*, (eds.) R. J. Brachman and T. Dietterich, Morgan & Claypool, pp. 1–130.

[119] Zou, H. and Hastie, T. (2005), "Regularization and variable selection via the elastic net," *Journal of the Royal Statistical Society: Series B (Methodological)*, 67(2), 301–320.

[120] Zubiaga, A., Fresno, V., and Martínez, R. (2009), "Is Unlabeled Data Suitable for Multiclass SVM-based Web Page Classification?" in *Proceedings of the NAACL HLT 2009 Workshop on Semi-Supervised Learning for Natural Language Processing*, Association for Computational Linguistics, pp. 28–36.